



Norwegian University of
Science and Technology

Magno: An Application for Detection of Dyslexia

Dyslexia and Interface Design

Thea Hove Johansen
Maja Kirkerød

Master of Science in Computer Science

Submission date: June 2017

Supervisor: John Krogstie, IDI

Norwegian University of Science and Technology
Department of Computer Science

Problem Description

Research at the department of Psychology at NTNU indicates that an underlying reason for dyslexia and dyscalculia may be a dysfunction in visual processing. Specifically, some nerve cells that move information from the eyes to the visual processing centre in the brain are slower in some dyslexics compared to a control group. These nerve cells are called magnocells and their main responsibility is to perceive rapid changes in the environment.

An app used to test visual information processing in order to find evidence of such interference has been previously developed, and the assignment is to evaluate, further develop, and implement the user interface of this application.

Supervisor: Professor John Krogstie

Sammen drag

Dysleksi er en funksjonshemming som har fått mer og mer oppmerksomhet de siste årene. Det å være i stand til å oppdage dysleksi tidlig er viktig for å redusere de negative effektene dysleksi har på mennesker diagnostisert med sykdommen. Denne masteroppgaven omhandler det å utvikle og implementere et grafisk brukergrensesnitt til en eksisterende applikasjon. Den eksisterende applikasjonen er ment å bli brukt som et verktøy for tidlig deteksjon av dysleksi. Brukergrensesnittet har blitt utviklet ved å identifisere en målgruppe, skape og teste prototyper av brukergrensesnittet, og implementere et ferdig design. Resultatet av dette prosjektet er en applikasjon med et fullt funksjonelt brukergrensesnitt. Brukergrensesnittet har en System Usability Scale (SUS) poengsum på 92.7, og er mer brukervennlig enn brukergrensesnittet til den originale applikasjonen. Denne masteroppgaven har gitt den originale applikasjonen et brukergrensesnitt som gjør det mulig for alle med grunnleggende leseferdigheter å teste seg selv for dysleksi. Forhåpentligvis vil denne applikasjonen bli lansert og distribuert til brukere, og kunne senke gjennomsnittsalderen for deteksjon av dysleksi, ved hjelp av vårt brukergrensesnitt.

Abstract

Dyslexia is a disability that has accumulated more and more attention over the latest years. Being able to detect dyslexia early on is vital for decreasing the negative effects dyslexia has on people diagnosed with this disability. This master's thesis deals with creating a graphical user interface of an existing application meant to be used as a screening tool in the process of detecting dyslexia. This is done by identifying a target user group, creating and testing prototypes, and implementing a finished design. The result of this process has been an application with a fully functional graphical user interface (GUI). The GUI has a System Usability Scale (SUS) score of 92.7, and increased usability compared to the original application. This master's thesis has provided this application with a user interface that allows anyone with basic reading skills to test themselves for a risk of having dyslexia. Hopefully, when this application is deployed, the average age of dyslexia detection will be lowered significantly, with the help of our user interface design.

Preface

The work presented in this thesis is a part of our master's degree in computer science at the Norwegian University of Science and Technology (NTNU). The project is conducted under the Department of Computer Science at the Faculty of Information Technology and Electrical Engineering, under the supervision of Professor J. Krogstie. The project was done in collaboration with Professor H. Sigmundsson at the Department of Psychology at NTNU.

We would like to thank Professor John Krogstie for his continuous support and advisory during the making of this master's thesis. Without his help, this thesis would not have been possible. We would also like to thank Professor Hermundur Sigmundsson for lending us his expertise in the field of dyslexia and dyslexia detection.

Our friends deserve to be thanked for being there for us during both good and bad times.

A special thanks is directed towards our families for their unprecedented support and for providing us with the foundation needed in order to complete our master's degree here at NTNU.

Trondheim, June 1, 2017



Maja Pedersen Kirkerød



Thea Hove Johansen

Contents

Problem Description	ii
Sammendrag	iv
Abstract	vi
Preface	viii
1 Introduction	2
1.1 Motivation	2
1.2 Project Objectives	3
1.3 Project Description	3
1.4 Thesis Outline	4
2 Research Approach	6
2.1 Research Questions	6
2.2 Research Methodology	7
2.2.1 Research Paradigm	7
2.2.2 Research Method	7
2.2.3 Participants and Stakeholders	8
2.3 Final Deliverables and Dissemination	9
3 Previous Work	10
3.1 Dyslexia	10
3.2 Dyslexia and Other Visual- and Motor Deficits	11
3.3 The Visual Magnocellular System	12
3.4 Detecting Dyslexia Today	14
3.5 App for Early Detection of Dyslexia	15
3.6 Dyslexia and Design	17
3.7 User Interface Design Process	18
3.8 The Final Requirement List	19

3.8.1	Functional Requirements	20
3.8.2	Non-Functional Requirements	21
3.9	Presentation of the Paper Prototypes	21
3.9.1	Views	21
3.9.2	Test Results and Further Work	27
4	Digital Prototype	30
4.1	Initial Digital Prototype	30
4.1.1	Changes From Paper Prototype	30
4.1.2	Usability Testing, Results and Changes	31
4.1.3	Changes to the Digital Prototype	34
4.2	Final Digital Prototype	35
5	Implementation	38
5.1	Application Overview	38
5.1.1	Application Appearance	38
5.1.2	Application Code	48
5.2	Design Choices	48
5.2.1	Structural Choices	48
5.2.2	Colour Choices	51
5.2.3	Navigation Structure	53
5.3	Final Graphic Profile	54
5.3.1	Colour Chart	54
5.3.2	Font	55
5.3.3	Logo and Icons	57
5.4	Software Architecture	58
5.4.1	Classes	58
6	Evaluation	62
6.1	Overall Evaluation	62
6.2	Evaluation of Technical Tools	63
6.3	Requirement Fulfilment	68
6.3.1	Functional Requirement Fulfilment	68
6.3.2	Non-Functional Requirement Fulfilment	71
7	Discussion, Conclusion, and Further Work	72
7.1	Discussion	72
7.2	Conclusion	73
7.2.1	Final Conclusion	75
7.3	Further Work	76

References	78
A Overview of the Existing Application	82
B Contract	96
C Questionnaire	98
D System Usability Scale	100
E Paper Prototypes	102
E.1 Paper Prototype 1	102
E.2 Paper Prototype 2	108
F Digital Prototype	114
F.1 Final Digital Prototype	114
G Finished Application	122
G.1 Home Screen	122
G.2 Tutorial	124
G.2.1 Step 1	124
G.2.2 Step 2	127
G.2.3 Step 3	133
G.2.4 Step 4	148
G.3 Tests	151
G.4 Result Screen	156
G.5 Settings	159

List of Figures

3.1	Images from the dynamic dot and form tests, testing coherent motion (left) and coherent form (right).	12
3.2	Improvement in writing, after using yellow glasses for one week. . .	13
3.3	Increase in literacy (reading age) after use of blue glasses.	14
3.4	Main menu.	15
3.5	Motion test.	16
3.6	Form fixed auto test.	17
3.7	Form random auto test at 100% coherency.	17
3.8	An example of the river effect (leftmost image), the blur effect (middle image) and the washout effect (rightmost image); commonly experienced by dyslexics [21].	18
3.9	The iterative design process [25].	19
3.10	The main menu view.	22
3.11	The first tutorial view.	23
3.12	The second tutorial view.	23
3.13	The third tutorial view.	24
3.14	The enter age view.	24
3.15	The test view (without dots and lines).	25
3.16	The score view.	25
3.17	The settings view for paper prototype 1.	26
3.18	The settings view for paper prototype 2.	27
3.19	SUS score for paper prototype 1.	28
3.20	SUS score for paper prototype 2.	29
4.1	SUS score for digital prototype.	33
4.2	The main view of the digital prototype and the first tutorial view.	35
4.3	The second and third tutorial screen of the motion test.	36
5.1	Home screen of the application.	39
5.2	The first step of the tutorial.	40
5.3	The second step of the tutorial.	42

5.4	The third step of the tutorial.	43
5.5	The third step of the tutorial.	43
5.6	The fourth step of the tutorial.	44
5.7	The test view.	45
5.8	Motion test result view.	45
5.9	Application settings.	47
5.10	Application settings; reset settings.	47
5.11	Navigational chart of application.	53
5.12	Application logo.	57
5.13	Application icon.	57
5.14	Class diagram	61
A.1	Main menu view.	83
A.2	Motion test at 100% coherency.	83
A.3	Illustration of the motion test at 50% coherency.	84
A.4	Form fixed auto test at 100% coherency.	84
A.5	Form fixed auto test at 50% coherency.	85
A.6	Form random auto test at 100% coherency.	85
A.7	Comparison between the auto and manual mode for the Form fixed test.	86
A.8	Comparison between the auto and manual mode for the Form random test.	87
A.9	Settings view.	87
A.10	Motion test settings.	89
A.11	Form test settings.	90
A.12	Screen settings.	91
A.13	Staircase settings.	92
A.14	Input settings.	93
A.15	Input settings key dialog.	93
A.16	Reset settings confirmation dialog.	94
E.1	The main view.	102
E.2	The first tutorial view after clicking on motion test.	103
E.3	The second tutorial view.	103
E.4	The last tutorial view.	104
E.5	The age view.	104
E.6	The test view (without dots and lines).	105
E.7	The score view.	105
E.8	The first tutorial view after clicking on form fixed test.	106
E.9	The second tutorial view.	106
E.10	The last tutorial view.	107

E.11	The settings view.	107
E.12	The app settings view.	108
E.13	The main view.	108
E.14	The first motion test tutorial view.	109
E.15	The second motion test tutorial view.	109
E.16	The third motion test tutorial view.	110
E.17	The age view.	110
E.18	The test view.	111
E.19	The score view.	111
E.20	The first form fixed test tutorial view.	112
E.21	The second form fixed test tutorial view.	112
E.22	The third form fixed test tutorial view.	113
E.23	The settings view.	113
F.1	The main view.	114
F.2	The first tutorial view of the motion test.	115
F.3	The second tutorial view of the motion test.	115
F.4	The third tutorial view of the motion test - before starting the trial.	116
F.5	The third tutorial view of the motion test - when first starting the trial.	116
F.6	The third tutorial view of the motion test - correct answer.	117
F.7	The third tutorial view of the motion test - wrong answer.	117
F.8	The third tutorial view of the motion test - completed trial.	118
F.9	The fourth tutorial view of the motion test.	118
F.10	The motion test.	119
F.11	The result view.	119
F.12	The settings view.	120
G.1	Home screen.	123
G.2	Motion test tutorial - step 1.	124
G.3	Form fixed test tutorial - step 1.	125
G.4	Form random test tutorial - step 1.	126
G.5	Motion test tutorial - step 2.	127
G.6	Motion test tutorial, "Instant click registration" disabled - step 2.	128
G.7	Form fixed test tutorial - step 2.	129
G.8	Form fixed test tutorial, "Instant click registration" disabled - step 2.	130
G.9	Form random test tutorial - step 2.	131
G.10	Form random test tutorial, "Instant click registration" disabled - step 2.	132
G.11	Motion test tutorial, prior to trial start - step 3.	133
G.12	Motion test tutorial, during trial - step 3.	134

G.13	Motion test tutorial, correct answer - step 3.	135
G.14	Motion test tutorial, wrong answer - step 3.	136
G.15	Motion test tutorial, completed trial - step 3.	137
G.16	Form fixed test tutorial, prior to trial start - step 3.	138
G.17	Form fixed test tutorial, during trial - step 3.	139
G.18	Form fixed test tutorial, correct answer - step 3.	140
G.19	Form fixed test tutorial, wrong answer - step 3.	141
G.20	Form fixed test tutorial, completed trial - step 3.	142
G.21	Form random test tutorial, prior to trial start - step 3.	143
G.22	Form random test tutorial, during trial - step 3.	144
G.23	Form random test tutorial, correct answer - step 3.	145
G.24	Form random test tutorial, wrong answer - step 3.	146
G.25	Form random test tutorial, completed trial - step 3.	147
G.26	Motion test tutorial - step 4.	148
G.27	Form fixed test tutorial - step 4.	149
G.28	Form random test tutorial - step 4.	150
G.29	Entered test - prior to start.	151
G.30	Entered motion test.	152
G.31	Entered form fixed test.	153
G.32	Entered form random test.	154
G.33	Completed test.	155
G.34	Motion test result screen.	156
G.35	Form fixed test result screen.	157
G.36	Form random test result screen.	158
G.37	App settings tab.	159
G.38	Advanced settings tab.	160
G.39	Advanced settings - dialog window appears when changing left patch key, right patch key, or back key.	161
G.40	Reset settings tab.	162

List of Tables

3.1	Functional requirements	20
3.2	Non-functional requirements	21
6.1	Functional requirements evaluation.	70
6.2	Non-functional requirements evaluation.	71

Chapter 1

Introduction

A well thought out user interface design is important like never before in today's society. No longer is the focus on heavy programs, packed with functionality. The point of having a lot of functionality is void when no one can use the program due to a poor GUI. It is said that the amount of code dedicated to frontend development and user interface now exceeds 50 percent [1].

Dyslexia is a disability heavily affecting a person's ability to read and comprehend the written word, without affecting the person's intelligence quotient (IQ). The importance of detecting dyslexia early on is undoubted, as proper intervention has up to 80 percent effectiveness when applied to children in the 1st or 2nd grade of primary school [2]. In addition, studies show that when teachers are properly trained in dyslexia detection and intervention, 90 percent of children with dyslexia can participate in a normal classroom environment [3].

1.1 Motivation

With the importance of user interface design and early dyslexia detection being as is, the combination of the two makes for an interesting and rewarding master's thesis. The opportunity to not only provide the public with a tool that enables early dyslexia detection not based on the ability to read or write, but also develop and implement a user interface that makes this tool easy to use, is unparalleled. Being able to contribute to the development of this program, is potentially contributing to the detection of several more cases of dyslexia, and hopefully increasing the quality of life of one or more individuals suffering from dyslexia.

Our thesis focuses primarily on developing the user interface of a program. The user

interface is the link between the user and the software, and is massively important to ensure usability and accessibility of the program functionality. Our motivation for focusing on this aspect of the development is to ensure that this program can be used by the masses, and is not forgotten due to poor usability.

Furthermore, this master's thesis provides the opportunity to contribute to the small field of designing user interfaces specifically directed towards dyslexic users. There is not a lot of research done in this field, and hopefully this thesis will supply the field with further observations and material.

1.2 Project Objectives

The main objective of this project is to provide a program meant to screen for dyslexia with a functioning user interface. In a larger sense, one might say that our objective is to ensure usability of the program, in order to promote use of the program. By tailoring the user interface of a program to meet the users needs, one lowers the threshold for taking the program into use. The ultimate objective of this project is to help in early dyslexia detection, and providing the masses with a tool to do so.

1.3 Project Description

This project is a continuation of the master's thesis "App for Early Detection of Dyslexia" conducted in the spring of 2016 [4], as well as the authors' specialisation project "Designing an Application for Detection of Dyslexia" conducted in the fall of 2016 [5]. "App for Early Detection of Dyslexia" was a master's thesis concerning the development and implementation of a digital test that might help in dyslexia screening. "Designing an Application for Detection of Dyslexia" was a specialisation project concerned with creating a concept for a user interface to go along with the implemented screening test. This project is concerned with further developing the user interface design, and implementation of this user interface. This project is meant to identify any problem areas in the existing user interface design, received from previous work, finding solutions to these problem areas, and implementing a finished user interface design.

1.4 Thesis Outline

This thesis consists of seven different chapters. This chapter is the first chapter, which includes the introduction. Chapter 2 contains information regarding the research approach used in this project, and chapter 3 introduces all previous work done in relation to this project. Chapter 4 contains a presentation of a digital prototype of a user interface that has been developed during this project. In chapter 5 the implementation process and the finished graphical user interface that has been implemented are explained. Chapter 6 contains an evaluation of the work done in this project, and chapter 7 encompasses discussion around the project, a conclusion and any further work that might be needed. In addition, this thesis includes an appendix with 7 chapters.

Chapter 2

Research Approach

In this chapter, we will describe the chosen research approach. In section 2.1 different research questions are presented, and in section 2.2 we will discuss the different selected research models and methodologies. The final deliverables are declared in section 2.3.

2.1 Research Questions

This master's thesis aims to evaluate the user interface design developed in the authors' specialisation project [5], improve the design if necessary, test the changes, and implement the finished design into an existing application for screening for dyslexia [4]. The research questions to complement this master's thesis description are as follows:

- RQ-1** What are the problem areas with the user interface design developed in the specialisation project?
- RQ-2** How can the interface design developed in the specialisation project be improved, in order to further promote usability?
- RQ-3** How can we compliment existing code with the implementation of the new user interface design?
 - RQ-3.1** What technical tools are best suited to aid in implementing a user interface design, based on previously implemented functionality?

2.2 Research Methodology

This section will discuss the research paradigm in which the research is conducted under, the chosen research method, and what participants and stakeholders are part of this research.

2.2.1 Research Paradigm

A research paradigm is the belief system under which the research is conducted. The research paradigm says something about how the research is to be interpreted, and will affect the test results and how they are to be viewed.

For this project, the research paradigm *interpretive research* is used. Interpretive research in information systems and computing means that the research is heavily based upon the social context in which the information system operates. In other words, the social processes by which it is developed and constructed by people and through which it influences, and is influenced by, its social setting.

In order to be able to both identify and correct problem areas in a user interface design, it is important to understand why they are problem areas. Different social settings enable different versions of “the truth”, and different views of what is perceived as the right way to do things [6].

In order to tailor an application and its user interface to specific user groups, it is important to understand the social setting in which these user groups reside, and how these social settings affect the users’ world view.

2.2.2 Research Method

A research method is the manner in which research is conducted, including the different steps along the way, and the manner in which results are collected.

This research will begin with a thorough review of the research the previous user interface design was based upon, and what problem areas exist with this design. Further, a new design is to be developed based upon the previously identified problem areas. This design is then to be converted into a digital prototype in order to enable testing in a realistic environment. A “*case study*” is to be conducted in order to collect data regarding the digital prototype. The case study will encompass conducting usability tests on the digital prototype. The results from these usability tests are then to be reviewed and quantified, and changes are to be

made to the design in order to complement the results found during the case study. The changed design is considered to be the finished user interface design. In order to implement the finished design, a review of the implementation of the existing application is to be conducted. When the different aspects of the existing code has been identified and understood, the finished design is to be implemented.

For data collection we are going to use a SUS form and a questionnaire, distributed in connection with controlled usability tests. The data will be used to review the different user interface designs. Observation during the usability tests is to be employed as well. The questionnaire and SUS form are to be the same as the ones used in the specialisation project [5], to enable easy comparison with previous work. The results from the usability tests should be divided into quantitative results that say something about the overall usability, and qualitative results that say something about critical problem areas and problems in understanding functionality. In order to review existing code, the code should be read and understood, and a literature review should be done in connection with the different java libraries that are to be used in the implementation.

2.2.3 Participants and Stakeholders

Participants of some form of research are the ones who have participated, contributed, or are otherwise featured in the research. Stakeholders are the ones who have something to either gain or lose depending on the outcome of the research.

The participants in this project are primarily the researchers, the authors of this master's thesis, who contribute with their work and research on the subject at hand. The project supervisor, Professor J. Krogstie, a professor at NTNU, contributes with his expertise and experience in research in the field of information systems, and is also a participant. Other than the researchers and the project supervisor, external experts on the subjects of dyslexia, dyslexia detection, user interface design, and software development have also been consulted. These external experts should also be viewed as participants of this project. Lastly, but not least, all test subjects who participate in usability tests during our case study are participants in the research. These test subjects are students at NTNU, in the age range from 18 to 25 years old. There is no need to collect personal data from these test subjects in order to determine usability of the application, meaning there is no great risk of ethical issues that need to be considered.

The stakeholders of this project are the researchers, as the research will impact the grade given, and we are interested in presenting work that will reflect our work

ethics and abilities. The project supervisor and NTNU will want this project to be conducted properly and with high quality, as the work reflects the quality of the school and its professors, and the publication will represent the quality and determination of the students who attend NTNU. Potential dyslectics, teachers, and doctors specialising in dyslexia and other reading disabilities will have something to gain as early detection of dyslexia minimises potential problems both inside and outside of the classroom. Detecting dyslexia early will make life easier for the ones who are diagnosed, teachers who have to teach them, and the doctors who treat and diagnose them. Psychology students and researchers can use this application as a tool and an aid in research concerning dyslexia.

2.3 Final Deliverables and Dissemination

This project will conclude in the delivery of this master's thesis. In addition a working version of the implemented application will be delivered, along with all vector graphics used to present the application, such as the application logo, and application icons. The master's thesis will show insight into the workings of the application, and the application itself can be used as is, or further developed in order to enable extended functionality. The vector graphics can be used in accordance with the graphic profile, presented in section 5.3.

Chapter 3

Previous Work

This chapter contains a summary of some of the work conducted in the specialisation project [5]. It provides background information affecting further research and development decisions. Section 3.1 gives a short description of what dyslexia is, and section 3.2 introduces the idea that some of the things causing dyslexia might lead to motor deficits as well. In section 3.3 the anatomical differences between the brains of known dyslexics and non dyslexics are presented. Section 3.4 contains an explanation of how dyslexia is detected in Norway today, and in section 3.5 we give a thorough walk through of the application that is the precursor to this project. In section 3.6, special measures that can be taken in order to tailor interface design to dyslexics are presented, and section 3.7 gives an overview of what a typical design process looks like. The requirements and paper prototypes developed during the specialisation project are presented in the last two sections; section 3.8 and section 3.9.

3.1 Dyslexia

The disability causing reading abilities to be significantly worse than expected given the IQ, is called dyslexia. It is a neurological syndrome and has no connection with a person's intelligence other than poor orthographic¹ skills [7]. There exist several definitions of dyslexia as it manifests itself a little differently for each individual. Some of the definitions are as follows:

“Dyslexia is a specific learning disability that is neurobiological in origin. It is characterised by difficulties with accurate and/or fluent word recognition and by

¹The study of spelling and how letters combine to represent sounds and form words.

poor spelling and decoding abilities. These difficulties typically result from a deficit in the phonological component of language that is often unexpected in relation to other cognitive abilities and the provision of effective classroom instruction. Secondary consequences may include problems in reading comprehension and reduced reading experience that can impede growth of vocabulary and background knowledge.”

- *International Dyslexia Association* [8]

“Dyslexia is a specific learning difficulty that mainly affects the development of literacy and language related skills. It is likely to be present at birth and to be life-long in its effects. It is characterised by difficulties with phonological processing, rapid naming, working memory, processing speed, and the automatic development of skills that may not match up to an individual’s other cognitive abilities.

It tends to be resistant to conventional teaching methods, but its effect can be mitigated by appropriately specific intervention, including the application of information technology and supportive counselling.”

- *British Dyslexia Association* [9]

“A general term for disorders that involve difficulty in learning to read or interpret words, letters, and other symbols, but that do not affect general intelligence.”

- *Oxford dictionary* [10]

The definitions have similarities, but they are not equal. A similarity is that none of these definitions mention anything about what the cause of dyslexia is, only the symptoms. There are many different opinions on what causes dyslexia, but it is generally agreed upon that dyslexia is a genetic disorder, and therefore hereditary [11].

3.2 Dyslexia and Other Visual- and Motor Deficits

There have been conducted several studies that link dyslexia to other motor skills. In 2005, H. Sigmundsson published a research paper suggesting a link between dyslexia and poor performance in reaction time tests. The tests conducted showed that dyslexics on average responded .19 seconds slower than test subjects without dyslexia [12].

A study conducted by Hansen et al., examined the connection between visual deficits in dyslexics and the dorsal stream function². They discovered that dyslexics were less sensitive to coherent motion in dynamic dot displays than a control group.

²The neural stream projected dorsally from the primary visual cortex in to the parietal lobe.

However, the dyslexics did not perform significantly different from the control group when tested in a static environment [13]. The dynamic dot test used in this study is the same as used in the application described in this thesis.

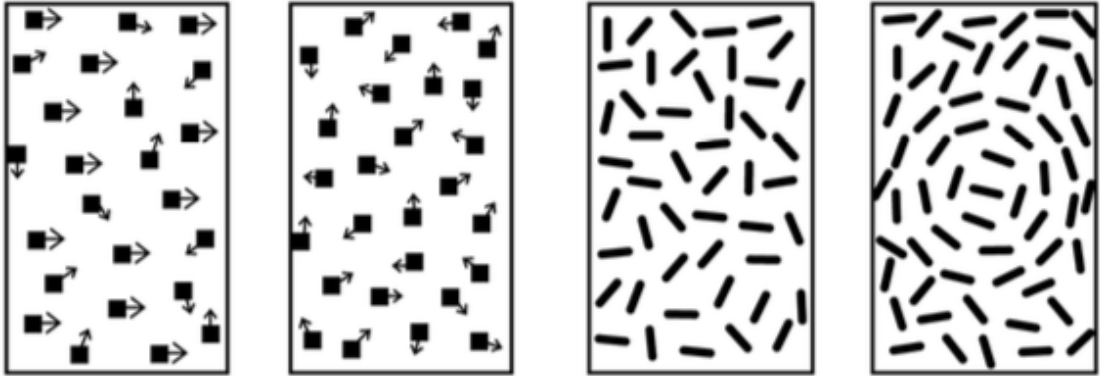


Figure 3.1: Images from the dynamic dot and form tests, testing coherent motion (left) and coherent form (right).

The dot kinematogram showed in figure 3.1 was also used in another study conducted by H. Sigmundsson, P. C. Hansen and J. B. Talcott, where they looked for a connection between developmental “clumsiness” and visual impairment. The results from this research showed that children considered as clumsy had a significantly higher threshold for the visual measures tested. This means that some clumsy children most likely have a visual impairment in addition to their obvious motor problems [14].

3.3 The Visual Magnocellular System

The dorsal pathway is one of the parts of the brain responsible for visual processing. Within the dorsal pathway, there are cells called magnocellular neurons that are specialised in detecting visual motion. Many dyslexics have reduced activation of the visual areas in the dorsal stream in response to moving targets. This reduced sensitivity to motion indicates reduced sensitivity of the visual magnocellular system in many dyslexics, crucial for visuomotor control.

In addition, in the average brain, there is some asymmetry favouring the left part of the brain, especially considering the dorsal pathway. Studies of the brains of known dyslexics show that dyslexics often lack this asymmetry. Small “brain warts”

(ectopias³) were also discovered clustered around the temporoparietal junction⁴ of the brain in these dyslexics. These warts are associated with widespread disruption of normal neurological connections.

The studies done in this field imply that some people with dyslexia also suffer from some kind of visual impairment, especially in relation to coherent motion detection [7].

Yellow filters can improve magnocellular function and improve reading, as magnocells are more sensitive to yellow light. This is exemplified in figure 3.2. Blue filters on the other hand, can help the letters keep still when reading [17]. The improvement of literacy when using blue tinted glasses can be seen in figure 3.3.

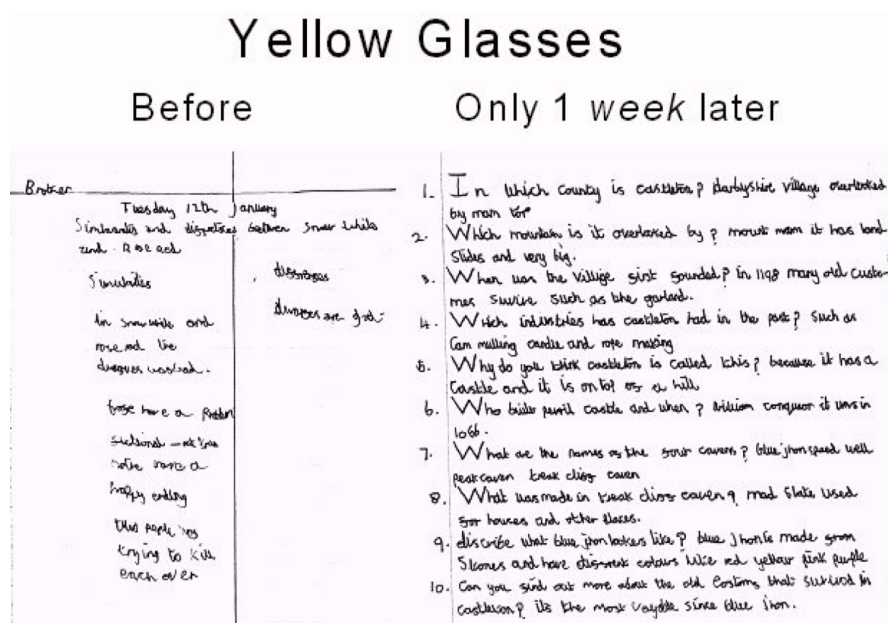


Figure 3.2: Improvement in writing, after using yellow glasses for one week.

³Malposition of an organ or structure [15].

⁴Relating to the temporal and the parietal bones or regions [16].

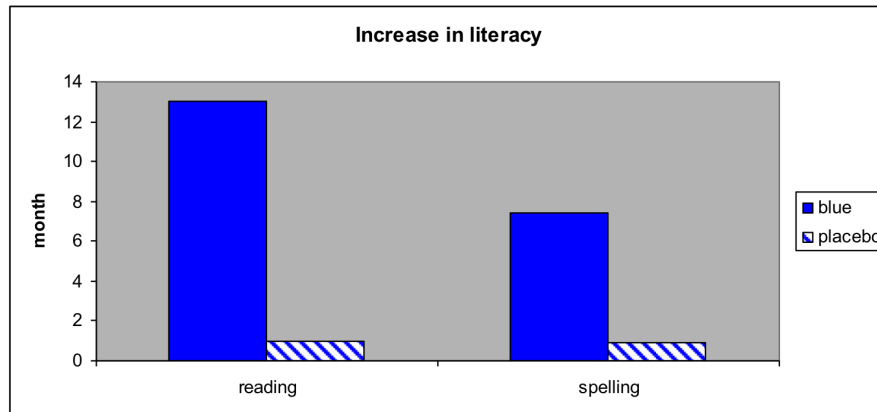


Figure 3.3: Increase in literacy (reading age) after use of blue glasses.

The differences in the brain function explained here might explain the fact that dyslexics score differently than a control group in the various tests described in section 3.2. As many dyslexics have a weakened magnocellular system, it allows for testing of signs of dyslexia by testing people's ability to detect motion and coherent motion. The benefit of this, is that one does not have to know how to read to be tested, and enables dyslexia screening early on in a persons life.

3.4 Detecting Dyslexia Today

There exist many different methods for detecting dyslexia already. The downside of these tests is that many of them rely on the subject already having some reading or writing capabilities. In Norway, dyslexia is normally diagnosed by a specialised doctor, who conducts these tests to ensure that struggles with reading and writing are not caused by some other factor than dyslexia. Parents in Norway are not allowed to enquire dyslexia testing in their children before the child has begun the 3rd grade of primary school [18].

The different tests used to screen for and diagnose dyslexia are often tailored to the subject's age, but one of the most used tests in Norway today, the word chain test (ordkjedetesten), can be performed on all subjects who have some knowledge in reading, regardless of age. The word chain test is a group exercise that is used to map a test subject's ability to decode words. The test consists of 90 words written in the following format: *boatfishcathome beerwinewhiskeyscotch*. The test subject is to place three lines within each word chain to divide them into four separate words. The test subject is given a score based upon how many word chains he or

she is able to divide within a time space of four minutes. The score will be equal to the number of correct answers, with a maximum score of 90 [12]. Test results derived from the application presented in section 3.5 have been compared to this word chain test.

3.5 App for Early Detection of Dyslexia

During the school year of 2015-2016, an application that is meant to function as a screening test for detecting dyslexia was developed [4]. The application incorporated the tests shown in figure 3.1. The summer and fall of 2016, H. Sigmundsson and K. S. Egset performed tests with the application on 100 different test subjects; the results were that there is a significant correlation between application test scores and test scores from the word chain test [19].

The goal for the application was to get the tests to function as correctly as possible. Hence, the user interface was secondary to the functionality. The user interface consists of simple white elements on a black background, with colouring to indicate certain functionalities. The screen captures below are taken with the default settings specified in the application. A full overview of the application can be found in appendix A.

The main menu, showed in figure 3.4, consists of buttons in the middle of the screen. You can choose between the three different tests, and settings. The exit button is coloured red to indicate its functionality. Looking at the figure of the main menu, you get a feeling of how the user interface is.

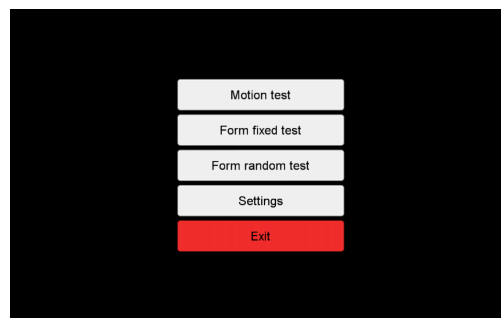


Figure 3.4: Main menu.

Figure 3.5(a) shows a screen capture of the motion test. Each patch contains 300 randomly placed dots with a radius of 1 pixel, and a minimum distance of 1 pixel between the dots. One of the patches are chosen at random at each interval to

contain a coherent motion target. In the chosen patch, a percentage of the dots will move either leftwards or rightwards, reversing every .572 seconds. The rest of the dots will move randomly, changing direction when colliding with other dots after .572 seconds.

The test taker should identify the patch with the coherent moving target during the 5 seconds of animation time. After the animation time, the dots will disappear and the test taker can click on the patch they believe contained the coherent motion target. The dots are recalculated with a change in coherency, and subsequently displayed on the screen when the input is registered. Figure 3.5(b) illustrates the dot animation at 50 percent coherency, where the blue dots are moving coherently to the right.

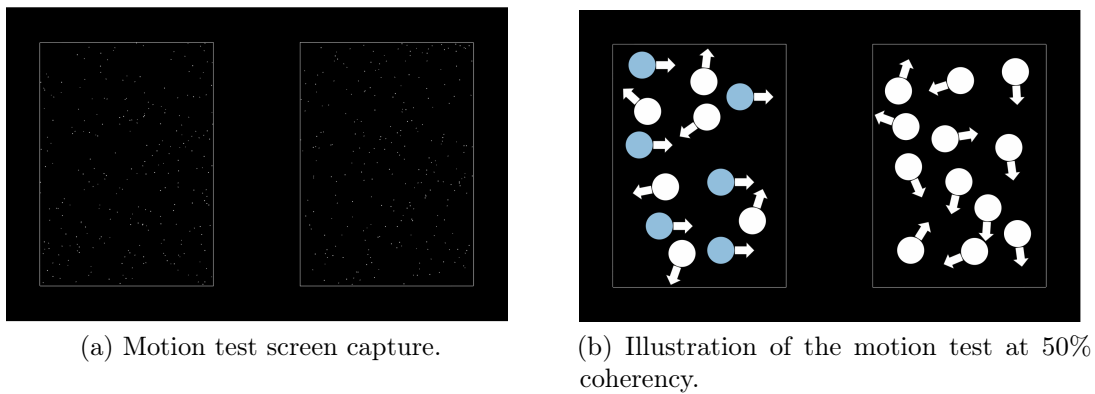


Figure 3.5: Motion test.

The form fixed auto test is shown in figure 3.6. Each patch consists of 600 line segments, and each of the lines have a length of 0.4° and a height of 1 pixel. One of the patches are chosen at random to contain a percentage of lines that form concentric circles at each interval. The centre of the concentric circles is locked in the centre of the chosen patch.

The test taker should search the patches for the concentric circles during an interval of 4 seconds. After the pattern disappears, the test taker can click on the patch they believe contained the circles. As in the motion test, the pattern is recalculated with a change in coherency and subsequently displayed on the screen when the input is registered. Figure 3.6(a) and (b) show the test at respectively 100 and 50 percent coherency.

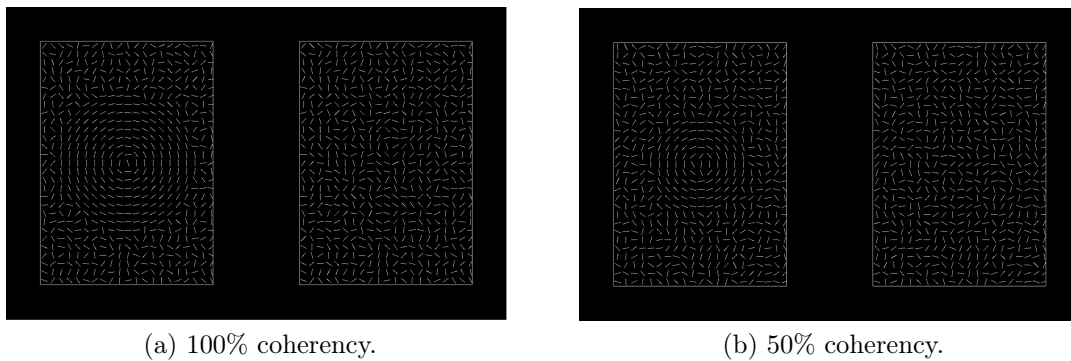


Figure 3.6: Form fixed auto test.

The form random auto test, shown in figure 3.7, is similar to the form fixed auto test. At each interval, one of the patches are chosen at random to contain a percentage of lines that form concentric circles. The difference is that the centre of the concentric circles will be placed randomly within the chosen patch, with its circumference confined within the patch. Due to the added difficulty of finding the target, the test taker is to search the patches for the concentric circles during the interval of 1000 seconds. The test taker can click the patches at any time.

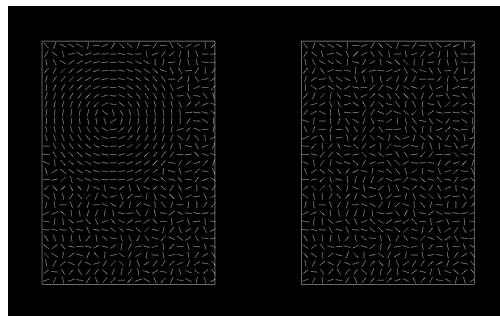


Figure 3.7: Form random auto test at 100% coherency.

The different parameters that are taken into account in the tests are listed and described in appendix A.

3.6 Dyslexia and Design

There are several aspects that can be taken into account when designing specifically for user groups with dyslexia. Common issues for people with dyslexia are text

distortion, blurred vision, pulsating vision, and discomfort while reading [20].



Figure 3.8: An example of the river effect (leftmost image), the blur effect (middle image) and the washout effect (rightmost image); commonly experienced by dyslexics [21].

To increase reading ability and comfort in dyslexic readers, there are several measures that can be taken into account. Coloured overlays, or changing the background colour of the reading surface might help in keeping the letters still on the page [20]. One should avoid using justified text alignment⁵ and double spacing after ending a paragraph, as this increases the risk of the subject experiencing the river effect seen in figure 3.8. Avoiding black text on top of a white background might help with the blur effect exemplified in figure 3.8. To prevent the washout effect seen in the same figure, the designer should avoid using serif fonts [21]. Studies show that the fonts best suited for readers with dyslexia are Courier, Helvetica, Arial, Verdana and CMU [23].

3.7 User Interface Design Process

The most common approach for designing user interfaces is the user-centred design process. This is an iterative process with focus on the user, the users needs, and how the user perceives the world. Each designer has a different approach to the user-centred design process, but there are some key principles that are applied by most interface designers [24];

⁵Paragraph or block of text in which all words in all lines are spaced-out such that the first word aligns with the left margin and last word with the right margin. Used commonly in books and magazines, it is not considered suitable for letters and other forms of personal communication [22].

- The work practises of the users control the development.
- Active user participation throughout the project, in analysis, design, development, and evaluation.
- Early prototyping.
- Continuous iteration of design solutions.
- Multidisciplinary design teams.
- Integrated design.

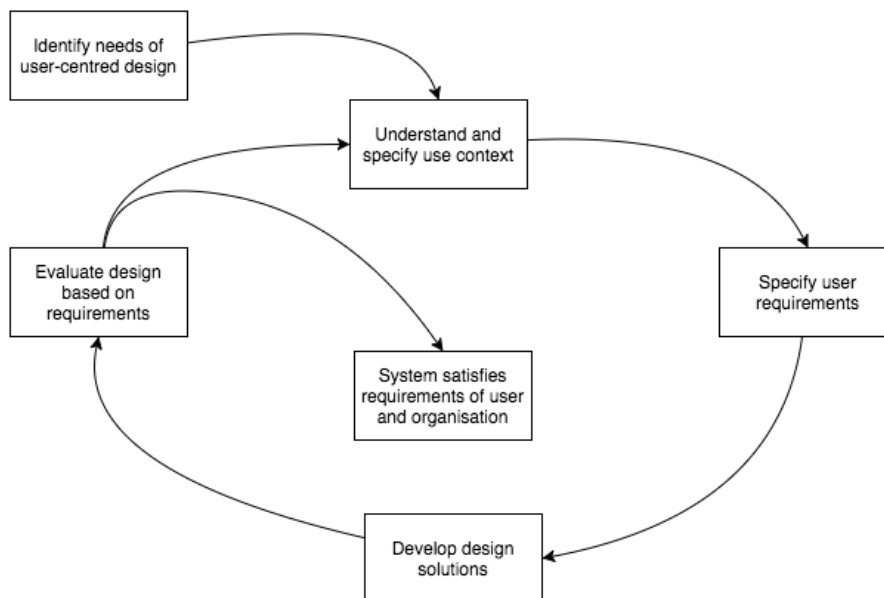


Figure 3.9: The iterative design process [25].

Normally, this process is divided into several stages, as seen in figure 3.9. By revisiting the requirements and design stages, the designer will be able to tailor the interface design the users' needs. It is important to keep in mind that the product should satisfy the requirements derived from the user as well as the organisation that owns it [25].

3.8 The Final Requirement List

This section contains the requirements derived during the specialisation project. The requirements were constructed together with Professor H. Sigmundsson based

on the information gathered during the interviews and observations, experiences from similar projects in design of applications, and from scenarios and use cases. The requirements have not changed from the specialisation project, though FR10 has been removed.

3.8.1 Functional Requirements

Table 3.1 specifies the functional requirements for the application.

Requirement #	Requirement	Priority
FR1	The application should use the screen size to make the design responsive to fit to different types of tablets and computers.	High
FR2	A tutorial must be implemented in the application to make it possible to use the application without any previous knowledge of the application.	High
FR3	The score view should contain an explanation of the threshold score.	High
FR4	The system should include descriptions that indicate that the user must click the boxes when taking motion, form fixed, or form random test.	High
FR5	It should be possible to cancel in the middle of a test without using the computers' keyboard or the back button on an Android tablet.	High
FR6	On desktop versions of the application, the cursor should change form when hovering over clickable content.	High
FR7	The "Settings" view in the application should sort different settings options under logical names.	Medium
FR8	It should be possible to start a test without completing a tutorial.	Medium
FR9	It should be possible to use the application without a supervisor.	Low
FR11	The application should be able to store test results, for use in comparisons based on age groups.	Low

Table 3.1: Functional requirements

FR10, "The score view should show how you performed compared to other people your age", was a part of the original requirements. The requirement was included as

it was a highly sought after functionality, and there was hope that data regarding test scores and age groups would be available before this project delivery. This data collection was not complete prior to project delivery, and FR10 was therefore removed.

3.8.2 Non-Functional Requirements

Table 3.2 specifies the non-functional requirements for the application.

As this research report focuses on enhancing usability in an existing application, our chosen quality attribute is *usability*.

Quality Attribute	Requirement #	Requirement
Usability	U1	It should be possible to reach any given system function from the main view within 3 clicks.
	U2	A user should be able to understand the application within 3 minutes.
	U3	The application should have a minimum SUS score of 80.

Table 3.2: Non-functional requirements

3.9 Presentation of the Paper Prototypes

This section presents the two different paper prototypes developed in the specialisation project, and the results from the usability tests conducted on the prototypes.

3.9.1 Views

The complete paper prototypes can be viewed in appendix E. The implemented design is supposed to be responsive where the layout adapts to the size of the screen; mainly tablets and desktop. The layout should look the same on a tablet and on desktop, and is therefore only shown on a mock up tablet.

Main Menu View

The main difference between the two prototypes is the main menu. The first paper prototype has a similar main menu as the existing application, with several buttons for the different options in the middle of the screen, while the second prototype has a static menu bar that is always present on the left hand side of the screen, except for during the actual application test. This is shown in figure 3.10. As can be seen in the figure, the main view in the second prototype has a lot of blank space. The meaning was for the application logo to be here, but as the application did not have a logo yet, this space was left blank.

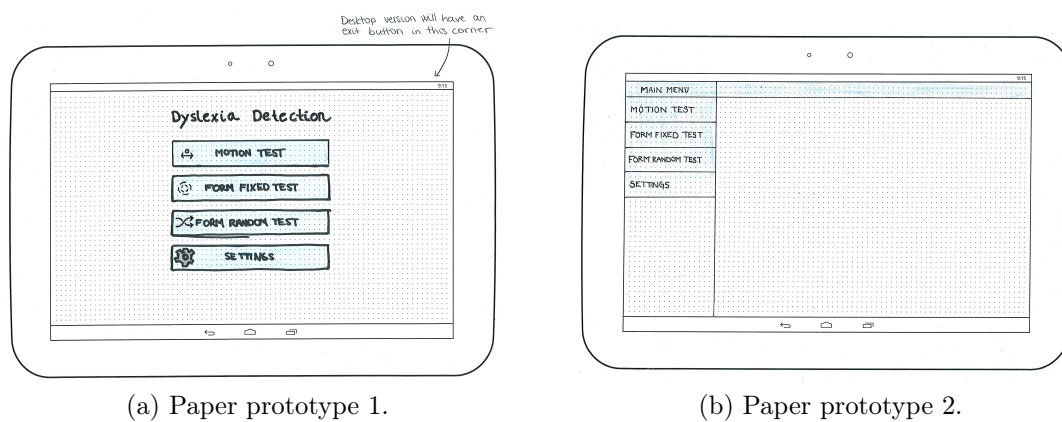


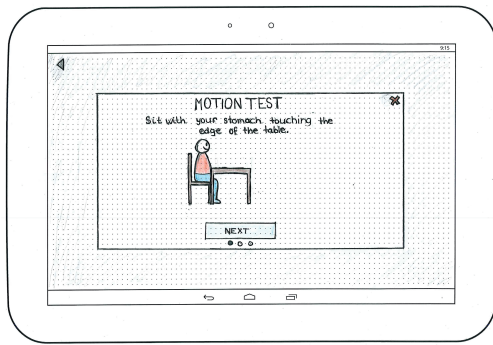
Figure 3.10: The main menu view.

Tutorial Views

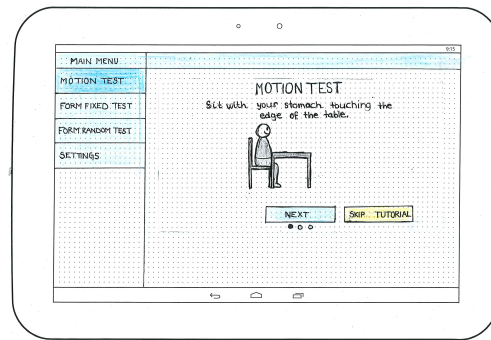
A tutorial was added in connection with every test, in order to explain what the user was to do during the tests. The tutorials for the three different tests within the application were very similar. The only difference was the text explaining what you should look for, and the content of the patches. The content of the tutorial was also identical across the two different paper prototypes. The figures in this section show the tutorial for the motion test. The tutorials for the form fixed test can be found in appendix E. Each tutorial consisted of three steps, with the same graphical layout on all three steps.

Figure 3.11 shows the first tutorial view in the two prototypes. The tutorial in the first prototype appears as a dialog window on top of the test view. You are able to see how many steps the tutorial consists of and you can close the window and start the test by clicking the “X” symbol at the top of the window. The tutorial in the

second prototype has the main menu available to the left, and the tutorial is not manifested as a dialog window. Other than that the two views are identical.



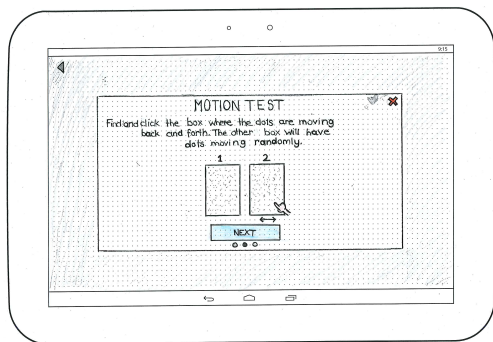
(a) Paper prototype 1.



