ICVGoggles: Wearable Personalised Simulations of Impaired Colour Vision

Babak Momen

AC40001 Honours Project  
BSc (Hons) Applied Computing   
University of Dundee, 2016  
Supervisor: Dr. David Flatla

**ABSTRACT**

Colour is used to tie in closely with a specific meaning, signal or message; however it is mostly used for aesthetics. People with Impaired Colour Vision (ICV) come across challenges every day to distinguish between colours, designers must consider users with ICV since information can be misinterpreted or even missed. Current software and hardware solutions provide real time simulations of various spectrums of impaired colour vision. This project aims to go above and beyond current technologies to provide designers with adjustable simulations viewed with a hands-free Oculus Rift headset.

# INTRODUCTION

## Background

Most cases of Impaired Colour Vision (ICV) are hereditary meaning it is passed on genetically from parent to offspring. They can also occasionally be acquired as a result of certain eye diseases. Failing to discriminate between red and green is the most common form of ICV (Protanopia / Deuteranopia) and the gene is X- linked recessive which explains the prevalence difference between genders (8% in males and 0.5% in females). Blue-yellow ICV or Tritanopia is rare and tritanomalous symptoms are more commonly acquired from environmental factors such as age, where the eye lens becomes increasingly yellow over time, cataracts or trauma to the front or the back of the head. Monochromacy is even rarer and affects two or more types of cone in the eye. Colour vision can be said to be an illusion created by the interactions of billions of neurons in our brain[[1]](#footnote-1), we do not all perceive colours the same way. There is an information gap for designers about ICV, many do not know the problems some users will suffer because of their colour choices. Impaired colour vision varies in severity and because of this, it affects people differently; sometimes it is unnoticeable. Figure 1 was created using an online tool and demonstrates different types of ICV, however just like most other tools available, these images show the maximum severity only.



Figure 1 - Left-to-right, top-to-bottom: Normal colour vision, Protanopia, Deuteranopia, Tritanopia.

Designers do not normally consider ICV in their design practice; they are a small demographic of users. Without doing so, designers run the risk of creating a colour palette which will be confusing to some users. Small measures such as colour checking and ICV simulations can vastly improve the users’ experience.

## Available Products

When designing for users, it can be useful to view content through the eyes of ICV users. This can reveal poor colour choices which can lead to confusion and even missed information. There are a plethora of applications available on many different platforms which can detect and manipulate pixels to simulate ICV. One flaw most software applications present when simulating ICV is the exclusion of environmental factors such as room brightness.

### Spectrum for Chrome

It is possible to obtain a browser add-on for Google Chrome called “Spectrum” which simulates ICV for the current web page[[2]](#footnote-2). This extension enables the user to select a type of ICV and see what the webpage looks like; it is especially helpful when looking at data visualisations where colours could be misinterpreted. Spectrum does not allow the user to adjust the severity of the condition specified and so it cannot simulate weak/mild ICV. This add-on is also limited to web pages only.

### Vischeck

Vischeck[[3]](#footnote-3) provides different toolsets for simulating ICV, some are available to use online whilst others require downloading and installing. The web version of Vischeck lets users upload images and see them through the eyes of an ICV user. There is also a Photoshop plugin which enables ICV simulations within the application. Vischeck does not provide a real-time ICV simulation tool however.

### Colourblind Vision

Colourblind Vision[[4]](#footnote-4) is an application developed for Android smartphones. It was created by Bradley C. Grimm and provides real-time ICV simulations using the smartphones back camera. The application also enables users to simulate ICV on images found on the camera reel. There are similar applications available for iPhone users, showing there is an interest in simulating ICV on handheld devices.

## Social Context

It can be difficult to understand how ICV affects people; emulating the effects on images is useful, however this confines the condition within just the images. ICVGoggles aims to provide a simulation tool which works in real time giving users a peek at ICV 'in the wild'. Once complete, ICVGoggles can be applied to different social contexts in order to raise awareness and to educate users. For example, the tool could be used on trainee teachers to educate them in how ICV affects people. This sort of training could eventually help teachers to recognise comments made by children who have ICV; the earlier ICV is diagnosed, the better. Another application could be the use of the tool on parents who have children with ICV to give them a rich learning experience.

