



Norwegian University of
Science and Technology

App for Early Detection of Dyslexia

Bjørnar Håkenstad Wold

Master of Science in Informatics

Submission date: April 2016

Supervisor: John Krogstie, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

As assessing dyslexia can be a time consuming and costly process, this thesis will look into the possibility of creating an application to be used as a screening tool. The application is based upon the work of Peter C Hansen, and will try to replicate the functionality of a program called Form. Information about the program is gathered through first hand knowledge from a previous user and research articles describing the testing process and test contents. During development, iterations of the tests is evaluated by an expert for feedback on the proposed implementation. For evaluating the finished product, test scores from a group of dyslexic and non-dyslexic is recorded.

The results from the preliminary evaluation shows a correlation between dyslexics and the test scores obtained for the coherent motion test. Where the dyslexic score on average 61% higher than the control group. For the coherent form tests, no distinct correlation between dyslexia and test score is prevalent.

For an initial evaluation, the results look promising. However, with the small sample pool of data, further evaluation with larger test groups is needed to assess the usefulness of the implemented solution.

Sammendrag

Siden utredning av dysleksi kan være en tidkrevende og kostbar prosess, ser denne oppgaven på muligheten for å lage en applikasjon som kan brukes til utredning. Applikasjonen baserer seg på tidligere forskning av Peter C Hansen, og vil forsøke å replisere funksjonaliteten til programmet Form. Innhenting av relevant materiale er gjort via samtaler med en tidligere bruker av programvaren, og ved hjelp av forskningsartikler som beskriver test prosedyrer og funksjonalitet. Gjennom utviklingsarbeidet av applikasjonen har en ekspert testet den foreslåtte implementasjonen og kommet med tilbakemeldinger som har ført til utbedringer. En evaluering av brukbarheten til applikasjonen ble foretatt ved å teste den på en gruppe dyslektikere og ikke-dyslektikere.

Testresultatene fra evalueringen viser en tydelig sammenheng mellom dysleksi og oppnådd test resultat for coherent motion testen. Dyslektikerne ligger på denne testen i snitt 61% høyere enn gruppen med ikke-dyslektikere. Dette gjenspeiler seg ikke i testene for coherent form, der det ikke finnes en tydelig sammenheng mellom dysleksi og oppnådd resultat.

Som en førstevurdering av applikasjonen ser testresultatene lovende ut. Men siden testgruppen er liten, vil det være nødvendig å gjennomføre ytterligere evalueringer av den implementerte løsningen med en større testgruppe.

Contents

List of Figures	v
List of Tables	vii
Preface	ix
Acronyms	xi
1 Introduction	1
1.1 Motivation	1
1.2 Research Goals	1
1.3 Research method	2
1.4 Report overview	3
2 Previous work	5
2.1 Psychophysics	5
2.2 Dyslexia	6
2.3 Diagnosing dyslexia	8
2.4 Tools and technologies	13
3 Implementation	15
3.1 Requirements	15
3.2 Application development on mobile devices	16
3.3 Software architecture	16
3.4 Application overview	23
4 Evaluation	41
4.1 Testing	41
4.2 Initial evaluation	46
5 Discussion, Conclusion and Further work	49
5.1 Discussion	49
5.2 Conclusion	51
5.3 Further work	52
Appendices	53

A	Appendix A	55
A.1	Installation	55
A.2	Test results	57
B	Appendix B	61
B.1	Magnofly Read Me.txt	61
B.2	Magnofly Parameters	63
C	Appendix C	67
	Bibliography	85

List of Figures

2.1	Relation between Weber's law and Fechner's law.	6
2.2	Demb et al. and Cornelissen et al. test results	9
2.3	Illustrations of Peters program	11
2.4	Visual angle	12
2.5	Magnofly game screen	13
3.1	Model view controller.	17
3.2	Class diagram.	18
3.3	Angle of reflection.	19
3.4	Quadtree.	20
3.5	Dot collision detection.	20
3.6	Staircase.	21
3.7	Coherency update sequence.	22
3.8	Main menu.	23
3.9	Motion test screen capture.	24
3.10	Motion test illustration.	25
3.11	Form fixed auto test 100%.	26
3.12	Form fixed auto test 50%.	27
3.13	Form fixed auto test 0%.	27
3.14	Form random auto test 100%.	28
3.15	Form random auto test 50%.	29
3.16	Form random auto test 0%.	29
3.17	Form fixed auto vs manual.	30
3.18	Form random auto vs manual.	31
3.19	Settings menu.	32
3.20	Motion test settings.	33
3.21	Form test settings.	35
3.22	Screen settings.	37
3.23	Screen settings apply.	37
3.24	Staircase settings.	38
3.25	Input settings.	39
3.26	Input settings key dialog.	40
3.27	Reset settings.	40

4.1	Motion test prototype 1	42
4.2	Motion test prototype 2	42
4.3	Motion test prototype 3	43
4.4	Form test prototype	43
4.5	Motion test prototype 4 screen settings	44
4.6	Motion test prototype 4 test settings	45
4.7	Motion test prototype 4	45
4.8	Motion test results	47
4.9	Form fixed auto test results	48
4.10	Form fixed manual test results	48
A.1	Test results screen	57

List of Tables

4.1	Test results	47
5.1	Random variation test.	50

Preface

The work in this thesis is a part of my masters degree in Informatics taken at the Norwegian University of Science and Technology. The project has been conducted by the Department of Computer and Information Science in cooperation with Professor Hermundur Sigmundsson. The work started in February 2015.

Finishing this project would not have been possible without the help from my supervisor, Professor John Krogstie. A big thank you for your patience and continues encouragement that have kept me working.

I would also like to thank Professor Hermundur Sigmundsson for his knowledge in the research field of dyslexia, and his shared knowledge about the original program.

I own a big thank you to my friends and family for their continuous support and help throughout the year.

A special thank you to my girlfriend Siw B. Reistadbakken for her support, care and motivation that have allowed me to focus on the project during stressful situations.

Bjørnar Håkenstad Wold
Oslo 13.04.2016

Acronyms

API	- Application Program Interface
App	- Application
CAD	- Computer-aided Design
IDE	- Integrated Development Environment
MS-DOS	- Microsoft Disk Operating System
MVC	- Model View Controller
RQ	- Research Question
YARC	- The York Assessment of Reading for Comprehension

Chapter 1

Introduction

This chapter contains the motivation for conducting this research, the research goals and research questions, and the methodology used during the research. Lastly an overview of the disposition and contents of the following chapters.

1.1 Motivation

Each fall excited children start their first year at school where they get the possibility to learn how to read and write among all the other subjects covered. Some can already read and write simple words or sentences while others haven't gotten quite there yet. Most of the children will learn fast, but there exists those who will continue to struggle with letters and words as long as they are alive. With an estimated 5-17% of the population facing these difficulties it is important to give them an early assessment that can prevent them from falling behind their classmates [1]. The screening tests already in use are often not available until school age, and it might be hard for teachers to figure out who needs to take them.

Research done by P C Hansen has proven that software can be used as a screening tool [2], by using specifically designed software to check for visual problems. The software was written to work on hardware and software dependencies available at the time. Computer software and hardware have seen rapid improvements over the last 15 years, creating the need for old software to either be rewritten or completely replaced. This is especially true when it comes to modern platforms that smart phones and tablets have created.

1.2 Research Goals

The goal for this project is to develop an application that can be used as screening tool for assessing dyslexia. The implementation will try to recreate the tests from the program Hansen developed for MS-DOS. Without having access to the program, the work will be based on information gathered from a previous user, and on articles describing the tests and test procedures.

The implementation will go through iterations where the application is sent to the previous user for feedback. The feedback will be used to make changes to the implementation, adding functionality when needed. If or when the feedback deems the implementation good enough, an evaluation will be performed where the application is tested on a group of non-dyslexics and dyslexics.

Research questions

RQ 1: How is the procedure for recreating a DOS-program to the modern tablet format?

RQ 1.1: What kind of information is needed?

RQ 1.2: How can information about the older software be obtained?

RQ 2: Do the test scores correlate with dyslexia?

RQ 2.1: Can the application be used for detecting dyslexia?

1.3 Research method

The method for this paper is separated into two parts, one for information gathering and one for evaluation of the implemented solution.

Information gathering

A literature review was chosen for the initial research, to provide a basic understanding of dyslexia, and the functionality of currently available screening tools. The literature will be obtained through the use of search engines for scientific articles:

ACM Library

Google scholar

Google books

IEEE Xplore

Interviews are to be conducted with a previous user of the program Hansen developed. The interviews will be unstructured with the aim of gathering relevant information about the test behaviour and the visual elements.

Information obtained from other sources, i.e regular search engines, is cross checked with findings from the above sources.

Evaluation

During the implementation phase, an expert will provide the necessary feedback for further work. This will involve sending working copies of the implementation in form of prototypes to be tested. The prototypes are going to be stand alone applications for each of the tests, with the possibility of altering test parameters on the fly.

A test group is to be set up, consisting of a group of known dyslexic and non-dyslexic to perform an initial evaluation of the implemented solution. The data from the initial evaluation will be used to evaluate the usefulness of the application.

1.4 Report overview

Chapter two in this paper starts with gathering information about the research field surrounding reading and writing difficulties, and the measures that exists to assess the problem. Chapter three explains the development process, from deciding on functional requirements to the implementation. Chapter four describes the testing and evaluation process of the application. Chapter five discusses the results from the testing and evaluation process, giving a conclusion and suggestions for future work.

Chapter 2

Previous work

The link between a brain deficit and problems with reading and writing is the key to be able to create software that tests for the deficit. This chapter aims to give a short overview of the relevant research regarding the deficit, how it is linked to reading and writing problems, and of existing testing methods. It will also give an introduction to the tools and technologies used during the implementation phase.

2.1 Psychophysics

Sensory threshold is central concept in the study of psychophysics, where the idea is that for events to be processed consciously they have to be stronger than some threshold. This absolute threshold represents the weakest, or smallest, amount of stimulus energy needed to create a sensation. As the threshold tends to fluctuate for each individual, it is necessary to conduct a series of tests to determine the absolute threshold. When the stimuli is above the absolute threshold, the difference threshold defines a just noticeable change in the stimuli. If stimuli is applied at 10 units and the change is only noticeable at 16 units, the difference threshold would be 6 units. The stimuli can be applied to one or more of the four dimmensions; intensity, quality, extension or duration, to cause a change in the sensation [3].

German physiologist E. H. Weber researched how much the intensity needed to change at a given level for it to be noticeable. By using weight of varying mass placed on the skin, he was able to figure out how large the difference in mass had to be for the sensation to be noticeable different. Lighter weights could only differ very little before it was noticeable, while with heavier weights the difference in mass had to be larger. From his observations he described Weber's law¹ to provide a mean for measuring the correlation between the difference threshold and the stimulus intensity [3].

¹Weber's law: $\delta\phi = c\phi$ Where $\delta\phi$ is the just noticeable change in intensity and c is a constant fraction of the intensity at the start of the stimuli ϕ . Important to note is that its not possible to compare two different stimuli with this function.

By building upon Weber's work, Gustav T. Fechner theorised a new field of research he named Psychophysics [4]. Having a background in mathematics and physics Fechner approached the problem in a different way than those before him. Seeking a precise way he could describe a private experience with numbers, he focused on the idea that mind and matter are equal. Fechner divided it into stimulus $\delta\phi$ (from Weber) and sensation ψ , thus splitting it into two individual dimensions [3].

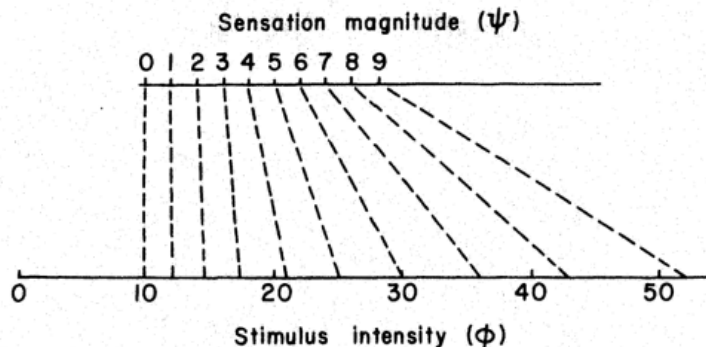


Figure 2.1: Relation between Weber's law and Fechner's law. Stimulus values that are marked off according to Weber's law were assumed by Fechner to result in equal steps in sensation magnitude.

Weber's law have never been verified to be correct, subsequently neither have Fechner's. However, their work gave way to the modern field of Psychophysics by theorising means to measure how external stimuli can be quantified as an internal sensation in the form of thresholds. This method for measuring performance can be used for an assessment of reading and writing problems.

2.2 Dyslexia

The terms reading disability and dyslexia are used interchangeable, and covers difficulties with reading, writing and spelling [5]. As each person afflicted with dyslexia is different from the others, a clear definition of the term is not present [6], [7]. Each definition focuses on a different concept than others, Høien and Lundberg defined it in 1991 as:

Dyslexia is a disturbance in certain language functions which are important for using the alphabetic principle in the decoding of language. The disturbance first appears as a difficulty in obtain automatic word decoding in the reading process. The disturbance is also revealed in poor writing ability. The dyslexic disturbance is generally passed on in families and one can suppose that a genetic disposition underlies the condition. Another characteristic of dyslexia is that the disturbance is

persistent. Even though reading ability can eventually reach an acceptable performance level, poor writing skills most often remain. With a more thorough testing of the phonological abilities, one finds that weakness in this area often persists into adulthood.

while in 1994 the American working definition worked out by the Orton Dyslexia Society Research Committee, in cooperation with the National Center for Learning Disabilities and researchers from the National Institute of Child Health and Human Development, was:

Dyslexia is one of several distinct learning disabilities. It is a specific, language-based disorder of constitutional origin characterized by difficulties in single word decoding, usually reflecting insufficient phonological abilities. These difficulties in a single word decoding are often unexpected in relation to age and to other cognitive and academic abilities; they are not the result of generalised developmental disability or sensory impairment. Dyslexia is manifested by variable difficulty with different forms of language, often including, in addition to problems with reading, a conspicuous problem with acquiring proficiency in writing and spelling.

In recent years the definition has somewhat changed and in 2002 the below definition was adopted by National Institute of Child Health and Human Development among others [8]:

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

while in 2007 the British Dyslexia Association approved the following definition [9]:

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.

For those interested, Rice and Brooks lists seven definitions of dyslexia in their research [5]. The above definitions have similarities but they are not equal, with the general concise being that a brain deficit is the cause of why people with dyslexia have difficulties with decoding text to spoken words, and transferring spoken words into text [5], [6], [10]–[12].

Magno- and parvo-cellular systems

The deficit have been linked to the way visual information is transferred from the eye to the brain [13]. Two pathways, called magnocellular and parvocellular, transfers information to visual cortex and then to the dorsal and ventral streams in the brain. The magnocellular pathway transfer information about luminance, motion and specialises in localization of objects. Fine details, colours and recognition of objects are transferred along the parvocellular pathway [14].

Psychophysical experiments have been derived from the theory that there exists a deficit in the magnocellular pathway indicating dyslexia. During a research on post mortem brains, Livingstone et al. [15] found the parvocells to be similar between non-dyslexics and dyslexics. With the magnocells being smaller and seemed disorganised in the dyslexic brain, compared to a non-dyslexic brain. Studies that have tested for contrast sensitivity² have not found conclusive evidence to this theory [13]. Instead of using contrast sensitivity, Cornelissen et al. [16] opted to use random dot kinematograms³ to check for motion discrimination among dyslexic. Finding the dyslectics detection threshold were 3-4% higher on average compared to the control group, thus suggesting there is a deficit in the magnocellular pathway. Demb et al. [17] used both methods finding small differences in contrast sensitivity while the difference in motion detection was significant compared to a non-dyslexic control group further asserting a magnocellular deficit.

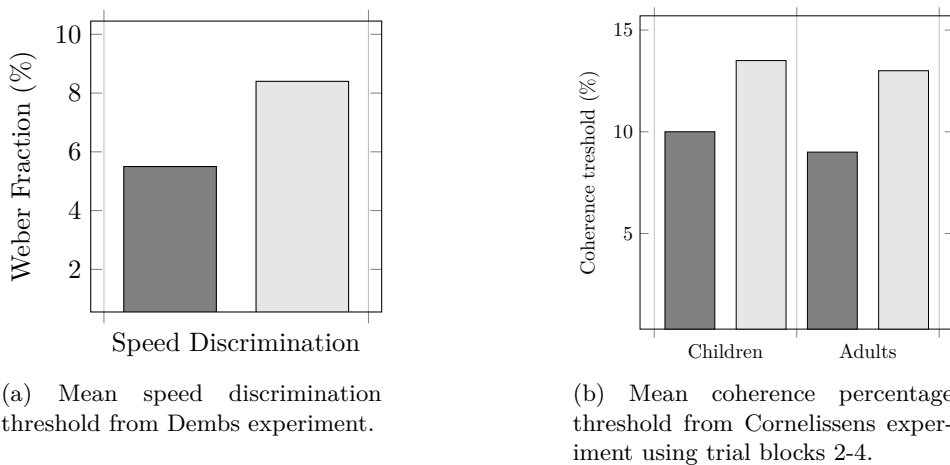
2.3 Diagnosing dyslexia

Diagnosing dyslexia can be a tedious and expensive procedure. From the moment a parent or other caretaker⁴ notice a problem. A series of doctor appointments and assessments tests follows before a diagnosis is given [1]. As assessments tests normally is not available until school age [11], there exists guidelines for mapping a child's development from an early age [1]. Possible warning signs could be difficulties with slow language acquisition, problems with understanding spoken instructions, and difficulties with recognizing his or hers written name [11]. Important to note is that these are only possible signs, and only a specialist can provide a diagnosis.

²Slowly increasing the contrast of an image or text until an observer says they can see it.

³Simplified: random dot kinematograms display small dots on two panes. On one pane a percentage of the dots moves coherently, while on the other pane the dots move at random. The observer is to point out the one with most coherency.

⁴Teachers and the like



(a) Mean speed discrimination threshold from Dembs experiment.

(b) Mean coherence percentage threshold from Cornelissens experiment using trial blocks 2-4.

Figure 2.2: Darker bars represent control groups while the lighter ones represents the dyslexic group.

Assessment tests

Evaluating a person's ability to read is done through language specific assessment tests. These standardized reading tests, measures reading speed, accuracy, fluency and comprehension.

York Assessment of Reading for Comprehension (YARC)

The English YARC assessment consists of three tests for different age groups. The tests are administered by the school, and is initially used to assess a group of pupils. Individual tests are used to follow up the progress of pupils. The information gathered from the tests are used to identify reading problems, and the need for extra tutoring and adaptation [18].

From ages 5-7 the Early Reading test assesses the pupils ability to: correctly sound letters, recognize words, and phoneme awareness.

The YARC Passage Reading for use in primary school, assesses the progress of a pupil trying to identify any reading problems. Two paragraphs are read aloud, with a set of eight comprehension questions to assess literal and inferential comprehension. The test can be given to pupils from 5 to 12 years.

The YARC Passage Reading test for use in secondary school, ages 12 to 18. The test consists of two paragraphs of fiction and non-fiction reading. It is a follow up assessment for monitoring a pupils progress in reading comprehension and fluency.

Språk 6 - 16 screeningtest

The Norwegian Språk⁵ 6-16 screeningtest consists of three obligatory tests and a number of supplement tests. Administered by the schools, the test aims to check if a pupils reading level is adequate for the age group, and if the pupil needs to be referred to a specialist for further assessment [19].

Ordspenn⁶ is considered a good test for assessing the phonological short time memory, testing the pupils ability to retain the sound structure of words [19]. Three words are read aloud and the pupil is to repeat the words back. The test giver writes down the order of the repeated words, and gives a score of 1 if they are in the correct order, and a 0 if one or more of the words are in the wrong order [20]

Setningsminne⁷ assesses the pupils ability to organize and retain information in sentences [19]. In this test, a sentence is read aloud and the pupil is instructed to repeat it back. A score of 1 is given if the pupil repeats the sentence correctly, and a score of 0 if there are 1 or more errors [20].

Begreper⁸ assesses the pupils semantic understanding of words [19]. There are two parts in this test. For the first part, the pupil is to find the opposite of a word read aloud. If the word is “start”, the correct answer would be “stop”. In the second part, the pupils understanding of concepts are tested. The test giver is to ask questions similar to “What is an umbrella?” or “What does loyalty mean?”. A score of 1 is given for each correct answer, and a 0 for answers that don’t fit. [20].

Wordchains test

The Wordchains test consists of 90 word chains written in the following format: *boatfishcathome beeralewhiskyscotch*. The test taker is to set three lines in each word chain to divide them into four words, within the allocated time of four minutes. His or hers score is equal to the amount of correct answers, a maximum of 90. The test is quick way of measuring the speed and accuracy of word recognition [21].

Custom software

Purposely made software have been shown to be successful at finding a correlation between dyslexia and the score obtained from visual stimuli tests. The tests consists of finding coherent motion or coherent form in elements displayed on a monitor.

Form by Peter C. Hansen

This old MS-DOS program consists of three testing methods for identification of dyslexia; a random dot kinematogram testing for motion discrepancy; static global pattern processing with a randomised target and; static global pattern processing with a fixed target.

⁵Language

⁶Word span

⁷Sentence memory

⁸Concepts

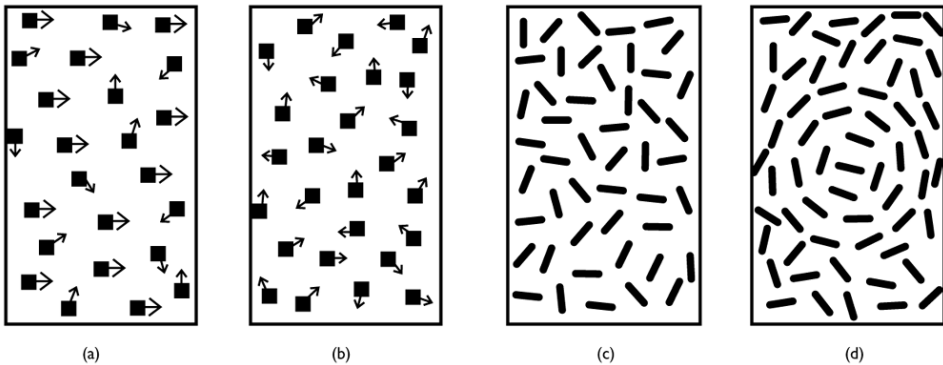


Figure 2.3: A and B of coherent motion, C and D of coherent form. Illustrations are not to scale and the colours are inverted.

