# Digitalization of Visual Acuity Testing

Anders Ohlsson
Christian Johansson
Svante Nilsson

May 28, 2015

**Abstract**

Digitalization of Visual Acuity Testing is a project about digitally displaying and manipulating optician's eye charts. The charts are displayed on a digital screen with a lightweight server running on a Raspberry Pi and controlled over wireless network via an Android tablet. This solves several problems related to the distance from the patient to the chart, while also providing more ways for the optician to tailor the examination to the patient. The Digitalization of Visual Acuity Testing project has improved upon a proof-of-concept system that has been used at the Uppsala University Hospital to test whether or not digital screens are a viable alternative to physical charts. The project fulfills our goals and has performed well under testing ~~by us,~~ but more testing is needed to make sure it meets the demands of practicing opticians.

# Contents

# List of Figures

# Introduction

## 1.1 Background

The standard way for an optician to measure a patients visual acuity is to use charts printed with rows of continuously smaller symbols called optotypes (see 3.2.1), most often normal Latin letters. The patient reads the symbols on the chart in order until he or she can no longer correctly identify them. Each row on the chart has a pre-calculated number printed beside it, which will help the optician determine which lenses the patient should try next.

These numbers have been calculated from a set of standardized conditions [1]. For example there must be adequate lighting in the room so the patient can read the letters, but more importantly the charts are meant to be placed a fixed distance from the patient. For most charts, this distance is six meters.

These conditions cannot always be met. Examination rooms are not always large enough to take measurements from six meters away for example. In those cases, the reference numbers on the sheet are no longer accurate and the optician has to compensate manually [2]. If the standardized conditions cannot be met it can lead to irregularities or faulty measurements in some tests.

There are also several different variations on these eye charts. Different sets of optotypes, ranging from different subsets of Latin letters to more abstract symbols, have been developed for different types of patients, such as those who are analphabetic, or for small children. The layout of these charts also have variations, from the classic Snellen chart (fig. 2.1) that was developed in the 19th century, to the more modern ETDRS chart (fig. 3.3).

In a field so dependent on visual aids and equipment, it's easy to see that there are several possibilities for improvement and expansion. Modern technology can be used to improve these tests, and digitalization can simplify the way opticians work with diagrams.

## 1.2 Purpose

Having efficient and accurate systems is important in a medical environment, especially in a medical environment that handles tests where inaccurate our false results in some cases can cause accidents. Therefore it is important to have a well-built system that mitigates these problems. With the help of Professor Per Söderberg as our expert in the field of ophthalmology we will try to create a system that will act as a prototype for further future work.

Accuracy is not the only important aspect to think of when implementing a system. It is also important to think about expanding and improving on how the system can be used. To introduce new functionality that will make the life of the system's user a lot more simple and efficient is one of those important improvements.

These are a few of the aspects that we will to build a system around. A simple system that implements digital eye charts has already been built by professor Söderberg, though that system is not as sophisticated or expendable as needed. We ~~will~~ base some of our system's content upon how this other system is built. Mostly this will be using the layouts of the charts already created since those charts have been shown to work.

Using this other system as a basis for our own we build a system that:

- Has an application for the Android tablet with an improved user interface, both aesthetically and functionally.
- Controls a server that renders the eye charts on a screen over WiFi using the Android tablet.
- Can alter the scale of the rendered symbols so that the patient can be closer or further away from the screen.
- Has functionality to tailor the testing process by, among other things, changing the sets of symbols displayed or removing clutter from the charts.

## 1.3 Motivation

~~Building upon what is said in the last section we need to find a clear motivation on why this should be done.~~ A few things can be found in the working environment of the optician. Determining visual acuity is a very labor- and equipment-intensive process. It is a lot of manual labor with much time wasted when setting up and changing the equipment used. This work-load is

also increasing drastically with time. More and more people are experiencing eye problems, at all ages and in many cultures. [3] With more people needing to test their visual acuity, faster and more efficient methods needs to be used.

Another fact is that physical eye charts are not as accurate as they should be for accurate measurements. They are also big in size and static, static in this context because they are fixed and cannot be changed. Because the handling of printed charts is slow and not as accurate as needed there exists a need for improvement.

A digital eye chart has increased functionality, accuracy and efficiency which are all important aspects to approve upon. A digitalization also gives the user a greater possibility for extending the usage of the system.

## 1.4 Issues

There are several technical problems that we need to solve. We need to find a suitable computer system that will act as the server, its job being storing and presenting the desired images to the screen. Our current plan is to use a Raspberry Pi, since it is a cheap and compact alternative to a full-size computer that is more than powerful enough for our purposes. Though, since the Raspberry Pi's CPU is running on an ARM-architecture there will be a few compatibility problems. For example, the newly redesigned Java graphics library JavaFX, that is going to replace the older Swing library, has very bad support for ARM. Not being able to use the latest features can create problems in the future when some things get deprecated. If Java Swing would happen to become deprecated it should not be too difficult to transfer the code to a hopefully compatible Java FX version.

Designing the Android app for the tablet will also bring technical problems, both from the limitations of portable computers, and making our software adaptable enough to work on as many brands of tablets as possible. For example, the symbols we are to present are stored in the .SVG (Scalable Vector Graphics) format, which is not natively supported on Android. Because of this we must find and work with a licensed third-party software library to render them.

Finally, we have the issue of networking. A lot can go wrong when trying to control a program over a network and also trying to have the response be fast and accurate. Any problems between the controller and the server will make actions seem slow and inconsistent. It can be even worse when working on a wireless network as in our case. This is an equally important problem as the other ones, but there exists several great technologies helping to prevent these kind of issues that will be a great help to us.

# Problem Description

## 2.1 Problem 

There are several weaknesses with visual acuity testing today that we have worked on improving. Physical charts are often large and cumbersome to switch if the patient needs a chart with another type of optotypes. Since they are also designed for standardized distances and testing conditions, an optician needs to manually compensate for any deviations from the standard.

There are also other problems with using standardized charts, such as older patients getting lost among the columns and rows, and other patients memorizing the rows of letters to cheat. Giving the optician the option of using randomly generated charts and easily switching between specialized chart layouts can greatly increase the speed and accuracy of testing.

Another problem we faced is designing a user interface that can give the optician easy access to these specialized charts and layouts. An intuitive design so that it is easy to learn, and a user friendly design so that users don't accidentally make unintentional or irreversible changes.