## Scientific Context

By providing an interactive tool which simulates ICV in real time, this project can be used to inform and educate. Teaching ICV can be a difficult task depending on who is learning; it may be more effective to provide a hands-on tool which can show how it affects people. The severity value can be adjusted by the user; many simulation techniques show the most severe forms of ICV only instead of incrementing or decrementing the severity. Because of this, ICVGoggles is a powerful tool to raise awareness for ICV.

## Previous Work

ICVGoggles is a project which has inherited work from previous research. The colour swapping in ICVGoggles is based off of CVDSimulation, a Processing application created by Dr. David Flatla. CVDSimulation provides the ground work for ICVGoggles and is itself based on the paper “A Physiologically-based Model for Simulation of Color Vision Deficiency” [5] and a tutorial[[5]](#footnote-5). The tutorial contains a large 4 dimensional array which contains float values. By applying the correct formula to red, green and blue pixels with these values, ICV can be simulated.

# SPECIFICATION

A specification of the problem and an explanation of how the student arrived at this specification. An initial work schedule including an overall project plan with time-scales, deliverables and resources. If using agile development, a prioritised product backlog.

## The Problem

The process of simulating ICV has been around for a while and there is no shortage of applications and tools which can do so. What many of these tools cannot do is simulate the surroundings of the user in real-time, this is the problem ICVGoggles aims to solve.

Currently most ICV simulation tools provide a solution to specific problems and have downfalls when applied in other areas. For example, Vischeck provides developers with a way to see their work through the eyes of an ICV user, however it does not take external lighting or screen brightness into consideration. Other solutions such as mobile applications[[6]](#footnote-6) are restricted by the hardware they are on; users must always use one hand to move the mobile device around.

## Motivation

It is becoming more commonplace for applications to have colour blind settings, however there are very few which display ICV simulations in real time. ICVGoggles aims to take simulating ICV to the real world and will use an Oculus Rift headset to do so. This will result in a simulation tool which will give the user both hands to operate the world around them. The tool could be used for designing both digital and physical products, or it could be used to educate and teach. ICVGoggles will be open to many different routes when it is complete.

## Project Plan

This projects lifecycle spanned over seven months and so it was necessary for the student to correctly plan out their course of action. The weekly meetings set up with the supervisor ensured progress could be monitored at regular intervals.

### Work Schedule

The project began on the 5th October 2015 with a meeting with Dr. David Flatla. It was decided that the student should have weekly meetings with their supervisor to ensure all progress was being monitored. The weekly meetings also meant the student could assign smaller deadlines on a weekly basis. Notes were taken at every session and have been compiled intoAppendix A. These minute meetings contain thoughts gathered from the meeting with the supervisor and were used to plan work and to be reflected upon for the following week. A GitHub repository was set up at the start to store all work done whilst the commits and punch card feature proved useful to track progress. For the mid-term progress hand in, the student produced a Gantt chart which would serve as a rough schedule.

### Deliverables

The student created deliverables in the Gantt chart seen in Appendix B. These were spread out over the course of the project lifecycle and extra time was given for all tasks to give the student some leeway if any complications occurred. Initially the deliverables were centred on the ethics submission which required multiple consent documents to be created and checked by the Ethics Committee. These documents included consent forms for the prestudy interviews and second stage evaluations, both for ICV and Non-ICV participants.

After these were completed, the student then moved onto producing the code for the project. As seen in the Gantt chart, this production was done over the course of several months whilst both user studies were taking place. The plan for the programming was created before any coding had begun, therefore the student made sure the tasks were split into manageable, realistic deadlines. The progression of coding is also seen on the chart and each deliverable is a logical increment towards the final goal.

### Resources

The student took advantage of resources available. Labs within the Queen Mother Building at Dundee University were used for both the prestudy interviews and the second stage evaluations. Within the labs is a PC which contains the Oculus Runtime and this enables the use of ICVGoggles.

Whilst studying in their final semester, the Student chose the ECVD Research Frontiers module to further aid in their understanding of colour vision deficiencies. This module covered many research papers to do with ICV and discussions were held on a weekly basis with the class to cover them in detail.

# DESIGN

## Decisions and Trade-offs

The student required a platform to develop ICVGoggles on. The Oculus SDK is available for Visual Studio and for Unity; both platforms could be used for the project. It was decided by both the student and their supervisor to write the software in C++ on Visual Studio as opposed to within Unity. This is because Unity was designed to create games; even though a simulation tool is entirely possible, it would have still been created in a platform centred on a different technical area. This means there would have been a large amount of unused overhead which can cause system slowdown if the application was not optimised correctly. On the other hand, projects created with C++ in Visual Studio contain just the essentials in order to get up and running. Since the system will be doing image manipulation at around twenty times per second, it is necessary to ensure there is as little overhead as possible to maintain a respectable frame rate.