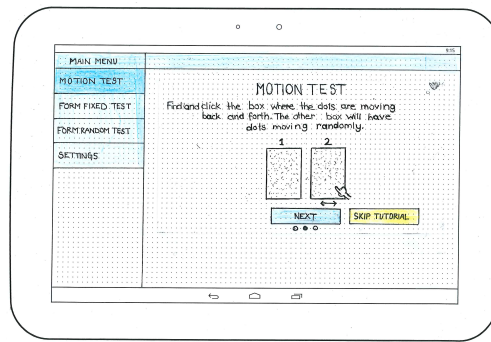
(b) Paper prototype 2.

Figure 3.11: The first tutorial view.

The next view of the tutorial is shown in figure 3.12. This view describes what the user should be looking for during the test. In the form fixed test, the dots inside the patches will be replaced with several concentric circles formed by many different lines.



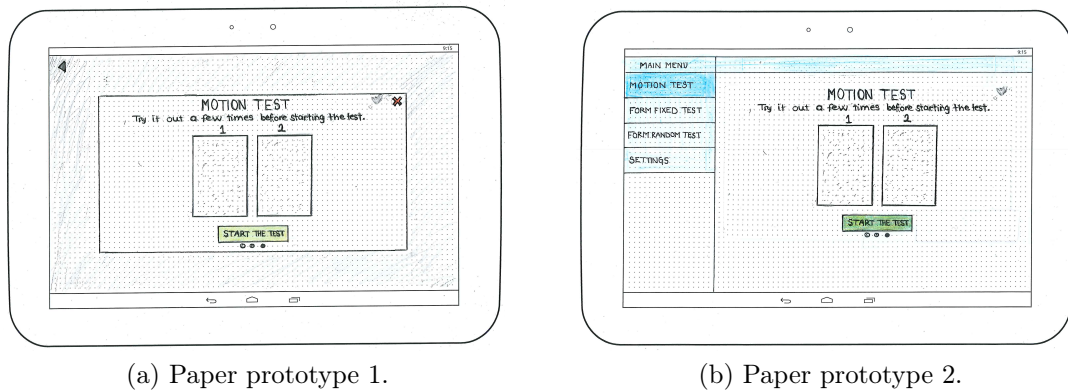
(a) Paper prototype 1.



(b) Paper prototype 2.

Figure 3.12: The second tutorial view.

The last tutorial view can be seen in figure 3.13. This view lets the user try the chosen test out a few times before starting the actual test. This makes it easier for the users to understand the task.



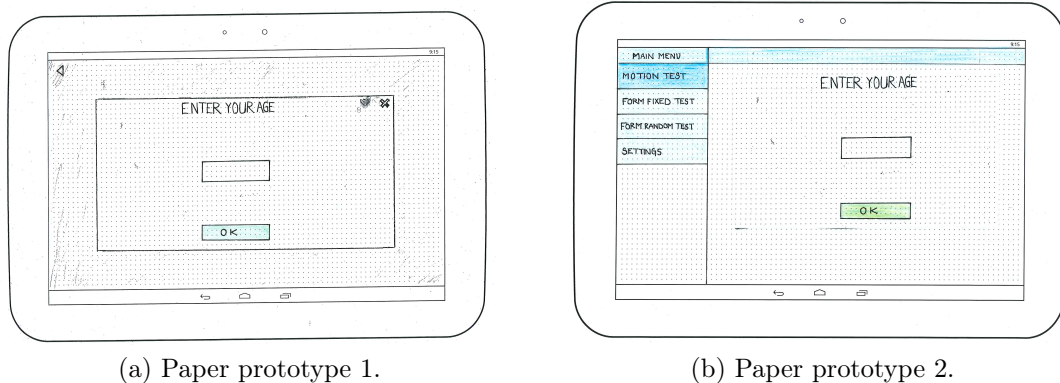
(a) Paper prototype 1.

(b) Paper prototype 2.

Figure 3.13: The third tutorial view.

Enter Age View

The enter age views have identical content across the tutorials for the different tests in the application. The first prototype still has the dialog window with the opportunity to close it, and the second prototype has the main menu on left hand side, where the chosen test is highlighted. In this view the user should enter their age and start the test. The views are shown in figure 3.14.



(a) Paper prototype 1.

(b) Paper prototype 2.

Figure 3.14: The enter age view.

Test View

The test views are the same in both prototypes and in the different tests. The only difference in the tests is the content of the patches. Figure 3.15 shows the test view.

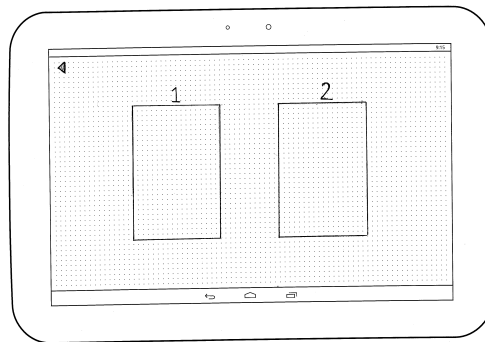
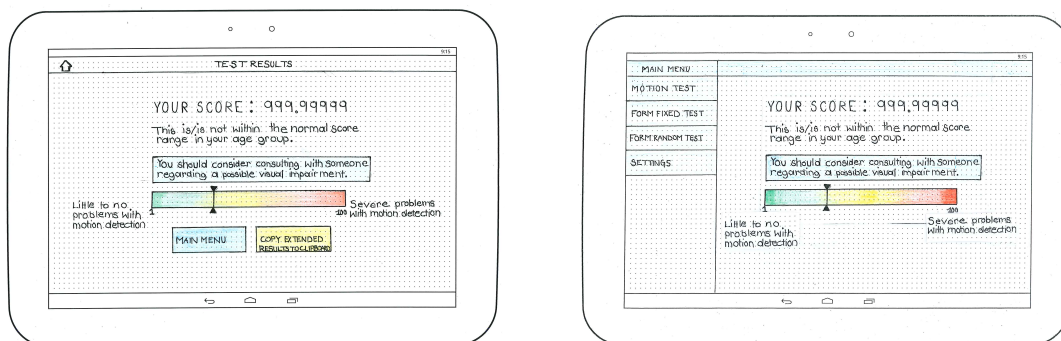


Figure 3.15: The test view (without dots and lines).

Score View

The score view, as shown in figure 3.16, shows the result of the test and whether or not the test score is within the normal score in your age group. The main difference here is the main menu available to the left hand side of the second prototype.



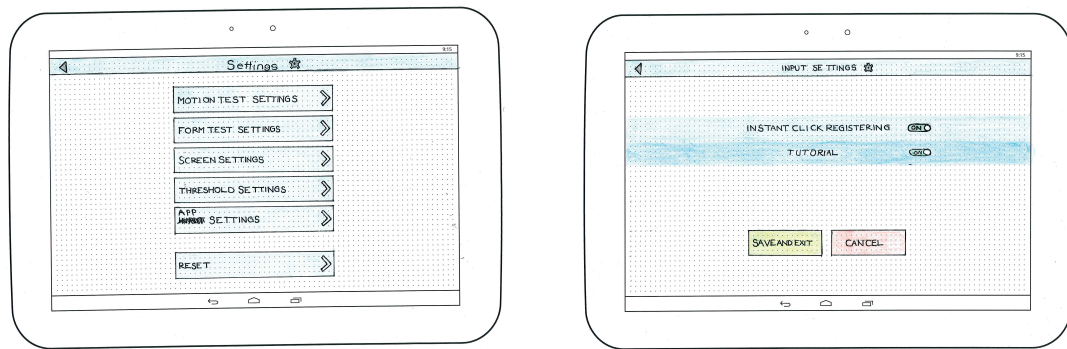
(a) Paper prototype 1.

(b) Paper prototype 2.

Figure 3.16: The score view.

Settings View

Figure 3.17 shows the settings view for the first paper prototype. The first settings view, seen in figure 3.17(a), consists of several buttons for the different settings types. The second view, seen in figure 3.17(b), shows the app settings view. The check box “*Continuous mode*” in the original application, has been changed to an on/off switch with the text “*Instant click registration*”. The user has to navigate backward in the navigational structure of the application to edit other parameters than the ones chosen.



(a) Settings view.

(b) App settings view.

Figure 3.17: The settings view for paper prototype 1.

Figure 3.18 shows the settings view for the second paper prototype. The main menu is still available on the left hand side, and the settings menu is placed next to the main menu. As can be seen, a standard convention used by many android applications and devices is applied in this interface design. The settings menu is hierarchically organised, with a left-to-right flow of information, where the most general information is placed on the left hand side, and the most specific information is displayed on the right hand side. The user is able to navigate freely between the different setting categories.

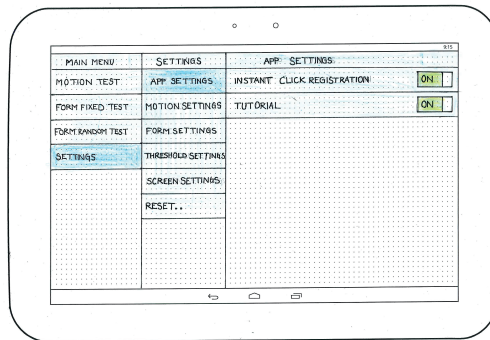


Figure 3.18: The settings view for paper prototype 2.

3.9.2 Test Results and Further Work

This section presents the results from the usability testing of the paper prototypes developed in the specialisation project. The results from the usability testing of the paper prototypes were in general very positive. The results were extracted from the questionnaires and SUS forms the test subjects were asked to fill out, as well as oral conversation during the tests.

Results Paper Prototype 1

It was clear that the first prototype was easy to use, and that the test subjects did not show any great difficulty in performing the tasks given. The test subjects demonstrated to navigate the prototype with ease, even without receiving any help from a supervisor. There were however some areas where the test subjects showed some stalled decision making when completing tasks. The problem areas demonstrated by the test subjects were as follows:

- Fully understanding the tutorial.
 - Understanding that you are to choose a box several times.
 - Understanding the test score.
 - Understanding what to look for during tests.
 - Exiting the tutorial.
- Understanding wording under “Settings”.
- Understanding the purpose of the application.

The SUS score of the paper prototype gives the designer a brief overview of some problem areas, as well as letting the designer know overall how satisfied the test subject was with the user interface. The SUS score can be seen in figure 3.19.



Figure 3.19: SUS score for paper prototype 1.

A score of 84 is very satisfactory, and is considered above average. The application scores low on intention to reuse, but as the application is not meant to be used several times by the same person, this parameter is ignored. The system also scores lower in confidence in using the system, ease of use, complexity, need for technical support and integration. 4.33 is not a low score, quite on the contrary, but these are the lowest scoring areas and are something to look at when further developing the design. On the other hand, the need for previous knowledge scores 5 in the SUS score, which is a perfect score. Overall, the test subjects were satisfied with the user interface design, and did not have major problems in navigating or completing tasks.

Results Paper Prototype 2

As the first paper prototype, the second paper prototype did not pose any critical issues either. The tutorial seemed to have the same problems as the first prototype, except for skipping the tutorial. This was not surprising as they are almost the same. There were however some differences. The identified problem areas were:

- Fully understanding the tutorial.
- Understanding the purpose of the application.
- Understanding the blank space in the main view.

The SUS score of the second paper prototype can be found in figure 3.20.



Figure 3.20: SUS score for paper prototype 2.

A SUS score of 91 is very satisfactory, and far above average. As can be seen, the second paper prototype scores a perfect score in several areas; complexity, need for technical support, consistency, and need for previous knowledge. The lowest scorers were integration and quickly learning the system. Again, here as in the first prototype, frequent use is ignored as this application is not meant to be used more than once.

Further Work

A new prototype needs to be developed and tested. Although one of the prototypes has a better score than the other, it is not easy to say which direction the prototype design should be taken. This is because of the small amount of test subjects used in the usability tests. The prototypes are not tested on enough people to be able to distinguish which of the prototypes that is significantly better than the other, as the SUS scores are relatively close in value. Because of this, and in light of the feedback we have received, a digital prototype that merges the best parts of the two prototypes, and improves on the identified problem areas might be the best solution. After testing and improving the digital prototype, the implementation of the user interface needs to be carried out.

Chapter 4

Digital Prototype

This chapter will encompass the development and testing of a digital prototype. In section 4.1, the first iteration of the digital prototype design development is presented. The final digital prototype design is presented in section 4.2.

4.1 Initial Digital Prototype

In our specialisation project, two different paper prototypes were developed, tested, and evaluated, as seen in section 3.9. Based on test results received from testing of these paper prototypes, we have developed a digital prototype. This section will include a presentation of the most important parts of the development of the initial digital prototype.

4.1.1 Changes From Paper Prototype

Based on the results presented in section 3.9 and discussion with our project supervisor, it was decided to develop a digital prototype with a mixture of elements from the two different paper prototypes. In order to keep easy error correction and navigation, a version including the menu on the left hand side was chosen. Inside the tutorial, the “Skip tutorial” button was kept from the second paper prototype. As the prototype was digital, the possibility of including moving images was a reality, enabling further enhancement of the user experience during testing. The trial function was enhanced in the tutorial to let the tester check if they had understood the task. We also added feedback in the trial to let the user know whether or not they had chosen the correct box, letting the user know if he or

she has understood the test functionality. The fourth step of the tutorial, where the user is to enter his or her age, was kept, even though requirement FR10 was removed. FR10 can be found in section 3.8. This was to make it easier to extract age data from the application, and analyse data based on age. The design concepts applied in the paper prototype have been conserved in the digital prototype, but the digital prototype has a more professional look and feel to it, as it has been digitally developed. In addition, the settings view was taken in a direction where the different settings options were sorted within an additional menu bar at the top of the screen. All relevant setting parameters were placed under an “App settings” tab, all non-relevant setting parameters were gathered under one “Advanced settings” tab, and there was a separate tab for “Reset settings”.

The digital prototype incorporated moving images, or GIFs (Graphics Interchange Format), in the tutorial and test screens where test-simulation was present. This, together with the possibility to incorporate gestures such as swipe navigation and realistic scroll, made the usability test environment closer to a realistic user environment. This also enhanced the user experience, and increased the validity of the tests.

4.1.2 Usability Testing, Results and Changes

This section will include a thorough walk through of the testing of the digital prototype, present the results of the tests, and a presentation of what changes were made before implementation.

Usability Testing

The usability tests were carried out in a booked room at NTNU, Gløshaugen. This was to ensure that there were no disturbances during the test. Ten test subjects tested the digital prototype, and the subjects were students at NTNU between the ages of 18 and 25 years old. All tests were carried out as if the test subjects were to use the application without a supervisor. The two test conductors were placed on each side of a table, where the one taking notes was situated on the left hand side of the test subject, and the one describing the test and informing about the tasks was situated directly across from the test subject.

When the test subjects arrived, they were asked to take a seat in the assigned chair. They were then given a form to sign, allowing them to be filmed and to use the test results in this report. The form can be found in appendix B. When the test subjects signalled that they were ready, one of the test conductors read a

prewritten description of the test.

Prewritten test description

“This is a usability test of a digital prototype of a system design. It would be nice if you at all times could say out loud what you are thinking. We are only here to observe and take notes of what you are thinking. We will not help you during the test if there is something you do not understand. If you do not understand something, you have to try your way, until you figure it out. Remember that you are not doing anything wrong if you cannot complete a task, this means the prototype design is not good enough.

You are going to test the user interface of an application that tests if you could have dyslexia or not.”

After the test description was read, the test subjects were presented with a tablet where the digital prototype application was opened. The test subjects were then given tasks to perform. Each task was read after the previous task was completed.

Usability test tasks

1. Complete a motion test.
2. Complete a form fixed test.
3. Change the setting that decides when a click is registered during the tests. A click should not be registered right away.
4. Turn off the tutorial at the beginning of each test.
5. Exit the application.

After the usability test, the test subjects were given a questionnaire and a SUS form to fill out. The SUS form and the questionnaire can be found in appendix C and D. Both the SUS form and the questionnaire were the same as the ones used during our specialisation project, to ensure comparability.

Results from Usability Testing

The results from the usability testing of the digital prototype were in general very positive, but there were a few aspects that had potential for improvement. The results were extracted from the questionnaire and SUS form the test subjects were asked to fill out, as well as oral conversation during the tests.

The main feedback given on the prototype was that it was easy to use and that it was aesthetically pleasing. As expected, the test subjects did not show any great difficulty in performing the tasks given, but there still existed some areas of improvement. The problem areas found were as follows:

- Not possible to go backwards in the tutorial.
- Confusion on whether or not you are inside the test or inside the tutorial.
- Problems understanding that you could try it out a few times before you could start the test.
- Too much white space in the design.
- No button for closing the application in desktop version.
- Problems understanding the sentence about “Instant click registration” in the tutorial description.

The SUS score of the digital prototype gives the designer a brief overview of some problem areas, as well as letting the designer know overall how satisfied the test subjects were with the user interface. The accumulated SUS score can be seen in figure 4.1.

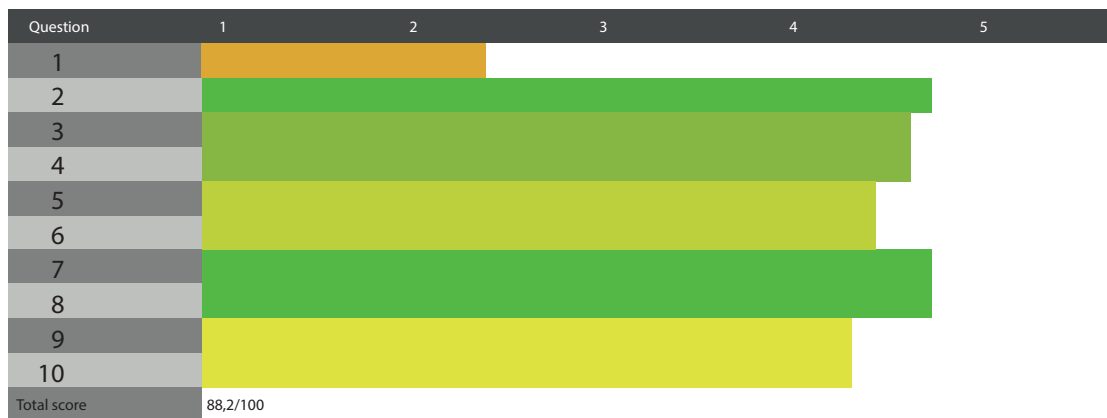


Figure 4.1: SUS score for digital prototype.

Question 1, “*I think I would like to use this system frequently*”, can be eliminated, as the question is not relevant for the application. The application is not intended to be used by one person more than once. If we calculate the new SUS score, by eliminating question 1 and calculating the new percentage, we receive a SUS score of **92.7**.

4.1.3 Changes to the Digital Prototype

Based on the feedback received during the usability testing, there were several changes made to the digital prototype.

- The content of the tutorial was scaled up to eliminate as much white space as possible.
- A “Back” button was added to the tutorial in place of the “Skip tutorial” button.
- A “Skip” button was added next to the tutorial header.
- The tutorial header was changed from “Motion test” to “Motion test tutorial”. Respective changes were made to all test tutorials.
- The tutorial screens for demonstration and trial were changed to emphasise whether or not it was possible to try the test or not.
- Information about the duration of the test was added in the tutorial description.
- Information about not receiving feedback during the actual test was added to the tutorial description.

The rest of the phrasing in the different tutorials was not changed, as this is easy to change during or after implementation rather than in the digital prototype.

4.2 Final Digital Prototype

This section presents the views of the digital prototype where the changes mentioned in section 4.1.3 have been incorporated. The entire finished digital prototype can be seen in appendix F.

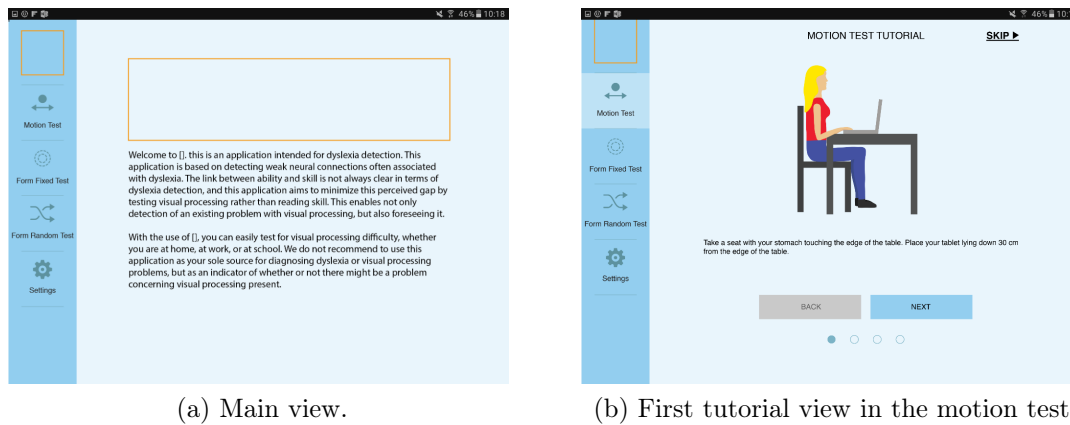
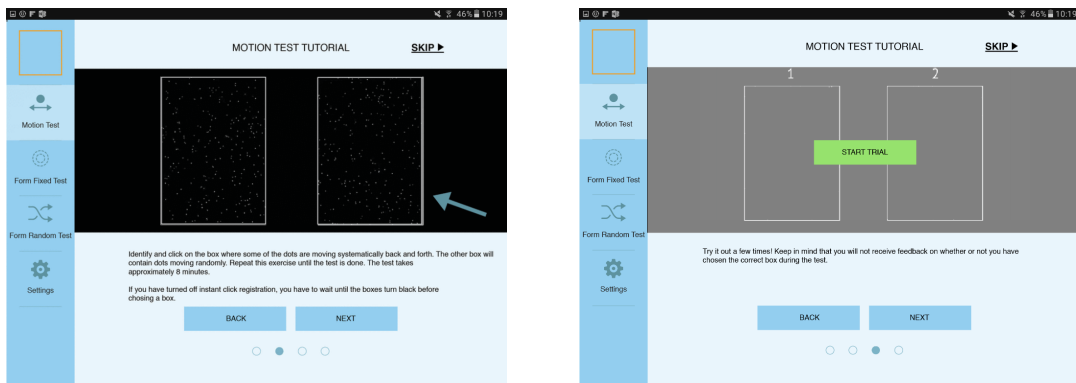


Figure 4.2: The main view of the digital prototype and the first tutorial view.

The main view of the final digital prototype for tablet is the same as the initial digital prototype. The main menu on the left hand side of the screen contains a placeholder where a logo can be placed, and the different navigational choices. When you click on one of the navigational choices, for example “Motion test”, as seen in figure 4.2(b), the chosen selection is highlighted. Changes from the first digital prototype are that the header is changed, and the “Back” button is added.



(a) Demonstration tutorial view.

(b) Trial tutorial view.

Figure 4.3: The second and third tutorial screen of the motion test.

In figure 4.3 you can see the difference between the demonstration screen and the trial screen in the tutorial. The white space in the tutorial has also been eliminated.

Chapter 5

Implementation

In this chapter we present some of the choices made for making a user interface of adequate quality, alongside explanations of solutions that were used to achieve the final implementation. Section 5.1 gives an overview over the application appearance and code structure. In section 5.2 we justify our design choices, and in section 5.3 the graphic profile is presented. Section 5.4 contains a description of the application architecture.

5.1 Application Overview

When implementing the user interface of this application, the aim was to make the user interface as close to the finished digital prototype design as possible, while making necessary changes where they were needed. All screen shots in this section are captured on a Samsung Galaxy Tab A [26], but as the design is responsive it will look the same on any android device as well as on desktop.

The implementation can be divided into two parts, further development of the design and appearance of the application, and converting this design into code.

5.1.1 Application Appearance

In this section the finished application is presented. A full overview of the application can be found in appendix G. The reasoning behind the different design choices can be found in section 5.2.

Home Screen

The home screen seen in figure 5.1 consists of a menu bar on the far left hand side of the screen, containing all the different navigational choices, as well as a description of the application. In addition to be able to select different tests and settings in the menu bar, the user can click on the icon in the top left corner to navigate back to the home screen. The icon at the bottom of the menu bar exits the application. The description was drafted in collaboration with psychology Professor H. Sigmundsson and project supervisor, Professor J. Krogstie.

Application description

“Welcome to Magno. This is an application intended for dyslexia detection. This application is based on detecting weak neural connections often associated with dyslexia. The link between ability and skill is not always clear in terms of dyslexia detection, and this application aims to minimise this perceived gap by testing visual processing rather than reading skill.”

With the use of Magno, you can easily test for visual processing difficulty on your own. We do not recommend to use this application as your sole source for diagnosing dyslexia or visual processing problems, but as an indicator of whether or not you might have a problem with visual processing.”



Figure 5.1: Home screen of the application.

Tutorial - Step 1

If you click on either of the test buttons in the side menu bar, you will by default enter the respective test tutorial. The first screen of the “Motion test tutorial” and the “Form fixed test tutorial” can be seen in figure 5.2. The menu bar is highlighted based on what the user has clicked, and the description in the first step of all the different test tutorials is the same. The “Back” button in the first step of the different tutorials is disabled. When inside the tutorial, swipe functionality is enabled, allowing the user to swipe between the different steps in the tutorial. The user can also click on the circles at the bottom of the different tutorials to directly navigate between specific steps. To skip the tutorial, the user can click on the “Skip” button at the top right corner of the screen. The tutorial can be turned off in the application settings.

Tutorial description - first step

“Take a seat with your stomach touching the edge of the table. Place your device 30 cm from the edge of the table.”

The viewing distance in the descriptive text changes based on what viewing distance is set in the application settings. The default value for the viewing distance is 30 cm.



(a) The first view in the motion tutorial.

(b) The first view in the form fixed tutorial.

Figure 5.2: The first step of the tutorial.

Tutorial - Step 2

The second step of the motion test tutorial and the form fixed test tutorial can be seen in figure 5.3. As one can see, the menu bar is still highlighted based on what choice the user has made. This step describes what the user is to look for during

the different tests, and how to complete the test. In this tutorial step, the “Back” button is no longer disabled. The other navigational choices for navigating within the tutorial are still functional.

In the second step of the motion test tutorial, the screen shows two patches filled with moving dots. The moving dots are moving in accordance with the parameter settings for the motion test. The rightmost patch will always contain a percentage of dots moving in a coherent motion. This patch is indicated with a blue arrow. Beneath the two patches, the user will find the test description.

Motion test tutorial description - step two

“During the test, you should identify and click on the box where some of the dots are moving systematically back and forth. The other box will contain dots moving randomly. Repeat this exercise until the test is done. The test takes approximately 8 minutes.”

Additional description - “Instant click registration” disabled

“You have to wait until the boxes turn black before choosing a box.”

In the descriptions, the word “box” is used instead of “patch” to make the wording more user friendly to users of a younger age.

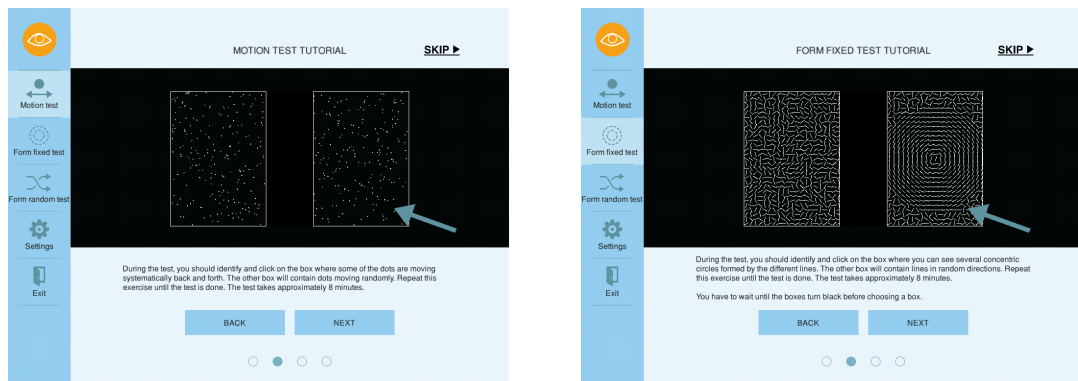
The setup of the second step of the form fixed and form random test tutorials is the same as in the motion test tutorial, except from the content of the different patches and the test description. Within the patches in the form fixed and form random tutorials, the user will see several small lines. In the rightmost patch, some of the lines will form several concentric circles. In the other patch the direction of the lines is completely random. In the form fixed test tutorial, the centre of the concentric circles will always appear in the middle of the patch. In the form random test tutorial, the centre of the concentric circles can vary in placement within the patch. The test description in the second step of both the form fixed and the form random test tutorial is the same.

Tutorial description for form fixed and form random test - step two

“During the test, you should identify and click on the box where you can see several concentric circles formed by the different lines. The other box will contain lines in random directions. Repeat this exercise until the test is done. The test takes approximately 8 minutes.”

Additional description - “Instant click registration” disabled

“You have to wait until the boxes turn black before choosing a box.”



(a) The second view in the motion test tutorial.

(b) The second view in the form fixed test tutorial.

Figure 5.3: The second step of the tutorial.

Tutorial - Step 3

Figures 5.4 and 5.5 show the third step of the tutorial. This step is identical for all three tests, except for the content of the patches. On the screen, there is a dedicated area for the trial, with a button in the middle for when the user wishes to begin the trial. This can be seen in figure 5.4(a). When the trial starts, the patches are populated based on the type of tutorial the user has chosen. When a patch is selected and clicked upon by the user during the trial, a text appears on top of the patches indicating whether the selected patch was the correct patch or not. The trial gives the user the opportunity to try and identify the correct patch three times, after which the trial will end.

Text for correct selection: *“Good job! You chose correctly”*

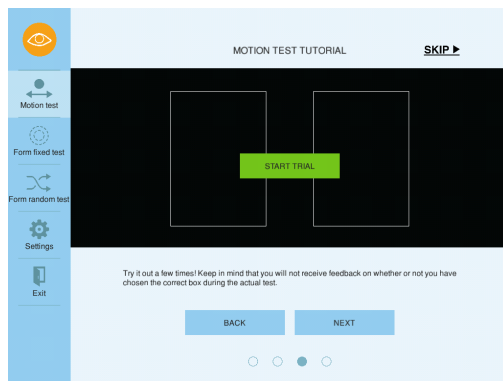
Text for wrong selection: *“Better luck next time”*

Text for ended trial: *“Continue to the next step”*

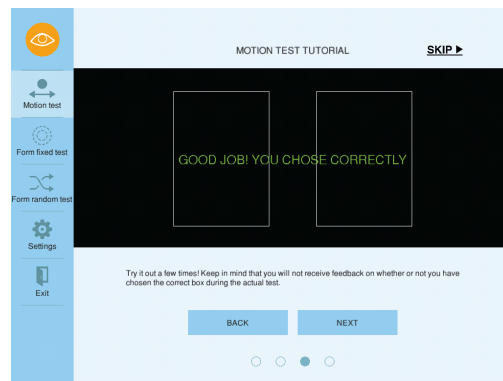
At the bottom of the screen, there is a description of the trial. This description is identical for all three test tutorials.

Tutorial description - step three

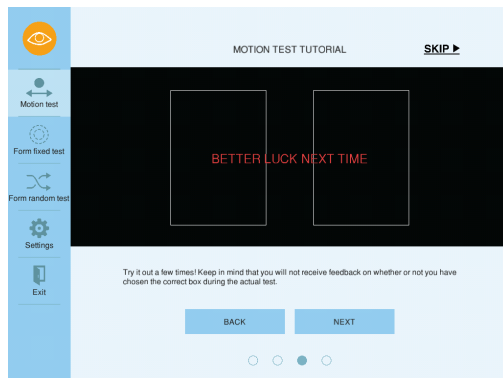
“Try it out a few times! Keep in mind that you will not receive feedback on whether or not you have chosen the correct box during the actual test.”



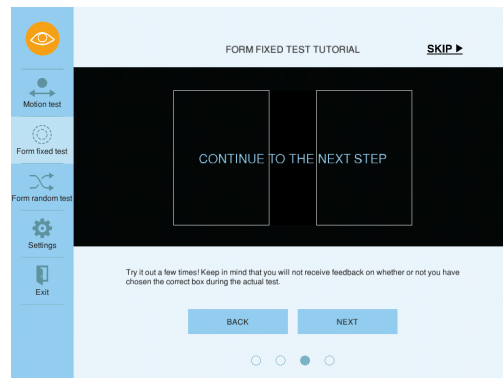
(a) Prior to beginning trial.



(b) Trial begun - correct patch chosen.

Figure 5.4: The third step of the tutorial.

(a) Trial begun - wrong patch chosen.



(b) Completed trial.

Figure 5.5: The third step of the tutorial.

Tutorial - Step 4, and Test Results

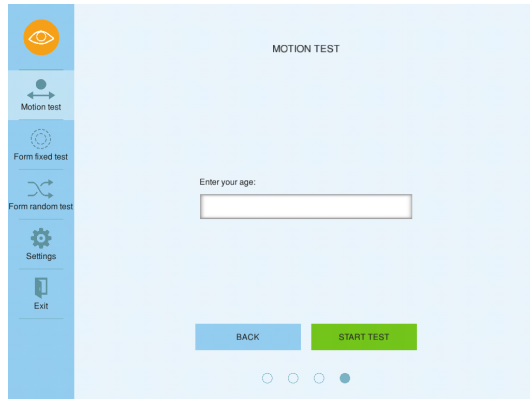


Figure 5.6: The fourth step of the tutorial.

The fourth tutorial step view is approximately the same, regardless of what test the user has chosen. The differences are the header and which section in the menu bar that is highlighted. This view can be seen in figure 5.6.

In this view you will see the header, indicating what kind of test you have chosen, and a text field. Above the text field there is a descriptive text, “*Enter your age:*”, indicating what you should write in the text field. At the bottom, the “Next” button is substituted by a “Start test” button. The “Skip” button in the top right corner has also been

removed. The user can still navigate between the different tutorial screens. If the user chooses to skip the tutorial, or turns off the tutorial in the application settings, the fourth tutorial step is what will be shown.

Test View

Figure 5.7 shows the test view for motion test and form fixed test. There are no groundbreaking changes in this view. An arrow in the top left corner makes it possible to exit the test without using the keyboard or the back button on the tablet. The numbers “1” and “2” are placed above the patches to make it easier to communicate the patch you want to choose if you are not clicking the patches your self. These changes are also made in the form random test. Other than that, all of the test views are the same as before. The tests are described in section 3.5.

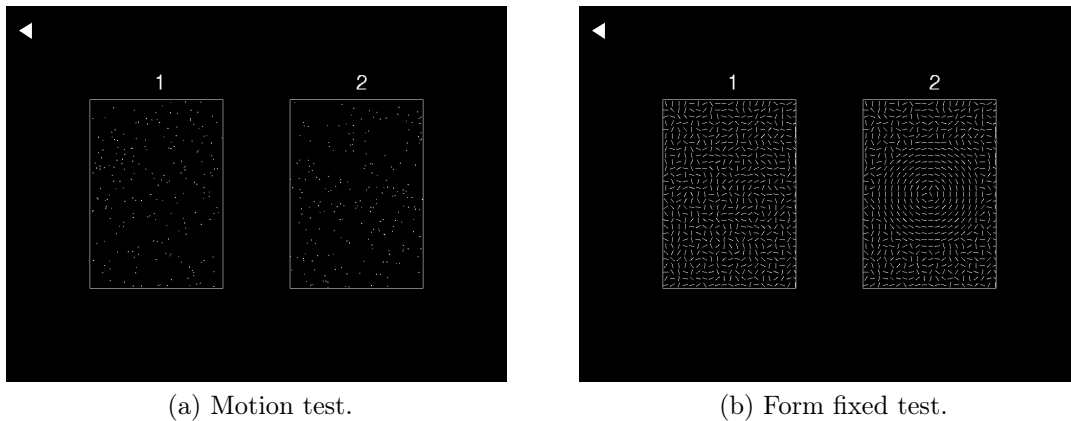


Figure 5.7: The test view.

Test Results

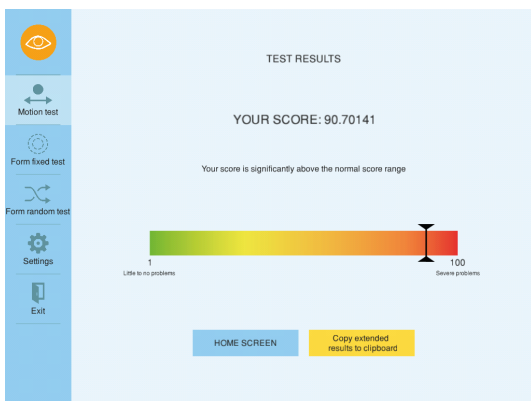


Figure 5.8: Motion test result view.

The content of test result view is the same, regardless of what test the user has chosen. The difference in appearance is which section in the menu bar that is highlighted. This is shown in figure 5.8.

The test result view appears when the user is finished with the chosen test. At the top of the screen, there is a header, telling the user that he or she is viewing the test results. Following, the user sees his or her score. The score is a value between 1 and 100, representing the final threshold value derived from the completed test. Beneath the score

value, there is a text telling the user whether or not the score is within the normal score range. As there is no available data on what score range is normal for different age groups, this text is static regardless of what age the user enters.

Score between 1 and 20 - description

“Your score is within the normal score range.”

Score between 20 and 50 - description

“Your score is slightly above the normal score range.”

Score between 50 and 100 - description

“Your score is significantly above the normal score range.”

In the middle of the screen, there is a progress bar showing a graphical representation of the test score. The progress bar has the number “1” and “100” indicated at each end, and a descriptive text at each extremity. The far left hand side of the progress bar represents low score values, and has the text *“Little to no problems”*. The rightmost side of the progress bar has the descriptive text *“Severe problems”*.

There are two buttons at the bottom of the result view. One button will take the user back to the main view of the application. The other button will copy the results along with the test parameters onto the device’s clipboard, enabling the user to paste the results into a spreadsheet or a text processing program.

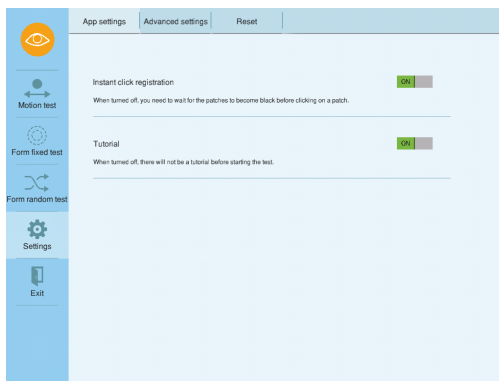
Settings

When entering the settings tab in the menu bar, the selection is highlighted in the menu bar. The layout of the settings views is the menu bar at the left hand side of the screen, as well as a new menu bar at the top of the screen, for selecting different kinds of settings. This can be seen in figure 5.9. The content of the view appears in the void between the two menu bars.

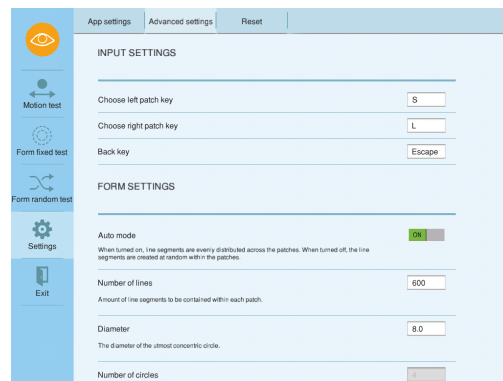
The default view within the application settings is the “App settings”, seen in figure 5.9(a). This selection is highlighted in the top menu bar. The contents of the “App settings” page are two parameters the user can either turn on or off. These two parameters are presented in a list, where the parameter name and description is on the left hand side of the list, and a toggle button is on the right hand side of the list. The two parameters are *“Instant click registration”* and *“Tutorial”*.

The “Advanced settings” view, has a similar layout to the layout of the “App settings” view. This can be viewed in figure 5.9(b). The different parameters that can be edited in “Advanced settings” change different aspects of the tests within the application. These parameters are also presented as a list, that can be scrolled to view more parameters. The parameters are sorted into several different sections based on what aspect of the tests they affect. Each section of the list is introduced by a descriptive header. The parameter name and description is presented on the left hand side of the list, and either a text field or a toggle button is present on the right hand side of the list. The sorting and explanation of the setting parameters can be seen in appendix A. The advanced settings sections are as follows:

- Input settings.
- Form settings.
- Motion settings.
- Threshold settings.
- Screen settings.



(a) App settings.



(b) Advanced settings.

Figure 5.9: Application settings.

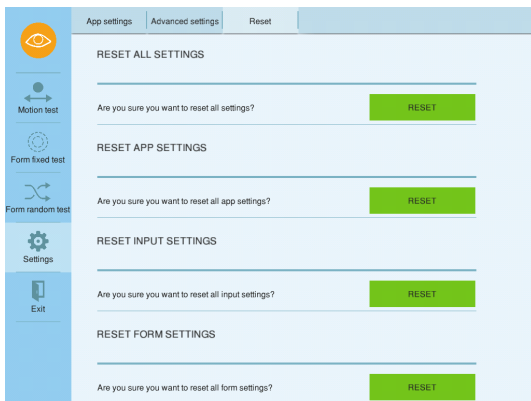


Figure 5.10: Application settings; reset settings.

The last tab in the top menu bar is “Reset”. A screen shot from this view can be seen in figure 5.10. As in the other settings views, the different parameters in the “Reset” view are presented as a list, and they are separated into the same different sections as “Advanced settings”, but also include “App settings” and “All settings”. Within the list, the different sections are described on the left hand side, and on the right hand side there is a button allowing the user to reset the settings within that section back to the default value. The text on the buttons change if the chosen setting section has been reset, from “Reset”, to “[section name] are reset”. The content of the page has enabled scrolling.

5.1.2 Application Code

A large part of this master's thesis has been implementing the chosen design in Java. The architecture and reasoning behind the application code can be seen in section 5.4. The main goal of the implementation has been to make the application design resemble the design of the digital prototype as much as possible, while at the same time creating code that is not cluttered or bewildering. In addition, an important aspect of the implementation was to not make any changes that might lower the credibility of the tests within the application.

5.2 Design Choices

In this section all the design choices that have been made within the application will be described, as well as why they were made.

5.2.1 Structural Choices

It is important for usability that the screen is organised and presented in an understandable manner to the user. This makes the application easier to navigate, as a properly organised screen promotes important elements and enables the user to ignore secondary information when necessary [1].

Side Menu Bar

The entire visual structure of the implemented user interface design is centred around the side menu bar. This menu bar is consistent within all the different views within the application, aside from inside the actual test environment. Consistency is an important aspect of interface design, and leads to an increase in user satisfaction, and reduction in task completion times and learning time. Many of the different views in the application are very similar, especially within different categories such as the different settings views and the different tutorial views across the different tests. The menu bar functions as an anchor point for the user, and enables easy error correcting, by providing the user with the possibility to navigate within the application regardless of the current navigational placement inside the application.

Placement of the different elements on the screen should follow the customary reading direction, sorted either by a *conventional standard*, *sequence of use*, *frequency*

of use, function or category, importance, or from most general to most specific. The menu bar is the first thing the user should use, and by placing the menu bar to the far left of the screen, it is the first thing they see due to the reading direction. The menu bar is also the element of the design that is meant to be most frequently used, and is the most important and general element of the design. In order to maintain a top-to-bottom, left-to-right flow through the screen, the menu bar at the left side of the screen is chosen to be vertical. This means the entirety of the menu bar will be viewed before the user progresses to the rest of the screen. The order of the elements within the menu bar is decided based on frequency of use, where the logo at the top of the menu bar brings the user back to the main screen, followed by the buttons leading to the different tests. Following the different test buttons, the button leading to the settings views is placed. At the bottom of the menu bar there is a button to exit the application [1].

Home Screen

The void next to the side menu bar is the part of the screen that is interchangeable between the different views. The home screen includes a logo and a brief description of the application in this space, to properly introduce the purpose of the application to the user. The logo is placed above the description to visually introduce the application before making the user read a lot of text [1].

Tutorial Views

The different steps in the different tutorials all have a similar setup to promote consistency, both across the different test tutorials, and within the different steps of the tutorials. All the tutorial steps incorporate the top-to-bottom, left-to-right flow, and have a header at the top of the screen. The user reads this header first, and understands that he or she is inside a tutorial, along with what test the tutorial is connected to. By highlighting the chosen test category in the side menu bar, the user knows the navigational placement within the application at all times. On the right side of the header, there is a skip button. This button is not coloured, as bright colours tend to draw attention to them selves. Yet the button is placed at the top of the screen, in order for the user to be informed early on that it is possible to skip the tutorial, though this is not recommended. The next element in the different tutorial views is an image of what you need to know to complete the step in the tutorial, followed by a descriptive text below. The image is presented above the descriptive text as the two are supposed to be understood together. Images and colours are usually perceived before text, and in this case the text

usually does not make sense if the user has not seen the image. At the bottom of the page of the tutorial views, there are two buttons and four circles. The two buttons have functionality to progress to the next step in the tutorial (or start the test), or to navigate backwards to earlier steps. The button that navigates backwards is placed to the left, and the button that navigates forward is placed to the right. Following western reading standards, progression is a motion towards the right, while a motion towards the left symbolises regression. The two buttons are placed below the descriptive elements of the tutorial because the user should process the information presented in the tutorial before moving forward to the next step. At the bottom of the screen, the four circles symbolise the user's placement within the tutorial, and the circle representing the current step of the tutorial is fully coloured. The rest of the circles are hollow. The circles are placed at the bottom of the screen because they convey the least important information on the screen [1].

Test Views

The test views have not undergone any significant changes, in order to preserve the integrity of the test and test results. The test screen, regardless of what test the user chooses, consists of two patches with varying content. On top of each patch there is the number "1" or the number "2". The numbers have been added because conversations with a test supervisor revealed that if the test subject is not to click on either of the patches themselves, it is easier to indicate which patch to choose if there is some sort of identifying element present. In addition, an arrow pointing to the left is added at the top left corner of the tests. This arrow exits the test, and is placed in the top left corner to signify regression [1].

Settings Views

All the different settings views are structured in the same way to promote consistency. At the top of the screen, a new horizontal menu bar is visible. The top menu bar has three different tabs, sorted by frequency of use. The different setting parameters are divided into two main groups; parameters relevant to casual users (the main user group) that can be found under the tab to the far left of the top menu bar, and parameters relevant to more experienced users found in the middle of the top menu bar. The content of the tab to the far right of the top menu bar gives the user the option to reset all changes made to any parameters back to their default value. The content within these three tabs is structured in the same way. All information is displayed as a list, and where necessary, the list has enabled scrolling. The lists

are organised with an optional category header where the content is divided into subcategories, followed by the category content. Each list entry consists of four elements; a label and a text describing the parameter found on the left hand side, a text field or button to edit the parameter on the right hand side, and a line beneath to divide the list entry from other list entries. The label and descriptive text are placed on the left hand side of the screen to ensure the user knows what parameter he or she is editing before making any changes. The absence of a “Save” button indicates that all changes are saved automatically. To promote consistency, the descriptive text, and the elements enabling parameter editing have a fixed placement along the x-axis of the screen, across all the different setting views [1]. By default the user will immediately access the “App settings” tab when entering settings, in order to display the most relevant setting parameters right away.

Application Logo

The application is called Magno. This name was derived from conversations with the project supervisor, Professor J. Krogstie. The logo is made up of the text “MAGN” and an orange circle with an eye in the middle. The entire logo is underlined with an orange double sided arrow. The circle along with the double sided arrow are elements collected from the motion test icon, signifying a dot moving back and forth. The circle represents the “O” in Magno, as well as the application test functionality. The eye represents visual processing, as the application tests the users’ visual processing abilities.

The application icon is made up of the orange circle with an eye in the middle, the same as in the complete logo. The representation is the same as in the complete logo.

5.2.2 Colour Choices

The colour choice of an application can have three different functions; either as an aid in structuring a screen, as a visual code for categorising and identifying information, or to make a screen more visually appealing to look at. Different colour choices can have a great impact on how an application is viewed and what functionality is implied, as well as the overall user experience.

The application in question has a blue theme. This choice was based on the research presented in section 3.3. Blue overlays can make it easier for dyslexics to read text. The menu bar has a soft, darker blue colour. The background is a very light blue colour. The top menu bar within the settings selection is slightly lighter than the

main menu bar, to break apart this functionality from the main menu bar. The buttons that are related have the same colour to signify this relation.

