Proceedings of
Umeå's 19th Student Conference in Computing Science

USCCS 2015.3

S. Bensch, T. Hellström (editors)

UMINF 15.03
ISSN-0348-0542

Department of Computing Science
Umeå University

Preface

The Umeå Student Conference in Computing Science (USCCS) is organized annually as part of a course given by the Computing Science department at Umeå University. The objective of the course is to give the students a practical introduction to independent research, scientific writing, and oral presentation.

A student who participates in the course first selects a topic and a research question that he or she is interested in. If the topic is accepted, the student outlines a paper and composes an annotated bibliography to give a survey of the research topic. The main work consists of conducting the actual research that answers the question asked, and convincingly and clearly reporting the results in a scientific paper. Another major part of the course is multiple internal peer review meetings in which groups of students read each others' papers and give feedback to the author. This process gives valuable training in both giving and receiving criticism in a constructive manner. Altogether, the students learn to formulate and develop their own ideas in a scientific manner, in a process involving internal peer reviewing of each other's work, and incremental development and refinement of a scientific paper.

Each scientific paper is submitted to USCCS through an on-line submission system, and receives two or more reviews written by members of the Computing Science department. Based on the reviews, the editors of the conference proceedings (the teachers of the course) issue a decision of preliminary acceptance of the paper to each author. If, after final revision, a paper is accepted, the student is given the opportunity to present the work at the conference. The review process and the conference format aims at mimicking realistic settings for publishing and participation at scientific conferences.

USCCS is the highlight of the course, and this year the conference received thirteen submissions (out of a possible sixteen), which were carefully reviewed by the reviewers listed on the following page. We are very grateful to the reviewers who did an excellent job despite the very tight time frame. As a result of the reviewing process, twelve submissions were accepted for presentation at the conference. We would like to thank and congratulate all authors for their hard work and excellent final results that are presented during the conference.

We wish all participants of USCCS interesting exchange of ideas and stimulating discussions during the conference.

Umeå, 7 January 2015

Suna Bensch
Thomas Hellström

Organizing Committee

Suna Bensch
Thomas Hellström

Special thanks to the reviewers

Jaya Baskar
Suna Bensch
Henrik Björklund
Johanna Björklund
Thomas Hellström
Pedher Johansson
Thomas Johansson
Jakub Krzywda
Juan Carlos Nieves

Table of Contents

Do Graphical Passwords Provide Faster and Less Error Prone Authentication than Numerical Passwords on iPhones?

Mikaela Berg

Department of Computing Science
Umeå University, Sweden
id10mbg@cs.umu.se

Abstract. In this paper we compare two password systems, one numerical and one graphical. We investigate if graphical passwords are better than numerical passwords with respect to the duration of the input and the amount of user error. In a user study consisting of 30 participants we measured these parameters by allowing the participants to enter each password five times, making for a total of ten times. Results show that there is a significant difference between the systems with respect to the duration of the input and the difference is in favor of the graphical password system that has a lower average duration. Results also showed that there is no significant difference in the amount of user errors between the two password systems. Our conclusion is that graphical passwords are faster than numerical passwords if and only if it satisfies the following condition: both passwords consist of exactly four unique dots/digits.

1 Introduction

User authentication is a central part of everyday activities and allows humans to hide sensitive and personal information. It is important to provide the ability to hide this type of information behind authentication systems for security reason but also because personal information should remain private. Everyone should identify themselves in order to access personal information.

In this paper, we investigate if graphical passwords provide faster and less error prone authentication than numerical passwords on iPhones. The graphical password system is pattern-based verification and is constructed as a 3×3 grid of dots.

Previous work in the area has shown that people remember patterns better than numbers [1]. According to human psychology people are able to remember pictures more easily than strings of characters [2, 3]. Previous work has also shown that shape-based authentication better supports the way the human brain remembers and stores information [4]. It has also been shown that humans tend to remember PINs (Personal Identification Numbers) as patterns rather than a combination of numbers [1].

A previous study that compared tapping and dragging on horizontal and vertical surfaces showed that tapping was five percent faster on vertical surfaces

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 1–10, January 2015.

(such as a mobile device in portrait mode) and dragging was five percent faster on horizontal surfaces (such as a mobile device in landscape mode) [5]. In a similar study that compared different ways of interacting (with finger, stylus and mouse) on a touch screen showed that using fingers were fastest for tapping activities but slowest for dragging [6].

In order to investigate whether graphical passwords are faster and less error prone than numerical passwords a study has been conducted. With a test application developed for iPhone we compared the two password systems. Previous studies similar to ours have used other parameters, mostly focused on security, to estimate the result. In our study, we focus on the duration of the input and the amount of user error. We believe that if a system is more error prone it will make the system slower and that can be the determining variable that separates the two systems with respect to the duration of the input.

In the following section (Section 2) the proposed method will be described including information about the user study, the test application and data assumptions. In Section 3, the results are presented together with related calculations. Section 4 contains our conclusions and in Section 5 we discuss our results, limitations and future work.

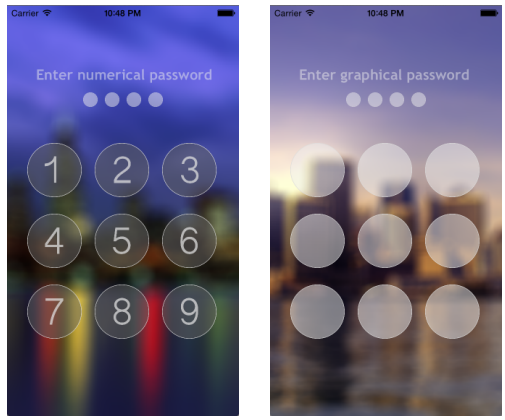
2 Method

2.1 Test Application

We developed an iPhone application specifically for this study (see Figure 1). The participants in our user study were provided one pair of passwords consisting of one numerical and one graphical password. In the test application the different password systems randomly appeared on the screen five times each and the participants were expected to enter the correct passwords. The application kept track of the duration per password system and the number of user errors per system. The systems were generated in a random order because we hoped it would reduce the possibility of creating a pattern in how the user solved the task. The following criteria applied to the parameters:

The duration of the input
: The duration was measured from when a password system was displayed on the screen, and not when the user started to type. We wanted to measure both planning and execution, not solely execution, because we believed planning was an essential part when entering a password. We expected that this approach would give the most reliable results for the two password systems. The duration ends when the correct password was entered successfully.

User error
: User errors appeared when the user entered the wrong password, and were then asked to enter the correct password. According to Moncur and Leplâtre [7] the most common type of error was entering the wrong digit. Other types of errors mentioned were double click and incorrect order but with correct digits. In our study, all types of user error were included without distinguishing them from one another.



(a) Numerical password system.

(b) Graphical password system.

Figure 1: Images from the test application that illustrate the two password systems.

We decided that all passwords should have the same length. Consequently each password consisted of exactly four dots/digits. Since we compared the duration of the input, we could not have a password that was twice as long as the other because it would affected the execution time. For the numerical password system, passwords containing a combination of four of the same number were excluded. Neither two or three consecutive alike numbers were accepted because we wanted the password systems to be as equal as possible with respect to difficulty.

2.2 User Study

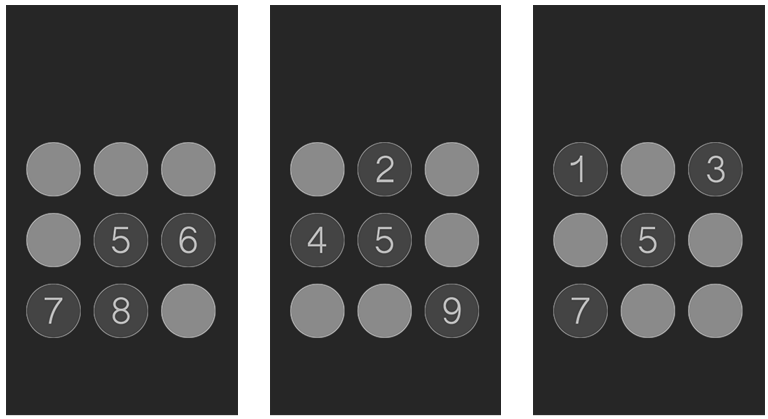
The test group in our user study consisted of thirty students in the age of 20 - 30 years who were accustomed to smartphones but not necessarily iPhones. All tests were performed on the same iPhone provided by us. Every test was also supervised by us in order to monitor that participants performed the tests correctly. Each participant performed ten tests, five with each system, which generated 150 tests for each system, giving a total of 300 tests.

We developed three sets of passwords and these three were alternated in order through our study. Figure 2 and Figure 3 show the six different passwords, three numerical and three graphical. These images were used in our user study to assign the participants one pair of passwords. Numerical passwords were visualized with both an image and a text with a combination of digits in order to visualize the correct order. For the graphical password system a star was used to display where the pattern began.

As mentioned above, we provided all passwords used in the user study, in order to give all participants the same conditions. Our intention was to give everyone passwords that they were unfamiliar with, because we wanted to avoid

the possibility that a password system obtained any advantage. An advantage that possibly could arise if participants used rehearsed passwords, therefore we gave them a password pair from Figure 2 and Figure 3.

We chose to create numerical passwords containing four different numbers. The reason why we decided to not repeat any numbers were to make visualization of the passwords easier. For the same reason our graphical passwords contained four different dots.



(a) Numerical password for pair 1.

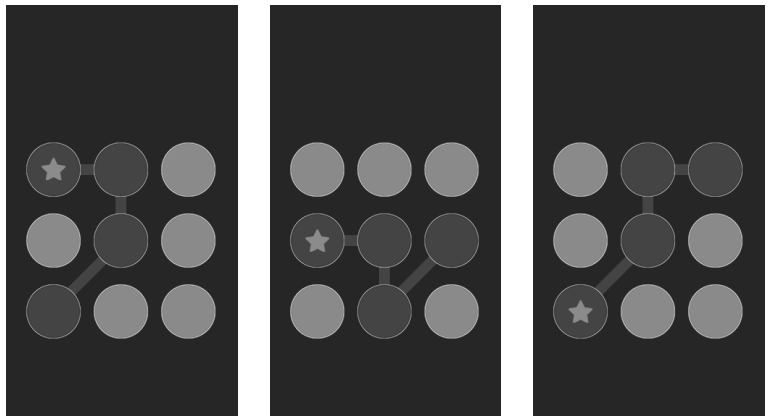
(b) Numerical password for pair 2.

(c) Numerical password for pair 3.

Figure 2: The three different combinations of numerical passwords that we used in our study. The numbers below the images correspond to the order the number must be entered.

When we created the passwords we tried to vary the pattern as much as possible but still maintain the same level of difficulty. Level of difficulty were based on how difficult it was to enter a password, for example, dots/digits were not arranged as a line but rather spread out on the grid to avoid the most common patterns such as line, square, diamond, perpendicular, etc. In order to maintain the same level of difficulty, all graphical passwords had the same strokes: one vertical, one horizontal, and one diagonal stroke, but not necessarily in that order. In the same manner, numerical passwords were not allowed to have several subsequent alike numbers.

At the beginning of the test participants were given a piece of paper containing one graphical and one numerical password. They were asked to memorize the passwords for as long as they needed. Participants were then told that the application randomly generated password systems and that their task was to fill out the correct password for each system. During the test, we measured the duration of the input and the amount of user errors.



(a) Graphical password for pair 1.

(b) Graphical password for pair 2.

(c) Graphical password for pair 3.

Figure 3: The three different combinations of graphical passwords that we used in our study. The star represented where the password began.

2.3 Data Assumptions and Methodology

According to the central limit theorem [8], a large number of independent random variables, possibly with different probability distributions but with finite variances, have a sum that converges to a normal distribution. We can assume that the samples are independent because participants performed the tests individually and therefore did not affected each other. The normal approximation in the central limit theorem will be good if n $>$ 30 regardless of the shape of population. Since we had 300 samples, samples can be assumed to be normally distributed according to the central limit theorem [8].

Given that our collected data is normally distributed and independent, we can perform an ANOVA (Analysis of Variance) [9] test. With an ANOVA test, we examine whether all means (μ) are equal which corresponds to the null hypothesis (h_0). We have chosen a 95 percent confidence interval. If the ANOVA test generates a significant result, we can reject the null hypothesis, which implies that we can be 95 percent confident that at least one of the means differs from the others in a way that is not random. We perform the following three ANOVA tests:

Sampled from the same distribution

We perform a test with the intention of investigating the probability that our three password pairs derive from the same distribution. In order to perform the next two ANOVA tests, this test must prove that all samples are sampled from the same distribution, otherwise we cannot continue. Our null hypothesis is $h_0 : \mu_{g_1} = \mu_{g_2} = \mu_{g_3} = \mu_{n_1} = \mu_{n_2} = \mu_{n_3}$; whether all means are equal ($g = graphical$ and $n = numerical$).

Difference in the duration of the input

If and only if the previous test was successful, we can perform a test with the

intention to investigate if there is a difference in the duration of the input between the two systems. We want our null hypothesis ($h_0 : \mu_{time_g} = \mu_{time_n}$) to be rejected. Which implies that at least one of the means differs from the others in a way that is not random. This result implicates that there is a difference between the password systems with respect to the duration of the input.

Difference in amount of user error

If and only if the first test was successful, we can perform a third test with the intention to investigate if there is a difference in the amount of user error between the two systems. We want our null hypothesis ($h_0 : \mu_{error_g} = \mu_{error_n}$) to be rejected. This result implicates that there is a difference between the password systems with respect to the amount of user error.

To perform the ANOVA tests, we used Minitab[1], a statistical software tool which also generated the figure linked to our result.

3 Result

As mentioned in Section 2.3, there are three possible test to perform. The first test is only intended to determine whether we can continue with the other two tests or not.

Sampled from the same distribution

We begin to examine whether all passwords are sampled from the same distribution. The ANOVA test generates a p-value equal to 0.221, which tells us that our samples are sampled from the same distribution with 22.1 percent probability. This means that we cannot reject the null hypothesis ($p \geq 0.05$), that all samples come from the same distribution. We can therefore continue with the other two tests.

Difference in the duration of the input

Because of the positive result in the previous test, we can perform another ANOVA test where we compare the duration of all the graphical passwords with the duration of all the numerical passwords. The test generates a p-value equal to 0.05. We obtain that there is a significant difference between the password system's with respect to the duration of the input, because $p \leq 0.05$. We can therefore confirm that the password systems are sampled from different distributions. By studying the password systems' mean times in Figure 4, we can deduce that the graphical password system has a lower mean time ($\bar{g} = 2,063$) than the numerical password system's mean time ($\bar{n} = 2,372$). The graphical system is faster than the numerical system.

Difference in amount of user error

Because of the positive result in the first test, we can also perform an ANOVA test on the amount of user errors for the various systems. Hence, we cannot

[1] `http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics/anova/how-to/one-way-anova/methods-and-formulas`

draw the same conclusions as the previous test. Results of the ANOVA test show that $p = 0.355$ concluding that there is no significant difference between the amount of user errors in the two systems, since $p \geq 0.05$. According to these results there is no conclusive evidence that one system is more sensitive to user error than the other.

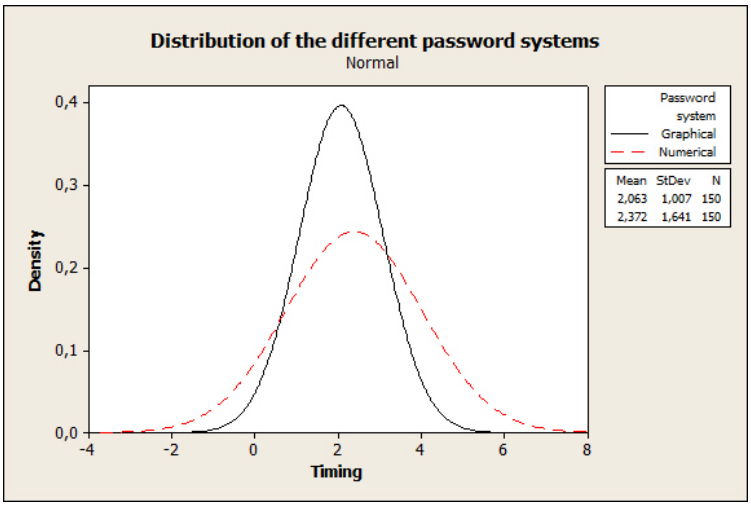


Figure 4: Normal distribution plot that illustrates the distribution of the two password systems with respect to the duration of the input.

4 Conclusion

Given that our first test showed a positive result, we were able to continue to the two tests that were intended for this study. With the results of these two tests, we can make the following conclusions:

Difference in the duration of the input between the two systems
The result of this test showed that there was a difference in the duration of the input between the two password systems. The mean time for the graphical system was lower than the mean time for the numerical system. Our conclusion is that graphical passwords are faster than numerical passwords if and only if it satisfies the following condition: both passwords must consist of exactly four unique dots/digits.

Difference amount of user error between the two systems
The result of this test showed that the differences in the amount of user errors were not sufficient enough. Hence we cannot say that there is a difference in the amount of user errors between the two password systems. Our hypothesis that the amount of user error affected the duration of the input cannot be verified.

5 Discussion

Since participants were assigned passwords at the beginning of the user study, we believe that the possibility of errors can increase due to memory loss. This is an aspect that might have affected our results. We believe that the two password systems have been affected with the same amount of memory loss, which means that we can ignore the impact on our results.

Another aspect that possibly had an impact on our results is people's previously established experience of numerical passwords. Numerical passwords (PINs) are widely used and established in today's society. Graphical passwords, however are not as frequently used as PINs and are most established among users whose primary phone runs the Android OS. Android uses graphical authentication as one of its standard verification system. We believe that the consequence of this approach could possibly be that the graphical password has a disadvantage compared to the numerical password, because participants are unfamiliar with the graphical approach.

We also kept in mind that there is a possibility that the password we provided the participants may resemble a password they previously used, however, this problem can be ignored due to the large sample size. Since the risk that our passwords are similar to a participant's password is very small, the impact on our final results will be negligible.

Another problem that can also be ignored is the possibility that participants were distracted during the test. This can be ignored because in a large amount of samples, outlying samples will not influence the final result.

5.1 Limitations

One of the major limitations in our user study is to provide the numerical password system the same conditions as the graphical password system. One part of the difficulty is how passwords are visualized for the participants. The graphical password is easy to visualize because lines can show the order and a suitable label can highlight the starting point. The numerical password needs an explanation of order besides highlighted digits.

A further difficulty is to find suitable lengths to compare. We chose to compare four digits in the numerical password to four dots in the graphical password. Another approach is to compare one digit against one stroke. Four digits versus four strokes correspond to five dots. This approach can possibly give a different result.

5.2 Future Work

In the future, similar user studies can be performed with focus on different parameters such as memory capacity or security, two interesting topics within user authentication. Security is an interesting area of user authentication on mobile devices because problems as shoulder surfing and smudges attacks exists [10, 11].

The results of our study only show that there is a difference in the duration of the input between the two password systems and not why there is a difference. Our results are only valid for exactly four unique dots/digits, which is a limited password space. An interesting future study would be to examine an extended password space.

6 Acknowledgments

The author would like to thank all the participants in the user study for their participation. We also thank the anonymous reviewers for their valuable comments and suggestions that improved the quality of the paper. Finally, we would also like to thank the peer reviewers whose valuable comments have been important to the paper's final structure.

References

[1] Weiss, R., Luca, A.D.: Passshapes - utilizing stroke based authentication to increase password memorability. In: Proceedings of the 5th Nordic Conference on Human-Computer Interaction. NordiCHI 2008, ACM (2008) 383–392

[2] Elftmann, P., Dr.-ing, P., Freiling, F., Bischof, P.C., D, P., inform Martin Mink, D.: Diploma thesis secure alternatives to password-based authentication mechanisms (2006)

[3] Khan, W.Z., Aalsalem, M.Y., Xiang, Y.: A graphical password based system for small mobile devices. CoRR (2011)

[4] De Luca, A., Hang, A., Brudy, F., Lindner, C., Hussmann, H.: Touch me once and i know it's you!: Implicit authentication based on touch screen patterns. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '12, New York, NY, USA, ACM (2012) 987–996

[5] Pedersen, E.W., Hornbæk, K.: An experimental comparison of touch interaction on vertical and horizontal surfaces. In: Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design. NordiCHI '12, New York, NY, USA, ACM (2012) 370–379

[6] Cockburn, A., Ahlström, D., Gutwin, C.: Understanding performance in touch selections: Tap, drag and radial pointing drag with finger, stylus and mouse. International Journal of Human-Computer Studies **70**(3) (2012) 218–233

[7] Moncur, W., Leplâtre, G.: Pictures at the atm: Exploring the usability of multiple graphical passwords. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '07, New York, NY, USA, ACM (2007) 887–894

[8] Alm, S., Britton, T.: Stokastik: sannolikhetsteori och statistikteori med tillämpningar. Liber (2008)

[9] Hassmén, P., Koivula, N.: Variansanalys. Studentlitteratur, Lund (1996)

[10] Wiedenbeck, S., Waters, J., Sobrado, L., Birget, J.C.: Design and evaluation of a shoulder-surfing resistant graphical password scheme. In: Proceedings of the Working Conference on Advanced Visual Interfaces. AVI '06, New York, NY, USA, ACM (2006) 177–184

[11] Aviv, A.J., Gibson, K., Mossop, E., Blaze, M., Smith, J.M.: Smudge attacks on smartphone touch screens. In: Proceedings of the 4th USENIX Conference on Offensive Technologies. WOOT'10, Berkeley, CA, USA, USENIX Association (2010) 1–7

The Stochastic Traveling Salesman Problem with Independent Discrete Random Variables

Johannes Blum

Department of Computing Science
Umeå University, Sweden
mcs13jbm@cs.umu.se

Abstract. We propose a variant of the Traveling Salesman Problem where the edge weights are independent discrete random variables. Then we present a partition algorithm which solves the problem by partitioning the state space into disjoint intervals. Finally we show how the computational effort can be reduced by making use of the problem structure.

1 Introduction

The Traveling Salesman Problem (TSP) is one of the most investigated combinatorial optimization problems of the last decades [1]. Given the distances between n cities, the goal is to find a shortest route which visits each city exactly once. This problem turned out to be quite hard in terms of computational complexity, but nevertheless it can be found in a multitude of applications varying from vehicle routing to genetics, scheduling and the design of integrated circuits. This might also be one of the reasons for the intensive research which is still going on within this area [1].

In some of the multiple real world applications the problem parameters may however not be deterministic as it is the case in the classical Traveling Salesman Problem. There are also cases where the parameters might differ depending on the situation, for example in vehicle routing where the travel times are highly dependent on the current traffic. This gave rise to several stochastic variants of the TSP. For instance, in [2] Kao presents a Traveling Salesman Problem where the distances between the cities are stochastic; in other versions the source of uncertainty is related to the subset of cities which actually have to be visited [3] or the exact location of each city [4].

In this paper we introduce a TSP variant called TSP_{ids} that is related to the variant investigated by Kao. Like in [2] the set of cities which have to be visited is fixed and the distances between the cities are stochastic, but in our case the distances can be modeled as independent discrete random variables. For this particular TSP variant we propose a partition algorithm.

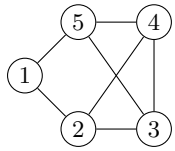
The remaining paper is structured as follows: Section 2 contains some preliminaries and a formal problem description. In Section 3 we give a general description of the partition algorithm whereas Section 4 explains how the feasibility of certain intervals can be computed in detail. Section 5 contains a short example before we conclude with a brief outlook in Section 6.

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 11–22, January 2015.

2 Problem Formulation

The usual way to describe the Traveling Salesman Problem formally is to model the given road network as a graph G having n vertices $\{1, \ldots, n\}$ where each edge $e_k = (i, j)$ has a weight c_k representing the distance between cities i and j. In the $\mathrm{TSP}_{\mathrm{ids}}$ each edge weight c_k is not a fixed value but rather an independent discrete random variable with a finite sample space. One application of this could be for example a road network where the edge weights vary depending on the traffic situation and the exact travel costs are not known ahead of time. We will denote the distribution function of c_k with F_k, i.e. $F_k(\omega) = P\{c_k \leq \omega\}$.

Figure 1 shows a simple example for a graph G and associated distributions for the edge weights.



Edge	Weight (Probability)
$e_1 = \{1, 2\}$	1(0.25) 2(0.40) 5(0.35)
$e_2 = \{1, 5\}$	2(0.75) 7(0.05) 8(0.20)
$e_3 = \{2, 3\}$	1(0.85) 6(0.10) 9(0.05)
$e_4 = \{2, 4\}$	5(0.15) 8(0.40) 9(0.45)
$e_5 = \{3, 4\}$	6(0.40) 7(0.35) 9(0.25)
$e_6 = \{3, 5\}$	3(0.10) 4(0.25) 5(0.65)
$e_7 = \{4, 5\}$	1(0.30) 2(0.40) 3(0.30)

Fig. 1. A simple undirected example graph with distributions for the edge weights c_k

For convenience we will identify the event $\{c_1 = x_1, \ldots, c_m = x_m\}$ also with the tuple $x = (x_1, \ldots, x_m)$ and call x a *state*. As the different x_k are independent the probability of state x is

$$P\{x\} = P\{c_1 = x_1, \ldots, c_m = x_m\} = \prod_{i=1}^{m} P\{c_i = x_i\}.$$

The set of all possible events is called *state space* and is denoted with Ω.

Each tour in G can be described as a sequence $t = (t_1, \ldots, t_n)$ of n edges and in state $x = (x_1, \ldots, x_m)$ the tour t has length

$$T_x(t) = \sum_{e_i \in t} x_i.$$

Given a graph G, a distribution function F of the edge weights and a fixed time limit k, the objective is now to compute the probability that there is a tour which can be completed within time bound k. In other words we want to compute the objective function

$$\mathrm{TSP}_{\mathrm{ids}}(G, F, k) = \sum_{\{x \mid \exists t \text{ s.t. } T_x(t) \leq k\}} P\{x\}.$$

3 A Partition Algorithm

3.1 Overview

In this section we present an algorithm for solving the $\mathrm{TSP}_{\mathrm{ids}}$. The naive approach would be to enumerate all possible states $x \in \Omega$ and to check for each state x individually if there is a tour t in the graph G such that $T_x(t) \leq k$. The problem of this method is the size of the state space. Just consider a simple graph containing 20 edges with 3 possible weights for each edge. Then we have already a state space containing $3^{20} \approx 3$ billion states. Obviously the size of the state space grows exponentially with the number of edges (in the nontrivial case that an edge has more than one possible weight) which makes even relatively small problems intractable for the naive algorithm.

A more efficient solution is the use of a so called *partition algorithm*, an approach introduced by Doulliez and Jamoulle [5] for maximum flows in networks with discrete random arc capacities. Other applications involve shortest paths [6] or minimum spanning trees [7] in stochastic graphs with discrete edge weights.

In order to be able to solve problems with a large state space Ω, we do not consider every single state individually, but we partition Ω into so called *intervals*, which are defined as follows:

If $\Omega \subseteq \mathbb{N}^n$ and $\alpha, \beta \in \Omega$, the interval $[\alpha, \beta] \subseteq \Omega$ is defined as

$$[\alpha, \beta] = \{x \in \Omega \mid \alpha_i \leq x_i \leq \beta_i \text{ for } i = 1, \ldots, n\}.$$

The probability $P\{[\alpha, \beta]\}$ of an interval $[\alpha, \beta]$ is defined as the sum of the probabilities $P\{x\}$ of all states $x \in [\alpha, \beta]$ and we have

$$P\{[\alpha, \beta]\} = \sum_{x \in [\alpha, \beta]} P\{x\} = \prod_{i=1}^{m} \sum_{x_i = \alpha_i}^{\beta_i} P\{c_i = x_i\}.$$

This follows directly from the definition of the intervals and the independence of the random variables x_i. Therefore the probability $P\{[\alpha, \beta]\}$ can be computed efficiently in only $2m$ operations given the (cumulative) probability distribution F_i as

$$\sum_{x_i = \alpha_i}^{\beta_i} P\{c_i = x_i\} = F_i(\beta_i) - F_i(\gamma_i) \text{ where } F_i(\gamma_i) = P\{c_i < \alpha_i\}.$$

In order to compute $\mathrm{TSP}_{\mathrm{ids}}$ we need to consider all states x for which there exists a tour t such that $T_x(t) \leq k$. A state for which this holds is called *feasible*, otherwise it is called *infeasible*.

If we extend this definition onto sets of states, we call $S \subseteq \Omega$ *feasible* if and only if all states $x \in S$ are feasible. If all states $x \in S$ are infeasible, S is called *infeasible*. Moreover the maximum feasible and infeasible subset of a set Φ is denoted with Φ_F and Φ_I, respectively.

Based on these definitions we have

$$\mathrm{TSP}_{\mathrm{ids}}(G, F, k) = P\{\Omega_F\}.$$

3.2 Algorithm Description

As stated before, the idea behind computing $\mathrm{TSP}_{\mathrm{ids}}(G, F, k) = P\{\Omega_F\}$ is to partition Ω_F into disjoint feasible intervals $F_1, \ldots, F_l$ and compute

$$\sum_{i=1}^{l} P\{F_i\} = P\{\Omega_F\} = \mathrm{TSP}_{\mathrm{ids}}(G, F, k).$$

Therefore we partition a given interval $U \subseteq \Omega$ into a feasible interval F, disjoint infeasible intervals $I_1, \ldots, I_m$ and disjoint so called *undetermined intervals* $U_1, \ldots, U_m$ which might be neither feasible nor infeasible and could be required to be partitioned again. Moreover in each step we compute upper and lower bounds $\mathcal{P}_u$ and $\mathcal{P}_l$ for $P\{\Omega_F\}$.

In the very first step we start with the interval $U = \Omega$ and initialize the bounds with $\mathcal{P}_u = 1$ and $\mathcal{P}_l = 0$. When the algorithm terminates we will have $\mathcal{P}_u = \mathcal{P}_l = P\{\Omega_F\}$.

Now let us have a look how the partitioning of an interval $U = [\alpha, \beta]$ is performed in detail. The trivial case is that U is feasible or infeasible. In such a situation no further partitioning is required and we can just increase $\mathcal{P}_l$ by $P\{U\}$ or decrease $\mathcal{P}_u$ by $P\{U\}$, respectively.

Now consider the case that U is neither feasible nor infeasible. Then we determine a so called feasible *cutoff state* $\bar{x}$ such that $F = [\alpha, \bar{x}]$ is feasible. After doing so we can increase $\mathcal{P}_l$ by $P\{F\}$. At first glance it might seem desirable to find an optimal cutoff state such that the increase of $\mathcal{P}_l$ is maximized, but in most cases it is actually better to work with a nonoptimal solution which can be computed fast, because finding an optimal cutoff state can be a quite hard problem [7]. One possibility to find a "good" cutoff state which provides a reasonable increase of the lower bound without too much computational effort is to use a greedy pushing strategy. This means we just repeatedly increase the single components of α until an infeasible state is reached. Then we go one step back and choose the last feasible state as the cutoff state.

When we have found such a cutoff state, the next step is to compute the so called *limiting feasible components* $\hat{x}_1, \ldots, \hat{x}_m$ which are defined as

$$\hat{x}_i = \max\{\gamma \mid \alpha_i \leq \gamma \leq \beta_i \text{ and } (\alpha_1, \ldots, \alpha_{i-1}, \gamma, \alpha_{i+1}, \ldots, \alpha_m) \text{ is feasible}\}.$$

Then the intervals

$$\tilde{I}_i = \{x \mid \hat{x}_i < x_i \leq \beta_i \text{ and } \alpha_j \leq x_j \leq \beta_j \text{ for } j \neq i\}$$

are infeasible as the lower limit of $\tilde{I}_i$ is infeasible by the definition of $\hat{x}_i$. The different $\tilde{I}_i$ are not disjoint, but we can partition their union into disjoint intervals

$$I_i = \tilde{I}_i \setminus (\tilde{I}_1 \cup \cdots \cup \tilde{I}_{i-1})$$

which is equivalent to

$$I_i = \{x \mid \alpha_j \leq x_j \leq \hat{x}_j \text{ for } j < i, \hat{x}_i < x_i \leq \beta_i, \alpha_j \leq x_j \leq \beta_j \text{ for } j > i\}.$$

Now we can also update the upper bound $\mathcal{P}_u$ by subtracting $\sum_{i=1}^{m} P\{I_i\}$ as the different I_i are disjoint. The remaining states which have not been considered yet are $U \setminus (F \cup I_1 \cup \cdots \cup I_m)$. These can be partitioned into the intervals

$$\tilde{U}_i = \{x \mid \bar{x}_i \leq x_i < \hat{x}, \alpha_j \leq x_j \leq \hat{x}_j \text{ for } j \neq i\}.$$

In general, these intervals are not disjoint either, but like in the previous case we can partition their union into disjoint intervals $U_1, \ldots, U_m$ where

$$U_i = \{\alpha_j \leq x_j \leq \bar{x}_j \text{ for } j < i, \bar{x}_i < x_i \leq \hat{x}_i, \alpha_j \leq x_j \leq \hat{x}_j \text{ for } j > i\}.$$

Now U is partitioned into a feasible interval F, infeasible intervals $I_1, \ldots, I_m$ and undetermined intervals $U_1, \ldots, U_m$ which are all disjoint. The undetermined intervals $U_1, \ldots, U_m$ can again be partitioned in the same manner. Once there are no undetermined intervals left, the algorithm terminates and we have $P\{x\} = \mathcal{P}_l = \mathcal{P}_u$. The entire algorithm is also summarized in Algorithm 1.