The test methods determines the detection threshold by using an adaptive staircase procedure. For each correct answer the coherence level is lowered by 1dB, while for each wrong answer the coherence is increased by 3dB. The test will end after 10 reversal points has been reached. The last 8 reversal points are used to calculate the geometric mean for a given test. After two runs of a test, the average of the two test scores are defined as the detection threshold [2], [22], [23].

The random dot kinematogram consists of two patches populated by 300 high luminescent white dots each. At the start of a test procedure one patch is chosen at random to contain only randomly moving dots, while the other patch will contain a given percent of coherently moving (rightwards or leftwards with angular velocity of $7^\circ/\text{s}$) dots with the rest moving at random in a Brownian manner⁹. To prevent tracking of individual dots, each one has a lifetime of four frames. At end of life it is removed, before it is redrawn at a random location within the same patch on the next frame. The coherency level is started at the highest possible percentage and subsequently lowered until the detection threshold is found. Each stimulus run has a running time of four seconds, during which a test subject is to inspect both patches and point out the one with coherent motion [2], [22], [23].

The static global pattern controls, Form-Fixed and Form-Random, consists of 600 line segments in each pane. They are oriented in concentric circles and in random orientations. The line segments forming circles are placed at the tangent of an imagined circle. As with the random dot kinematogram test, one patch holds the target (circles) and noise while the other one consists of noise only. The coherency starts at 75 percent. In Form-Fixed the circle can only appear in the middle of the patch, while in Form-Random it can appear anywhere as long as the circles fits within the patch. Opposed to Form-Random, Form-Fixed has a time limit of four seconds for finding the target [2], [22], [23].

⁹Random movement of particles in fluid

Subjects took the test in a darkened room, wearing their normal optical correction. Each patch occupied $10^\circ \times 14^\circ$, horizontally separated by 5° , of screen area at a viewing distance of 57cm. The patches were marked with notes above the screen with “one” and “two” written on them. Subjects were to call out either “one” or “two” during the four second interval, with the aim of choosing the patch with the target. An experimenter pressed a key on a keyboard corresponding to either “one” or “two” after the pattern disappeared, initializing a new interval [2], [22], [23].

Visual angle

A psychophysical method involving visual angle is used to calculate the size of objects on the screen. The further away you get from a screen, the smaller anything on it will seem. This is due to the visual angle getting smaller at greater distances. In psychophysical visual tests this is accounted by calculating the visual angle at a given distance. To do this the screen size, resolution and distance between the observer and the screen is needed. By describing objects on the screen with this method it is possible to make tests that will be perceived as the same regardless of the screen it is displayed on [24].

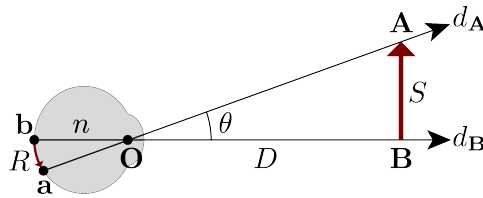


Figure 2.4: Using $V = 2\arctan(S/2D)$ to calculate the radian angle for an object where S is the height of the screen, D is the distance between the screen and the pupil O [25].

Random dot kinematogram

Cornelissen et al. [16] conducted an experiment by using a random dot kinematogram consisting of 1056 dots in each patch, displayed against a dark background. At 0% coherent motion, the dots were to be replaced every 20 millisecond giving the illusion of white noise. By increasing the coherent motion percentage, a proportion of the dots would start moving in a coherent direction. One of the patches were chosen at random at the beginning of each interval, to be separated into three sections. The dots in the middle section moved the opposite direction of the top and bottom sections.

Starting at a 100% coherent motion, the test takers were to identify the sectioned patch. Coherent motion was lowered by 1.5dB for each correct answer until an incorrect patch was identified and the coherency was increased by 3dB. The threshold for each participant was calculated by taking the mean of the 6 last reversal points.

Magnofly

Magnofly is a computer game used to test for magnocellular motion sensitivity in children. The game screen have four patches with Magnoflies moving at random. The Magnoflies in a singular patch, chosen at random, will start to move towards a baby. The player is to identify this patch and spray they Magnoflies to prevent them from attacking the baby. Points are added if the player prevents an attack, subtracted if the baby is attacked or the spray is used carelessly [26], [27]. A background process using the adaptive Bayesian threshold method called QUEST [28] finds the percentage of coherent moving Magnoflies low enough to not be distinguished from the random moving Magnoflies. Appendix B contains the limited amount of information that was found about Magnofly.



Figure 2.5: The game screen of Magnofly.

2.4 Tools and technologies

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

Android Studio

Android Studio is a well documented IDE based on the IntelliJ IDEA, that offers Android specific tools to increase productivity while developing Android applications. It is distributed with templates for creating activities and with the possibility to import samples from the Google Samples repository. With the included graphical user interface 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 [29].

OpenGL—ES

OpenGL for embedded systems is a free cross-platform API specifically designed to work on systems without a desktop environment such as phones. It uses a subset of the OpenGL standard to create a low-level interface between the software and hardware layer, specifically the graphic unit processor, to ensure reliable high performance. As OpenGL ES is compatible with the standard OpenGL libraries, it possible to run OpenGL ES applications on almost any system that supports it without modifications. The standard support 2D and 3D objects and can be used to display CAD drawings, medical images and virtual reality environments [30].

Libgdx

Libgdx is a Java based framework providing cross-platform tools to make games. It supports low-level access to file system, 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 [31].

Packr

Packr adds the possibility to package Libgdx applications with a standalone Java version and a native executable for Linux, Mac or Windows [32].

Adobe Creative Cloud

Adobe Creative Cloud is a collection of creative tools, including Photoshop, Illustrator and InDesign among others. Photoshop is an easy to use graphics editor with powerful editing tools; it will be used to edit screen captures and elements for the user interface. Illustrator uses vectors, instead of rasters like Photoshop, and is ideal to create quick flow charts and models without having to worry about pixelation when resizing objects. InDesign is a publishing tool from Adobe, where it is possible to create flyers, posters, documents etc. with full control over the layout.

Chapter 3

Implementation

From coming up with good software requirements to having a completed piece of software, there are many choices to be made. This chapter aims to show and explain some of those choices, alongside explanations of solutions that were used to achieve the final implementation.

3.1 Requirements

Deciding on the application requirements were done with the help of first hand knowledge from an expert about the original program, along with the articles describing the program and the test procedures written by Hansen et al. [2] and Sigmundsson et al. [22], [23].

Functional requirements

FR1 - High

The application have to use screen size, screen resolution and viewing distance to calculate the size of objects on the screen.

FR2 - High

It should be possible to tune the behaviour of the tests through a settings screen.

FR3 - Medium

Due to the number of setting parameters, default values are to be provided.

FR4 - High

Settings are to be stored for the next time the application runs.

FR5 - High

The application should calculate a threshold score after a successful test run.

FR6 - Low

Test results along with settings used during a test should be possible to share.

Non functional requirements

NFR1 - Medium

The application should be able to run close to 60 frames per second on any device.

NFR2 - Medium

A comprehensive user guide should be made, describing test procedures and settings.

3.2 Application development on mobile devices

Application development in its current form is relatively new. Before Apple made the iPhone not many phones supported installing extra software. The phones that did support this only had a few applications available and they were mainly produced by the phone companies or network operators. When Apple launched the iPhone in 2008, the App Store provided a centralized way for developers to distribute and promote their applications. With the ease of distribution and a good business model, a surge of developers started to create applications. Google shortly followed with their, now named, Google Play Store and today these two companies stand for 2/3 of the application market [33], [34].

With two large ecosystems¹ to choose from, accessibility and familiarity played a big role. By choosing Android, it was possible to focus on app-development without the need to learn a new coding syntax. The .apk format Android applications are packaged in, makes for easier distribution compared to iOS[35]. This will provide for easier distribution to the testers, as the application can be sent through email or made available for download from the web.

3.3 Software architecture

Creating menu and setting screens were done by taking advantage of the features built into Libgdx. The framework 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 text fields and buttons.

A model view controller pattern, see figure 3.1, was chosen for the Motion and Form tests, as this allows for better separation of the logic and user interface. The Model keeps track of the current state and contains the functionality related to a test. Tests are rendered with the View, and together with the Controller, make out the user interface [36].

Classes

Menus and setting screens have not been included in this section as they provide little functionality. The class diagram can be seen in figure 3.2.

¹and many small

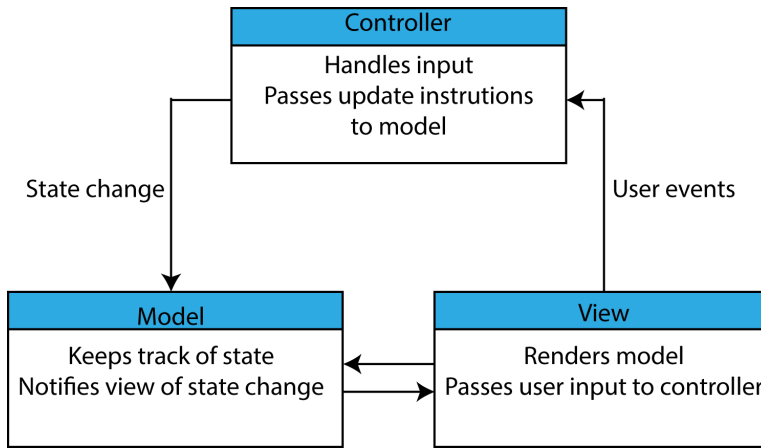


Figure 3.1: A visual representation of the model view controller pattern.

The *Assetloader* class function is to load sprites, textures and fonts into memory when the application starts. When the application closes, it disposes of the objects to free up memory.

The *Settings* class stores all setting parameters in a hash map. The settings are grouped into; Motion settings, Form settings, Screen settings, Staircase settings, and Input settings.

The *Psychophysics* helper class contains the methods for calculating visual angle, converting dB to amplitude ration, and geometric mean.

The *Dot* class describes how a dot should look and behave. Size, radius, direction of travel, velocity, and how long a dot has been alive is kept track of.

With the objects in the Form tests being static, the *LineSegment* class only contains information about how the line segments should look. This includes length, height and angle relative to the centre of the line segment.

The *World* interface defines the methods needed for updating coherency and keeping track of the test states. The methods are implemented in the *MotionWorld* and *FormWorld* classes.

The *Screen*, *World* and *Renderer* classes makes up the mvc-pattern for the tests. The *Screen* class is the controller, handles user events, and notifies the *World* class of any changes. Specifying the functionality and behaviour of the tests takes place in the model (*World*). While the *Renderer* class renders the objects specified in the model.

The *Results* class formats the extended test results into a JSON object. For more information regarding the extended test results, see appendix A.

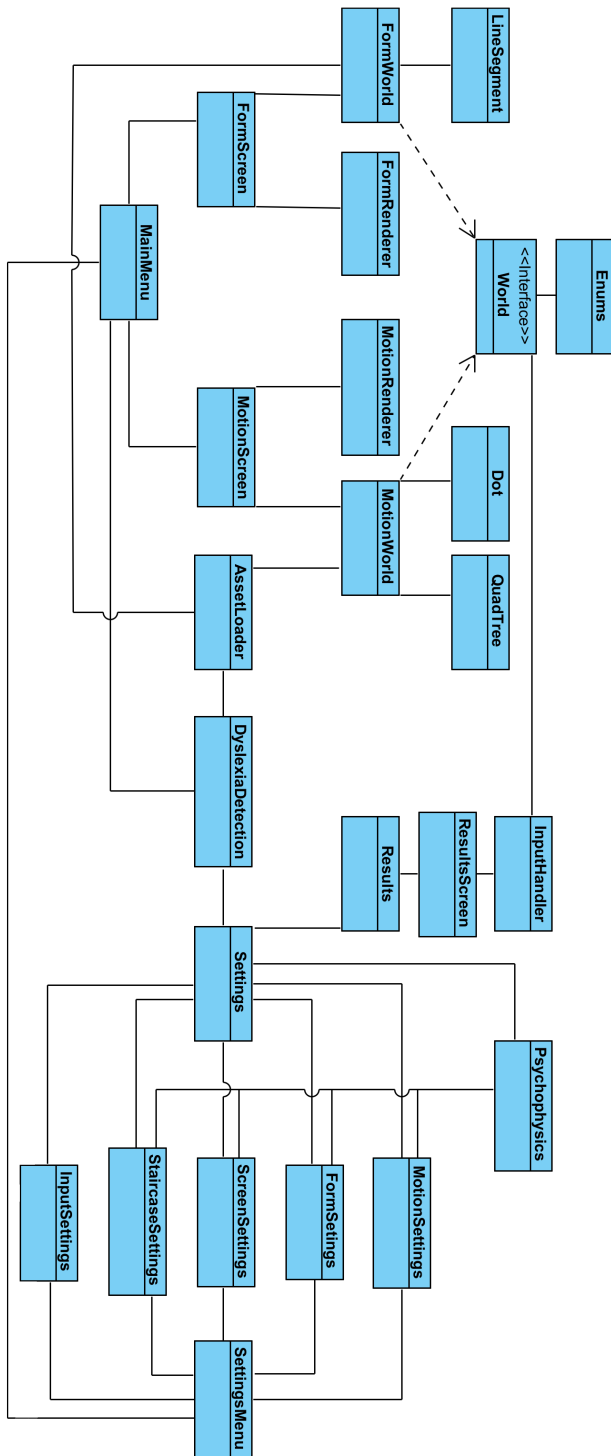


Figure 3.2: Class diagram.

Dot collision

For each frame update, each dot in the Motion test checks for two collisions: wall collision and dot collision. The dot velocity is to stay constant after a collision, and follows the principle of elastic collisions of equal mass objects [37].

For the wall collision, it is enough to check if the xy-coordinate of the dot is equal to, or larger than the patch boundaries. The law of reflection states that the angle of incidence is equal to the angle of reflection, see figure 3.3. Inverting the x velocity for vertical wall collisions, and the y velocity for horizontal wall collisions, reflects the dot in the correct direction.

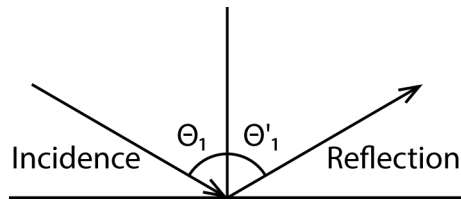


Figure 3.3: Angle of reflection.

Checking each dot for collision with all other dots is costly, in terms of calculations. This will have an effect on the frames per second on older and low performing devices, to an extent where the Motion test can become unusable. The solution was to implement a quadtree algorithm to lower the number of collision checks for each dot.

A quadtree creates a tree where the root node covers the entire patch. When dots are added to the tree, it will eventually split into four subnodes that cover a corner of the patch each. These subnodes will further split into fours, dividing the patch into more areas as needed, see figure 3.4 for an illustration. All dots held in a node can potentially collide with each other. Iterating over the dots in a node it is possible to detect a collision by checking the distance between the dots. If the distance is smaller than the sum of the radii, they are colliding, see figure 3.5.

Finding the angle colliding dots are to bounce off of each-other is calculated by using their current velocity and mass. $v'_1 = v_1(m_1 - m_2) + 2m_2v_2/m_1 + m_2$ where v is the current velocity, m the mass and v'_1 the new velocity for dot 1. For dot 2 the formula is $v'_2 = v_2(m_2 - m_1) + 2m_1v_1/m_1 + m_2$.

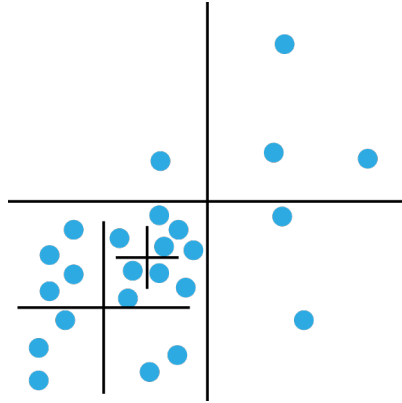


Figure 3.4: Illustration of a Quadtree.

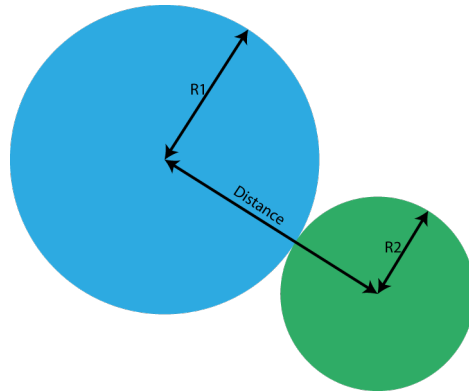


Figure 3.5: A collision occurs when the sum of $R1$ and $R2$ is equal to, or lesser than, the distance between two dots.

Scoring staircase

Finding the threshold score involves the use of a weighted one up one down staircase model. Values used to calculate the threshold score is added to an array when the coherency percent changes from going down to up, or from up to down, also known as reversal values. Figure 3.6 illustrates this, where R1, R2 and R3 is the reversal values. The test ends after a specified amount of reversal values has been recorded, the threshold score is then calculated by taking the geometric mean² of those.

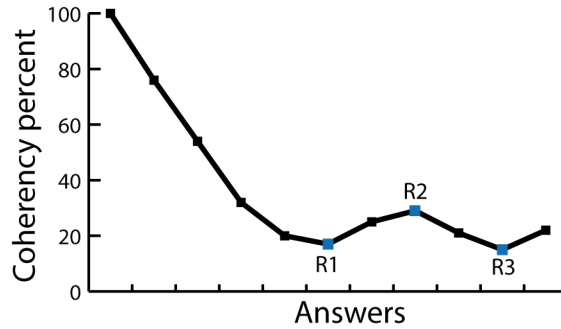


Figure 3.6: Illustration of the staircase implementation.

In figure 3.7 the sequence for updating coherency is shown with the associated classes.

²Geometric mean = $\left(\prod_{i=1}^n rp_i\right)^{1/n} = \sqrt[n]{rp_1 rp_2 \cdots rp_n}$

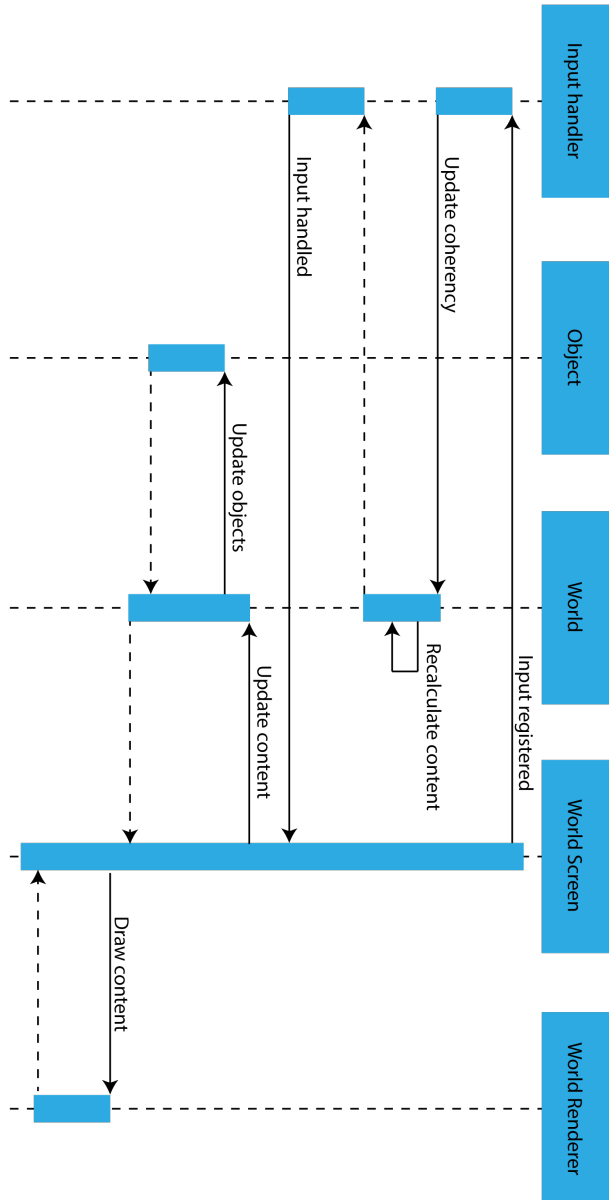


Figure 3.7: Sequence diagram showing how coherency is updated.

3.4 Application overview

The goal for the application was to get the tests to function as correctly as possible. This caused the user interface to come second to the functionality. With the amount of settings available to the user, it was difficult to come up with short but descriptive labels for each of them. The user interface consists of simple white elements on a black background (buttons and labels), with colouring to indicate certain functionalities.

Application screens

The screen captures below are taken with the default settings specified in the application. The patches have a size of $10^\circ \times 14^\circ$ with a 5° space between them. Either the Motion or the Form test displays its content within these patches.

Main menu

From the main menu, figure 3.8, it is possible to run one of the tests, go to the settings sub-menu or exit the application. The Exit button is coloured red to indicate its functionality.

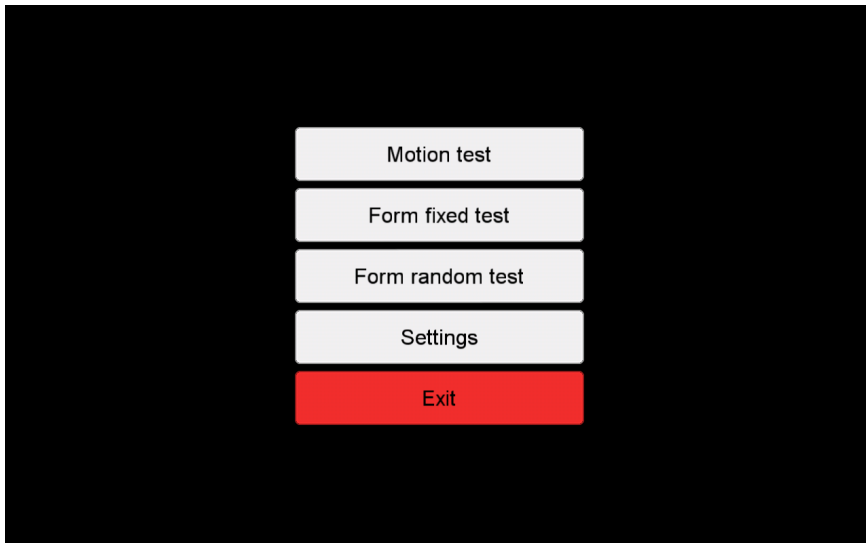


Figure 3.8: Main menu.