## 2.2 Delimitations

Due to limits in time and manpower, this project is not intended to be a complete working system to be used clinically, but rather a working prototype that can be used to evaluate digital visual acuity charts. In order to achieve this, as much as possible of Java's and Android's built-in libraries will be used. Though as SVG-files are going to be used for the rendering of optotypes and neither Java or Android supports it natively third-party libraries will be used.

We will not evaluate a digital screen's effect on the eye compared to a physical chart nor analyze why an ETDRS or Axanivis chart should have the layouts they have. As both charts are made after a very strict specification, no new chart layouts will be made. Because any new charts we create would

be considered useless without backing research they could also be misleading and misused by future users that in the end, could result in patients getting an inaccurate result.

No security features, such as encrypted network traffic or protection against several users connecting to the same server at the same time, will be implemented. Instead, this is left to the user to make sure that a secure network is used. Since there is no patient data or other private information being transmitted or stored we don't see the need for secure connections in this project.

No automatic update system will be included, instead the user will have to manually update both the Android application and the server. ~~We reasoned that~~ automatic updating was not a core feature, and could therefore be saved for future work.

## 2.3 Requirements

For this project to be considered successful, the system must be able to display the chart chosen by the user. This means that the user have to be able to control which chart type to display on the monitor from an Android device. The types of charts that should be included are the four Axanivis charts, the upper, the lower, the one line and the single optotype chart. The user also has to be able to choose if the current chart should use the Sloan, Tumbling E, HVOT or LEA optotypes.

It is very important that the chart on the monitor displays all optotypes with the correct sizes and positions. In order to do this, the user must to be able control the distance from which the patient is watching the monitor. It is also very important that the correct sizes, in LogMAR, is displayed on the Android device.

The intended user for this system should not need any specific background knowledge about technology more than the basic use of Android device, for example being able to start the device, start an application and how to perform simple Android gestures. This does not mean that it has to be easy to install and set up, though it is a goal for future expansion.

The Android application should be very user friendly and follow the Android design principles set up by Google [4]. This makes the application behave similar to other Android applications which will result in users already familiar with Android, to get the expected result from their actions.

The system should also allow for automatic server detection so that users don't have to manually connect by entering IP addresses. This allows a quick and easy way to connect to a server. In order for the server detection to be

considered working properly, it should detect a server on a stable network atleast 90% of the time.

As this project only aims for a working prototype of the system, it is very important that it is easy to continue the development and add more features without rewriting large parts of the code.

## 2.4 Evaluation

In order to evaluate if all requirements have been met, the system will undergo a few tests. The goal of the first test is to make sure that the user actually can choose a specific chart with specific optotypes to be viewed from a specific distance. This is done by the developers going through all charts, with all types of optotypes and setting the distance for each one of them. To achieve this, table A.1 is used to systematically walk through every setting.

The second test is to make sure that the optotypes have the correct sizes on the monitor. This test will done by, outside of the system, calculating the expected sizes of the optotypes in centimeters. The calculated sizes will then be compared to the sizes output by the system and see if they match. If this is a match, the optotypes will be measured by a physical measurement tool in order to double check the result.

The third test will determine how user friendly the system is. This will be done by letting a user with no previous knowledge of the system, attempt to connect to a server and perform some basic tasks like setting a specific chart with a specific optotype or setting a specific distance (App. A.2). If the user testing has no previous knowledge of visual acuity charts, a brief explanation will be given so that the test doesn't fail because the user, for example, doesn't know what "HVOT" is.

The last test is a quantitative test to see how well the server detection performs. The test involves the Android application trying to detect a server on a stable WiFi a 100 times and see how many times the server was detected. This test can be done several times and the average number of detection's in a 100 tries will determine its success rate.

## 2.5 Related work

### 2.5.1 Visual acuity charts

Several different versions of visual acuity chart exist today. The ETDRS (Early Treatment Diabetic Retinopathy Study) chart is the most commonly used today and is an improvement to the older *Snellen chart*(fig. 2.1). The ETDRS chart was designed and developed by Ferris et al [5], based on the work of Bailey and Lovie [1].

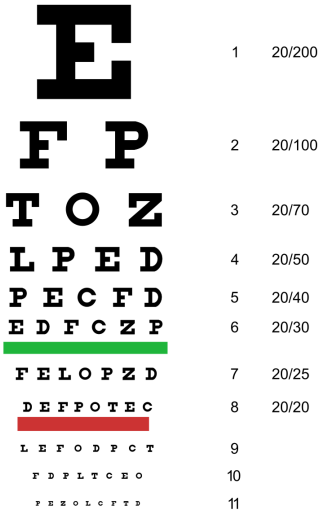



Figure 2.1: The Snellen chart
(Source: http://en.wikipedia.org/wiki/Snellen_chart).

### 2.5.2 Digital visual acuity charts

Digital Eye Chart [6] and iChartPlus [7] are two applications supposed to display a visual acuity chart on a digital screen. Both applications are computer applications, which requires the computer to be connected to the screen. The companies behind the products provide very little information and the exact specifications of the charts are not available.

### 2.5.3 Media streaming devices

Several media streaming devices exists that allow digital content to be received and displayed on a connected screen. Examples of such devices are

Chromecast [8] and Apple TV [9] which you can plug into a TV and allows you to view images or watch movies that are stored on your phone or computer. Both applications also allows developers to develop their own applications to display custom content but requires the developer to enter an agreement with respective company.

# Visual Acuity Testing

## 3.1 Visual Acuity

The measurement of a persons visual acuity has an important constraint that needs to be followed. This constraint is the fact that our eyes works best physiologically from greater distances. On the other hand, at shorter distances our eyes tend to make what we see look unfocused. To make up for this shift in focus a minimum testing distance has to be enforced to ensure that the shift in focus stays as small as possible. This distance has by scientific research been set to be six meters.

### 3.1.1 Measurement distance

The specific distance of six meters is derived from how our eyes bend light. From that distance or greater the rays of light entering our eyes are nearly parallel to each other, parallel enough to make it unnoticeable. It is important that the rays of light that hit the eyes are parallel because then the light hits a more focused area in the back of the eyes. This happens because of the convex shape of the eye's lens. A convex shaped lens focuses parallel light onto the focal point. Since a convex lens focuses parallel light to a focal point a concave lens instead spreads light coming from a focal point to parallel rays. Combining these two type of lenses we can create a more focused light.