Algorithm 1: The partition algorithm

$\mathcal{U} = \{\Omega\}$
$\mathcal{P}_u = 1$
$\mathcal{P}_l = 0$
while $\mathcal{U} \neq \emptyset$ **do**
  Remove an interval $U = [\alpha, \beta]$ from $\mathcal{U}$
  if *U is feasible* **then**
    $\mathcal{P}_l = \mathcal{P}_l + P\{U\}$
  else if *U is infeasible* **then**
    $\mathcal{P}_u = \mathcal{P}_u - P\{U\}$
  else
    Find a feasible cutoff state $\bar{x}$ such that $F = [\alpha, \bar{x}]$ is feasible
    $\mathcal{P}_l = \mathcal{P}_l + P\{F\}$
    for $i = 1$ *to* m **do**
      $\hat{x}_i = \max\{\gamma \mid \alpha_i \leq \gamma \leq \beta_i$ and $(\alpha_1, \ldots, \alpha_{i-1}, \gamma, \alpha_{i+1}, \ldots, \alpha_m)$ is feasible$\}$
    end
    for $i = 1$ *to* m **do**
      $I_i = \{x \mid \alpha_j \leq x_j \leq \hat{x}_j$ for $j < i, \hat{x}_i < x_i \leq \beta_i, \alpha_j \leq x_j \leq \beta_j$ for $j > i\}$
    end
    $\mathcal{P}_u = \mathcal{P}_u - \sum_{i=1}^{m} P\{F_i\}$
    for $i = 1$ *to* m **do**
      $U_i = \{\alpha_j \leq x_j \leq \bar{x}_j$ for $j < i, \bar{x}_i < x_i \leq \hat{x}_i, \alpha_j \leq x_j \leq \hat{x}_j$ for $j > i\}$
    end
    $\mathcal{U} = \mathcal{U} \cup \{U_1\} \cup \cdots \cup \{U_m\}$
  end
end

4 Computation of Feasibility

In the previous section we gave a general description of the partition algorithm, but one important detail is still missing, namely how we determine if a given interval $S = [\alpha, \beta]$ is feasible or infeasible. The naive approach would be to enumerate all states $x \in S$ and check their feasibility individually, but this would bring us back to the problem of the state space explosion. However it is possible to determine if $S = [\alpha, \beta]$ is feasible or infeasible by just considering the limiting states α and β as the following lemma shows.

Lemma 1. *1. An interval $[\alpha, \beta]$ is feasible if and only if β is feasible.*
2. An interval $[\alpha, \beta]$ is infeasible if and only if α is infeasible.

Proof. 1. The direction from left to right is trivial. For the other direction consider an interval $[\alpha, \beta]$ such that β is feasible. Then there is a tour t in G such that $F_\beta(t) \leq k$. This means that for all $x \leq \beta$ (componentwise) we have $F_x(t) \leq F_\beta(t)$ as in state x the edge weights are less or equal than in state β which means that the length of t can not increase. Therefore all $x \in [\alpha, \beta]$ are feasible and consequently $[\alpha, \beta]$ is feasible.
2. Analogous.

In order to determine if a single state x is feasible or not, one has to determine if there exists a tour t in G such that $T_x(t) \leq k$. This means one has to solve the standard Traveling Salesman Problem with the edge weights indicated by x, a problem which is known to be NP-complete [8, 9]. This implies there is probably no polynomial time algorithm that decides if x is feasible or not. However there are a few ways to speed up the computation of the feasibility of x.

At first one can use a heuristic in order to find a tour t with $T_x(t) \leq k$ and if the heuristic succeeds, x is feasible. However, if the heuristic fails we can in general not draw any conclusion about the feasibility of x. In such case we could indeed classify x as infeasible but this might be incorrect and the correctness of the solution might suffer from that.

Moreover, the computation of the cutoff state and the feasibility components involve repeated processing of states differing only in some components. This fact can be used in order to reduce the computational effort. Therefore consider a state x with a tour t such that $T_x(t) \leq k$. If we analyze the feasibility of a state x' obtained by increasing the i-th component of x to the value x'_i, there are two possible cases:

- If the edge e_i is not part of the tour t, the length of t is not affected by the change of x_i. Therefore we have $T'_x(t) = T_x(t)$ and x' is feasible.
- If the edge e_i is part of the tour t, the length of t increases by the value $\Delta = x'_i - x_i$. Therefore, if $T_x(t) + \Delta \leq k$ we have $T_{x'}(t) = T_x(t) + \Delta \leq k$ and x' is feasible.
  If $T_x(t) + \Delta > k$ we have $T_{x'}(t) > k$, but there might be another tour t' in G such that $T_{x'}(t') \leq k$, so we can not draw any conclusion about the feasibility of x'.

Based on these observations we can compute a cutoff state $\bar{x}$ for the interval $U = [\alpha, \beta]$ as follows. We start with $\bar{x} = \alpha$, compute a tour t for the configuration α with length less or equal k (either with an exact algorithm or a heuristic) and set $\bar{x}_i = \beta_i$ for all edges $e_i \notin t$. Then we increase the remaining components of $\bar{x}$ repeatedly such that we stay within the bound $T_{\bar{x}}(t) \leq k$. The algorithm is also summarized in Algorithm 2.

Algorithm 2: Cutoff

$\bar{x} = \alpha$
Find a tour t such that $T_\alpha(t) \leq k$
foreach $e_i \notin t$ **do**
  $\bar{x}_i = \beta_i$
end
foreach $e_i \in t$ **do**
  $\Delta = k - T_{\bar{x}}(t)$
  $\bar{x}_i = \max\{\gamma \in \{\alpha_i, \ldots, \beta_i\} \mid (\gamma - \alpha_i) \leq \Delta\}$
end

In the same manner we can also compute the limiting feasible components $\hat{x}_i$. We start again with computing a tour t such that $T_\alpha(t) \leq k$. For each edge e_i that is not part of t we can directly set $\hat{x}_i = \beta_i$. For the remaining e_i we start with the configuration $\gamma = \alpha$ and as previously we increase the component γ_i until we hit the bound $T_\gamma(t) \leq k$. Then we increase γ_i once more and try to find a tour t' such that $T_\gamma(t') \leq k$. If this fails, the last γ_i which was within the bound is the limiting feasible component, otherwise we have to repeat the previous procedure with the new γ. The entire algorithm can also be seen in Algorithm 3.

Algorithm 3: Limiting feasible components

Find a tour t such that $T_\alpha(t) \leq k$
foreach $e_i \notin t$ **do**
  $\hat{x}_i = \beta_i$
end
foreach $e_i \in t$ **do**
  $\gamma = \alpha$
$\star$ $\Delta = k - T_\gamma(t)$
  $\gamma_i = \min\{\zeta \in \{\alpha_i, \ldots, \beta_i\} \mid (\zeta - \gamma_i) > \Delta\}$
  if *there is a tour* t' *such that* $T_\gamma(t') \leq k$ **then**
    goto $\star$ with t' as t
  end
  $\hat{x}_i = \max\{\zeta \in \{\alpha_i, \ldots, \beta_i\} \mid (\zeta - \gamma_i) \leq \Delta\}$
end

5 An Example

In this section we give a short example of how the presented algorithm solves the example problem from Figure 1 in Section 2 for $k = 15$.

At first, the upper and lower bounds are initialized and the algorithm considers the interval

$$\Omega = [\alpha, \beta] = [(1, 2, 1, 5, 6, 3, 1), (5, 8, 9, 9, 9, 5, 3)] .$$

The configuration α is feasible, because for the tour $t = (e_1, e_3, e_5, e_7, e_2)$ we have $T_\alpha(t) = 11$, but β is infeasible as the shortest tour in configuration β has length 32. Therefore we have to partition the interval Ω.

The next step to compute a cutoff state $\bar{x}$. As we already saw, we have $T_\alpha(t) \leq k$ for the tour t, so we can increase the weights of the edges e_4 and e_6 such that we get $\bar{x} = (1, 2, 1, 9, 6, 5, 1)$. Then we can increase the length of t still by $\Delta = k - T_{\bar{x}}(t) = 4$, so we can increase $\bar{x}_1$ by 4 and get the cutoff state

$$\bar{x} = (5, 2, 1, 9, 6, 5, 1).$$

The feasible interval $F = [\alpha, \bar{x}]$ has probability $P\{F\} = 0.0765$, so we update $\mathcal{P}_l$ to $\mathcal{P}_l = 0.0765$.

No we have to compute the limiting feasible components. From the previous step we know already that $\hat{x}_4 = 9$ and $\hat{x}_6 = 5$. Moreover we could push the weight of edge e_1 until the boundary without becoming infeasible, so we have $\hat{x}_1 = 5$. But if we increase the weight of edge e_2, the shortest tour has length 16 which means the second limiting feasible component is $\hat{x}_2 = 2$. In the same manner we compute the remaining components and get

$$\hat{x} = (5, 2, 1, 9, 9, 5, 3).$$

This leads to the infeasible intervals

$$I_2 = [(1, 7, 1, 5, 6, 3, 1), (5, 8, 9, 9, 9, 5, 3)]$$

and

$$I_3 = [(1, 2, 6, 5, 6, 3, 1), (5, 2, 9, 9, 9, 5, 3)] .$$

The remaining intervals I_1, I_4, I_5, I_6 and I_7 are actually empty because the corresponding limiting feasible components already touched the boundary and there is no x_i such that $\hat{x}_i < x_i \leq \beta_i$.

We have $P\{I_2\} = 0.25$ and $P\{I_3\} = 0.11$, so we can update $\mathcal{P}_u$ to $\mathcal{P}_u = 0.64$.

The generated undetermined intervals are

$$\begin{aligned} U_2 &= [(1, 7, 1, 5, 6, 3, 1), (5, 2, 1, 9, 9, 5, 3)] \\ U_3 &= [(1, 2, 6, 5, 6, 3, 1), (5, 2, 1, 9, 9, 5, 3)] \\ U_5 &= [(1, 2, 1, 5, 7, 3, 1), (5, 2, 1, 9, 9, 5, 3)] \\ U_7 &= [(1, 2, 1, 5, 6, 3, 2), (5, 2, 1, 9, 6, 5, 3)] \end{aligned}$$

As for the infeasible intervals, there are also some empty undetermined intervals, namely U_1, U_4 and U_6. The nonempty undetermined intervals are however filed into the list $\mathcal{U}$ and the computation loop starts over.

At this point we will interrupt the example and state only that the algorithm terminates after 10 further iterations with a probability of approximately 0.38. An overview over the entire computation can be found in the appendix.

6 Summary and Future Work

In this paper we introduced the TSP_{ids} problem and presented a partition algorithm for it, including a description how the feasibility of intervals can be computed.

There are several aspects that remain to be discussed in future work. One is the computational complexity of the problem. Moreover one could analyze the proposed partition algorithm and investigate the number of (undetermined) intervals that are generated in every step. This could enable one to estimate the runtime of the partition algorithm and moreover give some indication how the choice of the cutoff state could be modified in order to increase the performance of the algorithm.

The used technique of state space partitioning has already been applied to several problems like shortest paths or minimus spanning trees in stochastic graphs and could also be useful for solving discrete stochastic variants of other graph problems such as graph coloring or covering problems.

References

[1] Chvatal, V., Applegate, D.L., Cook, W.J., Bixby, R.E.: The Traveling Salesman Problem: A Computational Study. Princeton University Press (2011)

[2] Kao, E.P.C.: A preference order dynamic program for a stochastic traveling salesman problem. Operations Research **26**(6) (1978) 1033–1045

[3] Jaillet, P.: A priori solution of a traveling salesman problem in which a random subset of the customers are visited. Operations Research **36**(6) (1988) pp. 929–936

[4] Goemans, M.X., Bertsimas, D.J.: Probabilistic analysis of the held and karp lower bound for the euclidean traveling salesman problem. Mathematics of Operations Research **16**(1) (1991) 72–89

[5] Doulliez, P., Jamoulle, E.: Transportation networks with random arc capacities. RAIRO-Operations Research-Recherche Opérationnelle **6**(V3) (1972) 45–59

[6] Alexopoulos, C.: State space partitioning methods for stochastic shortest path problems. Networks **30**(1) (1997) 9–21

[7] Alexopoulos, C., Jacobson, J.A.: State space partition algorithms for stochastic systems with applications to minimum spanning trees. Networks **35**(2) (2000) 118–138

[8] Karp, R.M.: Reducibility among combinatorial problems. In Miller, R.E., Thatcher, J.W., Bohlinger, J.D., eds.: Complexity of Computer Computations. The IBM Research Symposia Series. Springer US (1972) 85–103
[9] Garey, M.R., Johnson, D.S.: Computers and Intractability : A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)

Appendix: Protocol of the Example Computation

Iteration 1

- $U = [(1,2,1,5,6,3,1),(5,8,9,9,9,5,3)]$ needs to be partitioned
- $\hat{x} = (5,2,1,9,6,5,1)$
- $\mathcal{P}_l = 0.08$
- $F = [(1,2,1,5,6,3,1),(5,2,1,9,6,5,1)]$
- $\hat{x} = (5,2,1,9,9,5,3)$
- Infeasible intervals
  - $I_2 = [(1,7,1,5,6,3,1),(5,8,9,9,9,5,3)]$
  - $I_3 = [(1,2,6,5,6,3,1),(5,2,9,9,9,5,3)]$
- $\mathcal{P}_u = 0.64$
- Undetermined intervals
  - $U_2 = [(1,7,1,5,6,3,1),(5,2,1,9,9,5,3)]$
  - $U_3 = [(1,2,6,5,6,3,1),(5,2,1,9,9,5,3)]$
  - $U_5 = [(1,2,1,5,7,3,1),(5,2,1,9,9,5,3)]$
  - $U_7 = [(1,2,1,5,6,3,2),(5,2,1,9,6,5,3)]$

Iteration 2

- $U = [(1,2,1,5,6,3,2),(5,2,1,9,6,5,3)]$ needs to be partitioned
- $\hat{x} = (2,2,1,9,6,5,3)$
- $\mathcal{P}_l = 0.19$
- $F = [(1,2,1,5,6,3,2),(2,2,1,9,6,5,3)]$
- $\hat{x} = (2,2,1,9,6,5,3)$
- Infeasible intervals
  - $I_1 = [(5,2,1,5,6,3,2),(5,2,1,9,6,5,3)]$
- $\mathcal{P}_u = 0.58$
- Undetermined intervals
  - $U_1 = [(5,2,1,5,6,3,2),(2,2,1,9,6,5,3)]$

Iteration 3

- $U = [(5,2,1,5,6,3,2),(2,2,1,9,6,5,3)]$ is feasible
- $\mathcal{P}_l = 0.19$

Iteration 4

- $U = [(1,2,1,5,7,3,1),(5,2,1,9,9,5,3)]$ needs to be partitioned
- $\hat{x} = (2,2,1,9,9,5,1)$
- $\mathcal{P}_l = 0.27$

- $F = [(1, 2, 1, 5, 7, 3, 1), (2, 2, 1, 9, 9, 5, 1)]$
- $\hat{x} = (2, 2, 1, 9, 9, 5, 3)$
- Infeasible intervals
  - $I_1 = [(5, 2, 1, 5, 7, 3, 1), (5, 2, 1, 9, 9, 5, 3)]$
- $\mathcal{P}_u = 0.44$
- Undetermined intervals
  - $U_1 = [(5, 2, 1, 5, 7, 3, 1), (2, 2, 1, 9, 9, 5, 3)]$
  - $U_7 = [(1, 2, 1, 5, 7, 3, 2), (2, 2, 1, 9, 9, 5, 3)]$

Iteration 5

- $U = [(1, 2, 1, 5, 7, 3, 2), (2, 2, 1, 9, 9, 5, 3)]$ needs to be partitioned
- $\hat{x} = (2, 2, 1, 9, 7, 5, 3)$
- $\mathcal{P}_l = 0.37$
- $F = [(1, 2, 1, 5, 7, 3, 2), (2, 2, 1, 9, 7, 5, 3)]$
- $\hat{x} = (2, 2, 1, 9, 9, 5, 3)$
- No infeasible intervals
- $\mathcal{P}_u = 0.44$
- Undetermined intervals
  - $U_5 = [(1, 2, 1, 5, 9, 3, 2), (2, 2, 1, 9, 9, 5, 3)]$

Iteration 6

- $U = [(1, 2, 1, 5, 9, 3, 2), (2, 2, 1, 9, 9, 5, 3)]$ needs to be partitioned
- $\hat{x} = (1, 2, 1, 9, 9, 5, 2)$
- $\mathcal{P}_l = 0.38$
- $F = [(1, 2, 1, 5, 9, 3, 2), (1, 2, 1, 9, 9, 5, 2)]$
- $\hat{x} = (1, 2, 1, 9, 9, 5, 2)$
- Infeasible intervals
  - $I_1 = [(2, 2, 1, 5, 9, 3, 2), (2, 2, 1, 9, 9, 5, 3)]$
  - $I_7 = [(1, 2, 1, 5, 9, 3, 3), (1, 2, 1, 9, 9, 5, 3)]$
- $\mathcal{P}_u = 0.38$
- Undetermined intervals
  - $U_1 = [(2, 2, 1, 5, 9, 3, 2), (1, 2, 1, 9, 9, 5, 2)]$
  - $U_7 = [(1, 2, 1, 5, 9, 3, 3), (1, 2, 1, 9, 9, 5, 2)]$

Iteration 7

- $U = [(1, 2, 1, 5, 9, 3, 3), (1, 2, 1, 9, 9, 5, 2)]$ is feasible
- $\mathcal{P}_l = 0.38$

Iteration 8

- $U = [(2, 2, 1, 5, 9, 3, 2), (1, 2, 1, 9, 9, 5, 2)]$ is feasible
- $\mathcal{P}_l = 0.38$

Iteration 9

- $U = [(5, 2, 1, 5, 7, 3, 1), (2, 2, 1, 9, 9, 5, 3)]$ is infeasible
- $\mathcal{P}_u = 0.38$

Iteration 10

- $U = [(1, 2, 6, 5, 6, 3, 1), (5, 2, 1, 9, 9, 5, 3)]$ is infeasible
- $\mathcal{P}_u = 0.38$

Iteration 11

- $U = [(1, 7, 1, 5, 6, 3, 1), (5, 2, 1, 9, 9, 5, 3)]$ is infeasible
- $\mathcal{P}_u = 0.38$

Web Advertisement Awareness: From a Gender Point of View

Linnea Forsberg

Department of Computing Science
Umeå University, Sweden
lifo0037@student.umu.se

Abstract. Banner blindness is a phenomenon when Internet users are ignoring advertisements that appear on web pages. Some researchers claim that the more an advertisement calls for attention, the more it repels the gaze from the users. Some claim that perception can differ between men and women regarding advertising. In this article a study is conducted that compares the difference between men and women with respect to awareness of advertising. The result shows that women tend to ignore banners more than men do.

1 Introduction

It all started in 1994, when the first web advertisement in the history of the Internet was made. It was an advertisement for the American telecommunication company AT&T. The banner was placed on the site Hotwired, which was one of the first commercial magazines on the web [1, 2, 3]. Today the World Wide Web has come to be a platform, that most people in society are frequently using in their daily life. For example people in Sweden with jobs are using the web about 26 hours a week, and students 31 hours a week [4]. The Internet has also evolved to a huge marketing place for companies and corporations. Therefore, every time we open our web browsers we are automatically served with different kinds of commercials and advertisements. Companies are directing their campaigns to different groups of society like toy commercials for children, shaving products for men and make up for women.

In 2004 there was an article published [5] explaining that there is a difference between gender regarding how advertisement are being perceived. It describes both biological and social factors that influence how men's and women's perception works. The article classifies men as so called heuristic processors while women are portrayed as comprehensive processors. Therefore advertising aimed for heuristic processors (men) should be kept more simple and only have the key functions described. Advertising for comprehensive processors (women) on the other hand, should be verbally and visually rich and highly informative, because women are more likely to welcome that kind of approach [5].

Regarding perception, it is not only important to talk about ***what*** we see, but also what we choose not to see. This is because there is a behaviour that

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 23–30, January 2015.

has come to researchers' attention that is called "banner blindness" or "banner avoidance". This is a phenomena when a person browsing the web is more or less ignoring banners and advertisements [3]. There are a few explanations for banner blindness. One of them is that the Internet is viewed more as a task-performance medium rather than an entertainment medium, which can make users avoid advertisements especially if they have to perform a task within limited time [3]. Banner blindness can occur with protruding items that do not look like the other content on a website. That means that advertisements that are calling for too much attention might have an effect that is repelling rather than attracting [6].

There is a study [7] about banner avoidance that mentions a difference between men and women, where one of the conclusions is that men tend to ignore banners more than women do.

In this study on the other hand, the test method is different to [7], because the measured data used here, is how many advertising banners the test participants were aware of, when browsing a website. In [7], the only measured data was time, since the test was designed so that the test persons were to solve a given task, if the test persons noticed and clicked the banners they were able to finish the task faster than those who did not see them.

This article investigates the difference between men and women regarding awareness of advertising on websites. If the study shows that there is a difference, it might mean that web advertisement is more efficient towards one gender than the other. This will influence when marketing with banners should be used in order to attract the right target group. In Section 2 the approach of the study is described. All different advertisement banners used in the test are also explained. The test persons in this study were first asked to use a website that contained several advertisements. In order to make the test persons look at the website in a way that were as normal as possible, they were given a task that had nothing to do with the advertisements. When the test persons have finished the task, they were shown one by one, all the different advertisements that were visible on the website and asked if they had noticed them. Section 3 shows all the data from the outcome of the tests, and Section 4 describes conclusions, limitations and reflections of the study.

2 Method

In order to make the test group as homogeneous as possible, a group of ten students at Umeå University, Sweden studying Interaction Design was chosen. Five of them are men and five of them are women, and their ages vary between 20-29 years.

Since the test was supposed to interpret if there was a gender difference, the chosen advertisement material should attract both male and female students at the Interaction and Design program equally much. Therefore, the following eight kinds of advertisements were chosen, since they were believed to be as gender neutral as possible.

– Advertisement 1. Book advertisement

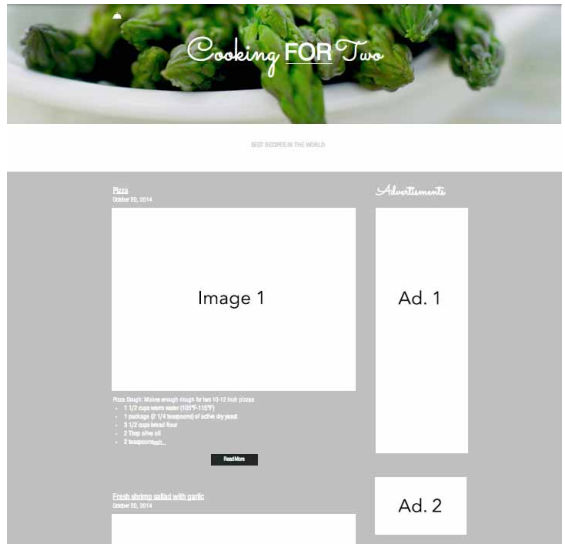


Fig. 1. Example print screen of the recipe website developed for this study. The advertisements are displayed on the right hand side and the recipes in the middle of the web page. Due to copyright restrictions some advertisements and images are not included in the figure.

- Advertisement 2. Cheap coffee offering
- Advertisement 3. Interactive glasses
- Advertisement 4. Cheap bike deal
- Advertisement 5. Umea University
- Advertisement 6. Studera.nu advertisement
- Advertisement 7. Hamburger restaurant commercial
- Advertisement 8. Smart watch commercial

The website that was used in the test was created for the study. It was a cooking website containing different kinds of dinner recipes. Figure 1 shows a print screen of the website. In the middle of the page all recipes are listed. Every recipe has a picture of the dish and when the ***read more*** button is pressed a detailed explanation of the recipe is presented. All advertisements were still pictures (rather than animations) and placed on the right hand side of the web page.

Each test person was placed at a laptop computer and told that this was a scenario where they were supposed to prepare dinner for a friend. In order to find something tasty to cook, they were to go to a specific recipe website and choose the recipe that they found most appealing. The test persons were also told not to rush, but to carefully read through the different recipes. Informing them of this, made the test persons look at all recipes on the website, which made them exposed to all advertisements.

After deciding on what recipe on the website was the best, the test person was given a questionnaire which is shown in Figure 2.

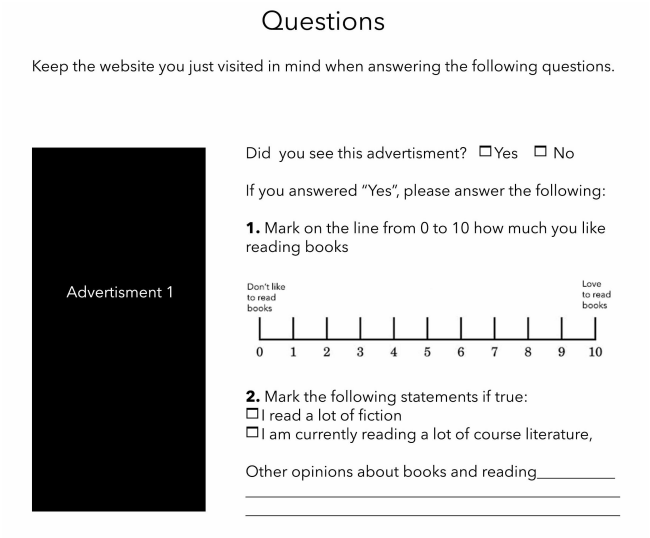


Fig. 2. The first page of the questionnaire

The questionnaire contained eight pages, where each of the advertisements in the test had an own page. On each advertisement the test person was supposed to answer "yes" or "no" to the question "Did you see this advertisement on the website?" If the answer was "no" the test person continued to the next page. If the test person answered "yes", they were asked to fill out some extra information regarding the advertisement. With this extra information we wanted to gather additional data, which we used when analysing the test. In the following we describe each page in the questionnaire.

For Advertisement 1 (book advertisement), the test persons who noticed the advertisement, were supposed to mark on a scale from 0 to 10 how much they like reading, where 0 meant "don't like to read books" and 10 meant "love to read books". It was also possible to check two check boxes with the statements "I read a lot of fiction" and "I am currently reading a lot of course literature", if they held for the test person.

For Advertisement 2 (cheap coffee offering), the test persons who noticed the advertisement were supposed to mark on a scale from 0 to 10 how much they like to drink coffee, where 0 meant they don't like coffee and 10 meant they loved it. The statements they could mark as true were "I am a coffee drinker", "I am not a coffee drinker", "I like things that are for free", "People in my surrounding are often drinking coffee".

If the test persons noticed Advertisement 3 (interactive glasses), they were supposed to mark on a scale from 0 to 10 how interested they are in interactive glasses, where 0 meant that they were not interested at all, and 10 meant they were very interested. The additional statements were "I am very interested in technology" "I have tried a pair of interactive glasses".

On advertisement 4 (cheap bike deal), the test persons who noticed the advertisement on the website, were supposed to rate on a scale from 0 to 10 how much they like bikes, where 0 meant they did not like them and 10 meant they really like bicycles. The additional statements were "I go by bicycle often", "I never go by bicycle", "I sometimes go by bicycle" and "I want a new bicycle".

The test persons who noticed Advertisement 5 (Umeå University), were supposed to rate on a scale from 0 to 10 how much they like Umeå University, where 0 meant they do not like it and 10 meant they really like it. The additional statements were "I sometimes sit by the university pond" and "I never sit by the University pond". The reason for these statements was because the pond was in the picture of the advertisement.

If the test persons noticed Advertisement 6 (studera.nu), they were asked to mark on a scale from 0 to 10 how much they like studera.nu, where 0 meant not liking it and 10 meant do like it.

If the test persons noticed Advertisement 7 (hamburger restaurant), they were asked to mark on a scale from 0 to 10 how much they like the specific hamburger chain, where 0 meant do not like it and 10 meant they like it a lot. The additional statements were "I often eat at this hamburger chain", "I never eat at this hamburger chain" and "I sometimes eat at this hamburger chain".

On Advertisement 8 (smart watch), the test persons who noticed the advertisement were asked to rate how much they like the company producing the specific smart watch in the advertisement, since this was a product very typical for this company. 0 meant that they do not like the company and 10 meant that they really like the company. The additional statements were "I am interested in smart watches" and "I am not interested in smart watches" All of the pages in the questionnaire had additional blank rows, where the test person could write eventual opinions about the advertisement of the page.

3 Result

As shown in Figure 3, each man in the study saw in average 3.6 advertisements. The women in average saw 2 advertisements each.

1. Advertisement 1. The average result about how much the test person likes to read on a scale from 0 to 10 was 6.75 for women and 7.3 for men. The result from the additional questions showed that 3 out of 5 men read a lot of fiction and 2 out of 5 men are currently are reading a lot of course literature. The result from the women was that 1 out of 4 women answered that they read lot of fiction and 1 out of 4 women answered that they currently are reading a lot of course literature.
2. Advertisement 2. The only woman who saw Advertisement 2 rated a 10 on the scale from 0 to 10 about how much she likes coffee. She also answered that she was a coffee drinker, that she likes free stuff and that a lot of people in her surroundings are drinking coffee. The only man who saw Advertisement 2 rated a 5 on the scale and also answered that he was a coffee drinker.

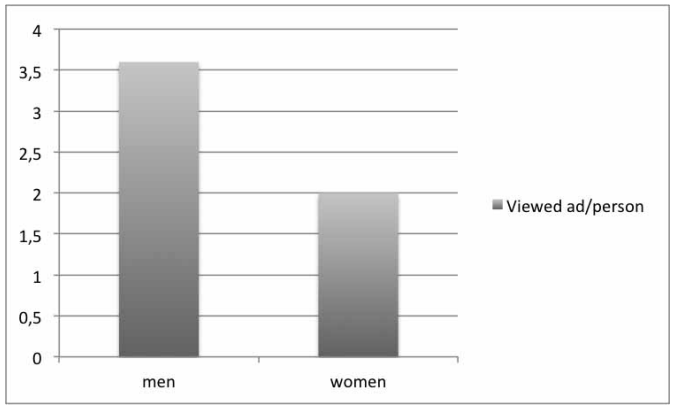


Fig. 3. The figure shows how many advertisements men and women saw per person in the test.

3. Advertisement 3. The two men who saw Advertisement 3 both rated a 10 on the scale about how interested they are in interactive glasses. One of them answered that he was interested in technology and the other answered that he once had tried a pair of smart glasses.The one woman who saw Advertisement 3 rated a 7 on the scale from 0 to 10 about how interested she is in smart glasses. She also answered that she was interested in technology.
4. Advertisement 4. On the scale from 0 to 10 about how much the test person likes bikes, one man rated a 0, another rated a 3 and the third one rated a 6. The average was therefore 3. The man who rated a 0 also answered that he does not have a bike. The man who rated a 6 answered that he wants a new bike and the man who rated a 3 answered that he uses his bike a lot. No one of the women saw Advertisement 4.
5. Advertisement 5. On the scale from 0 to 10 about how much the test person likes Umeå University the only woman who saw the advertisement rated a 6. She also answered that she never hangs out around the university pond. The only man who saw Advertisement 5 rated a 5.5 on the scale and also answered that he never hangs out around the university pond.
6. Advertisement 6. The only woman who saw Advertisement 6 did not rate the scale from 0 to 10 about what she thinks about studera.nu. The two men who saw the advertisement rated a 4 and a 7.5 on the scale.
7. Advertisement 7. On a scale from 0 to 10 about how much the test person likes the specific hamburger chain in Advertisement 7, the only girl who saw it rated a 3.5 and also answered that she sometimes eat at this hamburger chain. The two men who saw Advertisement 7 rated a 10 and a 5 on the scale. Both of them answered that they sometimes eat at this hamburger chain.
8. Advertisement 8. On the scale from 0 to 10 about how much the test person likes the specific electronics corporation, the only girl who saw the advertisement rated a 3.5. She also answered that she is not interested in smart watches. The two men who saw the advertisement rated a 8 and a 6.5 on

the scale. Both of the two men answered that they are interested in smart watches.