There are four different colours on the buttons in the application; blue, green, yellow, and grey. The buttons that are coloured blue have a functionality signifying navigational purposes, such as the menu bars, and the “Next” and “Back” buttons in the application tutorials. The colour blue was chosen for this purpose to maintain the blue theme of the application in order to promote consistency, and to make the text on the buttons easier to read for users with dyslexia. The green buttons have functionality that initiate something, either initiating a test or initiating changes to the application, such as in the “Reset” view. Green typically signifies the beginning of something, that something is activated, or a positive change. The chosen hue of green is also more prominent than the blue colour used in the rest of the application, as these buttons have a more important functionality. There is one yellow button in the application, “Copy extended results to clipboard”. The button’s functionality is different from the rest of the buttons within the application, and the difference in colour signifies this. Yellow is a fairly neutral colour, and yellow overlays make it easier for dyslexics to read, as described in section 3.3. All disabled buttons and text fields are grey with a dark grey font colour, to emphasise that these buttons are not available for interaction in the current view. When a toggle button is deactivated, it is also grey. Grey is a colour commonly used to signify that something is disabled, deactivated or otherwise not in function. By utilising established standards within the use of colour, the user interface designer ensures that the user is familiar with the meaning of the colour. It is important to maintain a degree of predictability within the interface design, to promote usability.

The application logo consists of two different colours, dark blue and orange. The dark blue colour is a part of the graphical profile, and is incorporated in the logo to promote consistency. The orange colour in the logo was added to implement some contrast in the logo. Orange is the complimentary colour to blue. Strong colours naturally catch the viewers gaze, and the logo should do exactly that [1].

5.2.3 Navigation Structure

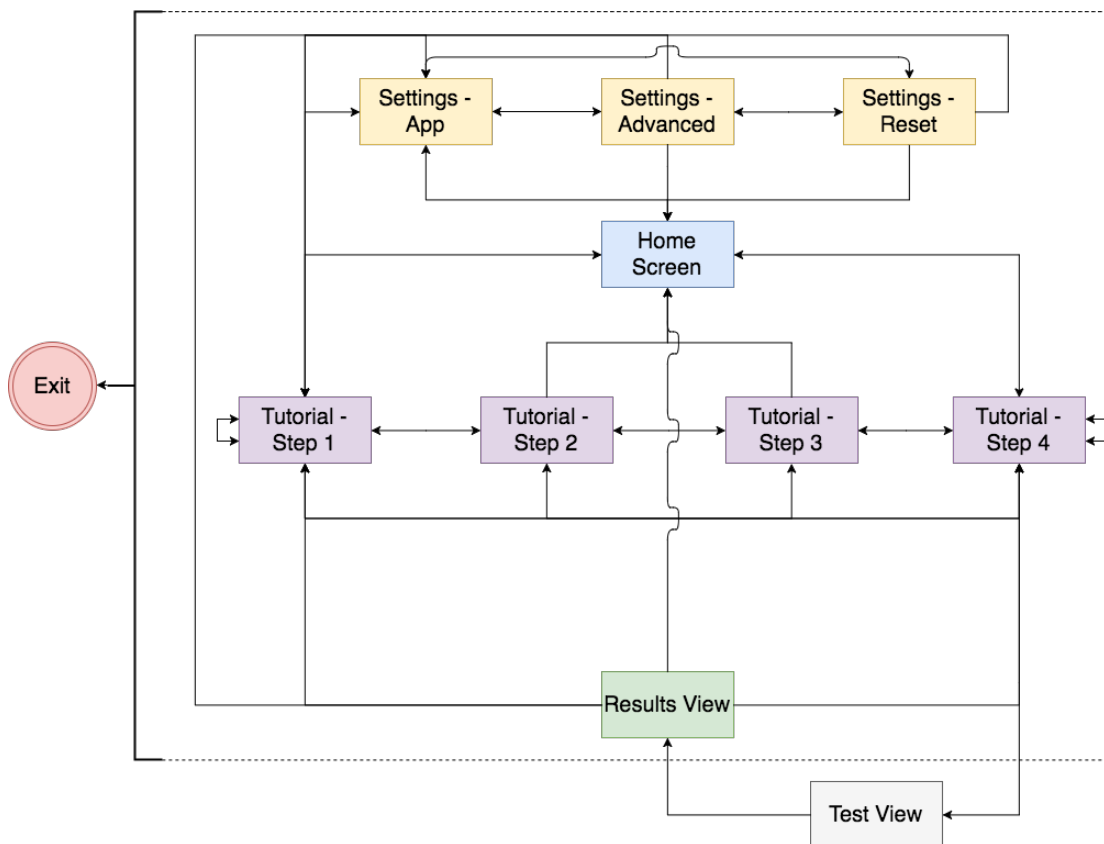


Figure 5.11: Navigational chart of application.

A graphical chart representing the structure of the different navigational choices can be seen in figure 5.11. As can be seen, most views can be accessed from the home screen, such as the first (and last, if the tutorial is turned off) step of the tutorial, the initial view within settings, and the possibility to exit the application. Within the different tutorials you can navigate within the tutorial to any step you might choose, as well as navigating to other tutorials, the main screen, and settings via the side menu bar. Within settings, the same navigational pattern is applied, where you can navigate freely between the different settings tabs, as well as navigating via the side menu bar. The only way to reach a test view is via the last screen in a tutorial, to make sure the user enters his or her age before beginning the test. The only way to reach the result view is to complete a test. From the result view you can navigate freely by the use of the side menu bar.




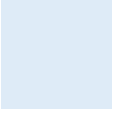
The navigational structure within the application is developed to ensure that the

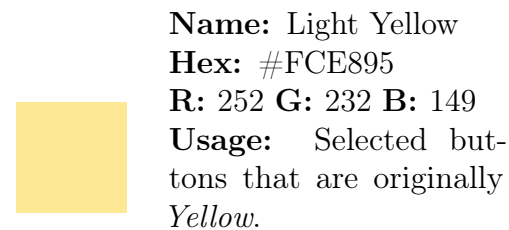
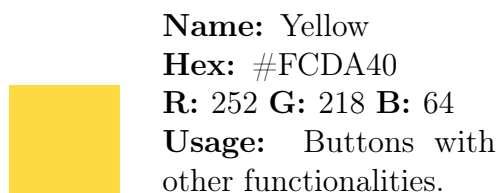
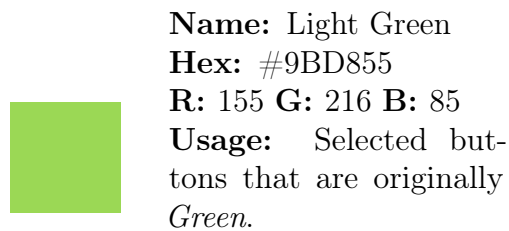
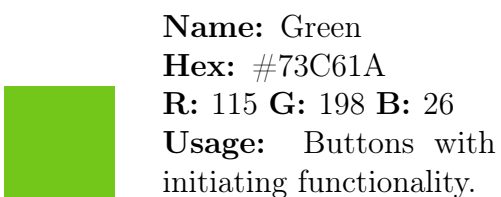
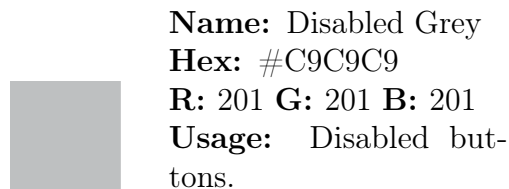
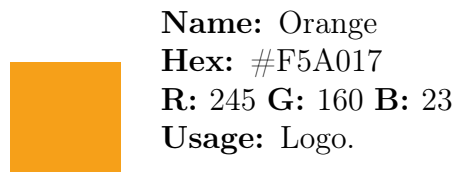
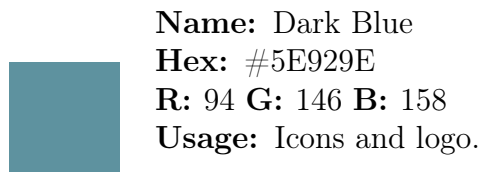
user is able to navigate easily. There are few functions that are visible and available at all times, but the ones that are available are general controls that are easy to understand. The first time a user utilises a system, he or she is not familiar with the navigational structure. Many users will not have the use for this application more than once, and therefore it is important with easy error correction. This means providing the user with clearly visible “Back” buttons, and the possibility to navigate between almost all screen categories no matter what the current location the user might have inside the applications structure. Within the tutorials, the navigational structure promotes progression, with a natural flow from left to right. The user is still provided with the possibility to freely navigate between the different tutorial steps to make sure the user understands all information given. All of the different views in the application can be reached within three clicks, with the exception of the result view. This is to prevent the user from feeling lost inside the application, and to prevent the user experience from being perceived as tiresome.

5.3 Final Graphic Profile

In this section we will present the final graphic profile, with the different colour codes, fonts, font sizes, icons, and logos used.

5.3.1 Colour Chart

	<p>Name: Main Blue Hex: #93CEEF R: 147 G: 206 B: 239 Usage: Main menu and buttons with progressive functionality.</p>		<p>Name: Light Blue Hex: #BEE2F4 R: 190 G: 226 B: 244 Usage: Selected buttons that are originally <i>Main Blue</i>.</p>
	<p>Name: Medium Blue Hex: #BDDBEF R: 189 G: 219 B: 239 Usage: Settings menu bar.</p>		<p>Name: Light Medium Blue Hex: #DEEFF7 R: 222 G: 239 B: 247 Usage: Selected buttons that are originally <i>Medium Blue</i>.</p>



5.3.2 Font

The font sizes are given in the format “*x.xf*”, where the numerical value represents the size in relation to the standard font size.

Helvetica **Name:** Standard text
Font size: Screen width * 20 / 1280
Font colour: Hex: #262626 R: 38 G: 38 B: 38

Helvetica **Name:** Standard disabled text
Font size: Screen width * 20 / 1280
Font colour: Hex: #7B7B7B R: 94 G: 94 B: 94

Helvetica **Name:** Header
Font size: 1.2f
Font colour: Hex: #262626 R: 38 G: 38 B: 38

Helvetica **Name:** Score
Font size: 1.5f
Font colour: Hex: #262626 R: 38 G: 38 B: 38

Helvetica **Name:** Score bar number
Font size: 1.1f
Font colour: Hex: #262626 R: 38 G: 38 B: 38

Helvetica **Name:** Score bar text
Font size: 0.7f
Font colour: Hex: #262626 R: 38 G: 38 B: 38

Helvetica **Name:** Feedback for correct answer
Font size: 0.8f
Font colour: Hex: #7AB642 R: 123 G: 182 B: 66

Helvetica **Name:** Feedback for incorrect answer
Font size: 0.8f
Font colour: Hex: #E24040 R: 206 G: 61 B: 58

Helvetica **Name:** Feedback for completed trial
Font size: 0.8f
Font colour: Hex: #93CEEF R: 123 G: 170 B: 197

5.3.3 Logo and Icons



Figure 5.12: Application logo.



Figure 5.13: Application icon.

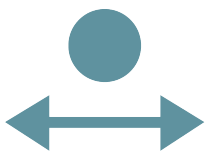
The application logo, seen in figure 5.12, is the graphic representation of this application. It is to be used in all settings where one wishes to represent the application graphically. It is displayed on the home screen of the application.

The application icon, seen in figure 5.13, is used as a button to direct the user back to the home screen. The application icon is also used as the launch icon on android devices. The application icon can be used in all settings where one wishes to represent the application, but there is not enough space available to incorporate the entire logo.

Name: Motion Test Icon

Usage: All places where the motion test is represented graphically.

Meaning: The circle represents the dots in the motion test, the arrow represents the horizontal coherent motion in the test.

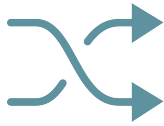


Name: Form Fixed Test Icon

Usage: All places where the form fixed test is represented graphically.

Meaning: The concentric circles with dotted lines represent the concentric circles in the test.

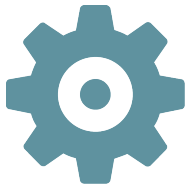




Name: Form Random Test Icon

Usage: All places where the form random test is represented graphically.

Meaning: Two arrows switching places is a standardised symbol for randomness.



Name: Settings Icon

Usage: All places where it is possible to change application settings.

Meaning: A gear is a standardised symbol for “Settings”.



Name: Exit Icon

Usage: All places where the one exits the application.

Meaning: The open door is a standardised symbol for exiting something.

5.4 Software Architecture

All of the functionality in the application was already implemented when we started to work on the user interface. The menu and settings screens were implemented by the use of the features built into libGDX. LibGDX provides a screen adapter class that holds a camera to display the content, and a stage to handle input and the behaviour of actors, like fields and buttons. The software was implemented with the model view controller (MVC) pattern for the motion and form tests. The classes representing the model of the MVC pattern, keep track of the current state and contain the functionality related to a test [4]. The classes representing the view and controller of the MVC pattern hold the interface and handle input from the user. We needed to do changes to the existing code to implement the new user interface. The use of MVC was a big advantage for us, as this made it possible to implement most of the new design without doing changes to the backend.

5.4.1 Classes

During the implementation, new classes were added, some classes were changed, and some of the classes were left unchanged. The class diagram can be seen in figure 5.14. The classes marked with green colour are new classes, the blue ones

are existing classes where major changes were made, and the white ones are classes that have remained the same or have no significant changes. The classes that are most important for the user interface are described below.

The *Assetloader* class loads sprites, textures, and fonts into memory when the application starts. When the application closes, it disposes of the objects to free up memory. Before we made any changes, a standard skin was used to load sprites, textures, and fonts into this class. To make it possible to make the different objects look like our design, a new skin was made and replaced big parts of the old skin.

The *MainMenu* class holds the main menu for the application. All the views that should contain the main menu extends this class. These classes are the *HomeScreen*, *TutorialScreen*, *SettingsMenu*, and *ResultsScreen*.

The *TutorialScreen* class is an abstract class that is extended by the different tutorial screens. The different tutorial screens contain many of the same objects, and the abstract class declares these objects as well as functions that are used by the tutorial screen classes. Some of the functions in *TutorialScreen* are abstract and are implemented by the classes that are extending *TutorialScreen*. The classes that are extending the *TutorialScreen* class are *TutorialSitDownScreen*, *TutorialTaskScreen*, *TutorialTrialScreen*, and *TutorialEnterAgeScreen*.

The application was made so that the *Screen*, *World*, and *Renderer* classes made up the MVC-pattern for the tests. The *Screen* class is the controller; it handles user events, and notifies the *World* class of any changes. Specifying the functionality and behaviour of the tests takes place in the *World* class, that plays the role of the model. The *Renderer* class renders the objects specified in the model [4], and is part of the view in the MVC pattern. As the *TutorialTaskScreen* class and the *TutorialTrialScreen* class contain a version of the tests, we needed to add new abstract classes to make this possible without touching the code creating the actual tests. These are called *AbstractFormRenderer*, *AbstractMotionRenderer*, *AbstractFormWorld*, *AbstractMotionWorld*, and *AbstractInputHandler*. These classes contain the objects and functions we did not need to change in order to make the trial and the task illustration possible. The functions that needed to be changed were made as abstract functions and were implemented in the classes that extend these abstract classes. The classes *FormRenderer*, *MotionRenderer*, *FormWorld*, *MotionWorld*, and *InputHandler* together with the abstract classes do not contain any changes in terms of functionality. The new *Renderer*, *World*, and *InputHandler* classes hold the functionality for the tutorial. In the actual test, the patches will empty when a given amount of time has passed. This functionality was not wanted in the tutorial task description, and was removed. In the tutorial trial, the changed

functionality made it possible to give the user feedback and let the user pick patches three times instead of taking the whole test. The latter was done by adding some extra states called TRIALCORRECT and TRIALWRONG.

The *SettingsMenu* class instantiates and builds the menu bar at the top of the screen in all the setting views. All views that should contain this menu bar extend this class. These classes are the *AppSettings* class, *ResetSettings* class, and the *AdvancedSettings* class. The *SettingsMenu* class extends the *MainMenu* class, as all views that should have the setting menu bar should also have the main menu bar.

The *AppSettings*, *AdvancedSettings* and *ResetSettings* classes build the content of the three corresponding views. The different click listeners in these classes edit values stored in the *Settings* class, where all setting parameters and default values are stored. The *AdvancedSettings* class does not instantiate its elements, because it contains a lot of elements. The instantiation of the elements has been moved to its own class; *AdvancedInstances*. The different tutorial views, as well as the setting views have swipe functionality incorporated, making it possible to swipe between the different setting views, or the different steps of the tutorial. Because each swipe movement leads to a different view based on what the current view is, it was not possible to add this piece of code in a super class. All classes that have swipe functionality has to have approximately the same piece of code, with minor differences.

The *ResultsScreen* class instantiates and builds the content of the result view. The result displayed, both in text and with the progress bar, is fetched from the *Results* class, that calculates the threshold score based on the user's performance during the test.

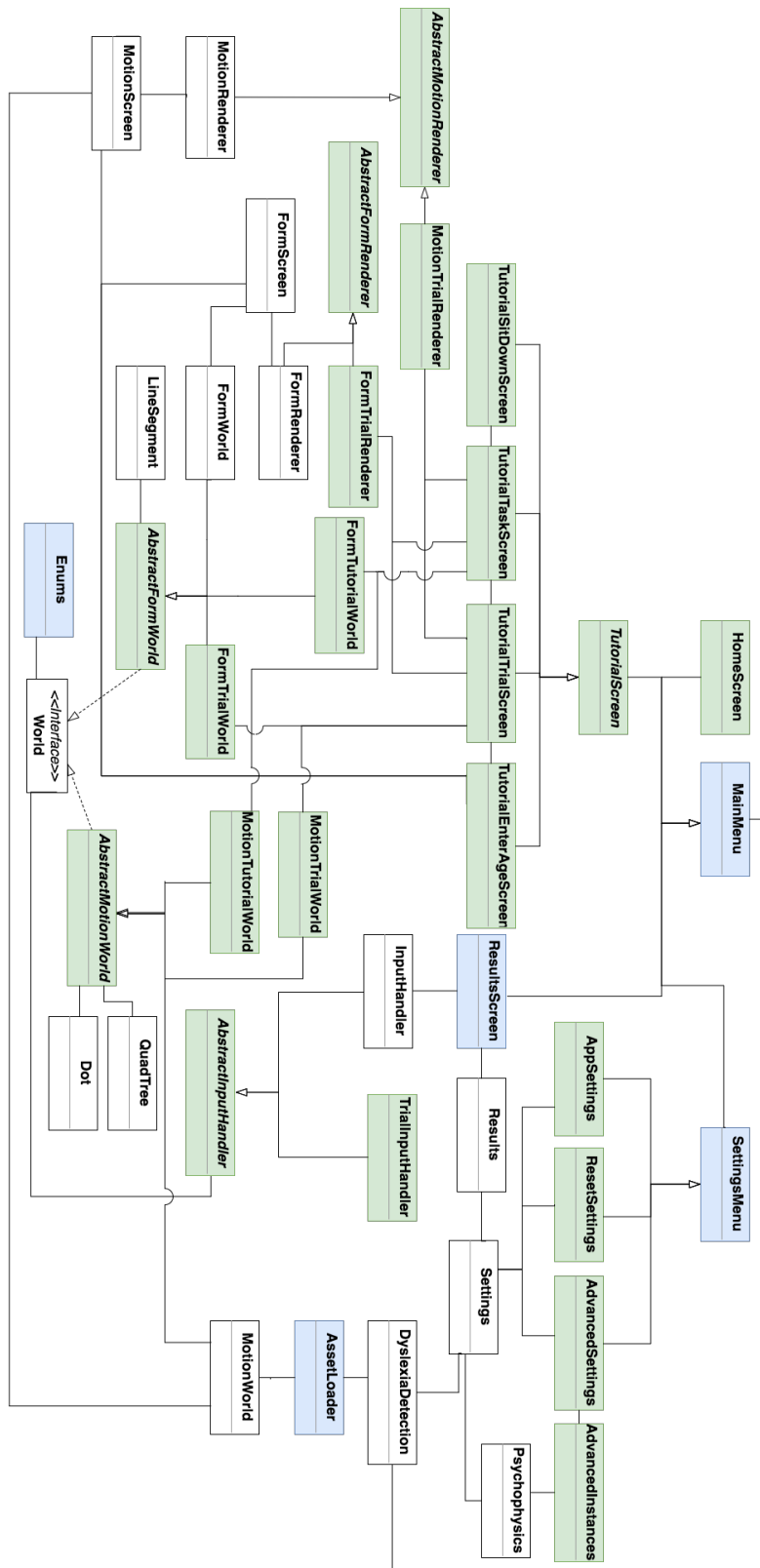


Figure 5.14: Class diagram

Chapter 6

Evaluation

This chapter will look at several different test results regarding the current interface design and review them regarding to a general standard. In addition, the interface design is to be evaluated in light of the predetermined requirements, found in section 3.8. In section 6.1, a general evaluation is given, and in section 6.2 the technical tools used are evaluated. The requirement fulfilment is evaluated in section 6.3.

6.1 Overall Evaluation

The feedback from the usability tests described in section 4.1.2, was overall very positive. By observing the test subjects, we have seen that the test subjects had no severe problems in understanding the application or its purpose.

Unfortunately, the digital prototype was never tested on any test subjects who had been diagnosed with dyslexia. In addition, the test subjects were all collected from a very specific user group; students attending information technology (IT) related studies within a small age range (18 to 25 years). This could have had an impact on the general feedback. Students within this age group are part of the target audience, but the target audience also includes other user groups that might not have the same experience with the use of technology and applications. This means that there are aspects of the interface design that have not been put to the test, such as the impact colour use might have on dyslexic users. If a person already has been diagnosed with dyslexia, they are not part of the target user group, but they are important in the testing process as people who have dyslexia but are not still diagnosed are part of the target user group. The test subjects were collected from

the above mentioned user group because these were the users that were available during the allotted time space, and because these people were the ones easiest to reach with our current network.

The finished digital prototype was not retested before implementation, as there were no major changes made. The initial plan was to test these changes when they were implemented. The short application response time makes the application easier to use than a digital prototype, but due to time constraints the implemented application design was not tested either. One can argue that because of the small amount of changes made this is not a major problem. Our experience also tells us that all changes made most likely have a positive impact on usability. The finished application should still be usability tested on a larger, more diverse test group before any potential release to the public.

That being said, within the utilised test group, the first digital prototype performed very well. The accumulated SUS score of 92.7 indicates that users find the application very easy to use, with a straight forward design. The finished interface design has a transparent navigational structure, in order to make it easy to understand, and the testing showed that this assumption was correct. The problem areas that were identified during testing can be found in section 4.1.2. Measures were taken to improve on all problem areas, as can be read in section 4.1.3. The problem areas were discovered through what is called qualitative feedback, in contrast to the SUS score that was derived from quantitative feedback. This means that measuring the effect of the improvements on these problem areas is more difficult. The only way to do so is to observe that the majority of users no longer experience any difficulty in these areas. One must keep in mind that even with these problem areas present, the interface design still had a SUS score far within the acceptable range.

Overall, the feedback on the design is that it is easy to understand and pleasing to look at. Regardless of problem areas, this is a satisfactory result. Optimally, all problem areas should be eliminated, but it is impossible to eliminate all potential problem areas for all users. It is also important to remember that the remaining problem areas are small compared to the initial problem areas; it is now possible to use the application without the need of a supervisor.

6.2 Evaluation of Technical Tools

This section provides an introduction to the relevant tools and technologies that have been used during this project.

Trello

In order to organise the work flow and keep track of what needed to be done in this project, a project management tool was needed. Both authors had previous experience with a tool called Trello. Trello is a web-based project management application that is free, flexible, and a visual way to manage projects and organise anything. It is a cloud based software, and is available both through an application and directly in the web browser [27]. Trello was chosen as the preferred project management tool, as it can assign members to different tasks, and the user can customise the work flow to fit the projects needs. The fact that both authors had previous experience with this tool also weighed in heavily on this decision.

Adobe Illustrator

We needed a tool to develop the different screens for the digital prototype, and to develop the skin, logo, and icons for the application. Both of the authors had experience with Adobe Illustrator. This is a vector graphics application where you can make logos, icons, sketches, typography and complex artwork for print, web, interactive media, video, and mobile devices [28]. We also reviewed other programs that offered vector graphics development, but none of the programs suited our needs in the same way as Illustrator.

InVision

In order to perform usability testing on a digital prototype, a tool was needed to develop a digital prototype. InVision is a prototyping, collaboration, and workflow platform. InVision makes it possible to design, review, and usability test products without coding. With tools for prototyping, task management, and version control, it incorporates the entire design process. You can upload your design files and add animations, gestures, and transitions to transform your static screens into clickable, interactive prototypes [29]. Both authors had previous experience with other prototyping tools, such as Axure [30], but this tool was not free, and too time-consuming and complicated. One of the authors had tried out InVision, and found this tool more user friendly than Axure, hence InVision was chosen. Had this decision been taken later in the project, another tool, Figma [31], had probably been chosen. Figma is a free online tool that allows for creation of views directly in the web browser, and images do not have to be uploaded. This functionality had eliminated the need for Adobe Illustrator as well. It also allows for editing in real

time, by multiple users. Figma was not chosen, as it was not a familiar tool to the authors at the stage of the project where the digital prototype was developed.

Lookback

When performing usability testing, it is not always easy to get a hold of everything that is being said and done. We wanted a tool that could make recording of the usability tests easier. We had heard of people using Lookback in their projects, and wanted to give it a try. Lookback is a software used to interview and record the face, screen, and voice of anyone. This is very useful during usability testing. There is no need for extra cameras, hard drives, or futzing with footage [32]. The fact that Lookback also records the activity on the screen was a big advantage for us. This was very useful to look at after the tests were completed to recall where the problem areas were located.

Android Studio

In order to implement the user interface design, a text editor tool was needed. Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA [33]. On top of IntelliJ's code editor and developer tools, Android Studio offers more features that enhance productivity when building Android apps. Some examples of these features are a flexible Gradle-based build system, a fast and feature-rich emulator, an unified environment where you can develop for all Android devices, and instant Run to push changes to your running app without building a new APK. Android Studio is distributed with templates for creating activities and with the possibility to import samples from the Google Samples repository. With the included GUI editor it is possible to click and drag elements to create a user interface. Together with the built in terminal it provides a complete package to develop Android applications [34]. Android Studio was chosen as it makes for easy testing on android devices and emulators, as well as being a familiar tool to the authors.

OpenGL ES

OpenGL for embedded systems is an API-specification designed to work on systems such as phones, without a desktop environment. It is a free cross-platform tool. With the use of a subset of the OpenGL standard, it can create a low-level interface between the software and hardware layer. This ensures a reliable, high

performance. OpenGL ES is compatible with standard OpenGL libraries, meaning it is possible to run OpenGL applications on almost any system that supports it without modifications. The OpenGL standard supports 2D and 3D objects, as well as displaying CAD drawings, medical images and virtual reality environments [35]. OpenGLs functionality was needed for the project, and was hence the preferred API-specification.

libGDX

In order to implement a lot of functionality in a short period of time, a Java library was needed. libGDX is cross-platform Java game development framework based on OpenGL (ES) that works on Windows, Linux, Mac OS X, Android, your WebGL enabled browser, and iOS. It supports low-level access to file systems, input devices and OpenGL through a unified OpenGL ES 2.0 and 3.0 interface. Alongside the OpenGL interfaces it provides APIs that renders text and sprites, with built in functions for linear algebra and trigonometric calculations [36]. Because one of our main priorities was to preserve the integrity of the application tests, and the test layout was developed with the use of libGDX, it was decided to continue with the use of libGDX when further implementing the GUI. In addition, this would prevent the need to make drastic changes to the backend of the code, potentially decreasing the workload. The downside of this decision was that it created the need to review the entire libGDX library, which neither of the developers were familiar with previous to beginning this project. LibGDX is not the most extensive library, meaning that a review of the library was not too difficult. On the other hand, libGDX does have a few shortcomings in its offered functionality, and limited the possibilities in what could be accomplished when striving for the goal to make the design resemble the digital prototype as much as possible.

Skin

The most practical way to change the appearance of libGDX widgets is by the use of Skins. Skin is a part of libGDX and the Skin class stores resources for user interface widgets to use. It is a container for texture regions, ninepatches, fonts, colours, and so forth. Resources in a skin typically come from a texture atlas, widget styles, and other objects added to the skin via code. As a starting point, Skin files from the libGDX tests can be used [37]. This enables you to quickly get started using scene2d.ui and later you can replace the skin assets with your own skin [38].

SkinComposer

SkinComposer is a program that makes the creation of skins for libGDX easier. SkinComposer imports images, bitmap fonts and colours, and exports them as a json file along with a matching texture atlas. The program provides a live preview of the skin, with customisation options. SkinComposer also supports adding additional classes to expand the functionality beyond the base Scene2D.ui widgets [39]. It is time consuming to create a skin manually from scratch, and none of the authors had any previous experience with creating skins. Even though SkinComposer is a volatile program, it was more user friendly than creating a skin manually. As a result, we chose to use SkinComposer to create the skin used in the application.

Scene2d.ui

Scene2d is libGDX's 2D scene graph. It provides basic 2D scene graph functionality at its core. This is functionality like actors, groups, drawing, events, and actions. In game development, Scene2d is practical as most actors are application specific. For building user interfaces, the scene2d.ui package provides common user interface widgets and other classes built on top of scene2d [40].

Packr

Packr is a tool that helps with the packaging of libGDX applications. It bundles your JAR file with a JRE file, so that end users do not have to install Java. It also creates a platform specific directory structure, makes your app look more native with a platform specific native executable for Linux, Mac, or Windows, and minimises the bundled JRE file [41].

Git

As we were two people who were going to develop the user interface of the application, we needed a version control system to make it easier to collaborate. We quickly chose Git since both of us had previous experience with it. Git is a free and open source distributed version control system designed to handle projects with speed and efficiency. It is easy to learn, has a tiny footprint, and fast performance. Git allows and encourages you to have multiple local branches that can be entirely independent of each other [42].

GitHub

We also had prior experience with GitHub. GitHub is a development platform inspired by the way people work. GitHub has built-in review tools that make code review an essential part of a team's process. You can also plan and manage projects from your repositories, create well-maintained docs, and make sure they receive a high level of care [43]. To make the collaboration and code structure better we chose to use GitHub.

ShareLaTeX

In earlier projects, we had both used ShareLaTeX to write reports. ShareLaTeX is an online collaborative LaTeX editor, with real-time editing incorporated. This allows for easy writing, graphical layout, and compiling, especially with multiple authors. As ShareLaTeX is available online, there is no need for installation [44]. All of these features, along with an incorporated review tool, made us choose to use ShareLaTeX to write and format this thesis.

6.3 Requirement Fulfilment

This section evaluates the functional and non-functional requirements.

6.3.1 Functional Requirement Fulfilment

Table 6.1 evaluates the functional requirements for the application. "Requirement #" is the requirement number, "Evaluation" describes or evaluates the requirement, while "Fulfilment" describes if the requirement was met or not. The requirements can be found in section 3.8.

Requirement #	Evaluation	Fulfilment
FR1	The application responds well to different screen sizes. If the screen resolution is too small, the formatting might look slightly different, but the application is not meant to work on smaller devices such as smart phones.	Attained
FR2	A tutorial has been implemented, and is fully functional.	Attained
FR3	The score view includes both a textual explanation and a graphical representation of the threshold score.	Attained
FR4	Within the application test tutorials, there is both a textual description of the tests, a graphical demonstration, and the possibility to try out the test beforehand.	Attained
FR5	An arrow has been added to the test views in the application, allowing the user to exit the test without the use of external controls.	Attained
FR6	The LibGDX backend used does not support this functionality without importing custom cursor graphics. The cursor does not change form when hovering over clickable content, but the clickable elements do change colour when hovering and the need for the mouse to change form is therefore not very pressing.	Not attained
FR7	The different setting parameters in the application are first roughly sorted within two different categories; “ <i>App settings</i> ” and “ <i>Advanced settings</i> ”. Within the “ <i>Advanced settings</i> ” tab, the different parameters are further sorted within different categories.	Attained
FR8	It is possible to skip a tutorial without completing it. It is also possible to turn off the tutorial, so that it does not appear when entering the different tests.	Attained

Requirement #	Evaluation	Fulfilment
FR9	The application can be used without a supervisor or any previous knowledge of the app, given that the user has some reading skill. Graphical representations make it possible to understand the application partially without reading, but there is no guarantee that the user will understand the application correctly.	Partly attained
FR11	The application does not store test results, setting parameter values connected with the result, age or any data from the completed tests, neither locally or externally. This should be possible to implement, but since this functionality is heavily based on backend development, and is outside the project description, this was not prioritised.	Not attained

Table 6.1: Functional requirements evaluation.

FR1, FR2, FR3, FR4, FR5, FR7, and FR8 have all been attained. FR9 is partly attained. There is no guarantee that the user will understand the application if he or she is not able to read the application description and tutorial explanation. FR6 and FR11 are not attained. In order to change the appearance of the cursor in the desktop version of the application, custom cursor graphics must be incorporated into the application, as the libGDX backend used, does not have this functionality originally incorporated. This had to have been done in conjunction with creating the custom skin used in the implementation, but the program used to create the skin files (SkinComposer) corrupted the original skin save file making it impossible to make changes to the skin without creating it from scratch again. Due to the time constraint of this project, it was not prioritised to do this in order to meet this requirement, even though the requirement had a high priority. As mentioned in the explanation, the clickable elements do change colour when hovering and makes this requirement less important. Since this project focuses on the development and implementation of a GUI, and does not encompass implementing backend functionality such as storage, exporting, and processing of data, FR11 was not prioritised. In addition, FR11 has a low priority in the project.

8 out of 10 requirements have been met, where one of the requirements that were not met had a low priority. This is considered high requirement fulfilment.

6.3.2 Non-Functional Requirement Fulfilment

Table 6.2 evaluates the non-functional requirements listed in chapter 3.

Requirement #	Evaluation	Fulfilment
U1	It is possible to reach any given system view with three or less clicks.	Attained
U2	During usability testing, the test subjects used on average 1 minute and 20 seconds to read the text on the home screen, and complete a tutorial.	Attained
U3	The accumulated SUS score of the last round of usability tests was 92.7.	Attained

Table 6.2: Non-functional requirements evaluation.

All non-functional requirements have been attained, well within the set margins. This means the application is user friendly enough, based on the standard decided prior to the project start.

Chapter 7

Discussion, Conclusion, and Further Work

In this chapter we present a discussion around the project, along with a conclusion and further work that needs to be done. In section 7.1, the results from the evaluation, and the project development in general, are discussed. In section 7.2, the research questions are answered, and in the last section, section 7.3, alternatives to further work are presented.

7.1 Discussion

This project began as a continuation of a previous master's thesis, viewed in section 3.5. The theoretical research and backend implementation of the precursor to this project was thoroughly conducted. Still, in order to be able to distribute an application to a larger user group, with several different needs, it is important that the application not only has working functionality, but also a user interface that promotes that functionality. Without a proper user interface, the functionality is not conveyed, and usability decreases. The finished user interface design has its shortcoming, based on the fact that it has not been sufficiently usability tested, especially for a target group with greater diversity. There have been made changes that can not be corroborated as to whether they improve on the design or not, even though we have sufficient experience with user interface design to say that the changes most likely will have a positive impact. It is still safe to say that the current GUI, developed during this project, provides for a much better user experience than the previous user interface.

During this project, a thorough review on the narrow field of user interface design specifically tailored to dyslexics needs has been conducted, and this is perhaps some of the most pioneering research done in this paper. Many aspects of user centred design are universal no matter who resides in the target user group, and user centred design principles have been thoroughly researched and thought through by others. A part of the user centred design process is to observe the target users and their needs, but the existing theory around designing specifically for dyslexics has proven to be insufficient.

The iterative process of creating a user interface design is time consuming, and to create a user interface design in about 6 months, with the proper testing and implementation needed, is probably not an optimal time estimation. This postulate is based on our experience with this project. Recruiting diverse test subjects without external help has proven difficult, and technical issues have slowed down the process. That being said, the process has been exceedingly educational in regards to several aspects; how to interact with test subjects in order to extract the most vital information during usability testing, how to quickly familiarise ourselves with new technologies, the different aspects of dyslexia, and responsive design in applications. When coming across a problem, the ability to work around the issue and find solutions has been vital to the project. Encountering obstacles increased both learning and motivation in us. This challenged our expertise and current knowledge, forcing us to look beyond our existing knowledge space and acquiring new experiences and further understanding of a subject and the world around us.

7.2 Conclusion

The objectives of this project were to further develop and implement a new user interface. Research questions were made to help in steering the research and development in the correct direction. The user centred design process was used to develop the user interface, and usability testing was used to collect data about the usability of the design. The results of the usability testing were analysed to help us answer the research questions that have guided the project. A technical review of the application developed prior to this project helped us in implementing this design. The conclusion to this project will review these research questions and determine whether or not they have been adequately answered.

RQ-1: What are the problem areas with the user interface design developed in the specialisation project?

There were found several problem areas with the user interface developed in the specialisation project. Two paper prototypes of the user interface were created. It was clear that they were not perfect, as we found several problem areas in both prototypes. The main problem area with the first prototype, the one without a menu on the left hand side, was that it was hard to fully understand the tutorial. This was because it was not clear that you are to choose a patch several times, it was hard to understand what to look for during the tests, and it was not clear that you could exit the tutorial. Some other problem areas were that it was not easy to understand the test score and the purpose of the application. Looking at the second paper prototype, there was only one problem area in addition to the ones identified in the first paper prototype. The problem areas in the second prototype were that it was hard to fully understand the tutorial, the purpose of the application, and understanding the blank space in the main view.

RQ-2: How can the interface design developed in the specialisation project be improved, in order to further promote usability?

The user interface design could be improved by making changes to the tutorial that would promote usability further. To ensure that the users understand what to look for during the tests, the illustration showing what to look for was improved. The illustration indicates the patch containing coherent moving dots or concentric circles. To make it easier to understand that you are to choose a patch several times, the usability was improved by enhancing the trial. The trial lets the user try the test functionality a few times, and gives feedback on whether or not the user has chosen the correct patch. This made sure that the users understood what to look for and understood how the test works. If the user should wish to skip the tutorial, it was better to use a button with the text "*Skip*", rather than an "X" icon. This is more user friendly as it explains the functionality of the button better than the "X" icon. To make the users fully understand purpose of the application, it was clear that some general information about the application needed to be available for the users. As this is something you may need to read the first time using the application, this information was placed on the home screen. This information eliminated the blank space found in the second paper prototype as well. It was also clear that the menu on the left hand side made the users feel in control and maintain an overview of the application's navigational structure. This promotes usability. Because of this, we decided to continue with a menu similar to the one in the second paper prototype. It is clear that the changes made, have improved

the usability. The latest SUS score was 92.7, and all the usability requirements have been met.

RQ-3: How can we compliment existing code with the implementation of the new user interface design?

In order to compliment the existing code, we reviewed the frameworks and libraries previously used. This helped us to understand which parts of the code belonged to the frontend, and which parts belonged to the backend. As our task description encompasses frontend development, we could easily understand which parts of the code to change and not.

RQ-3.1: What technical tools are best suited to aid in implementing a user interface design, based on previously implemented functionality?

The previously implemented functionality and design was implemented in Java with the use of libGDX. As none of us had experience with the use of libGDX, we wanted to try to use XML instead of making a new skin. We could see that making a new skin was complicated and not very easy. After some research, we quickly understood that the use of libGDX and skin was the best solution after all to avoid re-implementing the existing functionality. Luckily, we found a tool that helped us make the new skin for the user interface, called SkinComposer. It was not a very steady tool, but it was the best tool available. We also needed an IDE to use while implementing. First we wanted to use IntelliJ due to good experiences with the tool, but since libGDX uses Gradle, Andorid Studio was the most suited tool. We also found the use of Git and GitHub helpful. This made it possible for the two of us to implement the new design simultaneously, and to keep track of changes made in the code.

7.2.1 Final Conclusion

In our opinion, the application sufficiently answers the research questions and fulfils the task description. The usability of the application has been significantly improved, and it is considerably easier to take a test if you have never used the application before. There will always be areas for improvement, but our thesis manages to address the majority of elements specified prior to the project initiation, in the form of task description, research questions, and requirements.

The application delivered along with this thesis can be taken into use as is, though we recommend further development.

7.3 Further Work

This project has had its main focus on interface design development and implementation. While the interface design is finished, there are still some aspects that can be further developed before deploying this application to the masses. Some key elements for further work are as follows:

- Implementation of non-acquired requirements.
- Final usability testing and fine tuning of the interface design.
- Implementation of functionality enabling selection of language.
- Implementation of external storage of application data.
- Implementation of information processing, for processing of stored application data.
- Implementation of feedback to the user based on processed information, such as age and/or test score.

When all these elements have been considered and potentially implemented, one can consider revisiting and revising the requirements. One might see it necessary to add or subtract further functionality, and means of distribution must also be considered. Specifying the environments in which the application is to be used is key, and narrowing down the target audience. It is also important that the application tests are thoroughly documented and tested in order to validate any data delivered by the application.

References

- [1] Wilbert O. Galitz. *The Essential Guide to User Interface Design*. John Wiley Sons, Inc., 2002. ISBN: 0471084646. URL: <http://ps.fragnel.edu.in/~dipalis/prgdownl/eguid.pdf>.
- [2] Anne Lene Blystad. *Essensielt å oppdage dysleksi tidlig*. 2016. URL: http://www.dysleksiforbundet.no/no/rettigheter+rad/dysleksi/Essensielt+%C3%A5+oppdage+dysleksi+tidlig.b7C_wtFM18.ips.
- [3] Dyslexia International. *Better Training, Better Teaching*. 2014. URL: <http://www.dyslexia-international.org/wp-content/uploads/2014/10/DIReport-final-4-29-14.pdf>.
- [4] Bjørnar Håkenstad Wold. “App for Early Detection of Dyslexia”. MA thesis. Norwegian University of Science and Technology (NTNU), 2016.
- [5] Maja Kirkerød and Thea Hove Johansen. “Designing an Application for Detection of Dyslexia”. 2016.
- [6] Briony J. Oates. *Researching Information Systems and Computing*. SAGE Publications Ltd, 2006.
- [7] John Stein. “The magnocellular theory of developmental dyslexia”. In: *Dyslexia* 7.1 (2001), pp. 12–36. ISSN: 1099-0909. DOI: [10.1002/dys.186](https://doi.org/10.1002/dys.186). URL: <http://dx.doi.org/10.1002/dys.186>.
- [8] IDA Board of directors. *Definition of Dyslexia*. 2002. URL: <https://dyslexiaida.org/definition-of-dyslexia/>.
- [9] British Dyslexia Association. *Dyslexia Definitions*. 2016. URL: <http://www.bdadyslexia.org.uk/dyslexic/definitions> (visited on 10/20/2016).
- [10] Oxford English Dictionary. *Dyslexia*. 2016. URL: <https://en.oxforddictionaries.com/definition/dyslexia> (visited on 10/20/2016).

- [11] Dyslexia Research Trust. *Genetics of Dyslexia*. 2016. URL: <http://www.dyslexic.org.uk/research/genetics-dyslexia> (visited on 10/20/2016).
- [12] H. Sigmundsson. “Do visual processing deficits cause problem on response time task for dyslexics?” In: *Brain and cognition* 58.2 (2005), pp. 213–216.
- [13] P. C Hansen et al. “Are dyslexics’ visual deficits limited to measures of dorsal stream function?” In: *Neuroreport* 12.7 (2001), pp. 1527–1530.
- [14] H. Sigmundsson, P. C. Hansen, and J. B. Talcott. “Do ‘clumsy’ children have visual deficits”. In: *Behavioural Brain Research* 139.1 (2003), pp. 123–129.
- [15] The Free Dictionary Medical Dictionary. *Ectopia*. 2016. URL: <http://medical-dictionary.thefreedictionary.com/Ectopia> (visited on 01/11/2016).
- [16] The Free Dictionary Medical Dictionary. *Temporoparietal*. 2016. URL: <http://medical-dictionary.thefreedictionary.com/Temporoparietal> (visited on 01/11/2016).
- [17] John Stein. *The Current Status of the Magnocellular Theory of Dyslexia*. Powerpoint slide published at slideplayer.com. URL: <http://slideplayer.com/slide/6120/> (visited on 01/11/2016).
- [18] Caroline Solem. *Dysleksi: Spørsmål og Svar*. 2016. URL: <http://www.dysleksiforbundet.no/no/rettigheter+rad/dysleksi/Dysleksi\%3A+Sp\%C3\%B8rsm\%C3\%A5l+og+svar.9UFRjG1S.ips> (visited on 12/02/2016).
- [19] John Krogstie, Bjørnar Håkenstad Wold, and Hermundur Sigmundsson. *App for Early Detection of Dyslexia - Implementation and Ongoing Evaluation*. 2016. URL: <http://ojs.bibsys.no/index.php/Nokobit/article/view/386> (visited on 12/08/2016).
- [20] V. Goodwin and B. Thomson. *Making Dyslexia Work for You*. Taylor & Francis, 2013. ISBN: 9781136631504. URL: https://books.google.no/books?id=GH_p1JsYnzkC.
- [21] Netania Engelbrecht. *How to Design For Dyslexia*. 2015. URL: <http://blog.usabilla.com/how-to-design-for-dyslexia/> (visited on 10/21/2016).
- [22] The Business Dictionary. *Justified text alignment*. 2016. URL: <http://www.businessdictionary.com/definition/justified-text.html> (visited on 02/11/2016).

- [23] Luz Rello and Ricardo Baeza-Yates. “Good Fonts for Dyslexia”. In: *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '13. Bellevue, Washington: ACM, 2013, 14:1–14:8. ISBN: 978-1-4503-2405-2. DOI: [10.1145/2513383.2513447](https://doi.org/10.1145/2513383.2513447). URL: <http://doi.acm.org/10.1145/2513383.2513447>.
- [24] Jan Gulliksen et al. “Key principles for user-centred systems design”. In: *Behaviour & Information Technology* 22.6 (2003), pp. 397–409. DOI: [10.1080/01449290310001624329](https://doi.org/10.1080/01449290310001624329). URL: <http://dx.doi.org/10.1080/01449290310001624329>.
- [25] U.S. Department of Health Human Services. *User-Centered Design Basics*. URL: <https://www.usability.gov/what-and-why/user-centered-design.html> (visited on 10/31/2016).
- [26] Samsung. *Samsung Galaxy Tab A*. 2017. URL: <http://www.samsung.com/us/mobile/tablets/all-other-tablets/sm-t580nzkaxar-sm-t580nzkaxar/> (visited on 05/21/2017).
- [27] Inc Trello. *Trello*. 2017. URL: <https://trello.com> (visited on 05/19/2017).
- [28] Adobe Systems Software Ireland Ltd. *Design flotte vektorillustrasjoner*. 2017. URL: <http://www.adobe.com/no/products/illustrator.html> (visited on 04/12/2017).
- [29] InVision. *Design Better. Faster. Together*. 2017. URL: <https://www.invisionapp.com/> (visited on 04/12/2017).
- [30] Axure. *Axure*. 2017. URL: <https://www.axure.com/> (visited on 05/30/2017).
- [31] Figma. *Figma*. 2017. URL: <https://www.figma.com/> (visited on 05/30/2017).
- [32] Lookback. *Simple, powerful user research*. 2017. URL: <https://lookback.io/> (visited on 03/10/2017).
- [33] JetBrains. *IntelliJ IDEA*. 2017. URL: <https://www.jetbrains.com/idea/> (visited on 04/12/2017).
- [34] Android. *Meet Android Studio*. 2017. URL: <https://developer.android.com/studio/intro/index.html> (visited on 04/12/2017).
- [35] Khronos Group. *The Standard for Embedded Accelerated 3D Graphics*. 2017. URL: <https://www.khronos.org/opengles/> (visited on 04/12/2017).
- [36] Mario Zechner. *libGDX*. 2017. URL: <https://github.com/libgdx/libgdx/wiki> (visited on 04/12/2017).

- [37] libgdx. *libGDX tests*. 2017. URL: <https://github.com/libgdx/libgdx/tree/master/tests/gdx-tests-android/assets/data> (visited on 05/19/2017).
- [38] Julien Villegas. *Skin*. 2017. URL: <https://github.com/libgdx/libgdx/wiki/Skin> (visited on 05/19/2017).
- [39] Raymond Buckley. *SkinComposer*. 2016. URL: <https://ray3k.wordpress.com/software/skin-composer-for-libgdx/> (visited on 05/26/2017).
- [40] Julien Villegas. *Scene2d.ui*. 2016. URL: <https://github.com/libgdx/libgdx/wiki/Scene2d.ui> (visited on 05/19/2017).
- [41] D. Ludwig. *Packr*. 2016. URL: <http://www.badlogicgames.com/wordpress/?p=3428> (visited on 05/19/2017).
- [42] git. *git -local-branching-on-the-cheap*. 2017. URL: <https://git-scm.com/> (visited on 04/12/2017).
- [43] GitHub. *Built for Developers*. 2017. URL: <https://github.com/> (visited on 04/12/2017).
- [44] Henry Oswald and James Allen. *ShareLaTeX*. 2014. URL: <https://github.com/sharelatex/sharelatex> (visited on 05/20/2017).

Appendix A

Overview of the Existing Application

The screen captures below are taken with the default settings specified in the application.