### 3.1.2 Defocus correction

To make up for the shift in focus a suitable lens is put in front of the eye at about the distance that you would wear glasses. The type of lens mostly depends on the shape of your eye, any eye disease you might have or the distance from the lens to the eye. An important property of the lenses is at what distance the focal point lays at. So, to create a more focused light, which is exactly what glasses do, we have to put a lens in front of the eye to

create more parallel light. These lenses can either be plus or minus lenses, either spreading or contracting the light.

In optician's terms a minus glass is a lens that focuses light coming further away (subtraction) and a plus glass is a lens that focuses light from nearer (addition). A real world example of this is glasses correcting near- and farsightedness.

### 3.1.3 Angular magnification

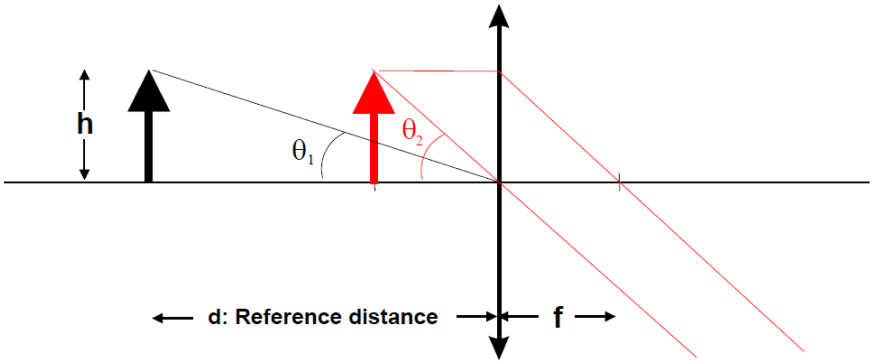



Figure 3.1: Angular magnification at short distances

Though using lenses is a way of corrected for testing at short distances, a few problems arise from this. Figure 3.1 shows the problem of angular magnification.

The magnification is a problem is visual acuity testing because if the optotypes on the eye chart are enlarged then an error will show up in the results.

| Distance to chart (m) | Addition (D) | Magnification (Rel.) | Error (logMar) |
|---|---|---|---|
| 4 | 0.25 | 1.5 | 0.17 |
| 2 | 0.5 | 3.5 | 0.48 |
| 1 | 1 | 6.0 | 0.78 |

Figure 3.2: Magnification due to addition for closer distance

There is need to have a standard to follow on how much addition or subtraction needed for a specific distance. Figure 3.2 shows what values to use when testing visual acuity at shorter distances than six meters. [2]

## 3.2 Chart Design

The visual acuity charts used in this project are based on ETDRS (Early Treatment Diabetic Retinopathy Study) charts (fig. 3.3) [5] and a project developed by P.G. Söderberg [10] at Uppsala University Hospital.

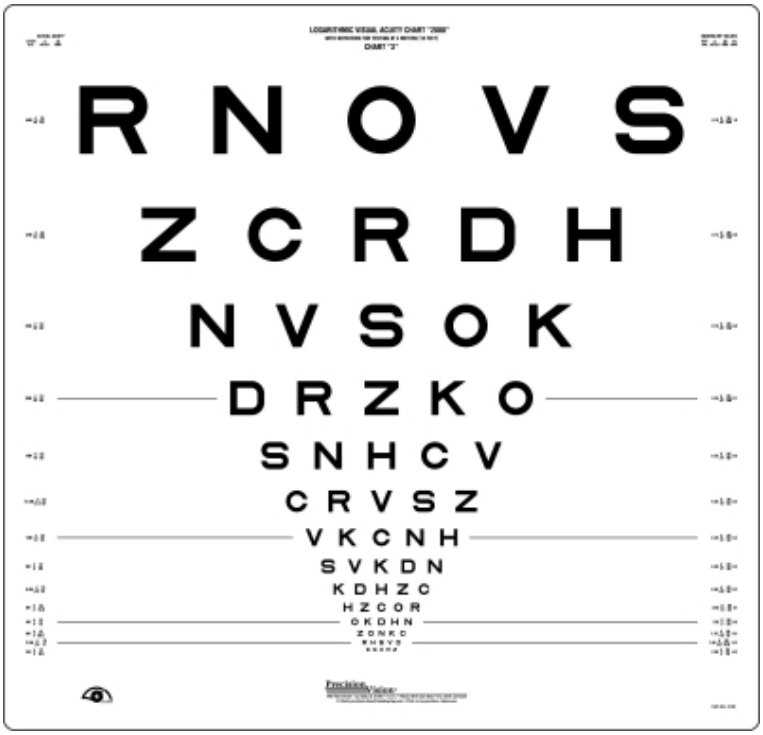



Figure 3.3: Example of a ETDRS chart
(Source: http://precision-vision.com/)

### 3.2.1 Optotypes

An optotype is a symbol that is used on a visual acuity chart. The optotypes are all of the same width and height and placed on a five by five grid. The most common type of symbol to use are letters from the Latin alphabet but can be different depending on the patient's needs [11].

*Sloan letters* are defined to be the ten optotypes *C, D, H, K, N, O, R, S, V and Z*. They have been chosen because they are easily differentiated from each other [5] and are used by most visual acuity charts [11].

*Landholt C* is a sloan *C* that is rotated by 0, 90, 180 or 270 degrees and is usually used in a more scientific environment. The Landholt C is used as reference when deciding the how difficult it is to read a specific optotype [11].

*Tumbling E* is like the Landholt C, but with the optotype *E* instead of *C*. This optotype can be used instead of the Sloan collection of optotypes if the patient doesn't know how to read. The patient can instead show or tell the optician in what direction the *E* is rotated [11].

*HVOT* are the letters *H,V,O* and *T*. These letters are chosen because they look the same if they are mirrored on the horizontal axis.

*LEA* are four symbols that look like a house, circle, box and apple. These symbols similar to HVOT since they look the same if you mirror them on the horizontal axis.