4 Discussion

Due to the time limitation, it was only possible to test 10 people in this study. There is a possibility that the outcome would have been different if there were more people tested.

Another limitation is that the advertisements were chosen by the author because they were believed to be gender neutral. There is a risk that they are not, which might affect the outcome.

The conclusions that can be drawn from the study is that the women in the test were ignoring advertising banners more than the men in the test, since the average amount of viewed banner per person was 3.6 for men an 2 for women.

In Figure 4 it is shown that 9 out of 10 test persons saw Advertisement 1, which makes it the most noticed advertisement of all banners in the test. Due to a study [8], users tend to scan websites in an F-shape, where they first read in a horizontal movement, usually the content bar then they move down a bit on the site and read in a second horizontal movement. After that they continue to scan the contents left side. This can be an explanation for why 9 out of 10 test persons noticed Advertisement 1, since it is a possibility that Advertisement 1 is in the second horizontal movement.

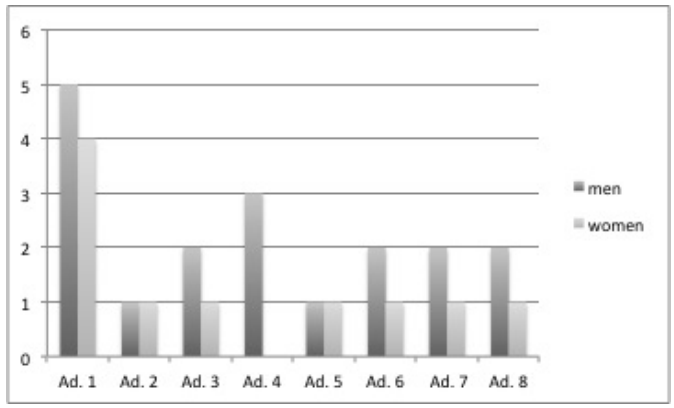


Fig. 4. The figure shows the different advertisements and how many of the men and women who saw them during the test.

A common denominator is that people who noticed an advertisement generally also had a positive attitude to the item that was being advertised. It was seldom that the test persons rated lower than a 5 on the scales of each advertisement. This only happened in 5 out of 28 ratings. A possible conclusion from this data is that people notice things they like. It would also have been interesting

to see what people who did not notice a specific advertisement thought about the advertised item.

References

[1] Krammer, V.: An effective defense against intrusive web advertising. In: Privacy, Security and Trust, 2008. PST '08. Sixth Annual Conference on. (Oct 2008) 3–14
[2] Hyland, T.: Why internet advertising? In: Webvertising. Vieweg+Teubner Verlag (2000) 13–17
[3] Cho, C.H.: Why do people avoid advertising on the internet? Journal of advertising **33**(4) (2004) 89–97
[4] Findahl, O.: Svenskarna och internet 2014. http://www.soi2014.se/ (2014)
[5] Putrevu, S.: Exploring the origins and information processing differences between men and women: Implications for advertisers. Academy of Marketing Science Review **10**(1) (2001) 1–14
[6] Hsieh, Y.C., Chen, K.H.: How different information types affect viewer's attention on internet advertising. Computers in Human Behavior **27**(2) (2011) 935–945
[7] Tullis, T., Siegel, M.: Does ad blindness on the web vary by age and gender? In: CHI'13 Extended Abstracts on Human Factors in Computing Systems, ACM (2013) 1833–1838
[8] Nielsen, J.: Usability 101: Introduction to usability (2003)

Does Practising with Wii Balance Board Affect Healthy Children's Balance?

Camilla Jakobsson

Department of Computing Science
Umeå University, Sweden
dit02ajn@cs.umu.se

Abstract. Wii fit balance boards are used as tools for rehabilitation for a number of balance related diseases but can it be used as a training method to avoid having balance and posture related issues? In this report we focus on investigating if and what effects Wii balance board training may have on healthy children's balance. This study includes six children, forming a training group (TG) with two five year olds and two seven year olds and a control person (CP) in each age group. The TG practiced Wii balance board games at five occasions within three weeks. Tests of balance exercises were conducted by all participants prior and after of the three weeks of training. The balance test composed by seven exercises included one leg standing on right/left foot with eyes open/closed, forward leaning and beam balancing both tandem and sideways balancing. Performing a paired t-test of the TG's pre- and post- test, for all balance exercises, none of the results could be proven as statistically affected by the WBB training. Looking at the paired t-test results from the exercises that showed visual improvements in TG compared to CG we found "standing on left leg with eyes open" showing a one tailed significance value of 0.0971, "forward leaning" with a one tailed significance value of 0.06725 and "tandem beam balancing" with a one tailed significance value of 0.38215.

1 Introduction

Wii gaming technology is frequently used for rehabilitation of diseases such as stroke [1], aquired brain injury [2], multiple sclerosis [3] and older adults with high risk of falling [4]. The Wii Balance Board (WBB) were in fact validated as a tool for measuring balance alongside an expensive laboratory device [5]. WBB training is also found in studies about children with cerebral palsy [6] and other balance impairments and body posture problems [7]. If balance training using WBB has a positive impact on children without balance related disorders, WBB training could help maintain good posture and balance so that children are less likely to develop balance impairments growing up. Wii Fit's balance board games (WFBBG) is probably one of the first games where the player is allowed to control the game character using balance skills. WFBBG attracts both adults and children, and research seems to focus on adults both with and without

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 31–45, January 2015.

balance impairments and also a little on children with balance impairments, thus there is a lack of focus on the area of children without balance disorders. To fill this gap, the focus of our study is to investigate what effects WFBBG may have on children's balance and also to investigate if there are any differences for children that have passed the age of six (that is seven year olds) at which age they reach the equilibrium of kinetics in their joint and the posture needed for advanced balance acts [8] and those who have not pass that age (that is five year olds). For this specific study it was also convenient to use this age grouping since there exists children in these ages that are already familiar with our test environment due to earlier visits. More details about the study can be obtained in further sections.

This report aims to investigate if and how a short period of training with balance games using WBB has an effect on children's balance. To complete this task a collection of six children are divided into two groups. The first group, training group(TG), gets to practice on the Wii balance board at five occasions during three weeks. The second group is participating as the control group (CG) getting no balance practice during the same amount of weeks.

One closely related study [9] conducted with physical education students in the age 20-22 compares training with WBB games with other balance training with eight weeks of training, in particular, 24 minutes two times a week in yoga, muscle strength and balance games. The results show that the balance improved and that there was no significant difference between the test group and control group. The study in [9] used Star Excursion Balance Test (SEBT) as a method for evaluation. The SEBT is easily set up and used for measurement by drawing four straight lines as an eight-way intersection. Standing at the intersection in the middle the purpose is to stretch each leg so that the foot reaches as far as possible towards the outer edge of each line. In [10] SEBT could not be validated as a tool for dynamic measurement due to the unreliable measures when the patient may force the body to make adjustments such as flex knee, hip movements so that the person's foot can reach even further along the line. Based on these results in [10] we decided against SEBT and created a new test influenced both by children and by Bergs Balance Scale (BBS) [11] adjusted to fit children and their abilities.

2 Method

The study consisted of three parts: pre test, training and post test. The study also included a background survey for us to become aware of other activities that may affect the outcome. The first part was to find a way to measure children's balance both before and after the training. This part also included gathering some relevant background information by having the participants (or their parents) to fill in a background survey shown in the appendix. In order to be able to compare results for each person and thereby discover the difference in performance in chosen exercises we set up a form where each exercise were measured and also a grade of stability visible during the exercise. The second part was

to manage the actual training with planning of exercises and logistics of the training sessions. This was conducted by planning and staying in touch with the children's parents and also motivating each child during training sessions and balance test.

2.1 Pre Test

To find a ground zero of balance abilities for our participants we needed a balance test. We contacted a physiotherapist at Norrlands Universitets sjukhus (NUS) in Umeå, Sweden to help finding a suitable method for measuring of balance, which resulted in BBS being the test that they use most frequently on older adults. We decided to investigate if BBS is a suitable method for measuring balance for this study. At first all participants filled in a survey for us to learn about their previous balance experiences and then a pre-balance test was conducted. After the pre-test the two subgroups were divided where the TG got to practice WFBBG five times within three weeks. The CG got no balance training during the same amount of weeks, and were asked not to train balance training other than performing daily activities that they would normally do.

2.2 Training

All participants in the TG practised this training, starting each session with a "Body test" consisting of a body scan detecting the child's standing posture followed by two small balance ability tests in order to determine the child's "Wii fit age". Each training session started with a "body test" consisting of two balance related tests. Prior to the first WBB training each participant created their own character (Mii), which is illustrated in Figure 1, to represent themselves in the Wii fit game.

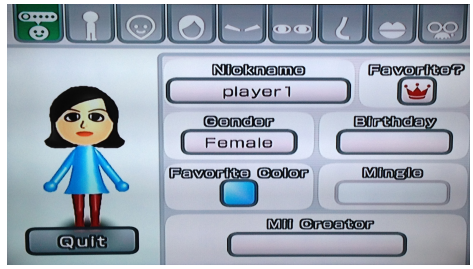


Fig. 1: The menu for creating a new Mii character, in the Wii fit game, which we use for WBB training.

After the body test warm-up the current participant started a ten minutes active training session of practising balance games according to our predefined schedule:

– Three minutes of soccer heading. Moving the center of balance from side to side in order to make their Mii head the balls coming in the center, the left

and right side of the screen, shown in Figure 2. In this game some of the flying objects were not balls, so the child has to be alert to identify the flying object too, shown in Figure 3.

Fig. 2: Heading: Mii in place of heading a ball.

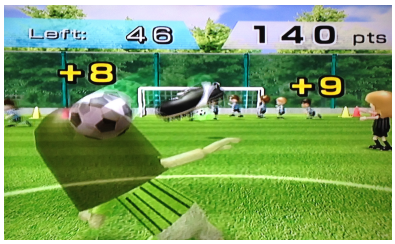


Fig. 3: Heading: Mii heading a ball and avoiding a shoe

– Two minutes of slalom shown in Figure 4 and Figure 5 using center of balance to navigate through the gates. Leaning forward makes the Mii go faster while leaning to the left or right makes the Mii turn that way.

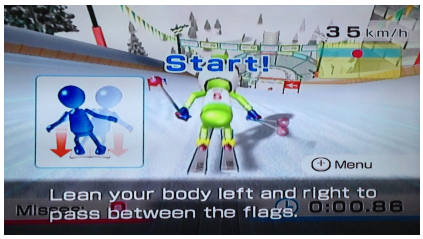


Fig. 4: Slalom: Hints on how to manage the course is displayed during start

Fig. 5: Slalom: An overview of the course getting feedback along the way

– Three minutes of ski jumping consisting of six tries. The Mii's performance depends on the player's ability to control their center of pressure (COP). As shown in Figure 6 there is a dot to aim for with your red COP dot. The Mii accelerate when the dots overlap, thereby the player can aim to accelerate, as in Figure 6 or not, as in Figure 7. When reaching the jump area, marked as red, the player is supposed to replicate a jump by flexing the knees and fast changing position to standing on their toes balancing "in the air", illustrated in Figure 7, all the way to landing. If the player looses balance or for some reason the WBB does not detect the "jump" the Mii trips and rolls down the hill as a snowball.
– Two (to three) minutes of table tilt. The Mii transforms into a marble, as in Figure 8 and the surface of which the marble is placed on tilts according to the child's center of balance that is detected by the WBB. The amount of time depends on how well the child performs since more seconds are added when completing a level by guiding all balls into a hole. For the children

Fig. 6: Ski jump: Showing the start of the exercise

Fig. 7: Ski jump: Showing the Mii balancing in the air

who did not reach the minimum of two minutes of practise on their first try were allowed a second try ending up with a total of two to three minutes of practise for the table tilt exercise. Figure 9 shows conditions for a more advanced level.

Fig. 8: Table tilt: The player's Mii represented as a marble.

Fig. 9: Table tilt: In more advanced levels the player tries to tilt both their own and their friends' marbles into the hole/holes

2.3 Post Test

After three weeks all participants took the post test. The results from the TG was then compared with the results from the CG to determine if training with WFBBG has any effect on the exercises comparing to data from the pre test.

2.4 Target Group Selection

In the following we describe on which grounds we selected the target group consisting of 6 children without balance related issues. At the age of 6 years the human body is at an equilibrium of kinetic in their joints and the posture needed for advanced balance acts. The 7 year old compared to an adult shows no difference in balance test [8]. Before the age of 6 children do not have the feed-forward posture [12](the ability to act upon the knowledge of posture balance instead of a learning-by-error way). This is why the age of 5 and 7 is of interest for this study. Selecting the three individuals in each age group is based on the children's interest to participate, the amount of time they (and their parents)

feel is appropriate to participate in this study. In each age group one of the three participants is appointed as control person (CP) in order for us to determine if WBB training has some effects or whether possible changes may be due to the participants' second try performing the same balance test exercises. The CG consisted of two CP's, one CP for each age group.

Since we want to be able to view and support them during practice we decided to select children that we think feel comfortable being in the test environment.

2.5 Evaluation of BBS

BBS [11] is a decision support test used at NUS in order to determine a patient's risk of falling. In BBS the patient performs a number of balance related exercises and depending on the time and ability to perform the exercises they are graded with zero to four points on every exercise resulting in an overall score for the test. If the patient scores less than 45 points the patient has an increased risk of falling [1].

2.6 Planning for Pre/Post Test

Testing the BBS on a pilot pair, composed by a girl age five and boy age seven, showed that some of the tests were not relevant in order to measure balance in this age interval, and most of the exercises in BBS have to be extended in amount of time per exercise as none of pilot testers had any problem to receive top scores in the BBS exercises. During the pre test a maximum time per exercise evolved and were set to two minutes. If any of our participants had been younger than four, no tandem walking but only feet together, as illustrated in Figure 10, would have been suitable according to [13].

Some of the chosen balance test exercises are indirect influenced by BBS.

Fig. 10: Blue slippers representing feet together (left) and red slippers for tandem feet position (right)

2.7 The Balance Test

- Exercise 1 and 2: Standing on one leg (right/left) for maximum 2 minutes. This exercise was inspired by BBS Exercise 14, for more details see [11] adding time and testing both feet.
- Exercise 3 and 4: One leg standing, eyes closed. (left/right) for as long as possible. This exercises are an extension of Exercise 1 and 2 above.
- Exercise 5: Forward leaning. The person holds their arms straight forward in a non stretched posture and the initial position is set by putting a 30 centimeter ruler between the wall and the fingertips. The ruler is marked at zero, twelve and 25. Starting at zero the person tries to stretch both hands as far as possible without touching the wall or moving their feet. This exercise was added directly from BBS Exercise 8 , for more details see [11].
- Exercise 6: Tandem balancing, as illustrated in Figure 10, on a beam (260 cm long, 5 cm wide). It is important to put the feet heel to toe in a slow pace to find balance in each step. This exercise were composed in order to test the children's ability of tandem balancing.
- Exercise 7: Sideways balancing with ears pointing towards each end of the length of the beam (260 cm long, 5 cm wide). Important to take small steps (feet are approximately 15-20 cm apart per step). This exercise was composed in order to test the children's ability of sideways balancing.

During these exercises we encouraged each child to not rush and informed them that there is no competition at all. The only comparison we are going to do is between pre- and post-test for the same person to be able to see if the WBB training makes any difference in these balance acts within the time frame of the training. An English translation of the complete Swedish Pre/Post test form is included in the appendix.

3 Results

One leg standing - eyes open

Visual differences between right and left foot can be seen from results in Figure 11a. Combining visual results from Figure 11b and the high error bar read from the box plot of Exercise 1, visualized in Figure 12a we can determine that there were at least one high measure causing variance of results in right foot standing, Exercise 1. From the paired t-test plot of Exercise 1, shown in Figure 13a, we can establish that the mean was higher during pre- than in post-test with a one tailed significance of (0.6042/2) 0.3021, which is >0.05, causing the result to be not statistically significant. In left foot standing a small visual improvement can be seen in Figure 11a. Compared with results from Figure 11b we can determine that the improvements comes from the seven-year-olds' measures causing variations within the TG in post- test, as shown in Figure 12a. In the paired t-test plot of Exercise 2, shown in Figure 13b, we can confirm that according to our measures the mean was higher during post- test, with a one tailed significance of (0.2053/2) 0.10265 that is >0.05, which means that this result is not statistically significant.

One leg standing - eyes closed

The result from Exercise 3 and Exercise 4, visualized in Figure 11a, shows that the plot from the TG followed the plot from the CG meaning that both groups had approximately the same improvements in Exercise 3 and approximately the same deterioration in Exercise 4. From Figure 11c, we can establish that in Exercise 3 the TG7 improved more than the TG5 deteriorated causing the overall results to show improvement and in Exercise 4 all results showed deterioration except the CP5 who performed the same result for both tests. Box plots, visualized in Figure 12b, shows the variation of the results within TG to be relatively low in pre- test of Exercise 3 and a lot of variation in post- test for the same exercise. In the box plot for Exercise 4, shown in Figure 12b, the distribution of the TG can be seen, showing a normal distribution with high variance for both pre- and post- test. In the paired t-test for Exercise 3, shown in Figure 13c, the mean was stated to be slightly higher in the post-test. The result has a one tailed significance of (0.5472/2) 0.2736 that is >0.05, which means that the results are not statistically significant. The paired t-test for Exercise 4, visualized in Figure 13d, shows that the mean was higher during the pre- test than the post- test with a one tailed significance of (0.1942/2) 0.0971 that is >0.05, which causes the result to be considered not statistically significant.

Forward leaning

In the visualization of the results for Exercise 5, Figure 11d, we find a clear improvement between pre- and post- test for the TG while the CG showed a deterioration in post-test results compared to pre-test. The clear result can also be seen in the box plot in Figure 12c, where the error bars are negligible compared to the results box. In the box plot we can also visually confirm that the range of the pre- test results is not overlapping the range of the post-test results which the paired t-test investigates further. The paired t-test for Exercise 5, showed in Figure 13e, states that the mean of the post test was higher than the mean of the pre- test with a one tailed significance of that result was (0.1345/2) 0.06725 that is >0.05, which means that this result is not statistically significant.

Tandem beam balancing

In the results from Exercise 6, shown in Figure 11e, there is a small visual improvement for the TG while the CG shows a deterioration comparing the results from pre- and post- test. This result can also be visually interpreted from Figure 11e, where the TG5 and the TG7 shows improvement while the CP5 and the CP7 shows a deterioration comparing their results from pre- and post- test. In the box plot for Exercise 6, shown in Figure 12d, the results from TG's pre- and post- test ranges are overlapping, which means that there is no clear difference to be detected by the paired t-test. By studying the paired t-test results from Exercise 6, shown in Figure 13f, the mean of the post test was stated to be higher than the mean of the pre-test with a one tailed significance of that

result was (0.7643/2) 0.38215 that is >0.05, which means that this result is not statistically significant.

Sideways beam balancing

The results from Exercise 7, visualized in Figure 11e, shows a visual deterioration for both the TG and the CG comparing the pre- test with the post- test result. In the box plot for Exercise 7, shown in Figure 12d, pre- test shows higher mean and wider span of data compared to the compact post- test span with negligible error bars. By studying the paired t-test results for Exercise 7, shown in Figure 14, the mean was stated to be higher during the pre- test compared to the mean of the post- test with a one tailed significance of that result was (0.2844/2) 0.1422 which is >0.05, which means that this result is not statistically significant.

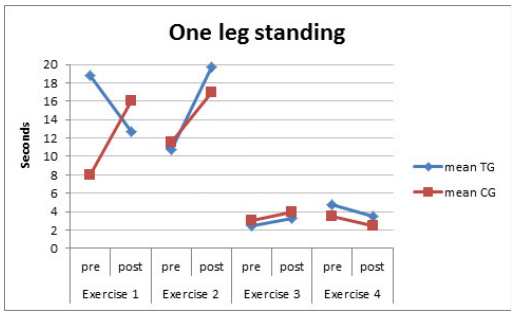
Summing up, the results shows that short term WBB training has most effect on forward leaning balance due to distinct improvements on all the TG compared to the CG in Figure 11d. In tandem beam balancing training with WFBBG might have had the effect of maintaining and make for small long term improvement. No one of the TG's pre-/post results could be found as statistically significant and thereby we can not exclude the data to be collected from the same range, as shown in Figure 12a, Figure 12b, Figure 12d except for Exercise 5, visualized in Figure 12c, where we had too few results to determine statistical significance.

4 Limitations

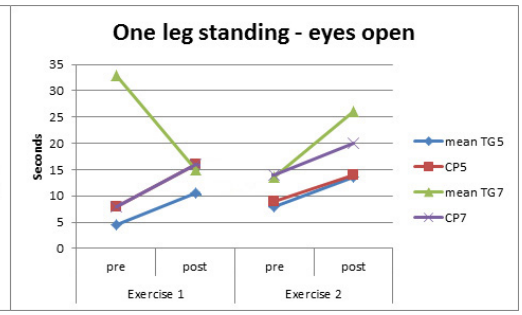
This study included children and real life testing based on measurements and on physical meetings for training. This brings up issues about voluntary participation since children in general have their own will and are not always in the mood of participating. Even though it is understandable after a whole day at school or preschool, it still requires lots of patience. Another issue for this study was that change of conditions such as participants dropping off, plan for training sessions to fit both our planned schedule and the participant's lives. Depending on what condition that changes it may have a huge effect on a small study.

In the planning of this study we also thought about the possibility for the sibling pair's to borrow one set of equipment to use in their homes. The plan was for us to start them up in their home so they could continue practising with their parents' supervision. Since there were too few participants and also the possibility that the kids could get different kind of instructions, we did not use this possibility.

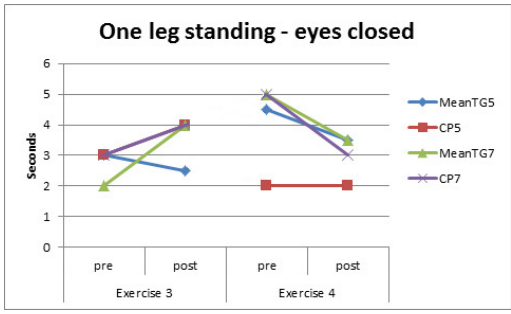
Since the Wii Fit game itself has some built in functions to choose appropriate exercises for the system to determine the fitness of the person playing, the body test's exercises were unfortunately out of our reach to control. The reason why we chose the body test was to allow the participants to get acquainted with the WBB before every training session. We also used the body test's built in function of logging the date in the Wii Fit game for each training session.



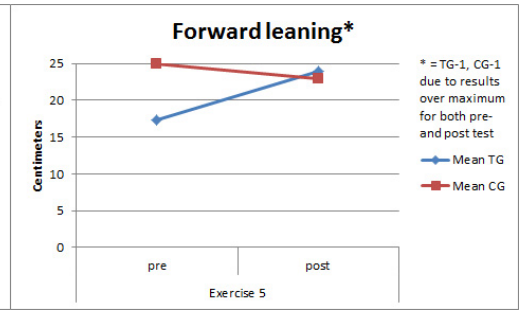
(a) Exercise 1-4 showing correlation of the TG and the CG



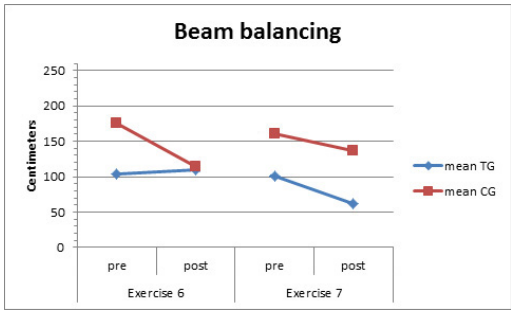
(b) Exercise 1-2 for each age group



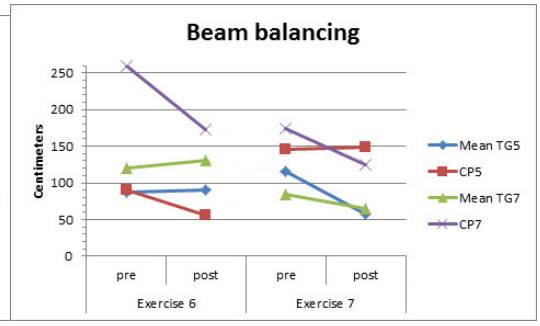
(c) Exercise 3-4 for each age group



(d) Exercise 5 showing correlation of the TG and the CG

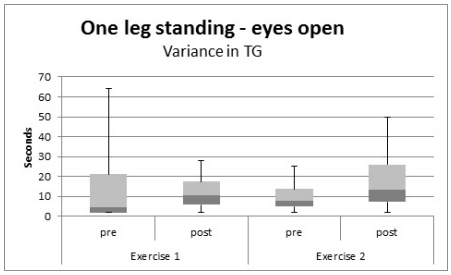


(e) Exercise 6-7 showing correlation of the TG and the CG

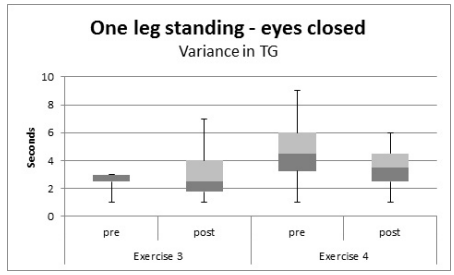


(f) Exercise 6-7 for each age group

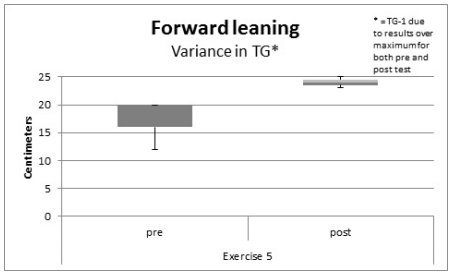
Fig. 11: Test results from pre- and post test



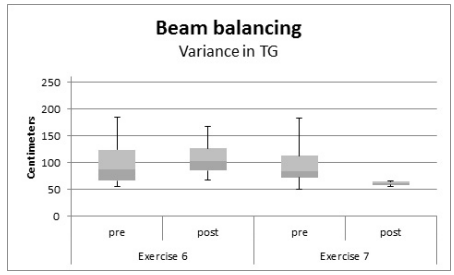
(a) Distribution of results within the TG for Exercise 1-2



(b) Distributution of results within the TG for Exercise 3-4



(c) Distribution of results within the TG for Exercise 5



(d) Distribution of results within the TG for Exercise 6-7

Fig. 12: Box plots of variance inside of the TG for pre and post test of each exercise

Exercise 1 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.6042
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 1 post minus Exercise 1 pre equals -6.00
95% confidence interval of this difference: From -39.07 to 27.07

Intermediate values used in calculations:
t = 0.5774
df = 3
standard error of difference = 10.392

Review your data:

Group	Exercise 1 post	Exercise 1 pre
Mean	12.75	18.75
SD	11.30	30.26
SEM	5.65	15.13
N	4	4

(a) Results of t-test for Exercise 1

Exercise 2 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.2053
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 2 post minus Exercise 2 pre equals 9.00
95% confidence interval of this difference: From -8.77 to 26.77

Intermediate values used in calculations:
t = 1.6121
df = 3
standard error of difference = 5.583

Review your data:

Group	Exercise 2 post	Exercise 2 pre
Mean	19.75	10.75
SD	21.20	10.05
SEM	10.60	5.02
N	4	4

(b) Results of t-test for Exercise 2

Exercise 3 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.5472
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 3 post minus Exercise 3 pre equals 0.75
95% confidence interval of this difference: From -2.78 to 4.28

Intermediate values used in calculations:
t = 0.6765
df = 3
standard error of difference = 1.109

Review your data:

Group	Exercise 3 post	Exercise 3 pre
Mean	3.25	2.50
SD	2.63	1.00
SEM	1.31	0.50
N	4	4

(c) Results of t-test for Exercise 3

Exercise 4 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.1942
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 4 post minus Exercise 4 pre equals -1.25
95% confidence interval of this difference: From -3.64 to 1.14

Intermediate values used in calculations:
t = 1.6667
df = 3
standard error of difference = 0.750

Review your data:

Group	Exercise 4 post	Exercise 4 pre
Mean	3.50	4.75
SD	2.08	3.30
SEM	1.04	1.65
N	4	4

(d) Results of t-test for Exercise 4

Exercise 5 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.1345
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 5 post minus Exercise 5 pre equals 6.67
95% confidence interval of this difference: From -5.07 to 18.41

Intermediate values used in calculations:
t = 2.4434
df = 2
standard error of difference = 2.728

Review your data:

Group	Exercise 5 post	Exercise 5 pre
Mean	24.00	17.33
SD	1.00	4.62
SEM	0.58	2.67
N	3	3

(e) Results of t-test for Exercise 5

Exercise 6 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.7643
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 6 post minus Exercise 6 pre equals 6.50
95% confidence interval of this difference: From -56.53 to 69.53

Intermediate values used in calculations:
t = 0.3282
df = 3
standard error of difference = 19.805

Review your data:

Group	Exercise 6 post	Exercise 6 pre
Mean	110.00	103.50
SD	43.00	58.07
SEM	21.50	29.04
N	4	4

(f) Results of t-test for Exercise 6

Fig. 13: Results from paired t-test of the TG for Exercise 1-6

Exercise 7 - Paired *t* test results of the TG

P value and statistical significance:
The two-tailed P value equals 0.2844
By conventional criteria, this difference is considered to be not statistically significant.

Confidence interval:
The mean of Exercise 7 post minus Exercise 7 pre equals -39.50
95% confidence interval of this difference: From -136.18 to 57.18

Intermediate values used in calculations:
t = 1.3002
df = 3
standard error of difference = 30.380

Review your data:

Group	Exercise 7 post	Exercise 7 pre
Mean	61.25	100.75
SD	4.86	57.41
SEM	2.43	28.70
N	4	4

Fig. 14: Results from paired t-test of the TG for Exercise 7

Since all children have their own abilities and motor skills we decided to consider the answers from the background survey, shown in the appendix, for their weekly activities (sports) as a part of every child's unique ability instead of trying to interpret what could have caused what outcome. If there would have been more participants in this study the mean value for each exercise would probably have been more tolerant to single deviating results to effect the outcome.

5 Discussions and Future Work

The results were showing that forward leaning, Exercise 5, were improved by the WBB training in both Figure 11d, Figure 12c and in the t-test the difference were confirmed but not statistically significant by 0.017 units. By having these results so close to being statistically significant with only four participants in the TG shows that the result is worth mentioning. If there would have been more TG data for this exercise pointing in the same direction, the results for this exercise would have been statistically significant.

Results showing visual improvements between the TG and the CG for Exercise 5 was surprising to us because we usually do not consider forward leaning when thinking of balance. One reason why the participants in the TG showed great progress in Exercise 5 might be that people probably do not use this forward leaning posture that often. Considering the result for Exercise 5, forward leaning balance for five- and seven year olds could probably be improved in three weeks by following the training procedure used in this study. We can also speculate that training WFBBG can lead to more stable performance in tandem beam balancing (TBB) and the long term effects of TBB might be of interest to investigate further.

The results from Exercise 4 could almost be found statistically significant for pre- test having a higher mean than post- test. Since our hypothesis was if the training had any effect on the exercises, it is not likely to believe that balance training caused the deterioration of post- test results compare to the pre- test results, since most training in general seems to have a positive effect in most cases.

In order to know if this kind of training has a preventive effect on future posture and balance related issues, we recommend to investigate how it may affect children in a long term study with validated test equipment and method. We also recommend to have many more participants and aim for a validated test method to avoid assessing differences. Something to consider is the number of tries to get a result on a balance test exercise or how to explain the exercise so that the first try is both valid and measurable. Another question is what should be measured in order to generate data that makes for conclusions to be drawn.