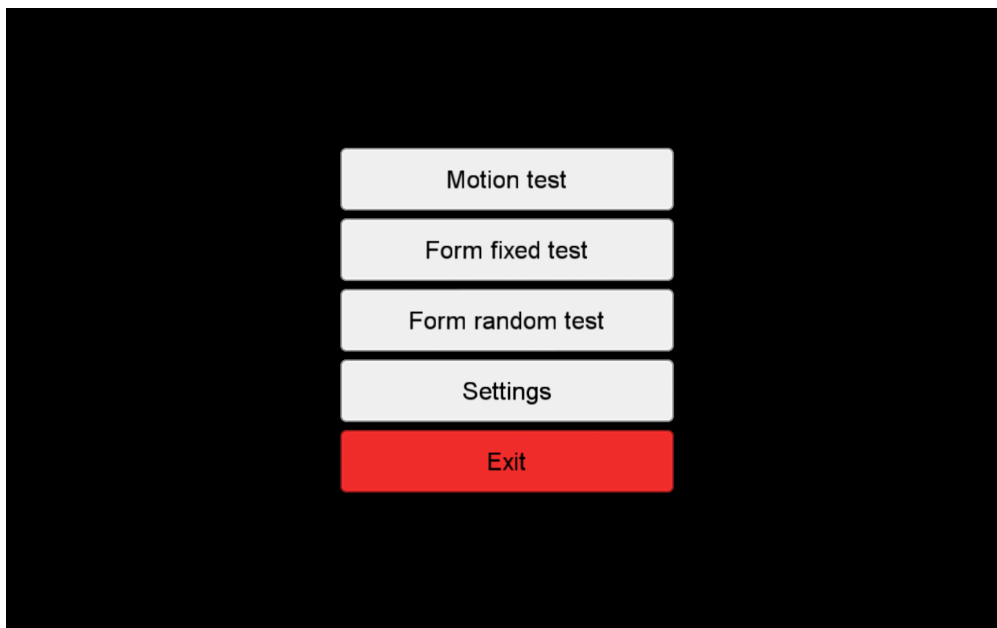


Figure A.1: Main menu view.

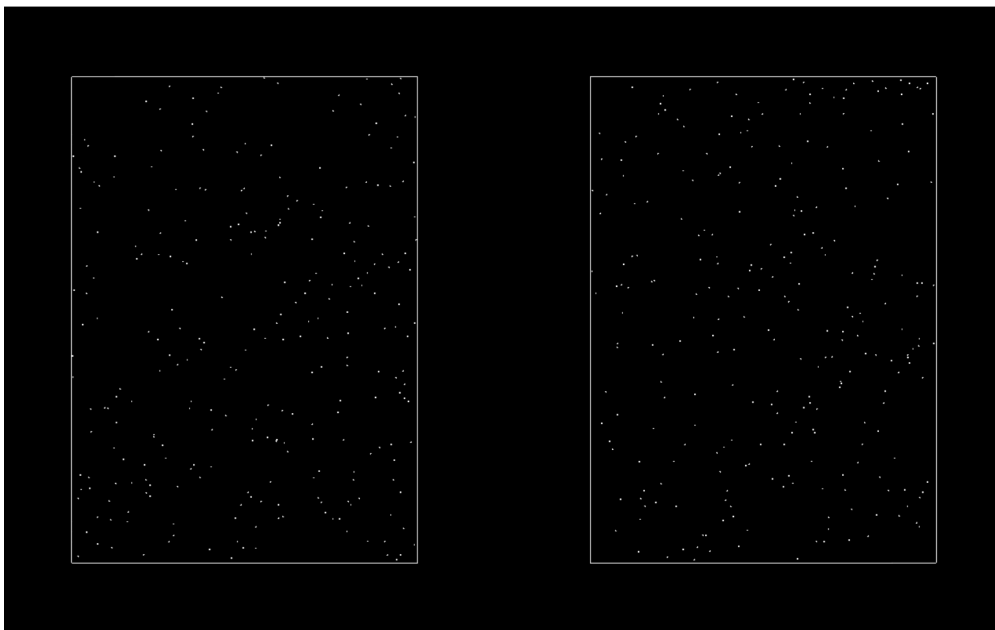


Figure A.2: Motion test at 100% coherency.

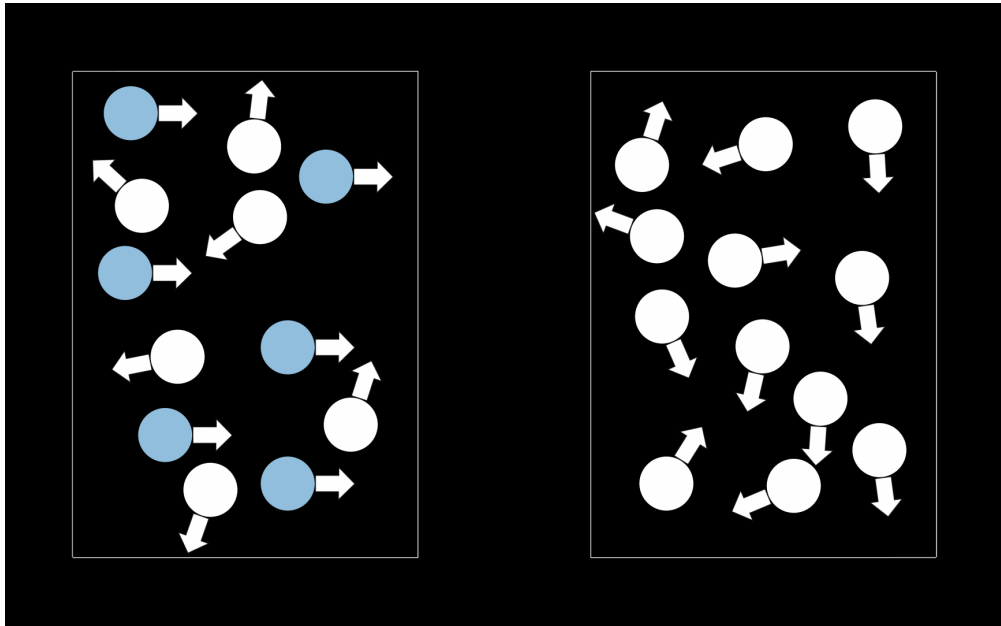


Figure A.3: Illustration of the motion test at 50% coherency.

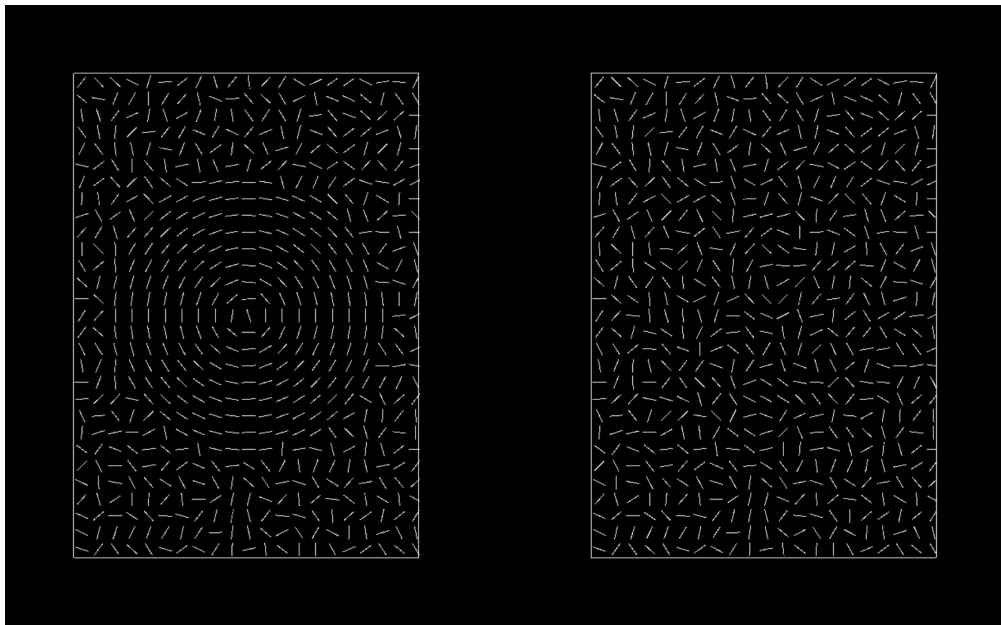


Figure A.4: Form fixed auto test at 100% coherency.

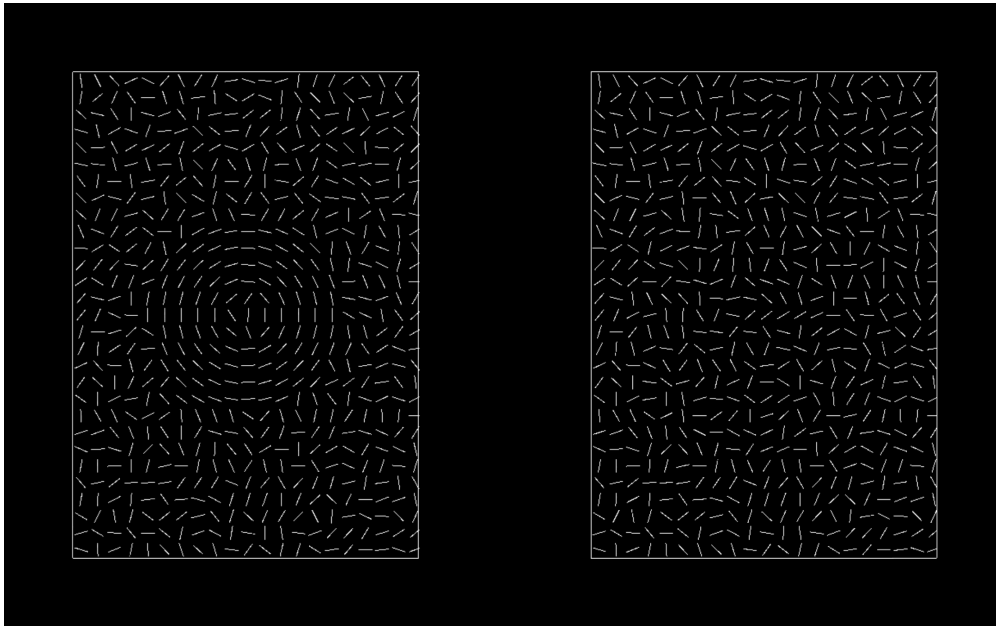


Figure A.5: Form fixed auto test at 50% coherency.

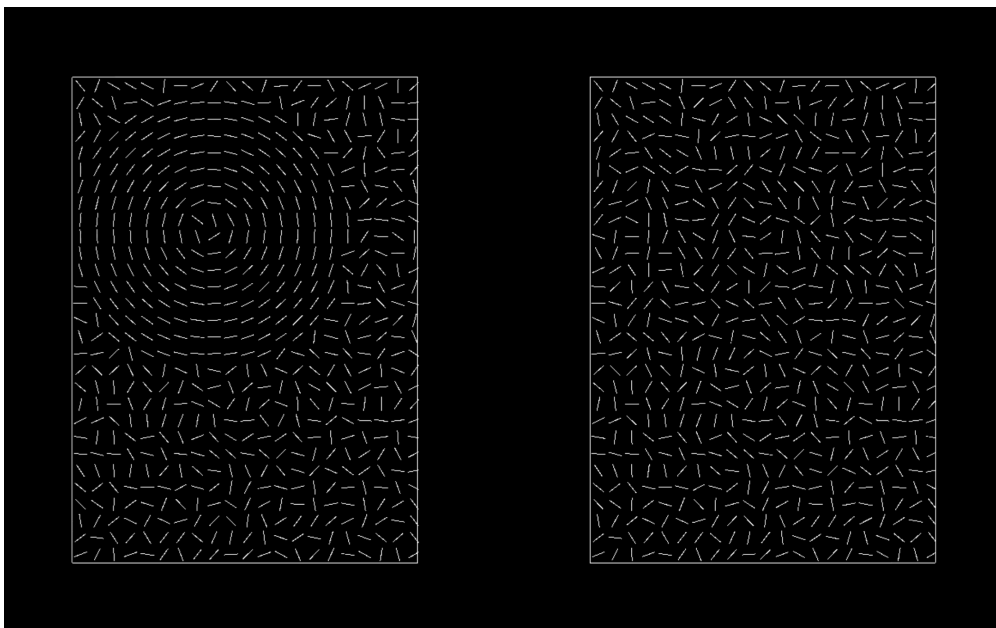


Figure A.6: Form random auto test at 100% coherency.

As mentioned in section 3.5, the form test has two modes called auto and manual. In the manual mode, the line segments are arranged randomly in both patches,

with one side containing line segments in concentric circles. In auto mode on the other hand, the line segments are distributed evenly within the patches and both sides are initially equal before arranging concentric circles in one of the patches. Figure A.7 and A.8 shows a comparison of the auto and manual mode.

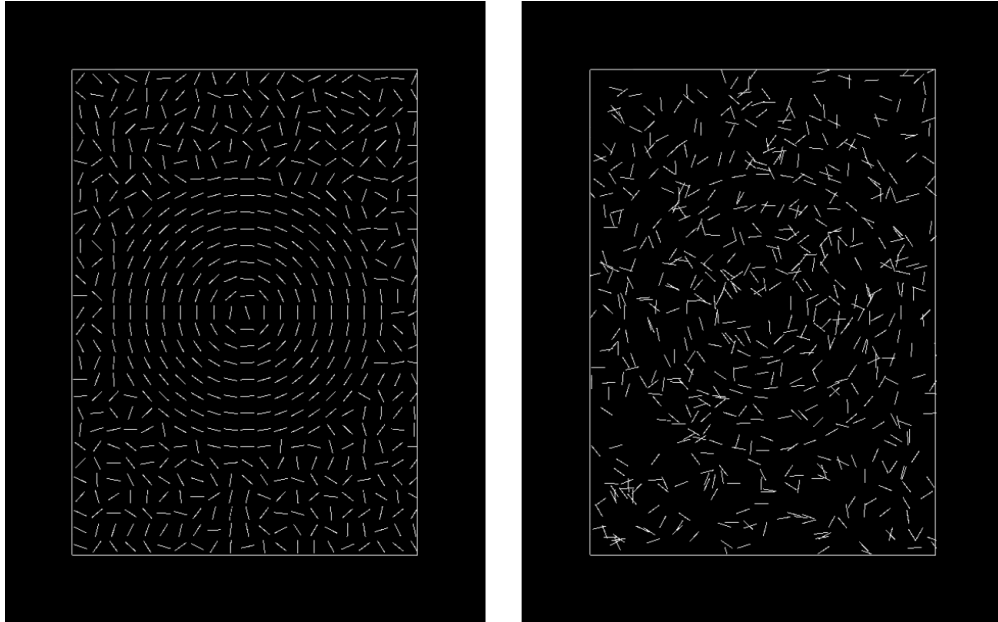


Figure A.7: Comparison between the auto and manual mode for the Form fixed test.

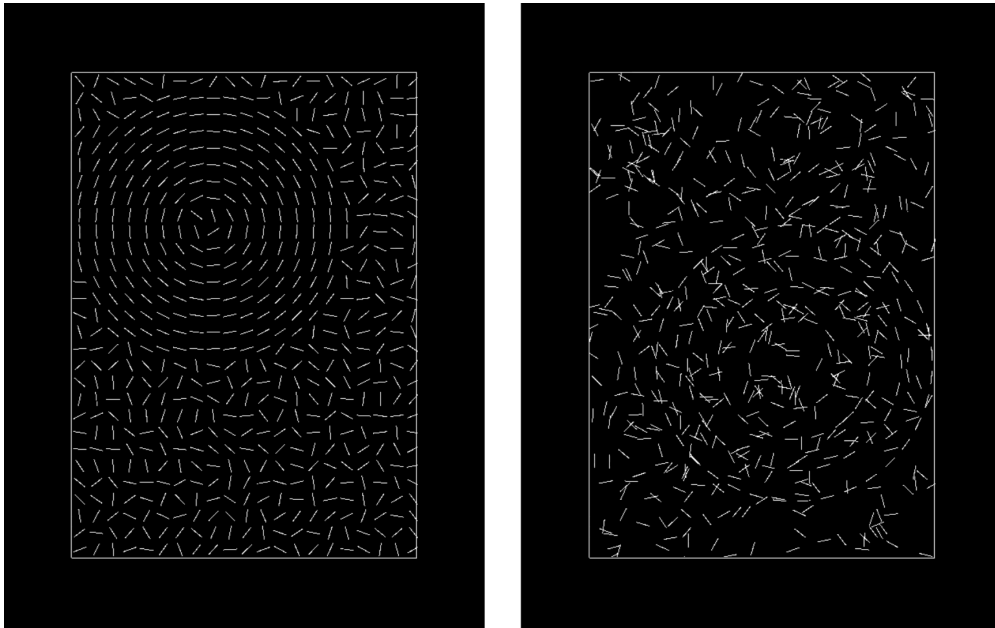


Figure A.8: Comparison between the auto and manual mode for the Form random test.

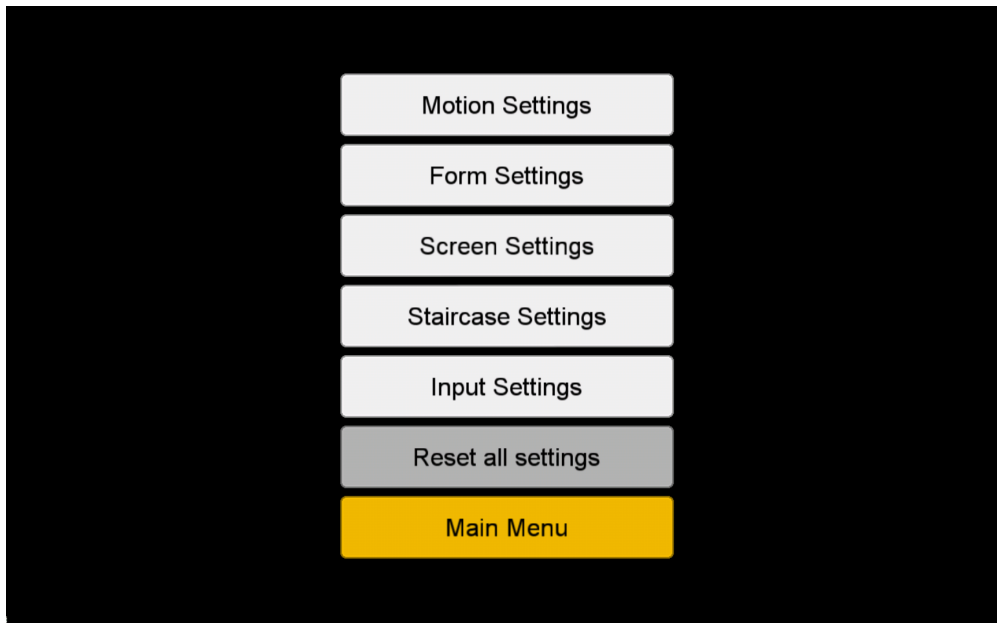
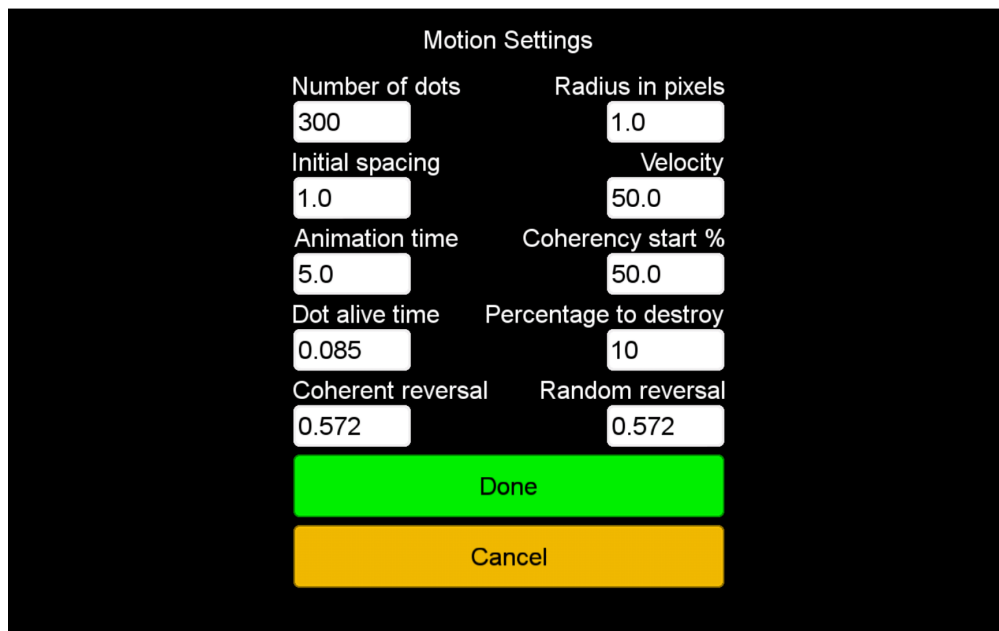


Figure A.9: Settings view.

Figure A.10 shows the motion settings. These settings can be changed to alter the

behaviour of the motion test. There several parameters can be changed, and they can be difficult to understand. These parameters are [4]:

- **Number of dots:** The number of dots to be contained within each patch.
- **Radius:** The radius for the dots in pixels.
- **Initial spacing:** The number of pixels all dots need to be away from each other at the start.
- **Velocity:** The movement speed of the dots in pixels per second.
- **Animation time (seconds):** How long each interval lasts.
- **Coherency start %:** The percentage of dots to move coherently at the first interval.
- **Dot alive time (seconds):** After the set time, the dot is destroyed and relocated within its patch before being redrawn at the next frame update.
- **Percentage to destroy:** The percentage of dots to destroy when dot alive time is reached.
- **Coherent reversal (seconds):** Dots moving coherently will reverse their direction 180 degrees when time is reached.
- **Random reversal (seconds):** Dots moving random will change their direction of travel when time is reached. Direction of travel is changed by a random degree between 0 and 360.



The image shows a 'Motion Settings' dialog box with a black background and white text. It contains several input fields for numerical values, arranged in two columns. At the bottom, there are two buttons: a red 'Done' button and a blue 'Cancel' button.

Parameter	Value
Number of dots	300
Radius in pixels	1.0
Initial spacing	1.0
Velocity	50.0
Animation time	5.0
Coherency start %	50.0
Dot alive time	0.085
Percentage to destroy	10
Coherent reversal	0.572
Random reversal	0.572

Figure A.10: Motion test settings.

Figure A.11 shows the form settings. These settings can alter the behaviour of the form tests. The parameters that can be changed are [4]:

- **Auto mode:** As described earlier, the line segments are evenly distributed across the patches when auto mode is checked. When auto mode is unchecked, the line segments are created randomly within the patches.
- **Number of line segments:** Number of line segments to be contained within each patch.
- **Diameter °:** The diameter of the utmost concentric circle.
- **Number of circles:** Number of concentric circles to be displayed. This setting is disabled when auto mode is checked.
- **Circle gap °:** The diameter difference for each concentric circle. This setting is disabled when auto mode is checked.
- **Line length °:** The length of each line segment.
- **Line height px:** The height of each line segment in pixels.
- **Line gap °:** The distance between each line segment in the concentric circles. This setting is disabled when auto mode is checked.

- **Fixed detection time (seconds):** The length of the interval in the form fixed test.
- **Random detection time (seconds):** The length of the interval in the form random test.

Form Settings

Auto mode

Number of lines Diameter°

Nr of circles Circle gap°

Line length° Line height px ~0.0227°

Line gap°

Initial coherency

Fixed detection time Random detection time

Figure A.11: Form test settings.

Figure A.12 shows the screen settings. These settings are related to the screen size and resolution. The width and height measurements are fetched automatically when the application starts, but they are possible to edit because screen panels can be larger than the viewable area. In the background you can see two patches that shows the area where test elements will display. The parameters that can be changed are [4]:

- **Width in mm:** The width of the viewable screen area in millimetres.
- **Height in mm:** The height of the viewable screen area in millimetres.
- **Width in pixels:** The width wise pixel count of the screen.
- **Height in pixels:** The height wise pixel count of the screen.
- **Distance in mm:** The viewing distance of the screen in millimetres measured from the eyes of the viewer to the centre of the screen.
- **Patch width °:** The width of each patch.

- **Patch height °**: The height of each patch.
- **Patch gap °**: The gap between each patch.

Screen settings

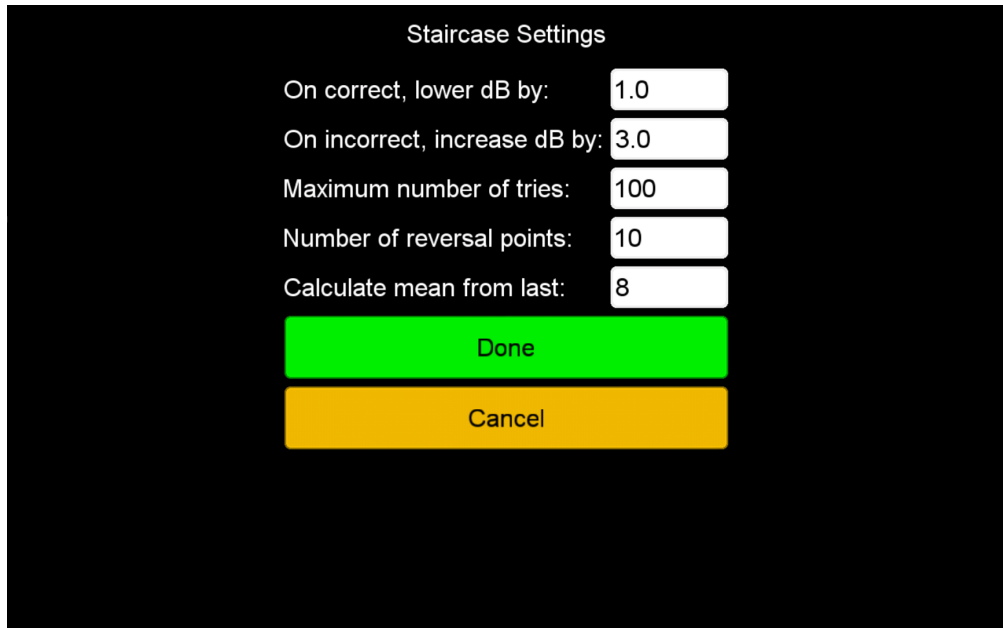
Width in mm	Height in mm
<input type="text" value="152"/>	<input type="text" value="95"/>
Width in pixels	Height in pixels
<input type="text" value="1280"/>	<input type="text" value="800"/>
Distance in mm	
<input type="text" value="300"/>	
Patch width in °	Patch height in °
<input type="text" value="10"/>	<input type="text" value="14"/>
Patch gap in °	
<input type="text" value="5"/>	
<input type="button" value="Done"/>	
<input type="button" value="Cancel"/>	

Figure A.12: Screen settings.

Figure A.13 shows the staircase settings that are related to recording tests results. The parameters that can be changed are [4]:

- **On correct lower dB by:** Uses the amplitude ratio¹ of the decibel as a factor to lower coherency on correct answer. 1 dB is equal to a 1.22% change in coherency.
- **On wrong increase dB by:** Uses the amplitude ration of the decibel as a factor to increase the coherency on wrong answer.
- **Maximum number of tries:** Maximum number of tries for a given test.
- **Calculate from last:** How many reversal points to use when calculating the geometric mean. The geometric mean is calculated from the X number of last reversal points, and is shown as the threshold score in the test results screen.

¹Amplitude ratio = $10^{(decibel/20)}$



The image shows a 'Staircase Settings' dialog box with a black background. It contains five input fields with white text and values: 'On correct, lower dB by: 1.0', 'On incorrect, increase dB by: 3.0', 'Maximum number of tries: 100', 'Number of reversal points: 10', and 'Calculate mean from last: 8'. Below the input fields are two buttons: a green 'Done' button and a yellow 'Cancel' button.

Setting	Value
On correct, lower dB by:	1.0
On incorrect, increase dB by:	3.0
Maximum number of tries:	100
Number of reversal points:	10
Calculate mean from last:	8

Figure A.13: Staircase settings.

The input settings are showed in figure A.14. These settings are related to user input when interacting with the application. Parameters that can be changed are:

- **Choose left patch key:** Key to choose the left patch with.
- **Choose right patch key:** Key to choose the right patch with.
- **Back key:** Key to go backwards in the application with.
- **Continuous mode:** When checked, it is possible to choose a patch before the animation time runs out [4].

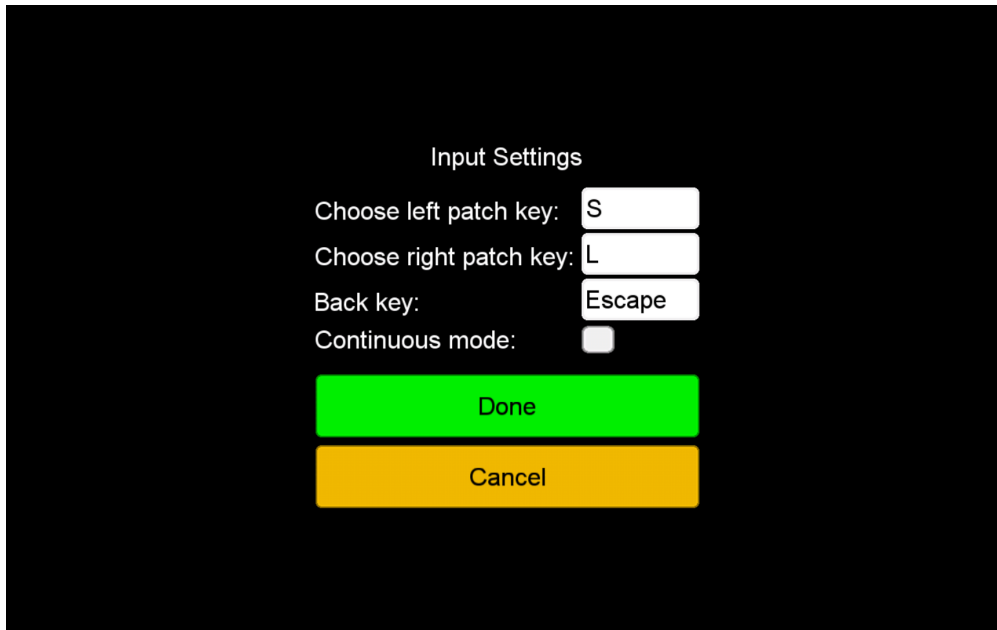


Figure A.14: Input settings.

When changing the key for an input option, a dialog window opens. This is showed in figure A.15.



Figure A.15: Input settings key dialog.

In reset settings, a dialog window, where the user has to confirm or cancel the action, opens to prevent accidental resetting of settings. This can be seen in figure A.16.

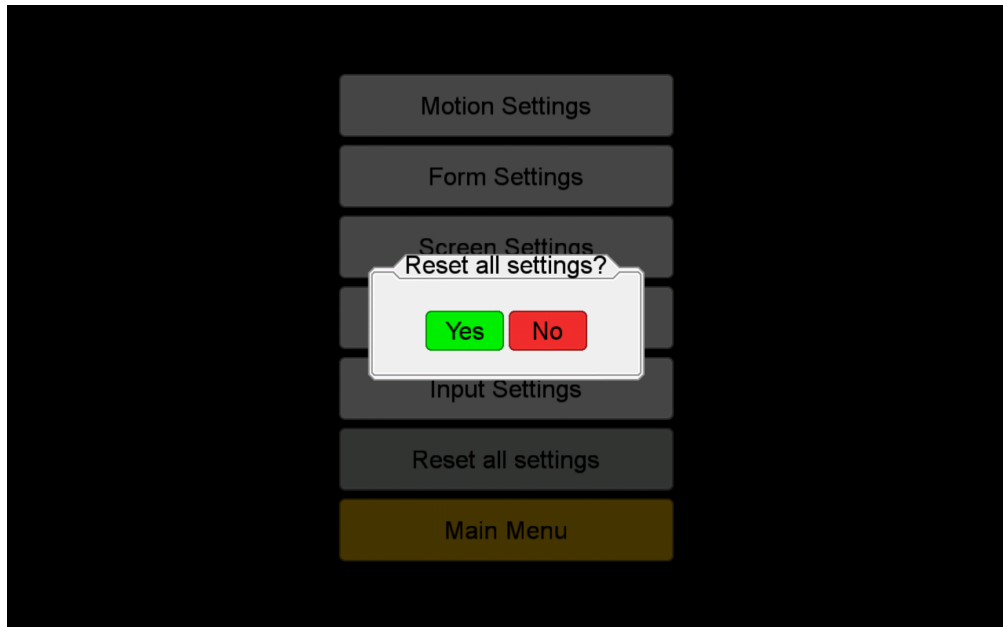


Figure A.16: Reset settings confirmation dialog.

Appendix B

Contract

Permission to use test results and film

I permit the use of my test results in the indicated project- and master's thesis at NTNU, during the academic year of 2016-2017.

I permit being filmed during the test, for later analysis.

I am informed that I will be kept anonymous in all published works.

Signature

Place and Date

Appendix C

Questionnaire

Questions to test subject

- What do you think about the application?
- What do you think about being tested?
- What were you thinking during the actual testing?
- What do you think about the design of the application?
- On a scale from 1 - 5, where 1 is difficult and 5 is easy, how was it to understand what to do in the application?

1 - 2 - 3 - 4 - 5

- Were there any things in the application that were particularly difficult to understand?

- Were there any things in the application that were particularly easy to understand?

- Were there any things in the application that were easier to understand with the help of a test supervisor?

Appendix D

System Usability Scale

1. I think I would like to use this system frequently.

Strongly disagree 1 2 3 4 5 Strongly agree

2. I found the system unnecessarily complex.

Strongly disagree 1 2 3 4 5 Strongly agree

3. I thought the system was easy to use.

Strongly disagree 1 2 3 4 5 Strongly agree

4. I think that I would need the support of a technical person to be able to use this system.

Strongly disagree 1 2 3 4 5 Strongly agree

5. I found the various functions in this system were well integrated.

Strongly disagree 1 2 3 4 5 Strongly agree

6. I thought there was too much inconsistency in this system.

Strongly disagree 1 2 3 4 5 Strongly agree

7. I would imagine that most people would learn to use this system very quickly.

Strongly disagree 1 2 3 4 5 Strongly agree

8. I found the system very cumbersome to use.

Strongly disagree 1 2 3 4 5 Strongly agree

9. I felt very confident using the system.

Strongly disagree 1 2 3 4 5 Strongly agree

10. I needed to learn a lot of things before I could get going with this system.

Strongly disagree 1 2 3 4 5 Strongly agree

Appendix E

Paper Prototypes

E.1 Paper Prototype 1

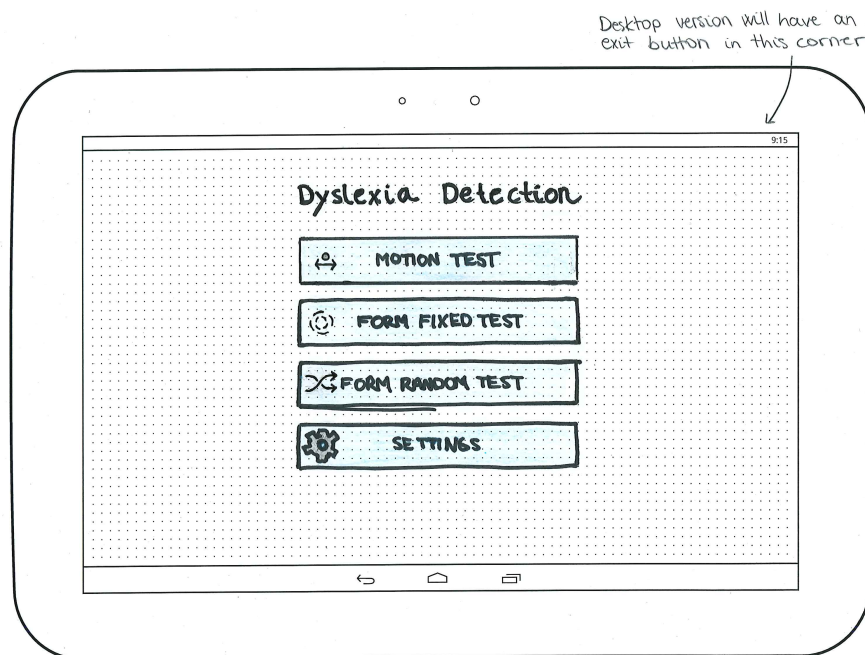


Figure E.1: The main view.

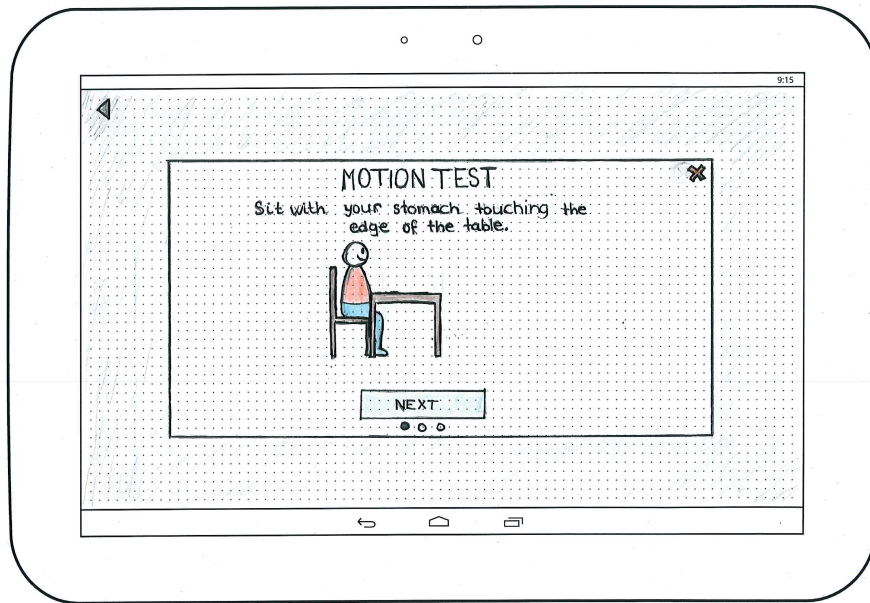


Figure E.2: The first tutorial view after clicking on motion test.

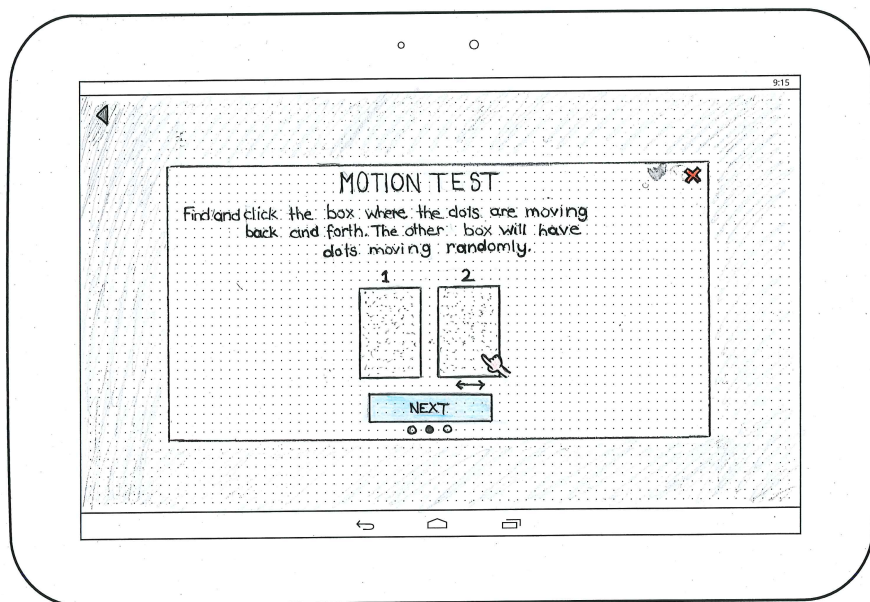


Figure E.3: The second tutorial view.

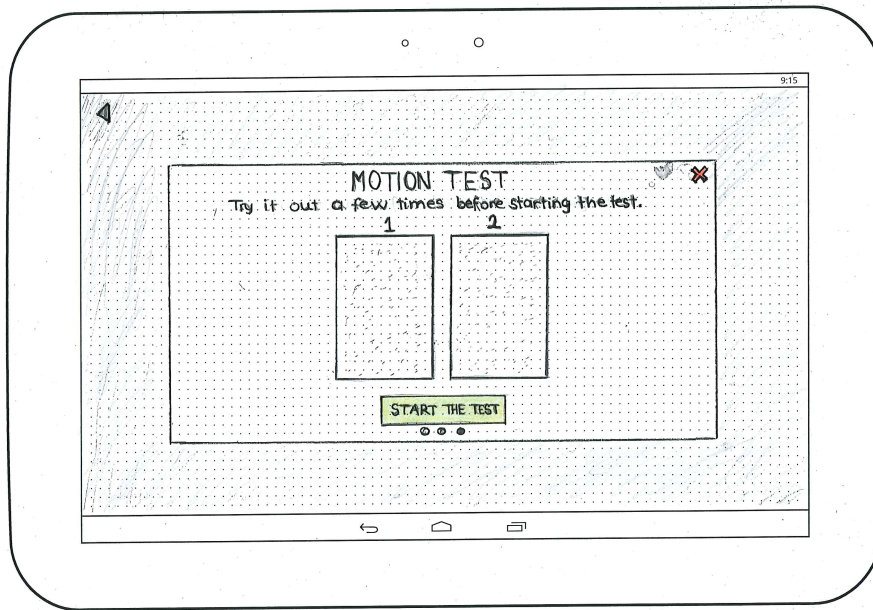


Figure E.4: The last tutorial view.

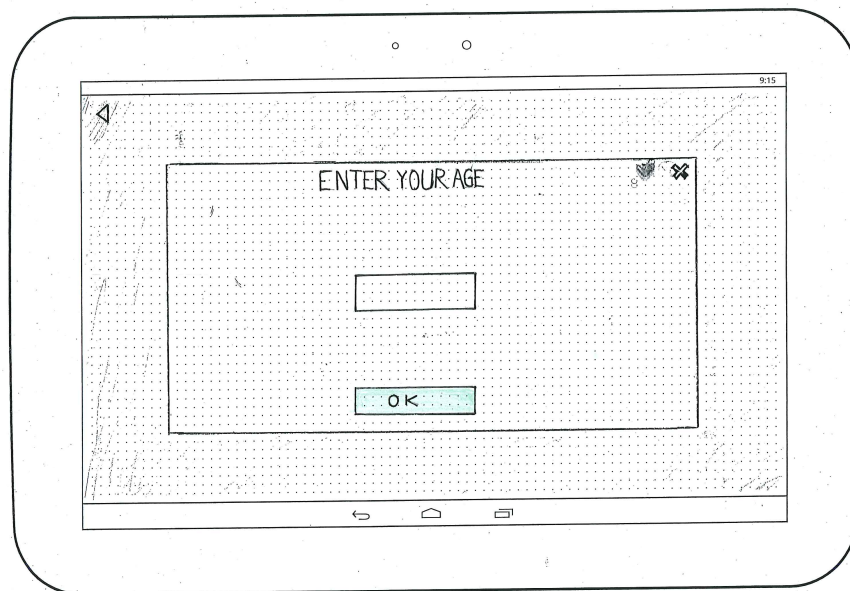


Figure E.5: The age view.

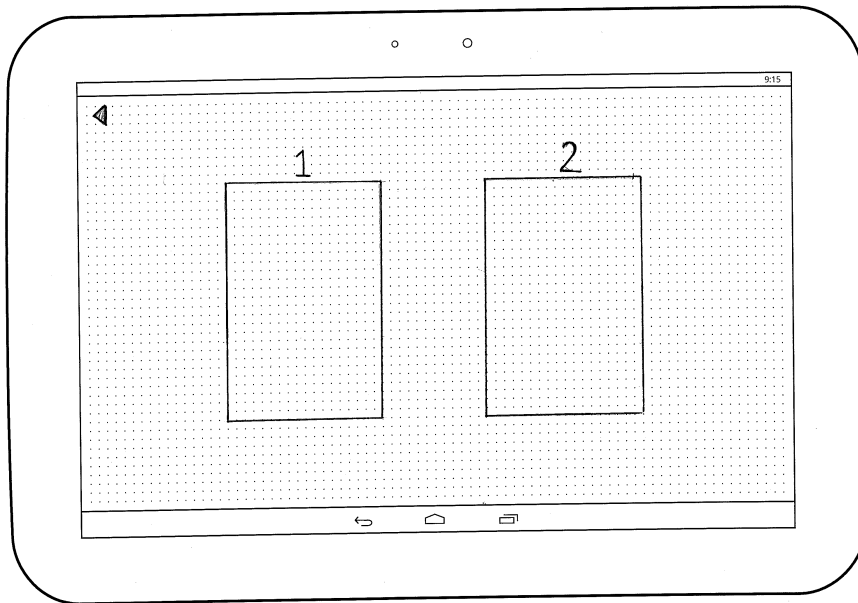


Figure E.6: The test view (without dots and lines).

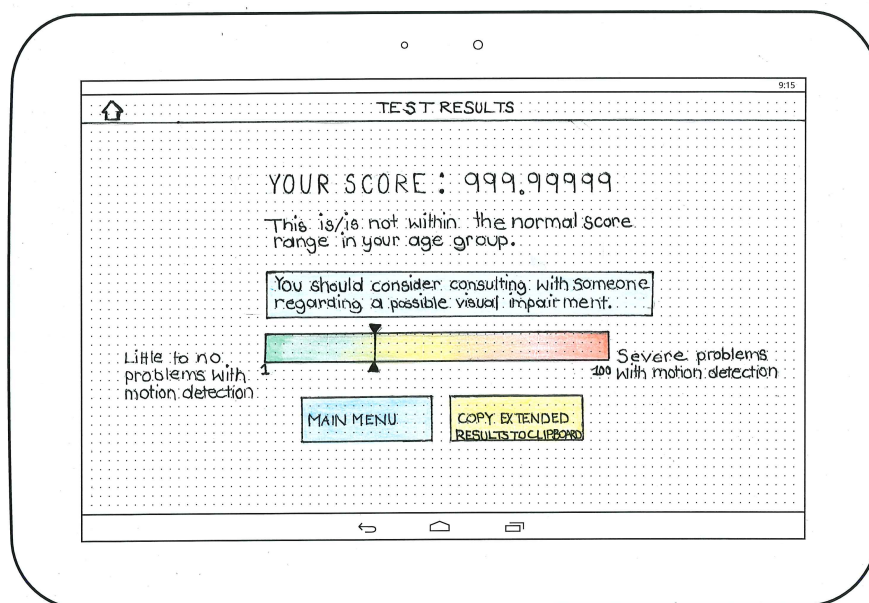


Figure E.7: The score view.

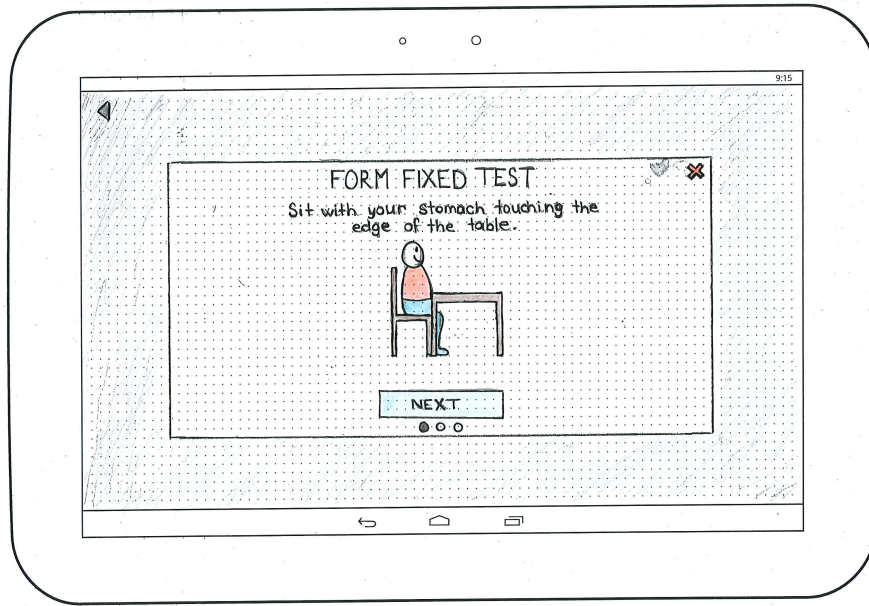


Figure E.8: The first tutorial view after clicking on form fixed test.