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.

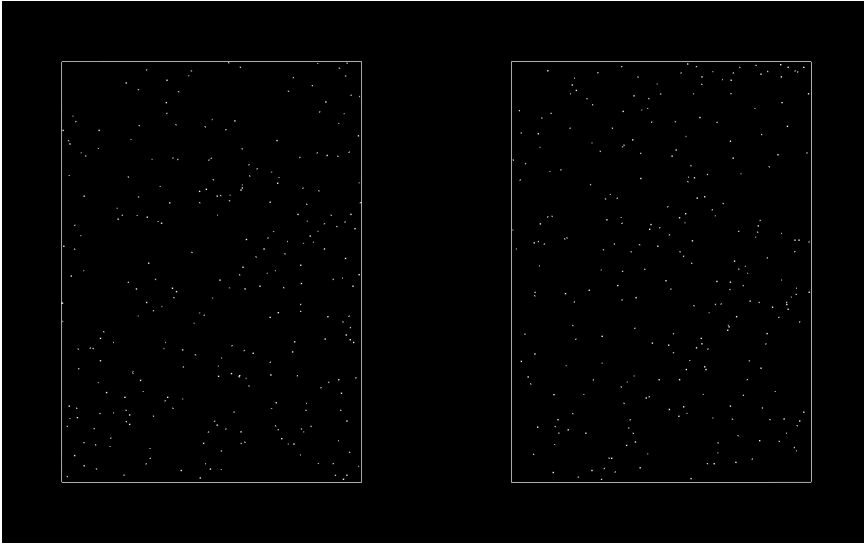


Figure 3.9: Motion test screen capture.

At each interval one of the patches are chosen at random to contain a coherent motion target. In that patch, a percentage of the dots will move either leftwards or rightwards, reversing every 0.572 seconds. The dots not moving coherently, will move at random, changing direction when colliding with other dots and after 0.572 seconds. The dots move at a velocity of 50 pixels per second.

To prevent the test taker from following a single dot, 10% of the dots are destroyed after 0.085 seconds, before they are repositioned and redrawn on the next frame. At 60 frames per second this means that 10% is destroyed every ~ 5 frames.

The test taker is to identify the patch with the coherent moving target during the 5 seconds of animation time. After the dots disappear, the test taker can click on the patch they believe contained the coherent motion target, taking a guess when needed. As default, input is not registered before the interval time is over, this can be changed under Input Settings. When input is registered the dots are recalculated with the change in coherency, and subsequently displayed on the screen.

Figure 3.10 illustrates the dot behaviour at 50% coherency. The blue dots are moving coherently to the right, while the white dots move at random.

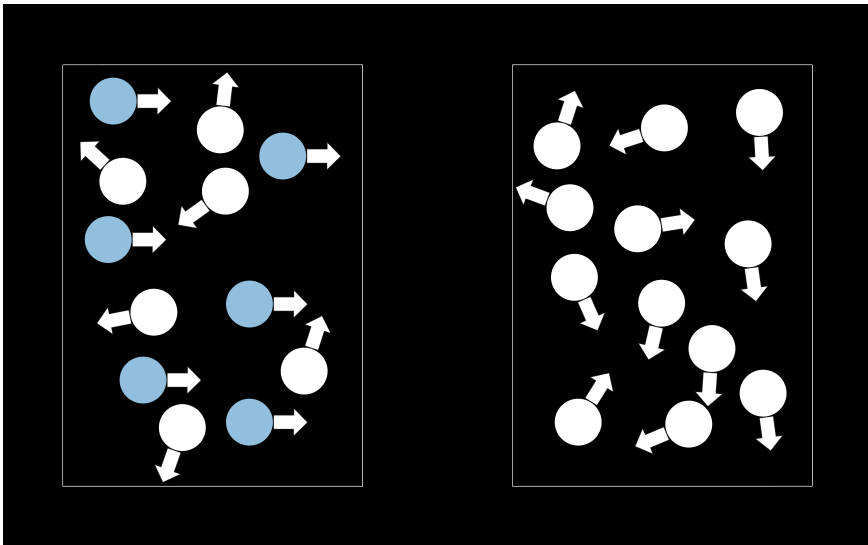


Figure 3.10: Illustration of the motion test at 50% coherency.

The form fixed test

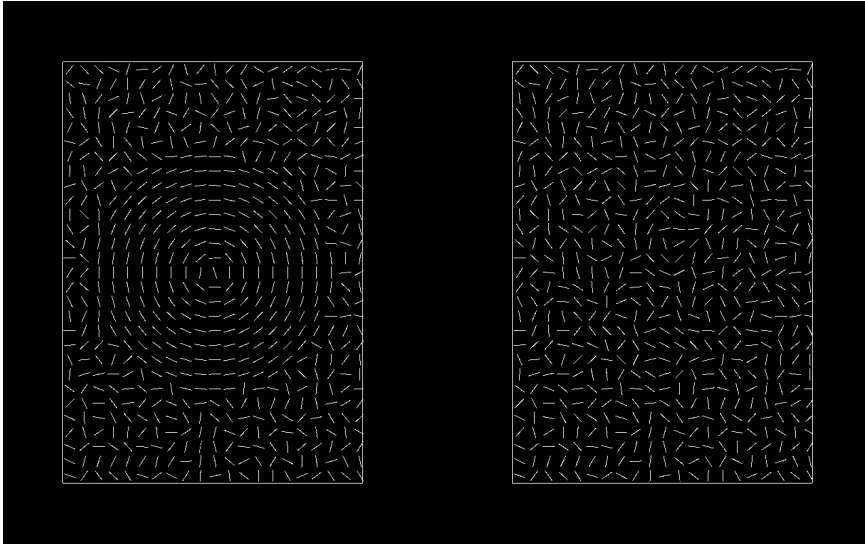


Figure 3.11: Form fixed auto test at 100% coherency.

Each patch contains 600 line segments each, with the lines having a length of 0.4° and a height of 1 pixel.

At each interval one of the patches are chosen at random to contain concentric circles. Line segments within the area of an imaginary circle with the diameter of 8° are oriented to the tangent of the imaginary circles centre. The centre of the circle is locked in the centre of either the left or right patch.

The test taker is to search the patches during the 4 second interval for the concentric circles. After the pattern disappears, the test taker can click on the patch they believe contained the concentric circles, taking a guess when needed. As default, input is not registered before the interval time is over, this can be changed under Input Settings. When input is registered the pattern is recalculated with the change in coherency, and subsequently displayed on the screen.

Figure 3.12 and figure 3.13 shows the test at 50 and 0 percent coherence.

Figure 3.13 shows the Form fixed auto test at 0% coherency.

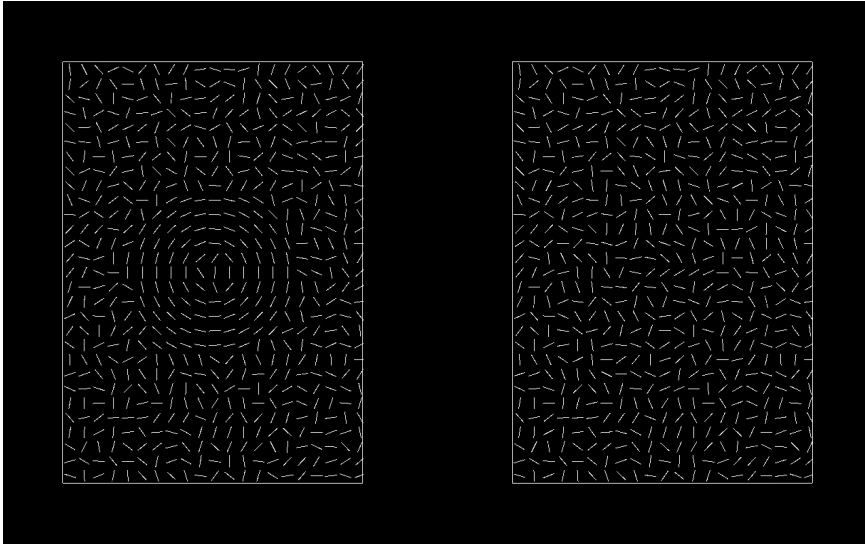


Figure 3.12: Form fixed auto test at 50% coherency.

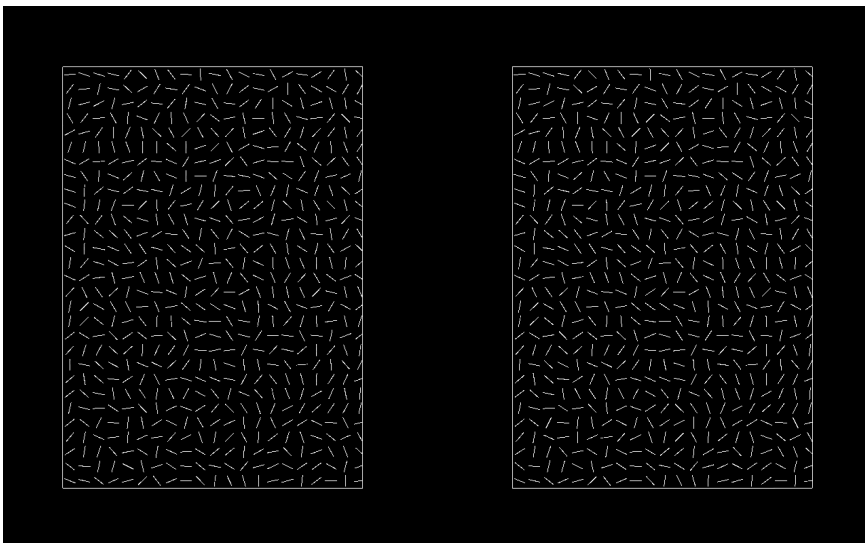


Figure 3.13: Form fixed auto test at 0% coherency.

The Form random test

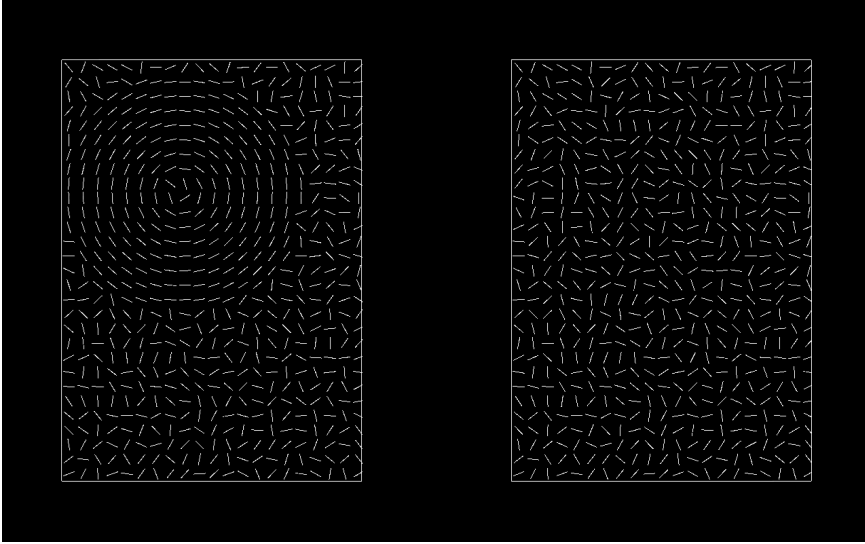


Figure 3.14: Form random auto test at 100% coherency.

At each interval one of the patches are chosen at random to contain concentric circles. Line segments within the area of an imaginary circle with the diameter of 8° are oriented to the tangent of the imaginary circles centre. The centre of the imaginary circle will be placed at random in either the left or right patch, with its circumference confined within the patch.

In this test there is a long time constraint for each interval, due to the added difficulty of finding the target. The test taker is to search the patches during the 1000 seconds interval for the concentric circles. At any time the test taker can click on the patch they believe contains the concentric circles, taking a guess when needed. When input is registered the pattern is recalculated with the change in coherency, and subsequently displayed on the screen.

Figure 3.15 and figure 3.16 shows the test at 50 and 0 percent coherence.

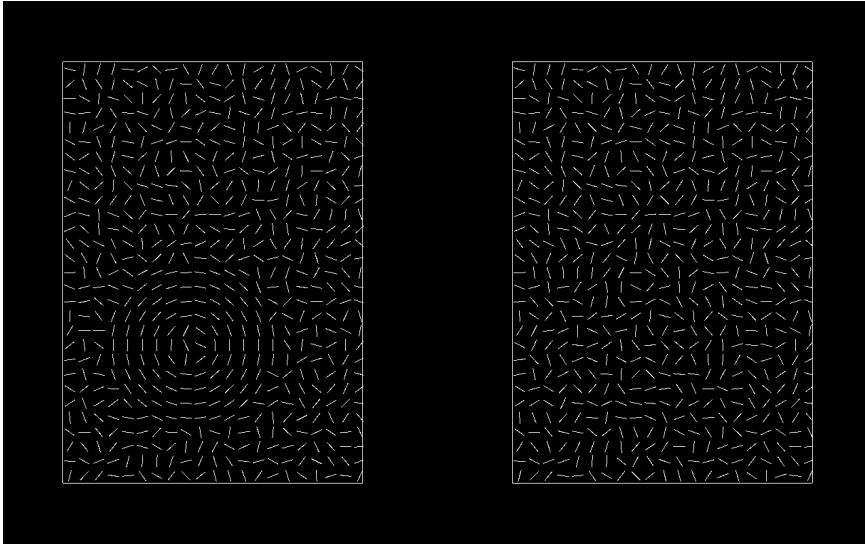


Figure 3.15: Form random auto test at 50% coherency.

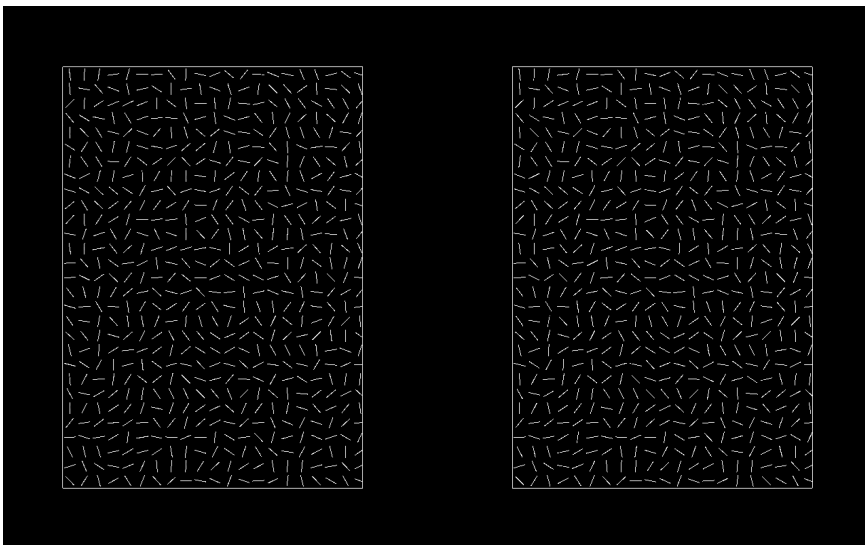


Figure 3.16: Form random auto test at 0% coherency.

Auto vs manual mode

The Form test have two modes, auto and manual. Initially the now manual mode was the only mode, the implementation followed the written description in Hansen et al. [2], and Sigmundsson et al. [22] alongside Sigmundssons recollection of the test. Line segments are arranged at random in both patches, with one side containing line segments in concentric circles.

The auto mode was added later to make the two Form tests look similar to the illustrations of Peter Hansen's tests in figure 2.3. The line segments are evenly distributed within the patches and both sides are initially equal before arranging line segments in one patch into circles.

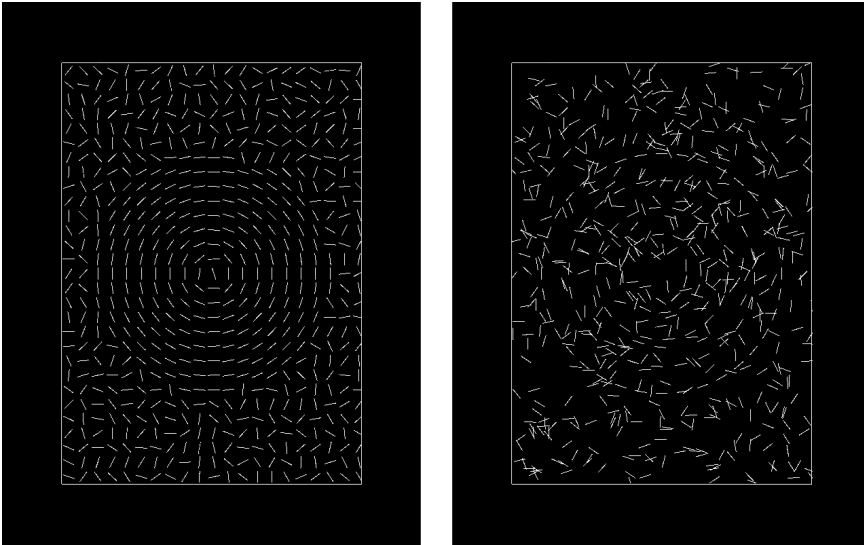


Figure 3.17: Comparison between the auto and manual mode for the Form fixed test.

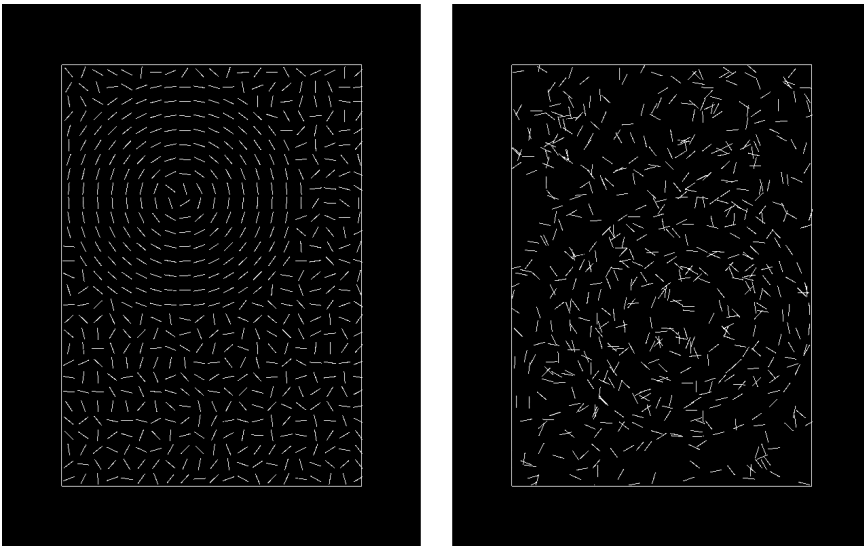


Figure 3.18: Comparison between the auto and manual mode for the Form random test.

Settings menu

Figure 3.19 shows the settings menu with an overview of the available sub-categories. Each of the sub-categories are described below.

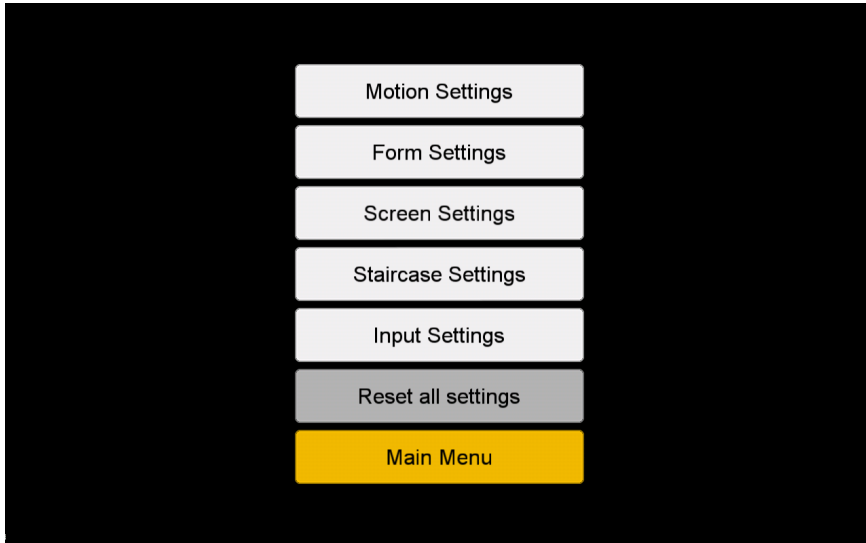


Figure 3.19: Settings menu.

Motion settings

Figure 3.20 shows the settings that can be changed to alter the behaviour of the Motion test.

Number of dots: The number of dots to be contained within each patch.

Radius: The radius for the dots in pixels.

Initial spacing: How far from each other dots should have its starting position, i.e. all dots need to be at least X pixels 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.

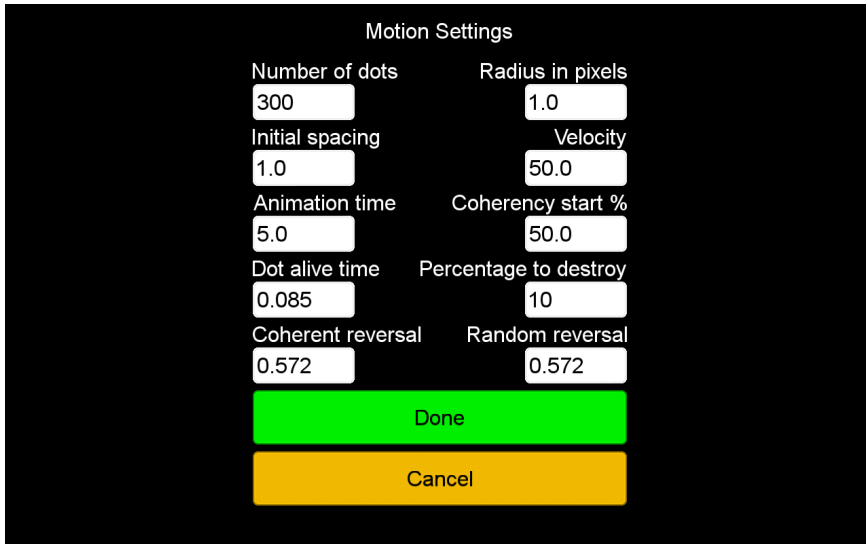
Coherency start %: The percentage of dots to move coherently at the first interval.