For future studies with children using Wii Fit game we strongly recommend to skip the body test since taking it obligates for the player to set a weight goal and the BMI is measured and weight is pointed out in a way that children do not have to be aware of in early ages. Within the body test some players were asked to answer questions about their friends' appearance in real life considering weight loss/gain according to their friend's measures within the game. Fortunately our participants had not reached the age of where they start to understand written English, which made it easier for us to skip those screens without mentioning the content.

References

[1] Lundholm, A., Skoglund, P., Vikström, M., Vikman, I.: Balansträning av patienter med stroke med hjälp av basal kroppskännedom. Luleå: Institutionen för Hälsovetenskap Sjukgymnastprogrammet, Luleå tekniska univeristet (2003)

[2] Gil-Gómez, J.A., Lloréns, R., Alcañiz, M., Colomer, C.: Effectiveness of a Wii balance board-based system (ebavir) for balance rehabilitation: a pilot randomized clinical trial in patients with acquired brain injury. Journal of neuroengineering and rehabilitation **8**(1) (2011) 30

[3] Prosperini, L., Fortuna, D., Giannì, C., Leonardi, L., Marchetti, M.R., Pozzilli, C.: Home-based balance training using the wii balance board a randomized, crossover pilot study in multiple sclerosis. Neurorehabilitation and neural repair **27**(6) (2013) 516–525

[4] Young, W., Ferguson, S., Brault, S., Craig, C.: Assessing and training standing balance in older adults: a novel approach using the nintendo wii balance board. Gait & posture **33**(2) (2011) 303–305

[5] Clark, R.A., Bryant, A.L., Pua, Y., McCrory, P., Bennell, K., Hunt, M.: Validity and reliability of the nintendo wii balance board for assessment of standing balance. Gait & posture **31**(3) (2010) 307–310

[6] Jelsma, J., Pronk, M., Ferguson, G., Jelsma-Smit, D.: The effect of the nintendo Wii fit on balance control and gross motor function of children with spastic hemiplegic cerebral palsy. Developmental neurorehabilitation **16**(1) (2013) 27–37

[7] Shih, C.H., Shih, C.T., Chu, C.L.: Assisting people with multiple disabilities actively correct abnormal standing posture with a nintendo wii balance board through controlling environmental stimulation. Research in developmental disabilities **31**(4) (2010) 936–942

[8] Assaiante, C.: Development of locomotor balance control in healthy children. Neuroscience & biobehavioral reviews **22**(4) (1998) 527–532
[9] Vernadakis, N., Gioftsidou, A., Antoniou, P., Ioannidis, D., Giannousi, M.: The impact of nintendo Wii to physical education students' balance compared to the traditional approaches. Computers & Education **59**(2) (2012) 196–205
[10] Kinzey, S.J., Armstrong, C.W.: The reliability of the star-excursion test in assessing dynamic balance. Journal of Orthopaedic & Sports Physical Therapy **27**(5) (1998) 356–360
[11] Lundin Olsson, L., Jensen, J., Waling, K.: Bergs balansskala, den svenska versionen av the balance scale. Sjukgymnasten, Vetenskapligt supplement **1** (1996) 16–9
[12] Berger, W., Trippel, M., Assaiante, C., Zijlstra, W., Dietz, V.: Developmental aspects of equilibrium control during stance: a kinematic and emg study. Gait & Posture **3**(3) (1995) 149–155
[13] Liao, H.F., Mao, P.J., Hwang, A.W.: Test-retest reliability of balance tests in children with cerebral palsy. Developmental Medicine & Child Neurology **43**(3) (2001) 180–186

Appendix

Balance Pre-/Posttest

Supports for balance evaluation for each exercise
4. No problem
3. A little wobble
2. Very swaying
1. Moving their feet
0. Loses balance / put down the other foot

Name:_____

Exercise	Pre_Time	Pre_Evaluation.	Post_Time	Post_Evaluation.
1. Eyes open. Stand on right leg. (Max. 2 min)				
2. Eyes open. Stand on left leg. (Max. 2 min)				
3. Closed eyes. Stand on right leg.				
4. Closed eyes. Stand on right leg.				
5. Fix the feet on the floor and faeces arms forward with body upright for zero position. Reach as far as your balance allow. (Maximum 25 cm)				
6. Tandem walking balancing on a bar (5cm wide, length approx. 2-2.5 m)				
7. Sideways walking balancing on a bar (5cm wide, length approx. 2-2.5 m)				

Fig. 15: Pre/Post test

Survey for background information concerning the study: "What effect on children's balance can identified having them practising Wii Fit games using Nintendo Wii balance board for three weeks?"

Name:_____

Age(ex. 4 ½):_____

□ *the child has no current balance related diseases (what we know of)*

Rides a bike..

□ independently

□ with someone holding/running along

□ on a running bike

□ with training wheels

□ not yet on a two wheel bike

Practices/have been practising balance related training. Put an X and estimate to what extent the activity is practiced. (during study)

□ downhill skiing_____

□ blading _____

□cross country skiing _____

□ ball sports _____

□ gymnastics_____

□ biking
(according to responses above) _____

□ other_____

□ no balance related practise

Thank you for your respond!

// Camilla Jakobsson, Interaktion & Design student at Umeå University

Fig. 16: Background survey

Solving the Layton Arrow Puzzle

Jonas Lindh Morén

Department of Computing Science
Umeå University, Sweden
dv08jmh@cs.umu.se

Abstract. In this paper we investigate a combinatorial puzzle dubbed the Layton arrow puzzle. By identifying and applying a number of heuristics and optimizations, we develop and present an efficient algorithm to solve general instances of the puzzle.

1 Introduction

The Layton arrow puzzle has its origins in the Nintendo DS game Professor Layton and the Spectre's Call. It is a puzzle game with many similar problems.

Fig. 1. The problem as shown in Professor Layton and the Spectre's Call.

The exact wording of the problem in the game can be seen in Figure 1. This is all the information the player is given. The player is then meant to swap any two of the arrows to create a path that allows us to enter the board via an arrow,

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 47–56, January 2015.

travel along all other arrows on the board, and then exit the board through the final arrow. The player is only allowed a single swap. The aim of our paper is to find an efficient (in terms of time complexity) algorithm to solve the general problem. We begin by first defining this general problem.

1.1 Related Work

Puzzle theory as a research field is far less active than its cousin, game theory (which deals primarily with games involving at least two players - though some consider puzzles a special case of game theory). For examples of related articles, see [1] [2] or [3]. For some excellent compilations of algorithmic puzzles and general background in the area, see [4] or [5].

1.2 The Problem Defined

The **board** is represented by a $n \times n$ matrix, where each element in the matrix represents a **tile**. In the original problem, the board has 3×3 tiles.

Each tile has a **direction**. There are 4 directions; up, right, down, and left. We say that a tile **faces** its direction; for example, a tile may face up, right, down, or left.

If a tile does not have neighboring tiles in all 4 directions, we say that it is a **peripheral** tile. If a peripheral tile faces the **edge** of the board (that is, does not face another tile), we say that the tile is an **exit** to the board. Conversely, if a peripheral tile faces the opposite direction, we call it an **entrance**.

If a tile t_1 faces a neighboring tile t_2, we say t_1 is an **incoming** tile of t_2. If all neighbors of t_2 face t_2, it would thus have 4 incoming tiles.

A tile may be **swapped** with any other tile by switching position with each other. Swapping a tile t_1 with another tile t_2 is represented by $swap(t_1, t_2)$.

An **instance** defines a $n \times n$ board with corresponding tiles and directions. An instance is **solved** if we can trace a path from an entrance to the exit, following the direction of the tiles, and visiting every tile exactly once. An instance is **solvable** if it may be solved by performing one swap.

A **solution** is thus given by a swap $swap(t_1, t_2)$ such that the instance is solved after applying the swap.

See Figure 2 and 3 for examples showing these terms.

For the remainder of this paper, we use n to refer to the board size (length of the sides of the board).

1.3 Research Question

How efficiently (in terms of time complexity) can a solution be found for an instance?

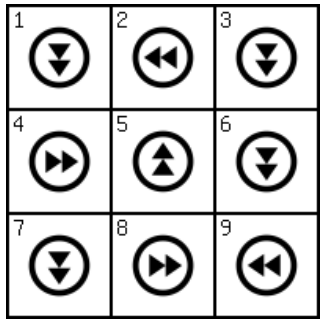


Fig. 2. A solvable instance representing the original problem. It has four possible entrances (tiles 1, 3, 4, and 9) and one possible exit (tile 7).

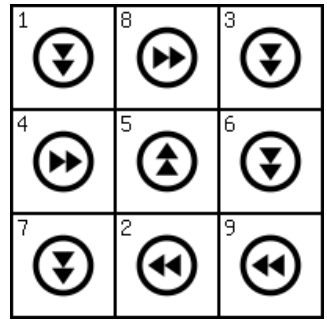


Fig. 3. The solved instance, where tiles 2 and 8 have been swapped. Using tile 1 as an entrance and tile 7 as an exit, we can trace a path from the entrance to the exit while visiting every tile exactly once.

2 The Problem Explored

Let us assume we are faced with an $n \times n$ solvable instance of the problem. We know that we must make exactly one swap in order to solve the instance, but we do not know which two tiles must be swapped.

If we were to take the brute-force approach, we have $\binom{n^2}{2}$ possible swaps that could be made. Furthermore, after each swap we would have to check if the instance is solved, which is an $\Omega(n^2)$ operation (assuming we attempt to traverse the board from the exit backwards).

However, by looking at certain heuristics, we are able to reduce the search space considerably. Consider some properties that must be true if an instance is solved.

- **There must be exactly one exit**
  As per the definition, a solution requires exactly one exit.
- **There must be exactly one entrance that has no incoming tiles**
  As per the definition, a solution requires an entrance. The entrance cannot have any incoming tiles without creating unreachable tiles (or a cycle).
- **All other tiles must have exactly one incoming tile**
  If there are any tiles except the entrance that does not have exactly one incoming tile, it means that at least one tile will be unreachable.

Fulfilling the above properties does not guarantee that a board is solved (see Figure 4), but for all solved boards they must be fulfilled. We can leverage this information to prune the search space.

2.1 Heuristics

In order to prune the search space, we consider a number of different heuristics. Primarily, we look at *exits* and *incoming tiles*.

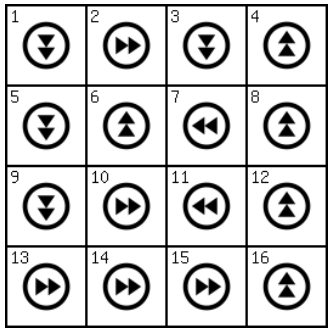


Fig. 4. An example of an instance that satisfies the properties established in Section 2, yet is not solved.

Consider a solved instance. In order to make the instance unsolved (and thus also solvable), we swap two random tiles (with different directions). What are the possible effects of our swap?

- We may move the existing exit to another spot
- We may create one or two additional exits
- We may remove the existing exit
- We may create tiles without incoming tiles
- We may create tiles with two or three incoming tiles

In order to find the solution for the instance, we have to undo these effects. Since a swap is guaranteed to produce the above effects around both tiles that are swapped, we can always solve an instance naively by swapping two tiles within the set of tiles containing:

- all exits,
- the incoming tiles of all tiles that have more than one incoming tile, and
- all tiles without any incoming tiles, and all their neighbors.

This set is of constant size. Counting incoming tiles and checking if an instance is solved are both $\Omega(n^2)$ operations (they require that we iterate over all tiles). Finding all the exits is an $\Omega(n)$ operation (it requires that we iterate over all the peripheral tiles). Since we have reduced the number of possible swaps to a constant number (assuming the instance was solvable), solving an instance in this manner is thus also an $O(n^2)$ operation.

The astute reader may ask himself why we include all tiles without any incoming tiles and not just their neighbors. Since we wish to ensure that only the entrance has zero incoming tiles, shouldn't we merely swap the neighbors of all other tiles with zero incoming tiles to ensure they have exactly one incoming tile? Consider Figure 5 and Figure 6 for an example when we must swap a tile with zero incoming tiles.

A pseudocode implementation of an algorithm utilizing the heuristics discussed in this section may be seen in Algorithm 1.

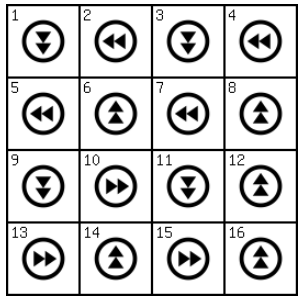


Fig. 5. An example of an instance where we must swap a tile that has no incoming tiles (tile 9).

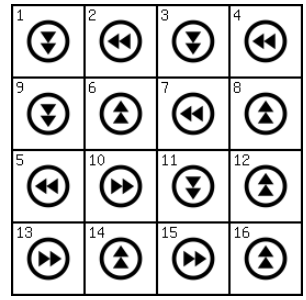


Fig. 6. The solved instance, where tile 5 has been swapped with tile 9.

Algorithm 1: NaiveArrowSolve(I)

Input: A $n \times n$ solvable instance I.
Output: A $swap(t_1, t_2)$ that solves the instance.

$E = \emptyset$;
foreach *peripheral tile* $t \in I$ **do**
    if t *is an exit* **then** add t to E;
    ;
$Z = \emptyset; T = \emptyset$;
foreach *tile* $t \in I$ **do**
    if t *has no incoming tiles* **then** add t to Z;
    ;
    if t *has more than one incoming tile* **then** add t to T;
    ;
$Z_N = \emptyset$;
foreach *tile* $t \in Z$ **do**
    add all neighbors of t to Z_N;
$T_I = \emptyset$;
foreach *tile* $t \in T$ **do**
    add all incoming tiles of t to T_I;
$S = E \cup Z \cup Z_N \cup T_I$;
foreach *tile* $t_1 \in S$ **do**
    foreach *other tile* $t_2 \in S$ **do**
        if t_1 *and* t_2 *face the same direction* **then** skip to next t_2;
        ;
        swap t_1 with t_2;
        if I *is solved* **then return** $swap(t_1, t_2)$;
        ;
        else swap t_1 with t_2;
        ;

2.2 Optimizations

We have established that we can reduce the search space of possible swaps to a constant size. However, there are several optimizations that can be made to reduce the search space further.

Assume we have a solvable instance. Let us consider a few scenarios where we can reduce the search space further.

- **Three exits**
  We know that the solution is given by swapping two of the exits, since the solved instance must have exactly one exit ($\binom{3}{2} = 3$ possible swaps).
- **A tile with three incoming tiles**
  We know that the solution is given by swapping two of its incoming tiles, since the solved instance cannot have any tile with more than one incoming tile ($\binom{3}{2} = 3$ possible swaps).
- **Two tiles with two incoming tiles each**
  We know that the solution is given by swapping two of their incoming tiles, since again, the solved instance cannot have any tile with more than one incoming tile ($\binom{4}{2} = 6$ possible swaps).
- **One tile t_1 with two incoming tiles, and two exits**
  We know that the solution is given by swapping one of the incoming tiles of t_1 with one of the exits, since we must reduce the number of incoming tiles of t_1 by one and remove one exit ($\binom{2}{1} \times \binom{2}{1} = 4$ possible swaps).

Furthermore, by looking at the local effects of each swap, we can avoid having to do an expensive $O(n^2)$ check for every swap to determine whether or not the swap solved the instance.

Assume we have a solvable instance. Let E be the set containing all exits, Z be the set containing all tiles with zero incoming tiles, and T be the set containing all tiles with two incoming tiles. As we discussed in Section 2.1, we know that the solution is given by swapping two tiles from the set containing all tiles in E, all neighbors of every tile in Z, and all incoming tiles of every tile in T.

After a swap is made, we check whether the swap is viable by looking at how the sets E, Z and T were affected. If the swap is viable, E should now contain exactly one exit, Z should contain exactly one entrance, and T should be empty. If these properties are not fulfilled, the instance *cannot* be solved (as established in Section 2). *However*, even if the swap is viable, the instance is not necessarily solved. We still have to do the $O(n^2)$ check to see if the instance is actually solved (see Figure 4).

To determine how E, Z and T have changed, we only need to look at the tiles and the neighbors of the tiles in E, Z, and T as they were before the swap. Since this is a set of constant size, checking if a swap is viable is also a constant operation.

A pseudocode implementation of an algorithm utilizing the optimizations discussed in this section (in addition to the heuristics discussed in Section 2.1) may be seen in Algorithm 2.

Algorithm 2: OptimizedArrowSolve(I)

Input: A $n \times n$ solvable instance I.
Output: A $swap(t_1, t_2)$ that solves the instance.

$E = \emptyset$;
foreach *peripheral tile* $t \in I$ **do**
    if t *is an exit* **then** add t to E;
    ;
if E *contains three exits* **then return** SwapBetween(E, E, E, I);
;
$Z = \emptyset; T = \emptyset$;
foreach *tile* $t \in I$ **do**
    if t *has no incoming tiles* **then** add t to Z;
    ;
    if t *has more than one incoming tile* **then** add t to T;
    ;
if T *contains a tile* t *with three incoming tiles* **then**
    $T_I = \{t_i \mid t_i$ is an incoming tile of $t\}$;
    return SwapBetween(T_I, T_I, $E \cup Z \cup T$, I);
else if T *contains two tiles* t_1 *and* t_2 *with two incoming tiles each* **then**
    $T_1 = \{t_i \mid t_i$ is an incoming tile of $t_1\}$;
    $T_2 = \{t_i \mid t_i$ is an incoming tile of $t_2\}$;
    return SwapBetween(T_1, T_2, $E \cup Z \cup T$, I);
else if E *contains two exits and* T *contains one tile* t_1 *with two incoming tiles* **then**
    $T_I = \{t_i \mid t_i$ is an incoming tile of $t\}$;
    return SwapBetween(T_I, E, $E \cup Z \cup T$, I);
$Z_N = \emptyset$;
foreach *tile* $t \in Z$ **do**
    add all neighbors of t to Z_N;
$T_I = \emptyset$;
foreach *tile* $t \in T$ **do**
    add all incoming tiles of t to T_I;
$S = E \cup Z \cup Z_N \cup T_I$;
return SwapBetween(S, S, $E \cup Z \cup T$, I);

Algorithm 3: SwapBetween(T_1, T_2, C, I)

Input: Two sets of tiles T_1 and T_2 to swap between, a set of tiles C to check for local changes, and a solvable instance I.
Output: A $swap(t_1, t_2)$ that solves the instance.

```
foreach tile t1 ∈ T1 do
    foreach other tile t2 ∈ T2 do
        if t1 and t2 face the same direction then skip to next t2;
        ;
        swap t1 with t2;
        if IsViable(C, I) and I is solved then return swap(t1, t2);
        ;
        else swap t1 with t2;
        ;
```

Algorithm 4: IsViable(C, I)

Input: A set of tiles C to check for local changes and an instance I.
Output: **true** if the instance is viable, **false** otherwise

```
seenExit = false;
seenEntrance = false;
foreach tile t ∈ C do
    i = number of incoming tiles of t;
    if i = 1 and t is an exit then
        if seenExit then return false;
        ;
        else seenExit = true;
        ;
    else if i = 0 and t is an entrance then
        if seenEntrance then return false;
        ;
        else seenEntrance = true;
        ;
    else if i ≠ 1 then
        return false;
return seenExit;
```

3 Results and Discussion

Let us look back at the research question as stated in Section 1.3.

> *How efficiently (in terms of time complexity) can a solution be found for an instance?*

We have developed an algorithm that can solve general instances of the Layton arrow puzzle in $O(n^2)$ time. There are two things that limit us to an $O(n^2)$ time complexity:

1. Determining the number of incoming tiles for all tiles, which requires iterating over all tiles (an $\Omega(n^2)$ operation which is done once)
2. Determining whether an instance is solved, which also requires us to iterate over all tiles assuming we attempt to trace a path from the exit backwards (an $\Omega(n^2)$ operation which is done a constant number of times)

As we can see, we are fundamentally limited by the fact that we at some point need to iterate over all tiles on the board. This suggests that we will never be able to solve general instances faster than $O(n^2)$ since any two tiles may need swapping and we cannot know which without iterating over the tiles to discover inconsistencies.

What we *can* do, however, is to reduce the constant factors. While the naive algorithm presented reduces the search space of possible swaps to a constant size, we have shown that there are several optimizations that can be made to further reduce the number of possible swaps. We have also shown that we can look at the local changes of each swap to avoid doing a full $O(n^2)$ solution check after each swap made. Undoubtedly there are additional optimizations that can reduce the number of swaps required even further.

4 Future Work

There are several variations of the Layton arrow puzzle that could be interesting to explore. For example, one might imagine a version with non-square boards, a version where two or more swaps must be made, or perhaps even a version requiring two entrances, two exits, and two paths to be traced between from entrance to exit.

As mentioned in Section 3, there are also undoubtedly further optimizations that can be made to reduce the constant factors of the algorithm even further.

A parallel version of the algorithm could also be an interesting method of solving larger instances of the problem.

References

[1] Lagae, A., Dutré, P.: The tile packing problem. Geombinatorics **17** (2007)

[2] Demaine, E., Hearn, R.: Playing games with algorithms: Algorithmic combinatorial game theory. Games of No Chance 3 **56** (2009)
[3] van de Liefvoort, A.: An iterative algorithm for the reve's puzzle. The Computer Journal **35** (1989)
[4] Levitin, A.: Algorithmic puzzles. Oxford University Press (2011)
[5] Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.: Algorithmic Game Theory. Cambridge University Press (2007)

Hand Sign Recognition Using a Leap Motion Sensor and k-Nearest Neighbors Classification

Martin Lärka

Department of Computing Science
Umeå University, Sweden
id10mla@cs.umu.se

Abstract. Hand sign recognition can be used to achieve a more natural way of human computer interaction. And by finding a more natural human computer interaction the boundaries that conventional interaction with keyboards and mice can have. This paper researches if a Leap motion sensor could be used as an input device for a computer. A computer program was trained and tested on the hand signs for 16 letters, and the numbers one to nine, from the American sign language. The input data consists of the directional values of each bone in the hand, which is then classified using the kNN. The program achieved an average correct classifications of 85% on the letters, and 95.37% on the numbers. The program reached a 100% correct classifications on seven out of the 16 letters, and six out of the nine numbers.

1 Introduction

The search for a natural way of communicating with computers is almost as old as the computer itself. As early as 1980, Bolt searched for a way to control a graphical interface using only voice and gesture commands [1]. Yet even today, the most common way to communicate with a computer is still to use a keyboard and a mouse. As society craves for computers that are embedded in our daily lives, a natural way of interacting with computers is crucial, or else the interaction between humans and computers can be a reason for slowing down the development of more embedded computer systems [2].

In 1997, Cui et al. constructed a framework that could recognize 28 different hand signs with a 93.2% recognition rate [3]. Their approach was an analytic way of classifying the gestures, as apposed to the neural network that Murthy et al. used in 2010 to receive an 89% recognition rate with their framework that would recognize hand gestures from 10 categories [4].

Accomplishing a non-invasive natural way of communicating with a computer uses a substantial amount of computing power. This need for computer power is why earlier methods in computer vision used gloves and markers to make it easier for computers to analyze the input data. But those methods were an inconvenient way of interacting with a computer [2], and never made a break through with the every day user.

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 57–67, January 2015.

With the available computer power of today, the invasiveness of computer vision can be greatly reduced. Numerous researchers have been able to successfully recognize hand gestures using video input [2, 4, 5, 3, 6]. Gesture recognition has also entered the field of gaming. Controllers like Xbox Kinect[1] and Playstation Move[2] let users interact with the games in a more natural way, using their whole bodies instead of just manipulating a controller with their hands.

A device that also tracks the user's movement is the Leap Motion[3] sensor. The sensor is a device that works much in the same way as the Xbox Kinect and Playstation Move, but in a smaller scale. The sensor tracks and models the user's hands when they are placed inside the sensor's field of vision. With the sensor's accompanying application programming interface(API), the position of each bone in the hand is extracted.

In this project, the Leap Motion sensor was used to collect input data of the American sign language(ASL) alphabet. This data was then used to train a computer program to recognize the hand signs. The computer program developed for this project was used to measure how accurate a computer can recognize ASL hand signs performed outside of a training set using a Leap Motion sensor.

The rest of this paper is organized as follows. In Section 2, an overview of the approach that was used in this project is described. The overview is then followed by Section 3 where the results of the study is presented. The results is then discussed in the concluding Section 4 of the paper.

2 Overview of the Approach

The purpose of this project was to create a computer program that with a Leap Motion sensor would be able to recognize different hand signs. The computer program uses the output generated from the sensor and classifies it by comparing the output data to the data that was used when training the computer program. Comparing the data from the hand signs is performed using the k-nearest-neighbors algoritm (kNN)[7].

The Leap Motion Sensor uses two stereo cameras and three LEDs to track infrared light with a wavelength of 850 nanometers. The cameras have wide angle lenses that gives the sensor a large interaction space. The interaction space is formed as an inverted pyramid ending roughly 60 centimeters away from the sensor.

When an image has been taken by the sensor, the Leap Motion service uses algorithms to analyze and reconstruct a 3d representation of what the sensor is perceiving. After that, a tracking layer matches the data to extract entities like fingers and tools. Then information can be obtained from the accompanying

[1] The Xbox Kinect information page. http://www.xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one

[2] Information page from Wikipedia on the Playstation Move controller. http://en.wikipedia.org/wiki/PlayStation_Move

[3] The Leap Motion sensor information page. https://www.leapmotion.com/product

API. The information can be position of joints and direction of the different bones[4].

The 3D representation generated by the Leap Motion API can be visualized using a program that can be seen in Figure 1. This program was used while the test subjects was inputing data to ensure that the data was not misleading. Misleading data could could sometimes occur when the sensor had to handle occlusion, e.g. when fingers was hidden from the sensors field of view.

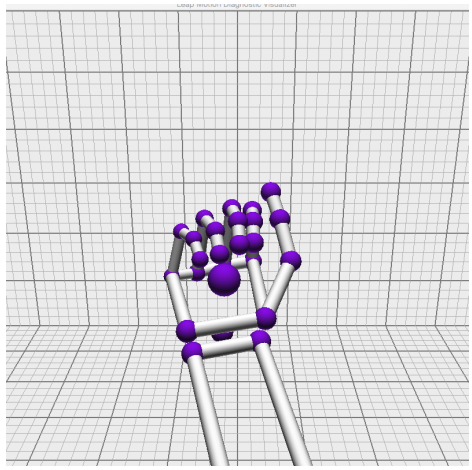


Fig. 1. A view from the visualizer program that was used to ensure that the hand signs was correctly read by the Leap Motion sensor

To simplify the process of gathering data for training the computer software, all hand signs were performed with the users right hand. The users where also told to have the palm of their hand roughly facing the sensor at a distance of approximately 20 centimeters, as can be seen in Figure 2. These restrictions assured that the Leap Motion sensor would be able to see the hand sign being performed, which means that finger occlusion were to be as rare as possible. The restrictions also substantially reduced the variety of the input data when the same hand signs were performed by different users.

The hand signs that this computer program was trained and tested on are all from the official ASL alphabet, which can be seen in Figure 3. All hand signs from the ASL alphabet only require one hand, and in Figure 3 as in this project they are all performed by the user's right hand. The image is inspired by lifeprint.com[5].

Because of three reasons, limitations of the training and test set had to be made. First of all, similar hand signs were impossible to differentiate because of limitations with the Leap Motion sensor. These hand signs were all close to the

[4] http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/

[5] ASL alphabet charts, http://www.lifeprint.com/asl101/topics/wallpaper1.htm

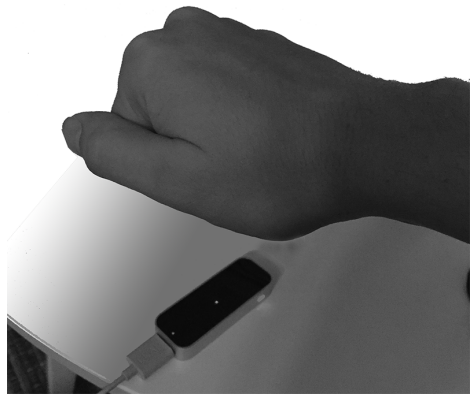


Fig. 2. The test environment containing the Leap Motion sensor and the hand sign corresponding to "A".

shape of a fist, thus often occluded the thumb, namely "A", "E", "M", "N", "S" and "T". But for the interest of testing the computer program, the hand signs "A" and "E" still included in the test and training sets. Secondly, some hand signs had to be removed from the training and test sets because the users had problems preforming them in an uniform manner. These were the hand signs "G", "H", "P" and "Q". Furthermore, the two hand signs "J" and "Z" had to be removed because of their requirement of hand movement to be performed. With the above mentioned hand signs excluded, the training and test sets ended up containing 16 letters(set A), and 9 numbers(set B).

The approach to recognize hand signs was separated into two phases, a training phase and a testing phase. For training and testing, input from 16 test subjects was collected with the Leap Motion sensor. Every one of the test subjects performed one version of each sign that would then be included in the test and training sets. The real value of the performed sign was also saved with the corresponding hand sign data.

2.1 Training

After the training set was collected, the program was to be trained and tested. To get as much results from the testing phase as possible, the training and testing was done on each of the test subjects input data. This means that the program was trained on 15 out of the 16 test subjects, and then tested on the test subjects excluded from the training set. This was done 16 times, thus the program would perform tests on the hand sign data from every test subject. This method is also called cross-validation [8].

The training phase consisted of creating the tree used for searching in the test phase. To create a training set, the hand sign data that would be used for that particular training set was inserted into a 57 dimensional binary tree.

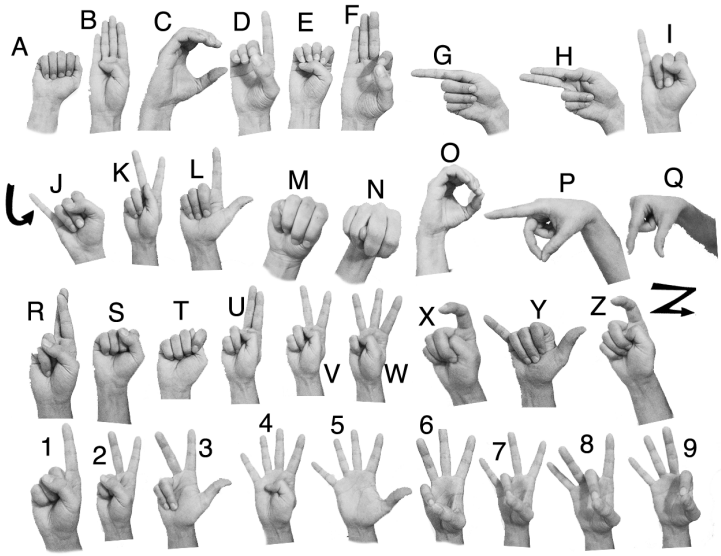


Fig. 3. The American sign language alphabet and numbers from one to nine. Image inspired by lifeprint.com

It takes three dimensions of that tree to describe the direction of one bone in the hand performing the sign. The three dimensions consists of the x, y and z values corresponding to a normalized vector describing the direction of the bone. The direction of the bone was given by the Leap Motion API. Because of the total of 19 bones that is used to model the hand(Figure 4), 57 dimensions are required to represent one hand sign. The image is inspired by the Leap Motion API overview[6].

2.2 Testing

After the 57 dimensional tree was constructed, testing with the excluded set of hand signs was performed using the k-nearest-neighbors algorithm. The kNN algorithm is a classifier that both generally and as in this project is based on the Euclidean distance between instances in a 57 dimensional tree and the instance which is being classified. The points $p = (p_1, p_2, p_3, \ldots, p_n)$ and $q = (q_1, q_2, q_3, \ldots, q_n)$ are two points in a Euclidian n-space, or as in this project, two vectors in a 57 dimensional tree. Then the distance d from p to q is given by equation 1 [7].