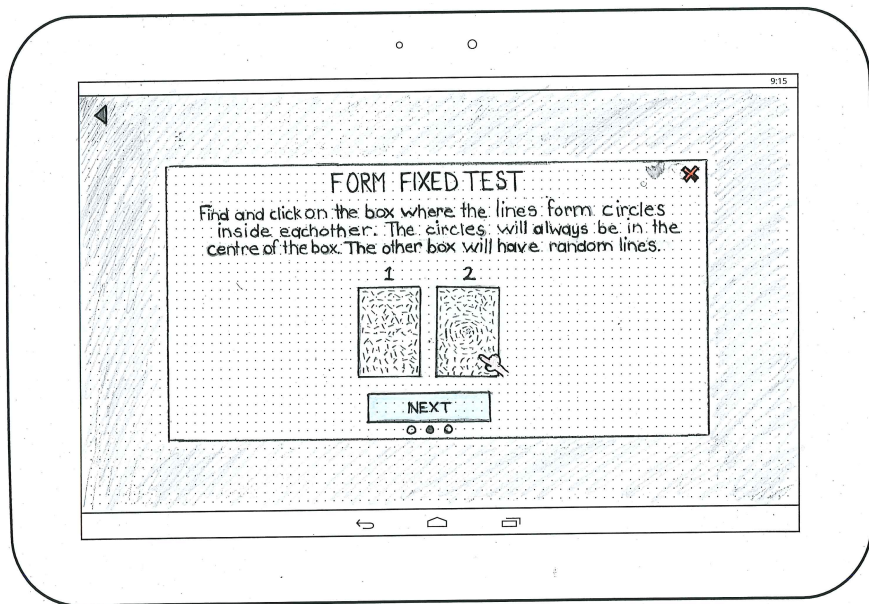


Figure E.9: The second tutorial view.

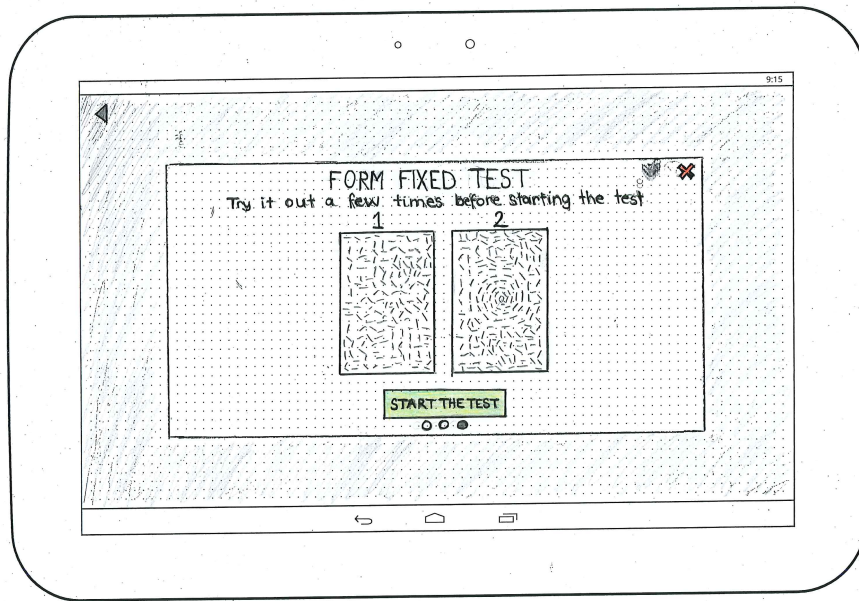


Figure E.10: The last tutorial view.

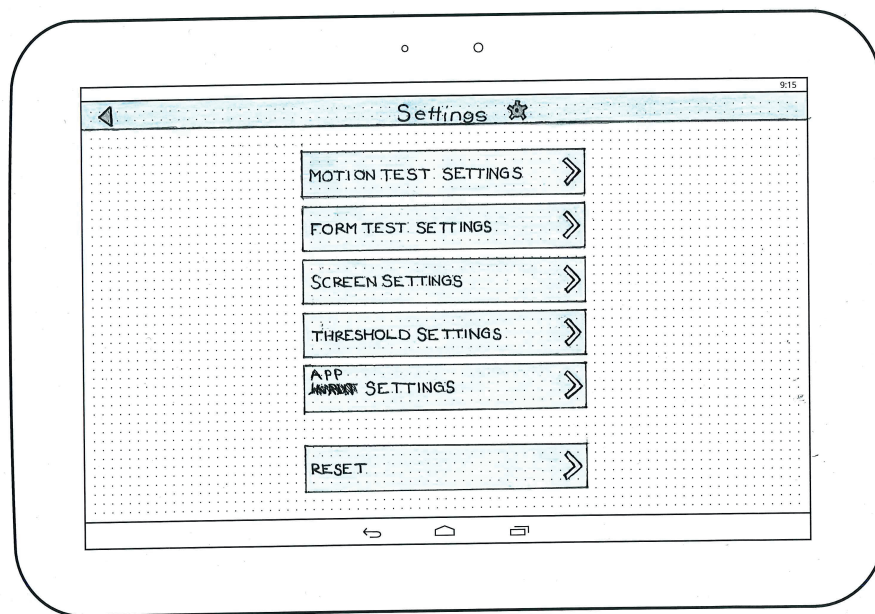


Figure E.11: The settings view.

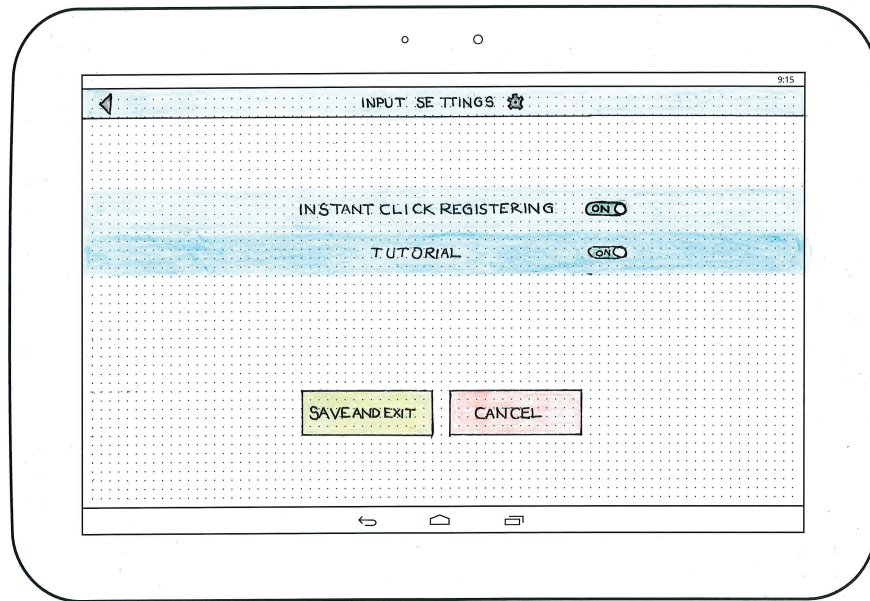


Figure E.12: The app settings view.

E.2 Paper Prototype 2

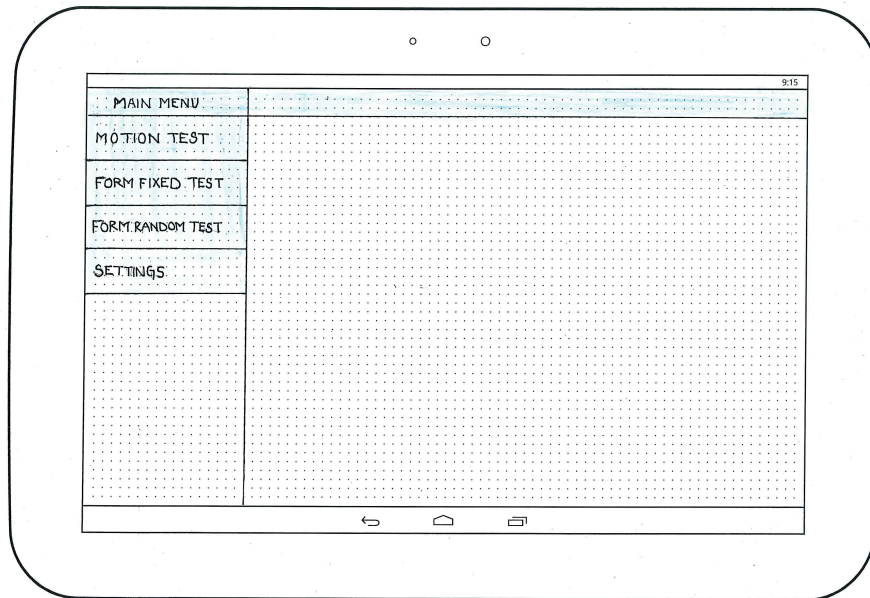


Figure E.13: The main view.

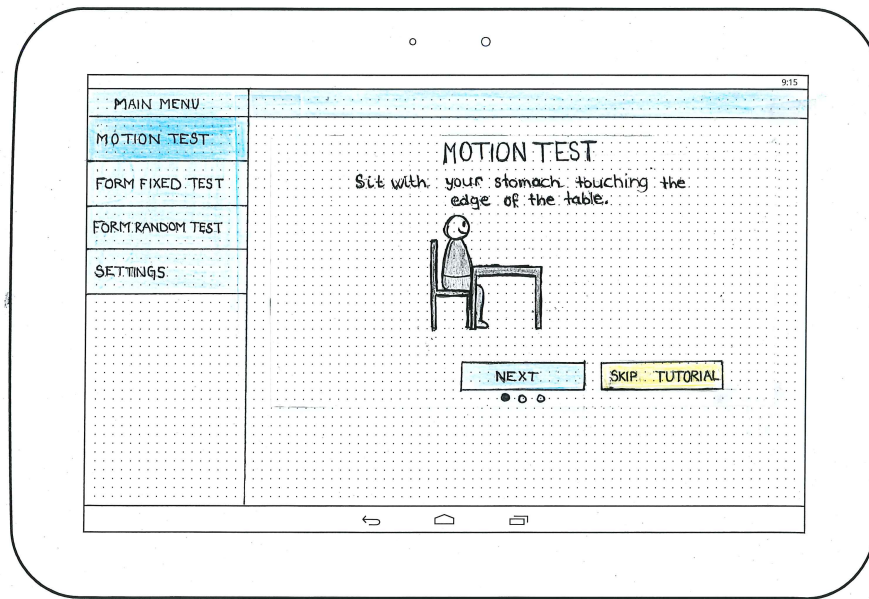


Figure E.14: The first motion test tutorial view.

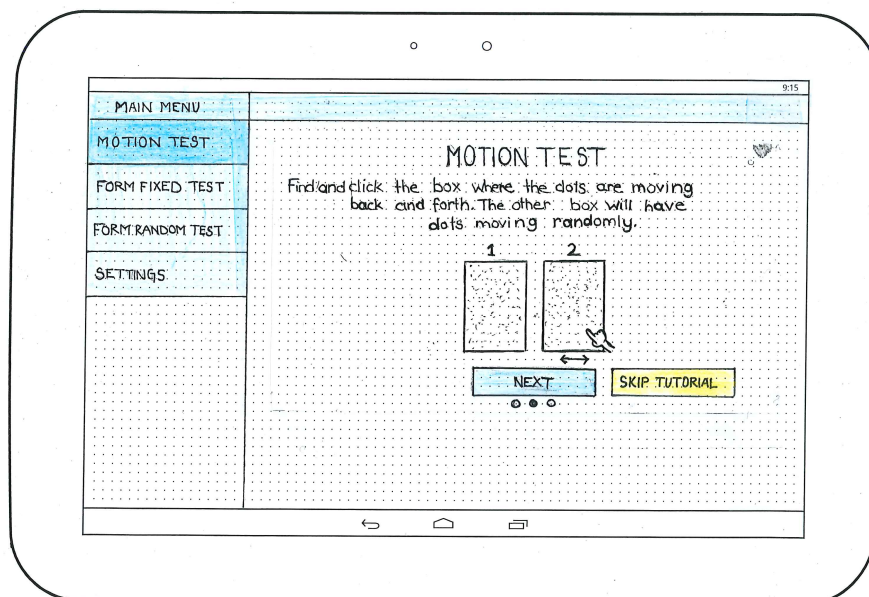


Figure E.15: The second motion test tutorial view.

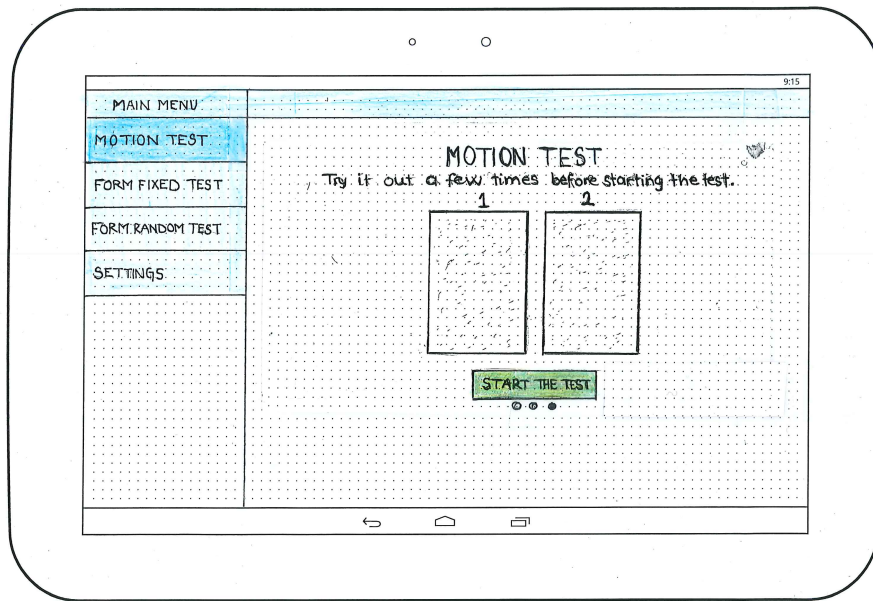


Figure E.16: The third motion test tutorial view.

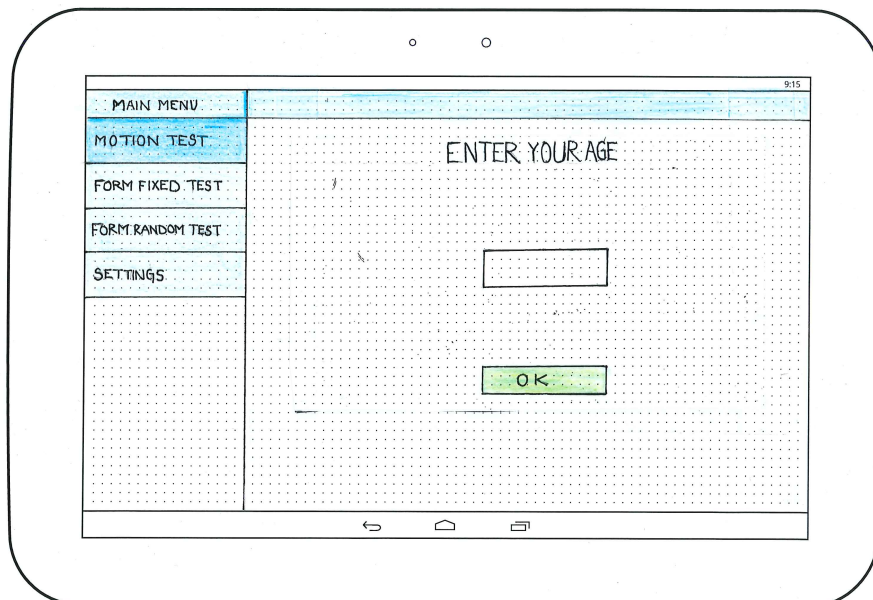


Figure E.17: The age view.

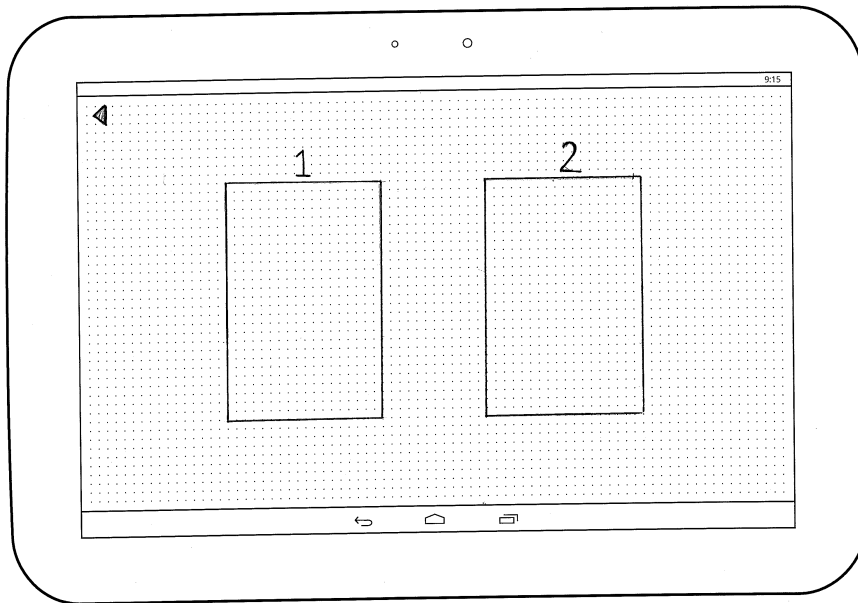


Figure E.18: The test view.

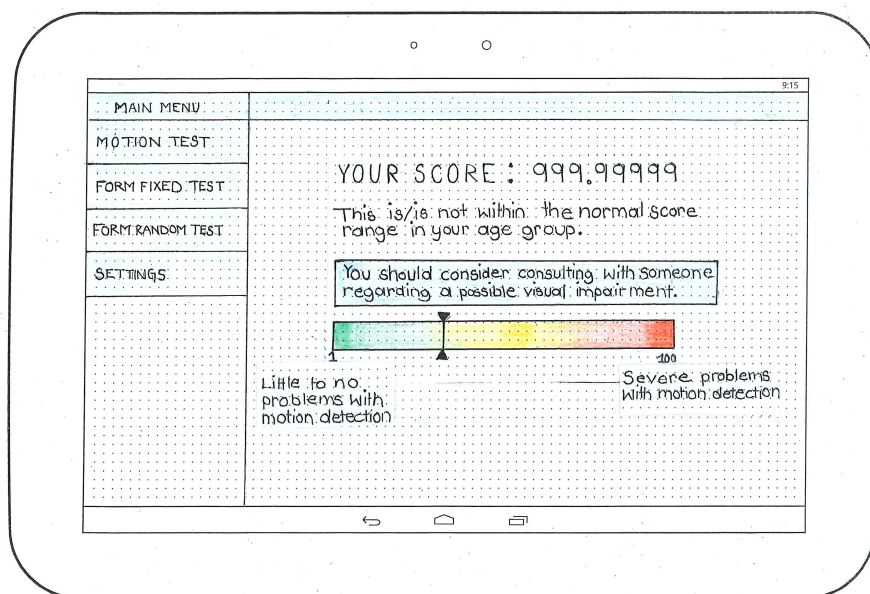


Figure E.19: The score view.

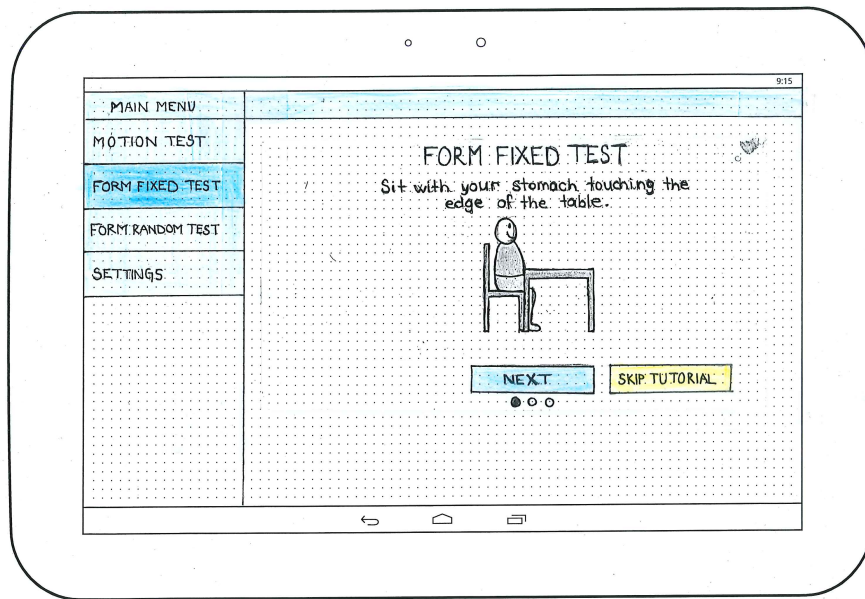


Figure E.20: The first form fixed test tutorial view.

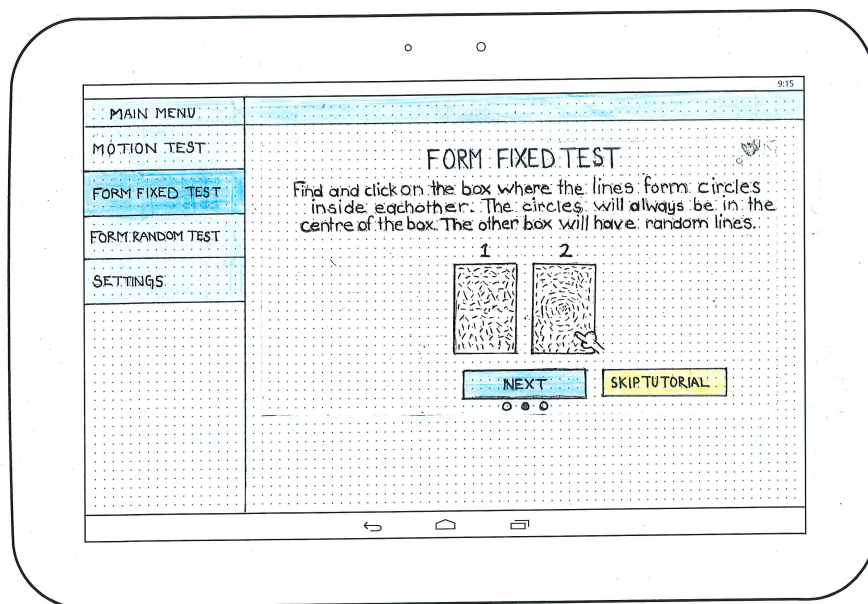


Figure E.21: The second form fixed test tutorial view.

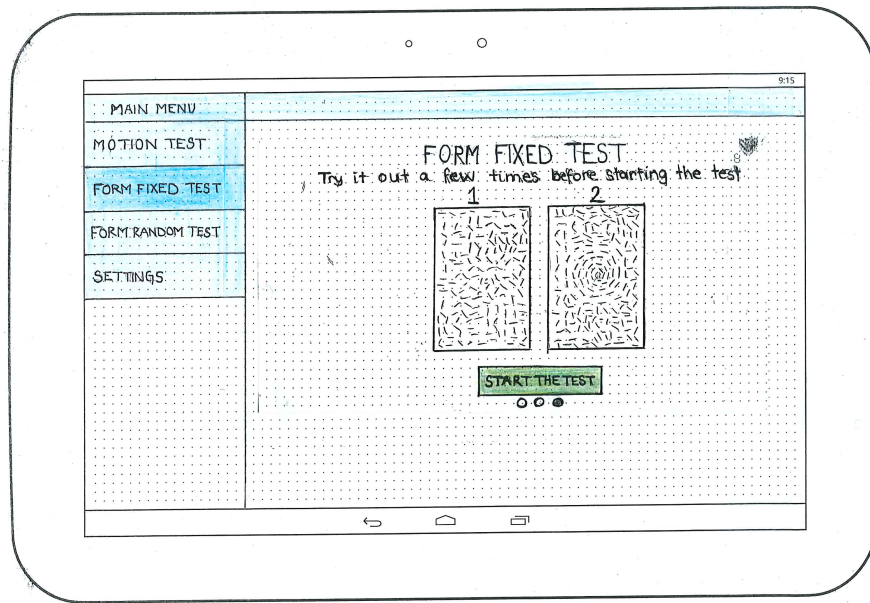


Figure E.22: The third form fixed test tutorial view.

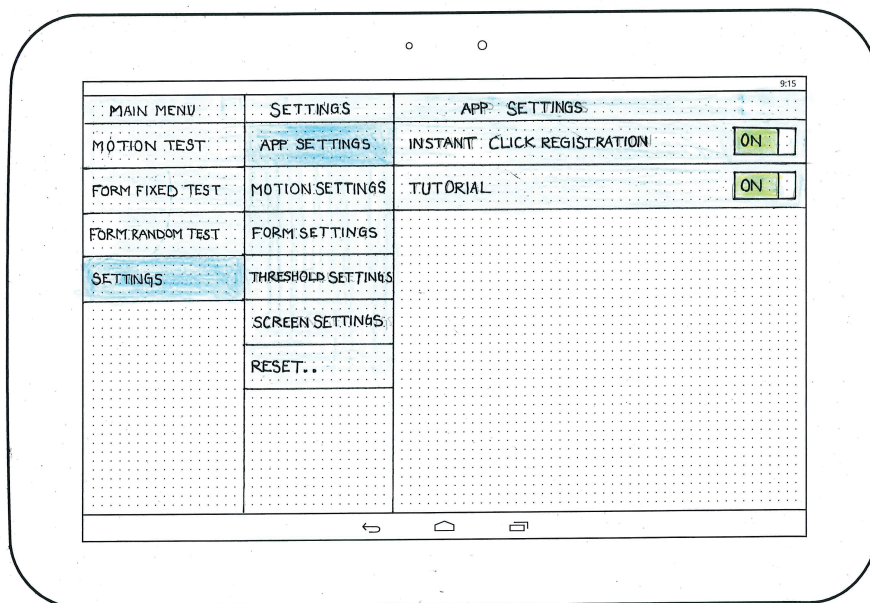


Figure E.23: The settings view.

Appendix F

Digital Prototype

F.1 Final Digital Prototype

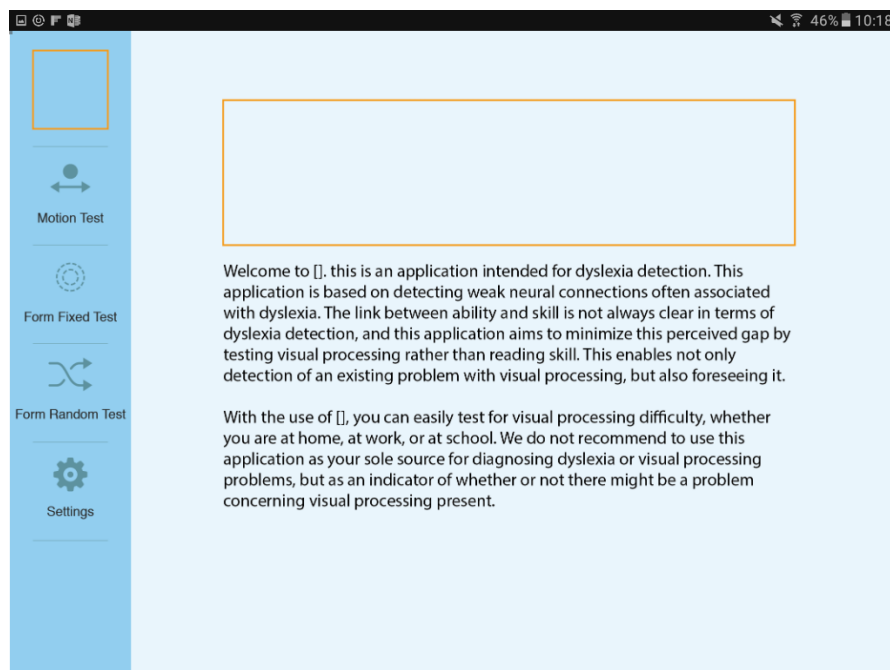


Figure F.1: The main view.

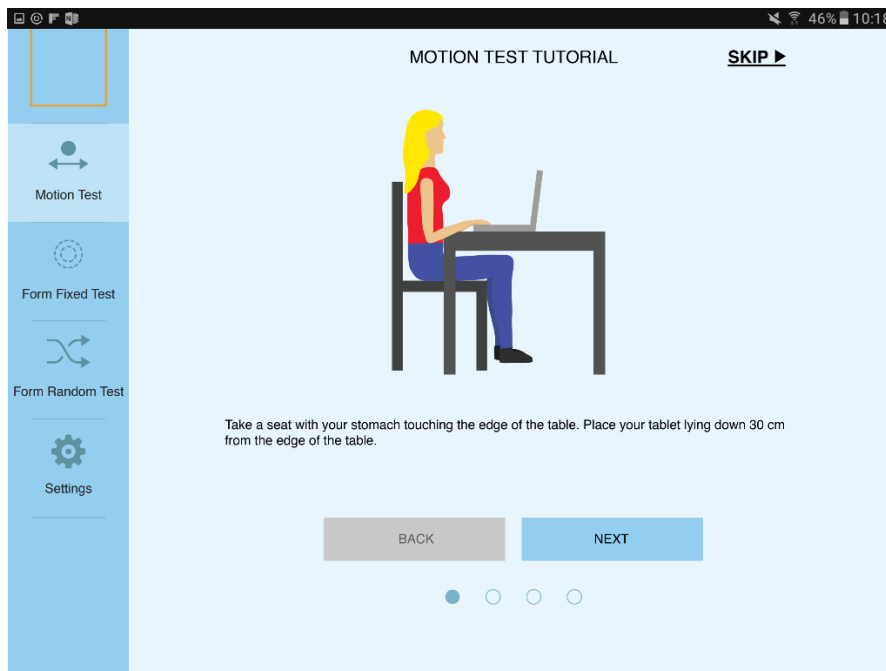


Figure F.2: The first tutorial view of the motion test.

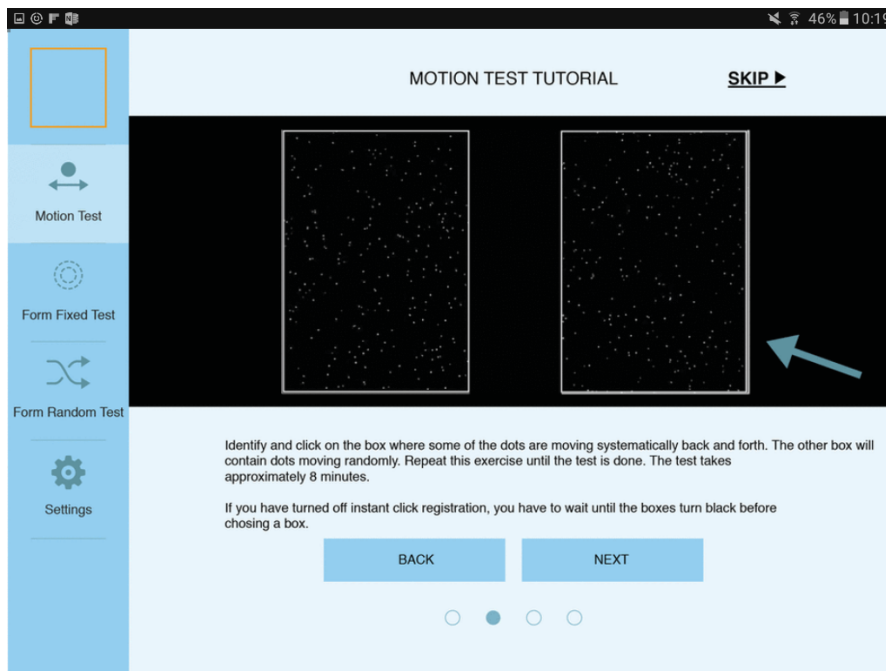


Figure F.3: The second tutorial view of the motion test.

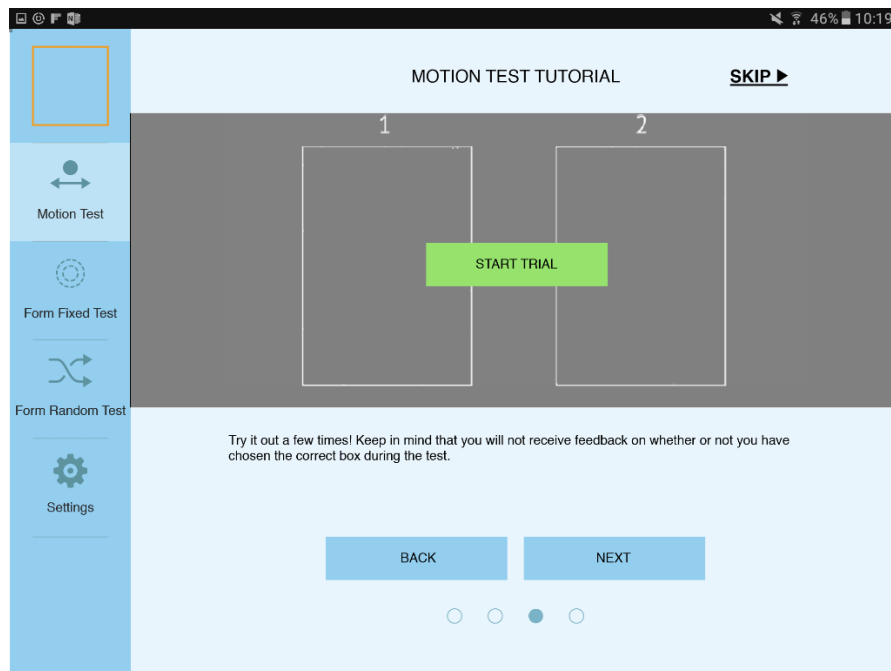


Figure F.4: The third tutorial view of the motion test - before starting the trial.

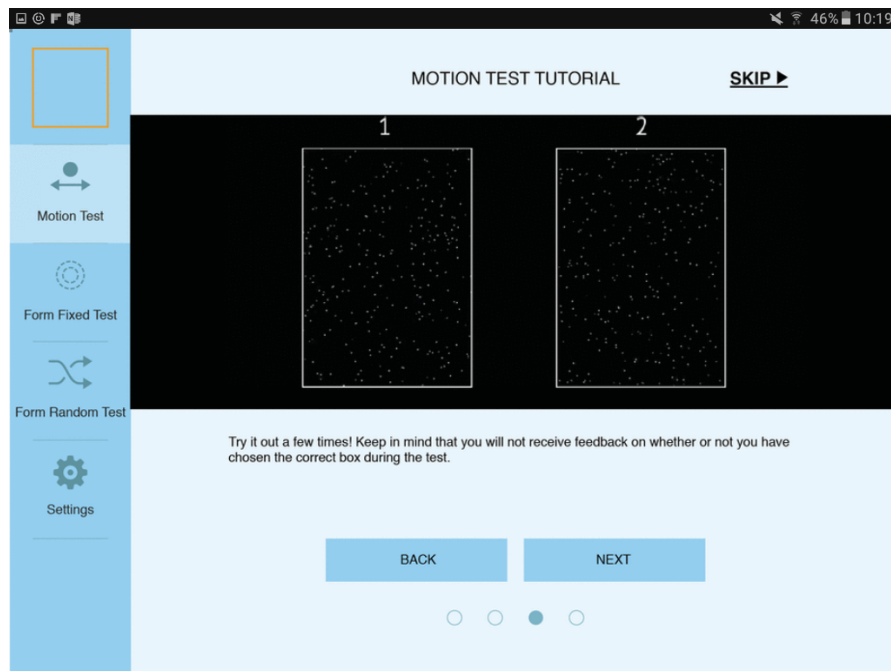


Figure F.5: The third tutorial view of the motion test - when first starting the trial.

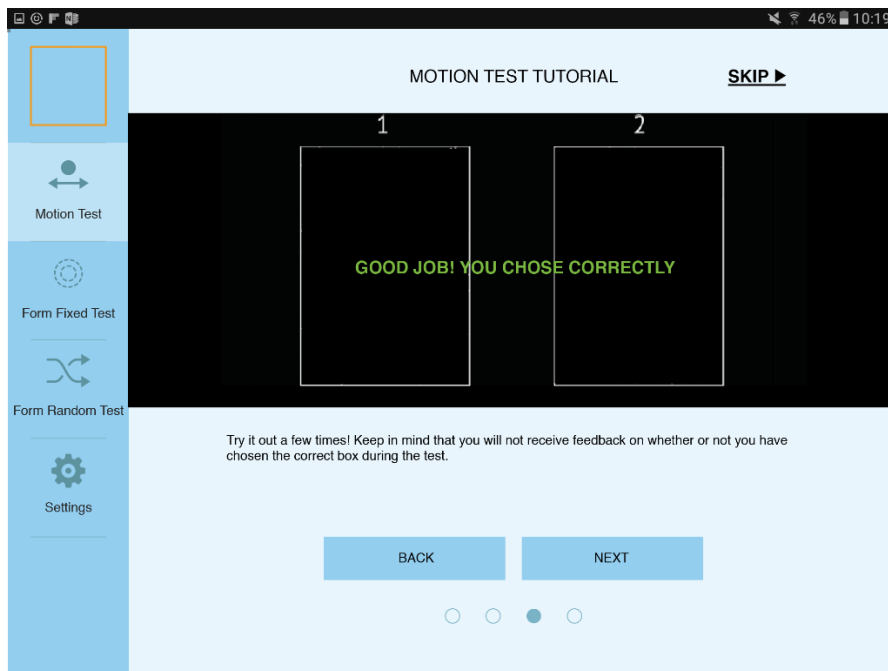


Figure F.6: The third tutorial view of the motion test - correct answer.

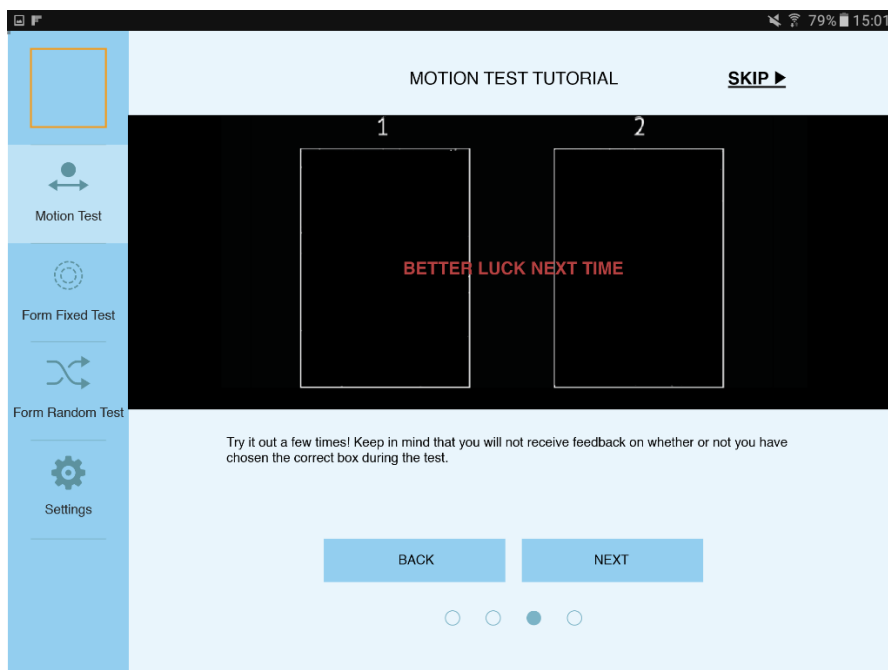


Figure F.7: The third tutorial view of the motion test - wrong answer.

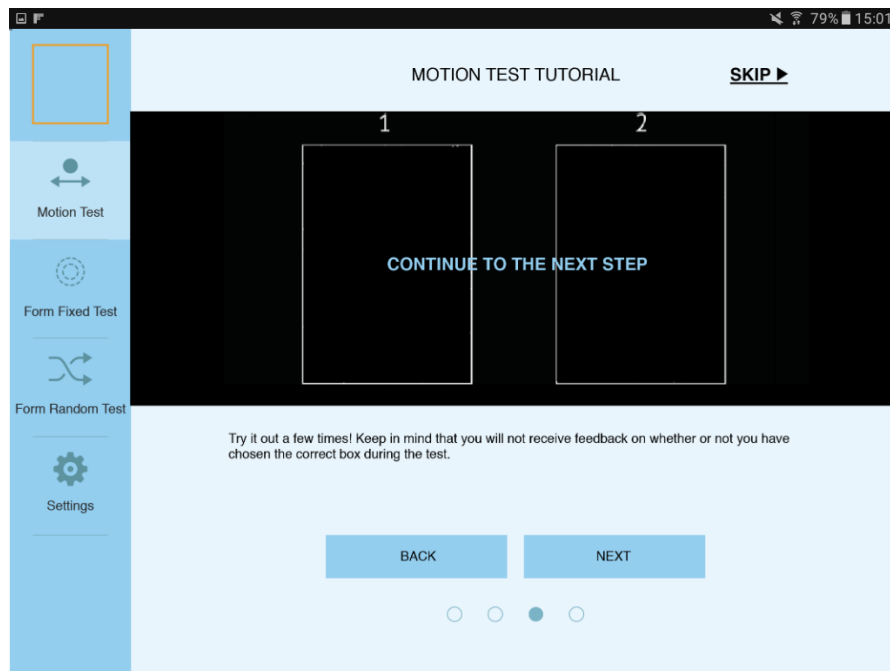


Figure F.8: The third tutorial view of the motion test - completed trial.

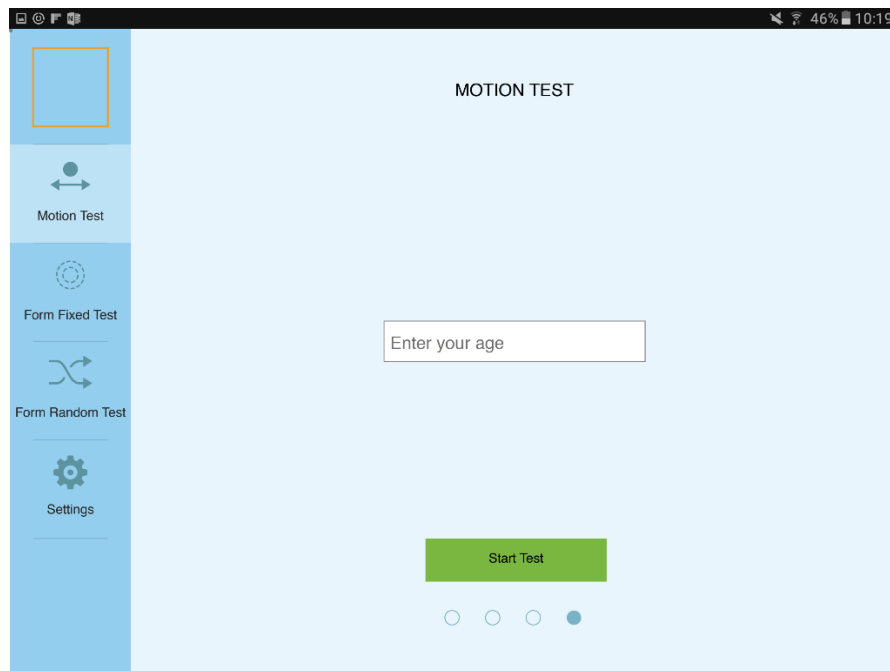


Figure F.9: The fourth tutorial view of the motion test.

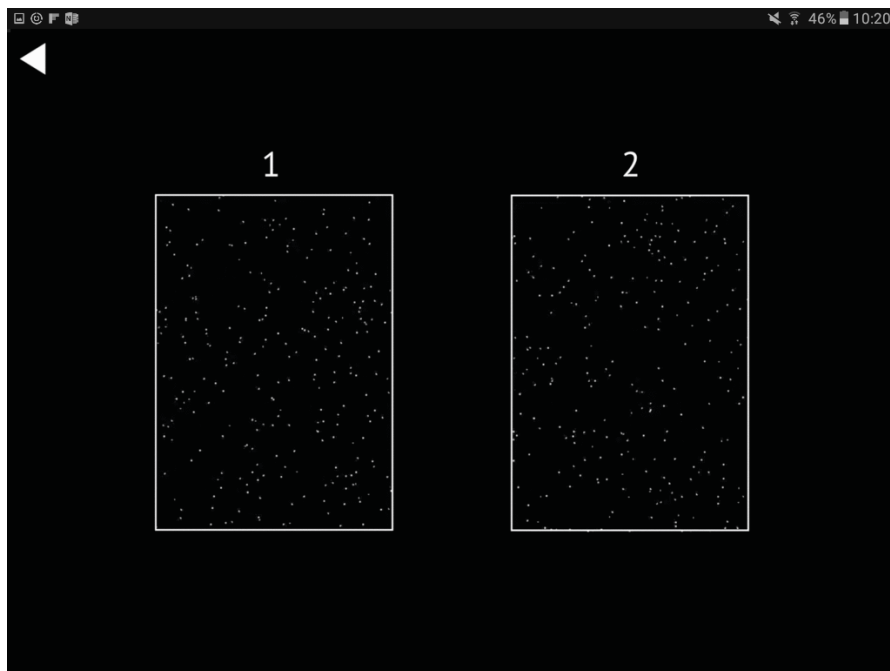


Figure F.10: The motion test.

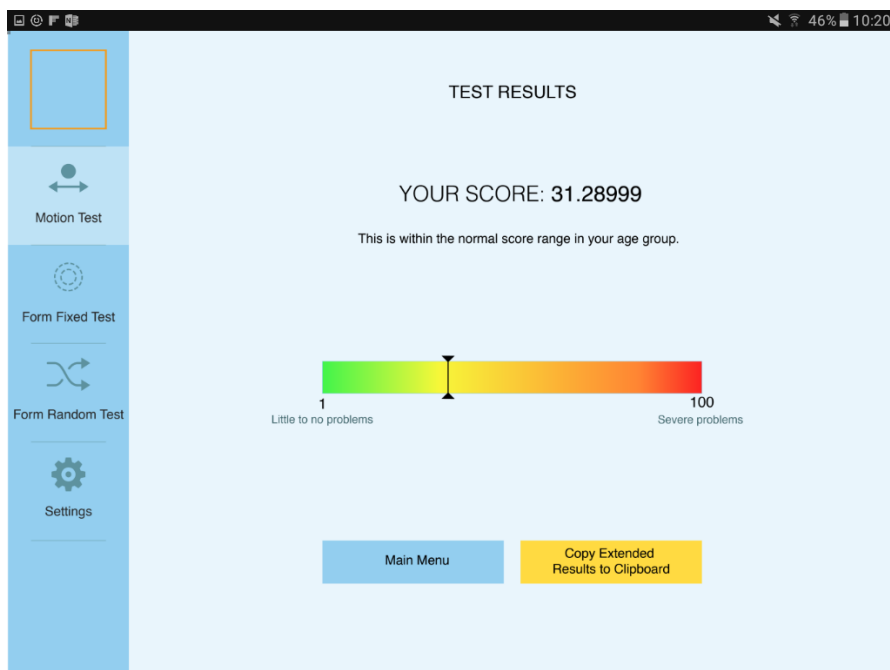


Figure F.11: The result view.

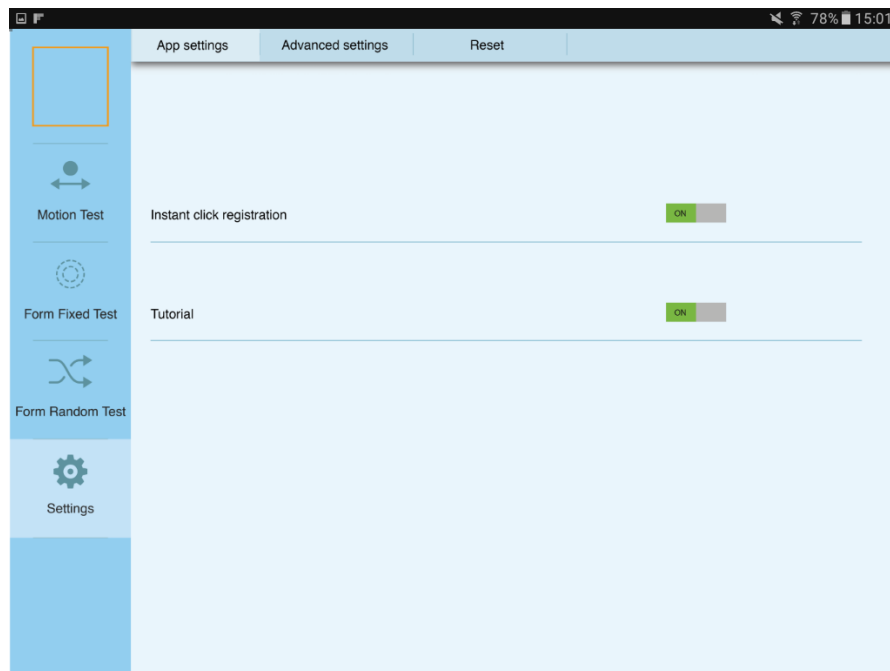


Figure F.12: The settings view.