(a) Landholt C (b) Tumbling E (c) LEA

Figure 3.4: Some example optotypes

### 3.2.2 ETDRS Chart

The ETDRS chart consists of fourteen rows with five optotypes on each row. The size of each optotype is measured in LogMAR (Logarithmic of the Minimum Angle of Resolution) which is the base 10 logarithm of the angle of one fifth of an optotype (eq. 3.1 and fig. 3.5) [1]. The angle unit is measured in arcminutes, where one arcminute is $\frac{1}{60}$ degrees or $\frac{\pi}{10800}$ radians. If the angle v in figure 3.5 is equal to 8 arcminutes, then the size of the letter would be $log_{10}(8) \approx 0.903\ LogMAR$.

$$LogMAR = log_{10}(angle) \tag{3.1}$$

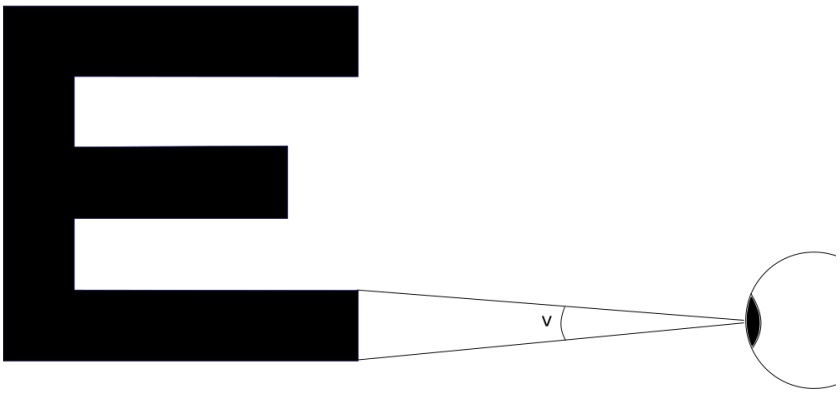



Figure 3.5: The angle v is used to calculate the size in LogMAR of the optotype.

The optotypes on the top row of the chart is of the size 1.0 LogMAR and then each row below is decremented by 0.1 LogMAR to the last row which is -0.3 LogMAR. This means that each optotype is approximately 1.2589 the height of the optotypes on the row below (fig. 3.7) [5]. The spacing between each optotype is equal to the width of an optotype from the same row and the spacing between each row is equal to the height of an optotype on the lower row (fig. 3.6).

### 3.2.3 Axanivis Charts

The Axanivis charts are an extension to the ETDRS chart that adds two new chart layouts and digitalizes them so they can be displayed on a screen.

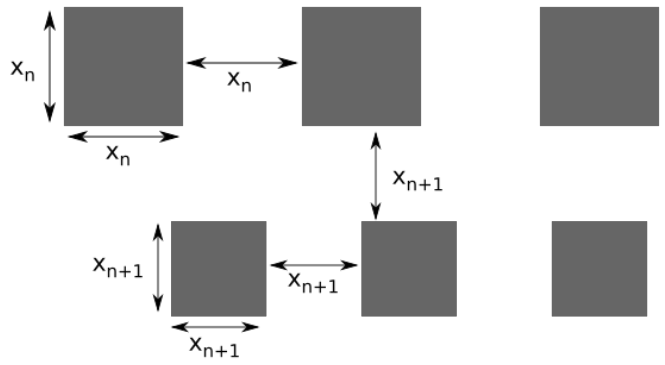



Figure 3.6: The sizes and positions of the first three optotypes of row $n$ and $n+1$ where $x_n \approx 1.2589 x_{n+1}$.

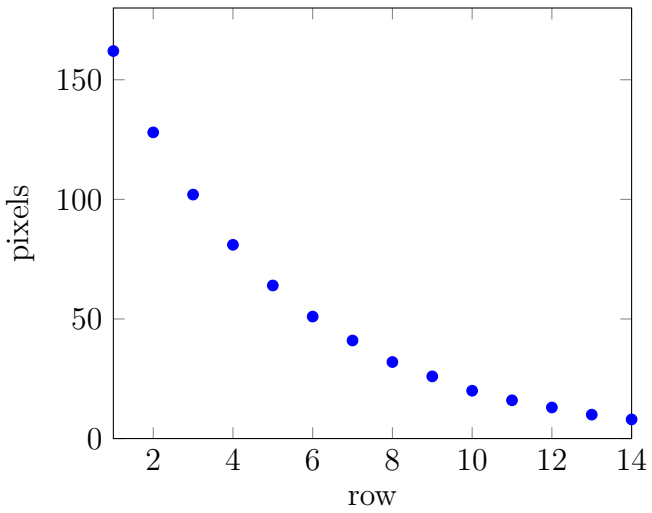
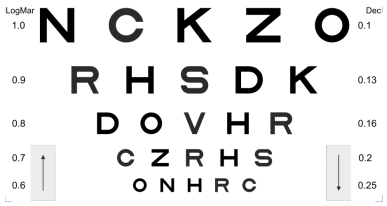


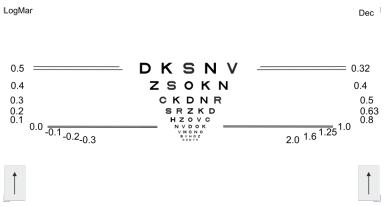

Figure 3.7: The height of optotype from 2 meters on a 15.6" screen with a resolution of 1920x1080 pixels.

Since the original ETDRS chart is made for a physical chart, it doesn't have an aspect ratio [5] that is close to a common monitor. To solve this, the original chart is split into an upper (fig. 3.8a) and a lower chart (fig. 3.8b). The upper chart display the five first rows and the lower chart display the remaining nine.

The two new chart layouts used is constructed using the same principles, like height and spacing of optotypes, as the original ETDRS chart. But




(a) Upper Chart




(b) Lower Chart

Figure 3.8: The upper and lower ETDRS charts

instead of having 14 rows of 5 optotypes each, the first new layout takes a single column of the ETDRS chart and rotates it 90 degrees (fig. 3.9) so it's all on one line. This makes it wider than it's high which suits a monitor better. This also allows the patient to read from left to right, instead of from top to bottom, which is the common way to read in Europe and America. Since the patient only have to read one letter of each size, this layout gives a quicker but more inaccurate result. This is useful to get a quick estimate on the visual acuity of the patient.

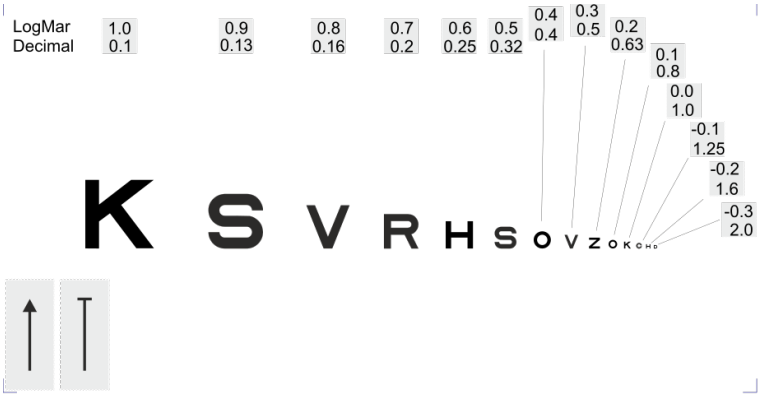



Figure 3.9: The one line chart

Once an estimated visual acuity have been achieved from the one line chart, the user can select an optotype and display the second new layout, which only display one optotype at a time (fig. 3.10). This layout removes all clutter and allow the patient to easily focus on only a single optotype at a time. The user can then page between the different optotype sizes in order to determine the visual acuity.