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2 + \ldots + (p_n - q_n)^2} \qquad (1)$$

[6] Leap Motion API overview, https://developer.leapmotion.com/documentation/python/devguide/Leap_Overview.html

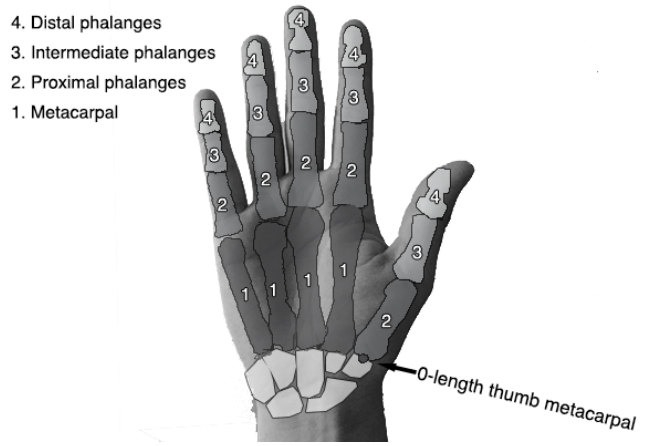


Fig. 4. Bones in the hand that are detected by the Leap Motion sensor. Image inspired by Leap Motion API overview

To enable a kNN classifier to achieve good results, an optimal value for k has to be determined, and there are numerous methods for determining it. Predetermining k to be one or to choose k to be the square root of the number of instances (Duda et. al. p. 175) [9] is a good rule-of-thumb. In this project, the computational time needed to train and classify the entire dataset was measured only in milliseconds, therefore the training and classifying could be done on numerous values for k. The k with the most correct number of correct classifications was then selected.

3 Results

To determine the optimal k for this project, the program was tested ten times using the fold cross-validation with k varying between one and ten. The total correctly classified hand signs for each iteration was compared to determine the k value that achieved the highest number of correct classifications.

The results from testing on set B was only varying by one more correct classified hand sign up to $k = 8$, as can be seen in Figure 5. There were only small differences in the total amount of correct classifications.

Testing on set A led to more conclusive results. The peak value of total correct classifications occurred using $k = 7$, which can be seen in Figure 6. This experimental approach led to defining $k = 7$. For simplicity this k was then used to classify both set A and set B.

The results from testing set A using the 7-nearest-neighbors classifier and cross-validation are presented in Figure 7. The average correct classification of the 16 hand signs in set A was at a rate of 85%. Seven of the sixteen classified hand signs achieved 100% correct classifications.

In Figure 8, the result from testing set B using the 7-nearest-neighbors classifier and cross-validation is presented. The average correct classification of the

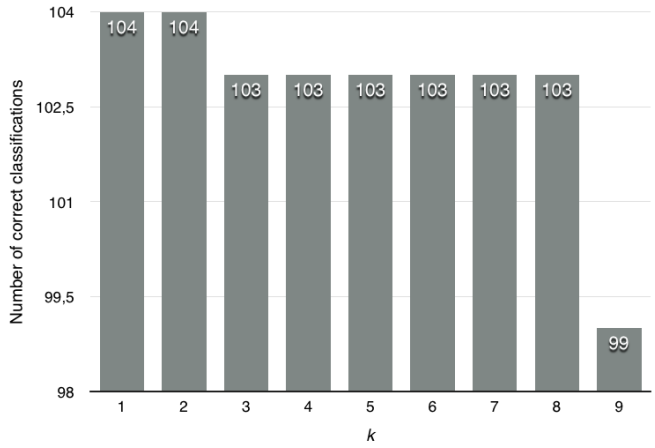


Fig. 5. The total number of correct classifications using cross-validation on set B for each values of k.

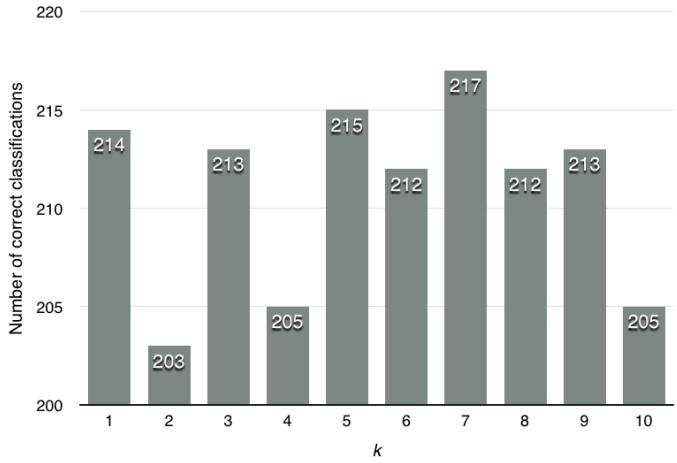


Fig. 6. The total number of correct classifications using cross-validation on set A for each values of k.

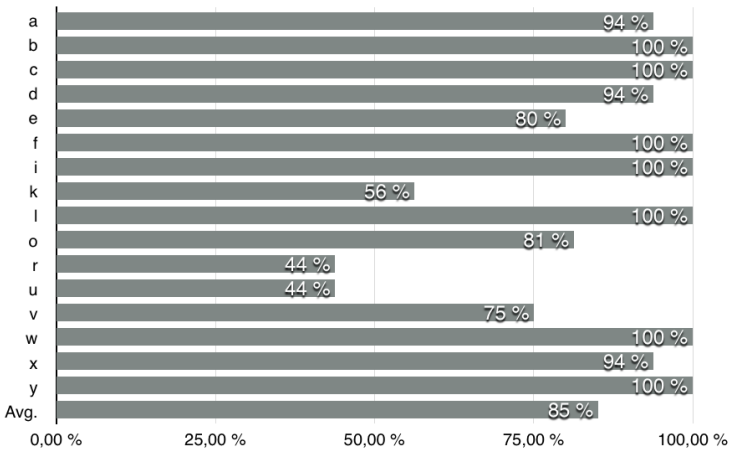


Fig. 7. The results of using cross-validation with a 7NN classifier on set A.

9 hand signs in set B was at a rate of 95.37% a total of 6 out of the 9 classified hand signs achieved 100% correct classification.

The hand signs that were not correctly classified can be seen in Figure 9 for set A. In the figure, the bars in the chart are constructed of the wrong classifications made for each hand sign. For example the hand sign for "U" was miss-classified to "R", "V", and "K". For set B, the hand sign for "1" was miss-classified to "6", and the hand signs for "5" and "8" was miss-classified to "4".

4 Limitations

During the test phase of this project, the method for finding the best performing k should not have been used. This could lead to misleading performance measures for the approach. For this tuning process to be valid, it would have had to be preformed on a separate set of hand signs then the once that was used for testing.

Instead of using the method for finding a good k a predetermined k should have been used. Using $k = 1$ instead of $k = 7$ would have resulted in three less correct classifications for set A(Figure 6), and it would have resulted in one more correct classification for set B(Figure 5).

What also has to be taken into account when evaluating the result from this project is that ten hand signs were removed from the American sign language to create set A. Because set A is not the complete alphabet, the solution cannot be used in any applications needing the complete alphabet.

This solution could only be used in applications that would only use around 16 different hand signs. The hand signs appearances would also have to be different between each other for such an application to be able to function properly.

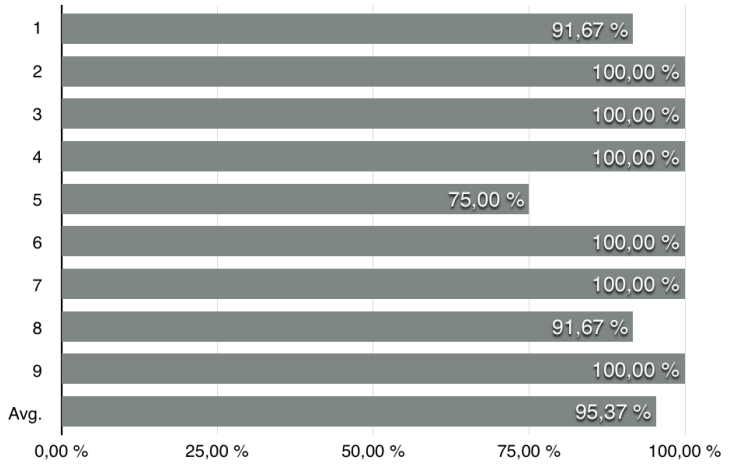


Fig. 8. The results of using k-fold cross-validation with a 7NN classifier on set B.

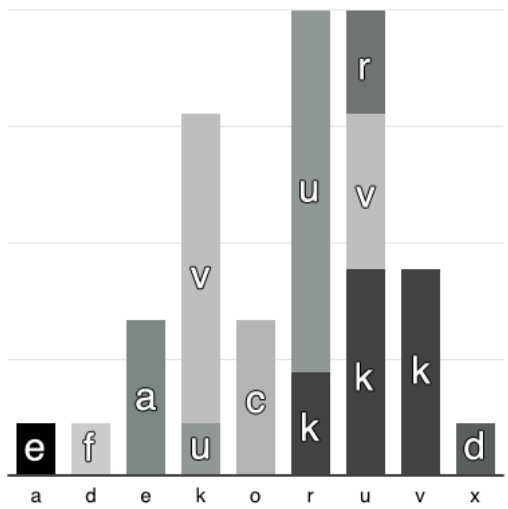


Fig. 9. The miss-classifications that were made by the classifier on set A. For example the classifier miss-classified the hand sign for "R" to "K" and "U".

5 Discussion

The purpose of this project, which has been enabling a computer to read hand signs from the American sign language with the help of a Leap Motion sensor, could be deemed successful. The project correctly classified signs with an average of 85% when classifying the letters from set A and 95.37% when classifying numbers from set B.

What can be seen in Figure 7 is that this approach was able to classify as many as seven signs in set A achieved 100% accuracy, and ten of the 16 signs achieved over a 94% correct classification rate. What also can be seen in Figure 7 is that the signs that did not have as many correct classifications was the signs for "E", "K", "O", "R", "U", and "V". These signs have all in common that other signs are closely related to how to perform them. For example, the signs for "K", "R", "U", and "V" all only uses the index and middle finger in different poses. The signs "A" and "E" that both uses a closed fist, and the signs "C" and "O" that both look like a pinching gesture, are examples where one sign had good performance, while the other sign was sometimes classified as sign it is similar to.

Figure 9 shows that the subset "K", "R", "U", and "V" were often miss-classified as each other. This is also true for the subset "A" and "E". The reason for the miss-classifications could be the Leap Motion sensor's difficulties to handle occlusion, which could have led to misleading input data.

In set B 6 out of 9 signs was classified with 100% accuracy, and the only sign that was classified under 90% was the sign for "5". In set B the miss-classifications for hand sign for "5" was confused with the hand sign corresponding to "4". The difference between "5" and "4" is only the position of the thumb, which can explain why the classifier miss-classified, but not why the classifier correctly classified all hand signs corresponding to "4".

The miss-classifications most definitely could have been lowered by removing variables from the input data. One example could be to remove the angle of the whole hand, thus removing variations in how the test subject's hand were positioned above the Leap Motion sensor.

This approach could be a part in the search for more natural ways of interacting with computers. With its high correct classification rate, using the approach on signs with widely different appearances should render in a 100% correct classification rate. Then the approach could be used in applikations where the user would not have the ability to focus on the interaction with the computer, for example driving a car.

The approach can aid individuals with disabilities interacting with a computer, for example if an individual would have problem in reaching all the keys on a keyboard but is able to perform sign language. The approach is can recognize hand signs without similar appearances, which would be suitable for some sort of navigation. However, for the approach to provide a full alphabetical input, it would need better correct classification rates on hand signs that are similar to each other.

To improve performance for hand signs with similar appearances, the approach could be extended with a weight system. The weight system would predict what the user was typing (similar to autocorrect solutions in smart phones) and give letters which would fit into possible words a higher possibility to be classified.

References

[1] Bolt, R.A.: "put-that-there": Voice and gesture at the graphics interface. SIGGRAPH Comput. Graph. **14**(3) (July 1980) 262–270
[2] Pavlovic, V., Sharma, R., Huang, T.: Visual interpretation of hand gestures for human-computer interaction: a review. Pattern Analysis and Machine Intelligence, IEEE Transactions on **19**(7) (Jul 1997) 677–695
[3] Cui, Y., Weng, J.: Appearance-based hand sign recognition from intensity image sequences. Computer Vision and Image Understanding **78**(2) (2000) 157 – 176
[4] Murthy, G., Jadon, R.S.: Hand gesture recognition using neural networks. In: Advance Computing Conference (IACC), 2010 IEEE 2nd International. (Feb 2010) 134–138
[5] Iwai, Y., Hata, T., Yachida, M.: Gesture recognition based on subspace method and hidden markov model. In: Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on. Volume 2. (Sep 1997) 960–966 vol.2
[6] Naoum, R., O, H.H., J, S.: Development of a new arabic sign language recognition using k nearest neighbor algorithm. Journal of Emerging Trends in Computing and Information Sciences **3**(8) (2012)
[7] Peterson, L.E.: K-nearest neighbor. Scholarpedia **4**(2) (2009) 1883
[8] Kohavi, R., et al.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: IJCAI. Volume 14. (1995) 1137–1145
[9] Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. John Wiley & Sons (2012)

Gazing Habits While Typing in Regard to Field of Education

Michael Mellquist

Department of Computing Science
Umeå University, Sweden
mikemellquist@gmail.com

Abstract. This paper examines whether there is a difference in gazing habits while typing in regard to field of education. The study was to measure how often and how long students looked at the keyboard while writing. Two groups of students took part in the study, one group of economy students and one group of interaction design students. A text in Swedish was read aloud to them while they simultaneously wrote what they heard. The test subjects were filmed while they wrote and the video was later analysed to find out how many times and how long the test subjects looked at the keyboard versus the screen. The result between the groups with different educational background showed that the interaction design students looked less at the keyboard than the economy students.

1 Introduction

People use computers for different purposes, some just browse the Internet, others write in documents, and some use it as a tool for programming. The fact that everyone uses computers differently create different typing habits [1]. The purpose of this paper is to find out whether people's tendencies to look at the screen while typing varies depending on their field of education, which in this case is economy and computer science. A gender perspective will also be taken into account in the result, though is not the main topic in this paper.

In this study the eye activity of the test subjects was recorded on camera and later analysed to show whether the students' focus was on the screen or the keyboard. The difference between the previous studies [1, 2, 3] and this study was that this study focused on how/if field of education reflects on typing and eye movement habits instead of the productivity of various typing habits and styles of typing.

A study in text production between writers who look at the screen while writing and writers who look at the keyboard [2] shows that monitor gazers type significantly faster. Computer science and economy students write different kinds of reports in their studies which leads to different writing habits [1]. Our study uses these two facts to find out whether there is a difference between the groups' gazing habits.

Economy students tend to write more and longer papers than interaction design students. Economy students' papers are usually longer and more descriptive due to the analytical solutions to the assignments. Economy students are

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 69–76, January 2015.

taught to find several different solutions to a problem even if one is more correct. Economy students read through a lot of qualitative and quantitative studies and draw conclusions from them which often tend to be long[1].

Interaction design students learn different programming languages in their education programme. They take multiple courses and do programming assignments, thus learning the programming languages. The assignments often contain a lot of code and also a report where the student writes about their solution. Interaction design students' assignments contain fewer characters but take a long time to write because of the problem solving[2].

The result of this study can be used in future studies and bring more insight into how different fields of study affect simple tasks; like typing on a computer. The research question in this study was whether field of education affects if students focus on the screen or on the keyboard. This paper is organized as follows. Section 2 describes earlier work in the area. Section 3 describes the method and equipment used in the study. Section 4 shows the result of the study. Section 5 contains a conclusion and a discussion.

2 Earlier Work

In this section earlier work will be presented.

In 1984 a study about typing speeds in regard to typing habits was performed [1]. The test was designed so that 190 test subjects wrote 5 sentences of varying difficulty during a short period of time. The study showed that typists that shared the same typing habits differed only ± 15 words per minute and significant differences in habits occur at ± 30 words per minute.

A study about text production differences was performed [2] in 2009 where the productivity between people who looked at the screen or on the keyboard was measured. The study used eye-tracking equipment and key-stroke-loggers to document exactly where the test subjects looked and when they were typing. The result from this test was that monitor gazers wrote significantly faster than those who also looked at the keyboard.

3 Method

In this section the method of the study and the equipment used will be presented and explained.

3.1 The Study

The method used was a study with 12 people. Half of the test subjects were economy students and the other half interaction design students. Half of the two groups were females and the other half males. The test subjects performed the

[1] Interview with economy student at Umeå University, Sweden
[2] Interview with interaction design student at Umeå University, Sweden

test individually. They were asked to write in a document on a laptop what was read aloud to them during a period of 60 seconds. Because the test subjects could be nervous or stressed as the test started a method was used to try to get the test subjects in the same state of mind. The method was to let the test subjects start the experiment but the first 30 seconds of data was omitted *without their knowledge*. The video from the tests were later analysed by counting the number of times and total amount of time they looked at the keyboard. If only the number of times the test subject looked at the keyboard was counted, the test result could be misleading as a test subject could just look at the keyboard during the whole test. The result from that test would then be interpreted as the test subject looking at the keyboard once. The test environment was a place where many students typically write their papers, in an open loud environment with people around. The text which the test subjects wrote down was in Swedish due to Swedish being their mother tongue, this to reduce the margin of error so that the students' knowledge in English would not be a factor to take in account. The following text was written for the purpose of this study and was what the test subjects were to write down:

Nu var året 1922, förmodligen ett år som inte skulle innebära något nytt, utan precis som det innan, och alla innan dem. Han tog ett sista bloss och slängde cigarren över räcket. Han justerade hatten som av någon anledning inte satt som den brukade. Det var många minusgrader ute, det kändes som en av de kallaste nyårsaftnar han varit med om. Snön på balkongen var djup och gav ifrån sig ett skarpt och kallt ljud med varje steg han tog, som den får vid runt trettio minusgrader. De andra gästerna såg så glada ut när de gick omkring och skålade med sina champagneglas och kramade om varann. De hade ingen aning om vad som väntade. Kvällen hade varit trevlig med god mat och dryck, värdarna hade slagit på stort och köpt in det allra finaste och exklusiva som fanns att tillgå. Hans bordsdam verkade intresserad av honom men han hade bara tittat bort när deras blickar mötts. Nu var hon på väg bort mot honom med ett glas i vardera hand. Han tittade på klockan, det var snart dags att utföra jobbet.

Translated into English:

The year was now 1922, a year which would probably be as meaningless as all the years before. He took a last puff at his cigar before he threw it over the railing. He adjusted his hat, which did not sit as it used to, but for some reason it still felt misplaced. It was many degrees below zero outside; it felt like one of the coldest New Years Eve's he had experienced. The snow on the balcony was deep and gave off a sharp and icy sound with each step he took, as it does at around thirty degrees below zero. The other guests looked so happy as they mingled and toasted with their champagne glasses and hugged each other. They had no idea of what was to come. The evening had been pleasant with appetizing food and drinks, the hosts had spared no expenses and bought the most exclusive food money could buy. His dinner partner seemed interested in him, but he had only looked away when their eyes met. She was now on her way over with a glass of champagne in each hand. He glanced at his watch; it was soon time to carry out the task.

Because peoples typing speeds differ, someone could have finished writing the whole text in under a minute. The test would then be rendered unusable, hence the long text.

3.2 Equipment

The equipment used was a camera, a text editor, a stopwatch, and a computer. The camera was a "FaceTime HD-camera" with a resolution of 720 pixels, it is the laptop camera in the MacBook Pro and by analysing the videos at a later stage we were able to distinguish whether the test subject looked at the screen or keyboard when writing. The computer used was a MacBook Pro of a late 2013 model. An alternate keyboard was connected via USB if the test subject felt more comfortable with it. The text editor was Microsoft Word 2011 for Mac. The stop watch was a standard stopwatch application on an Android phone.

4 Result

In this section the results of the study will be presented. The results from the two groups of students with different educational backgrounds and the difference between the first and second half of the tests will be compared and analysed.

As seen in table 1 the three male interaction design test subjects only looked down a total of 3.4% during the test. Whereas the three male economy students looked at the keyboard a total of 69.4% of the test. The female test subjects however showed an opposite result, as the female interaction design students looked at the keyboard a total of 24.5% of the time compared to the female economy students who looked down a total of 8.5%. One female and one male interaction design student did not look at the keyboard at all.

The definitive result shows us that the economy students look at the keyboard 78% of the time and the interaction students look 27.9% of the time.

	Economy students	**ID students**
Male 1	92.3	4.6
Male 2	58.6	5.6
Male 3	57.4	0
Female 1	11.6	0
Female 3	7.1	28.5
Female 3	6.9	45
Total average	78	27.9

Table 1. This table shows the amount of time (in percent) the test subjects looked down at the keyboard.

The purpose of omitting the first half of the test was to reduce unrelated factors that may have affected the outcome of the test. The results show that this method worked as predicted and that the test subjects focused more on the

	Economy students	**ID students**
Male	69.43	3.4
Female	8.53	24.5

Table 2. This table shows the total amount of time (in percent) the test subjects looked down at the keyboard.

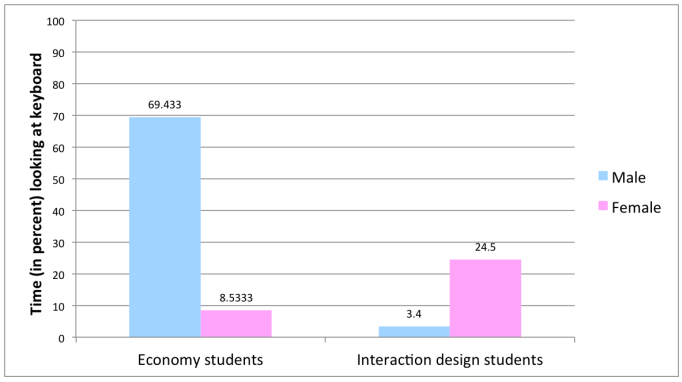


Fig. 1. Bar chart showing the average result of the test subjects, categorized in field of education.

	Economy students	**ID students**
Male 1	2	1
Male 2	9	3
Male 3	12	0
Female 1	7	0
Female 2	3	16
Female 3	2	12

Table 3. This table shows the number of times the test subjects looked down at the keyboard during the second half of the test.

	Economy students	**ID students**
Male	3.633	-2.20
Female	-9.44	-6.13

Table 4. This table shows the difference in time between the first and second half of the test (in percent) that the test subjects looked down at the keyboard.

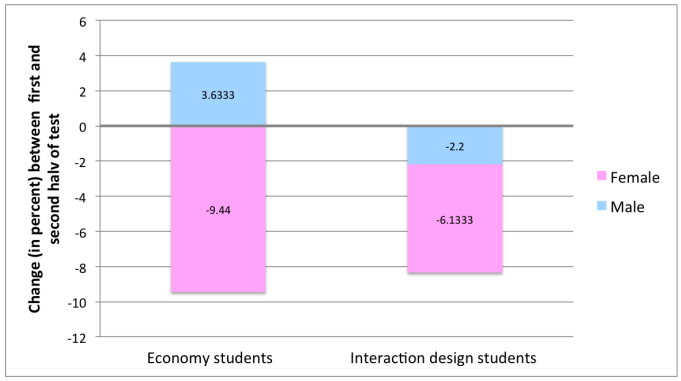


Fig. 2. Bar chart showing the change in result of the first and second half of the test. The charts are categorized in field of education.

screen the second half of the test. The average overall change in percentage was a 3.5% decrease in focus on the keyboard in the second half of the test. The test subjects were aware of the purpose of the test before they took part in the study.

4.1 Difference Between Education

Even though the female economy students look less at the keyboard than female interaction design students the average time looking at the keyboard for the economy students is higher than the interaction design students. This is due to the males showing such big differences in average time looking at the keyboard.

5 Limitations

In this section the limitations will be presented. From this section, one may learn what not to do in future studies.

Due to different keyboard layouts, sizes, etc. the test subjects were used to another type of keyboard, another keyboard was therefore brought as an alternative in the hope that it would minimize the unfamiliarity. After a few tests it stood clear that keyboards, despite the alternative keyboard, were different than what several test subjects were used to. This has most definitely affected the results, as the test subjects could not write to their full ability. One way to prevent this problem from happening in future tests is to let the test subjects use their own laptop computers and to mount a camera on their screen. When the test subjects were finished with the test they could then send what they had written to the test leader.

Another possible source of error was when the text was read aloud to the test subjects. A few times the text was read to quickly so that the test subjects would ask what had been said. The text contained a few words that were difficult to spell on purpose. The text also contained numbers to signify a year, this to make the test subjects look at the keyboard at least once, which everyone did

when they wrote the year "1922". This had no impact on the test, as it was in the first half of the test. The test subjects also looked at the keyboards when they misspelled a word and pressed the backspace button to see what characters they deleted, as in the study about text production processes [2].

A Tobii Eye-X eye-tracker was meant to be used in the study but was excluded due to a misunderstanding in the devices features. Using the eye-tracker would have rendered a more exact result in where the test subjects looked. The eye-tracker was bought to the "Tillämpad Elektronik och Fysik"-institution and we were allowed to use the device in the study. The device worked as a mouse in the way that the cursor moved where the user looked. The problem was that the Tobii company would not release an application which would allow the eye-tracker to record where the user had looked. To do so one would have to buy another eye-tracker, which was 20 times more expensive.

6 Discussion

In this section a discussion about the method and results will be presented. From this section one may get a better understanding of the results. The research question was whether field of education affects if students focus on the screen or on the keyboard. The results show that there was a difference in the groups with different educational background; interaction design students looked less at the keyboard. Whether the result is valid or if there were limitations which obstructed the result will be discussed in the conclusions section.

During the test the test subjects did not wear any equipment to record where the eyes focused, which may have resulted in a more comfortable and natural test for the test subject and therefore a more successful study. The fact that the three male interaction design students looked at the keyboard 3.4% on average of the test can be a result of their extensive use of computers. In addition to using computers in school work, interaction design students are often interested in computers in general and often use them during their free time for many hours every day[3].

That a person looks at the keyboard while typing on a computer could mean that the person is less used to writing and also needs more working-memory resources [4] to perform the task of writing. This would also mean that it takes longer time for the test subject to write the text, and also alternate between looking at the screen, formulating what to write, and writing [2, 5].

6.1 Analysis of Omitting the First Half

As mentioned above, omitting the first half of the test was a good method to use as the overall result showed a 3.5% decrease in looking at the keyboard during the second half of the test. All the participants except the male economy students showed a decrease in looking at the screen during the second half of the

[3] Interview with a interaction design student at Umeå University, Sweden

test. When the test subjects sat down to participate their first reactions were do ask questions and to say how nervous they were before the test.

6.2 Conclusion

Is there a difference? Does field of education affect gazing habits while typing? From this study we have determined that there is a difference between students with different educational background. This difference might be due the interaction design students' interests in computers in their free time, it does not have to mean that economy students need to practise writing more. Though the result was as expected one must keep in mind that there were only 12 participants in the study. There was a difference between men studying economics and men studying interaction design, although the result was the opposite for the women. If this were a larger study with many more test subjects the result would probably be different. Because half of the test was interaction design students, and half of them were men and the other half women, there were only 3 people of each sex. 3 people are not enough to make this result statistically correct.

References

[1] West, L.J., Sabban, Y.: Hierarchy of stroking habits at the typewriter. Journal of Applied Psychology **67**(3) (1982) 370
[2] Johansson, R., Wengelin, Å., Johansson, V., Holmqvist, K.: Looking at the keyboard or the monitor: relationship with text production processes. In: Reading and Writing. (2009) 835–851
[3] Gentner, D.R.: Expertise in typewriting. The nature of expertise (1988) 1–21
[4] Kellogg, R.T.: A model of working memory in writing. (1996)
[5] Olive, T., Kellogg, R.T.: Concurrent activation of high-and low-level production processes in written composition. Memory & Cognition **30**(4) (2002) 594–600

Determining Handedness Through Keystroke Dynamics Using Hidden Markov Models

Karl Petersson

Department of Computing Science
Umeå University, Sweden
id10kpn@cs.umu.se

Abstract. We propose a method for determining handedness through keystroke dynamics by using a classifier based on Hidden Markov Models. Data was collected through a web application, where users were instructed to type sentences on a keyboard while the application recorded a keystroke template for each individual. The template for each individual was mapped to observation sequences and used as input to the HMM classifier. We only used one feature for the classification: key hold time, i.e. how long a particular key is held. Two HMMs were implemented for determining right- and left-handedness and they were trained using the Baum Welch algorithm. The resulting classifier achieved a successrate of 66% for determining handedness on novel sequences of keystrokes.

1 Introduction

Keystroke dynamics is a behavioural biometric which models keyboard typing patterns of an individual. It considers features such as key hold time and timing between keystrokes [1] to construct keystroke signatures for individuals. Through keystroke dynamics, machine learning algorithms can become proficient at differentiating between individuals based on how they type [2, 1]. In the context of user authentication, such classifiers have been successfully implemented using a variety of algorithms: for example, Neural Networks [3] and Hidden Markov Models (HMMs) [2, 4]. HMMs work well in this context because of their ability to handle series of events and stochastic processess. A set of keystrokes within a time interval can be viewed as a series of non-deterministic events, in the sense that they consist of seemingly random continous values ordered by time. These aspects suggests that HMMs are suitable for modeling keystroke patterns of individuals [2].

The concept of keystroke dynamics infers an assumption that there are neurophysiological factors that make typing patterns unique, similiar to written signatures, which have been observed in previous work [1]. The Kestroke Dynamics Template (KDT) of an individual refers to the constructed keystroke signature for that individual. Attempts to extract and use information from the KDT of an individual in order to recognize physical traits have been made in [5], where a classifier built using Support Vector Machines was shown to produce promising results when trying to determine handedness, which refers to whether an individual is right- or left-hand dominant.

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 77–88, January 2015.

In this paper we use HMMs to classify handedness of individuals through keystroke dynamics using a single feature, namely key hold time (how long a key is pressed down). We choose this feature based on two premises:

1. We believe that there is a difference with respect to key hold time when typing letters that correspond to keys lying either on the left or right side of the keyboard, depending on whether the person typing is left- or right-handed.
2. According to [2], key hold time is the most efficient feature of Keystroke Dynamics.

The proposed method for classification consists of three steps: (i) mapping the KDTs to HMM parameters, (ii) training the classifier based on those parameters using the Baum Welch algorithm and (iii) testing whether the trained classifier can determine handedness on data not included in the training set.

This paper is organized as follows. In Section 2 we provide a brief introduction to HMMs, in Section 3 we present the proposed method for data collection followed by an explanation of how we implemented and trained the HMM classifier, in Section 4 we present the results obtained from testing the classifier and in Section 5 we discuss conclusions, limitations and future work.

2 Hidden Markov Models

An HMM is a finite state machine which produces sequences of observation symbols by moving between a finite number of hidden states. The system determines the next state to move to by its state transition matrix, which assigns probabilites to trainsitions between any two states. When the model arrives at a state, it emitts an observation symbol, determined from the emission distribution of that state, and then proceeds to a next state(which could be the same state). The system is generative in that it can emit a sequence of any number of observations. In some cases, the idea of an end state is considered, where this state has zero probability of transitioning to any other state. The system begins at an initial starting state (which could be any hidden state) and for any particular number of steps, it will have produced a sequence of observation symbols [6].

Following the formal definition of an HMM in [2], an HMM is a five-tuple $\lambda = (S, V, A, B, \pi)$, where $S = \{s_1, s_2, ..., s_N\}$ denotes the set of hidden states, where N denotes the number of states, $V = \{V_1, V_2, ..., V_K\}$ denotes the set of observation symbols, $A = \{a_{ij}\}$ denotes the state transition probability matrix, where a_{ij} denotes the probability for state i to transition to state j, $B = \{b_j(o_t)\}$ denotes the emission probability distribution where $b_j(o_t)$ denotes the probability of observing symbol o_t in state j, and $\pi = \{\pi_i\}$ denotes the initial state matrix, where π_i represents the initial state probability of state i. An observation sequence is denoted by $O = \{o_1, o_2, ..., o_T\}$, where T denotes the number of observations in that sequence.

An HMM can either be discrete or continous, determined by the emission probability distribution B, meaning that B either has a discrete or continous

probability density function which describes the likelihood of an emission taking a particular value.

Figure 1 shows an HMM with an initial state START and two hidden states 1 and 2. The start state is not part of the state set S, it is only used for the purpose of illustration. The transition probabilities are written down on the arrows between the states. For example, in Figure 1, moving from START to 1 has a probability of 1. The transition between two states, for example, 1 and 2, is denoted by $a_{12} = 5/6$. Note that $\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \leq i \leq N$.

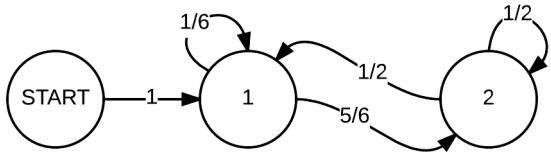


Fig. 1: Example HMM with start state START and S = $\{1, 2\}$

When testing an HMM, we compute $P(O|\lambda)$ which is the probability for the sequence O to have been generated by model λ. This is done using the foward algorithm. We denote $\alpha_t(j)$ to represent the probability of a partial observation sequence $\{o_1, o_2, ..., o_t\}$ in state j at time t. For estimating $\alpha_t(j)$ we use the forward procedure.