Appendix G

Finished Application

In this chapter, the final application is presented.

G.1 Home Screen



Figure G.1: Home screen.

G.2 Tutorial

G.2.1 Step 1

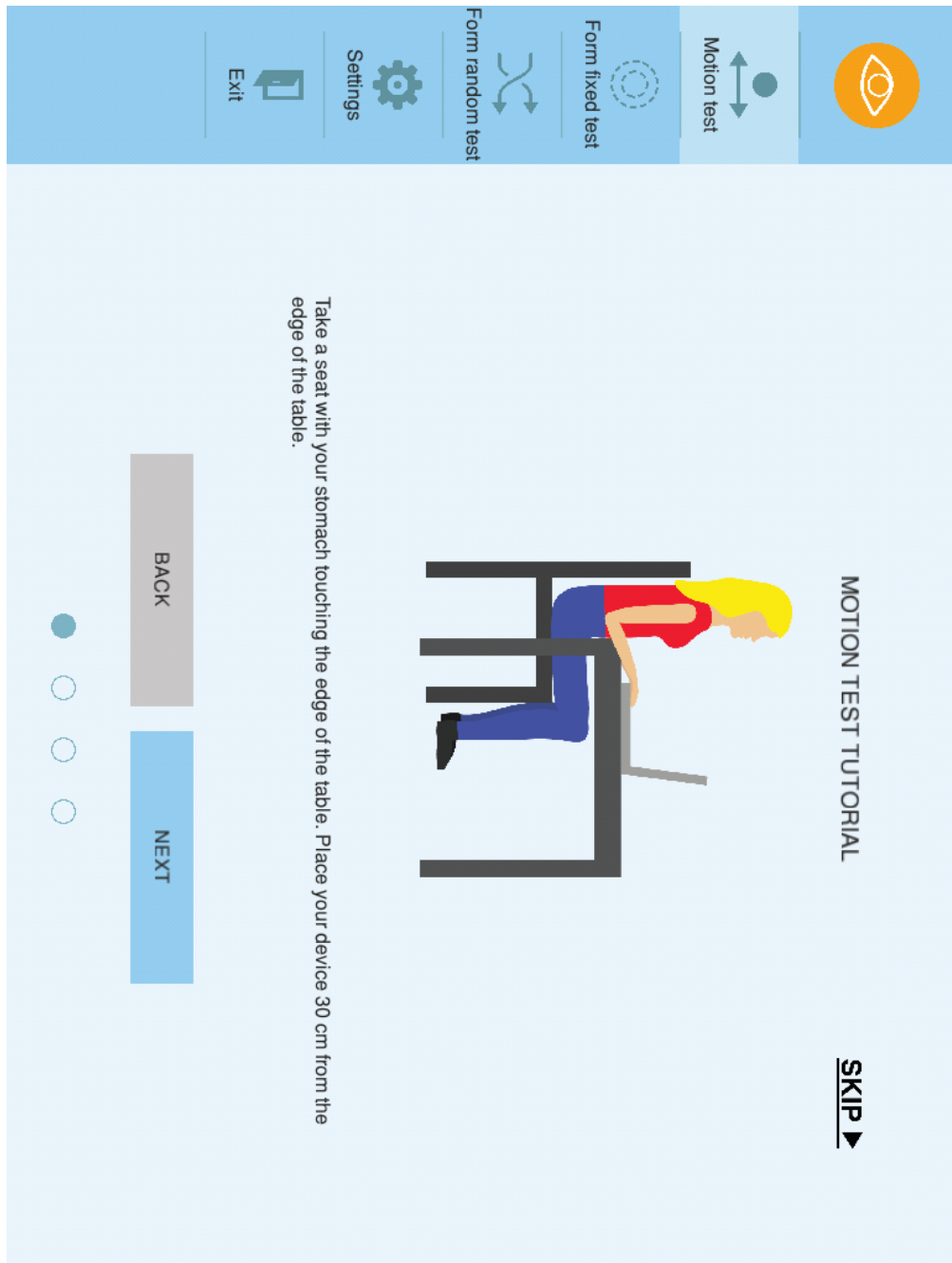


Figure G.2: Motion test tutorial - step 1.

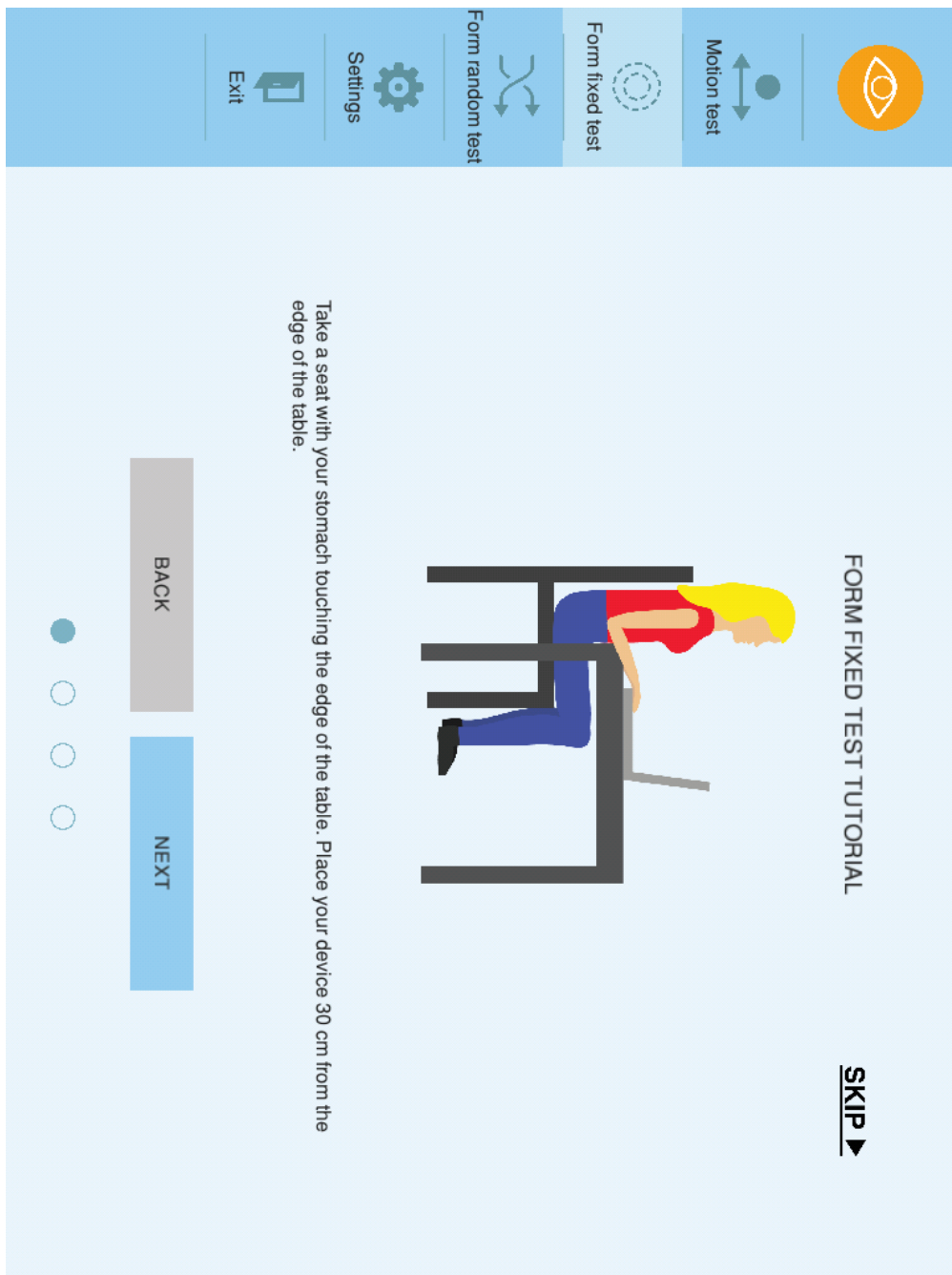


Figure G.3: Form fixed test tutorial - step 1.

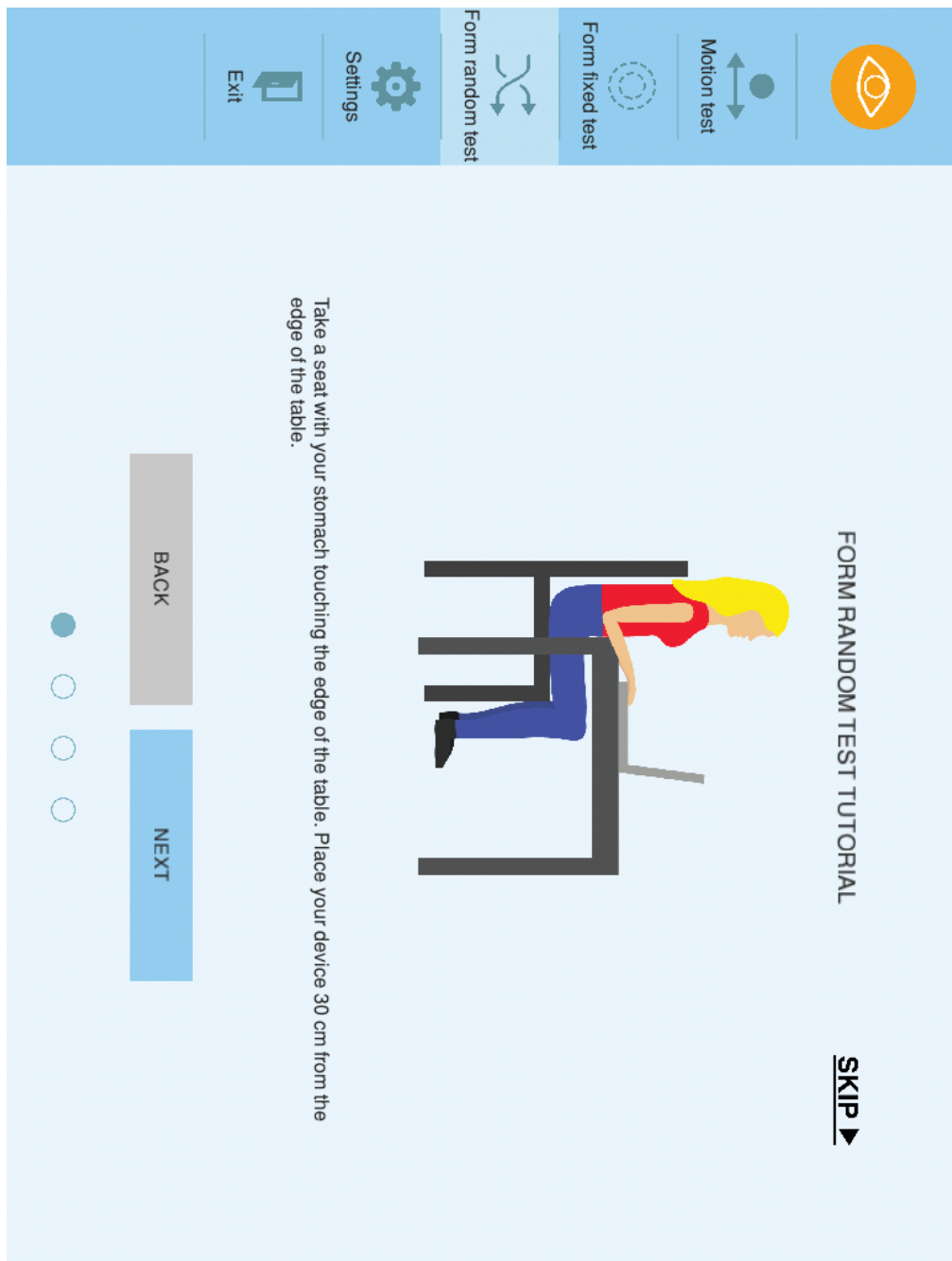


Figure G.4: Form random test tutorial - step 1.

G.2.2 Step 2

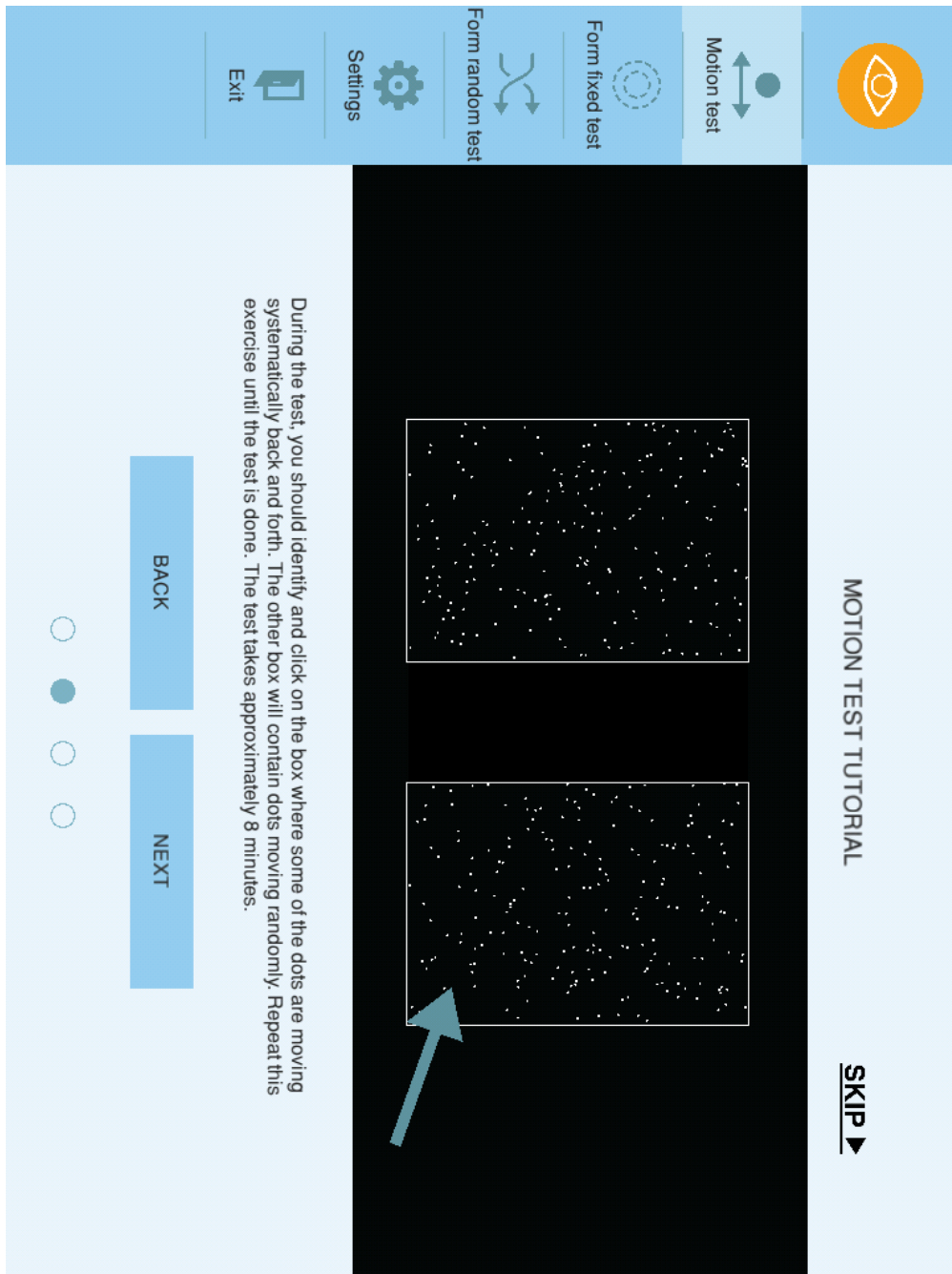


Figure G.5: Motion test tutorial - step 2.

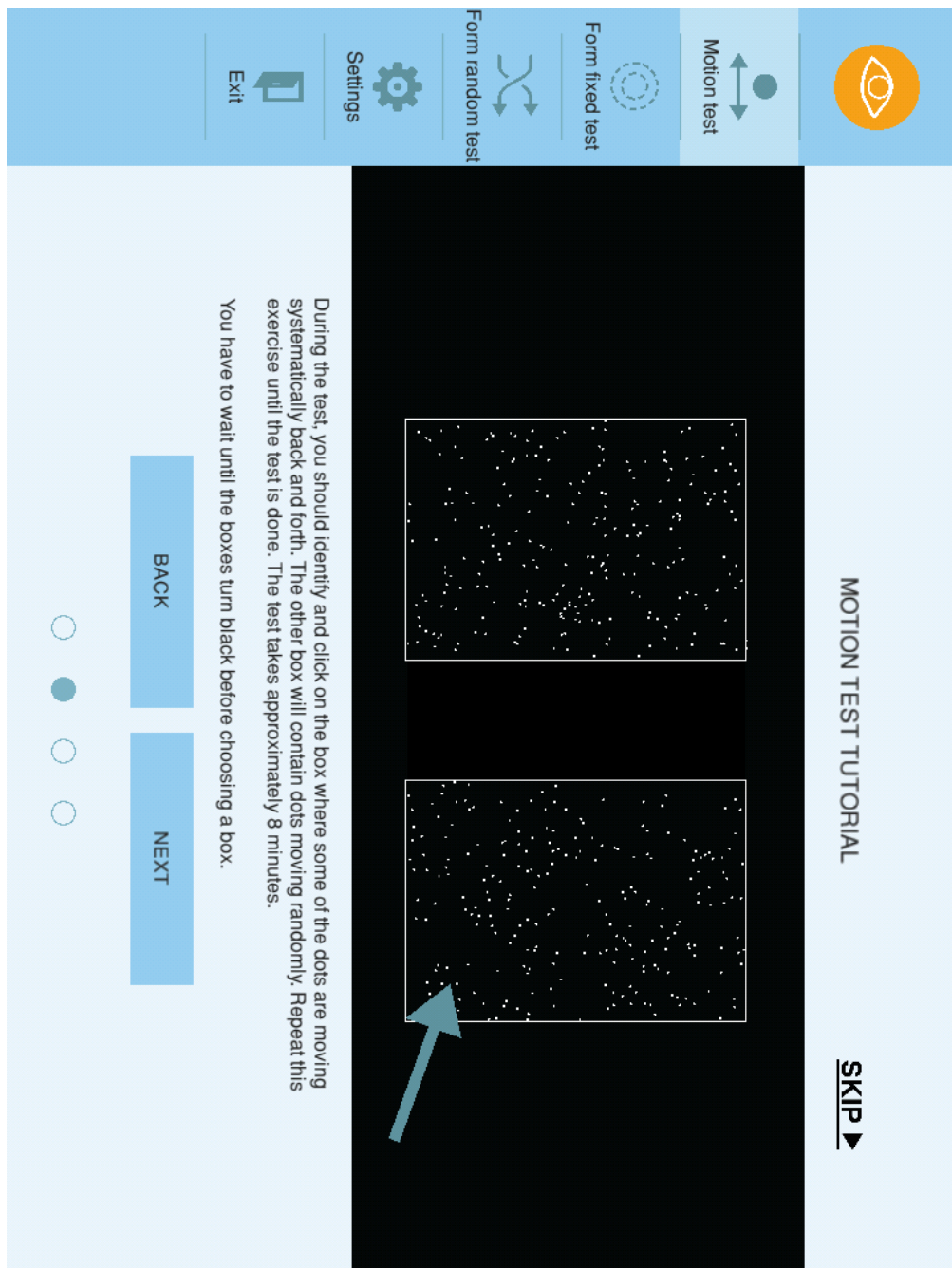


Figure G.6: Motion test tutorial, "Instant click registration" disabled - step 2.

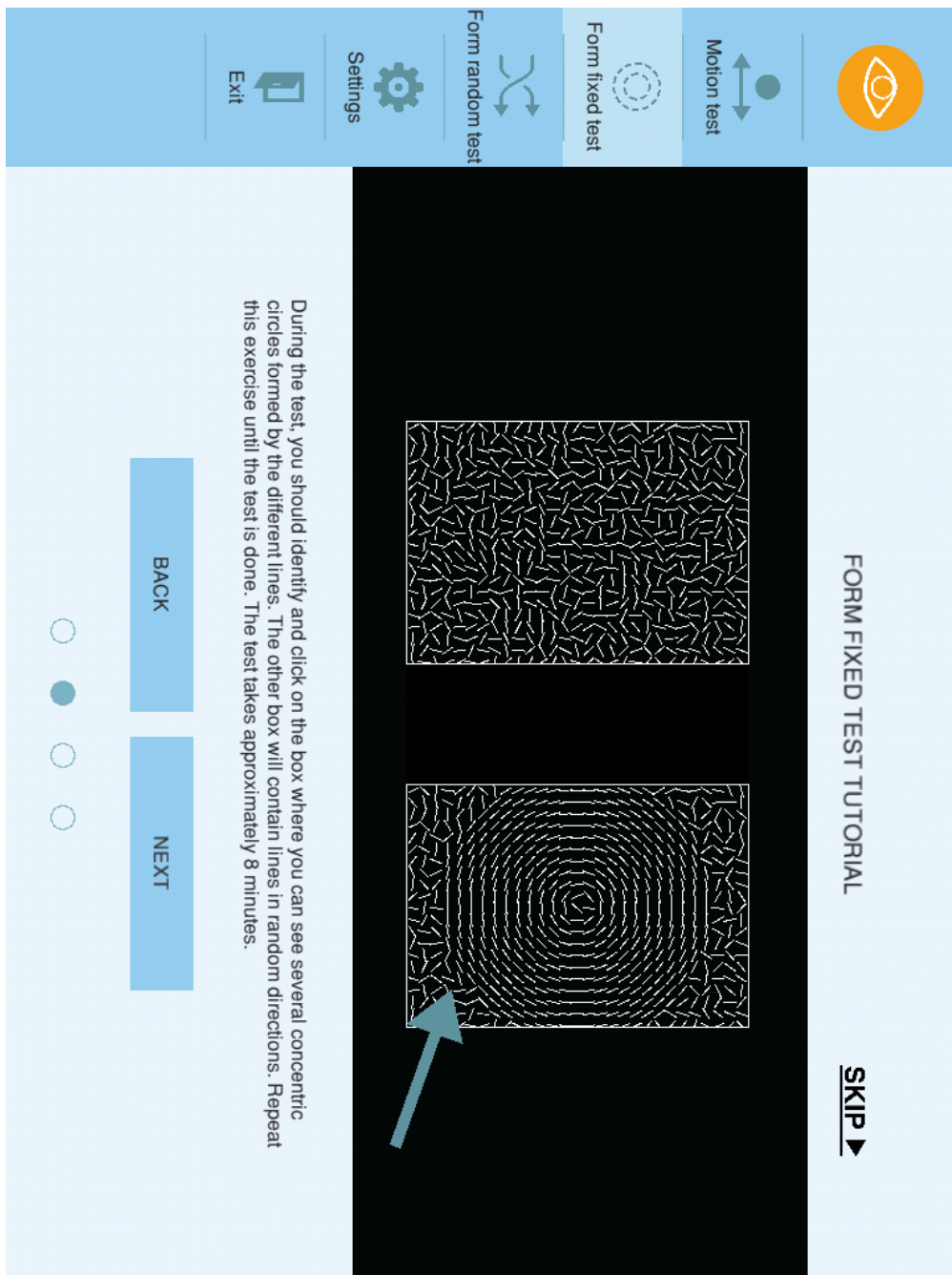


Figure G.7: Form fixed test tutorial - step 2.

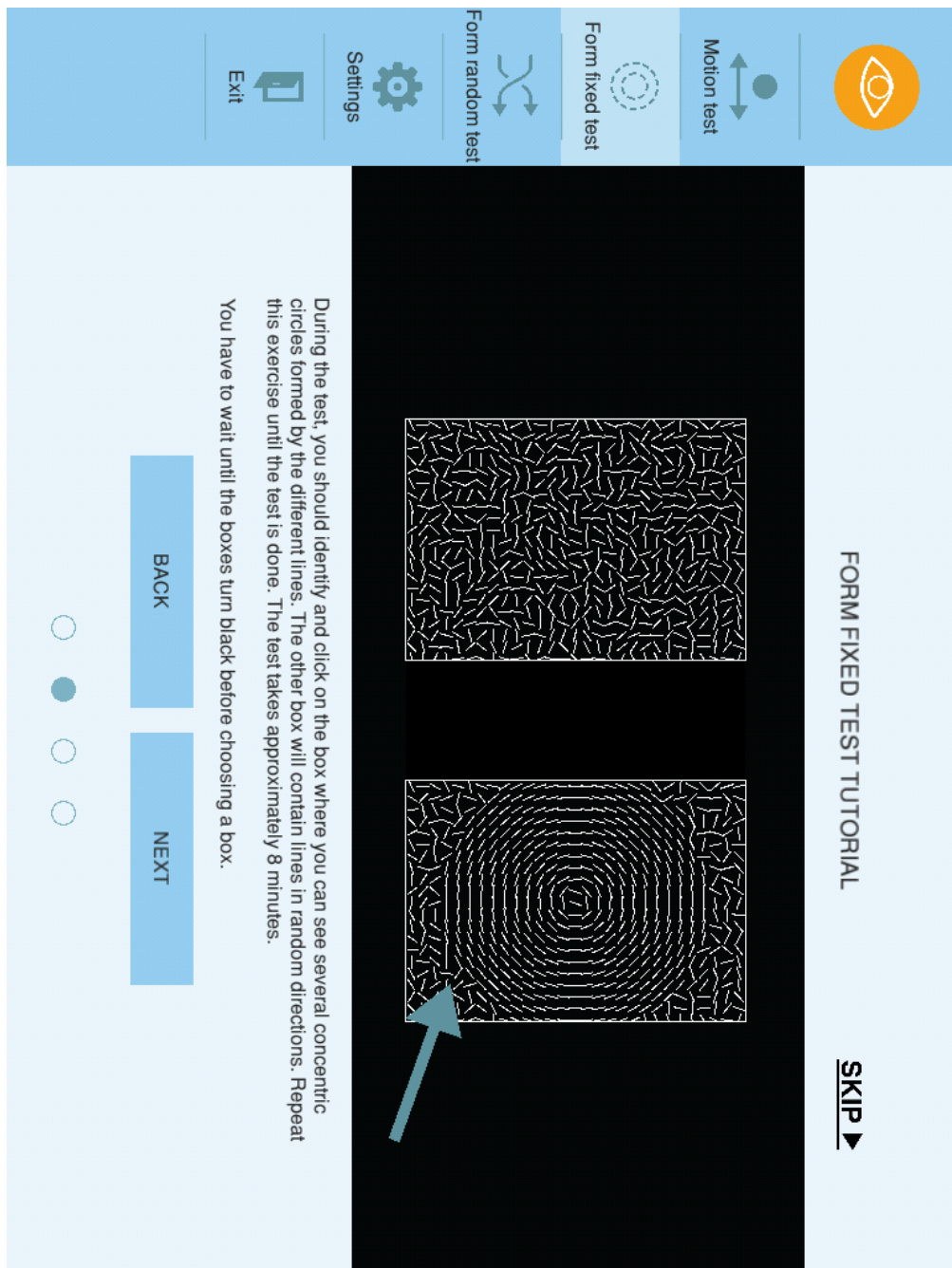


Figure G.8: Form fixed test tutorial, “Instant click registration” disabled - step 2.

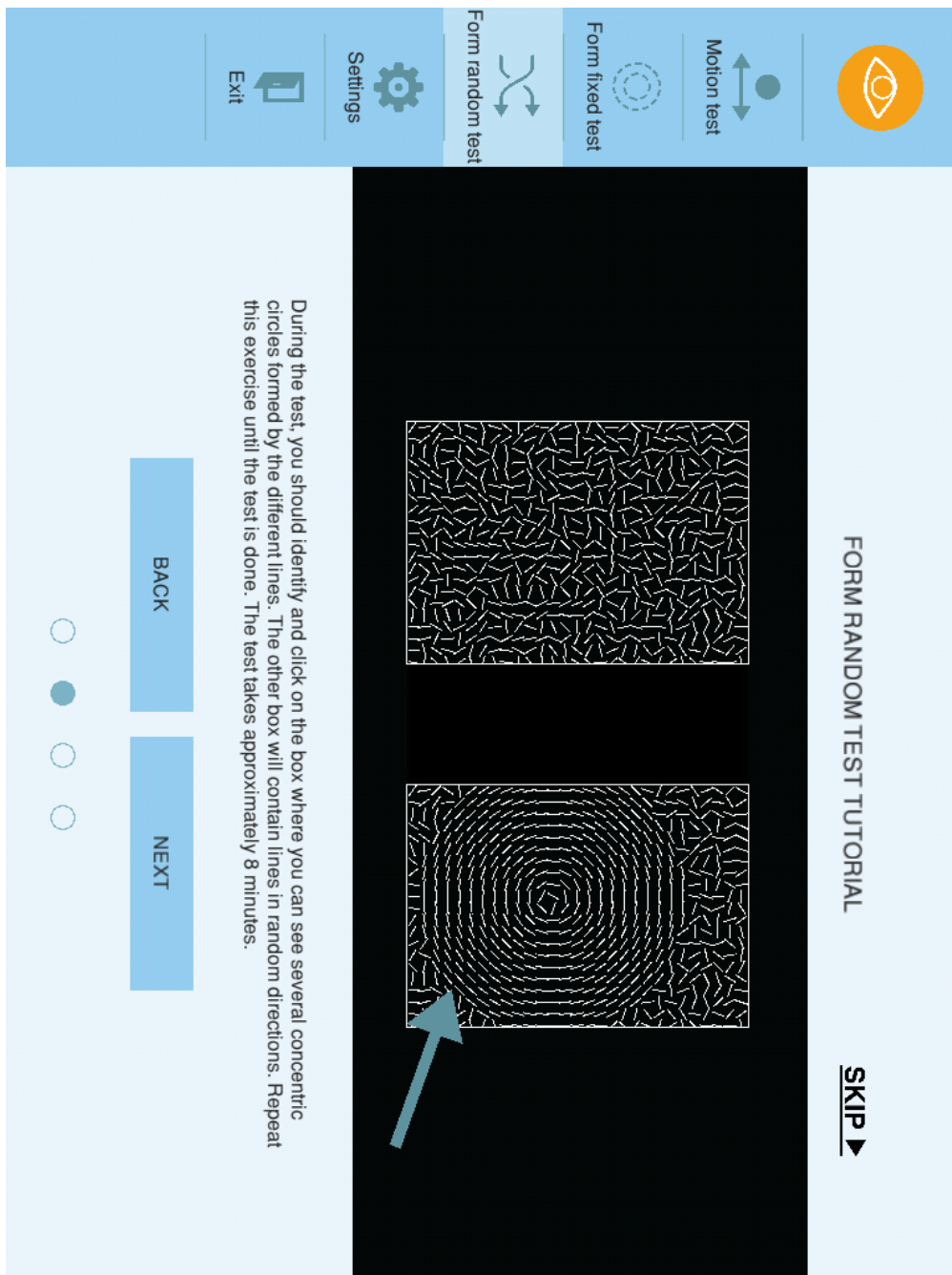


Figure G.9: Form random test tutorial - step 2.

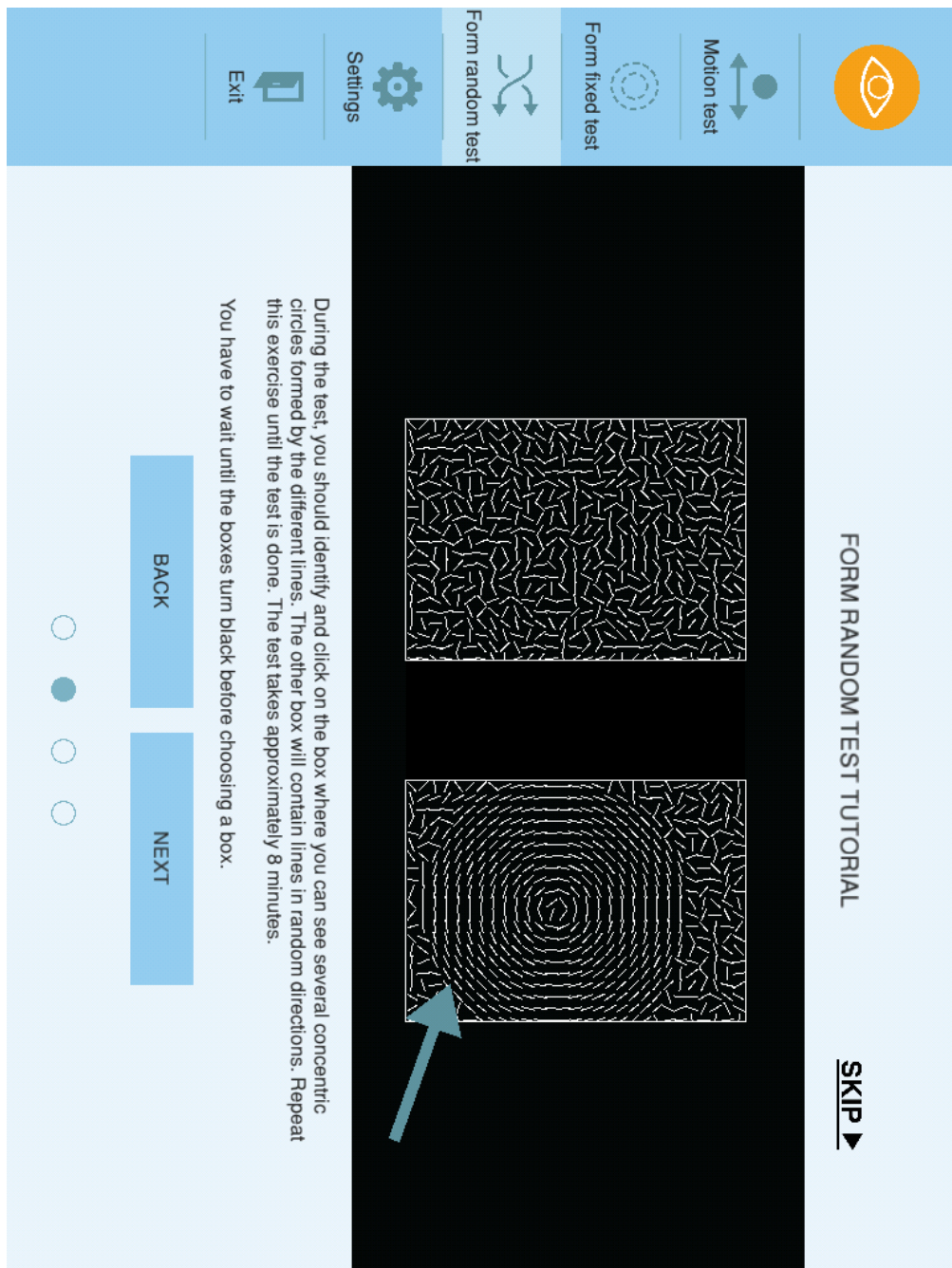


Figure G.10: Form random test tutorial, “Instant click registration” disabled - step 2.

G.2.3 Step 3

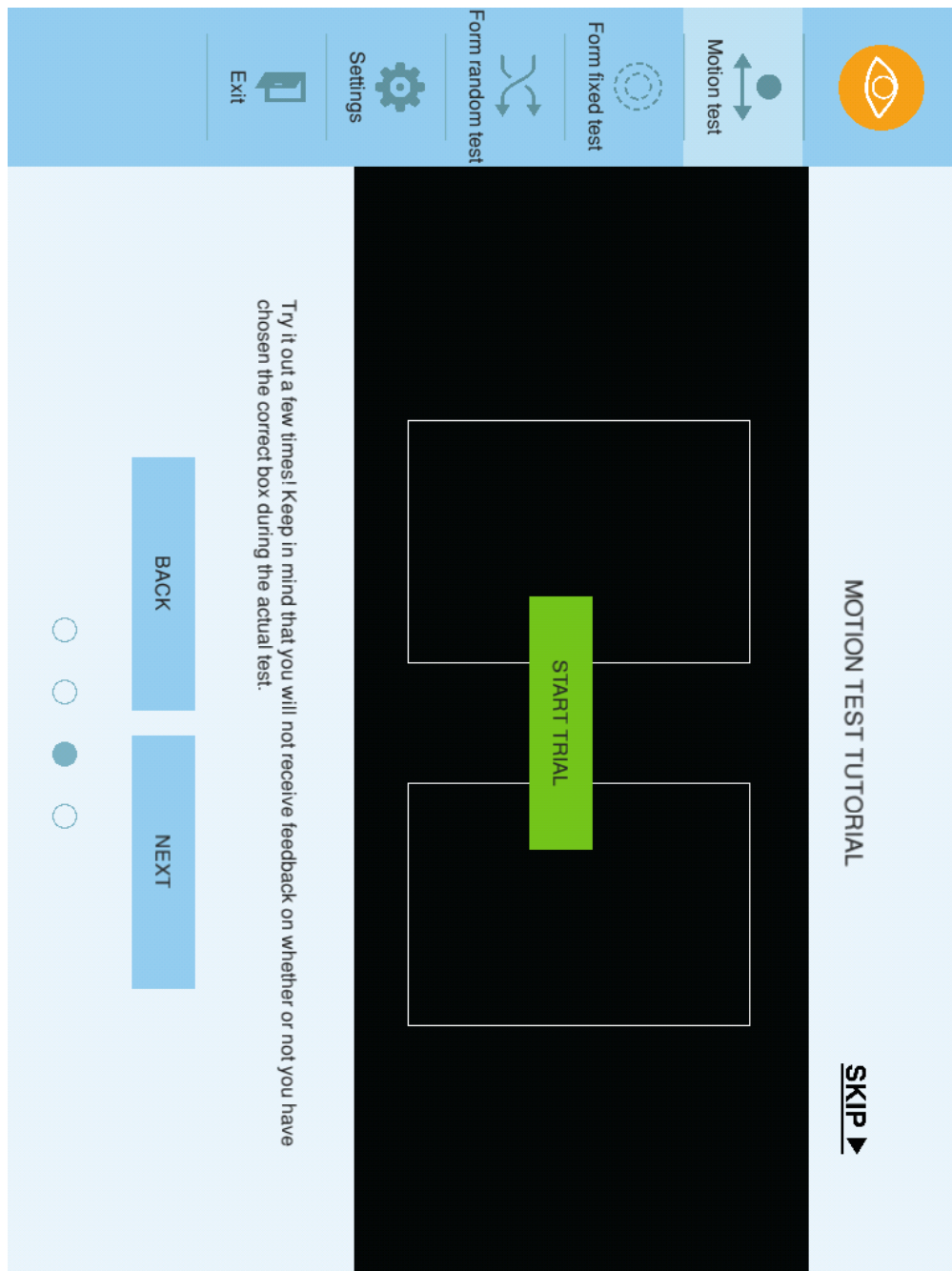


Figure G.11: Motion test tutorial, prior to trial start - step 3.

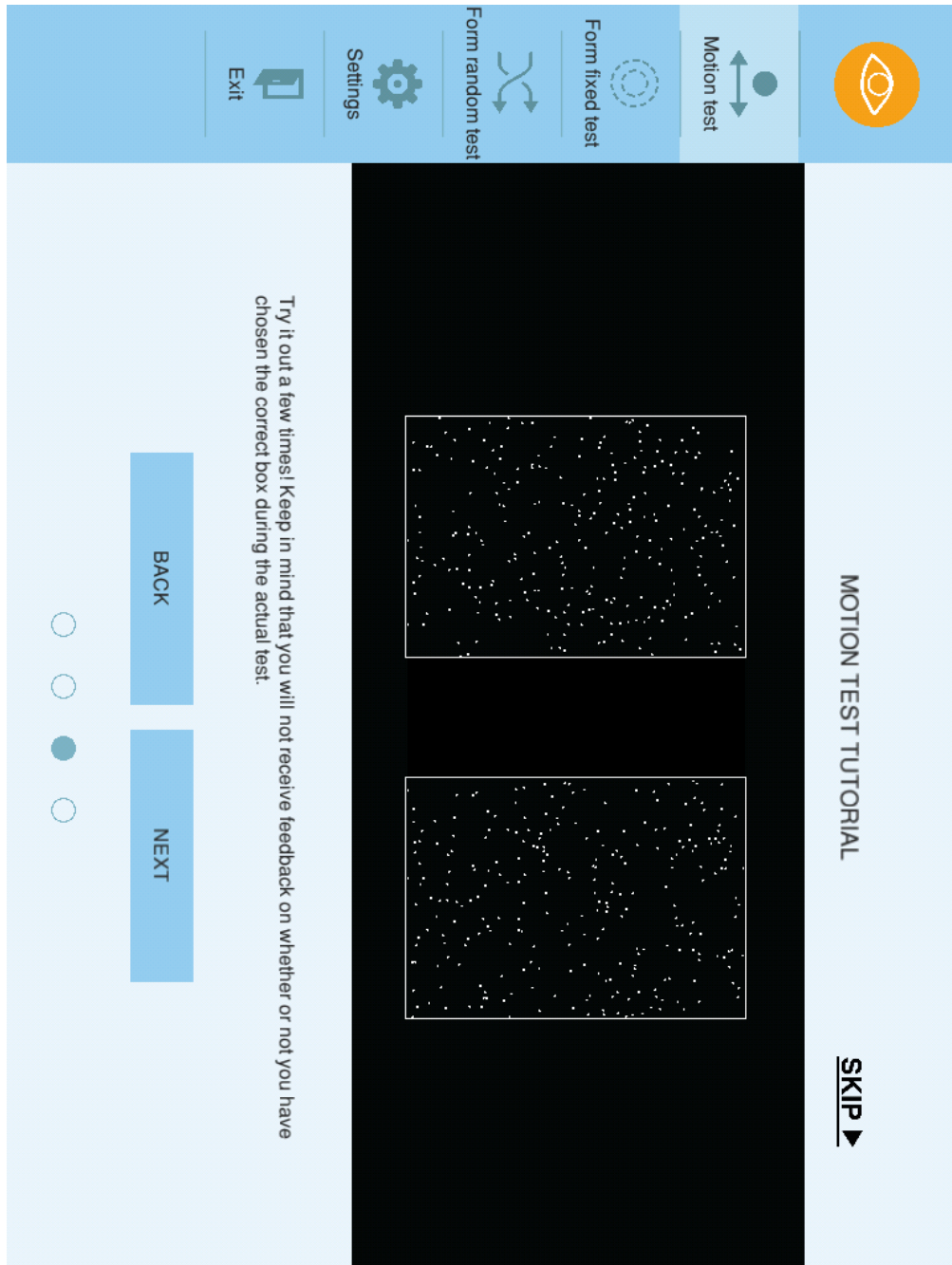


Figure G.12: Motion test tutorial, during trial - step 3.

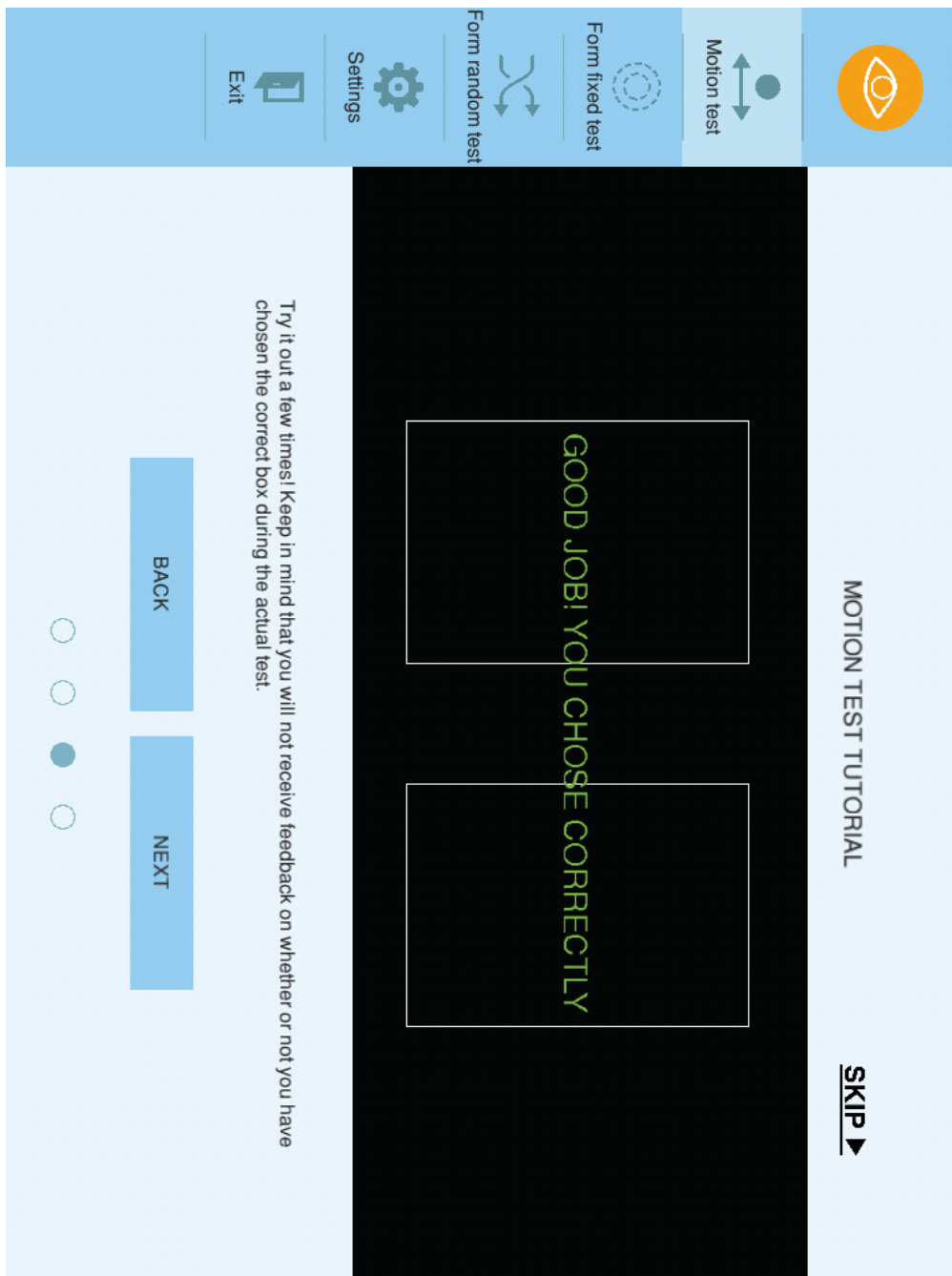


Figure G.13: Motion test tutorial, correct answer - step 3.

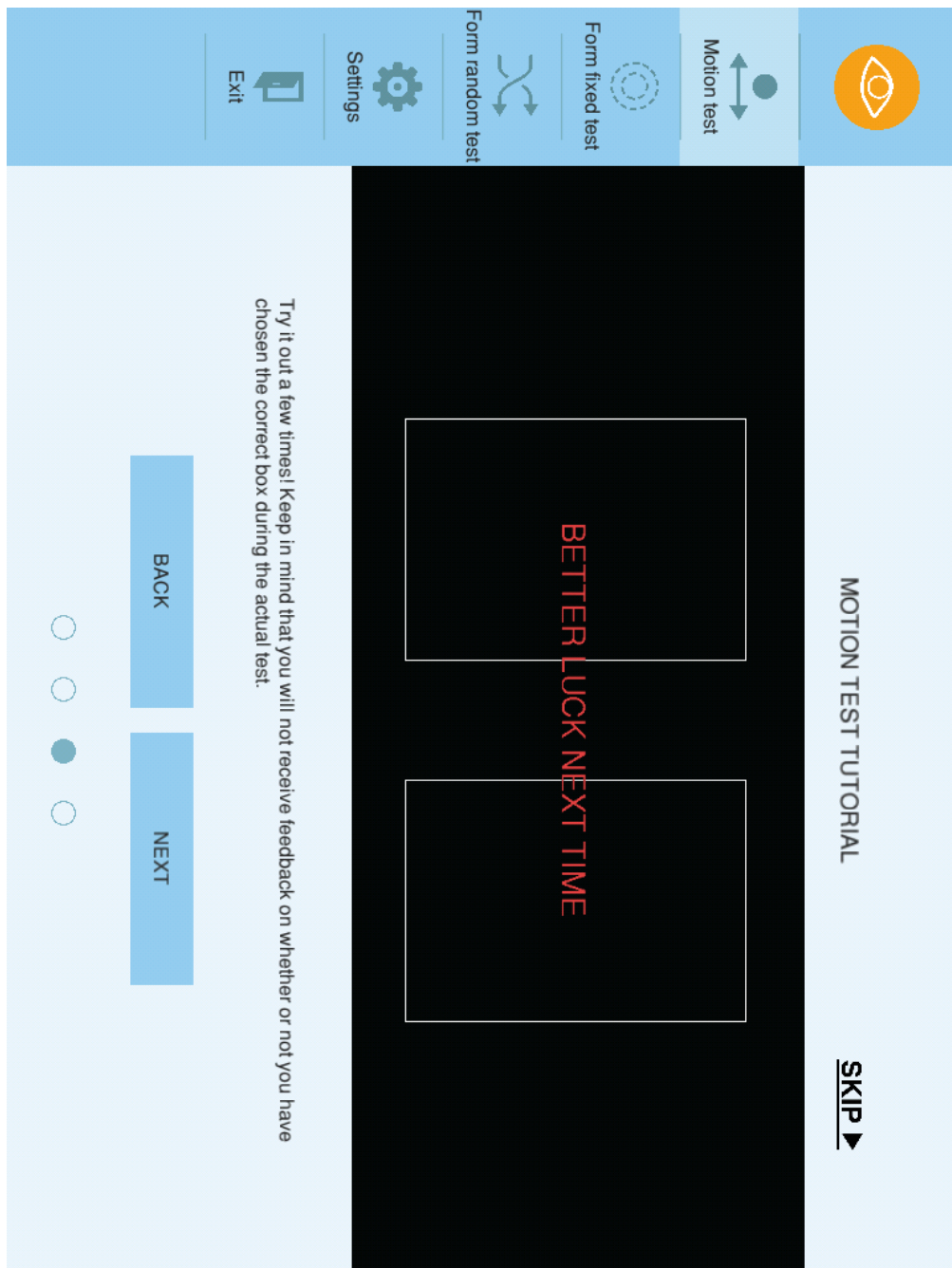


Figure G.14: Motion test tutorial, wrong answer - step 3.