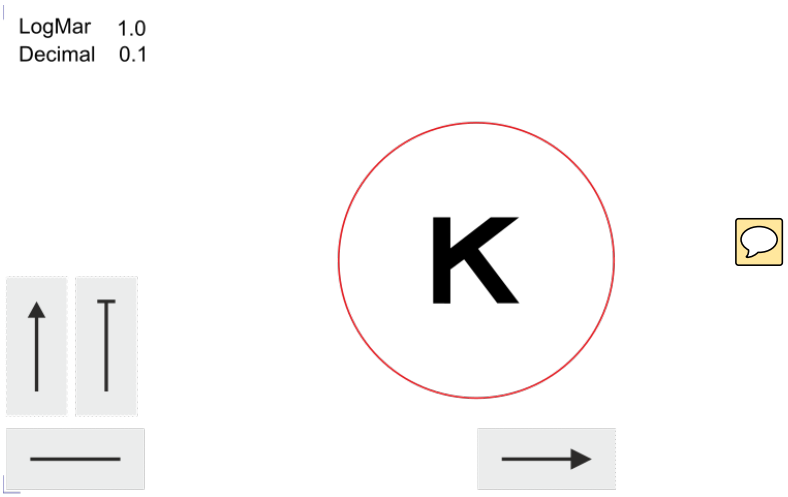



Figure 3.10: The single optotype chart

# Implementation

## 4.1 Methods

### 4.1.1 System structure

The system is divided into three major parts, an Android application, a server and networking. The Android application and server is modeled after the MVC (Model-View-Controller) pattern, where the model is the logic and the view is the GUI (Graphical User Interface). The controller part is the part on the Android application that listens for button clicks and gestures. Usually the networking would fall under controller in the MVC pattern but was, in this system, chosen to be a separate larger structure. Figure 4.1 shows how the different parts are connected.

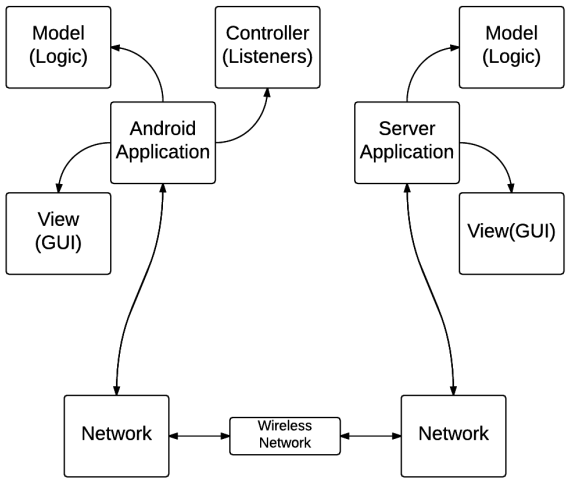



Figure 4.1: Overview of the system structure

### 4.1.2 Android application

The Android application is written in Java, which is the official language for Android. The static parts of the GUI, which will never change unless application updates are released, are written in XML. This is similar to how HTML is written and is supported by Android natively.

The charts shown on the Android device are generated at runtime. If the charts would have been written in XML, there would be no guarantee that they would fit on the screen or keep the correct proportions on tablets with different dimensions. It also avoids hard to read code as the large chart, for example, would need 70 defined areas for images, all with a unique text string id. The chart layouts are all drawn at the same time and in the same place, but sets the active chart visible and the rest invisible. As Android does not natively have support for vector graphics and the chart optotype images are vector images the AndroidSVG library is used to display them [12].

The size of the optotypes is calculated using equation 4.1, where $d$ is the distance in centimeters, $v$ is the angle in radians and $h$ is the height of the optotype in centimeters.

$$2d\ tan(\frac{v}{2}) = h \tag{4.1}$$

The settings associated to each chart are also generated at runtime. If this were not the case, every different version of the chart would need its own XML file, as each version contains a different number of predefined charts.

In order to control the application, *controllers* were created to listen to button clicks and gestures when the user attempts to do something.

The design of the application follows the Android design principles by using Androids built in navigation drawer and the built in views i.e. buttons, radio buttons, lists etc. [4].

### 4.1.3 Server application

The server application is written in Java and uses the built-in *Swing* library for drawing the GUI and the images. Swing is a very flexible and easy-to-use library with a lot of great tools for creating a good looking GUI. In our case we won't be needing a very advanced GUI since the main purpose of the server is to render a fullscreen image at all times. This greatly simplifies the implementation of the GUI. Other than the rendering of images, the GUI ~~will~~ consist of a small network controller where the user can specify a number of settings. These settings ~~will~~ be needed to configure for example what name

the server will have on the network. It's also necessary to have settings for the location of the server.

The rendering of images, specifically SVG vector graphic images, ~~will~~ be handled by an *Apache* library called *Batik*. It is well supported and extensive library for rendering a multitude of different file formats. The SVG files contain information about the optotypes used on the charts. Using procedural positioning of the SVG images, we can easily create a well structured image to render to the screen.

Network communication will be handled using a submit-poll system. The networking part of the server will receive information from the network and submit it to a thread-safe queue. While this happens, the GUI polls the information from the queue and handles the information. This methods is necessary because of how Swing is constructed. Swing runs on a dispatch-thread basis, meaning only the dispatch-thread itself can change things in the GUI. Trying to change things in the GUI from another thread would cause a lot of problems. Now the GUI can try to poll whenever it has time, though tries to do this periodically.

### 4.1.4 Network

The network communication between the client and the server consists of two parts. The first part is server detection and the second is server communication. All network communications are handled in threads separate from the main thread as the network operations would halt the flow of the program otherwise. Once the operations are complete and the main threads finds it convenient, it will grab the result from the network threads.