Following the explanation in [2], first we initialize $\alpha_1(i) = \pi_i b_i(o_1)$. Next, we compute

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right) b_j(o_{t+1}).$$

We can then calculate $P(O|\lambda)$ by the formula

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

where the following conditions hold: $1 \leq i \leq N, \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1$.

In the context of classification, one HMM per class is constructed and trained. In our case, we use two HMMs, representing right- and left-handedness. To classify a particular sequence, we can compute $\mathcal{L}(O|\lambda)$ for both models λ_1 and λ_2, where $\mathcal{L} = logP(O|\lambda)$ is the log-likelihood of observing sequence O.

However, the log-likelihood value $\mathcal{L}(O|\lambda)$ does not hold much information on its own, since it can vary a lot depending on, for instance, the length of observations and on the model itself. Hence, we cannot compare $\mathcal{L}(O|\lambda_1)$ and $\mathcal{L}(O|\lambda_2)$ directly. Instead, we can examine the ratio $\Lambda = \mathcal{L}(O|\lambda_1)/\mathcal{L}(O|\lambda_2)$. Ratios obtained for sequences between the two classes can then be compared

to find a threshold θ that separates them, such that the system classifies the sequence as belonging to one of the classes based on a condition, for example, $\Lambda < \theta \Rightarrow righthanded$.

HMMs are trained by introducing a set of sequences, whereupon algorithms such as Baum Welch are used to adjust the transition probability matrix, as well as the emission distribution, in order to increase the probability for the HMM to produce the input sequences (training examples).

Baum Welch is an iterative algorithm that uses expectation-maximization to estimate the parameters of an HMM, given a set of observation sequences. It is based on the two functions $\alpha_i(t)$ and $\beta_t(i)$, where $\beta_t(i)$ is the backward probability calculated by the backward procedure, following the definition in [2]:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N,$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad 1 \leq j \leq N, \quad t = T-1, T-2, ..., 1$$

where $\beta_t(j)$ is the probability of the partial observation sequence $o_{t+1}, o_{t+2}, ..., o_T$ and state j at time t. Once $\beta_t(i)$ and $\alpha_t(i)$ is computed, they are used to weigh the contributions of all observations O_t to the HMM according to reestimation formulas, which can be found in [7], such that each observation O_t contributes to all HMM parameters [8].

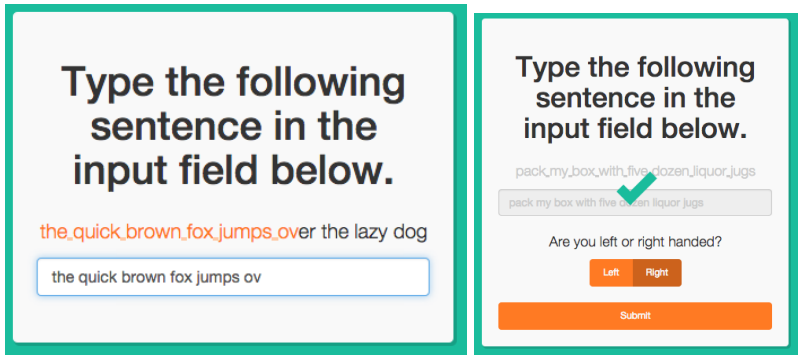
3 Method

3.1 Data Collection

Data was collected via a web-application designed as a test, shown in Figure 2. Participants were instructed to type three different sentences, where each sentence was a pangram which is a sentence containing all letters in the English alphabet. The idea behind using pangrams was to get a broad dataset in terms of number of different keys pressed. The three sentences were: "a quick brown fox jumps over the lazy dog", "grumpy wizards make toxic brew for the evil queen and jack" and "pack my box with five dozen liquor jugs".

Each sentence could only be completed by typing the letters in correct order. At each point at which a new subset of the sentence was completed, the participant received visual feedback indicating the progress by coloring completed letters. Any errors in terms of typing incorrect letters or pressing other keys stopped the progress - i.e. the coloring of letters - until the next correct letter was typed. The participant was not required to erase incorrectly typed letters. The test ended when all letters in each sentence had been typed, whereupon the participant was asked if he or she was right- or left-handed. The participant then had the choice to submit their results.

During the test, for each key pressed and released by the participant, the software logged the Unix time of when the event occurred which is defined as the time passed in milliseconds since january 1st 1970 (referred to as timestamp)



(a) Typing sentences (b) Submitting results

Fig. 2: Screenshots of data-collection application

and which character the key represented. If the participant pressed a key that did not correlate to the next letter at any point in the sentence, the event was also logged as an error. Each logged error contained additional information regarding to what key should have been pressed instead.

Collected data for an individual was stored in a database as a list of objects representing key events for each sentence. Each set of lists hence constituted the KDT of a particular individual. A link to the application was openly distributed through various web channels[1][2]. A total of 56 people completed the test, of which 44 were right-handed and 12 left-handed.

3.2 Implementation

The classifier was implemented in Python using the library ghmm[3], which contains an implementation of HMMs including the Baum Welch and forward algorithms. Two HMMs were constructed; one for modeling left-handedness, denoted by λ_L, and one for right-handedness, denoted by λ_R. The number of states for both models was set to 2. This descision was made based on experimentation, where we found that any additional states converged to having transition probabilites close to 0 from other states. The state transition matrices were initialized to

$$A_{\lambda_L} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}, \quad A_{\lambda_R} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}.$$

Since we did not have an hypothesis concerning what the transition matrices would converge to when training the models, transitions were arbitrarily set to 0.5. For both models, the probability emission distributions B_n for each state s_n was set to be a gaussian distribution so that $B_n \sim N(\mu, \sigma)$ where $\mu = 1.0$ and

[1] facebook group for students in interaction design
[2] forum for machine learning, www.reddit.com/r/machineLearning
[3] General Hidden Markov Model library (GHMM), http://ghmm.sourceforge.net/

$\sigma^2 = 0.05$. These parameters were chosen based on the fact that the training examples were normalized around 1 to adjust for different overall typing speed between individuals (explained in Section 3.2). The initial state matrix π was set to $[0.5, 0.5]$ for both models, with the same argument as the one considering intializing transition matrices.

3.3 Training the Classifier

KDTs of individuals can be mapped to sequences of observation symbols in the HMMs by modeling them as vectors of n-tuples, where n is determined by the number of Keystroke Dynamics features used. Since we only considered key hold time, other features were excluded. Keystrokes representing the space-character was ignored, since when pressing space participants may pause and think about what to type next [2]. Thus for each KDT, we had a total of 115 keystroke events representing the three sentences written in the experiment.

The recorded KDT of an individual can be expressed as sets of events of the form $S_n = \{(t_1, x_1, y_1), (t_2, x_2, y_2), ..., (t_m, x_m, y_m)\}$ where S_n is the set corresponding to sentence n in the experiment, t denotes timestamp for a single keystroke event, x denotes key hold time, y denotes keycode and m denotes the number of events.

Training examples were constructed by first concatenating the sentences, producing set $A = S_1 \cup S_2 \cup S_3$, then extracting key hold time x for all keystroke events in A and ordering them by the timestamp t at which they were logged. Key hold time for a single keystroke was given by $x_m = KR_m - KP_m$, where KR_m denotes the timestamp of when the key was released, and KP_m denotes the timestamp of when the key was pressed. Ordering the events by timestamp ensures that they follow the order of the letters in the sentences written in the experiment. A vector $\boldsymbol{v}^T = [x_1, x_2, ..., x_n]$ could then be produced for each individual, consisting of a series of continous key hold time values, which constituted the training examples. These vectors were used as input to the HMMs, representing sequences of observation symbols such that $\boldsymbol{v}_i = O_i$.

Some training examples were identified as outliers for having very high variance, meaning that they contained key hold time values that were much higher than the mean for that example. These outliers were excluded from the training sets as well as the test sets, according to the assumption that they would introduce too much noise with respect to the low sample size. A total of 3 right-handed and 1 left-handed examples were excluded based on the above mentioned premises, resulting in a final set of 41 right-handed and 11 left-handed examples.

The two HMMs were trained each using either a set of right- or left-handed examples. The two training sets consisted of 75% of the total number of samples belonging to that group, meaning 31 right-handed and 8 left-handed examples. Training examples were normalized by their mean hold time to adjust for typing speed bias between examples, according to the formula below:

$$\boldsymbol{v}_i = \frac{\boldsymbol{v}_i}{\bar{\boldsymbol{v}}}, \quad \bar{\boldsymbol{v}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{v}_j.$$

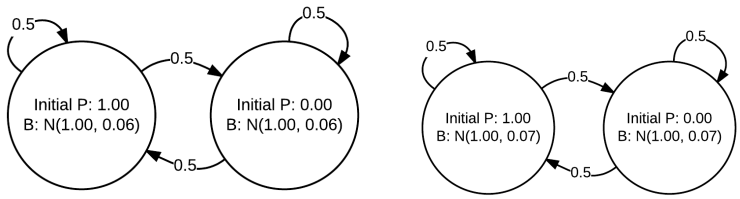
The Baum Welch algorithm was used to train the two models over 100 iterations for each sequence. Additional examples not included in the training set were then used for prediction and testing over the two HMMs. A single run of the algorithm produced a log-likelihood ratio value for each test sequence, calculated by the formula $\Lambda_k = \mathcal{L}(\boldsymbol{v}_k|\lambda_L)/\mathcal{L}(\boldsymbol{v}_k|\lambda_R)$, where $\mathcal{L}(\boldsymbol{v}_k|\lambda)$ denotes the log-likelihood for sequence $\boldsymbol{v}_k$ to have been genereated by model λ.

A simluation was run over the course of 5000 iterations, where the data was shuffled between iterations, so that for each run the training sets and test sets consisted of different sequences. Each iteration, the HMMs were initialized, trained and tested. A log-likelihood threshhold $\theta = 1.26$ was chosen based for the classifier to assert whether a sequence belonged to a particular model.

This value was chosen based on examining ratio values outputted by both HMMs, depending on which type of sequence they received as input. If $\Lambda_k \geq \theta$ for a sequence $\boldsymbol{v}_k$, the system classified the sequence as left-handed. Consequently, if $\Lambda_k < \theta$, the system classified it as right-handed. Over the course of 5000 iterations, the total number of correct and incorrect guesses was recorded.

4 Results

Figure 3 shows the trained HMMs and their parameters after an example run of the program. Both models consistently converged to having state transitions of [0.5, 0.5] after training between runs. The models can only be discerned by the variance in their emission distributions; $\sigma^2_{\lambda_L}$ consistently converged to 0.06, and $\sigma^2_{\lambda_R}$ to 0.07, where σ^2_λ denotes the variance of the emission distributions in all states for model λ. For both models, the emission distributions retained a mean value of $\mu = 1.0$ after training.



(a) Left-hand HMM (b) Right-hand HMM

Fig. 3: Resulting HMMs

Table 1 shows the obtained log-likelihood ratios for test sequences in an example run of the program. *Type* denotes whether the sequence derives from left or right-hand data.

sequence	**type**	Λ
1	Left	1.24036404876
2	Left	1.2908747397
3	Left	1.3585111138
1	Right	1.20462112535
1	Right	1.13497062357
2	Right	1.11040876502
3	Right	1.22675071417
4	Right	1.07064931066
5	Right	1.22009969051
6	Right	1.07063851172
7	Right	1.31815481743
8	Right	1.14431040707
9	Right	1.38663228569
10	Right	1.13681625805

Table 1: Example run log-likelihood values

After performing 5000 runs of the program, the values in Table 2 were achieved. *True right* and *true left* describes how many sequences were correctly classified as right- and left-handed, while *true right ratio* and *true left ratio* describes the ratio between correctly and incorrectly classified sequences. The combined ratios correlates to a total successrate of 66%.

true right ratio	0.6568
true left ratio	0.6685
true right	36125.0
true left	10028.0
false right	4972.0
false left	18875.0

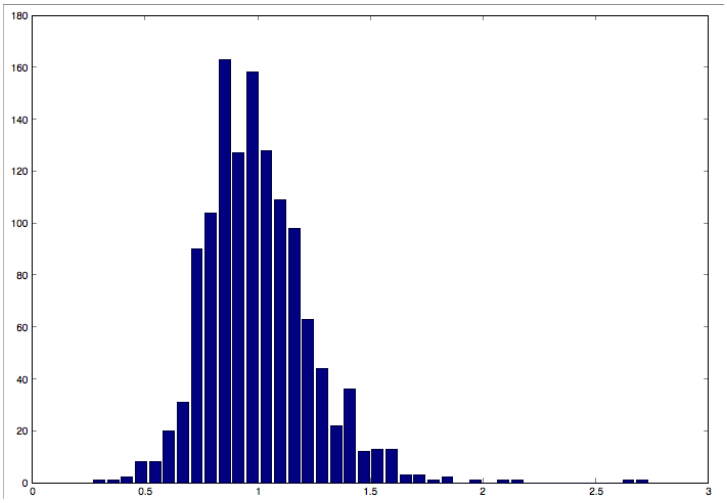
Table 2: Successrates

5 Discussion

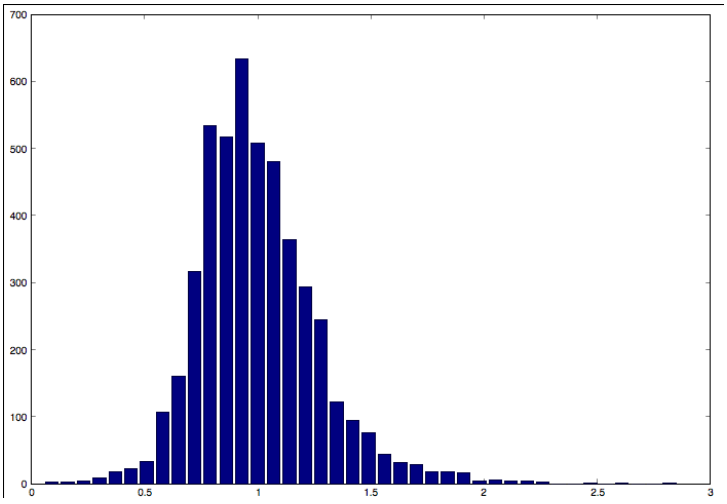
5.1 Conclusions

The results show that between the two groups of left and right-handed individuals in the data set, there is a slight difference in how those groups type on a keyboard with respect to key hold time. However, the sample size is not large enough to dismiss the possibility that we achieved a successrate of 66% merely due to a slight difference in variance between the groups, which may or may not depend on handedness. Indeed, the only discernable difference between the groups with respect to key hold time seems to be that the samples come from slightly different distributions. This would mean that the method of using HMMs could easily be substituted by simply examining the mean and variance of the two groups.

Figure 4 shows a histogram of all sequences across the two groups, from which we can observe that the distributions between the groups seems to be similiar. However, performing a F-test of equal variance shows that the hypothesis $\sigma_L^2 = \sigma_R^2$ can be rejected with a significance level of $\alpha = 0.01$. This suggests that the two groups might derive from different distributions. One interpretation of this is that there is an overall difference in typing speed between left and right-handed individuals with respect to key hold time.



(a) Left-hand sequences



(b) Right-hand sequences

Fig. 4: Histograms of key hold time

Examining either of the two models, we see that both states for a particular model had equal emission distribution parameters μ and σ^2. This suggests that the models can be reduced to only one state without impairing them, which further indicates that each HMM merely modeled the distribution of the sequences for the corresponding group. Due to the similiarity in the resulting HMMs, we conclude that the classifier could not separate left-hand sequences from right-hand sequences by any other factor than the mean variance in key hold time between the groups.

The achieved successrate is lower than the results presented in [5], where a successrate of 75% was achieved when classifying handedness. There are some key differences between our method and theirs, the first one being that they used Support Vector Machines as their method of classification. Secondly, they used the fact that some of the sentences were recorded where the participant only typed with their dominant hand. They also used a much larger data set, consisting of 11000 sentences typed by 110 participants. Finally, they used a different set of keystroke dynamics features, namely timings between keys being pressed and released.

5.2 Limitations

In the study, we used pangrams as sentences to be typed by participants. The motivation for using pangrams was to capture a broad set of different keys pressed. These sentences are not commonly typed sentences, which coupled with the fact that the participants did not have English as their primary language, may have introduced more errors. However, since we disregarded incorrect keypresses and only considered key hold time, not pauses between letters, we assumed these errors would not significantly affect the results.

A problem associated with training HMMs using reestimation methods such as Baum Welch is that because sequences are finite, there is often not enough occurences of different events to give proper estimates of the model parameters [9]. There are solutions to this problem, one being to increase the number of training sequences. Another solution is to decrease the number of states. Since we had few left-hand training sequences, these problems might have come into effect regarding the parameters of the model λ_L.

Another limitation is that we did not used supervised tests. If we had done this, according to the method of data collection used in [5], we might have achieved a data set consisting of less errors. It is possible that, for instance, a left-handed individual submitted his or hers results as a right-handed example, and vice versa. Also, it is possible that some individuals did not type naturally or with two hands during the test.

We cannot guarantee that the classifier was implemented correctly. The library we used(ghmm) has very little documentation, which forced us to many times rely on trial and error during the implementation stage. Using other libraries instead could have led to different results and a more reliable implementation.

5.3 Future Work

Key hold time as a feature may not be sufficient to determine handedness when typing with both hands. Future work could include examining other features, such as timing between keystrokes, as well as evaluating other methods of classification. The presented conclusions regarding the distributions of key hold time between right- and left-handed individuals could be used in conjunction with other methods to achieve better results. Having prior knowledge of some of the HMM parameters might be useful when implementing a second version of the classifier.

Regarding the study conducted in [5], the fact that their participants typed certain sentences only with one hand, poses questions about the usefulness of the results with respect to determining handedness. An assumption can be made that it is unusual for individuals to type with only one hand when typing naturally. Therefore, their results may have little use in a real world context, as opposed to results obtained from studying individuals typing naturally.

Our method, as well as the methods used in [2] and [4], used continous HMMs. This choice may seem natural, due to keystrokes being continous intervals in time. However, another possibility would be to line up all keystrokes for a typed sentence, and then sample that set with some frequency. By adding a one when a key is pressed, and a zero when no key is pressed, this would yield a vector of discrete values to be used with discrete HMMs. We could not find any work where such a method was used in the context of handedness classification, which suggests that it could be an idea for future work.

Handedness classification has few obvious application areas. However, it could be used to assist existing user authentication systems to better recognize users by the way they type [5]. An interesting topic is whether the concept of classifying physical traits through keystroke dynamics can be meaningful outside of the context of using computers. For example, a recorded keystroke profile can be synchronized between the computer and mobile devices. In this case, one possible application area of classifying handedness is automatically adjusting UI elements on the screen of the mobile device. Menu elements could, for example, be placed to the left or right depending on whether the user is left or right-handed.

References

[1] Monrose, F., Rubin, A.D.: Keystroke dynamics as a biometric for authentication. Future Generation Computer Systems **16** (2000) 351–359

[2] Vuyyuru, S.K., Phoha, V.V., Joshi, S.S., Phoha, S., Ray, A.: Computer user authentication using hidden markov model through keystroke dynamics. Technical report, Louisiana Tech University, Pennsylvania State University (2012) Also available at: `http://mne.psu.edu/ray/patents/HMMkeyStroke.pdf`.

[3] Bergadano, F., Gunetti, D., Picardi, C.: User authentication through keystroke dynamics. ACM Transactions on Information and System Security (TISSEC) **5 Issue 4** (2002) 367–397

[4] Chang, W.: Improving hidden markov models with a similarity histogram for typing pattern biometrics. In: Proceedings of IEEE International Conference on Information Reuse and Integration. (2004) 467–474
[5] Zulkarnain, S., Idrus, S., Cherrier, E., Rosenberger, C., Bours, P.: Soft biometrics for keystroke dynamics. International Conference on Image Analysis and Recognition (ICIAR), Povoa de Varzim : Portugal (2013)
[6] Smith, N.A.: Hidden markov models: All the glorious gory details. Technical report, Department of Computer Science, John Hopkins University (2004) Also available at: `http://www.cs.cmu.edu/~nasmith/papers/smith.tut04a.pdf`.
[7] Huang, X.D., Ariki, Y., Jack., M.A.: Hidden markov models for speech recognition. Edinburgh University Press (1990) 847–848
[8] Rodríguez, L.J., Torres, I.: Comparative study of the baum-welch and viterbi training algorithms applied to read and spontaneous speech recognition. LNCS 2652 (2003) 847–85
[9] Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings Of The IEEE **77** (1989)

Can Beauty Improve the Perceived Usability of a Form?

Daniel Rosendal

Department of Computing Science
Umeå University, Sweden
daniel.rosendal@gmail.com

Abstract. This article describes a user study where the perceived usability of a computer-based questionnaire was evaluated based on changes to the form's aesthetics. To this end two versions of a questionnaire were developed, one aesthetically pleasing (AQ) and one plain (PQ), the only differences being the color schemes and interface element styles. The test was conducted in two steps. The first step consisted of verifying that the implemented designs differed in aesthetics, and that one was consistently preferred by the targeted user group. In step two, two groups consisting of ten users got to use one version of the questionnaire each. Each user got to fill in the questionnaire and rate in a System Usability Scale (SUS) form [1], a questionnaire consisting of ten questions that evaluate a software's perceived usability, In addition they could also add a comment regarding their experience with the software. The test results gave the AQ a mean SUS score of 74 and the PQ a mean of 69,5. The difference was not judged to be significant enough to draw a conclusion. Mainly due to two user comments that likely skewed their scores. Which could make a substantial difference in sample groups of this size. The study setup does however seem promising for a future study with a larger user group.

1 Introduction

Successful software development as with all product development is achieved by allocating resources to the most beneficial aspects of the product. All software has functionality that it needs to support for it to be meaningful. These are requirements whose value is easy to understand, for example if you develop a word processor it needs to be able to create text documents. Aspects like user centered design and aesthetics are harder to measure and their benefits as well. This often pushes them down the priority list in many software development projects. Research in usability is needed to show that good usability yields good return of investment. Since computers have grown more powerful user interfaces have become more complex which has lead to a greater focus on usability.

Traditional usability has not focused on the feeling that software provides. A sign of this is that Jacob Nielsen's ten usability heuristics only mention of aesthetics is to keep user interface design minimalistic as to not clutter the

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 89–97, January 2015.

UI [2] and Donald Norman complains how many every day objects are made harder to use when form takes precedence over function [3].

This has changed in recent years. The International Standards Organization (ISO) standard for usability includes three dimensions, effectiveness, efficiency and satisfaction [4]. Effectiveness and efficiency relate to the classic values of user-centered design while satisfaction includes the holistic user perspective. Terms like emotional design and user satisfaction are now used in the HCI community and the field is broadened to include the feeling the user gets when using software. Norman seemed to have embraced the notion of satisfaction when he released the book Emotional Design where he focuses on what products make him feel [5]. An aspect of making a product pleasurable is its aesthetics. Aesthetics has many different meanings depending on in which context it's used.

In this article we use the meaning "A pleasing appearance or effect: Beauty" [6]. A number of papers have looked into the effect of beauty in software and how much it might affect perceived system usability. A study by [7] showed that the perceived usability of a product is strongly affected by the beauty of the system. They coined the terms Apparent Usability (AU) and Inherent Usability (IU). Apparent Usability is a measure of how usable a system is perceived to be by its users. Inherent usability is a measure of the actual usability of the system. For example a system that misses important features and is cumbersome to use because of bad information architecture will get a bad IU rating as task performance times are long and some important tasks can't be performed while maintaining a higher UA rating as long as its users are satisfied when using it. They might not know of a better alternative or the system could have some other redeeming features that overshadow the negative sides. [7] showed that AU has a stronger relation to beauty than IU. Their test was limited to before usage impressions. It might seem like a narrow scope but first impressions are of great importance in user interface development [8]. A later a study by [9] went further by evaluating the effects of beauty on perceived usability after system use. He found that beauty influenced the perceived usability after system use as well. The psychological mechanism that is theorized to cause the effect is the so-called "Halo effect" where the system beauty causes users to ascribe other good characteristics to the system in the same way physically attractive humans are [10]. There has been much debate regarding the findings where others have found that the effect of beauty has been overemphasized [11] and yet others who have found it to be true [12] [13]. This leaves the question of how software aesthetics affects the users impression of the product unanswered.

The aim of this study is to narrow the scope of the question to a specific UI, a questionnaire. It will answer the question: "Can beauty improve perceived and actual performance of a form?" The study will be conducted by letting users evaluate the beauty of two questionnaires that share content but are different in aesthetics. A user group will decide which one is the most aesthetically pleasing. A second group of users will be assigned a questionnaire version at random, and at completion rate it using a SUS form [1]. The SUS is described in the next section.

1.1 SUS Usability Scale

Since usability is a combination of many user aspects such as task completion times and error rates it is measured in several different ways. Depending on what aspects of usability are to be measured and the availability of users, it can be done either heuristically or through user tests. To perform meaningful user testing, the collected data must be properly interpreted so it can be used to improve the tested product.

To measure the classical usability properties there are many popular frameworks and tools including but not limited to Goals, Operators, Methods, and Selection rules (GOMS) however those tools are not suited for measuring the users experience of the products usability. To this end the System Usability Scale (SUS) [1] was developed. It is meant to capture the holistic usability experience. SUS is a questionnaire containing ten questions where answers are selected from a 5 level Likert Scale ranging from "Strongly Agree" to "Strongly Disagree". Each response is converted to a number and all numbers are condensed to a single score between 0 and 100 where a better score indicates better usability.

SUS is one of many similar questionnaires but was chosen as it has been extensively used and researched. Its scores have been shown to correlate well with classic usability scores. The popularity of SUS stems from three main benefits. It is technology agnostic, which makes it usable in a wide range of products and also yields possibilities to compare different interfaces. Product comparisons can otherwise be problematic as pragmatic usability values such as good task completion times vary widely between products. Second it is quick and easy to administrate and use from both a test subjects and an administrators perspective. Also since the result is a single score, other interested parties such as project lead and management can easily understand the results without the need to have UX experience. The third reason is financial, since the form is non-proprietary it requires no investment other than time to use [14].

However, there are downsides as well, [14] showed that the scores even though they correlate well with actual usability seem to get inference from other factors when evaluating larger systems. Users gave the system a good rating even though some of the tasks they were given couldn't be completed. They speculate that it might be a halo effect from successful tasks that over weigh the negative experience of others. The SUS output can also be somewhat problematic, it detects the users impressions of the system but it does not identify the mechanisms that led to the given score.

None of these problems prove problematic for the evaluation of the experiment as the test consists of a single task and it is only the comparison between system styles that are of interest.

2 Defining Beauty and Aesthetics

In the Oxford dictionary aesthetics is defined as "A set of principles concerned with the nature and appreciation of beauty", however it's meaning has changed

over time. According to [15] it originates from ancient Greece where it meant "to perceive" or "things perceptible". During the medieval ages heavily influenced by the church aesthetics was a branch of theology as beauty was considered a reflection of god. During the renaissance it became a normative discipline. The modern definition of the word was coined in the eighteenth century when Alexander Gottlieb Baumgarten (1714-1762) linked it to a sense of beauty, it became a theory of beauty [16].

The opinions of what is beautiful have been seen from different perspectives as well. On one end architect Louis Henry Sullivan (1856-1924) claimed that "Form follows function" in other words something that is functional is beautiful. On the other end Kant (1724-1804) claimed that beauty is disconnected from an objects function. It is a product from a person's disinterested feel for it [17]. These differing views illustrate an ongoing debate if beauty is subjective or objective. [15] suggests that it might be both. They divide the empirical studies of aesthetics in two categories, the Experimental and the Exploratory approaches. Experimental studies assumes that there are objectively pleasing aspects that can be isolated and manipulated to create a aesthetic whole. Examples of such characteristics are the golden ratio and Pythagorean proportions. Exploratory studies the whole rather than isolated parts and is more concerned with the users perception of the product. This study is conducted through an Exploratory approach and bases the interface design on user feedback.

3 Method

3.1 Participants

Fifteen individuals participated in the study, eleven men and four women with a mean age of 34. All participants were working as engineers in software development. They had all filled out electronic questionnaires in the windows environment before.

3.2 Material

The software used in the experiment was developed in WPF .Net 4.5. It was chosen as WPF easily allows heavy customization of interface elements. All tests were run on a Mac Book pro with a screen resolution 2560x1600.

3.3 Procedure

The goal of the study was to evaluate the effect of beauty has on the perceived usability of a form. The software that was used in the test was a questionnaire containing 30 questions that was developed for the study. It was implemented with two different visual styles, one with a mix of the Windows Aero and Classic themes (Figure 2) and the other with a flat design custom theme (Figure 1).

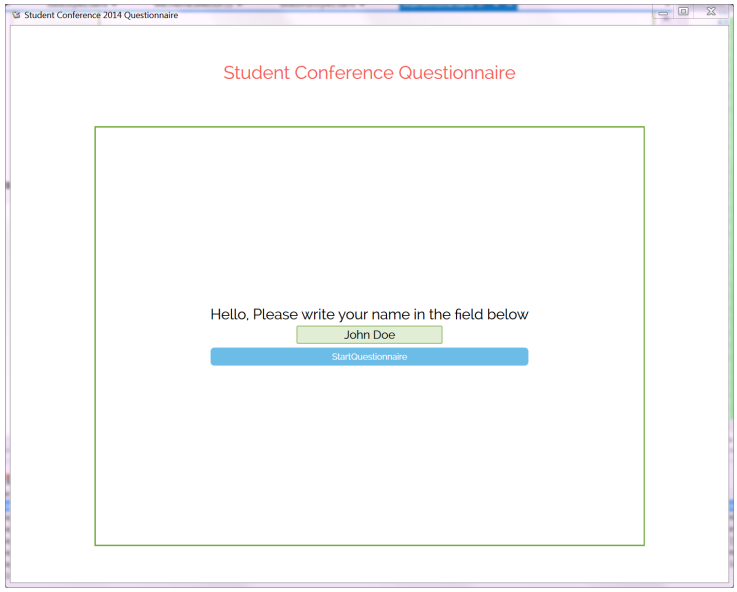


Fig. 1. Start screen of a Flat design themed questionnaire used in the user study

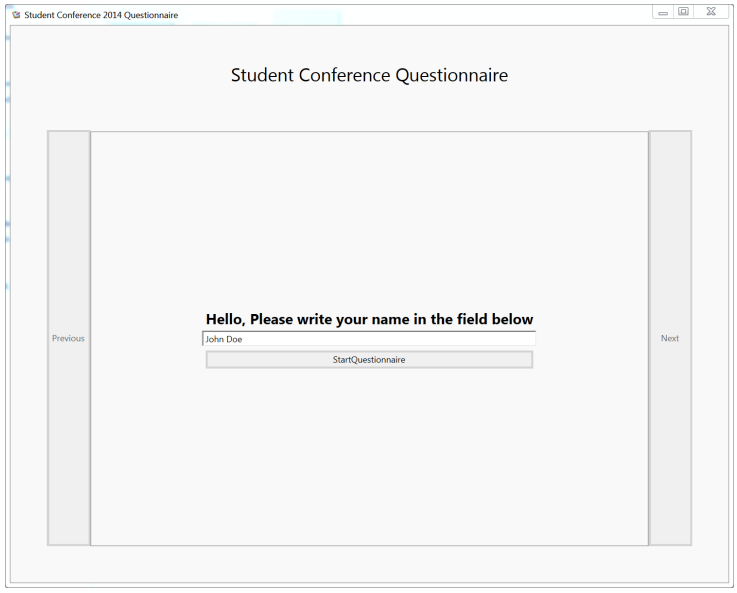


Fig. 2. Start screen of a Windows themed version of a Questionnaire used in the user study

Artificial Interaction Flaws (AIF) were implemented in the form of slow response times and ignored mouse clicks for certain questions. These flaws were always the same for both styles. They were introduced as the questionnaire was deemed a simple UI and it was theorized that the scores would generally be on the upper region of the SUS scale. This could lead to loss in granularity if the ratings were set to the maximum score on many questions. The test consisted of two steps where the first one was done to verify that assumptions regarding the users preferences regarding the themes were correct. The beauty of the questionnaires had to differ significantly and consistently for all users for the tests to be meaningful. The questionnaire was presented to a group of 5 subjects, each person got to fill in both versions of the questionnaire and afterwards rate which one was the more aesthetically pleasing and also if they wanted to add any comments. The system that was rated as most pleasing was labeled the Aesthetic Questionnaire (AQ) and the other as the Plain Questionnaire (PQ). In this step the users were informed of the AIF and were told to ignore them and only focus on the aesthetics.