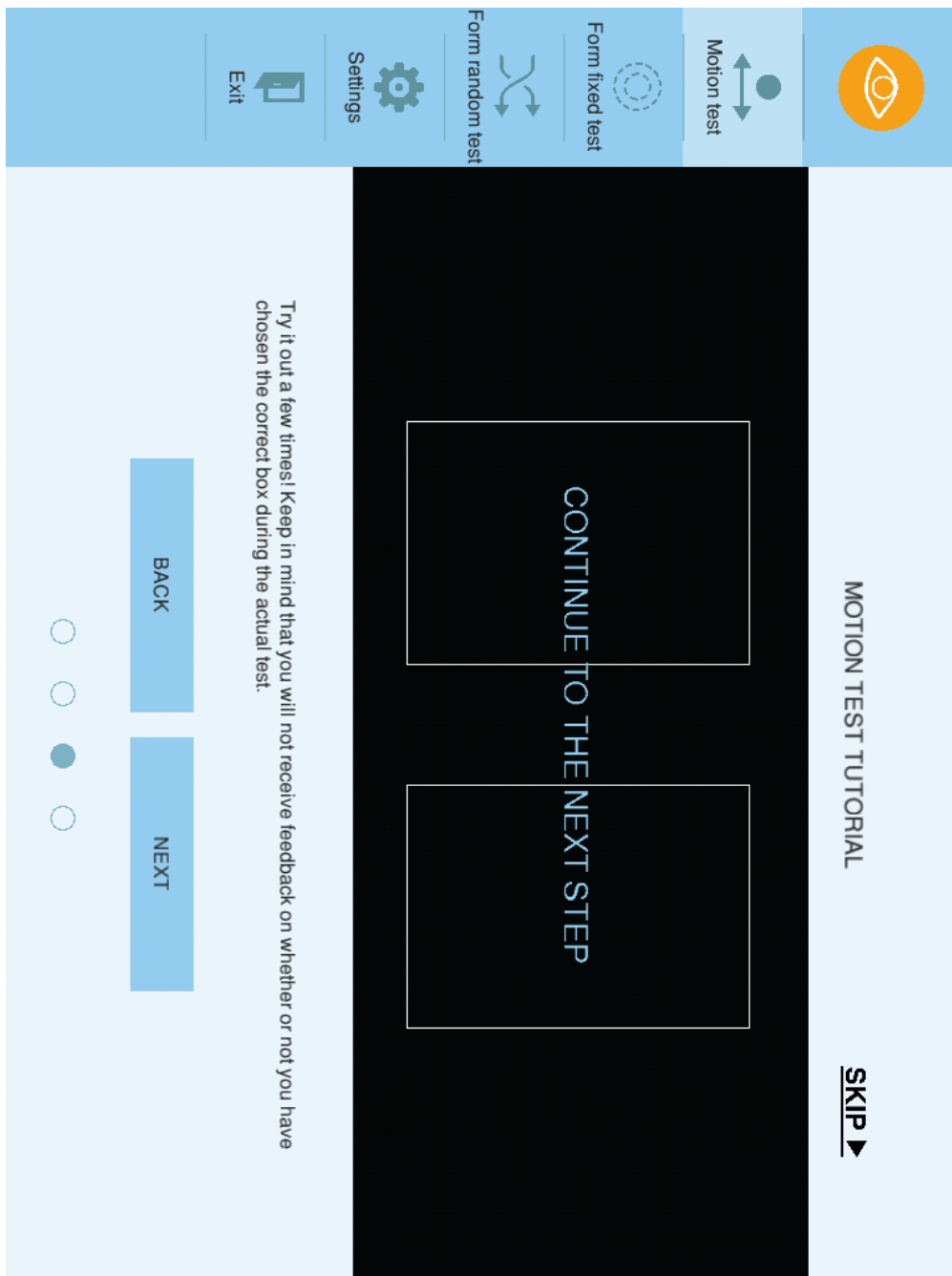


Figure G.15: Motion test tutorial, completed trial - step 3.

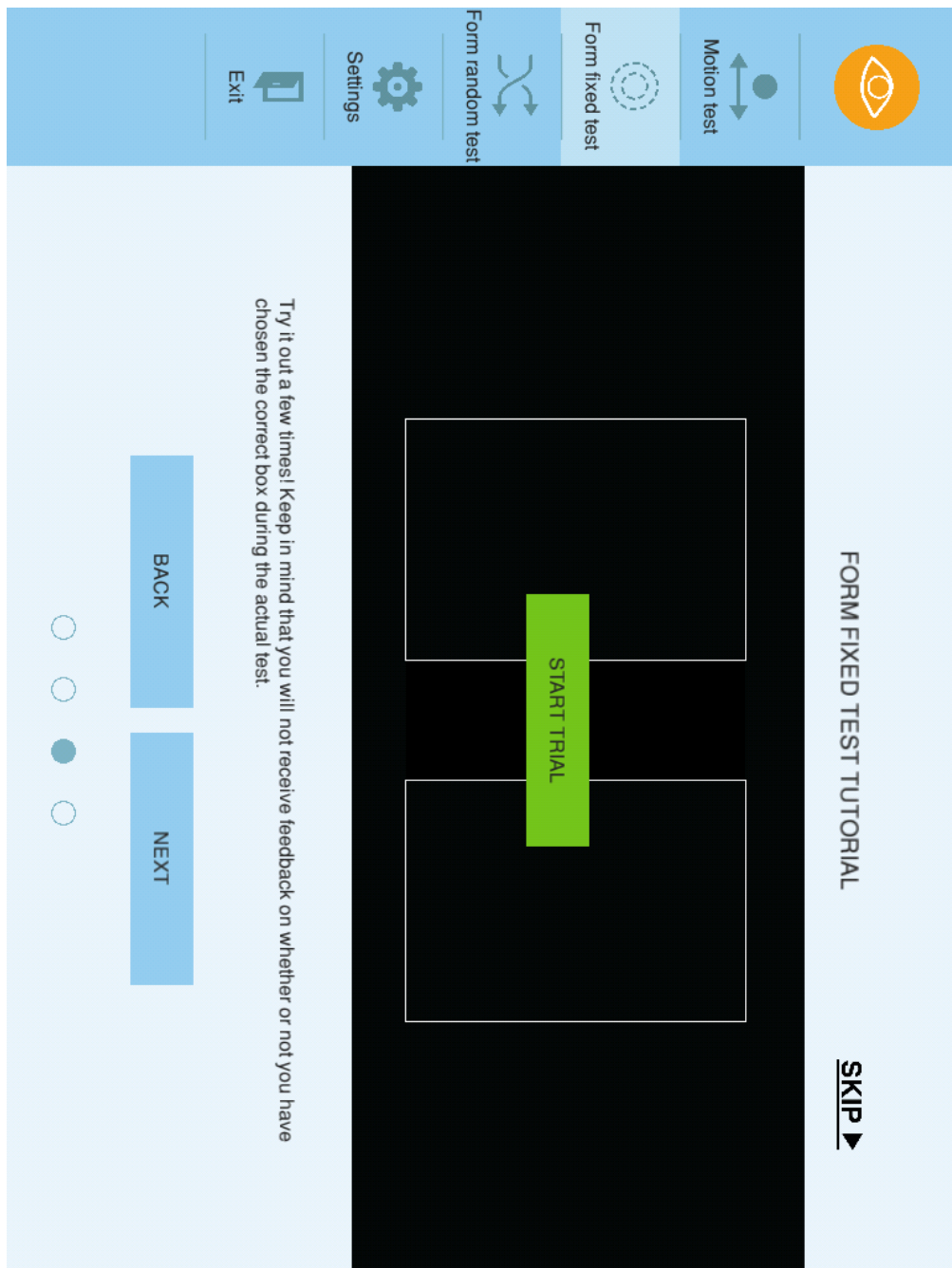


Figure G.16: Form fixed test tutorial, prior to trial start - step 3.

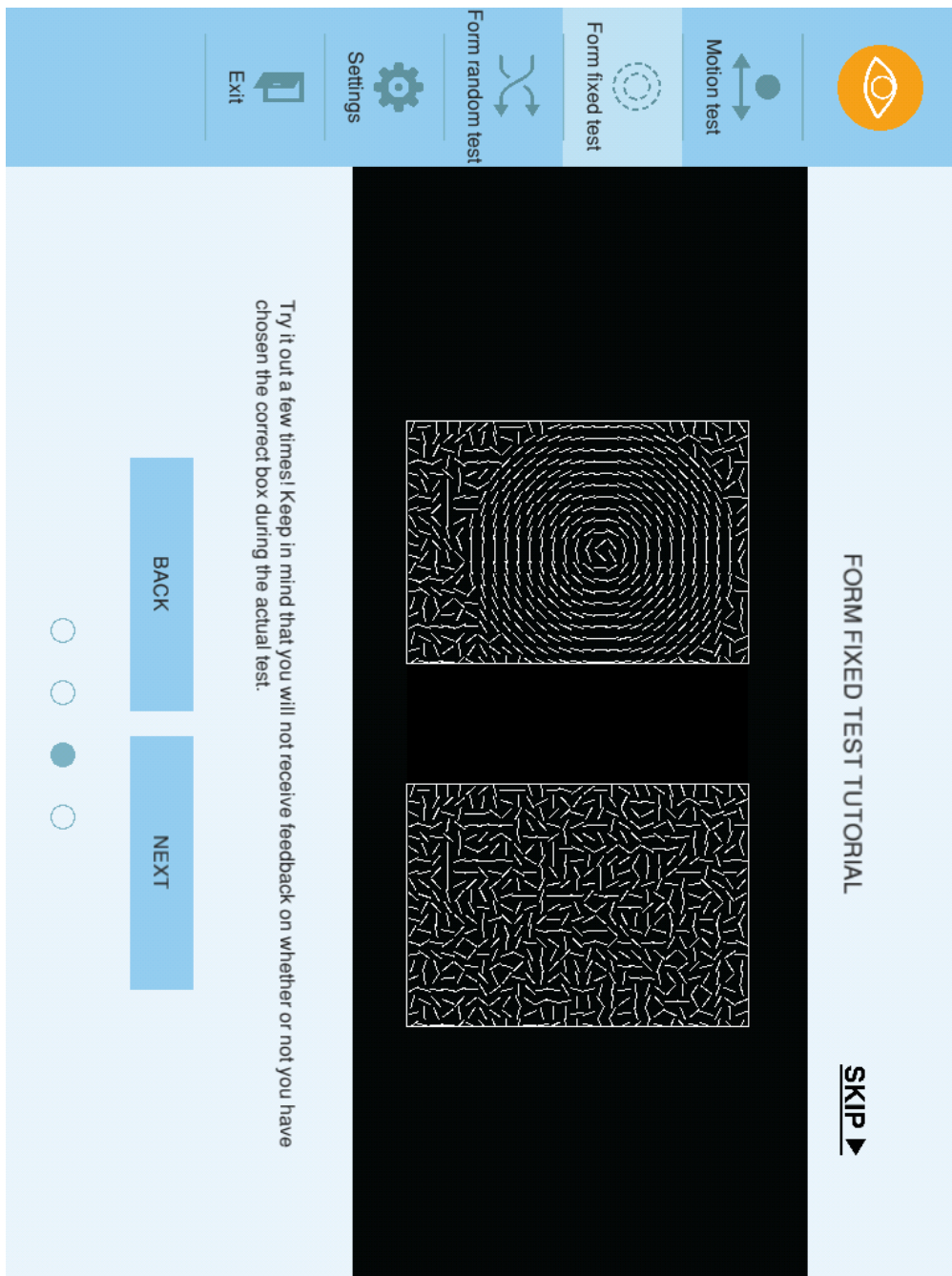


Figure G.17: Form fixed test tutorial, during trial - step 3.

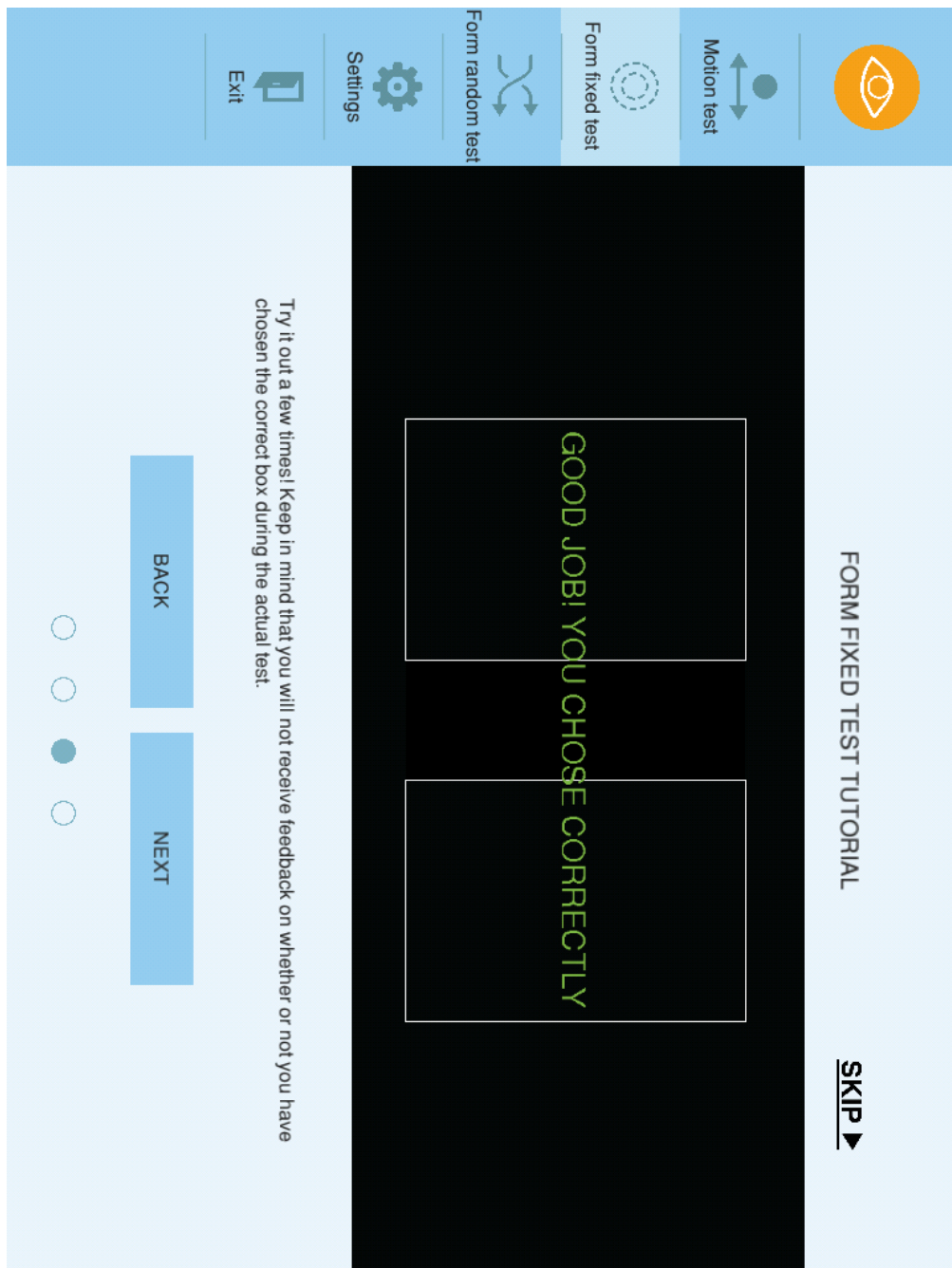


Figure G.18: Form fixed test tutorial, correct answer - step 3.

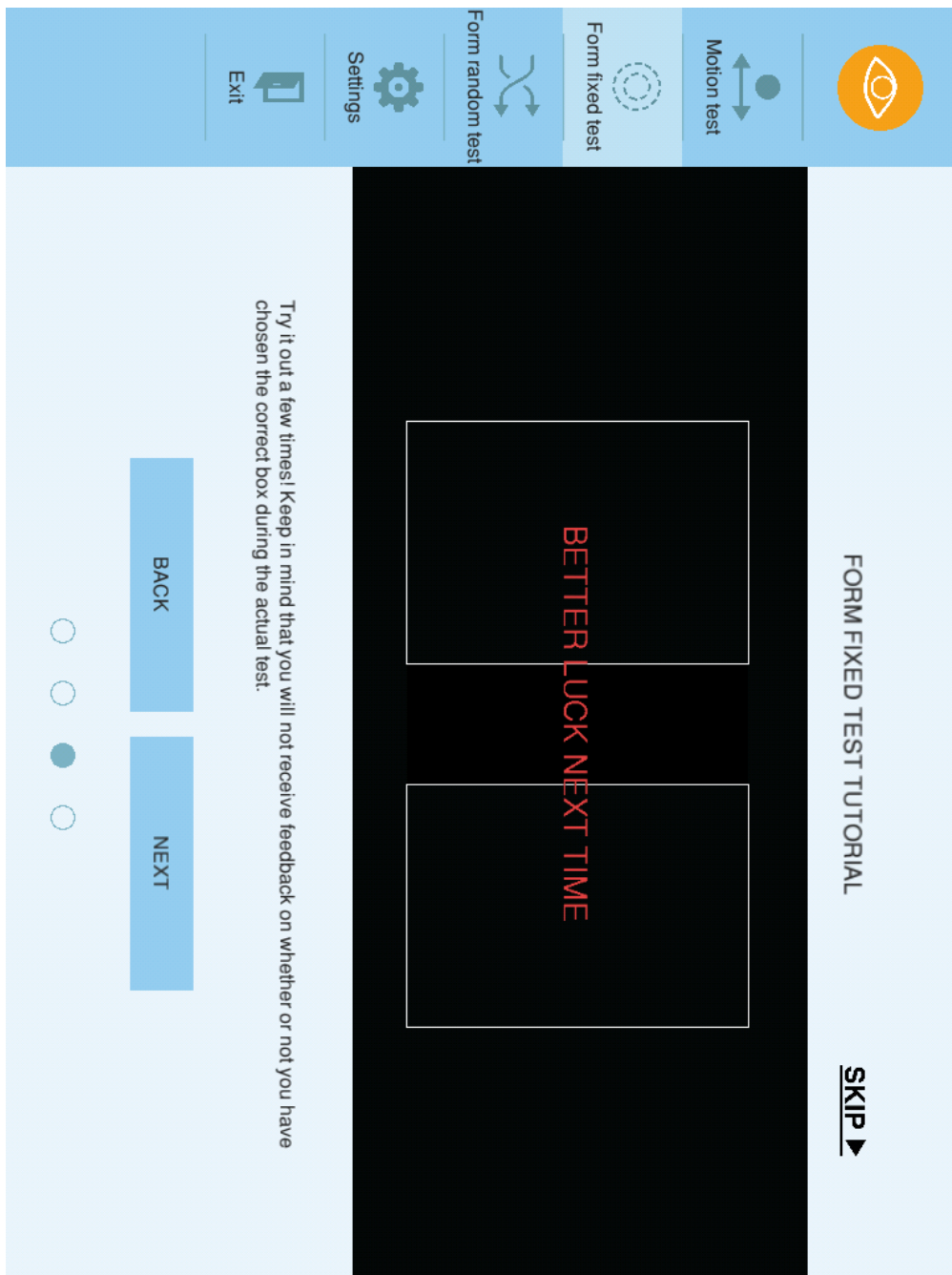


Figure G.19: Form fixed test tutorial, wrong answer - step 3.

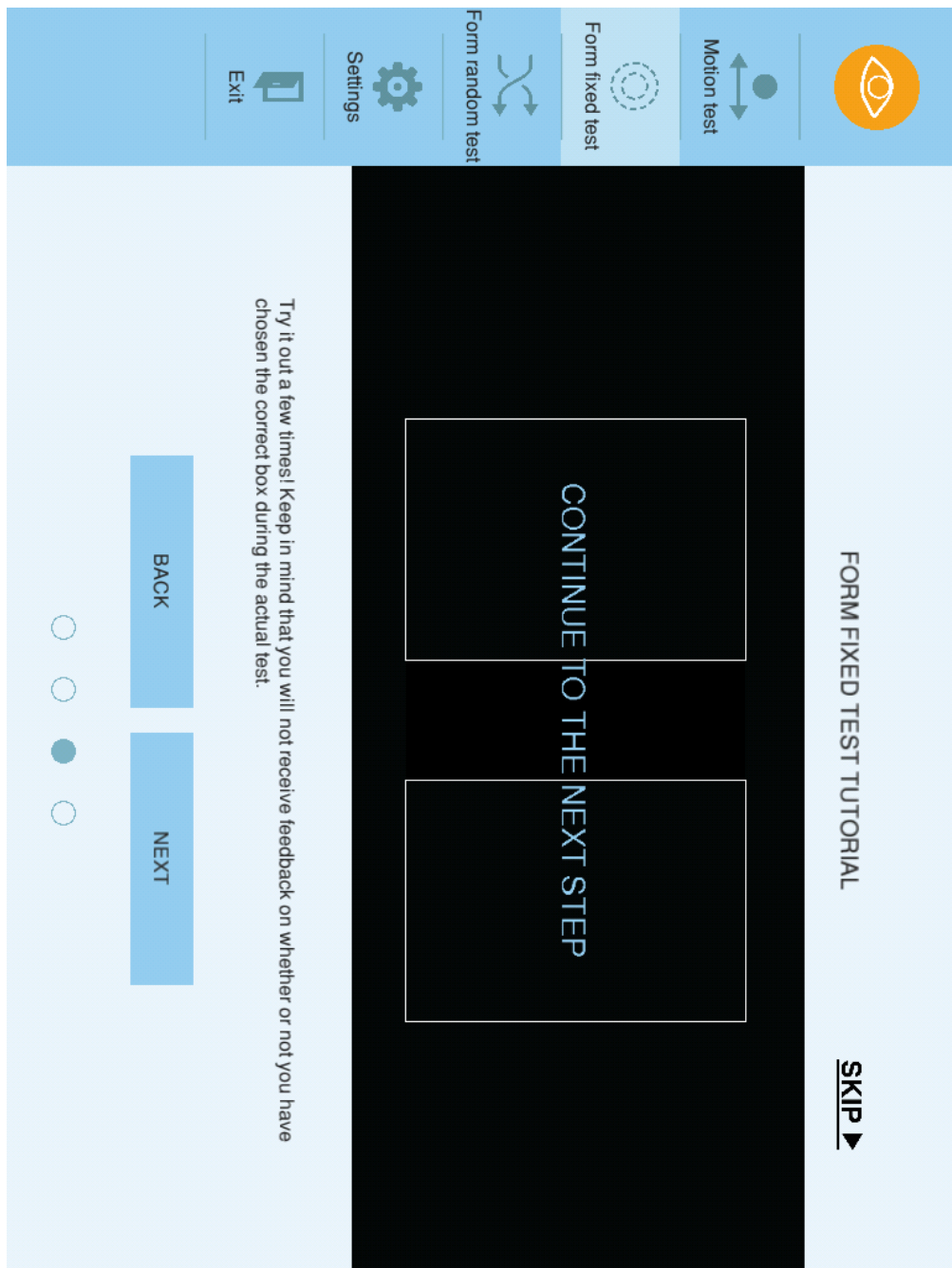


Figure G.20: Form fixed test tutorial, completed trial - step 3.

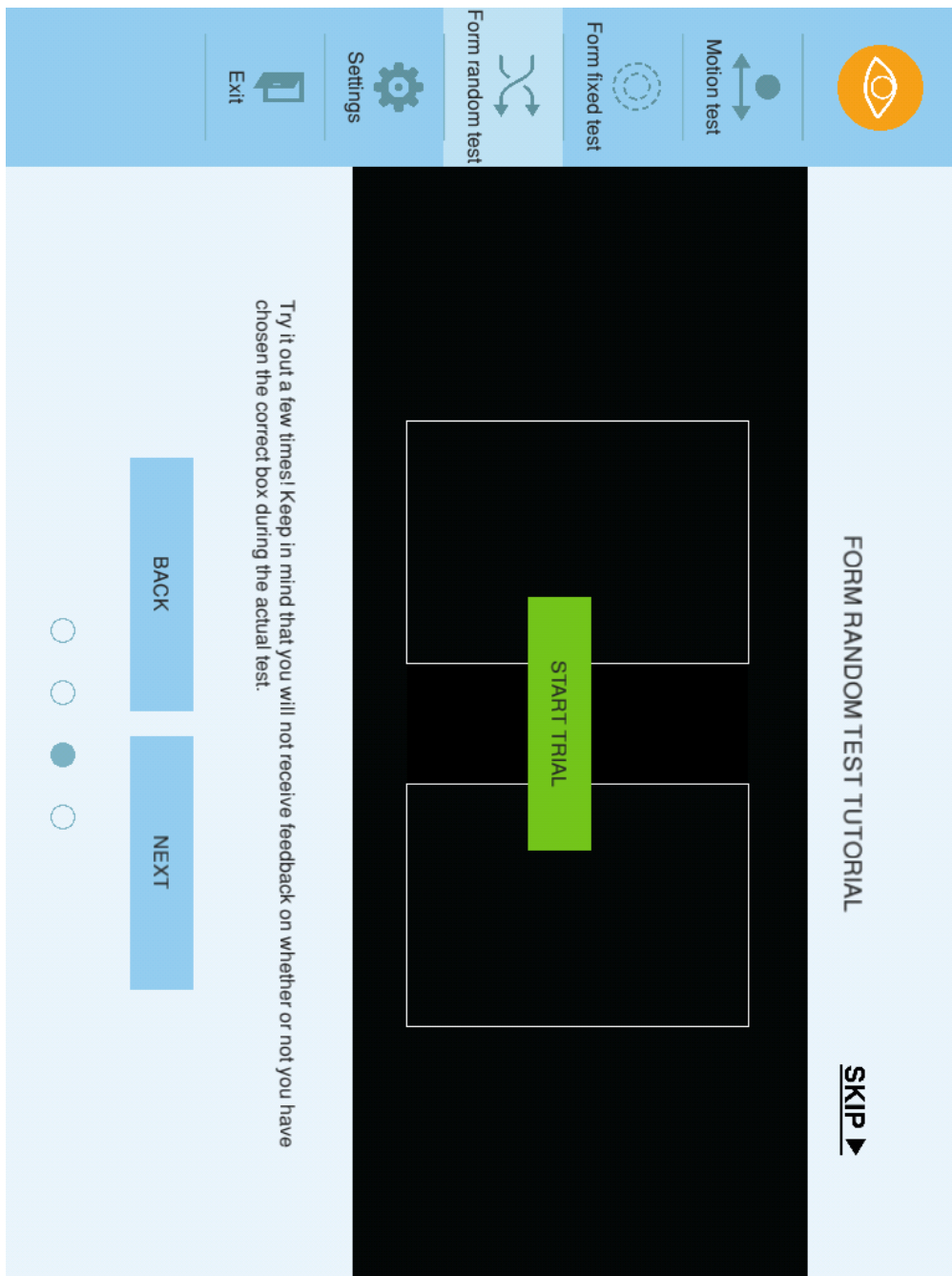


Figure G.21: Form random test tutorial, prior to trial start - step 3.

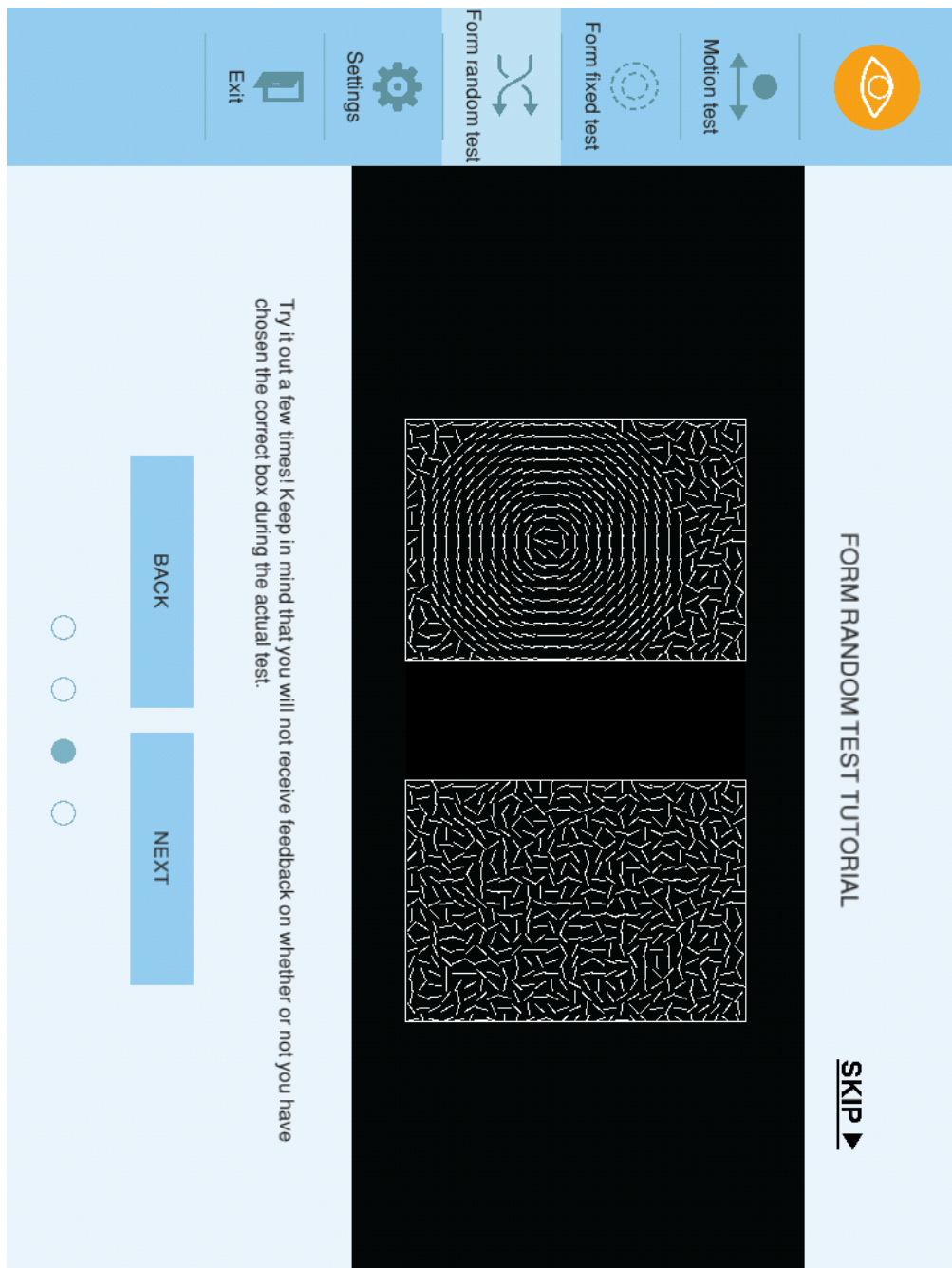


Figure G.22: Form random test tutorial, during trial - step 3.

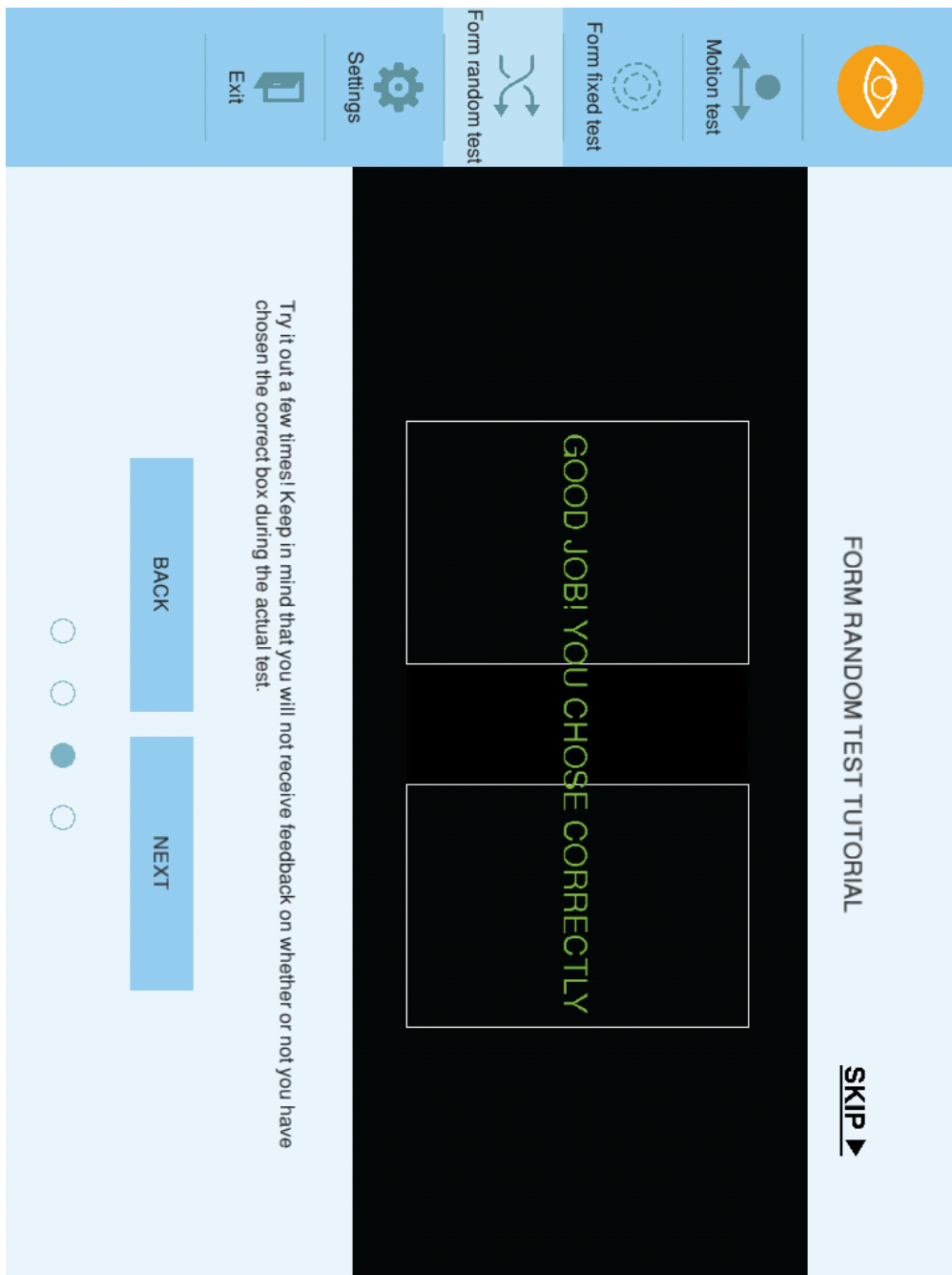


Figure G.23: Form random test tutorial, correct answer - step 3.

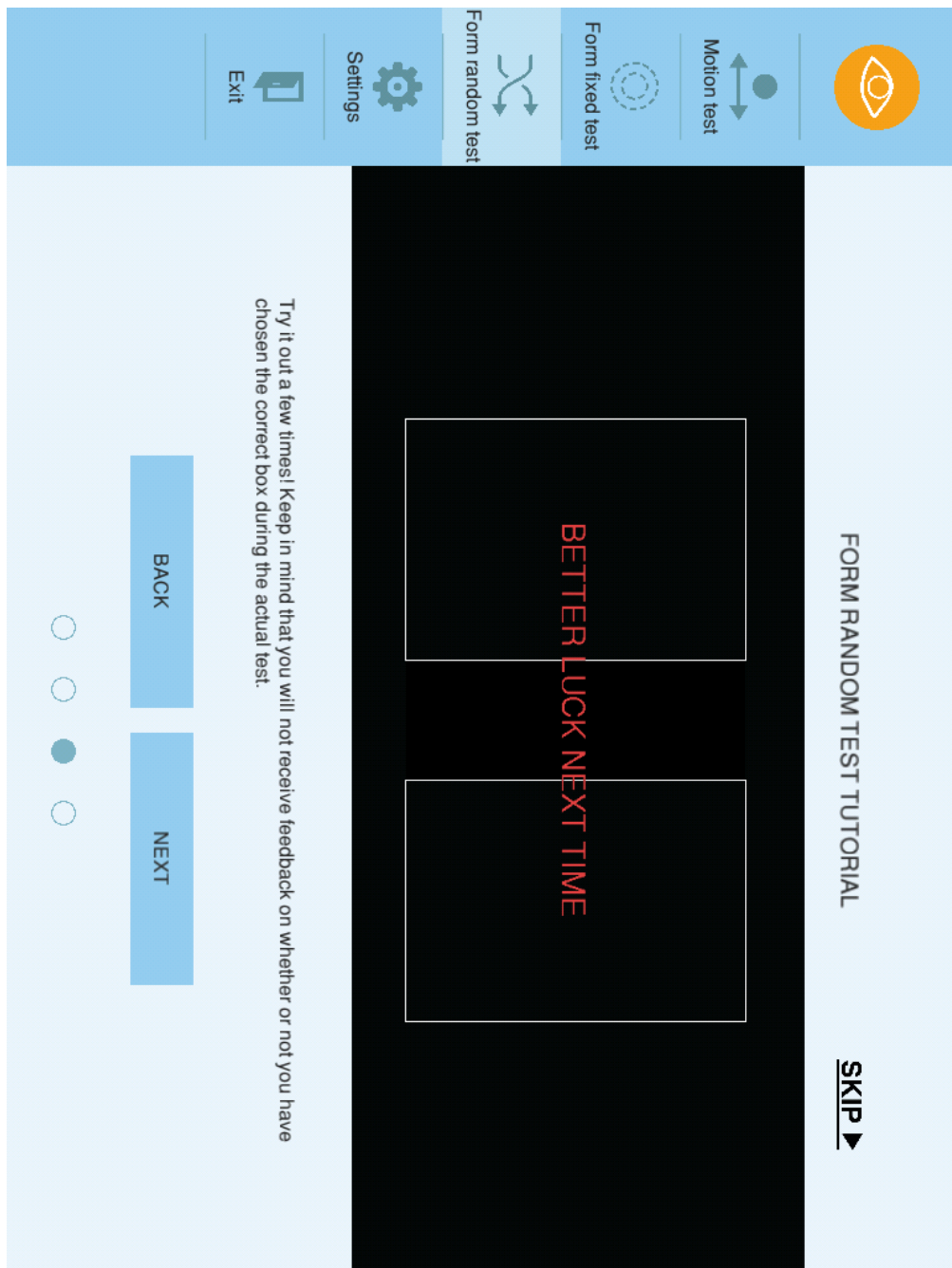


Figure G.24: Form random test tutorial, wrong answer - step 3.

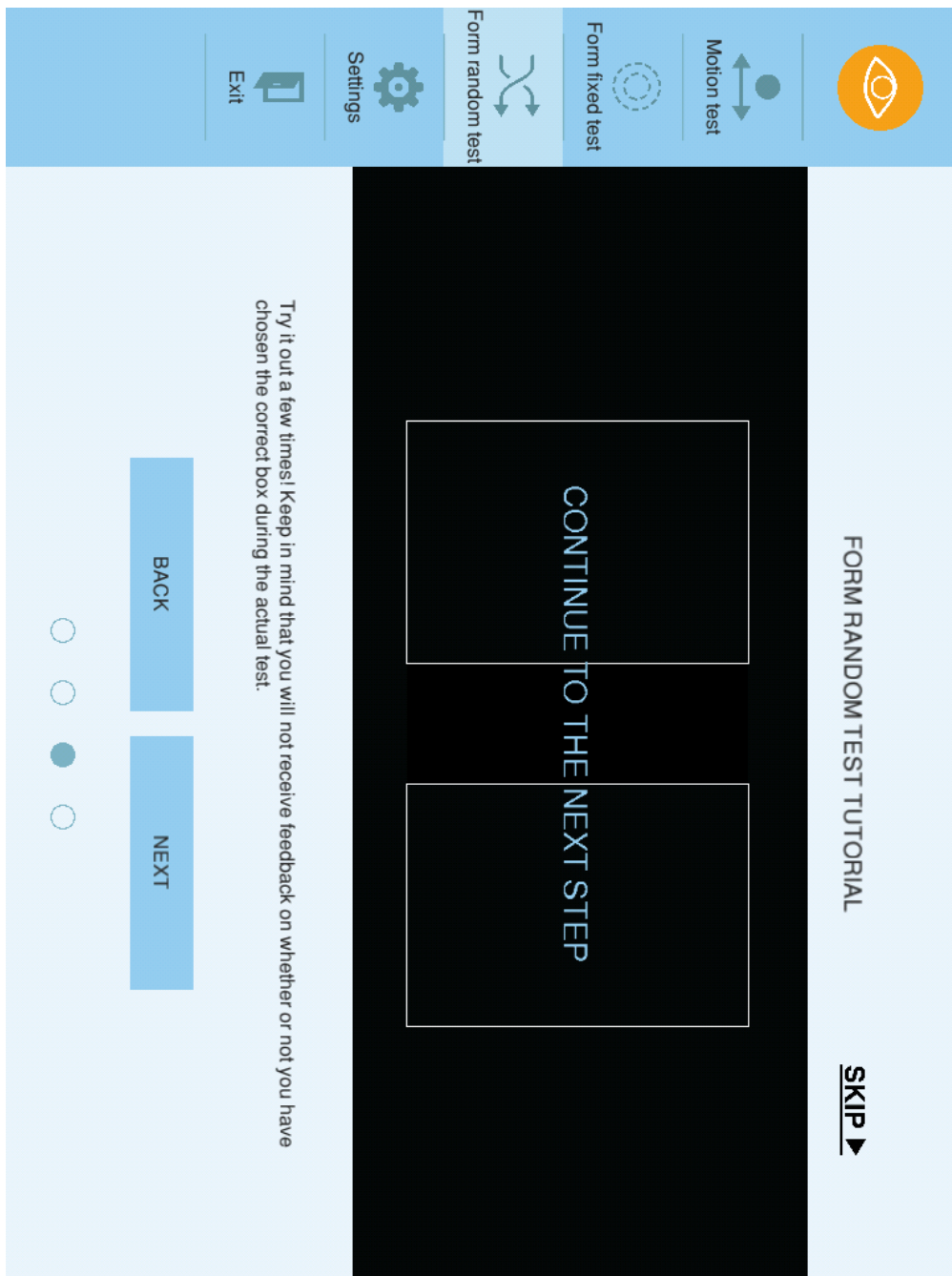


Figure G.25: Form random test tutorial, completed trial - step 3.

G.2.4 Step 4

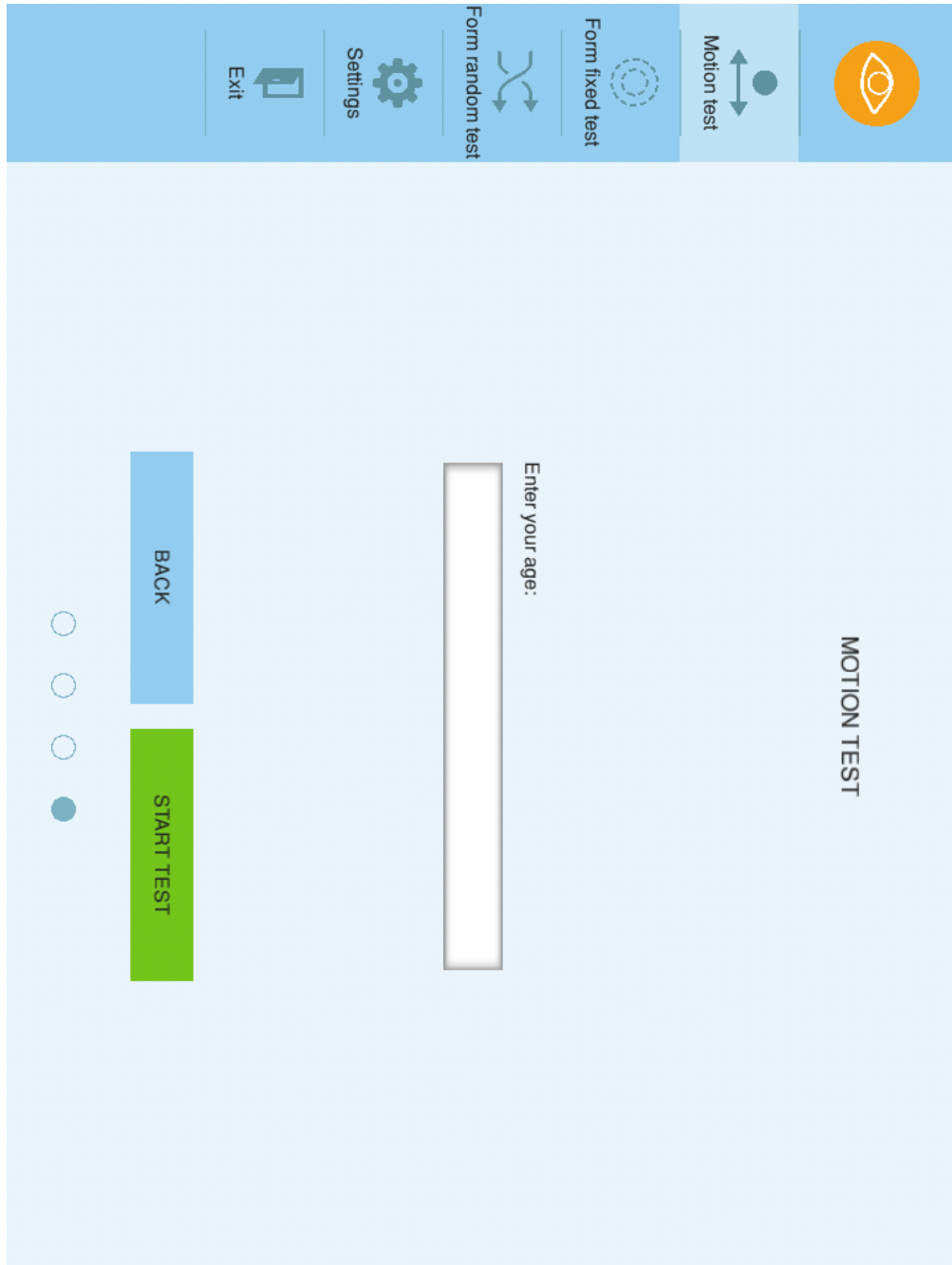


Figure G.26: Motion test tutorial - step 4.