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 at 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 configuring motion parameters. 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 3.20: Motion test settings.

Form settings

Figure 3.21 shows the settings that can be changed to alter the behaviour of the Form tests.

Auto mode: With Auto Mode checked, line segments are evenly distributed across the patches. Line segments inside the imaginary circle are oriented to its tangent. At 0% percent coherency, the left and right patch will look identical.

When Auto Mode is unchecked (manual mode) line segments are created at random within the patches. As coherence goes down, a portion of the line segments in the circles are relocated at random within the same patch.

The difference in between auto and manual mode is described on page 30.

Number of line segments: Amount of line segments to be contained within each patch.

Diameter °: The diameter of the utmost concentric circle. Visual angle is used to translate degrees into pixels.

Nr of circles: Number of concentric circles to be displayed. This field is updated when diameter or circle gap is edited. Evenly distributed concentric circles are calculated with a fault of +-1. This setting is disabled when Auto mode is checked.

Circle gap °: The diameter difference for each concentric circle. Visual angle is used to translate degrees into pixels. This setting is disabled when Auto mode is checked.

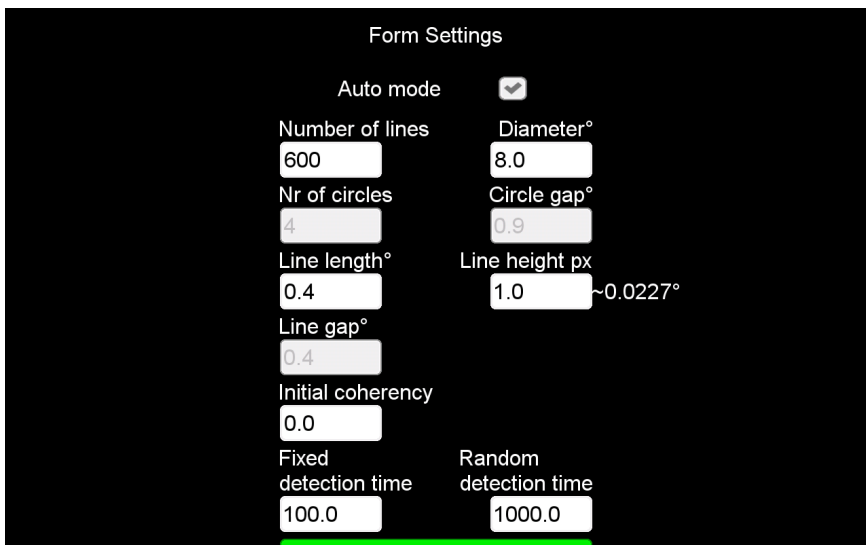
Line length °: The length of each line segment. Visual angle is used to translate degrees into pixels.

Line height px: The height of each line segment in pixels. Visual angle is used to translate degrees into pixels.

Line gap °: The distance between each line segment in the concentric circles. Visual angle is used to translate degrees into pixels. This setting is disabled when Auto mode is checked.

Fixed detection time (seconds): How long each interval should last in the Form Fixed test.

Random detection time (seconds): How long each interval should last in the Form Random test.



The image shows a 'Form Settings' dialog box with a black background and white text. The settings are organized into two columns. The 'Auto mode' checkbox is checked. The 'Number of lines' is set to 600, 'Nr of circles' to 4, 'Line length°' to 0.4, 'Line gap°' to 0.4, and 'Initial coherency' to 0.0. The 'Diameter°' is 8.0, 'Circle gap°' is 0.9, and 'Line height px' is 1.0. The 'Fixed detection time' is 100.0 and the 'Random detection time' is 1000.0. A small note next to the 'Line height px' value indicates it is approximately 0.0227°. A red horizontal bar is visible at the bottom of the dialog box.

Parameter	Value
Auto mode	<input checked="" type="checkbox"/>
Number of lines	600
Nr of circles	4
Line length°	0.4
Line gap°	0.4
Initial coherency	0.0
Fixed detection time	100.0
Random detection time	1000.0
Diameter°	8.0
Circle gap°	0.9
Line height px	1.0 (~0.0227°)

Figure 3.21: Form test settings.

Screen settings

Figure 3.22 shows the settings related to screen size and resolution. The width and height measurements in mm and pixels are fetched automatically when the application starts. They are made editable due to screen panels being larger than the viewable area, causing the values to be off in some cases. The patch rectangles in the background shows the area where test elements will be displayed.

Width in mm: The width of the viewable screen area in millimetres. This field is automatically calculated at first start up and when resetting all settings.

Height in mm: The height of the viewable screen are in millimetres. This field is automatically calculated at first start up and when resetting all settings.

Width in pixels: The width wise pixel count of the screen. This field is automatically calculated at first start up and when resetting all settings.

Height in pixels: The height wise pixel count of the screen. This field is automatically calculated at first start up and when resetting all settings.

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. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

Patch height °: The height of each patch. Visual angle uses the height measurements of the screen to convert degrees to pixels in this case.

Patch gap °: The gap between each patch. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

Editing any value will change the text on the "Done" button to "Apply", see figure 3.23. The patch rectangles updates when the new settings are applied.

Screen settings

Width in mm	Height in mm
152	95
Width in pixels	Height in pixels
1280	800
Distance in mm	
300	
Patch width in°	Patch height in°
10	14
Patch gap in°	
5	

Done

Cancel

Figure 3.22: Screen settings.

Screen settings

Width in mm	Height in mm
160	95
Width in pixels	Height in pixels
1280	800
Distance in mm	
300	
Patch width in°	Patch height in°
10	14
Patch gap in°	
5	

Apply

Cancel

Figure 3.23: Changes in the screen settings need to be applied for changes to take effect.

Staircase settings

Figure 3.24 shows the settings related to recording test results. The amplitude ratio³ represents the percentage change that occurs when a user answers correct or incorrect. At 1dB this is equal to a 1.22% change in coherency. See section 3.3 for further explanation of how this works.

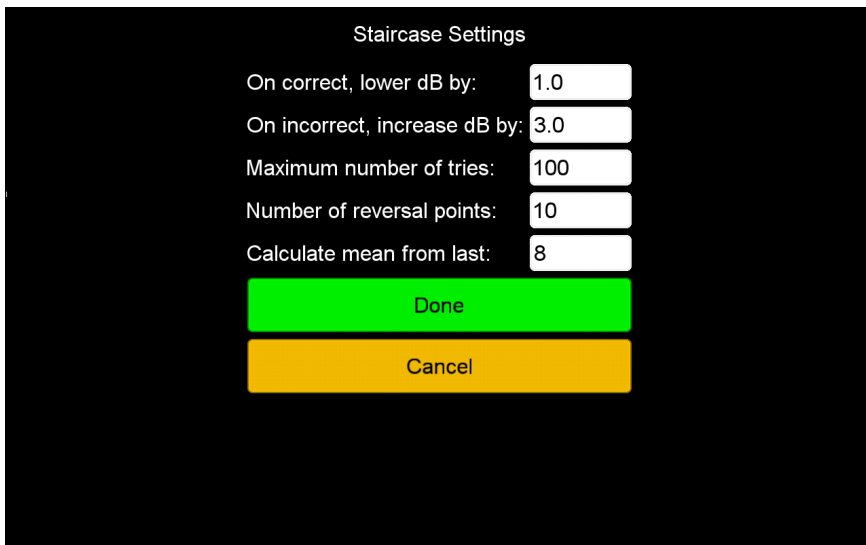
On correct lower dB by: Uses the amplitude ratio of the decibel as a factor to lower coherency on correct answer. Can be a negative or positive number.

On wrong increase dB by: Uses the amplitude ratio of the decibel as a factor to increase coherency on wrong answers. Can be a negative or positive number.

Maximum number of tries: Maximum number of tries for a given test.

Number of reversal points: Number of reversal points allowed before the test is ended.

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.



Staircase Settings

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

Done

Cancel

Figure 3.24: Staircase settings.

³Amplitude ratio = $10^{(\text{decibel}/20)}$

Input settings

Figure 3.25 shows the settings related to user input when interacting with the application.

Continuous mode: When checked, it is possible to choose a patch before the animation time runs out. In this mode the content of the patches are recalculated and redrawn straight after the user makes a choice.

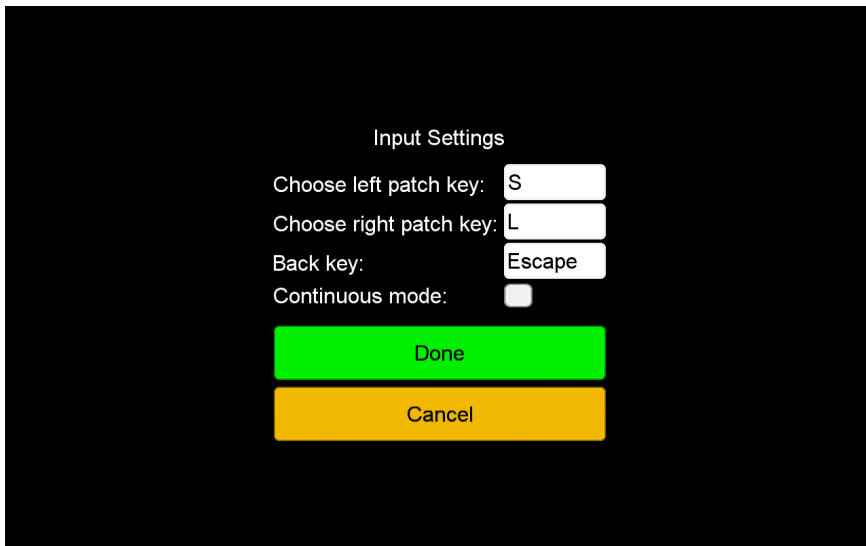


Figure 3.25: Input settings.

When changing the key-binding for an input option, a dialog opens, see figure figure 3.26, to prevent the pushed key to perform an action. For example, if the user press the back key when this dialog is open, it prevents the application from interpreting it as a the user wants to go back to the previous screen.

Reset dialog

To prevent accidental resetting of settings, a dialog prompt opens where the user has to confirm or cancel the action.

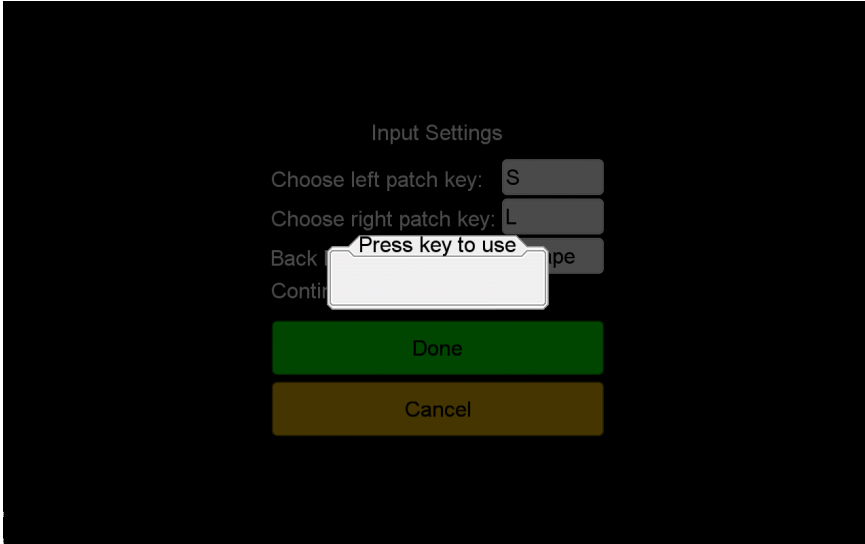


Figure 3.26: Input settings key dialog.

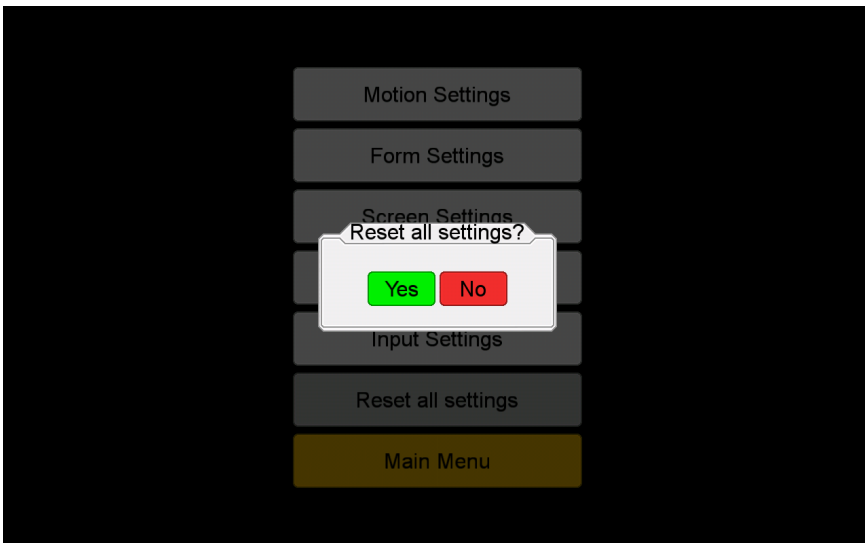


Figure 3.27: Reset settings confirmation dialog.

Chapter 4

Evaluation

The application have been under a continuous testing process during the implementation phase, with a main focus on getting the tests to be visually similar to the tests developed by Peter C. Hansen [2]. A group of volunteers tested the application for it's usefulness after the implementation phase. In this chapter the testing and evaluation process of the application is described.

4.1 Testing

An expert in the field of dyslexia evaluated the prototypes to give feedback on parameter values forming the base for further work. When work continued away from the test prototypes to an application, the feedback changed from being about sizes of elements to the behaviour of the Motion and Form tests.

Test prototypes

Two prototypes were created with help an expert in the field, alongside the information found in the articles of Hansen et al. [2] and Sigmundsson et al. [22], [23]. With sizes of the individual elements being critical to the visual impression of the tests, the prototypes aimed to find the right sizes for individual elements.

The Motion test prototype went through four iterations, changing the behaviour and adding additional parameters for each one. For the first prototype, see figure 4.1, only the radius and travel length of each dot was editable. Coherence was locked to 50%, while the non-coherent dots could move in any direction. All dots reversed after a set travel distance.

The non-coherent dot behaviour in the second prototype, see figure 4.2, changed from travelling a set distance before reversing, to change direction every 0.2 seconds without any limit on their travel distance. Coherent dots would still travel a set distance before reversing. Changing coherency was possible at this stage, for better visualisation of how the test would look at different coherency percentages.

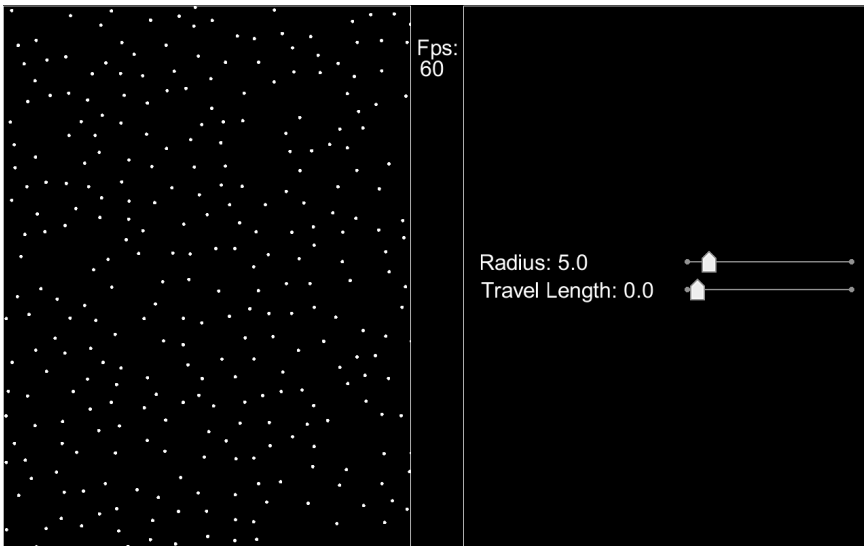


Figure 4.1: Screen capture of the first Motion test prototype.

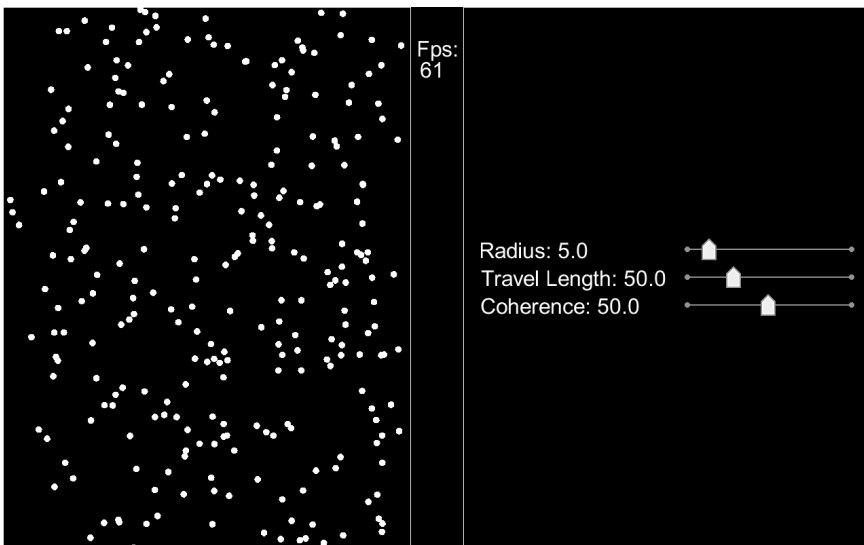


Figure 4.2: Screen capture of the second Motion test prototype.

From the third prototype, see figure 4.3, the focus changed from finding the correct size of the dots, to get the behaviour of the dots correct. Dots were given a finite lifetime, were they all would relocate after the timer ran out. This caused a very staccato behaviour of the test that was not wanted. The solution was to relocate a percentage of the dots at a given time interval.

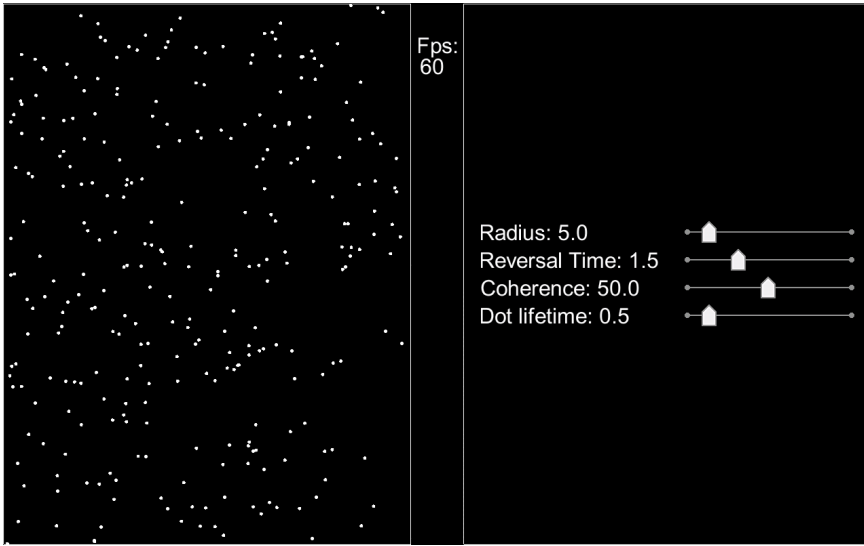


Figure 4.3: Screen capture of the third Motion test prototype.

Figure 4.4 shows the only prototype made for the Form tests. The centre point for the circles was static in the middle of the patch. Since this test only displays a static image, the sliders only control the size and placement of the different elements.

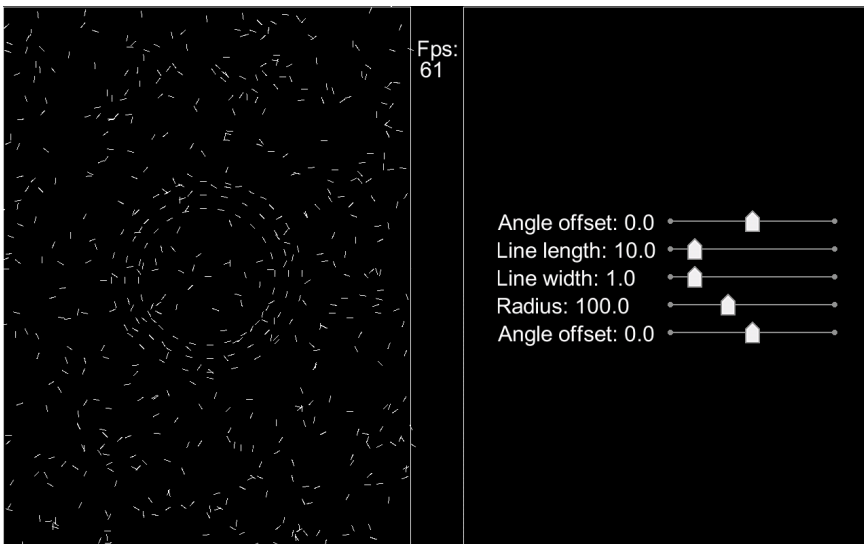
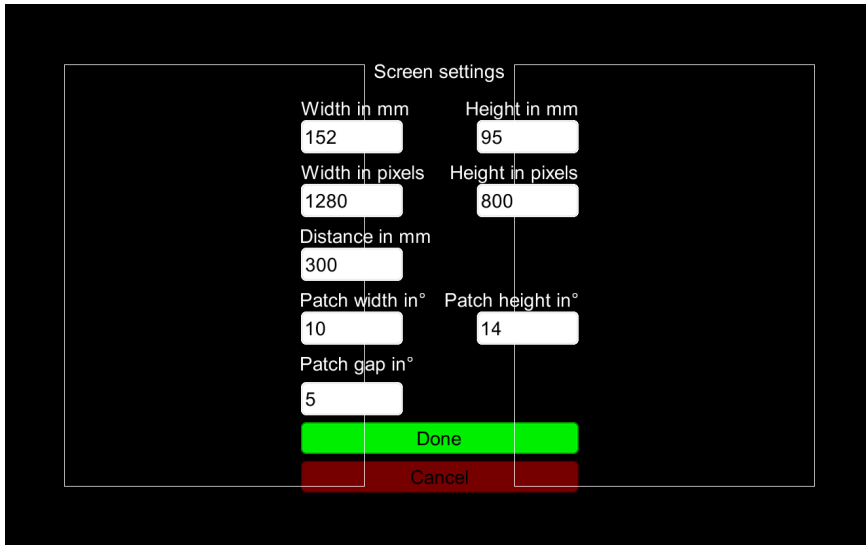


Figure 4.4: Screen capture of the form test prototype.