#### Server detection

In order to detect all servers on the network, the Android application sends out a discovery request message 5 times with a 1 second interval using UDP (User Datagram Protocol) on the networks broadcast address. The servers, which constantly listens for broadcasts containing the correct message, will then respond to that message with it's name, again using UDP. Once the Android application receives a response it adds the server to a list. The user can then choose to connect to one of the found servers or attempt the detection again if no servers where found.

Since UDP is a connectionless protocol, there is no guarantee that the server actually receives the message, the discovery request message is sent 5 times, which means the server have 5 chances to receive the message and respond to it. This means that the server will receive a discovery request 0

to 5 times and respond to it the same number of times. To avoid duplicate server entries in the list, a server is only added if it doesn't already exist in the list.

### Server communication

The communication between the Android application and the connected server is handled using strings of characters divided to three lines using line separators. The first line consists of a check word, which is the same for every message. If this check word is wrong or doesn't exist, the server will ignore the message. The second line is the command to the server. This could be a command to set the currents chart type or size to something else. The third line is the data required for the command. The data can be empty (null) or a string of characters. Depending on the command, the string of characters can be read as either a sequence of ASCII characters or a sequence of numbers.

The messages are sent using a TCP (Transmission Control Protocol) connection to the server, which guarantees that the entire message will be correct and that it will be retrieved by the server as long as the server exists on the network. If there are no servers available on the network, the user will be informed and have to reconnect to a server.

```
AXANIVIS
DISTANCE
500
```

Figure 4.2: An example of how a network message would look like.

# Results

## 5.1 System functionality

The system fulfills all specified requirements and allow the user to quickly set up a chart on the monitor. No system crashes have occurred during testing, and no bugs that causes the system to not work properly have been encountered. With a stable WiFi connection, the system runs smoothly without any lag and the server detection finds a server in under a second.

### 5.1.1 Android application

The Android application uses Android's standard GUI elements. To navigate between the connection settings and different type of charts, Android's built in navigation drawer is used (fig. 5.1). In order to reach the connection settings, the user must click the name of the connected server, in the case that no server is connected, it will, instead of the name of a server, say *Click to connect*.

Once the connection settings page have been reached, the user will be shown the connection status and a *scan network* button (fig. 5.2a) which can be used to scan the network for servers. Once the button is pressed, the button will be disabled and found servers will appear in a list below (fig. 5.2b). When the desired server have been found, the user can click on it to connect to it and once the entire scan is complete, the *scan network* button will be enabled again.

Once the application has connected to a server, the user can navigate to the chart that should be viewed on the screen. This is done by opening the navigation drawer, either by swiping from the left edge of the screen to the right, or by pressing the navigation drawer icon in the top left corner. When the navigation drawer is open, the user can choose between the *ETDRS* (Sloan letters), *Tumbling E*, *HVOT* or *Icons* (LEA) to display that chart (fig. 5.3).

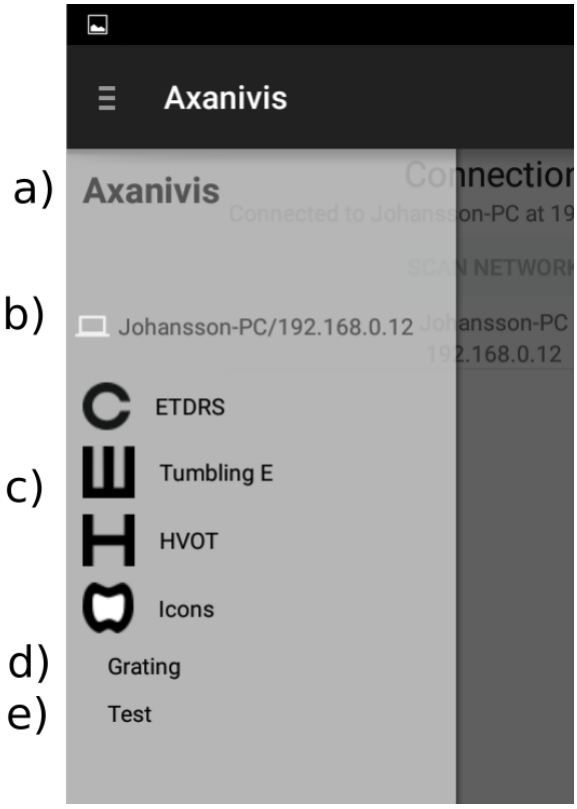
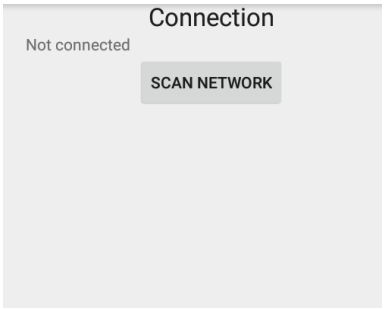


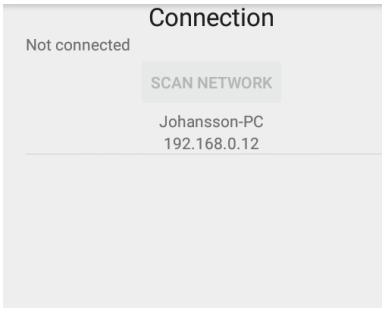

Figure 5.1: The Android application navigation drawer used for navigating the application. a) Name of the application b) The currently connected server c) Available optotypes d) Grating, not yet implemented e) Test, used for testing new functions

The user can now choose which chart version, size and distance to use. The chart version decides which optotype to use at which position. The ETDRS chart have 3 pre-made versions [5] and the rest of the charts have 1. There is also an option to use a randomly generated version which is generated when the application start. If this option is selected, the generate button below enables, so that the user can generate a new version if necessary. The random version is generated with no restrictions and should be used with caution.