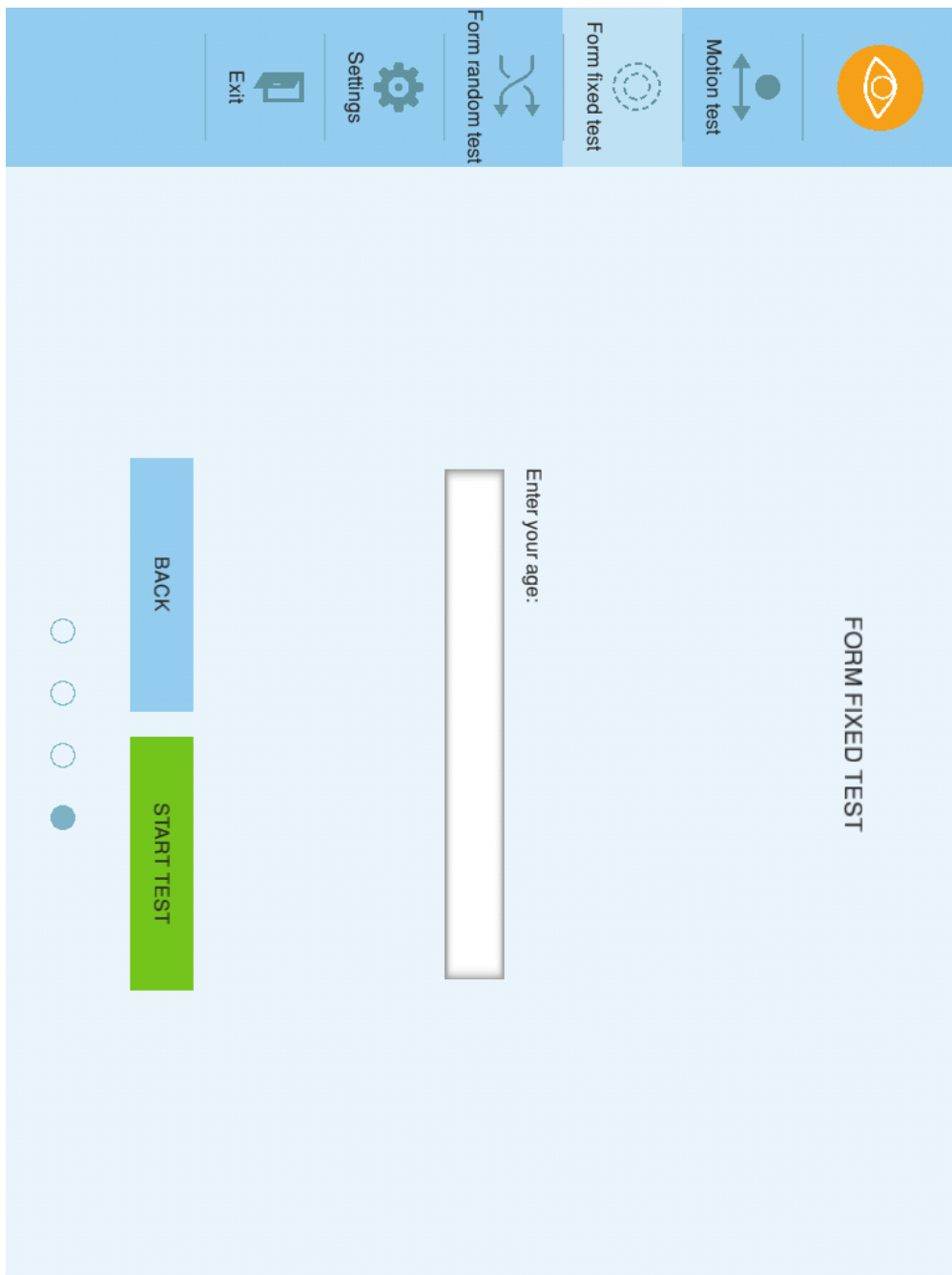


Figure G.27: Form fixed test tutorial - step 4.

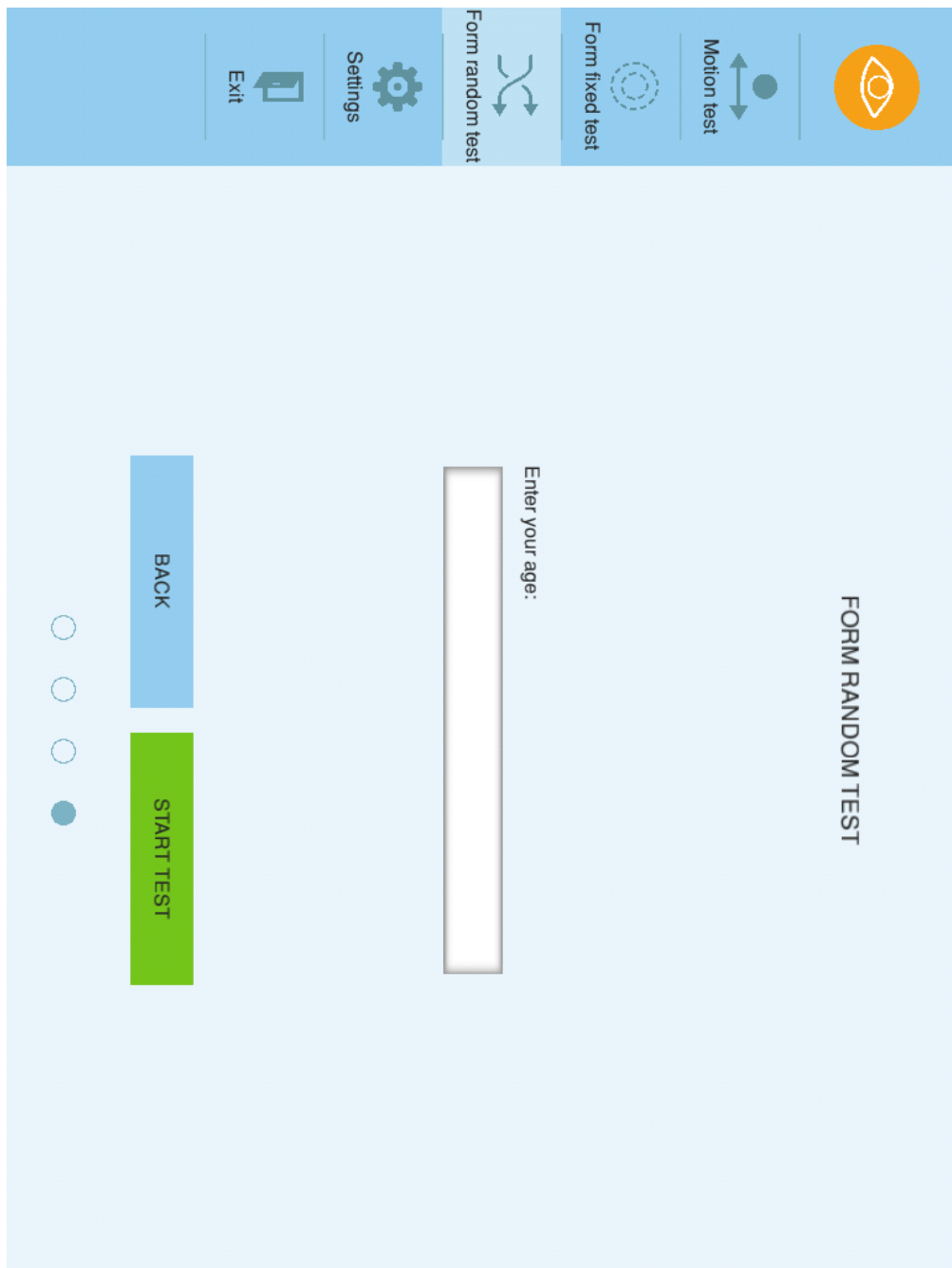


Figure G.28: Form random test tutorial - step 4.

G.3 Tests

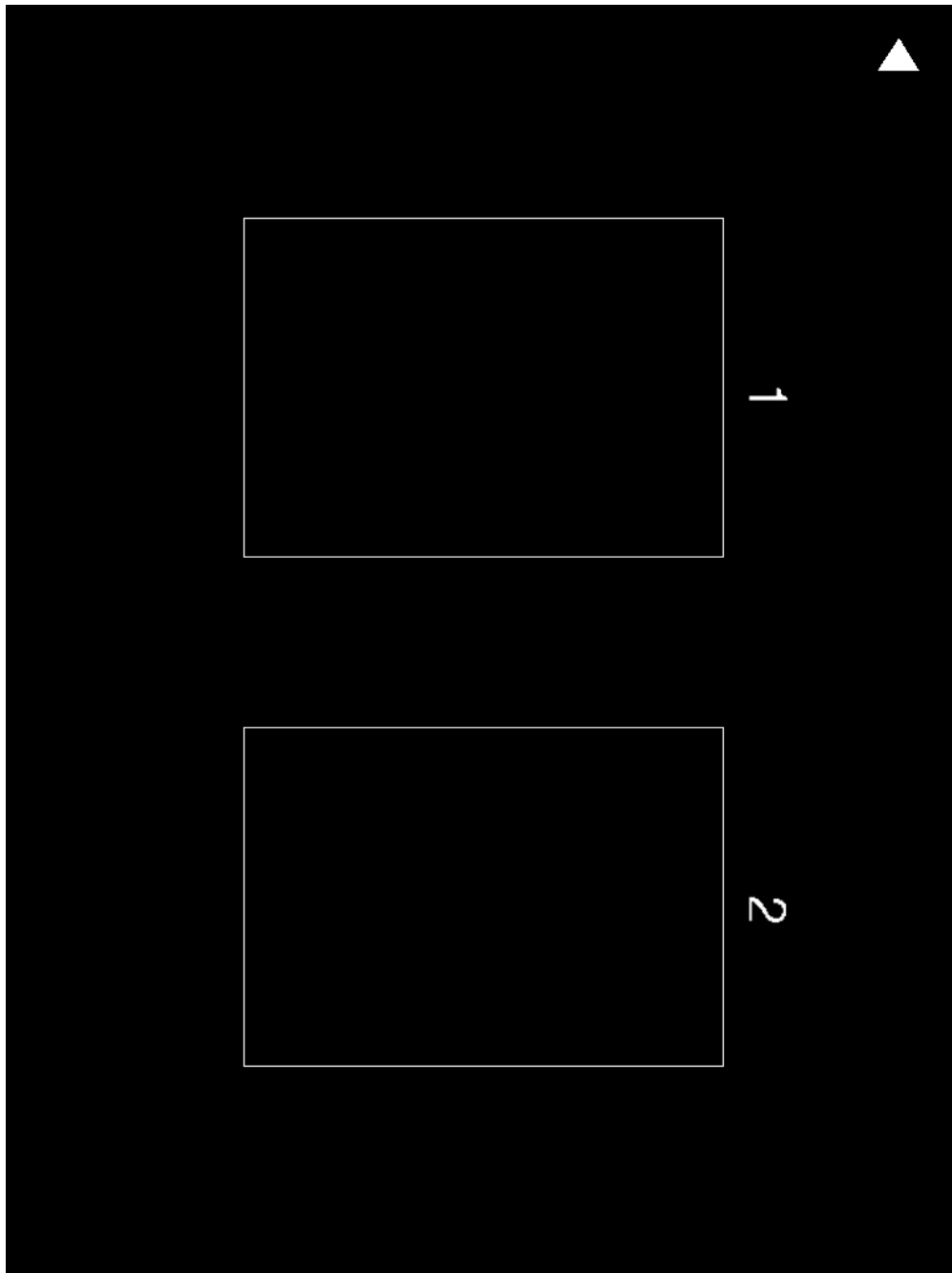


Figure G.29: Entered test - prior to start.

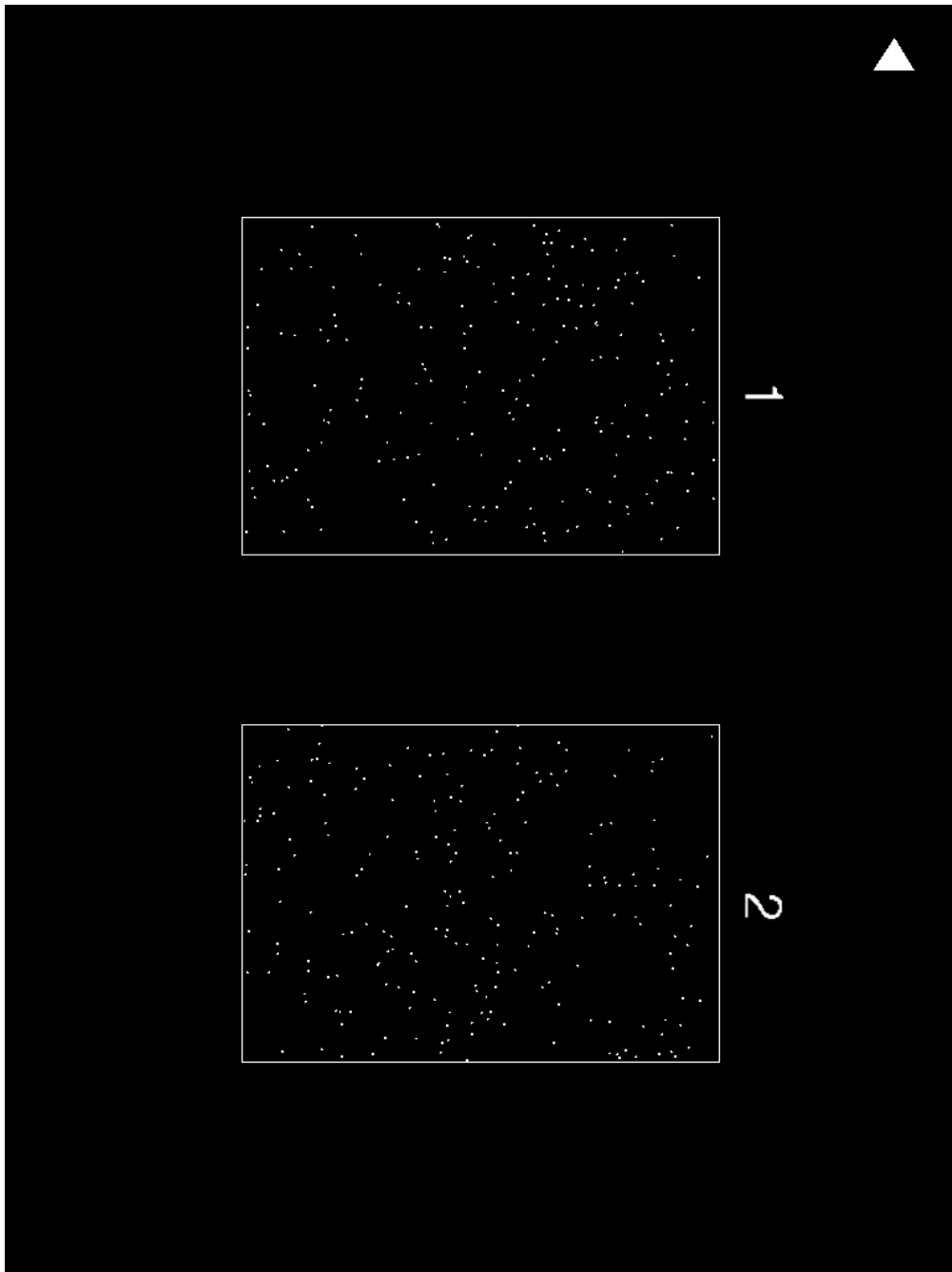


Figure G.30: Entered motion test.

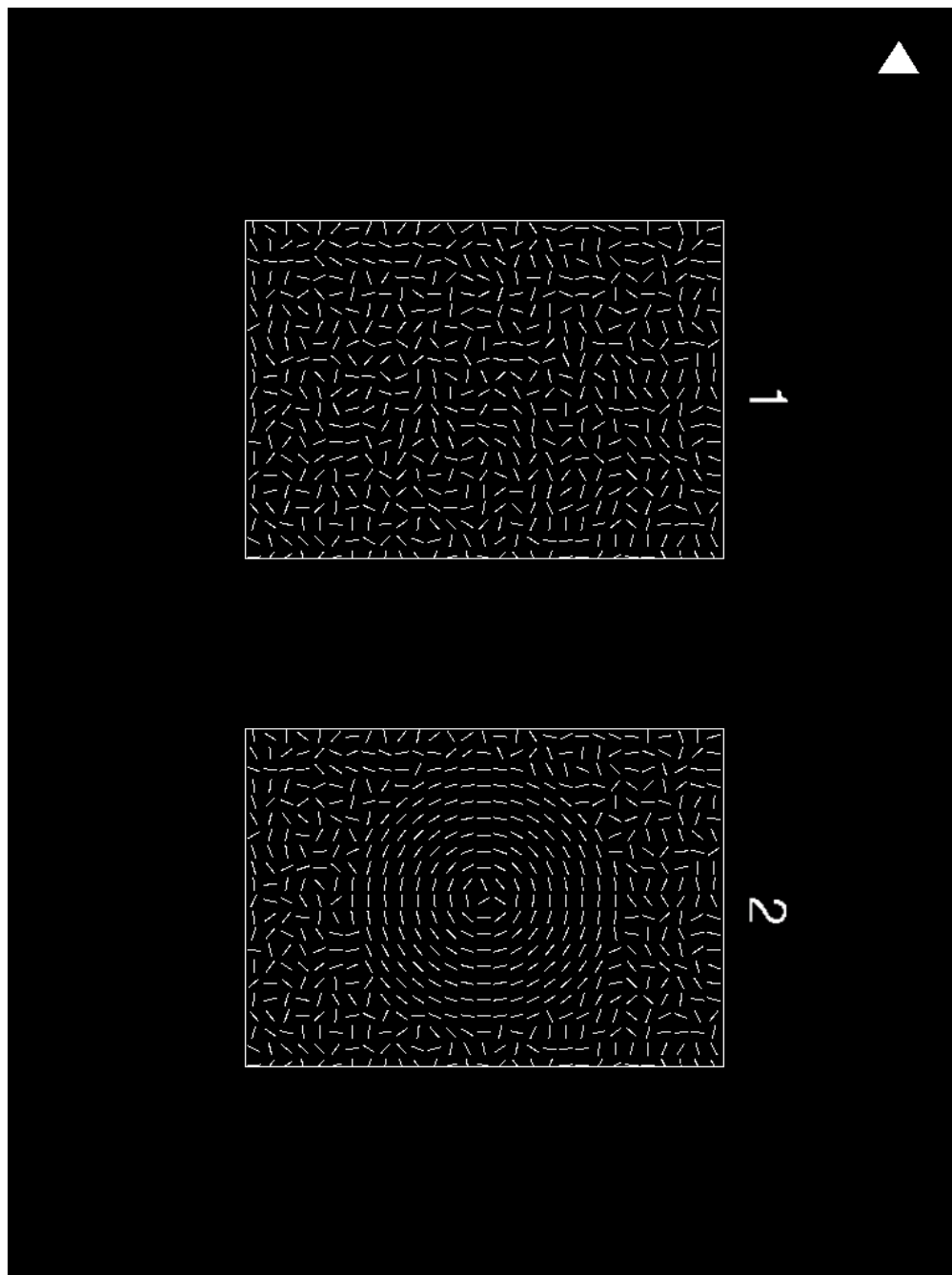


Figure G.31: Entered form fixed test.

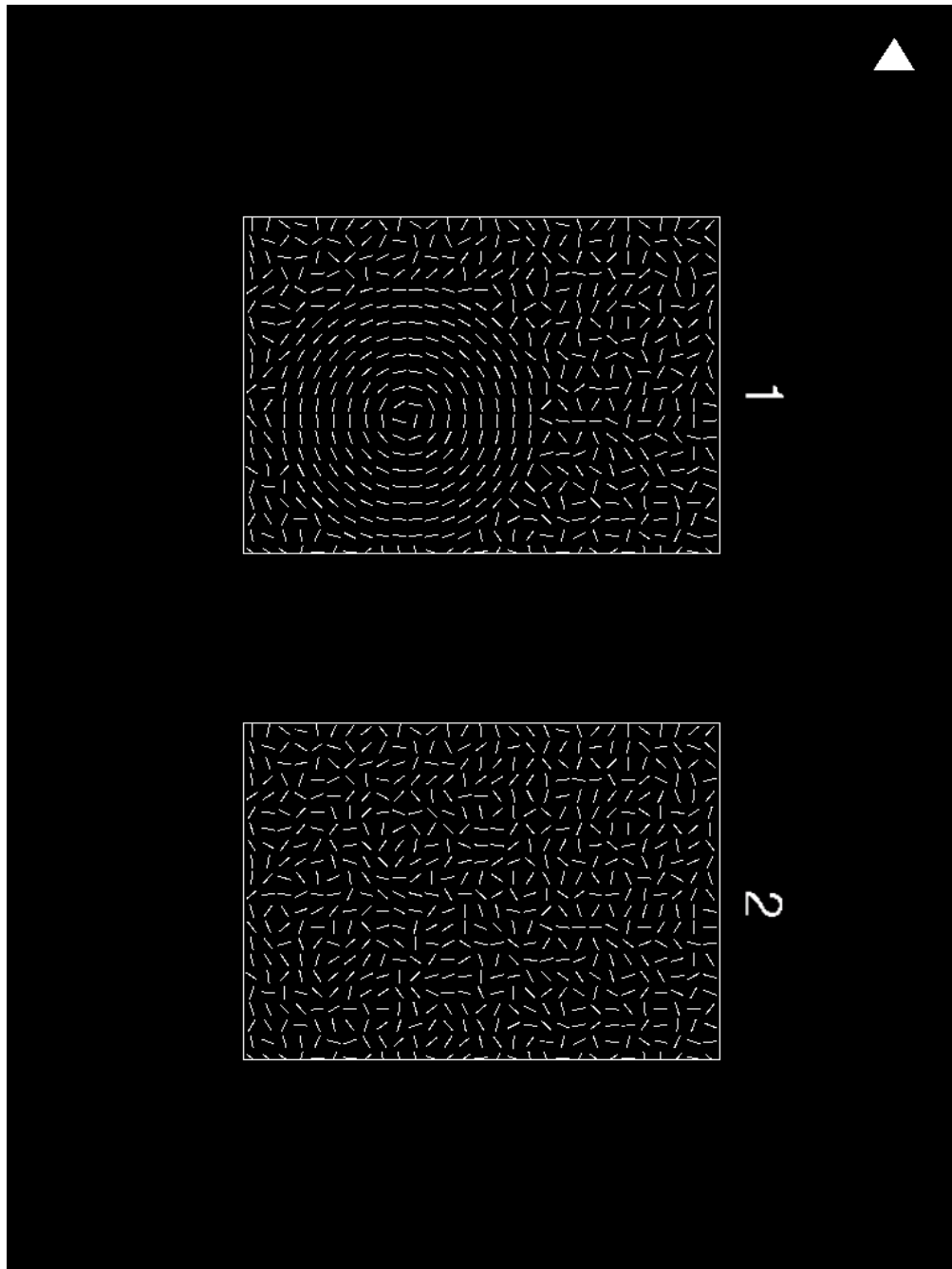


Figure G.32: Entered form fixed test.

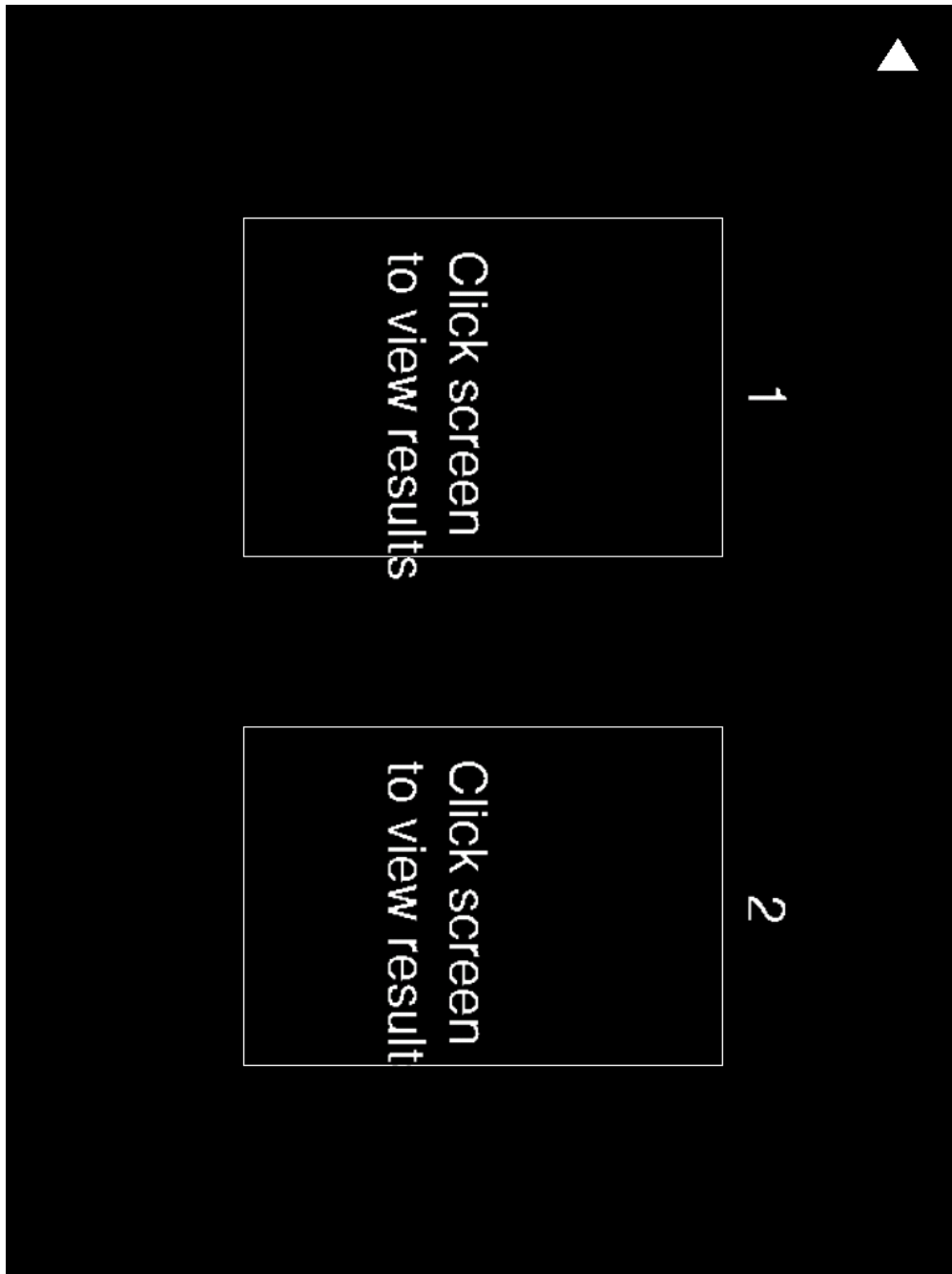


Figure G.33: Completed test.

G.4 Result Screen

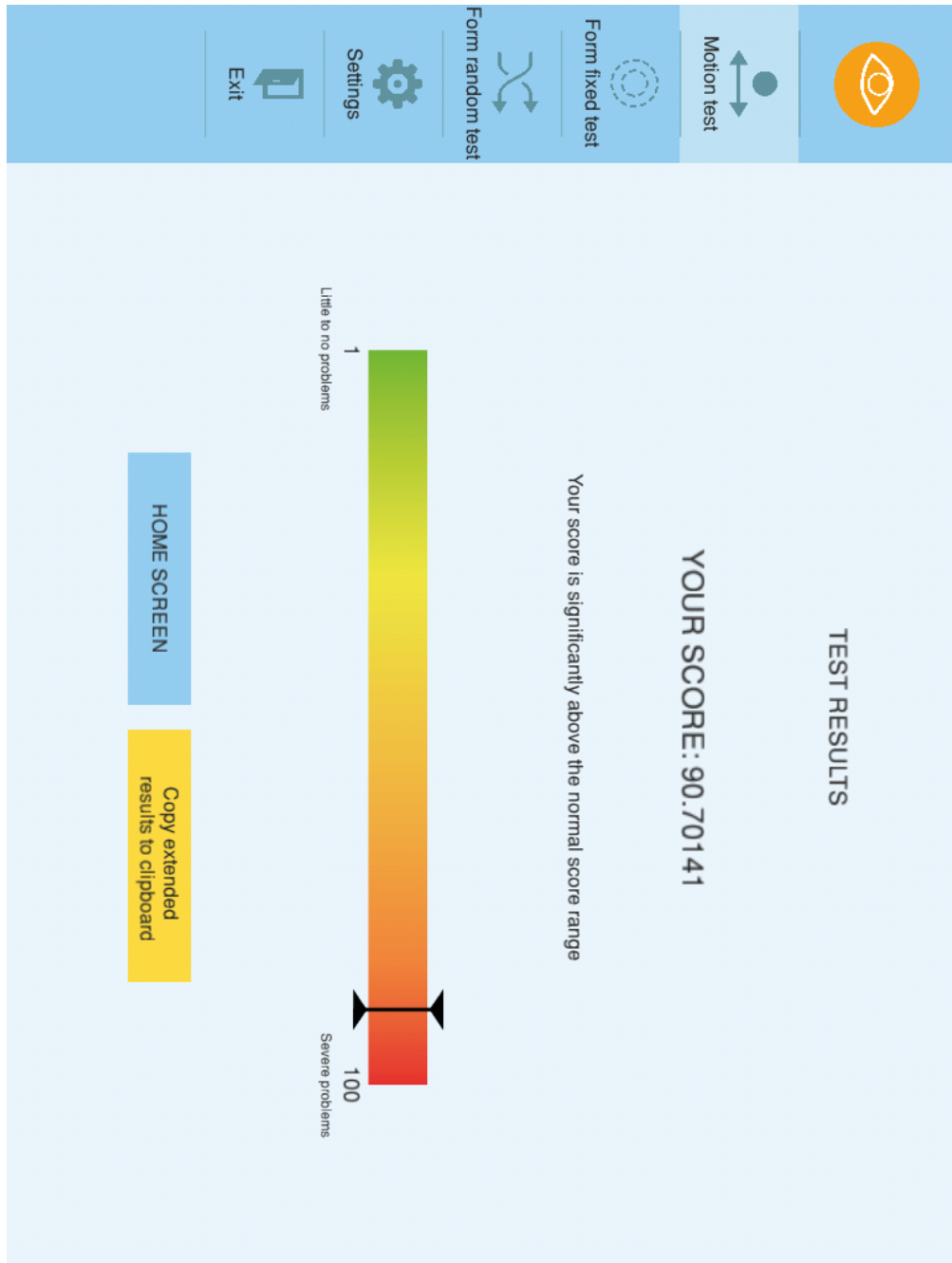


Figure G.34: Motion test result screen.

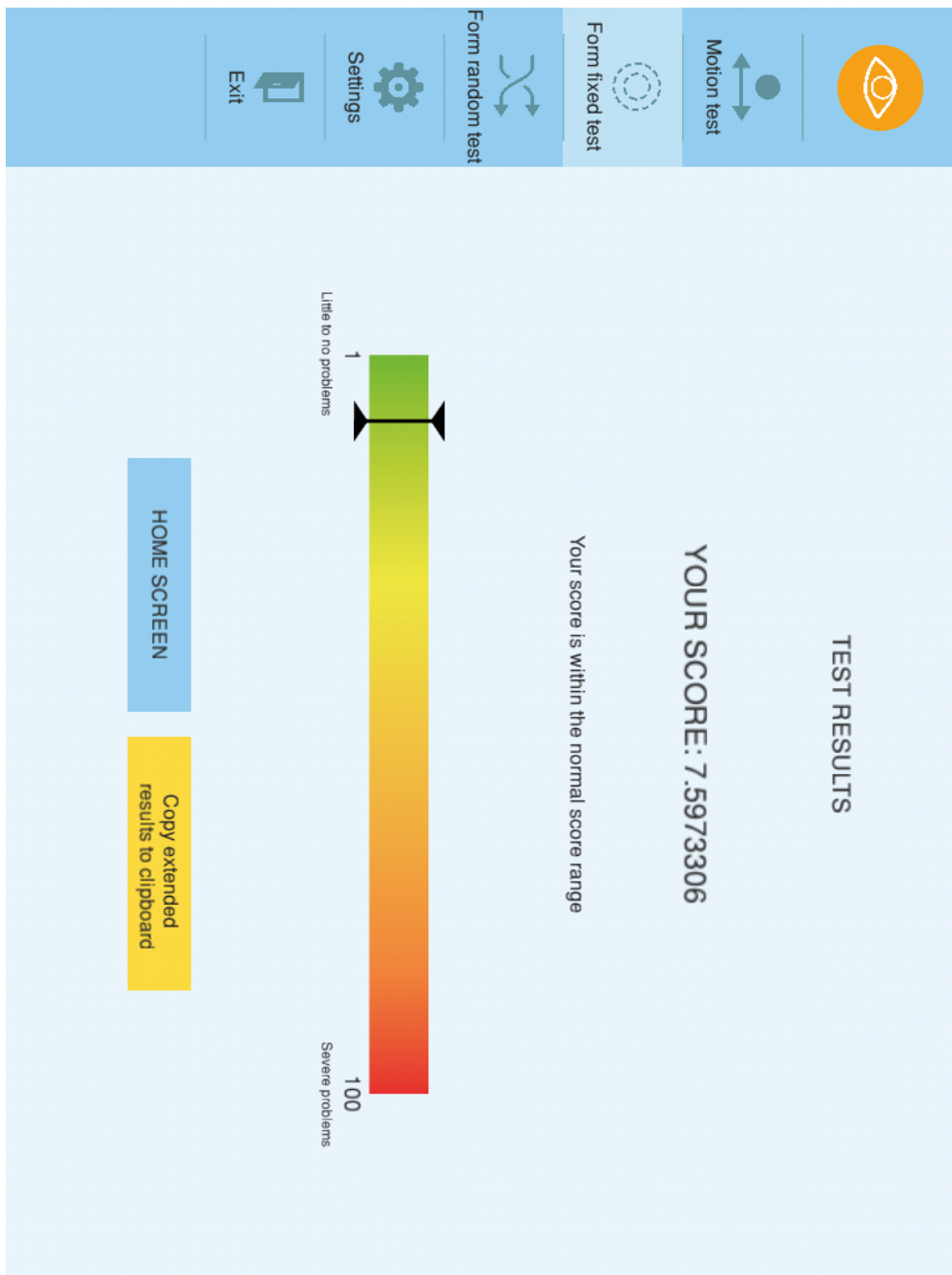


Figure G.35: Form fixed test result screen.

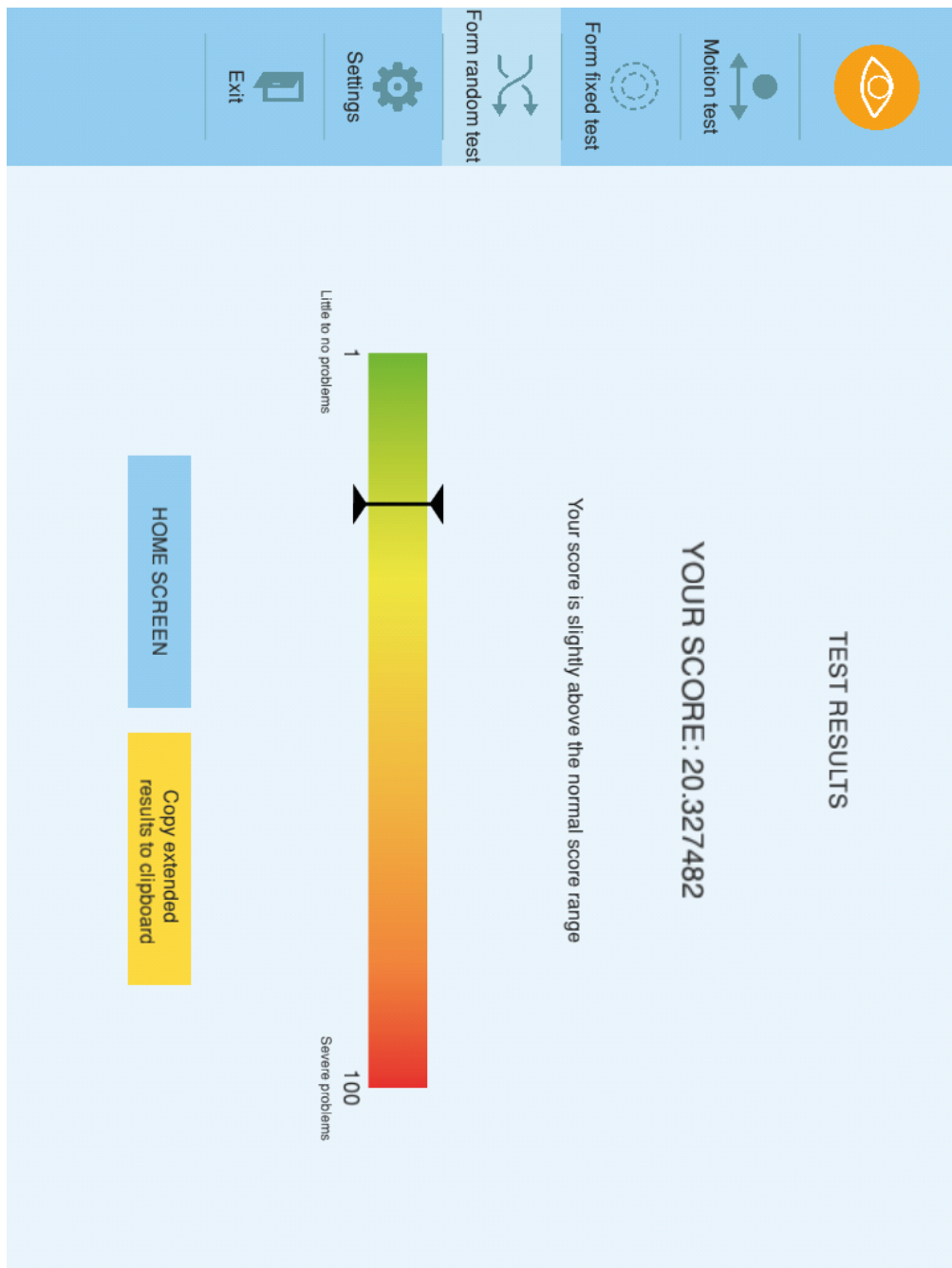


Figure G.36: Form random test result screen.

G.5 Settings

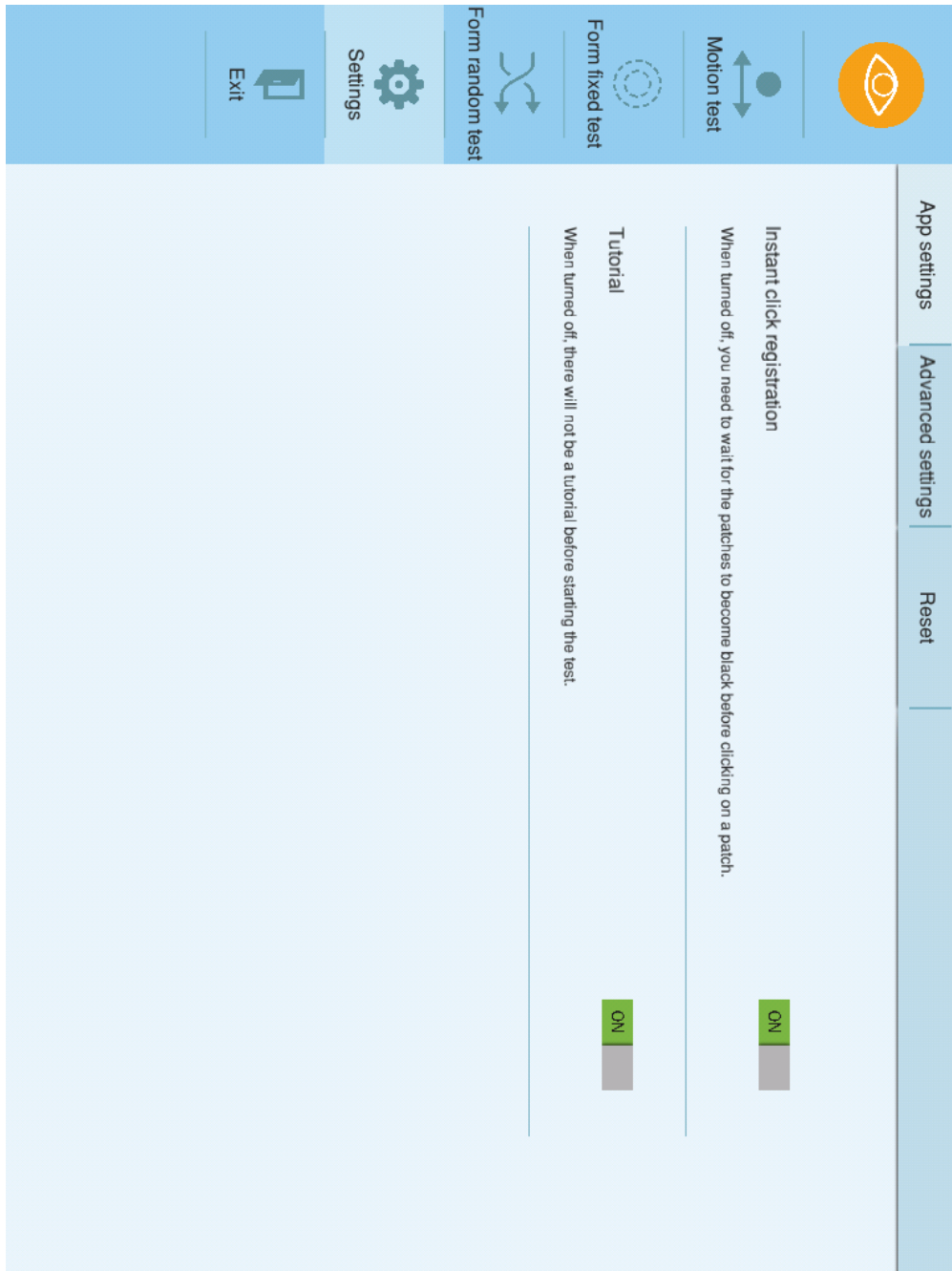


Figure G.37: App settings tab.



Figure G.38: Advanced settings tab.

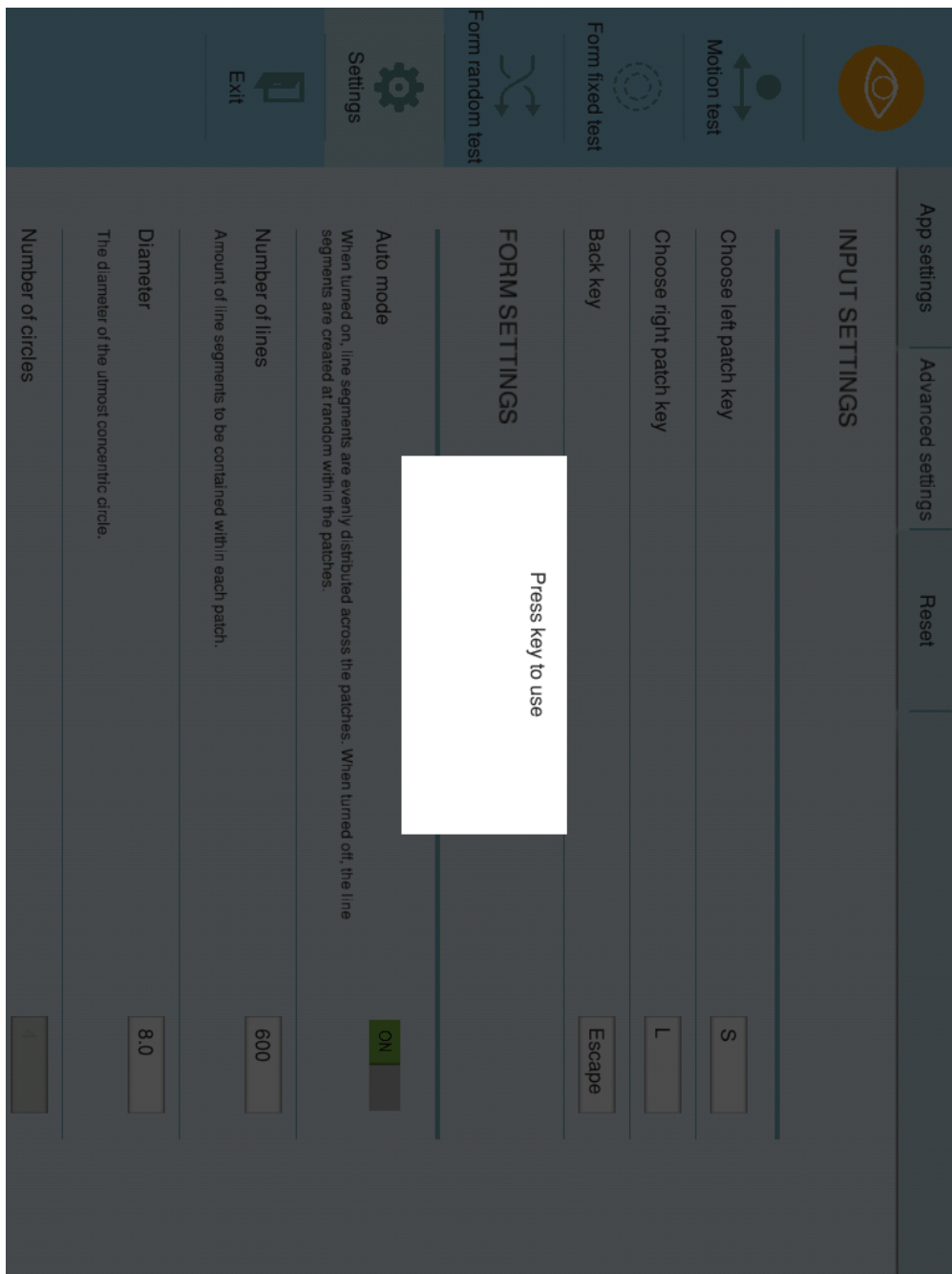


Figure G.39: Advanced settings - dialog window appears when changing left patch key, right patch key, or back key.

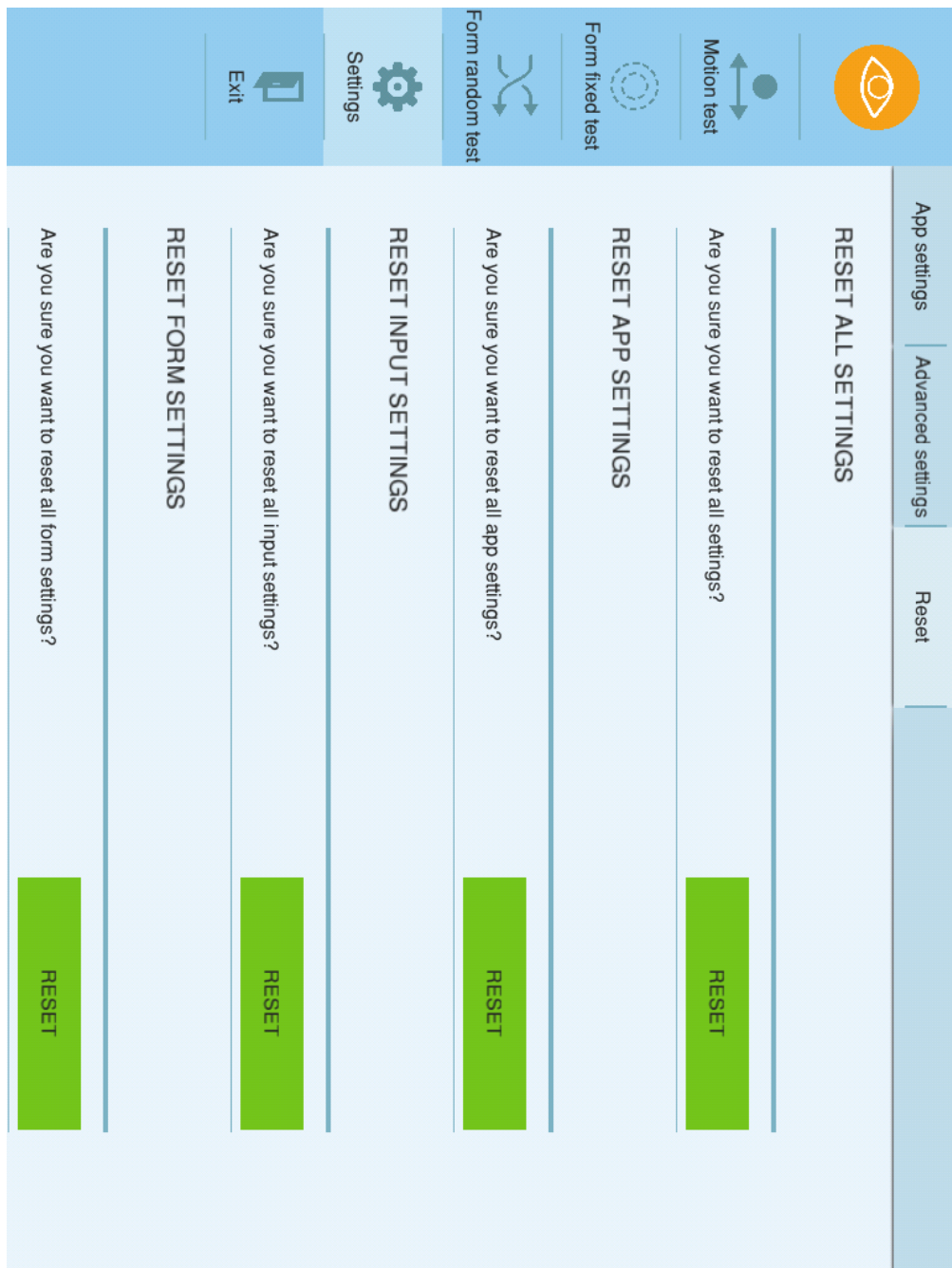


Figure G.40: Reset settings tab.