With the feedback from the third prototype, the models for each of the tests were

re-implemented. Additional parameters describing the Motion and Form tests were added, and the tests took advantage of visual angle to give a uniform visual representation across devices, see figure 4.5. The adjustment of the parameters was now moved to its own screen, see figure 4.6, making it possible to run a test sequence with both patches containing test elements, see figure 4.7.



Screen settings	
Width in mm	Height in mm
152	95
Width in pixels	Height in pixels
1280	800
Distance in mm	
300	
Patch width in°	Patch height in°
10	14
Patch gap in°	
5	
Done	
Cancel	

Figure 4.5: Screen capture of the screen settings in the fourth motion test prototype.

From the fourth prototype out, parameters, parameter values and test behaviour were collected from the articles written by Hansen et al. [2] and Sigmundsson et al. [22], [23]. Testing of the subsequent iterations were aimed towards getting feedback about the menu system, setting names and of changes that could be made to the Motion and Form tests. The tester prompted for a user guide at this point, as the functionality of the different settings can be hard to understand from their names alone. This can be viewed in Appendix C.

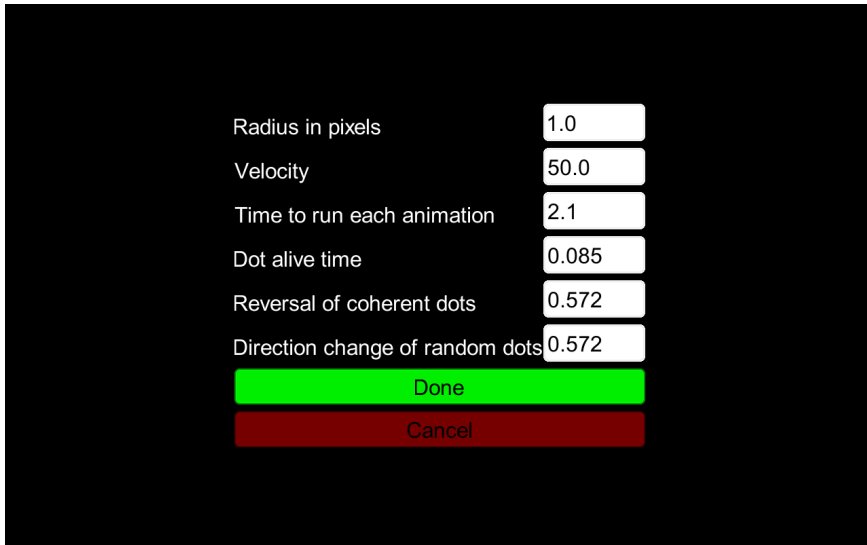


Figure 4.6: Screen capture of the Motion test settings in the fourth motion test prototype.

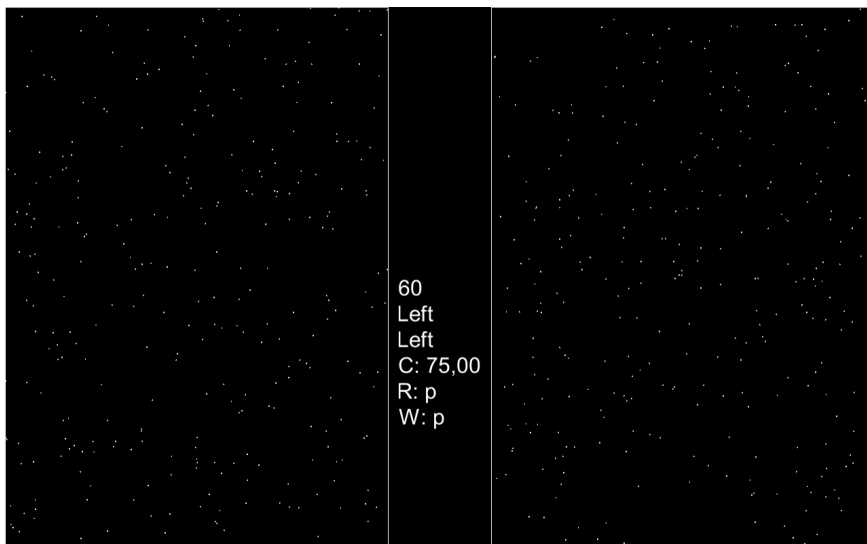


Figure 4.7: Screen capture of the fourth motion test prototype.

4.2 Initial evaluation

A Nexus 7 2012¹ tablet in landscape orientation was used as the testing device. The application ran with the default settings, with an adjustment of the viewing distance to 346mm instead of 300mm. The testing were performed under similar conditions, in a low light environment with candidates using any sight correcting remedies they normally use. For female 2 it was not possible to use the same device due to her geographical location. She used her own computer at home, with guidance through the phone and a remote desktop application to set up and explain the testing procedure. Male 3 did the test at a cafeteria where noise and lighting might have had an effect on the test scores. He was the only one to run the Motion test twice, the test score for the Motion test in table 4.1 is an average of the two runs.

Of the seven volunteers, four were female, three male, and all between the ages of 24 and 30. Three volunteers had a previous diagnosis of dyslexia, and one reporting difficulties with reading. Three none dyslexic acted as the control group. Female 2 reported a reading level of about a 15 year old. Male 3 reported having a reading level similar to 15 year olds with the addition of dyscalculia². Female 3 have no diagnosis, but reported being a slow reader who loses context at times. And Female 4 reported having phonological³ dyslexia causing her to have spelling problems and being a slow reader.

Each volunteer were informed that the test would be anonymous and it would be possible to end the test at any moment if they wished to. A trial run was set up before each test, to explain the test and make sure the volunteer understood the test procedure. For the trial runs, the coherency was set to 100% as the starting point. A reset of the coherency setting was done when the volunteer felt safe about the test procedure and was ready to record test results.

Test results

Female 3 have not been included in the dataset when calculating the average scores and differences between the control group and the diagnosed dyslexic. She has a history of being a slow reader, but without a diagnosis, it is not possible to place her in either group. Her scores are included in table 4.1.

The motion test scores shows a 61.10%⁴ difference between the control group and the group with diagnosed dyslexics. Disregarding Male 3s results, as his score is relatively high; there is a 32.19% difference between the two groups.

In the Form auto tests the discrepancy between the two groups are lower, with a 10.86% difference on the Fixed portion of the test, and 1.33% on the Random portion.

¹7" display with a resolution of 800x1280 (portrait)

²Problems with learning and understanding arithmetic [38]

³Difficulty with word sounds

⁴Percentage difference = $\left(\frac{\text{controlaverage} - \text{dyslexicsaverage}}{(\text{controlaverage} + \text{dyslexicsaverage})/2}\right) * 100\%$

Candidate	Motion	Fixed auto	Random auto	Fixed manual	Random manual
Female 1	13.63	5.70	4.77	2.30	9.65
Female 2	16.46	5.50	8.35	3.30	16.26
Female 3	30.48	6.75	6.12	3.76	8.49
Female 4	22.06	8.11	5.50	4.91	8.12
Male 1	12.36	5.75	7.38	6.65	8.04
Male 2	15.76	7.45	8.03	7.05	7.98
Male 3	39.97	7.37	6.61	3.10	12.82

Table 4.1: Test results. Diagnosed dyslexics highlighted.

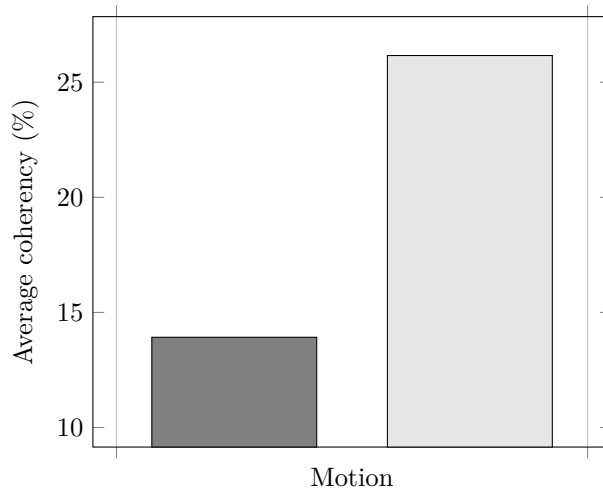


Figure 4.8: Darker bars represents controls while lighter once represents diagnosed dyslexics. Female 3 is not included in this dataset.

With the Form tests in manual mode, dyslexic group do better, on average, than the control group by 34.29% on the Fixed portion of the test. On the Random portion of the test, the difference is 36.64% in favour to the control group.

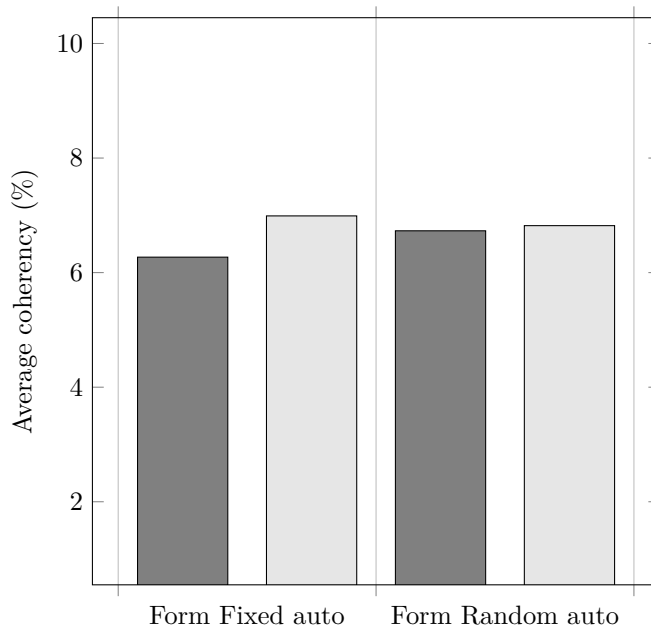


Figure 4.9: Darker bars represents controls while lighter once represents diagnosed dyslexics. Female 3 is not included in this dataset.

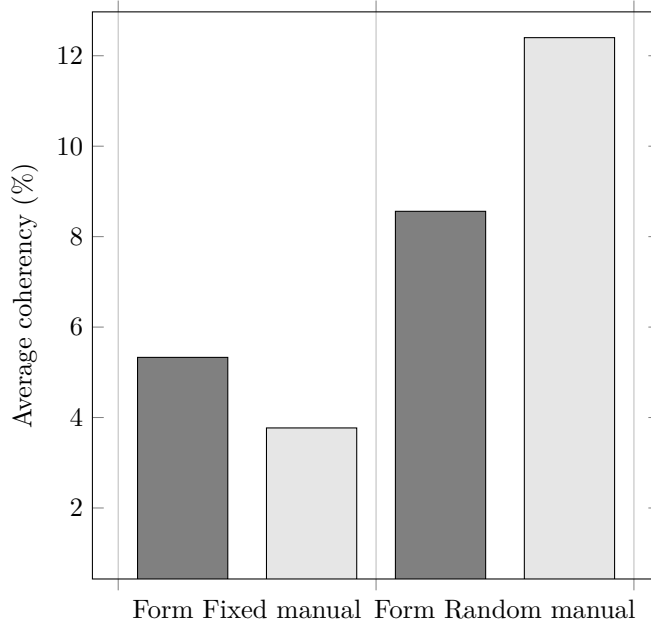


Figure 4.10: Darker bars represents controls while lighter once represents diagnosed dyslexics. Female 3 is not included in this dataset.

Chapter 5

Discussion, Conclusion and Further work

In this chapter the results from the initial evaluation is discussed. The discussion shows how the test results correlates with previous research, and the differences in scoring. Under conclusion I answer the research questions, with regards to the work and findings in this paper. And finally, two alternatives for further work is proposed.

5.1 Discussion

Correlating with the findings of Hansen et al. [2] and Sigmundsson et al. [23], the group of dyslexic are on average less likely to identify the patch containing coherent motion in the random dot kinematogram test (Motion test) compared to a non-dyslexic group. However, with Female 2s score being close to Male 2s in the control group, the testing method comes into question. Three factors could have had an impact on the results; the implementation is not good enough, the use of a difference device, or the type of dyslexia she exhibits. Comparing the score between Female 2 and Male 2 shows that Female 2 scores 4.24% higher in the Motion test. This is consistent with the findings of Cornelissen et al. [16] and Ridder et al. [39], where they found all sub-groups of dyslexia to score on average 3-4% higher than a control group.

The minor differences in test scores for the Form Fixed auto and Form Random auto tests is similar to the findings of Hansen et al. [2] and Sigmundsson et al. [23], with one exception. Hansen et al. and Hermundsson et al. reported an average coherency above 20% for both tests, compared to the findings in this paper where the average coherency is below 8%. However, the difference between the dyslexic and control group is relatively close to each other across the research.

The results from the Form Fixed manual and Form Random manual tests shows an interesting variation. The dyslexic group scores on average lower than the control group in the Fixed manual test, while they score higher in the Random manual

test. This suggests dyslexic finds the target circles easier than the control group as long as the centre is static in the middle of either patch. With the small sample pool of data, it is not possible to conclude this will be the effect across a larger sample size. Variations caused by random placement of line segments in these tests could have had an influence on the test results. I tested this by taking each of the four Form tests 10 times each, the scores recorded can be viewed in table 5.1. The Random manual scores tend to fluctuate more than the scores for the other tests, but it is not a conclusive evidence of random placement having an effect on the test scores.

Test run	Fixed auto	Random auto	Fixed manual	Random manual
1	6.47	6.18	5.99	8.96
2	5.97	6.90	7.51	12.82
3	7.87	5.32	7.85	9.53
4	8.12	6.47	8.83	5.99
5	5.93	8.11	6.46	10.07
6	6.31	7.14	8.29	12.40
7	5.26	5.50	5.10	9.55
8	8.28	6.40	6.69	11.25
9	7.49	6.40	7.59	9.44
10	4.71	6.53	6.91	8.93
	6.64	6.50	7.12	9.89

Table 5.1: Testing for variations caused by random placement of line segments. Average scores for each test highlighted.

To provide a certain answer about the validity of the tests is difficult. The evaluation method contains flaws, as it is lacking an assessment of the participants relative reading level before taking the test. However, the test scores indicates a connection between known reading and writing problems, and the results from the Motion test. This correlates with the previous studies of Cornelissen et al. [16] and Demb et al. [17], linking the problem with a deficit in the magnocellular pathway. With the magnocellular pathway specialising in locating objects [14], a correlation between the Motion test score and Form Random scores was expected [23]. In Form Random auto this was not found, while in Form Random manual the difference in average score is significant. This difference can be explained by the fluctuation observed in table 5.1, providing no conclusive evidence to support the correlation.

5.2 Conclusion

RQ1: How is the procedure for recreating a DOS-program to the modern tablet format?

Recreating software is an iterative process with four phases: information gathering, implementation, testing, and evaluation.

The information gathered about the program is used to define functional and non-functional requirements for the new implementation. During the implementation phase, prototype testing is used to gain feedback of changes needed to reproduce the functionality of the original program. In the evaluation phase, the usefulness of the application is assessed.

RQ1.1: What kind of information is needed?

Three categories of information was needed for recreating the program: test procedures, test behaviour, and visual representations of each test.

Information about the test procedure will give insight on how the program is to be used. This includes the test setting (i.e darkened room), and how scores are calculated.

Descriptions of the test behaviour provides information about the functionality of the tests. Number of elements represented on the screen, size and movement of elements, and time frames is translated into parameters for the implementation.

A visual representation of the program to recreate, in the form of screen captures or illustrations. This will help with getting the visual aspect of the implementation correct.

RQ1.2: How can information about the older software be obtained?

There are four ways of obtaining information about older software.

1. Get hold of the creator and ask if he or she has any information specifying the functionality of the software.
2. Talk with previous users of the program to obtain first hand knowledge.
3. Take advantage of scientific search engines to find articles written about the software.
4. Research similar types of software.

RQ2: Do the test scores correlate with dyslexia?

The results from the Motion part of the application shows scores to correlate with dyslexia. For the Form Fixed and Form Random tests, there is no conclusive evidence to support correlation between test scores and dyslexia.

RQ2.1: Can the application be used for detecting dyslexia?

With only a preliminary evaluation of the application, it is not possible to give a definite yes or no answers to this question. The results look promising, but the application is in need of further evaluation to validate the findings in this paper.

5.3 Further work

Two possibilities for further work is proposed in this section.

Clinical experiment

With the results from the initial evaluation of the application, it stands out that the application needs further testing. For this, I suggest a clinical experiment, where the test is taken under equal conditions and that the test takers are subjected to a preliminary assessment of their reading level. The test group should consist of a group of dyslexic and a group of non-dyslexic to be used as control. Both groups should have female and male participants within the same age group.

Choosing between auto and manual mode for the Form test, will be up to the test giver.

Adjustment of the test parameters in the Motion and Form tests should be considered if the test results show no correlation between test scores and dyslexia. Starting with lowering the time constraint for each test. If this does not prove sufficient, lower the target values. For the Motion test this will be to lower the dot alive time and increase the kill percentage. While in the Form tests, the circle diameter can be reduced.

Making it a game

Implementing features similar to those in mobile games can help with making the test suitable for testing in lower age groups. Magnofly is an example of this.

Appendices

Appendix A

Appendix A

A.1 Installation

All release versions are available from <http://folk.ntnu.no/bjornawo/release/>.

Android

Dependencies: Android version 3 (Honeycomb) or newer.

1. If you have previous version installed, go ahead and uninstall it.
2. Make sure that you have enabled *Unknown sources* under *Settings* → *Security* → *Device administration*.
3. Download the application from <http://folk.ntnu.no/bjornawo/release/android-realease.apk>.
4. Go to *App drawer* → *Downloads* and tap on *android-release.apk*. (The file location might vary from device to device.)
5. Tap on *Install*.
6. The application can now be found under *App drawer* → *Dyslexia Detection* or on your home screen.
7. Tap the icon to run the application.

Optional

8. Go to *Settings* and tap on *Reset all settings*.
9. Tap *Yes* in the confirmation dialogue.

Java

Dependencies: Java SE JRE.

1. Download the application from <http://folk.ntnu.no/bjornawo/release/java-realease.jar> to a location you will find it again.
2. Run the application by double-clicking on the *java-release.jar* file.

Optional

3. Go to *Settings* and click on *Reset all settings*.
4. Click *Yes* in the confirmation dialogue.

Mac with standalone Java

Dependencies: Xquartz.

1. Download the application from <http://folk.ntnu.no/bjornawo/release/mac-realease.zip> to a location you will find it again.
2. Unpack the files to a location of your own choice.
3. Double-click on the *mac-release* folder.
4. Double-click on the *Contents* folder.
5. Double-click on the *MacOS* folder.
6. Double-click on *Dyslexia Detection* to run the program.

Optional

7. Go to *Settings* and click on *Reset all settings*.
8. Click *Yes* in the confirmation dialogue.

Windows with standalone Java

Dependencies: None

1. Download the application from <http://folk.ntnu.no/bjornawo/release/win-realease.zip> to a location you will find it again.
2. Unpack the files to a location of your own choice.
3. Double-click on the *win-release* folder.