The user can also choose which chart size, *Large*, *Medium* or *Small*, to use. The large chart is the same chart as the *upper* and *lower ETDRS* charts (fig. 5.3), the medium chart is the same as the *one line chart* (fig. 5.4a) and the small chart is the same as the *single optotype chart* (fig. 5.4b). To navigate between the upper and lower chart, the user must swipe the screen up or down and to page between the optotypes on the small chart, the user must




(a) No connection




(b) Scan button pressed


(c) Connected to server

Figure 5.2: Server connection steps

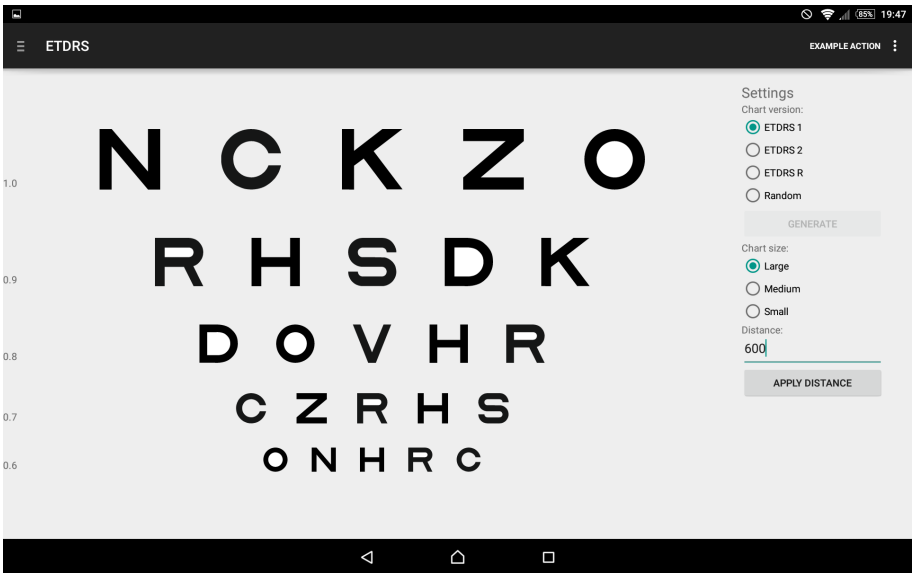
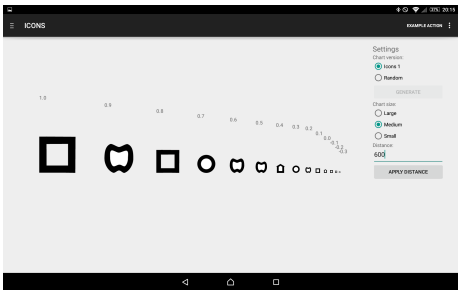



Figure 5.3: The upper large ETDRS 1 chart displayed with each rows LogMAR values to the far left.

swipe left or right.



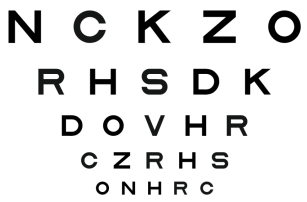

(a) The medium Icons 1 chart (b) The small Tumbling E 1 chart

Figure 5.4: A medium and small chart

The last setting the user can choose is the distance setting. Here the user can enter the distance a patient is from the screen, in centimeters, and then press *Apply Distance*. This has no visible effect on the application but is used to set the optotype sizes on the monitor.

### 5.1.2 Server

The server application is in many cases very similar to the Android application. Both use the same methods for the calculations, such as the optotype sizes. The charts are built using the same logic as the Android application with minor changes due to the fact that Java's Swing components have to be drawn rather than Android components. The monitor connected to the server only displays the currently chosen chart (fig. 5.5) and nothing else, such as LogMAR values or settings shown on the Android device.





(a) The large ETDRS chart (b) The medium ETDRS chart

Figure 5.5: Example of charts as shown on the monitor connected to the server

### 5.1.3 Known bugs and issues

Even though the system fulfills the requirements, it doesn't mean it's bug free. If the network is unstable or slow, the server can react several seconds later than the Android application. This can cause a user to repeat an action which will, in the end, result in the server acting on on both actions. In the case that the connection gets broken, all actions performed will be ignored by the server, even after reconnecting. In this case, the user can change to another chart size or version and then back again to resolve the issue.

## 5.2 Tests

The four tests presented in section 2.4 where performed in order to find major faults with the system. All tests is considered to have met the requirements, though the usability could still use some improvements.

### 5.2.1 Developer walkthrough

During the developer walkthrough of the system, all charts versions where found and all settings where usable on every chart by using table 5.1. The connection settings where also reachable from everywhere in the application and both connecting and reconnecting was working properly.

| Chart version | Large | Medium | Small | Distance |
|---|---|---|---|---|
| ETDRS 1 | ✓ | ✓ | ✓ | ✓ |
| ETDRS 2 | ✓ | ✓ | ✓ | ✓ |
| ETDRS R | ✓ | ✓ | ✓ | ✓ |
| ETDRS Random | ✓ | ✓ | ✓ | ✓ |
| Tumbling E 1 | ✓ | ✓ | ✓ | ✓ |
| Tumbling E Random | ✓ | ✓ | ✓ | ✓ |
| HVOT 1 | ✓ | ✓ | ✓ | ✓ |
| HVOT Random | ✓ | ✓ | ✓ | ✓ |
| Icons 1 | ✓ | ✓ | ✓ | ✓ |
| Icons Random | ✓ | ✓ | ✓ | ✓ |
| | Scanning | Connecting | Reconnecting | |
| Connection | ✓ | ✓ | ✓ | |

Table 5.1: Result of the developer walkthrough.

### 5.2.2 Optotype Sizes

The optotype sizes were calculated for and measured on a 15.6 inch computer screen and a 37 inch TV, both with 1920x1080 resolution. As the TV is larger than the computer screen, and have the same resolution, each pixel is bigger and thus, the calculated size in pixels is smaller (table. 5.2). The height in centimeters, rounded to two decimals, are both equal to the expected height and the test is considered successful.

| | 15.6", 1920x1080 | | 37", 1920x1080 | | Expected |
|---|---|---|---|---|---|
| row | px | cm | px | cm | cm |
| 1 | 162 | 2.91 | 68 | 2.91 | 2.91 |
| 2 | 128 | 2.31 | 54 | 2.31 | 2.31 |
| 3 | 102 | 1.84 | 43 | 1.84 | 1.84 |
| 4 | 81 | 1.46 | 34 | 1.46 | 1.46 |
| 5 | 64 | 1.16 | 27 | 1.16 | 1.16 |
| 6 | 51 | 0.92 | 22 | 0.92 | 0.92 |
| 7 | 41 | 0.73 | 17 | 0.73 | 0.73 |
| 8 | 32 | 0.58 | 14 | 0.58 | 0.58 |
| 9 | 26 | 0.46 | 11 | 0.46 | 0.46 |
| 10 | 20 | 0.37 | 9 | 0.37 | 0.37 |
| 11 | 16 | 0.29 | 7 | 0.29 | 0.29 |
| 12 | 13 | 0.23 | 5 | 0.23 | 0.23 |
| 13 | 10 | 0.18 | 4 | 0.18 | 0.18 |
| 14 | 8 | 0.15 | 3 | 0.15 | 0.15 |

Table 5.2: Height of optotypes for a 15.6 and 37 inch screen, both with a 1920x1080 resolution. The rightmost column displays the expected height of the optotypes in cm

### 5.2.3 Usability

The systems usability was tested with the help of two persons with no previous background related to visual acuity charts or eye examinations. After a brief explanation to the different types of charts they where asked to follow the procedures in A.2. Both persons had no problem performing any of the tasks with the exception of one of them didn't realize that you had to swipe the screen in order to switch between the upper and lower charts in the application.