During the second step the remaining 10 participants were divided in two groups where each group filled in one version of the questionnaire. Each participant got the same introduction to the task. They were told to answer the questions as best they could and that they would not be evaluated on their performance nor the time it took them to complete the task. After they completed the task they rated the Software by answering a printed version of the SUS. They were informed of the AIF and asked if they would like to write any additional comments on their experiences with the software. The SUS-score for each participant was calculated as well as the group mean. The mean comparison formed the basis for the evaluation.

4 Result

The theme evaluation resulted in the AQ being considered more aesthetically pleasing by 4 out of 5 users (Table 1).

	AQ	PQ
Preferred Theme	4	1

Table 1. Scores from the preferred aesthetics test for the AQ and PQ questionnaires.

However both themes got low scores from the comments. The responses to the PQ said it looked old and unappealing which was the desired result. The AQ version was rated as having too much color as well as bad color composition. The results triggered a redesign of the AQ, with the feedback a new theme was developed. A second evaluation followed where AQ distinguished itself enough to start the next step (Table 2).

	AQ	PQ
Preferred Theme	5	0

Table 2. Scores from the preferred aesthetics test for the AQ and PQ questionnaires after theme adjustments.

Step two resulted in the AQ receiving a mean SUS-score of 74 and the PQ a mean score of 69,5 (Table 3).

AQ	PQ
55	80
47,5	70
92,5	95*
95	47,5
80	55**
Group Mean	
74	69,5

Table 3. The SUS scores for the AQ and PQ questionnaires received in the user study. *User assumed AIF to be hardware related **User affected by software bug

Two users in the PQ group chose to add additional comments. One was in regard to the AIF. The user had never used a Mac before and assumed that the unresponsiveness of the UI was due to low quality hardware. The second user noted a bug in the software, in response to ignored mouse clicks he had clicked the next button multiple times and as a result was unsure if one of the question had accidentally skipped. When going back to check it was noted that answers were erased when doing so which led to extra work and frustration on the users part.

5 Discussion

The results show that the AQ got a slightly higher SUS-score than its non-styled counterpart. This suggests that the system aesthetics does effect the perceived usability. However considering the small data size and the comments collected from the PQ group makes the results inconclusive. The user who attributed the AIF to the hardware likely gave the system a higher score than others who assumed that the problems related to bad internal software quality. The effect on commenter two's bug likely had a worse experience with the system than other users and might have rated the system higher if he had not encountered the problem. The two users scores might cancel the effect of each other, commentator one gave a shared highest score of all users (95) while commenter two gave a below average score. Ultimately it does not matter since there is no way to tell how much their experiences affected their scores. The fact that each group consisted of 5 users means that any change in an individual's score has significant

effect on the group mean score. When taking these factors in account it is hard to draw any reliable conclusions from the study.

Even though it yielded a weak base for conclusions the test setup and software seems like a promising start for a more extensive study. Before such an experiment is undertaken the software should undergo a test session to identify any bugs in the interaction and the test persons should be familiarized with the hardware platform that the tests are performed on.

References

[1] Brooke, J.: Sus-a quick and dirty usability scale. Usability evaluation in industry **189** (1996) 194

[2] Nielsen, J., Molich, R.: Heuristic evaluation of user interfaces. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM (1990) 249–256

[3] Norman, D.A.: The design of everyday things. Basic books (2002)

[4] ISO, W.: 9241-11. ergonomic requirements for office work with visual display terminals (vdts). The international organization for standardization (1998)

[5] Norman, D.A.: Emotional design: Why we love (or hate) everyday things. Basic books (2004)

[6] Inc., M.W.: Merriam-Webster's collegiate dictionary. Merriam-Webster (2004)

[7] Kurosu, M., Kashimura, K.: Apparent usability vs. inherent usability: experimental analysis on the determinants of the apparent usability. In: Conference companion on Human factors in computing systems, ACM (1995) 292–293

[8] Lindgaard, G., Fernandes, G., Dudek, C., Brown, J.: Attention web designers: You have 50 milliseconds to make a good first impression! Behaviour & information technology **25**(2) (2006) 115–126

[9] Tractinsky, N., Katz, A., Ikar, D.: What is beautiful is usable. Interacting with computers **13**(2) (2000) 127–145

[10] Dion, K., Berscheid, E., Walster, E.: What is beautiful is good. Journal of personality and social psychology **24**(3) (1972) 285

[11] Tuch, A.N., Roth, S.P., Hornbæk, K., Opwis, K., Bargas-Avila, J.A.: Is beautiful really usable? toward understanding the relation between usability, aesthetics, and affect in hci. Computers in Human Behavior **28**(5) (2012) 1596–1607

[12] Hassenzahl, M., Monk, A.: The inference of perceived usability from beauty. Human–Computer Interaction **25**(3) (2010) 235–260

[13] Sonderegger, A., Sauer, J.: The influence of design aesthetics in usability testing: Effects on user performance and perceived usability. Applied ergonomics **41**(3) (2010) 403–410

[14] Bangor, A., Kortum, P.T., Miller, J.T.: An empirical evaluation of the system usability scale. Intl. Journal of Human–Computer Interaction **24**(6) (2008) 574–594

[15] Lavie, T., Tractinsky, N.: Assessing dimensions of perceived visual aesthetics of web sites. International journal of human-computer studies **60**(3) (2004) 269–298
[16] Saw, R., Osborne, H.: Aesthetics as a branch of philosophy. The British journal of aesthetics (1) (1960) 6–20
[17] Borev, Y., Belskaya, N., Philoppov, Y.: Aesthetics: a textbook. Progress Publishers (1985)

Using Hidden Markov Models to Classify Head Gestures When Using Google Glass

Emil Sjölander

Department of Computing Science
Umeå University, Sweden
oi10esr@cs.umu.se

Abstract. This paper explores the use of Hidden Markov Models (abbreviated HMMs) to detect head gestures performed while wearing a Google Glass device. Gyroscope values in the x and y axis are used as features which participants input by performing head nods and head shakes while wearing Google Glass. A total of 20 people were asked to input 10 of each gesture. These sequences of gyroscope values were then used to train two HMMs using the Baum-Welch algorithm. When tested the HMMs achieved more than 90% correct classifications on the testing set. We discuss the advantages and disadvantages of HMMs for this classification task as opposed to a heuristic approach or using Artificial Neural Networks.

1 Introduction

Gesture recognition is the ability to use some form of sensor data to reliably predict what gesture a participant is performing. One approach to hand gesture detection is through vision based systems [1]. Another approach to hand gesture recognition is to use motion sensors such as accelerometers to detect gestures [2]. This is a popular approach as many mobile devices include such sensors.

We will be investigating head gesture detection. This is conceptually similar to hand gesture detection but instead focuses on head movement. There exists hardware and software to track head movement today but it requires an external camera faced towards the participant [3]. Google Glass[1] (henceforth denoted Glass) is a device that is mounted to the participants head and we can therefore use movement sensors to detect head gesture instead of using a visual system. The question we investigate in this paper is if we can make use of Hidden Markov Models (henceforth denoted HMMs, or HMM for a single model) to detect a limited set of head gestures (nodding and shaking) performed while wearing Glass.

HMMs were chosen because of their ability to accurately classify signals (functions of time). HMMs have seen many successes in classifying signals, an early example of such a success is in speech recognition [4]. In more recent years

[1] http://wwww.google.com/glass

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 99–110, January 2015.

HMMs have been widely used in classifying gestures (a signal of gyroscope, accelerometer, or pixel values) as well [2].

The motivation for this work is the ability for head gesture recognition technology to open up new forms of interaction. This could be of great help to people with disabilities which make it hard or impossible for them to control modern technology devices with their hands. It would also be very helpful in professions such as medicine and construction where the hands of the professional are busy.

There are multiple examples of earlier work using sensors in mobile devices to classify gestures, one such example is the work of Marco Klingmann in classifying gestures using an iPhone [2]. There has however not been any work done using Glass in the context of head gesture recognition. The difference between this research and prior work within gesture recognition is the placement of the sensors. With Glass the sensor is places on the head of the wearer which in turn brings with it a very different environment compared to using the sensors in a handheld device.

This paper is organized as follows. We start with some background knowledge on gesture recognition, Glass, and HMMs in Section 2. Section 3 covers the methods and tools used for gathering and analyzing the data. In Section 4 we present the results obtained from testing our classifier. Finally Section 5 discusses the results presented in Section 4 and discusses limitations, possible alternative solutions, and future work.

2 Background

2.1 Gesture Recognition

Gesture detection is about extracting meaning from a sequence of movements and distinguishing one sequence of movements from another. The motivation for trying to get computers to recognize gestures is that if a computer can recognize human gestures it can hopefully provide a more intuitive interface to its functions. Currently almost all input to computers is via explicit input. If a computer instead learned to recognize gestures it could perform certain tasks without the user needing to explicitly ask it to [5].

Gesture recognition can be performed in a lot of different ways but is usually done through the use of motion sensors or cameras. In this paper we focus on gesture detection through motions sensors, specifically a gyroscope. We will use this data to detect when a user nods or shakes their head.

Previous work within this field has had great success in classification of hand gestures using HMMs with sensor data from a mobile device as input [2]. Another examples of HMMs being used to successfully classify sensor data is for authentication based on walking patterns [6].

2.2 Google Glass

Glass is a wearable computing device currently being developed and tested by Google. A beta program makes Glass available to a larger audience but the

device should still be considered a beta product. Glass has hardware very similar to that of a smartphone. Instead of a typical display Glass has a prism which the interface is projected onto simulating a large screen in front of the wearer. For this research we made use of the motion sensors in Glass. Glass has three motion sensors, an accelerometer, a gyroscope, and a magnetometer. Glass runs the android KitKat operating system which we made use of to build the application that collects the sensor data we needed.

2.3 Hidden Markov Models

An HMM is a generative model which builds upon the Markov properties [7] of a data set. An HMM contains N unique hidden states [4] defined by S:

$$S = \{S_0, S_1, ..., S_N\}.$$

At each time step t the HMM is in one of these states. We denote the state which the HMM is in at time t as q_t. The state in which the HMM starts in is defined by π which is a discrete probability density function over the hidden states. Thus π_i describes the probability of starting in state i:

$$P(q_0 = S_i) = \pi_i.$$

For every time step the HMM has a certain probability to transition into another state, these probabilities are defined by a state transition matrix $A = \{a_{ij}\}$ where a_{ij} is the probability to transition from state i to state j:

$$P(q_t = S_j | q_{t-1} = S_i) = a_{ij}.$$

After this transition occurs the HMM will emit an observation. V defines the emission domain of the HMM where v is any single value from this domain. The observation at time step t is defined by O_t. The probability that state i emits an observation symbol v is defined by the vector of emission probability distributions $B = \{b_1(v), b_2(v), ..., b_N(v)\}$:

$$P(O_t = v | q_t = S_i) = b_i(v).$$

In summary an HMM M can be parameterized by a initial state distribution π, a state transition matrix A and an emission probability distribution vector B:

$$M = (\pi, A, B).$$

Finding the probability that a sequence of observations was generated from an HMM is done with the forward algorithm [4] which computes the probability that at some time step t the HMM is in state S_i given the observations $O = \{O_1, O_2, ..., O_t\}$. We use this algorithm during both classification and training procedures. The forward algorithm sets up a recursion on the previous time

step so it can use the principals of dynamic programming to efficiently solve the problem:

$$F(s,0) = \pi(s)b_s(O_1),$$
$$F(s,t) = b_s(O_t)\sum_{k=0}^{N} a_{ks}F(k,t-t).$$

Another important algorithm is the backward algorithm [4]. Instead of calculating the probability that of ending in a state after observing a sequence it calculates the probability of observing a sequence given that the HMM is in a certain state. This can be used to predict the probability that the HMM will end up emitting a certain sequence of observations, we use this algorithm during training:

$$B(s,0) = 1,$$
$$B(s,t) = \sum_{k=0}^{N} a_{sk}b_k(O_t)B(k,t-1).$$

Training an HMM is done with the Baum-Welch algorithm [8]. The Baum-Welch algorithm implements the well known expectation maximization algorithm [9] to increase the likelihood that a given sequence of observations was generated from a given HMM. As with any expectation maximization algorithm Baum-Welch performs a hill climbing maneuver which will result in local maximum. To ensure that a sufficiently good local maximum is reached one has to carefully choose the initial parameters of the HMM or test with multiple randomly chosen parameters. Baum-Welch makes use of both the forward and backward algorithms when calculating the new adjusted model parameters. The Model re-adjustment is done by computing the expected model parameters given an observed sequence.

Using above named algorithms we can train our HMMs and determine the probability that an HMM generated a sequence. This is how an HMM is used for classification. For every class that we want to classify, in our case head nods and head shakes, an HMM is constructed and trained. M_{nod} is the model for head nods and M_{shake} is the model for head shakes. When a sequence S is classified each HMM is asked what the probability p was of them having generated that sequence. In our case we have two such probabilities. p_n,the probability that M_{nod} generated the sequence, and p_s, the probability that M_{shake} generated the sequence:

$$p_n = P(S|M_{nod}),$$

$$p_s = P(S|M_{shake}).$$

We cannot compare p_n and p_s directly as they are derived from different HMMs. The probability of a sequence given an HMM cannot be compared to the same sequence given another HMM. This is because their probabilities can be at very different scales. We therefore need to look at the ratio r between the probability of a sequence given each of the models. With these ratios at hand the problem of classification is reduced to finding a linear boundary between two clusters of ratios and assigning new ratios to one of these clusters:

$$r = \frac{p_n}{p_s}.$$

3 Data Collection and Classification

Data was collected using Glass. An application[2] was written which asked the users to input head gesture data by nodding and shaking their heads. Each user was asked to perform ten nods and ten shakes in alternating order. No user was asked to input data more than once to make sure the HMM generalized well and did not over train on a certain person's way of performing a gesture. A total of 20 people were asked to input data which resulted in 200 nod samples and 200 shake samples. The data was collected under supervised conditions so that we could monitor that the participants were performing the tests correctly. The data was collected on Glass in multiple CSV[3] files so that it could easily be retrieved and analyzed in any program or programming language.

Training of the HMMs was not done on the Glass device but on a standard home laptop. To construct and train the HMMs an open source Java library called jahmm[4] was used. Jahmm contains code for instantiating a HMM as well as code implementing all the algorithms discussed in Section 2.3. Jahmm was chosen because of its good documentation and good references from others using it. Other libraries and tools such as ghmm[5] were investigated but later dropped because of the lack of documentation making it hard to be certain that the library was used correctly. A python script was written for transforming the CSV data captured on Glass to the format jahmm uses to model sequences of observations.

While all the available sensor data was collected during our data collection we chose to only use the gyroscope values in the x and y axis as our features. This dimensionality reduction was performed because of the simplicity of the gestures we wanted to recognize. A shake is a rotation around the y axis and a nod is a rotation around the x axis. Using any more features could just add unnecessary noise. The sensors were sampled at 60hz for 2 seconds giving us a

[2] https://github.com/emilsjolander/glassheadgestures
[3] http://en.wikipedia.org/wiki/Comma-separated_values
[4] https://code.google.com/p/jahmm/
[5] http://ghmm.org/

sequence of 120 tuples of gyroscope values $\{(x_0, y_0), (x_1, y_1), ..., (x_{120}, y_{120})\}$ as our input data to the HMMs.

As noted in Section 2.3 choosing the initial values of an HMM is very important. Both the HMM modeling the nod gesture and the HMM modeling the shake gesture were initialized with the same values which was done for simplicity's sake. Each HMM was chosen to have four hidden states. The number of hidden states was decided by reasoning about what each state might represent. The states that we imagined were:

1. Before a gesture is started.
2. A positive rotation.
3. A negative rotation.
4. After a gesture is finished.

The initial value for π was chosen to be $[0.25, 0.25, 0.25, 0.25]$ because we did not have any prior knowledge about which state the HMM should start in. We could have used our reasoning about what each hidden state may represent as prior knowledge when choosing π. However we did not do this because we had no evidence that our reasoning was correct, it was mearly used as a way to estimate the number of states the HMM would need. Instead we let π be optimized during training.

Given our theory on what the states could represent we could have chosen a π which represented this theory but decided to let this be optimized during training instead.

By using Octave[6] to find the mean and variance of our input data we could make a good estimate of the emission distribution. Each state was set to have the same initial emission distribution because we lacked the prior knowledge to choose them. A multivariate Gaussian emission distribution was chosen with a mean of $[0, 0]$ and the covariance matrix $[[0.1, 0], [0, 0.1]]$. Because of the many random variables involved in measuring movement with a gyroscope, guassian emissions could be assumed in accordance to the central limit theorem [10].

The initial state transition matrix A was also set to have an equal probability for all state transitions. The initial state transitions could not be set to be exactly equal though as this put the HMM into a local maximum. Initializing them with slight variations led to a much better result:

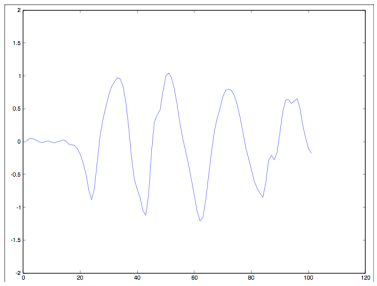
$$A = \begin{bmatrix} 0.24 & 0.26 & 0.22 & 0.28 \\ 0.24 & 0.26 & 0.22 & 0.28 \\ 0.24 & 0.26 & 0.22 & 0.28 \\ 0.24 & 0.26 & 0.22 & 0.28 \end{bmatrix}$$

Once the HMM was trained it was exported so that it could later be used in a separate Glass application with the purpose of testing the resulting HMMs. This Glass application would display either yes, no, or nothing at all when the user performed head gestures while wearing Glass.

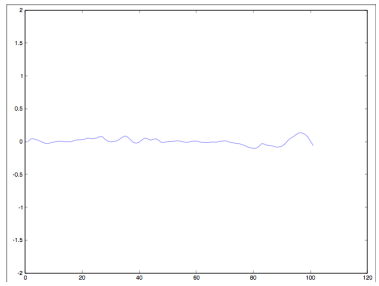
[6] https://www.gnu.org/software/octave/

4 Results

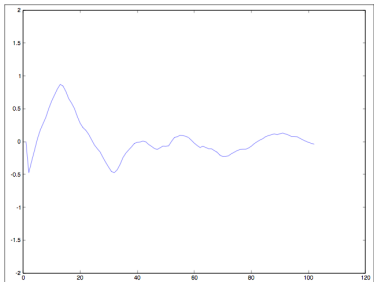
Figures 1a, 1b, 1c, and 1d show time series plots of the gyroscope values while performing nodding and shaking gestures. The graphs clearly show more noisy data for the axis that does not correspond to the gesture.



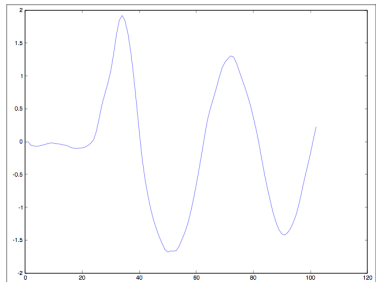
(a) Gyroscope x axis values while nodding



(b) Gyroscope y axis values while nodding



(c) Gyroscope x axis values while shaking



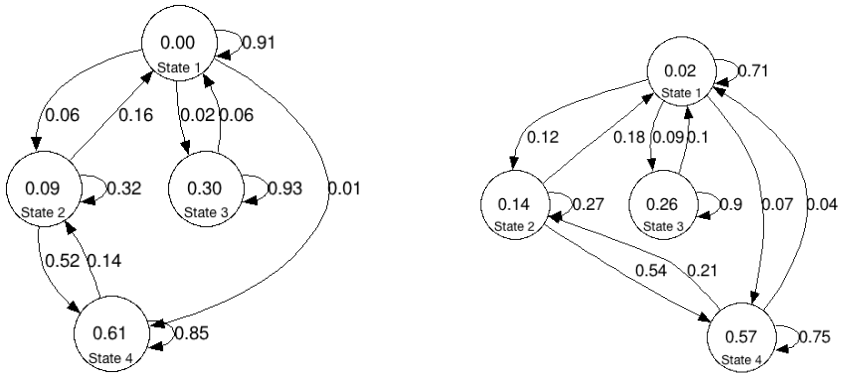
(d) Gyroscope y axis values while shaking

Fig. 1: Time series plot of gyroscope values

Figures 2a and 2b show the resulting HMMs used to classify nodding and shaking gestures. The graphs only show the probability to start in a state and the probability to transition between states. The parameters to the multivariate Gaussian emission distributions in each state are shown in Figure 3. The mean vector of each state in each of the HMMs is labeled with $\mu_{shake}(k)$ and $\mu_{nod}(k)$ respectively, where k is the state's index as seen in Figures 2a and 2b. The covariance matrix for each state in each of the HMMs is labeled $\Sigma_{shake}(k)$ and $\Sigma_{nod}(k)$, respectively.

Table 1 shows extracted results of 10 shaking gestures and 10 nodding gestures, each row represents two sample sequences, one nod sequence and one shake sequence. The values in each column are the ratios r for shaking and nodding gestures, respectively. This is just a small sample of the total results obtained but it is fairly representative of the rest of the results.

These results show a clear split between shaking and nodding ratios. The ratio for the shaking gesture is very close to zero while the ratio for the nodding gesture



(a) Resulting HMM modeling the nodding gesture

(b) Resulting HMM modeling the shaking gesture

Fig. 2: Resulting HMMs

$$\mu_{nod}(1) = \begin{bmatrix} -0.035 & -0.009 \end{bmatrix} \quad \Sigma_{nod}(1) = \begin{bmatrix} 0.384 & -0.007 \\ -0.007 & 0.004 \end{bmatrix}$$

$$\mu_{nod}(2) = \begin{bmatrix} -0.012 & -0.009 \end{bmatrix} \quad \Sigma_{nod}(2) = \begin{bmatrix} 0.029 & 2.474 \times 10^{-4} \\ 2.474 \times 10^{-4} & 0.001 \end{bmatrix}$$

$$\mu_{nod}(3) = \begin{bmatrix} 0.065 & -0.011 \end{bmatrix} \quad \Sigma_{nod}(3) = \begin{bmatrix} 0.413 & -0.045 \\ -0.045 & 0.202 \end{bmatrix}$$

$$\mu_{nod}(4) = \begin{bmatrix} -0.011 & -0.009 \end{bmatrix} \quad \Sigma_{nod}(4) = \begin{bmatrix} 0.003 & 1.474 \times 10^{-4} \\ 1.474 \times 10^{-4} & 5.681 \times 10^{-4} \end{bmatrix}$$

$$\mu_{shake}(1) = \begin{bmatrix} -0.015 & -0.02 \end{bmatrix} \quad \Sigma_{shake}(1) = \begin{bmatrix} 0.004 & -0.015 \\ -0.015 & 0.478 \end{bmatrix}$$

$$\mu_{shake}(2) = \begin{bmatrix} -0.015 & -0.006 \end{bmatrix} \quad \Sigma_{shake}(2) = \begin{bmatrix} 0.001 & -0.001 \\ -0.001 & 0.056 \end{bmatrix}$$

$$\mu_{shake}(3) = \begin{bmatrix} -0.025 & -0.051 \end{bmatrix} \quad \Sigma_{shake}(3) = \begin{bmatrix} 0.104 & -0.095 \\ -0.095 & 1.250 \end{bmatrix}$$

$$\mu_{shake}(4) = \begin{bmatrix} -0.014 & -0.008 \end{bmatrix} \quad \Sigma_{shake}(4) = \begin{bmatrix} 7.382 \times 10^{-4} & -2.880 \times 10^{-5} \\ -2.880 \times 10^{-5} & 0.008 \end{bmatrix}$$

Fig. 3: Resulting Emission Distribution parameters

Shaking	**Nodding**
8.055×10^{-40}	2.492×10^{79}
9.668×10^{-51}	1.262×10^{45}
1.281×10^{-110}	1.889×10^{90}
2.004×10^{-51}	7.197×10^{74}
2.375×10^{-60}	8.118×10^{149}
1.591×10^{-108}	8.136×10^{73}
3.845×10^{-18}	6.569×10^{69}
7.653×10^{-50}	2.916×10^{93}
1.109×10^{-39}	9.592×10^{59}
1.957×10^{-73}	2.885×10^{69}

Table 1: Ratio for a gesture given a shaking model and a nodding model

is very large. This makes it very easy to choose a threshold that maximizes the number of correct classifications. Choosing a threshold of 1 yields to >95% correctly classified samples. This threshold can be further improved to avoid classifying something that is neither a nodding or a shaking gesture as one of the two. Experimentation led to choosing 1.0×10^{-30} and 1.0×10^{30} as the threshold values for the shaking gesture and the nodding gesture respectively. Anything in between would count as a an unknown gesture. Doing so still leaves us with >90% correctly classified samples while also giving us the ability to dismiss other gestures. The exact threshold that should be used depends on what properties we want our classifier to have.

5 Discussion

5.1 Conclusion

The results in Section 4 are very promising. They show that we can detect shaking and nodding gestures from Glass using HMMs with a certainty of 90%. Exporting the resulting HMMs back onto Glass to test the classification capabilities showed even better results than the data in Section 4. Because the sensor values are a continuous stream of data we can detect a gesture with high certainty even if we only with a 90% certainty can detect a gesture at a certain time step. This is because the sequence input to the HMMs are rolling windows with a width of 120 samples taken from the continues stream of sensor data. The downside of this approach is that it can take a longer to classify new gestures in a sequence of gestures. However the advantages of high stability and high certainty outweigh the disadvantage of classification delay in most cases.

As long as the threshold values described in the Section 4 are high enough that no misclassification are made we do not care that much about not being

able to classify a gesture at each time step as it is natural to continue a gesture until it is has been received and understood.

5.2 Limitations

The data collected from users included errors. There were known cases of participants inputing the wrong gesture but because of our large sample size we took for granted that a 1-2% error rate would not harm the resulting HMM's ability to correctly classify the gesture. Looking at the results this seems to have been a fine assumption to make.

Initializing both the HMMs with the same values was as stated in Section 3 done for sake of simplicity. Initializing each HMM separately with values more appropriate for the class it was supposed to classify would probably have led to faster convergence. Given the classification results which were quite good we do not think that setting the initial parameters to be equal had a big effect on the quality of the classifier.

The results are quite promising with respect to the number of correctly classified sequences, however there are a few limitations worth discussing. The first limitation is that the model will often classify gestures that are neither nods or shakes. This can be somewhat avoided by choosing a high threshold. A downside is that a high threshold leads to many smaller movements not being classified as a shake or a head when they should be.

This limitation should not be a big problem in applications of the classifier. Applications will probably make use of this classifier to answer short yes or no questions, such as "Would you like to share this". In this case the gesture recognition would only be active while the user is answering the question which lowers the risk of them performing any other gesture than a nod or a shake.

Another limitation that could have a larger impact on applications is the noise from the environment. The classifier was not trained in environments with a lot of sensor noise such as while walking, bicycling, or driving a car. If this classifier is used to ask any important questions it is vital that driving over a bump in the road does not classify as a gesture. This has not been tested and should of course be done before using this in any real world applications.

5.3 Possible Alternate solutions

The problem of detecting head nods and shakes that we have focused on in this paper could have been solved in a number of alternate ways. We will discuss three different ways that the problem could have been solved in and their advantages and disadvantages compared to the approach used in this paper.

A slightly different solution that would still use HMMs would make use of discrete data instead of continuous data. The way this would be done is by mapping the continuous values to a range of integers. This has a direct benefit of less noisy data. Also implementing HMMs for discrete data is simpler than the implementation for continuous data so the risk of implementation error is

reduced. The disadvantage with this approach is that mapping the continuous stream of sensor data to a range of integers might not only remove noise but also important information.

By far the simplest solution to this problem is to use a heuristic approach to classification. Just looking at the variance of movement in the x axis compared the the y axis of the last 2 seconds would probably have given very similar results as the ones we have seen. The benefit of this approach is of course that is is much easier. The downside is that is is harder to detect more complex gestures. The approach used in this paper could be directly applied to any head gesture given that the correct sensor data is used.

Another approach would be to use an Artificial Neural Network [11, Chapter 18.7] (henceforth denoted ANN). An ANN has the benefit of being trained with both negative and positive training examples. This would hopefully reduce the number of classifications of non-gesture sequences as the ANN would be trained to not respond positively to them.

5.4 Future Work

There is still a lot of work left to be done within the field of head gesture recognition. Firstly while this paper covers the basics of gesture recognition with Glass it only lays the ground work by implementing models for two gestures which are quite easy to distinguish. Future work should look into detecting many more gestures in addition to more similar gestures which might be harder to distinguish. While the gestures covered in this paper are enough to implement some new more intuitive gesture driven applications for Glass they are also very limiting in what kind of functionality they can provide.

Secondly this paper has not put any focus on the user interaction point of view. Do gestures like the ones discussed in this paper contribute to an easier use of Glass? Can this kind of gesture recognition help those with major disabilities? These are very important questions that should be answered before implementing this in any application.

Acknowledgments

We would like to thank Thomas Hellström, Lennart Steinvall, Karl Pertersson, Mikaela Berg, and Norman Louis for proof reading the many drafts of this paper.

References

[1] Wu, Y., Huang, T.S.: Vision-based gesture recognition: A review. Proceeding GW '99 Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction (1999) 103–115

[2] Klingmann, M.: Accelerometer-based gesture recognition with the iphone. Technical report, Goldsmiths University of London (2009)

[3] Morency, L.P., Sidner, C., Lee, C., Darrell, T.: The role of context in head gesture recognition. Proceedings of the 21st national conference on Artificial intelligence **2** (2006)
[4] Rabiner, L.R.: A tutorial on hidden markov models and selcted applications in speech recognition. Proceedings of the IEEE **77**(2) (1989)
[5] Kelly, S.D., Manning, S.M., Rodak, S.: Gesture gives a hand to language and learning: Perspectives from cognitive neuroscience, developmental psychology and education. Language and Linguistics Compass 2 (2008)
[6] Nickel, C., Brandt, H., Busch, C.: Benchmarking the performance of svms and hmms for accelerometer-based biometric gait recognition. Proceedings of the 2011 IEEE International Symposium on Signal Processing and Information Technology (2011) 281–286
[7] RÃ¸nn-Nielsen, A., Hansen, E.: Conditioning and markov properties. Technical report, Department of Mathematical Sciences, University of Copenhagen (2014)
[8] Smith, N.A.: Hidden markov models: All the glorious gory details. Technical report, Johns Hopkins University (2004)
[9] Dellaert, F.: The expectation maximization algorithm. Technical report, College of Computing, Georgia Institute of Technology (2002)
[10] Alm, S., Britton, T.: Stokastik: sannolikhetsteori och statistikteori med tillämpningar. Liber (2008)
[11] Russell, S., Norvig, P.: Artificial Intelligence, A Modern Approach. 3 edn. Pearson Publishing (2009)

Evaluating the Importance of Color on Call-to-Action Buttons in User Interfaces

Lisa Sundberg

Department of Computing Science
Umeå University, Sweden
oi10lsg@cs.umu.se

Abstract. Today, almost all e-commerce sites have a Call-to-Action button. Because of the impact a Call-to-Action button can have on a company's revenue increase, it is a widely discussed subject in social media. For instance, some companies claim that the color of the Call-to-Action button can make a big difference in revenue increase. Despite, there has been very few research studies within the area of Call-to-Action buttons. Additionally, none of these studies emphasize the importance of the button's color. The present study investigates if a user rather clicks a button in a color they like, than a button in a color they dislike or like less. The result shows that 87 % of the 95 users participating, clicked the button with the color they specified as their favorite. However, a high number of the participators mentioned that they clicked the button they did because they earlier had specified that color as their favorite. This answers is seen as something affecting the credibility of the result negatively, and it is therefore believed that more studies have to be made in order to draw any general conclusions.

1 Introduction

A "Call-to-Action" (CTA) button, is a term used in interface design for buttons that invite the user to take some kind of action [1]. It could be an "add to cart" button, a "download now" button, a "save now" button, a "learn more" button or for instance a "sign up" button. Because of the CTA buttons' widespread use and because of their importance for all companies that sell goods or services from their website, you could claim that almost all e-commerce sites today have one.