4. Double-click on *Dyslexia Detection(.exe)* to run the program.

Optional

5. Go to *Settings* and click on *Reset all settings*.
6. Click *Yes* in the confirmation dialogue.

A.2 Test results

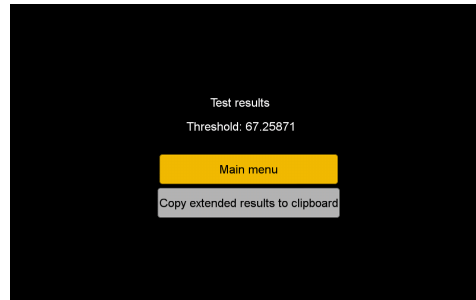


Figure A.1: The threshold calculated by taking the geometric mean of the X last reversal values. $\left(\prod_{i=1}^n x_i\right)^{1/n} = \sqrt[n]{x_1 x_2 x_3 \cdots x_n}$

Extended test results

The extended test results contains more information about the test procedure. The extra information have additional parameters regarding the test, as well as all settings used during the test. This can be obtained by pushing the *Copy extended results to clipboard* button on the test results screen. The copied text is formatted with JSON to make it easier to parse at a later stage. Below is the copied text from a test run with added comments.

```
{
correctAnswers: 12 (The number of correct answers.)
wrongAnswers: 17 (The number of incorrect answers.)
threshold: 89.237976 (The threshold value for the test.)
lowestCoherency: 59.42122 (The lowest coherency percentage obtained during the test.)
reversalValues: [ (An array holding the values of the reversal points.)
  1: 59.42122 (The coherency percent at the first reversal point.)
  2: 100.0 (The coherency percent at the second reversal point.)
  3: 77.08516 (The coherency percent at the third reversal point.)
  4: 100.0 (The coherency percent at the fourth reversal point.)
  5: 87.79816 (The coherency percent at the fifth reversal point.)
  6: 100.0 (The coherency percent at the sixth reversal point.)
]
```

7: 77.08516 (*The coherency percent at the seventh reversal point.*)
8: 100.0 (*The coherency percent at the eight reversal point.*)
9: 77.08516 (*The coherency percent at the ninth reversal point.*)
10: 100.0 (*The coherency percent at the tenth reversal point.*)
]

screenSettings: [(*An array holding the screen settings used during the test.*)
screen_w_mm: 508 (*The screen width in millimetres.*)
screen_h_mm: 285 (*The screen height in millimetres.*)
screen_w_px: 1920 (*The pixel count over the width of the screen.*)
screen_h_px: 1080 (*The pixel count over the height of the screen.*)
viewing_distance: 300 (*The distance to the screen measured in millimetres.*)
patch_width: 10 (*The patch width measured in degrees.*)
patch_height: 14 (*The patch height measured in degrees.*)
patch_gap: 5 (*The distance between each patch measured in degrees.*)
]

motionSettings: [(*An array holding the settings used for the motion part of the test.*)
dot_amount: 300 (*Number of dots in each patch.*)
dot_radius: 1.0 (*The radius of each dot in pixels.*)
dot_spacing: 1.0 (*The initial space between each dot in pixels.*)
dot_velocity: 50.0 (*The speed of each dot measured in pixels per second.*)
dot_coherency: 50.0 (*The initial coherency percent used in the test.*)
dot_animation_time: 5.0 (*The maximum time each interval lasted in seconds.*)
dot_max_life_time: 0.085 (*The time each dot was alive.*)
dot_kill_percentage: 10 (*How many percent of the dots to be killed every 0.085 seconds (max.life_time).*)
dot_horizontal_reversal_time: 0.572 (*The amount of seconds horizontal moving dots are to move in on direction before reversing.*)
dot_random_direction_time: 0.572 (*The amount of seconds random moving dots are to move in on direction before changing to a new random direction.*)
]

formSettings: (*An array holding the settings used for the form part of the test.*)[
form_auto_mode: true (*True if auto mode was used, false if manual mode was used.*)
form_line_amount: 600 (*The number of line segments in each patch.*)
form_diameter_wb: 8.0 (*The diameter of the circle used to arrange coherent lines, measured in degrees.*)
form_nr_of_circles: 4 (*If manual mode was used: the maximum amount of circles drawn.*)
form_circle_gap: 0.9 (*If manual mode was used: the gap between each circle.*)
form_line_length: 0.4 (*The length of each line segment measured in degrees.*)
form_line_height: 1.0 (*The height of each line segment measured in pixels.*)
form_line_gap: 0.4 (*If manual mode was used: the gap between each line segment in the circles.*)
form_coherency: 100.0 (*The initial coherency percent used in the test.*)


```
    form_f_detection_time: 4.0 (The maximum time each interval lasted in seconds
for the form fixed test.)
    form_r_detection_time: 1000.0 (The maximum time each interval lasted in sec-
onds for the form random test.)
]
staircaseSettings: (An array holding the staircase settings used during testing.)[
    stair_correct_db: 1.0 (The dB change to coherency when a correct answer was
registered.)
    stair_wrong_db: 3.0 (The dB change to coherency when an incorrect answer was
registered.)
    stair_max_tries: 100 (The maximum number of answers allowed during testing.)
    stair_reversal_points: 10 (The number of reversal points to used during testing.)
    stair_mean_from_last: 8 (The number of reversal points used to calculate the
threshold value.)
]
inputSettings: (An array holding the input settings used during testing.)[
    input_key_left: S (The hardware key used to choose the left patch.)
    input_key_right: L (The hardware key used to choose the right patch.)
    input_key_back: Escape (The hardware key used to go back or cancel a test.)
    input_continuous_mode: false (True if answers were registered on the fly in the
motion and form fixed test, otherwise false.)
]
}
```


Appendix B

Appendix B

B.1 Magnofly Read Me.txt

Magnofly

Version 1.0

Originally designed for a 1280x1024 LCD monitor at 60 FPS (frames per second).

Getting Started

Just double-click on Magnofly.exe, and the game should load. If it doesn't, review your system requirements (see above).

It is very important for the Magnofly program to know what your system's FPS is, along with other settings such as the size of your monitor and your average viewing distance away from your monitor. So before you run the program, open up the Game Parameters.txt file in the program directory. If the text file does not exist, double click Magnofly.exe and exit the program, and it should appear. Read through it carefully, adjusting the parameters appropriately. All the timing values are in terms of frames. Thus the following parameters are dependent on frame rate:

Monitor Refresh Rate (FPS)

Baby Movement Speed

Num. Frames for Round Stage- Initialization

Num. Frames for Round Stage- Fly Cloud Emerging

Num. Frames for Round Stage- Visual Cue

Num. Frames for Round Stage- Visual Cue ISI

Num. Frames for Round Stage- Coherence Cycle

Num. Frames for Round Stage- Success

Num. Frames for Round Stage- Failure

These parameters are of the form (Monitor FPS)*(number of seconds desired for task) as an integer, the number of frames for the task. So these values are very important to change off the bat, otherwise events in the game will be far faster or slower than expected. Similarly, the physical monitor's size is also important: Be sure to immediately fill out Viewable Image Size and Viewer Distance. Once these timing and physical monitor parameters have been set, we're ready to try the game.

Double click on the executable Magnofly.exe. A menu screen should pop up with buttons on the left. Click on "Edit Profile" first. Click on the text fields, fill out your information, and press "Ok" to escape out. Then click on "Instructions," and read what it has to say. After reading about the basic game mechanics, click on "Demo"—an exact but shortened version of the game. Once you're comfortable with detecting the subtleties of insect motion, play the game (clicking "Start Game").

Theory

This game was designed with an examiner in mind who customizes the Game.Parameters.txt file and makes sure that everyone who plays the game has a correct profile. The game is essentially a psychophysical procedure—one that tries to identify the threshold for a player in perceiving coherent motion—collecting the trial data in the /profiles/ folder. Magnofly uses an adaptive Bayesian threshold finding procedure called QUEST to find that threshold percentage—i.e. at what percentage of bugs that move in one direction is low enough so that the user can't really distinguish those moving bugs with the randomly moving bugs in the swarm. Magnofly is named from the idea, promoted by certain researchers who study dyslexia such as John Stein and Joel Talcott, that dyslexics require a higher threshold than control patients possibly due to magnocellular deficiencies. This game tries to replicate the random dot kinematogram testing procedure (which does correlate to magnocellular deficiencies) that is performed in dyslexia research to test correlate that performance to other diagnostic features of dyslexia (poor reading scores, etc). If there is a correlation, then perhaps magnocellular impairment could be a major neurological reason for some of the problems that dyslexics experience.

For more information about QUEST: <http://www.psych.nyu.edu/pelli/>

[List papers for Magnocellular and dyslexia]

Troubleshooting

If your game crashes, check the error text files. If nothing is amiss there, there could be a problem with the sounds. As an ad-hoc fix, try disabling them by typing false in the Game.Parameters.txt file for the fields of Enable Sound Effects (or, if the problem continues, Enable Music).

Credits

The project was developed with support from the National Science Foundation through grant #IIS-0113310 to James A. Ferwerda.

Thanks to the Cornell Program of Computer Graphics

<http://www.graphics.cornell.edu/>

For more information visit:

<http://www.graphics.cornell.edu/~jaf>

Copyright 2005 James A. Ferwerda.

All Rights Reserved.

B.2 Magnofly Parameters

The size (in inches) of the user's monitor along the diagonal. Viewable Image Size: 19

The distance (in inches) away the user is from the screen. Viewer Distance: 24

The number of frames per second which the monitor can display. Monitor Refresh Rate (FPS): 60

The size of each fly cloud in terms of visual angle # (i.e. for a value 10, then the cloud subtends a 10 degree visual angle patch in BOTH X,Y directions). Fly Cloud Visual Angle: 10

Each fly's velocity in terms of visual angle per second Individual Fly Velocity: 7

Should we use visual angle to determine the size of each fly? ('true' is yes, 'false' is no) Use Visual Angle For Individual Flies?: false

The dimensions of each fly in terms of visual angle. This value is read if above boolean is 'true'. Fly Size (In Terms of Visual Angle): 0.2

The dimensions of each fly in terms of pixels (i.e. if 2, then each fly is 2x2 pixels). Fly Size (In Terms of Pixels): 3.5

The number of frames until a fly has to reset its position in the swarm. Fly Lifetime: 5

NOTE: For more information about the following four fly cloud structure parameters, see attached Matlab files. # The number of ellipses that form the bounding region of the swarm (i.e. flies are culled outside of this region). Preferably a multiple of four. Number of Bounding Ellipses: 40

Percent of fly cloud radius (in [0,1], 1 being the entire fly cloud radius) where flies are /guaranteed/ to not be removed for being 'outside the swarm'. Minimum Cloud Dimension: 0.86

Randomness factor (in [0,1], 1 being most random) of the clouds that form both the swarm's bounding region and reset locations for culled flies # (for either being older than their lifetime or escaping out of their bounding region. Cloud Randomness: 0.1

Randomness factor (in [0,1], 1 being most random) of the randomness of the spacing between clouds that form both the swarm's bounding region and reset locations for culled flies. Cloud Spacing Randomness: 1

A /relative/ scalar that adjusts each baby's size. Keep at 1 if you're happy with the default baby size Baby Size Scalar: 1

How quickly each baby goes through their movements. It is a multiple of FPS, # so if we want a baby that goes through its movements every 10 seconds, we set this to 10*(given FPS). Experimentally, [5*FPS,10*FPS] is a good range. Baby Movement Speed: 600

Toggles whether to not allow possible duplicate toy positions in two different sand boxes. 'true' is yes, 'false' is no. Pick Unique Toys For Each Sandbox: true

The number of flies in each fly cloud (150 in Stein paper) Number of Flies: 300

The method which QUEST uses to estimate the next threshold between trials.

Options:

0: Q_QUANTILE: Recommended by Pelli (1987)

1: Q_MEAN: Recommended by King-Smith et al. (1994)

2: Q_MODE: Recommended by Watson & Pelli (1983)

Quest Trial Choice: q-quantile

The method which QUEST uses to determine the final resultant threshold.

Options:

0: Q_MEAN: Recommended by Pelli (1989) and King-Smith et al. (1994)

1: Q_MODE: According to Pelli, similar to/better than Watson & Pelli (1983)

Quest Result Choice: q_mean

To initialize Quest, we need a guess threshold which it can start from. Quest - Threshold Guess: -6

The standard deviation for the user given guess threshold. Quest - Threshold Guess Standard Deviation: 8

A term that defines the psychometric function which Quest uses to find the threshold. # Try replacing this beta with the beta found with beta analysis at the end of a typical gamelog Quest - Beta: 3.5

Probability that user will identify coherent motion diagram incorrectly even at very high coherence levels. Quest - Delta: 0.01

Expected probability of choosing coherent motion at threshold. Given by $1 - (1 - \gamma) * e^{(-1)}$, where $\gamma = .25$, since the game uses a 4-alternative choice test. Quest - Probability Threshold: 0.72409042

Define the range (in decibels) plus and minus a given threshold for that round Threshold Deviation: 4

Percent threshold by which the decibels are relative to (see Quest source code for more information) Relative Threshold: 0.15

The space (in terms of visual angle) between the fly clouds. Space Between Fly Clouds: 2.5

Total number of trials for the game/experiment Number of Trials: 40

Total number of trials for the demo. Number of Demo Trials: 5

Instead of a gradually increasing coherence cloud between [mean # force the cloud to only have one coherence level? ('true' for yes, 'false' for no.) Enable 'Stein Mode': false

Have a visual cue appear during the visual cue period (see below)? ('true' for yes, 'false' for no.) Enable Posner Task: false

Percent chance (from [0,1]) that the Posner task cue is a valid cue (e.g. if a quad is coherently moving, the visual cue had appeared on it beforehand). Only applies when 'Enable Posner Task' is true. Valid Cue Percentage: 0.8

Make the babies and flies appear in random left/right positions after every round? ('true' for yes, 'false' for no.) Enable Random Round Robin: false

Use the exact threshold that the user clicked on during the interactivity period (not recommended)? ('true' for yes, 'false' for no.) Use User Clicked Threshold: false

Time period (in # of frames) allowed for round initialization/a fixation period (nothing occurs). Num. Frames for Round Stage- Initialization: 40

Time period (in # of frames) allowed for the fly emerging animation. Num. Frames for Round Stage- Fly Cloud Emerging: 40

Time period (in # of frames) allowed for a visual cue. Note: this is useless unless Posner Task is on. Num. Frames for Round Stage- Visual Cue: 0

Time period (in # of frames) reserved as a gap between the Posner task and coherence cycle. # Not used unless Posner task on, and make sure that it is LESS THAN 'Visual Cue' frames parameter above! Num. Frames for Round Stage- Visual Cue ISI: 20

Time period (in # of frames) allowed for the interactivity period (time when user has to click on the fly cloud). # Note: Parameter only read in if it's Stein Mode. Otherwise, it is one second per standard deviation (see Threshold Deviation) Num. Frames for Round Stage- Coherence Cycle: 80

Time period (in # of frames) allowed for the 'success' animation (the flies fall onto the ground). Num. Frames for Round Stage- Success: 80

Time period (in # of frames) allowed for the 'failure' animation (the flies converging on the unlucky baby). Num. Frames for Round Stage- Failure: 60

Play looping music during gameplay? ('true' for yes, 'false' for no.) Play Music: true

Play in-game sound effects? ('true' for yes, 'false' for no.) Play Sound Effects: true

Appendix C

Appendix C

Contents

Installation	3
Main menu overview	4
Motion test	5
Form fixed test	7
Form random test	8
Test Results	9
Extended Test Results	10
Settings menu overview	11
Motion Settings	12
Form Settings	13
Screen Settings	15
Staircase Settings	16
Input Settings	17

Figures

Fig 1.1 Main menu	4
Fig 1.2 Illustration of the motion test at 50% coherency	5
Fig 1.3 Screengrab from a motion test running with the default settings	6
Fig 1.4 Form fixed test in auto mode at 100% coherency	7
Fig 1.5 Form random test in auto mode at 100% coherency	8
Fig 1.6 Test results	9
Fig 1.7 Extended test results	10
Fig 1.8 Settings menu	11
Fig 1.9 Motion Settings	12
Fig 1.10 Form Settings	13
Fig 1.11 Form fixed auto mode at 100% coherency	14
Fig 1.12 Form fixed manual mode at 100% coherency	14
Fig 1.13 Screen Settings	15
Fig 1.14 Staircase Settings	16
Fig 1.15 Input Settings	17

INSTALLATION

All release versions are available from [here](#).

For Android, download [android-release.apk](#)

1. If you have previous version installed, go ahead and uninstall it.
2. Make sure that you have enabled “*Unknown sources*” under “*Settings->Security->Device administration*”.
3. Go to “*App drawer->Downloads*” and tap on “*android-release.apk*”.
(The file location might vary from device to device.)
4. Tap on “*Install*”.
5. The application can now be found under “*App drawer->Dyslexia Detection*”.
6. Tap the icon to run the application.

Optional:

3. Go to “*Settings*” and tap on “*Reset all settings*”
4. Tap “*Yes*” in the confirmation dialogue.

For a runnable Java file, download [java-release.jar](#)

Dependency: [Java SE](#)

1. Save the file to a destination you will find it again.
2. Double-click on “*java-release.jar*” to run the program.

Optional:

3. Go to “*Settings*” and click on “*Reset all settings*”
4. Click “*Yes*” in the confirmation dialog.

For a Macintosh version packaged with standalone Java, download [mac-release.zip](#)

Dependency: [Xquartz](#)

1. Save the file to a destination you will find it again.
2. Unpack the files to a destination of your own choice.
3. Double-click on the “*mac-release*” folder.
4. Double-click on the “*Contents*” folder.
5. Double-click on the “*MacOS*” folder.
6. Double-click on “*Dyslexia Detection*” to run the program.

Optional:

3. Go to “*Settings*” and click on “*Reset all settings*”
4. Click “*Yes*” in the confirmation dialog.

For a Windows version packaged with standalone Java, download [win-release.zip](#)

1. Save the file to a destination you will find it again.
2. Unpack the files to a destination of your own choice.
3. Double-click on the “*win-release*” folder.
4. Double-click on “*Dyslexia Detection(.exe)*” to run the program.

Optional:

3. Go to “*Settings*” and click on “*Reset all settings*”
4. Click “*Yes*” in the confirmation dialog.

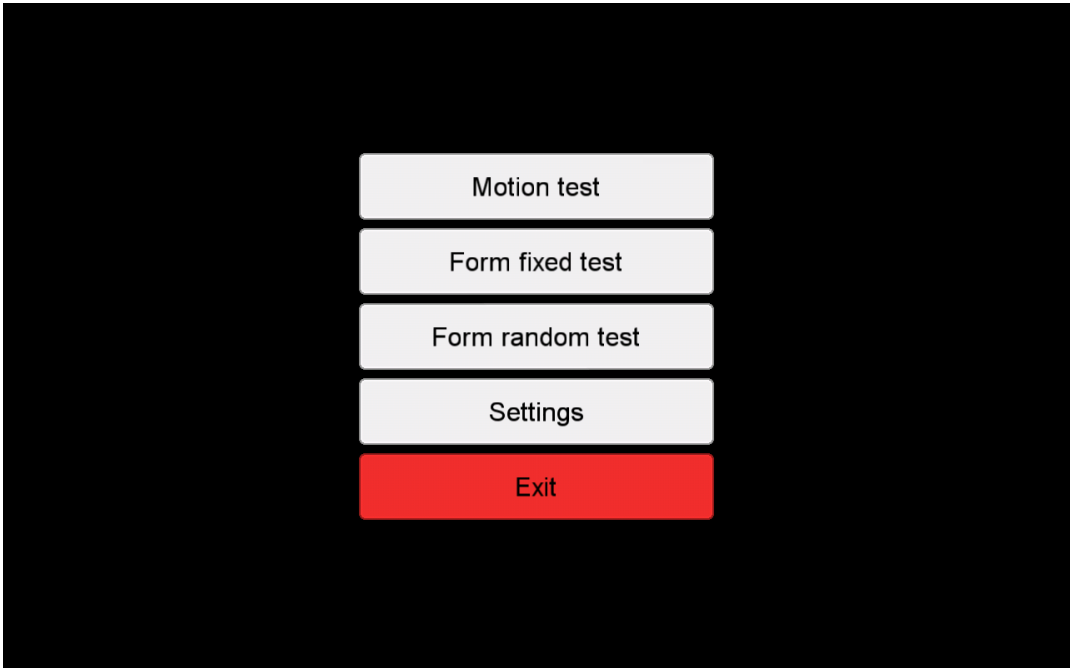


Fig 1.1 Main menu

MAIN MENU OVERVIEW

An overview of the main menu screen. Test and settings are described in the following pages. Check [screen settings](#) to make sure both patches are fully visible in the screen area before performing any tests.

Motion test	5
Form fixed test	7
Form random test	8
Test Results	9
Extended Test Results	10
Settings menu	11

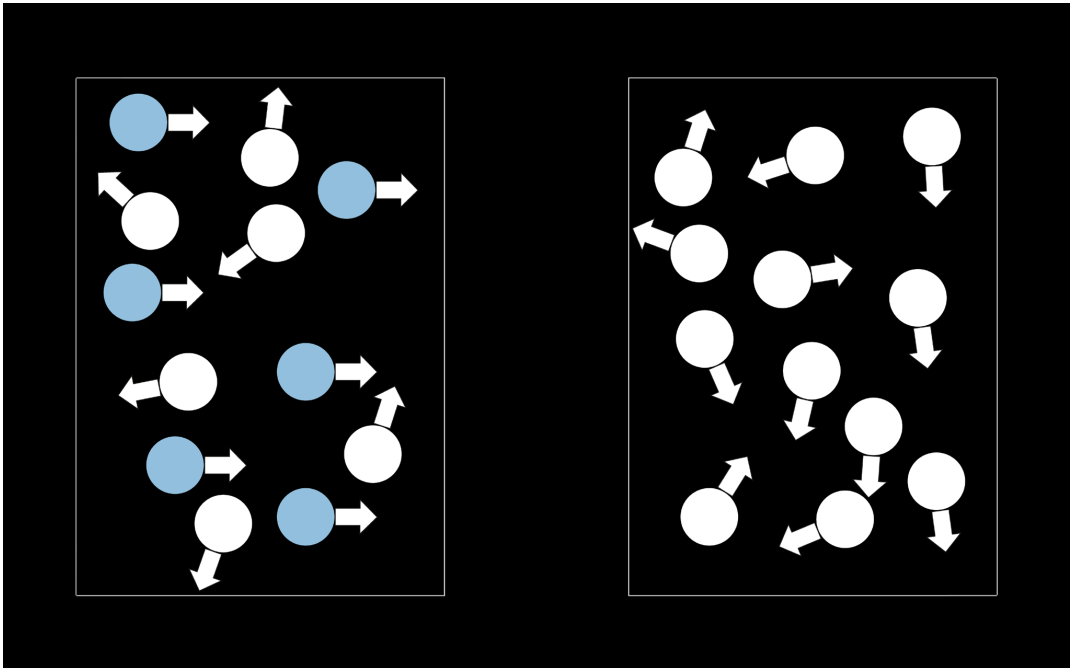


Fig 1.2 Illustration of the motion test at 50% coherency
Dots are not to scale, and are colored for this image.

MOTION TEST

The patches are drawn with the size $10^{\circ} \times 14^{\circ}$ (default) with a 5° (default) space between them. Each patch contains 300 randomly placed dots (default) with a radius of 1 pixel (default), and a minimum distance of 1 pixel (default) to other dots.

At each interval one of the patches are chosen at random to contain a coherent motion target. In that patch, a percentage (50% default) of the dots will move either leftwards or rightwards, reversing every 0.572 seconds (default). The dots not moving coherently, will move at random, changing direction when colliding with other dots and after 0.572 seconds (default). The dots move at a velocity of 50 pixels per second (default).

To prevent the test taker from following a single dot, 10% (default) of the dots are destroyed after 0.085 seconds (default), before they are repositioned and redrawn on the next frame. At 60 frames per second this means that 10% is destroyed every ~ 5 frames.

The test taker is to identify the patch with the coherent moving target during the 5 seconds (default) of animation time. After the dots disappear, the test taker can click on the patch they believe contained the coherent motion target, taking a guess when needed. As default, input is not registered before the interval time is over, this can be changed under [Settings->Input Settings](#). When input is registered the dots are recalculated with the change in coherency, and subsequently displayed on the screen.

The change in coherency follows a weighted staircase principle where each correct answer lowers the coherency by 1dB (default), equivalent to a factor of 1.222, and each wrong answer increases the coherency by 3dB (default), equivalent to a factor of 1.413.