### 5.2.4 Server Detection

The server detection test was done 3 times on a stable WiFi network. According to the results (Table. 5.3), every single one of the tests performed was successful in finding a server over 90% of the time. The average success rate of the 3 tests was about 96.3% which means that the server detection requirements are met.

| Attempt | failures | succeses |
|---|---|---|
| 1 | 7 | 93 |
| 2 | 1 | 99 |
| 3 | 3 | 97 |

Table 5.3: The number of failures and successes during the server detection tests.

# Conclusion

## 6.1 Discussion

Even though the system turned out as intended, there is a lot left to do until it can be used clinically. First of all, the system must be tested for more bugs and unintended behavior. This should preferably be done by users other than the developers since the developers know how the system works and can't safely predict what another user will attempt to do. Then, after all found bugs have been removed, the system must be tested by experts in the field of ophthalmology in order to make sure that the built system actually is a valid solution. This will also give credibility to the system.

## 6.2 Advantages of a digital system

The advantages of a digital system is the ability to change the layout, optotypes and size of the chart without having to manually exchange charts. This will save both time and space as only one screen and a small Android device is necessary, while also providing easier access to specialized charts that will help patients with different needs. This system will also allow the visual acuity tests to take place in rooms that previously where too small, as the system will display optotypes at a size suitable for the distance the patient is from the monitor.

## 6.3 Disadvantages of a digital system

A disadvantage with this system is that digital screens become very expensive at large sizes, which limits the maximum distance a patient can sit from the screen. Another disadvantage is that the resolution, or rather pixel density, will limit how close to a screen a patient can sit without the optotypes looking pixelated. Pixel density is a measurement of how many pixels (dots) can fit

in a small section of the screen, usually denoted as Pixels Per Inch (PPI) or Dots Per Inch (DPI). The higher the DPI is the better the quality of small details will be. This is very important when rendering type fonts because the edges should be as smooth as possible.

### 6.3.1 Alternative solutions

A possible alternative solution to a custom server running on a Raspberry Pi or other computer use a Chromecast instead. This could simplify the system for the user as this would make the Android application work similar to the many media streaming applications that exists today. Using a Chromecast would also make sure the user only have to worry about the Android application as the Chromecast is just a plug and play unit maintained by Google. The disadvantages to this is, as mentioned earlier, that an agreement with Google have to be signed. The application would also require an internet connection, opposed the the current system which only need a local network.

## 6.4 Future Work

Future work will include more charts and features for the system. One such feature is the grating acuity test, which allows testing of visual acuity on very young children or people with difficulties communicating [10]. Another feature is to include the option to, instead of changing the size of the optotypes, change the LogMAR, so the top row doesn't necessarily start at 1.0 LogMAR. This will allow the chart to use the entire monitors screen no matter the distance from the patient to the screen.

Other charts to be included could be, for example, charts using the Landholt C, charts for testing color blindness or charts used for testing astigmatism. The random function for the charts should be improved, as it currently have no constraints whatsoever. These constraints would make sure that the same optotype, or to similar optotypes like $C$ and $O$, won't be repeated to many times in a row.

# References


[1] I. Bailey and J. Lovie, "Design principles for visual acuity letter charts," *American Journal of Optometry & Physiological Optics*, vol. 53, 1976.

[2] P. Söderberg, "Impact of magnification when measuring va at shorter distance than 6 m."

[3] F. V. Jinan B Saaddine, K.M Venkat Narayan, "Vision loss: a public health problem?" 2003.

[4] Google, Inc. (2015) Android design principles. Visited on 2015-05-22. [Online]. Available: http://developer.android.com/design/get-started/principles.html

[5] L. Ferris III, A. Kassoff, H. Bresnick, and I. Bailey, "New visual acuity charts for clinical research," *American Journal of Ophthalmology*, vol. 94, 1982.

[6] digitaleyechart.com. (2015) Digital eye chart. Visited on 2015-04-27. [Online]. Available: http://digitaleyechart.com/

[7] iChartPlus. (2015) ichartplus. Visited on 2015-04-27. [Online]. Available: http://ichartplus.com/

[8] Google, Inc. (2015) Chromecast. Visited on 2015-04-27. [Online]. Available: https://www.google.se/chrome/devices/chromecast/

[9] Apple, Inc. (2015) Apple tv. Visited on 2015-04-27. [Online]. Available: https://www.apple.com/se/appletv/

[10] Söderberg, PG., "Ophthalmology," 2015, dept. of neuroscience, Uppsala University.

[11] A. Colenbrander, "Measuring vision and vision loss," 2001.

[12] paul.leb...@gmail.com. (2015) Androidsvg. Visited on 2015-05-11. [Online]. Available: https://code.google.com/p/androidsvg/

# Testing

## A.1 Developer walkthrough

| Chart version | Large | Medium | Small | Distance |
|---|---|---|---|---|
| ETDRS 1 | | | | |
| ETDRS 2 | | | | |
| ETDRS R | | | | |
| ETDRS Random | | | | |
| Tumbling E 1 | | | | |
| Tumbling E Random | | | | |
| HVOT 1 | | | | |
| HVOT Random | | | | |
| Icons 1 | | | | |
| Icons Random | | | | |
| | Scanning | Connecting | Reconnecting | |
| Connection | | | | |

Table A.1

## A.2 Usability test procedure

1. **Start application**
2. **Connect to server**
3. **Display ETDRS 1 Large upper chart with a distance of 2 meters**
4. **Display ETDRS 1 Large lower chart with a distance of 2 meters**
5. **Display HVOT Random chart medium with a distance of 1.7 meters**
6. **Generate a new HVOT chart**
7. **Display Icons 1 small chart with a distance of 1.5 meters**
8. **Page through Icons 1 small chart with a distance of 1.5 meters**
9. **Reconnect to server**