The efficiency of a CTA button is calculated by dividing the number of clicks with the number of visitors and is called conversion rate. Thus, by keeping track of the conversion rate it is possible to measure, for instance, how effectively a site is meeting a business goal. Stressing the importance of CTA buttons even more, a small change in conversion rate, as for instance, one-tenth of one percent, can result in a revenue increase of ten percent or more [2].

It is probably due to the possible increase in revenue, suggestions of how you boost your conversion rate, is a widely discussed subject in social media today. Some companies claim that they can increase their conversion rate through

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 111–119, January 2015.

changing the color on their CTA buttons[1]. This claim has never been proven, but there is research that points to the fact that color does influence how we are affected by an interface.

For instance, color is the most effective visual variable when wanting to increase the visibility of a symbol [3]. Moreover, there are studies that emphasize the importance of using sharp color contrast on CTA buttons to increase the visibility aspect [4]. There are also studies examining the impact of using different font colors and background colors. One example is a study where four different background and font color combinations were tested. The color combinations in this study were: black text on white background, white text on black background, light blue text on dark blue background and cyan text on black background. The result showed that the most preferred combination led to higher ratings of aesthetic quality and intention to purchase [5].

One interpretation of this result is that color can influence a user to be more willing to buy something and if so, it probably also influence which buttons he or she clicks. Additionally, Don Norman, one of the most prominent researchers within the Human Computer Interaction field, has recently focused on the need to consider emotions and aesthetics in design [6]. Studies have been made to figure out general color preferences but their accuracy has been questioned [7]. Moreover it is known that our preferences are likely to change over time as a result of social and cultural influences [8].

The aim of this study is to investigate if a user rather clicks a button in a color they like than a button in a color they dislike or like less. This is an area within user interface design that is still unexplored in today's research.

Furthermore, we believe, it is an area in which many companies offer services and where companies spend big money. We therefore hope that this study is a contribution that is both sought for and could make a big difference in future user interface development.

The study is conducted by using a web form on which the user is asked to answer several different questions about what they like and dislike. In the first question the user is asked to choose which one of four showed colors, (red, blue, green and yellow), they like the most and least. The following tasks consists of simple questions that has nothing to do with colors. As the last task the user is asked to click one of two CTA buttons. The buttons the user can choose from will have the colors that they previously ranked as most preferred and least preferred.

2 Method

In order to test whether users rather click a button in a color they like than a button in a color they dislike or like less, a user study were conducted. Randomly chosen people were asked to join in a short anonymous test through email or by

[1] Smashing Magasine, A Field Guide to Usability Testing, Smashing eBook 21. Freiburg, Germany, Smashing Media GmbH, 2012, 13. [E-book], Google Books

a verbal invitation by the researcher. If the person agreed, they were sent a link to a website on which the form shown in Figure 1 was displayed.

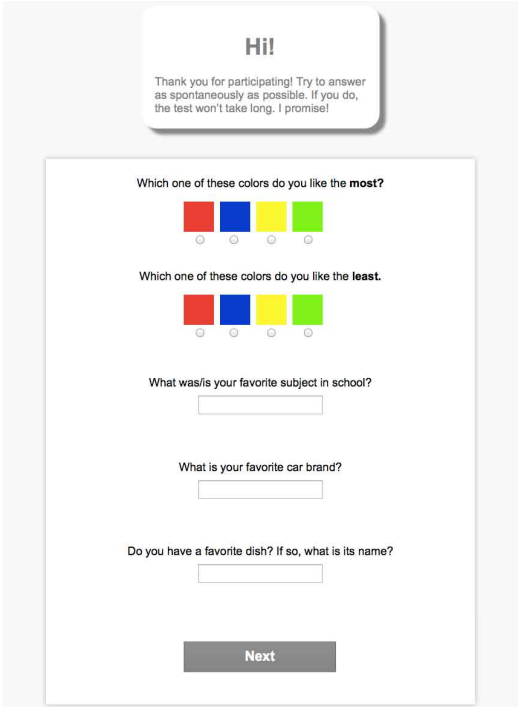


Fig. 1. Image of the first interface view used when testing. (The text shown in the figure is translated from Swedish to English.)

On the form the user answered several different questions. As the first question the user was asked to choose the color that he or she likes best among the four colors displayed. Similarly, the second question consisted of asking the user which color he or she likes least of the colors displayed. Then questions used for distracting the user were asked in order to disguise the aim of the study and to prevent the user from remembering what color he or she specified as his or hers favorite color. The questions used for distraction were about the users' favorite subject in school, their favorite car brand, favorite dish and gender. As the last task the user was asked to click one of two CTA buttons to quit. The buttons the user could choose from, had the colors he or she previously specified as the most preferred and least preferred, as shown in Figure 2.

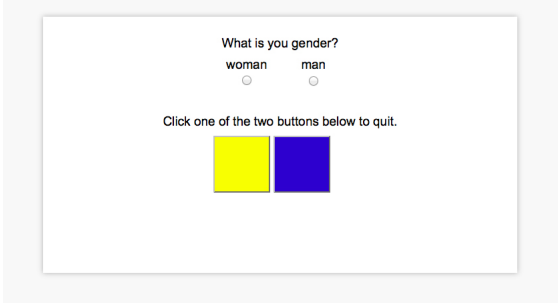


Fig. 2. Image of the second interface view used when testing containing the CTA buttons. (The text shown in the figure is translated from Swedish to English.)

After clicking one of the quit buttons, a new page was shown. On this page the user were invited to explain why he or she clicked the button he or she did on the previous page (see Figure 3).

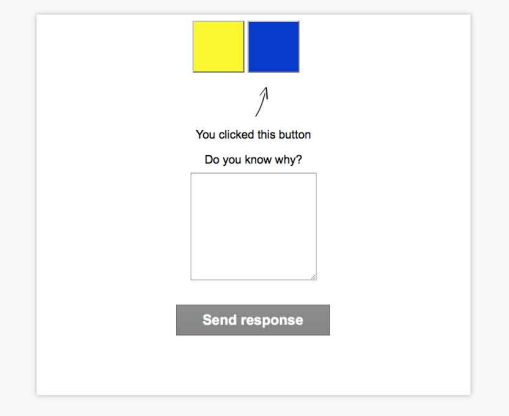


Fig. 3. An image of the last interface view used when testing. (The text shown in the figure is translated from Swedish to English.)

All data from the form was then gathered and analyzed. The data from the last question was categorized in a suitable number of categories depending on the spread of the different answers. Answers that were unique and that did not fit into a category, were put in a category named "Other responses".

The design of the test form that were used in the study, is based on research within interface design. As mentioned earlier, high color contrast increases the visibility aspect. It is therefore an advantage to have a CTA button with a high contrast in comparison with the rest of the interface. [4]. Due to this visibility aspect, the interface used in the study were colorless apart from the colors shown when asking which color the user likes and dislikes and the color used on the final CTA buttons (see Figure 1, 2 and 3). Additionally, findings show that the best place to put a CTA button, generally is, in the lower right corner [9].

Furthermore, it is believed that the result may be affected by if the user is right-handed or left-handed, when using a small mobile device. This, because one of the two CTA buttons then will be closer too their thumb and therefore is believed to be more convenient to click. Because of these two findings, the CTA buttons in this study were placed in the middle of the screen so that none of the buttons would be closer to neither the lower right corner nor the users thumb. Furthermore, the CTA button with the color the user specified as their most preferred, were randomly set to the right or left side of the other CTA button, so that their placement would not affect the result. The colors that the user could choose from in the form were the primary colors red, blue, yellow and green. In this test, equivalent to the RGB values FF3300, 0033CC, FFFF00 and 00FF00. Those colors were chosen because they have been used in previous color studies and because of their high affective value [8] [10].

The interface of the test form were implemented in Html5 and was made to be responsive so that the test form scaled with the size of the screen. In this way the test form looked almost exactly the same independently of whether the test was performed on a mobile phone, on a lap top, or on a desk top computer. Additionally, the responsive design made the test accessible for users with different devices and make it easy to participate in the study. The structure of the design was created such that the interface differed as little as possible between different browsers and screen sizes. Most importantly, the CTA buttons that the user could choose from in the last task, were shown in the middle of the screen, independently of its size.

3 Result

In total 95 persons participated in the test, out of which 83 persons, 87 % clicked the button with the color they specified as their favorite color. Of the 12 people that chose to click the button having the color they least preferred, the result is mainly divided by green and yellow as shown in Figure 4.

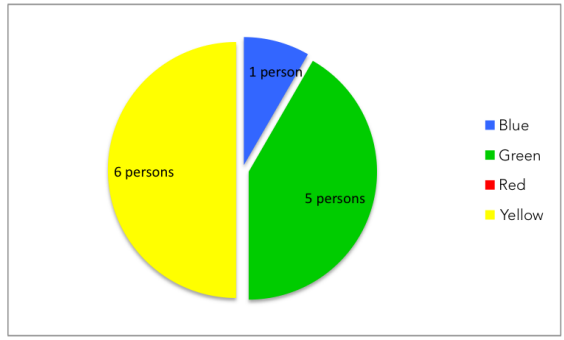


Fig. 4. Diagram showing which color the users that did not click the color they preferred, clicked. As for example none of the 12 users clicked the red button.

Overall, the button with the color that most users clicked on was blue as shown in Figure 5. After green, blue followed red, and finally yellow, as the color the fewest number of users clicked.

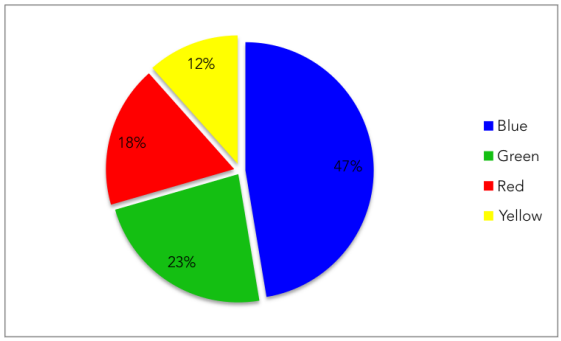


Fig. 5. Chart summarizing in percentage which button color all participants clicked. For example 45 out of 95 participating users, i.e. 47 %, clicked on a button with the color blue.

When categorizing the answers to the question why the user clicked the button they did, following categories were found suitable: "Because green means done/go", "Button was to the right", "I like the color/dislike the color of the other button", "It was more appealing", "Because I do not like that color", "Because I chose that color as my favorite" and "Other responses". As shown in Figure 6, users answered that they clicked the button because they liked the color of the button or that they did not like the color of the other button. The next most common answer was that they clicked the button because it looked better or was more appealing. The third most common answer was that they clicked the button because they had chosen that color as their favorite one. A few users also answered that it was because that color was to the right and because green means go or done. One person answered that it was because he or she did not like that color. Five users chose not to answer this question and were left out from the chart.

3.1 Study limitations

The aim of distracting the user from being affected by their choice of color in the first task when choosing which CTA button to click, may have failed. The three questions asked may have been too few and too simple to distract the user. Though, if choosing to use more and harder questions the test will take longer time and was believed to attract fewer participants. As mentioned earlier, our color preferences are likely to change over time [8]. It is therefore also important that the user is invited to choose between the two CTA buttons before he or she changes his preference for color.

Additionally, the sentence "Click one of the two buttons to quit", may also affect the users choice because of the fact that red, in western cultures, generally

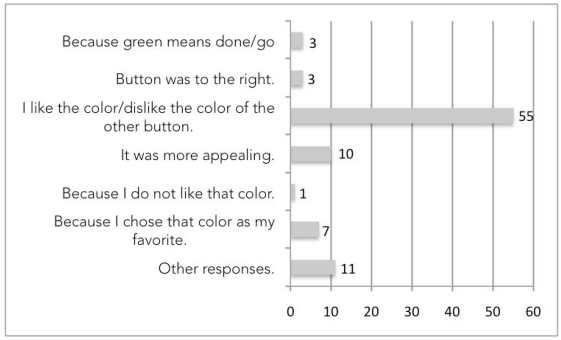


Fig. 6. Categorized answers from users when asked why they clicked the button they did. Bars showing number of answers in each category.

means stop [11][3][12]. Furthermore, it is also known that green means "safety" or "go" [3][12], which therefore, also can affect the users choice. Another limitation is the fact that it is proven that the best place to put a CTA is in the lower right corner. Even when placing the CTA buttons in the middle of the screen, one of the buttons will always be closer to the lower right corner.

4 Discussion

The result shows that 87 % of the users clicked the CTA button with the color they specified as the one they preferred the most. Furthermore, 61 % of the users that answered the question why they chose to click the CTA button they did, answered that they clicked it because they liked the color or disliked the color of the other button. Additionally, 11 % answered that they chose button by how good it appealed to them. Knowing that the buttons looked exactly the same, besides their different colors, this answer can be interpreted as if they chose button for which color that appealed more to them. Combined, both the results mentioned above, strengthens the hypothesis that we are more willing to click a button in a color we like, than a button in a color we dislike or like less. In contradiction, seven of the 95 users, i.e. 7,4 % answered that they clicked the CTA button they did, because they earlier had chosen that color as the one they preferred most. This answer implies that some or all users, got affected by their answer in the first task, which reduces the credibility of the result.

Regarding the limitation that the user could be affected by the fact that red means stop, is not seen as something that affected the result substantially. Mostly because none of the 12 users that clicked the button with the color they had specified as the one they least preferred, clicked the red button. Additionally, none of the users mentioned that red means stop or quit in their answer. On the contrary, participants clicked the green button and also mentioned in their answer that green means go or done. This result may imply that some users chose which button to click, depending on the meaning of the color, rather than because of their fondness of the color.

5 Conclusion

In summary, the result of the study indicates that the majority of the participants rather click a button in a color they like than a button in a color they dislike or like less. Though, an alarmingly large number of users mentioned that they clicked the button they did because they earlier had chosen that color as their favorite. Further studies is therefore required before any general conclusions can be drawn.

6 Future work

In general, when you ask women to rate how much they like different colors, they tend to rate them lower than men [7]. Thus there may be reasons to believe that women are affected differently by color than men. Additionally, if a woman rates a color lower than a man, could it also mean that the woman is less affected by that color? Those questions are believed to be important in the future and could be investigated in future studies related to the one presented in this paper.

References

[1] Gube, J.: Call to action buttons: Examples and best practices. Smashing Magazine (2009)
[2] Garrett, J.: Elements of User Experience,The: User-Centered Design for the Web and Beyond. Voices That Matter. Pearson Education (2010)
[3] Derefeldt, G., Swartling, T., Berggrund, U., Bodrogi, P.: Cognitive color. Color Research & Application **29**(1) (2004) 7–19
[4] Öztürk, O., Rızvanoğlu, K.: Selection and implementation of navigation and information search strategies in bank web sites: Turkish case. In Marcus, A., ed.: Design, User Experience, and Usability. Web, Mobile, and Product Design. Volume 8015 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 284–293
[5] Hall, R.H., Hanna, P.: The impact of web page text-background colour combinations on readability, retention, aesthetics and behavioural intention. Behaviour & Information Technology **23**(3) (2004) 183–195
[6] Norman, D.: Emotion & design: Attractive things work better. interactions **9**(4) (July 2002) 36–42
[7] Guilford, J.P., Smith, P.C.: A system of color-preferences. American Journal of Psychology **72** (1959) 487–502
[8] Terwogt, M.M., Hoeksma, J.B.: Colors and emotions: Preferences and combinations. Journal of General Psychology **122**(1) (1995) 5
[9] Hernandez, A., Resnick, M.L.: Placement of call to action buttons for higher website conversion and acquisition: An eye tracking study. Proceedings of the Human Factors and Ergonomics Society Annual Meeting **57**(1) (2013) 1042–1046

[10] Osgood, C.E., Miron, M.S., May, W.H.: Cross-cultural universals of affective meaning. University of Illinois Press Urbana (1975)
[11] Brown, C.M.: Human-computer interface design guidelines. Intellect Books (1999)
[12] Russo, P., Boor, S.: How fluent is your interface?: Designing for international users. In: Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems. CHI '93, New York, NY, USA, ACM (1993) 342–347

Towards Analyzing the Impact of Semantic Highlighting on Programming Productivity

Anna Viklund

Department of Computing Science
Umeå University, Sweden
tm09avd@cs.umu.se

Abstract. This paper explores semantic highlighting, and whether this can improve programming productivity. Since colors can help the human brain to categorize information, code highlighting is expected to make it easier to deal with code. Semantic highlighting attempts to color elements based on the semantics of the code alone, in contrast to regular syntax highlighting, which colors elements based on the syntax only. We performed a study where the relation between different types of code highlighting and productivity was analyzed. In the study, we measured the time it took for the test persons to solve tasks with different highlighting methods. The result showed that a majority of the test persons produced better results when programming with semantic highlighting only. A possible explanation to the results could be that semantic highlighting categorizes information in a more relevant way than regular syntax highlighting.

1 Introduction

Syntax highlighting is a well known text editor feature that displays source code in different colors in order to help when programming. Normally the highlighting is based on the syntax only, and the actual meaning of the code is ignored. Lately, some text editors have introduced a new highlighting method called *semantic code highlighting*, which is a feature that includes the meaning of the code in the highlighting process. In semantic coloring, the editor tries to point out what is important about the code, rather than displaying syntactical differences.

Since colors can help the human brain to categorize information [1], code highlighting is expected to help programmers in their work. Semantic code highlighting attempts to categorize information in a more relevant way than regular syntax highlighting, which means that this type of highlighting has the possibility to improve the productivity for programmers even more. Therefore, the question that will investigated in this paper is whether semantic code highlighting will ease the work for programmers more than regular syntax highlighting.

Understanding existing programs is one of the most time-consuming tasks for programmers. Often, the code itself is the only source of information. Although understanding code is time-consuming, it is essential for reusing and debugging

S. Bensch, T. Hellström (Eds.): Umeå's 19th Student Conference in Computing Science USCCS 2015.3, pp. 121–130, January 2015.

existing code [2, 3]. Various research on code comprehension has resulted in theories about the cognitive processess involved. One conclusion that has been drawn is that graphical representation of the code can enhance understanding of existing programs [2]. Since code highlighting is a type of graphical representation it is likely that this is a helpful feature for programmers.

To answer the research question, we performed a study where the relation between different types of code highlighting and productivity was analyzed. In order to do this, the time for programmers to solve different tasks was measured. Productivity is closely related to effort [4], which can be measured in time. In software engineering, the 1990s and beyond can be viewed as the quality and efficiency era [5], which make productivity a relevant aspect to study.

2 Background

2.1 Categorization Based on Colors

Categorization is one of the most fundamental processes of human cognition. It is defined as the mental processes by which the brain classifies objects, and is essential when gathering knowledge about the world [1]. Objects can be categorized in different ways, for example based on attributes such as shape and color. In this manner, visual perception is often involved in the categorization process. How the brain processes visual information is not yet fully understood [6]. However, it is confirmed that categorical perception based on colors is an inborn human ability [1].

2.2 Code Highlighting

Code highlighting is a feature that makes use of the human ability to categorize objects based on color. By coloring elements, the understanding of the code is enhanced. This can be done based on the syntax or the semantics of the code.

The syntax of a programming language is the set of rules that determines which combination of symbols the programmer has to use to write correctly structured code [7]. In order to highlight the code based on the syntax, the editor has to be syntax aware, which means that the editor has to analyze the code based on the syntax in some way. Most code editors achieve this by a continuously working parser that categorizes the text based on the syntax [7]. The editor can then display the code in different colors based on the categorization, as shown in Figure 1.

The first syntax-aware code editors were invented in the 1960s. The idea of syntax highlighting overlaps with the idea of syntax-directed editors. Syntax-directed editors are document editors that are aware of the underlying syntactic structure and the hierarchy of the text. In 1969, Wilfred Hansen created a code editor called Emily based on this concept. Emily provided advanced code-completing options to the programming, which actually made it impossible to write syntactically incorrect code [8].

```
int main(){
  int i,j;
  int n = 10;
  int array[n];

  //Fill array with random numbers between 1 and 10
  srand(time(NULL));
  for(i=0; i<n; i++){
    array[i] = rand() % 10 + 1;
  }

  //Display array
  cout<<"Start array:"<<endl;
  for(int i=0; i<n; i++){
    cout<<"\tarray["<<i<<"] = "<<array[i]<<endl;
  }
  cout<<endl;

  //Bubble sort
  int temp;
  for(int i=0; i<n-1; i++){
    for(int j=0; j<n-i-1; j++){
      if(array[j]>array[j+1]){
        temp=array[j];
        array[j]=array[j+1];
        array[j+1]=temp;
      }
    }
  }
```

Fig. 1. An example of syntax highlighting in the editor Sublime Text with the color theme iPlastic. The code is displayed in different colors based on the syntax. For example, all string literals are green.

Around 1985, a text editor called LEXX was created by Mike Cowlishaw. This was one of the first text editors that used live parsing to get structure information. The information was used to present the text with different colors, fonts and formatting in order to make it easier to deal with the code. Thanks to the live parsing technique, LEXX could handle a variety of structured data, and not only one particular programming language [9]. In this manner, LEXX was a general-purpose editor.

Syntax-based coloring turned out to be a major improvement for code editors [9]. Even if the highlighting is not a part of the actual text meaning, it serves to reinforce it, and today most editors provide this feature. However, the help syntax highlighting provides is limited, since it is entirely based on what the code looks like. To overcome this limitation, knowledge about the actual meaning of the code is required. This is where semantic highlighting comes in.

In contrast to syntax highlighting, semantic highlighting puts colors to elements based on the semantics, instead of highlighting the obvious, such as every instance of a loop or the word function. These ideas are relative new, and there are quite few implementations of this feature today. The problem with semantic coloring is that it is more difficult to highlight elements based on properties such as context, in contrast to highlighting based on simple keywords. The actual meaning of the code must be analyzed in some way, which can be a complex task. However, some attempts to semantic coloring has been done. For example, Valerij Primachenko has implemented a plugin for Sublime Text called Colorcoder [1]. Colorcoder is a package that makes it possible to apply semantic parsing

[1] https://github.com/vprimachenko/Sublime-Colorcoder

on an existing coloring theme in Sublime. For example, variable and function names get their own color, as shown in Figure 2. Colorcoder uses a CRC (Cyclic Redunancy Check) hash technique to give variables with similar names distinct colors. The variable-name highlighting is a feature that might help following the logical structure of the code and see how data flows in the program, since it makes it possible to distinguish a variable without reading its full name.

```
4   int main(){
5     int i,j;
6     int n = 10;
7     int array[n];
8
9     //Fill array with random numbers between 1 and 10
10    srand(time(NULL));
11    for(i=0; i<n; i++){
12      array[i] = rand() % 10 + 1;
13    }
14
15    //Display array
16    cout<<"Start array:"<<endl;
17    for(int i=0; i<n; i++){
18      cout<<"\tarray["<<i<<"] = "<<array[i]<<endl;
19    }
20    cout<<endl;
21
22    //Bubble sort
23    int temp;
24    for(int i=0; i<n-1; i++){
25      for(int j=0; j<n-i-1; j++){
26        if(array[j]>array[j+1]){
27          temp=array[j];
28          array[j]=array[j+1];
29          array[j+1]=temp;
30        }
31      }
32    }
```

Fig. 2. An example of semantic highlighting in the editor Sublime Text with the color theme iPlastic and the package Colorcoder installed. The code is displayed in different colors based on the semantics. For example, variable and function names get their own color.

2.3 Measuring Productivity in Programming

Productivity is defined as the amount of output produced per unit of input used. In software development, the output is the size of the product produced. The input is the effort required to develop the product. Therefore productivity can, in the case of software development, be defined as the ratio between size and effort [4].

To estimate productivity, the product size and the project effort must be measured. This can be done in several ways, and different methods often have both advantages and disadvantages. For example, in some cases productivity can be measured in lines of code produced per day [4]. This is easy to measure and can also be done consistently. However, the method is far from perfect since lines of code not necessarily reflects information about the quality of the code and the resulting product. There are lots of theories about measuring productivity, but no standard model that can be applied in general.

3 Method

In the study, the time for solving different tasks was measured, and regular syntax highlighting was compared to semantic highlighting. Each person in the test group solved three tasks. Every task consisted of a program and the test persons were asked to predict a print out at the end of each program. In the first task (Task 1) the code was not highlighted at all. In the second task (Task 2) the code was highlighted with regular syntax highlighting. In the third task (Task 3) the code was semantically highlighted.

The test group consisted of six students from the 4th year of the Master of Science Programme in Computing Science and Engineering at Umeå University, Sweden. This test group was considered appropriate for the study since they are a homogenous group with a similar level of knowledge in programming. They all get in touch with programming on a daily basis and have more or less same experience in computing science.

All tasks were created in the editor Sublime Text. For the task with regular syntax highlighting the theme iPlastic was used. For the task with semantic highlighting the same theme was used, but with the Colorcoder package by Valerij Primachenko included.

As mentioned earlier, the test persons were asked to predict a print out at the end of each program. The print outs were based on three variable values. The time it took for the test persons to predict correct values was measured. The number of errors before the correct values were predicted was also noted. To avoid stress, the test persons were not aware of that time and number of errors were measured during the test.

To avoid that the difficulty of the tasks would affect the result, all programs were constructed in a similar way. For example, all tasks consisted of five variables, two functions, one for-loop and three if-statements. To prevent the test persons from building up an experience from one task to another, different variable values and operations were used. In some cases the order of code blocks varied. To test the degree of difficulty of the tasks, they were solved by an independent person beforehand with no highlighting at all.

The reason for letting the test persons predict variable values, and not find syntax errors for example, was that semantic highlighting is expected to help programmers to follow the logical structure of the code and see how data flows in the program, as mentioned earlier. The tasks were chosen considering the purpose of semantic highlighting.

The tests were performed online in order to get a more accurate time measurement. The test was created using HTML, JavaScript, PHP and a MySQL database. Figure 3 shows a screenshot from a part of the test; the syntax highlighting task (Task 2). The test persons were allowed to take notes with paper and pen during the test.

Since the purpose of the paper is to compare syntax highlighting with semantic highlighting, the time it took for the test persons to solve the second task and the third task was compared. To avoid individual programming knowledge to affect the result, the reduction in required time with semantic highlighting,

Fig. 3. A screenshot from the online test. The screenshot shows one part of the test; the syntax highlighting task (Task 2 of 3).

compared to syntax highlighting was calculated for each test person. The time to solve the task with semantic highlighting was also compared to a mean value of the time with no highlighting and syntax highlighting. As mentioned earlier, productivity is defined as the amount of output used per unit of input used. In the study, all test persons produced the same output (predicting correct variable values), but achieved this with different effort (which was measured in time).

4 Results

Table 1 shows the time it took for the test persons to solve the different tasks. Five of six persons solved the task with semantic highlighting (Task 3) faster compared to the task with syntax highlighting (Task 2). Table 2 shows the number of errors in each task. No improvement concerning the number of errors could be noticed when comparing semantic highlighting with syntax highlighting.

Figure 4 shows the reduction in required time with semantic highlighting compared to syntax highlighting. The mean reduction among the test persons was 27,49%. Figure 5 shows the reduction in required time with semantic highlighting compared to a mean value of no highlighting and semantic highlighting. The mean reduction among the test persons was 26,13%.

Test person	**Task 1**	**Task 2**	**Task 3**
Person 1	129 s	104 s	114 s
Person 2	253 s	194 s	152 s
Person 3	230 s	500 s	152 s
Person 4	84 s	112 s	82 s
Person 5	129 s	175 s	125 s
Person 6	152 s	143 s	103 s

Table 1. Time for test persons to solve different tasks

Test person	**Task 1**	**Task 2**	**Task 3**
Person 1	0	0	0
Person 2	0	0	0
Person 3	0	0	0
Person 4	0	1	2
Person 5	0	0	0
Person 6	0	0	0

Table 2. Number of errors in different tasks

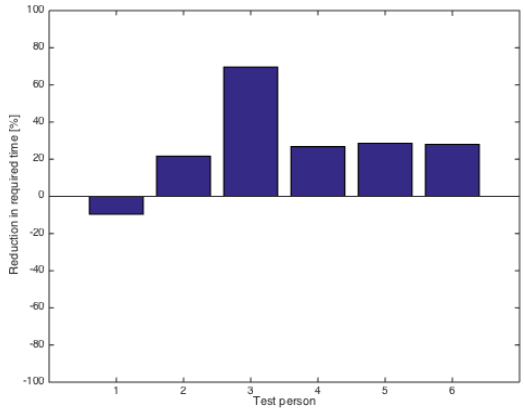


Fig. 4. The reduction in required time with semantic highlighting, compared to syntax highlighting

5 Discussion

5.1 Conclusions

In the study, almost all test persons received a better result with semantic highlighting. Theories about human cognition can explain this tendency. As mentioned in Section 2.1, colors can help the human brain to categorize information. Therefore, a more relevant coloring of elements makes it easier for the test persons to deal with the code, and solve the tasks in a more efficient way.

No actual conclusion regarding semantic highlighting can be drawn, although the result indicates that semantic highlighting has the possibility to help programmers in their work. The major reason for this is the low number of test persons involved in the study. More than six test persons are required to make statistical analyzes on the result and draw conclusions about the hypothesis.

5.2 Limitations

Besides the low number of test persons, the study has some other limitations that makes the result quite weak. Possible differences in the difficulty of the tasks is one such limitation. Even if the tasks were tested in advance with no highlighting, there is still a risk that the difficulty of the tasks differed since they were only tested by one single person. This could have affected the results.

The order of the tasks could also have influenced the results. In the study, the test persons solved the semantic highlighting task last of all tasks. Since the tasks were created in a similar way, there is a risk that the test persons could have built up an experience during the test, which made the last task easier to solve compared to the previous ones. To minimize this risk, different variable values and operations were used in the tasks, but there is still a risk that the results were affected by the order of the tasks.

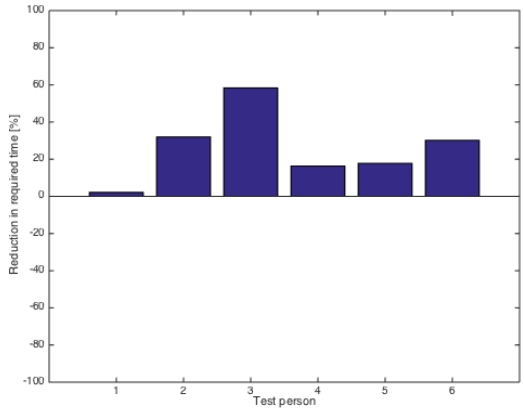


Fig. 5. The reduction in required time with semantic highlighting, compared to no highlighting and syntax highlighting

5.3 Future work

One way to improve the study would be to randomize the highlighting method, but keep the code in the tasks. Each highlighting method could still be tested on each test person, but the order of the highlighting methods could differ. In this way, the degree of difficulty on the different tasks would not have mattered. Even if the last task would be easier to solve compared to the other ones, this would not have an impact on the results. This could also eliminate the risk of building up an experience during the test.

In the future, it would be interesting to test semantic highlighting on other types of tasks since this study only deals with one type of task. It would also be interesting to test different implementations of semantic highlighting. Semantic highlighting is not yet fully investigated, and we will probably see large advancements in knowledge in the near future. Since programs get more and more complex, tools that makes it easier to deal with code will be required. Therefore, semantic highlighting is an important area for future studies.

References

[1] Cohen, H., Lefebvre, C.: Handbook of Categorization in Cognitive Science. Elsevier Science (2005)

[2] Storey, M.A., Fracchia, F., Müller, H.: Cognitive design elements to support the construction of a mental model during software exploration. Journal of Systems and Software **44**(3) (1999) 171–185

[3] Storey, M.A., Wong, K., Müller, H.: How do program understanding tools affect how programmers understand programs? Science of Computer Programming **36**(2-3) (2000) 183–207

[4] Kitchenham, B., Mendes, E.: Software productivity measurement using multiple size measures. IEEE Transactions on Software Engineering **30**(12) (2004) 1023–1035
[5] Kan, S.: Metrics and Models in Software Quality Engineering, Second Edition. Addison Wesley (2002)
[6] Fairchild, M.: Color Appearance Models, Second Edition. John Wiley & Sons, Ltd. (2005)
[7] Friedman, D., Wand, M.: Essentials of Programming Languages, Third Edition. The MIT Press (2008)
[8] Hansen, W.: Creation of Hierarchic Text with a Computer Display. PhD thesis, Stanford Unviersity (June 1971)
[9] Cowlishaw, M.: Lexx - a programmable structured editor. IBM Journal of Research and Development **31**(1) (1987) 73–80

Author Index