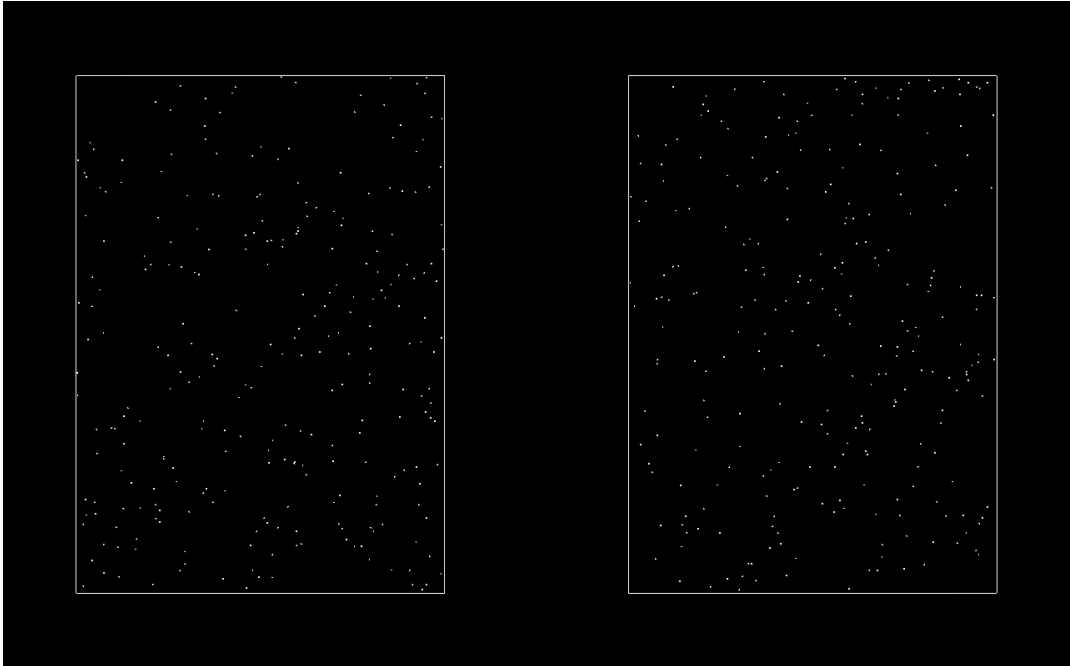


Fig 1.3 Screenshot from a motion test running with the default settings

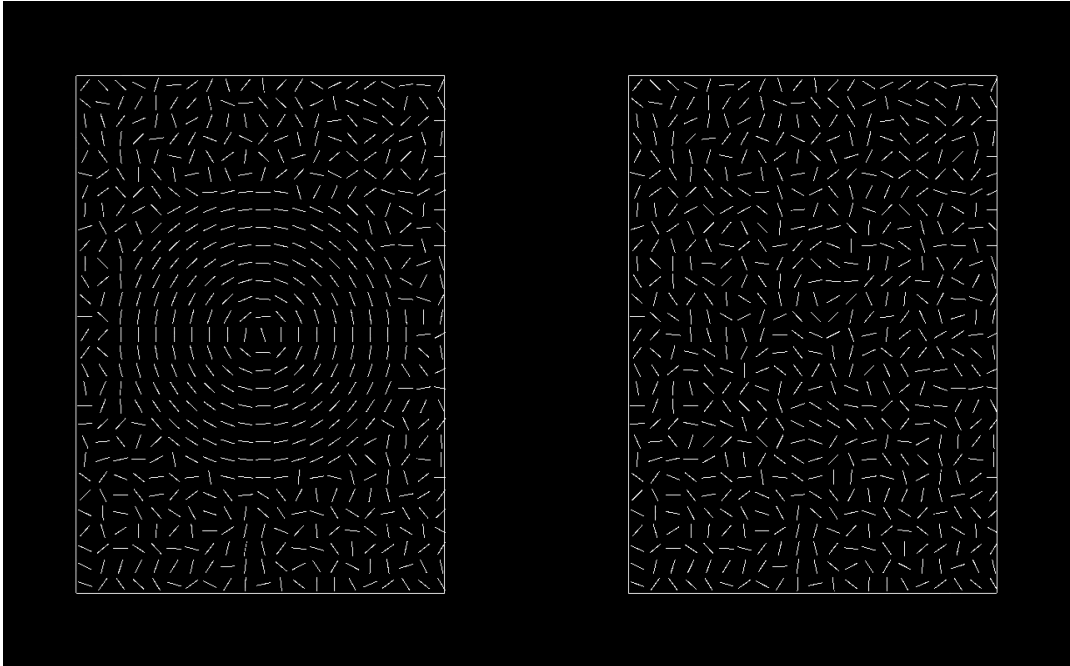


Fig 1.4 Form fixed test in auto mode at 100% coherency
See page [13](#) and [14](#) for the difference between auto and manual mode.

FORM FIXED TEST

The patches are drawn with the size $10^\circ \times 14^\circ$ (default) with a 5° (default) space between them. Each patch contains 600 line segments (default) each, with the lines having a length of 0.4° (default) and a height of 1 pixel (default).

At each interval one of the patches are chosen at random to contain concentric circles. Line segments within the area of an imaginary circle with the diameter of 8° (default) are oriented to the tangent of the imaginary circles center. The center of the imaginary circle will always be placed in the center of either the left or right patch.

The test taker is to search the patches during the 4 second (default) interval for the concentric circles. After the pattern disappears, the test taker can click on the patch they believe contained the concentric circles, taking a guess when needed. As default, input is not registered before the interval time is over, this can be changed under [Settings->Input Settings](#). When input is registered the pattern is recalculated with the change in coherency, and subsequently displayed on the screen.

The change in coherency follows a weighted staircase principle where each correct answer lowers the coherency by 1dB (default), equivalent to a factor of 1.222, and each wrong answer increases the coherency by 3dB (default), equivalent to a factor of 1.413.

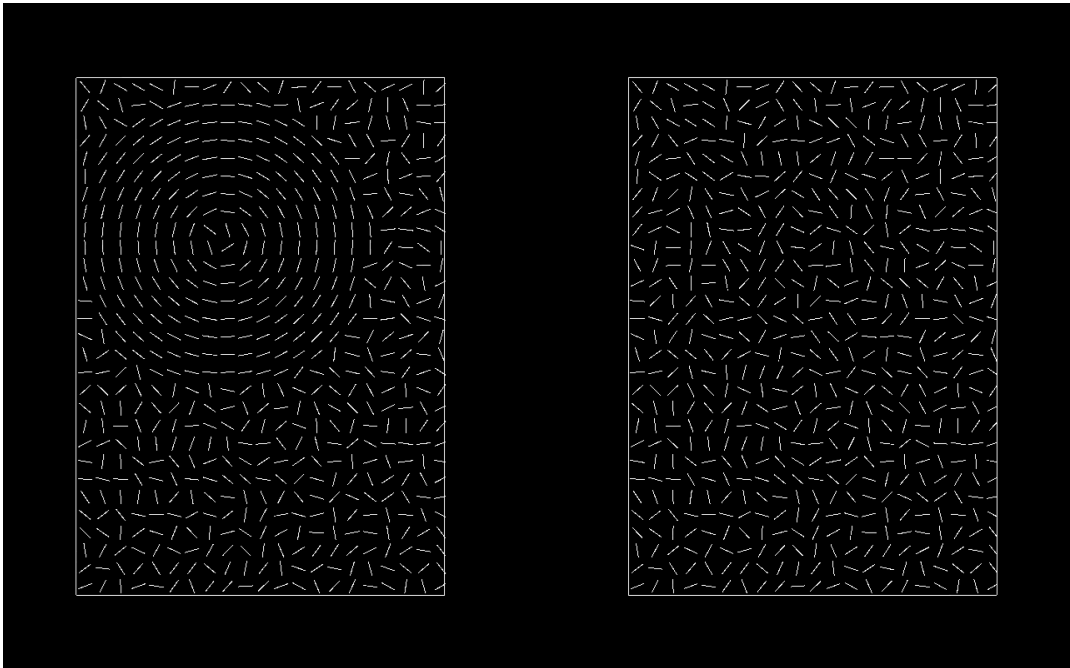


Fig 1.5 Form random test in auto mode at 100% coherency
See page [13](#) and [14](#) for the difference between auto and manual mode.

FORM RANDOM TEST

The patches are drawn with the size $10^{\circ} \times 14^{\circ}$ (default) with a 5° (default) space between them. Each patch contains 600 line segments (default) each, with the lines having a length of 0.4° (default) and a height of 1 pixel (default).

At each interval one of the patches are chosen at random to contain concentric circles. Line segments within the area of an imaginary circle with the diameter of 8° (default) are oriented to the tangent of the imaginary circles center. The center of the imaginary circle will be placed at random in either the left or right patch, with its circumference confined within the patch.

In this test there is a long time constraint for each interval, due to the added difficulty of finding the target. The test taker is to search the patches during the 1000 seconds (default) interval for the concentric circles. At any time the test taker can click on the patch they believe contains the concentric circles, taking a guess when needed. When input is registered the pattern is recalculated with the change in coherency, and subsequently displayed on the screen.

The change in coherency follows a weighted staircase principle where each correct answer lowers the coherency by 1dB (default), equivalent to a factor of 1.222, and each wrong answer increases the coherency by 3dB (default), equivalent to a factor of 1.413.

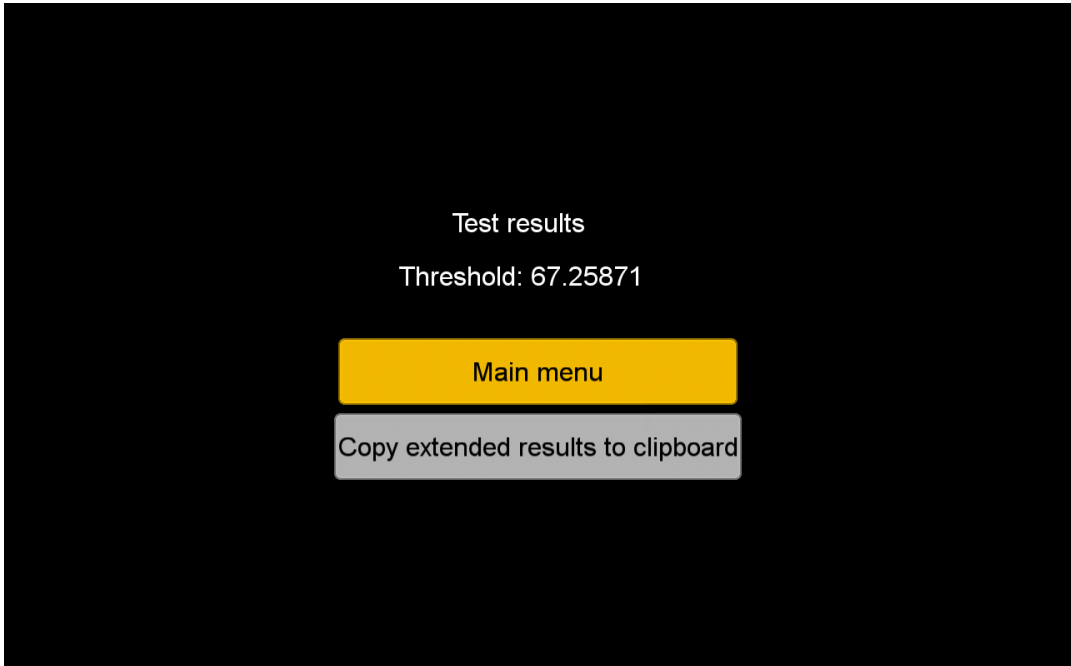


Fig 1.6 Test results

TEST RESULTS

Threshold is obtained by calculating the geometric mean of the last 8(default) reversal points in the staircase. Geometric mean¹ on the set of reversal points is given by:

$$\left(\prod_{i=1}^n rp_i \right)^{1/n} = \sqrt[n]{rp_1 rp_2 \cdots rp_n}$$

Hansen et al.² ran each test twice in similar program, then calculated the arithmetic mean to obtain the threshold value.

Arithmetic mean(average)³ is given by:

$$A = \frac{1}{n} \sum_{i=1}^n t_i$$

1. Wolfram MathWorld, Geometric Mean

2. Hansen, Peter C., et al. "Are dyslexics' visual deficits limited to measures of dorsal stream function?." Neuroreport 12.7 (2001): 1527-1530.

3. Wolfram MathWorld, Arithmetic Mean

EXTENDED TEST RESULTS

```
{
  correctAnswers: 12
  wrongAnswers: 17
  threshold: 89.237976
  lowestCoherency: 59.42122
  reversalValues: [
    1: 59.42122
    2: 100.0
    3: 77.08516
    4: 100.0
    5: 87.79816
    6: 100.0
    7: 77.08516
    8: 100.0
    9: 77.08516
    10: 100.0
  ]
  screenSettings: [
    screen_w_mm: 480
    screen_h_mm: 270
    screen_w_px: 1920
    screen_h_px: 1080
    viewing_distance: 300
    patch_width: 10
    patch_height: 14
    patch_gap: 5
  ]
  motionSettings: [
    dot_amount: 300
    dot_radius: 1.0
    dot_spacing: 1.0
    dot_velocity: 50.0
    dot_coherency: 50.0
    dot_animation_time: 5.0
    dot_max_life_time: 0.085
    dot_kill_percentage: 10
    dot_horizontal_reversal_time: 0.572
    dot_random_direction_time: 0.572
  ]
  formSettings: [
    form_auto_mode: true
    form_line_amount: 600
    form_diameter_wb: 8.0
    form_nr_of_circles: 4
    form_circle_gap: 0.9
    form_line_length: 0.4
    form_line_height: 1.0
    form_line_gap: 0.4
    form_coherency: 100.0
    form_f_detection_time: 4.0
    form_r_detection_time: 1000.0
  ]
  staircaseSettings: [
    stair_correct_db: 1.0
    stair_wrong_db: 3.0
    stair_max_tries: 100
    stair_reversal_points: 10
    stair_mean_from_last: 8
  ]
  inputSettings: [
    input_key_left: S
    input_key_right: L
    input_key_back: Escape
    input_continuous_mode: false
  ]
}
```

Fig 1.7 The printout of the copied extended test results

The extended results are formatted in Json with arrays holding information about reversal points and the settings used during the test.

CORRECTANSWERS: Number of correct answers.

WRONGANSWERS: Number of wrong answers.

THRESHOLD: The geometric mean of X last reversals.

LOWESTCOHERENCY: The lowest obtained coherency % achieved during the test.

REVERSALVALUES[]: An ordered list of the all the reversal points.

SCREENSETTINGS[]: The screen settings used during the test.

MOTIONSETTINGS[]: The motion settings used during the test.

FORMSETTINGS[]: The form settings used during the test.

STAIRCASESETTINGS[]: The staircase settings used during the test.

INPUTSETTINGS[]: The input settings used during the test.

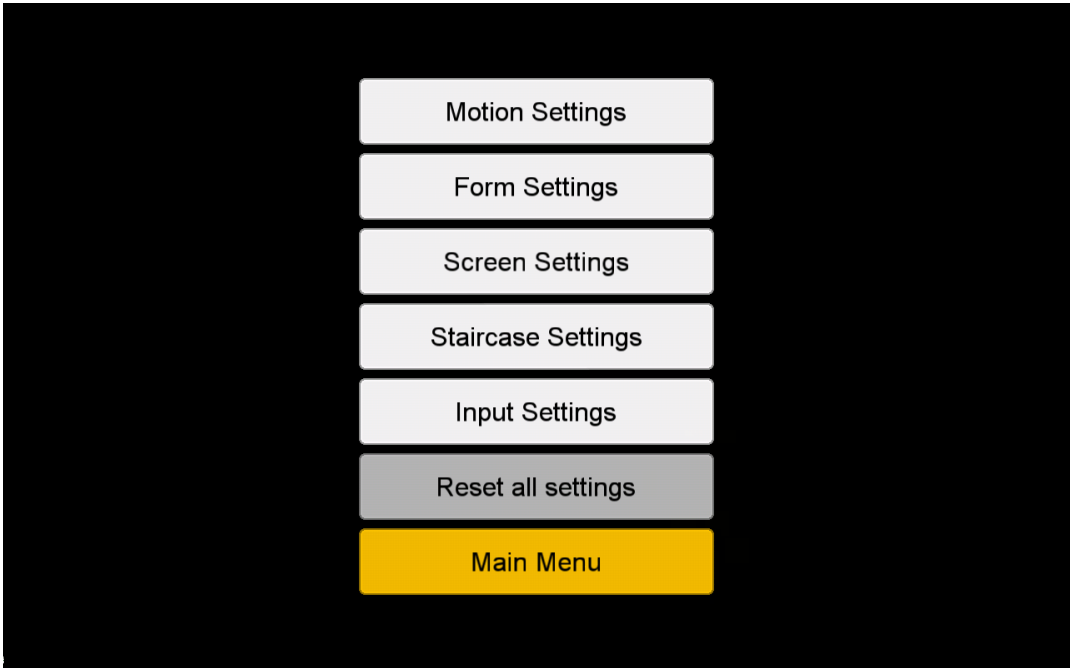


Fig 1.8 Settings menu

SETTINGS MENU OVERVIEW

Motion Settings	12
Form Settings	13
Screen Settings	15
Staircase Settings	16
Input Settings	17

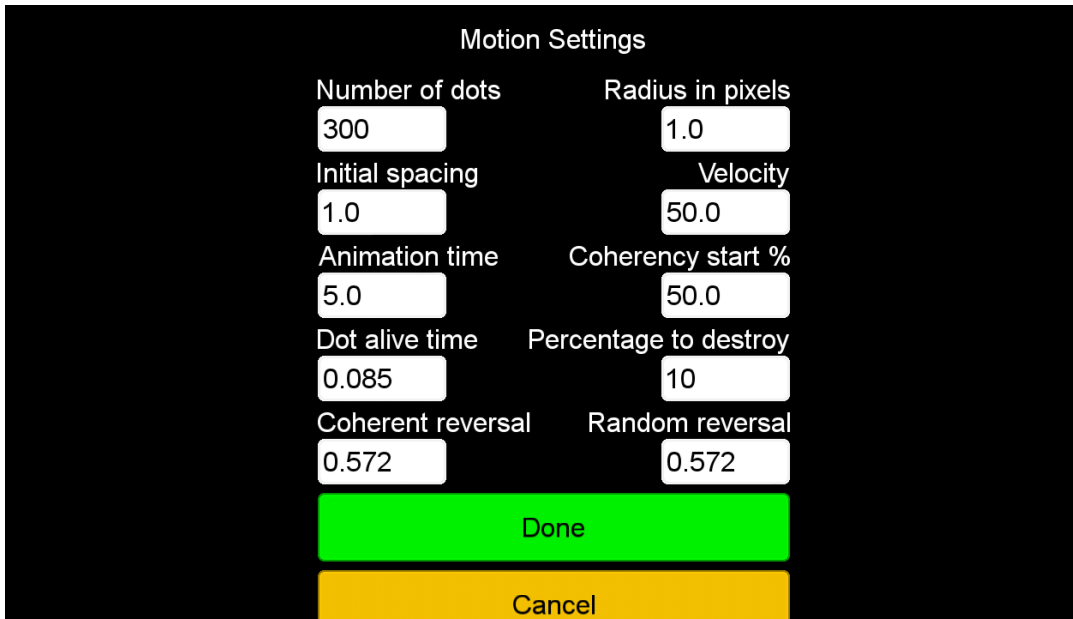


Fig 1.9 Motion Settings

MOTION SETTINGS

NUMBER OF DOTS: Amount of dots to be contained within each patch.

RADIUS: The radius for the dots in pixels.

INITIAL SPACING: How far from each other dots should have its starting position, i.e. all dots need to be at least X pixels away from each other at the start.

VELOCITY: The speed the dots are to move defined in pixels per second.

ANIMATION TIME (SECONDS): How long each interval lasts.

COHERENCY START %: The amount of dots, in percentage, 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 at random will change their direction of travel when time is reached. Direction of travel is changed by a random degree between 0 and 360.

Auto mode		<input checked="" type="checkbox"/>
Number of lines	<input type="text" value="600"/>	Diameter [°]
		<input type="text" value="8.0"/>
Nr of circles	<input type="text" value="4"/>	Circle gap [°]
		<input type="text" value="0.9"/>
Line length [°]	<input type="text" value="0.4"/>	Line height px
		<input type="text" value="1.0"/> ~0.0227°
Line gap [°]	<input type="text" value="0.4"/>	
Initial coherency	<input type="text" value="0.0"/>	
Fixed detection time	<input type="text" value="100.0"/>	Random detection time
		<input type="text" value="1000.0"/>

Fig 1.10 Form Settings

FORM SETTINGS

AUTO MODE: With Auto Mode checked, line segments are evenly distributed across the patches. Line segments inside the imaginary circle are oriented to its tangent. At 0% percent coherency, the left and right patch will look identical.

When Auto Mode is unchecked line segments are created at random within the patches. As coherence goes down, a portion of the line segments making out circles are relocated at random within the same patch.

See [next page](#) for screengrabs showcasing the difference between auto and manual mode.

NUMBER OF LINE SEGMENTS: Amount of line segments to be contained within each patch.

DIAMETER °: The diameter of the utmost concentric circle. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

NR OF CIRCLES: Number of concentric circles to be displayed. This field is updated when diameter or circle gap is edited. Evenly distributed concentric circles are calculated with a fault of +-1. You might have to add or subtract one to make the test look nice. This setting is disabled in Auto Mode.

CIRCLE GAP °: The diameter difference for each concentric circle. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case. This setting is disabled in Auto Mode.

LINE LENGTH °: The length of each line segment. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

LINE HEIGHT PX: The height of each line segment in pixels. Visual angle uses the height measurements of the screen to convert pixels to degrees in this case.

Line gap °: The distance between each line segment in the concentric circles. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case. This settings is disabled in Auto Mode.

FIXED DETECTION TIME (SECONDS): How long each interval should last in the Form Fixed test.

RANDOM DETECTION TIME (SECONDS): How long each interval should last in the Form Random test.

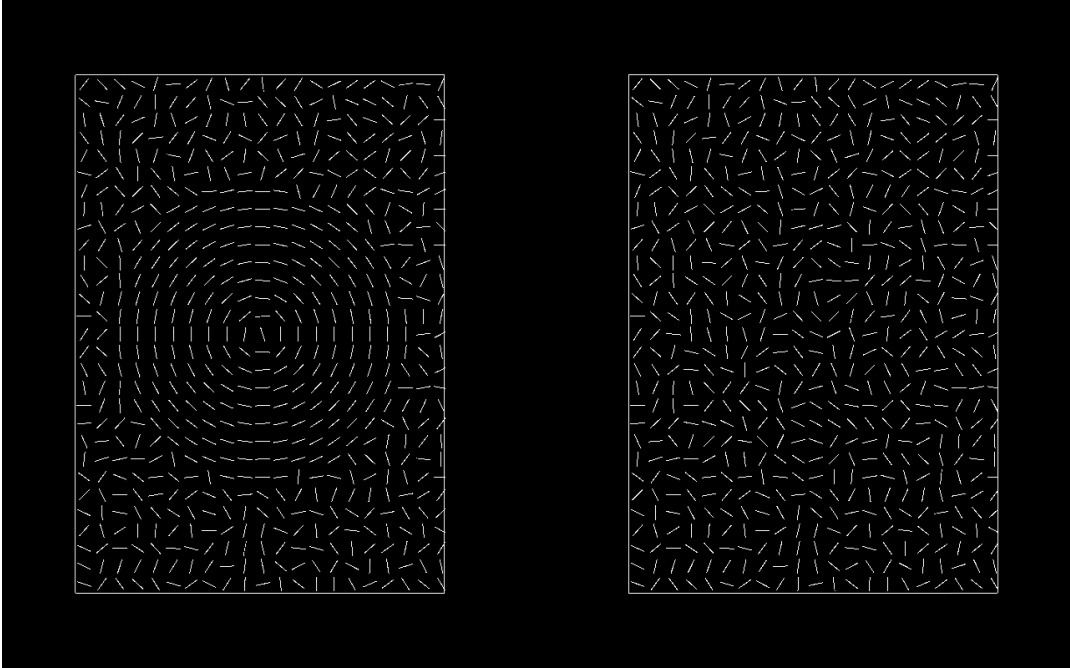


Fig 1.11 Form fixed auto mode at 100% coherency with the default settings

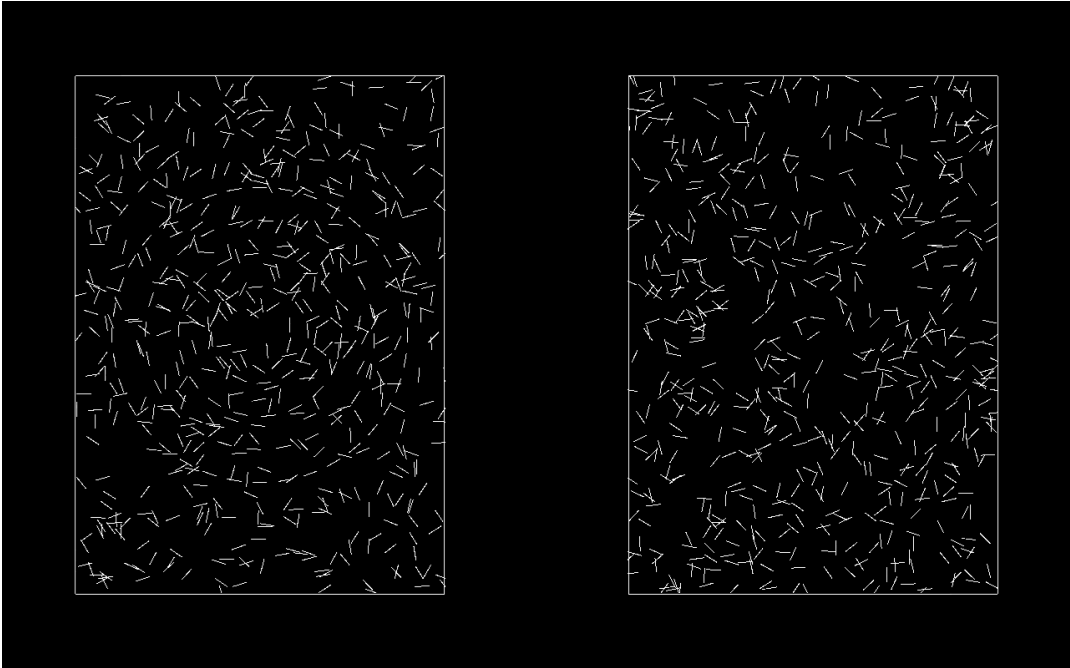


Fig 1.12 Form fixed manual mode at 100% coherency with the default settings

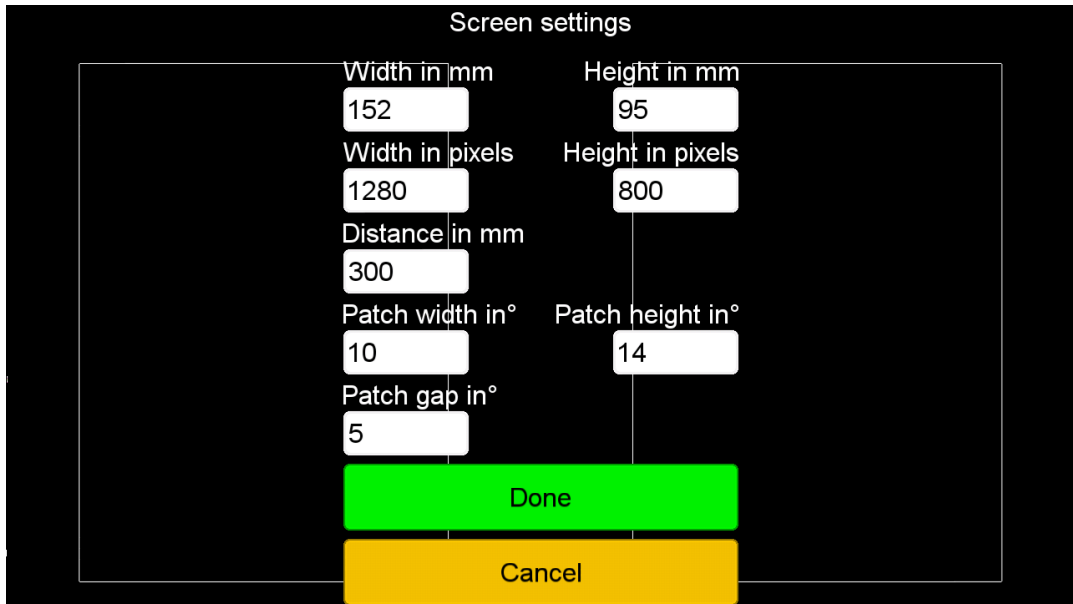


Fig 1.13 Screen Settings

SCREEN SETTINGS

Automatically calculated fields may have to be edited due to screen panels being larger than the viewable area.

WIDTH IN MM: The width of the viewable screen area in millimeters. This field is automatically calculated at first startup and when resetting all settings.

HEIGHT IN MM: The height of the viewable screen area in millimeters. This field is automatically calculated at first startup and when resetting all settings.

WIDTH IN PIXELS: The width wise pixel count of the screen. This field is automatically calculated at first startup and when resetting all settings.

HEIGHT IN PIXELS: The height wise pixel count of the screen. This field is automatically calculated at first startup and when resetting all settings.

DISTANCE IN MM: The viewing distance of the screen in millimeters. Measured from the eyes of the viewer to the center of the screen.

The patches are represented by the outlined rectangles and should fit within the screen boundaries. Updates when settings are applied.

PATCH WIDTH °: The width of each patch. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

PATCH HEIGHT °: The height of each patch. Visual angle uses the height measurements of the screen to convert degrees to pixels in this case.

PATCH GAP °: The gap between each patch. Visual angle uses the width measurements of the screen to convert degrees to pixels in this case.

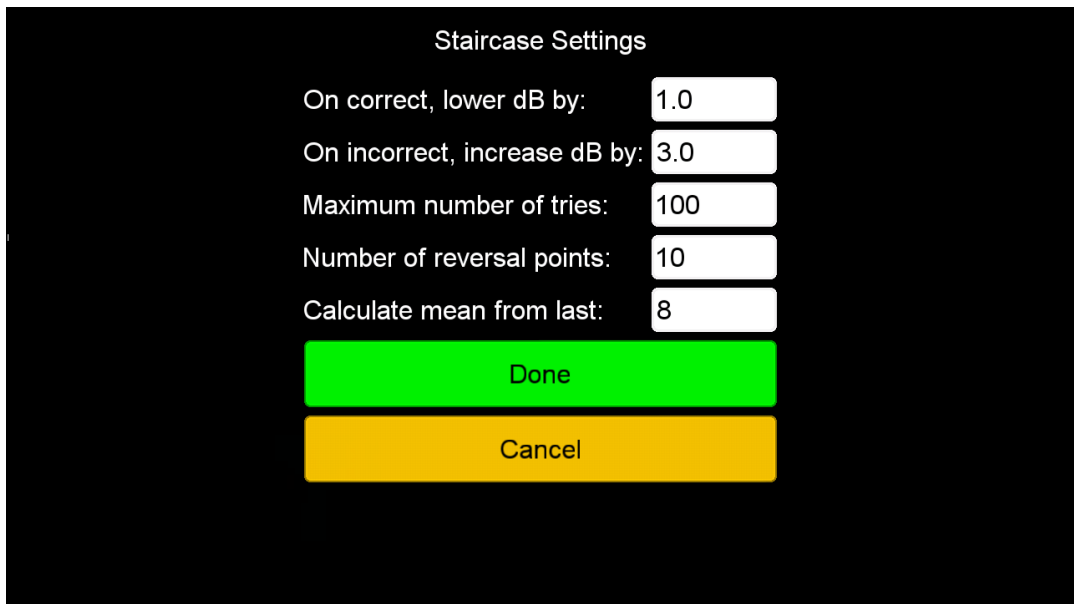


Fig 1.14 Staircase Settings

STAIRCASE SETTINGS

ON CORRECT LOWER DB BY: Uses the amplitude ratio of the decibel as a factor to lower coherency on correct answer. Can be negative or positive.

ON WRONG INCREASE DB BY: Uses the amplitude ratio of the decibel as a factor to increase coherency on wrong answers. Can be negative or positive.

See [Wikipedia](#) for a handy chart.

MAXIMUM NUMBER OF TRIES: Maximum number of tries for a given test.

NUMBER OF REVERSAL POINTS: Number of reversal points allowed before the test is ended.

CALCULATE FROM LAST: How many reversal points to use to calculate the geometric mean.

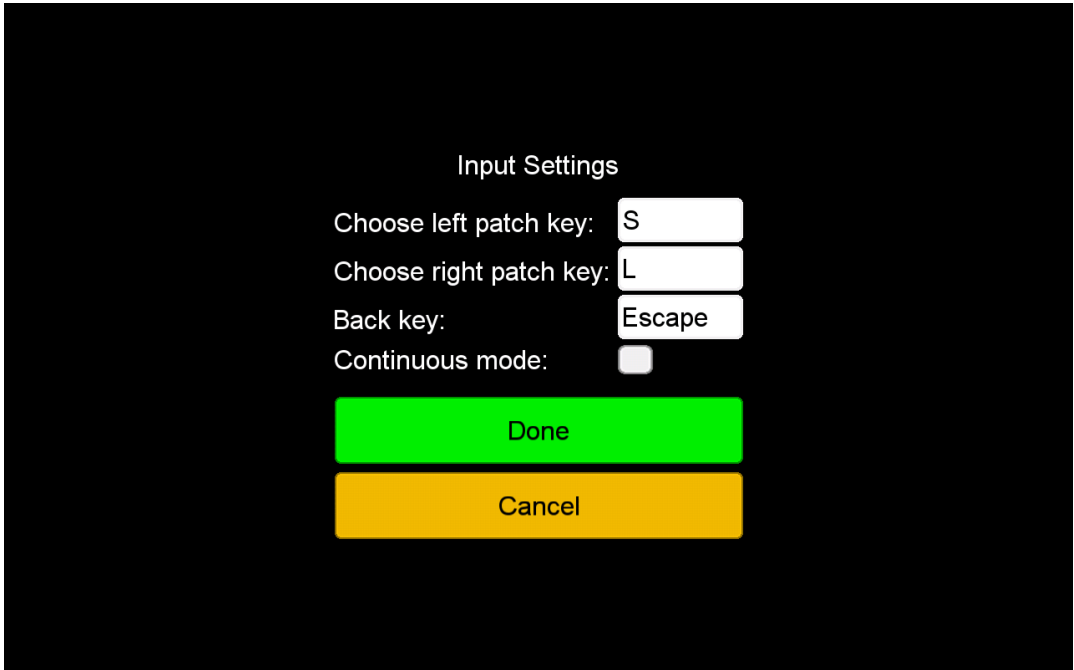


Fig 1.15 Input Settings

INPUT SETTINGS

When a keyboard is attached it is possible to assign keys for choosing a patch and for going back from the current to the previous screen in the application. By default these values are set as the above picture.

CONTINUOUS MODE: When checked it is possible to choose a patch before the animation time runs out. In this mode the content of the patches are recalculated and redrawn straight after the user makes a choice.

Bibliography

- [1] S. Hasle, *Kartlegging og utredning*. [Online]. Available: http://www.dysleksiforbundet.no/no/rettigheter+rad/Kartlegging+og+utredning.b7C_wlDSXd.ips (visited on 07/03/2015).
- [2] P. C. Hansen, J. F. Stein, S. R. Orde, J. L. Winter, and J. B. Talcott, "Are dyslexics' visual deficits limited to measures of dorsal stream function?" *Neuroreport*, vol. 12, no. 7, pp. 1527–1530, 2001.
- [3] G. Gescheider, *Psychophysics: The Fundamentals*. Taylor & Francis, 2013, ISBN: 9781134801299. [Online]. Available: <https://books.google.no/books?id=fLYWFcuamPwC>.
- [4] G. T. Fechner, "Elements of psychophysics, 1860," 2012. [Online]. Available: <http://dx.doi.org/10.1037/11304-026> (visited on 10/01/2015).
- [5] M. Reiss and G. Brooks, "Developmental dyslexia in adults: A research review," 2004.
- [6] C. Solem, *Definisjoner på dysleksi*. [Online]. Available: <http://www.dysleksiforbundet.no/no/rettigheter+rad/dysleksi/Definisjoner+p%C3%83%C2%A5+dysleksi.9UFRjM4P.ips> (visited on 11/02/2015).
- [7] T. Høien and P. Sundberg, *Dyslexia: From Theory to Intervention*, ser. Neuropsychology and Cognition. Springer Netherlands, 2013, ISBN: 9789401713290. [Online]. Available: <https://books.google.no/books?id=fWkcBQAAQBAJ>.
- [8] N. Gov, *Definition of dyslexia*, 2014. [Online]. Available: <https://www.nichd.nih.gov/health/topics/reading/conditioninfo/Pages/disorders.aspx> (visited on 11/06/2015).
- [9] B. D. Association, *Dyslexic definitions*. [Online]. Available: <http://www.bdadyslexia.org.uk/dyslexic/definitions> (visited on 11/03/2015).
- [10] National Institute of Neurological Disorders and Stroke, *Ninds dyslexia information page*, 2015. [Online]. Available: <http://www.ninds.nih.gov/disorders/dyslexia/dyslexia.htm> (visited on 11/02/2015).
- [11] Norsk Helsinformatikk, *Dysleksi*, 2014. [Online]. Available: <http://nhi.no/foreldre-og-barn/barn/livsstil/dysleksi-15298.html?page=all> (visited on 11/03/2015).

- [12] Science and Technology Committee - Second Report, *Evidence check 1: Early literacy interventions*, 2009. [Online]. Available: <http://www.publications.parliament.uk/pa/cm200910/cmselect/cmsctech/44/4402.htm> (visited on 11/03/2015).
- [13] B. C. Skottun, "The magnocellular deficit theory of dyslexia: The evidence from contrast sensitivity," *Vision Research*, vol. 40, no. 1, pp. 111–127, 2000, ISSN: 0042-6989. DOI: [http://dx.doi.org/10.1016/S0042-6989\(99\)00170-4](http://dx.doi.org/10.1016/S0042-6989(99)00170-4). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0042698999001704>.
- [14] M. Turatto, V. Mazza, S. Savazzi, and C. Marzi, "The role of the magnocellular and parvocellular systems in the redundant target effect," English, *Experimental Brain Research*, vol. 158, no. 2, pp. 141–150, 2004, ISSN: 0014-4819. DOI: 10.1007/s00221-004-1884-3. [Online]. Available: <http://dx.doi.org/10.1007/s00221-004-1884-3>.
- [15] M. Livingstone, G. Rosen, F. Drislane, and A. Galaburda, "Physiological and anatomical evidence for a magnocellular defect in developmental dyslexia," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 88, no. 18, pp. 7943–7947, 1991, cited By 0. [Online]. Available: <http://www.scopus.com/inward/record.url?eid=2-s2.0-0025874742&partnerID=40&md5=aee434a2d18526b2bd8a84dc9319c7b0>.
- [16] P. Cornelissen, A. Richardson, A. Mason, S. Fowler, and J. Stein, "Contrast sensitivity and coherent motion detection measured at photopic luminance levels in dyslexics and controls," *Vision Research*, vol. 35, no. 10, pp. 1483–1494, 1995, ISSN: 0042-6989. DOI: [http://dx.doi.org/10.1016/0042-6989\(95\)98728-R](http://dx.doi.org/10.1016/0042-6989(95)98728-R). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/004269899598728R>.
- [17] J. B. Demb, G. M. Boynton, M. Best, and D. J. Heeger, "Psychophysical evidence for a magnocellular pathway deficit in dyslexia," *Vision Research*, vol. 38, no. 11, pp. 1555–1559, 1998, ISSN: 0042-6989. DOI: [http://dx.doi.org/10.1016/S0042-6989\(98\)00075-3](http://dx.doi.org/10.1016/S0042-6989(98)00075-3). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0042698998000753>.
- [18] New Zealand Council for Educational Research, *York assessment of reading for comprehension - australian edition (yarc)*, 2012. [Online]. Available: <http://www.nzcer.org.nz/pts/york-assessment-reading-comprehension-australian-edition-yarc> (visited on 11/03/2016).
- [19] Statped, *Språk 6-16 screeningtest av språkvansker*, 2015. [Online]. Available: <http://www.statped.no/Laringsressurs/Fag/Sprak-og-tale/Sprak-6-16-screeningtest-av-sprakvansker-for-barn/> (visited on 03/26/2016).
- [20] E. Ottem, *Fonologisk minne of spesifikke språkvansker*. [Online]. Available: <http://www.n-t-a-f.org/Fonologiskminne.pdf> (visited on 01/04/2016).
- [21] H. Sigmundsson, "Do visual processing deficits cause problem on response time task for dyslexics?" *Brain and cognition*, vol. 58, no. 2, pp. 213–216, 2005.

- [22] H. Sigmundsson, P. Hansen, and J. Talcott, “Do ‘clumsy’ children have visual deficits,” *Behavioural Brain Research*, vol. 139, no. 1, pp. 123–129, 2003.
- [23] H. Sigmundsson, S. Anholt, and J. B. Talcott, “Are poor mathematics skills associated with visual deficits in temporal processing?” *Neuroscience letters*, vol. 469, no. 2, pp. 248–250, 2010.
- [24] S. Mathot, *Visual angle*. [Online]. Available: <http://osdoc.cogsci.nl/miscellaneous/visual-angle/>.
- [25] N. Drakos, *Calculation of visual angle*, Revised by Jens Lippmann, Marek Rouchal, Martin Wilck and others. [Online]. Available: <http://www.yorku.ca/eye/visangle.htm>.
- [26] J. Ferwerda and B. Rehon, “Magnofly: Game-based screening for dyslexia,” *Journal of Vision*, vol. 7, no. 9, pp. 520–520, Jun. 2007, ISSN: 1534-7362. DOI: 10.1167/7.9.520. [Online]. Available: <http://dx.doi.org/10.1167/7.9.520>.
- [27] —, *Magnofly: Game-based screening for dyslexia*. [Online]. Available: http://www.cis.rit.edu/jaf/publications/magnofly_poster_vss07_v2.pdf (visited on 11/05/2015).
- [28] A. B. Watson and D. G. Pelli, “Quest: A bayesian adaptive psychometric method,” *Perception & psychophysics*, vol. 33, no. 2, pp. 113–120, 1983.
- [29] Android Open Source Project, *Android studio overview*. [Online]. Available: <http://developer.android.com/tools/studio/index.html> (visited on 03/02/2015).
- [30] Khronos Group, *The standard for embedded accelerated 3d graphics*. [Online]. Available: <https://www.khronos.org/opengles/> (visited on 06/28/2015).
- [31] M. Zechner, *Libgdx*. [Online]. Available: <https://github.com/libgdx/libgdx/wiki> (visited on 04/25/2015).
- [32] D. Ludwig, *Packr*. [Online]. Available: <https://github.com/libgdx/packr> (visited on 01/09/2016).
- [33] B. Insight, “The mobile application market,” *Retrieved April*, vol. 27, p. 2013, 2013.
- [34] A. I. Wasserman, “Software engineering issues for mobile application development,” in *Proceedings of the FSE/SDP workshop on Future of software engineering research*, ACM, 2010, pp. 397–400.
- [35] Apple Inc, *Exporting your app for testing*. [Online]. Available: <https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/TestingYouriOSApp/TestingYouriOSApp.html> (visited on 01/30/2016).
- [36] D. Schmidt, M. Stal, H. Rohnert, and F. Buschmann, *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*. John Wiley & Sons, 1996, ISBN: 0471958697.

- [37] C. Berchek, *2-dimensional elastic collisions without trigonometry*, 2009. [Online]. Available: <http://www.vobarian.com/collisions/2dcollisions2.pdf> (visited on 04/04/2016).
- [38] K. Landerl, A. Bevan, and B. Butterworth, “Developmental dyscalculia and basic numerical capacities: A study of 8–9-year-old students,” *Cognition*, vol. 93, no. 2, pp. 99–125, 2004, ISSN: 0010-0277. DOI: <http://dx.doi.org/10.1016/j.cognition.2003.11.004>. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0010027704000149>.
- [39] W. H. RIDDER III, E. Borsting, and T. Banton, “All developmental dyslexic subtypes display an elevated motion coherence threshold,” *Optometry & Vision Science*, vol. 78, no. 7, pp. 510–517, 2001.