Due to the timescale of the project, the student and supervisor felt it would be best to postpone the production of personalised simulations until the adjustable ones were complete. By doing so, it gave the student enough time to finish the adjustable simulations for user testing. The personalised simulations have been added to future work.

## Design Method

## Software & Hardware

### C++ in Visual Studio

The coding language of choice for ICVGoggles is C++ since it is what the Oculus SDK is coded in. C++ is a middle level language and is extremely versatile; it allows for low-level memory manipulation [1] which is ideal for changing RGB values. The student had not touched upon this programming language for a few years and so it was important to solidify the fundamentals early. Thankfully OpenFrameworks contains a project generator which aided in starting up greatly. Visual Studio was chosen as it is the development platform the student is most confident with.

### OpenFrameworks

Coding ICVGoggles with just C++ and the Oculus SDK is a difficult task without the help of a framework. OpenFrameworks is an intuitive open-source C++ framework designed to assist creative processes [2]. The toolkits design philosophy is DIWO (Do it with Others), where creating content is done as a community. This proved helpful during the design process as members of the online forum actively aided in progressing with some small issues for ICVGoggles. OpenFrameworks is very similar to the programming language Processing [3]; a language the student had touched upon in previous years of study. Because of this, it was easier for the student to learn the framework documentation.

OpenFrameworks comes with a project generation tool. After searching and finding which IDE's are installed on the machine, the tool allows the user to select additional add-ons to include in the project. For ICVGoggles, the only add-on used was ofxOculusDK2.

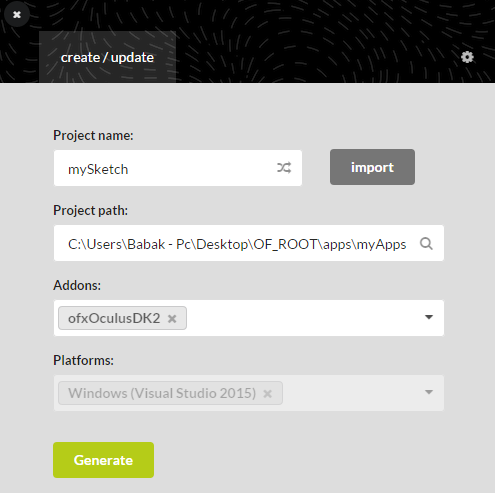


Figure 2 - OpenFrameworks Project Generation

### Oculus SDK v0.6

In order to utilise the Oculus Rift, the software development kit (SDK) must be installed. This large package contains all code required to use the headset. The Oculus Rift is mostly used for Virtual Reality (VR) and so many code examples and tutorials cater for this. ICVGoggles only requires feeding a camera stream to the headset and so the student decided to use a community add-on which facilitated this functionality. It was important for the student to use the SDK version 0.6 as the community addon required this specific version.

### ofxOculusDK2 Add-on

Since OpenFrameworks is open-source, there is a vibrant and active community which revolves around it. It is commonplace for developers to create helpful packages and share them amongst the community. ofxOculusDK2 (OpenFrameworks Oculus Developer Kit 2) was created by developer Andreas Muller and refactored by James George [4]. This add-on accesses certain parts of the Oculus SDK to enable basic rift rendering functionality, giving the student a lightweight gateway to simulate ICV. The fish eye distortion is applied automatically to anything drawn on the left or right eye, meaning the student had one less task to worry about.

### Oculus Rift

The Oculus Rift was chosen as the headset for ICVGoggles to use; specifically the Dev Kit 2 which was released on March 25th 2014. The headset has an OLED display and uses a positional tracking system which enables the movement within a 3D space. For ICVGoggles, both eyes will be drawing and manipulating a camera feed to simulate the real world.

### OVRVision Cameras

In order to simulate the world in real-time, cameras are required for the Oculus Rift. OVRVision was created by a small tech start-up in Japan and provides two cameras to be mounted on the Oculus Rift. OpenFrameworks was able to detect both mounted cameras and so it was then possible to continue coding.



Figure 3 - The Oculus DK2 with OVRVision Cameras attached.

### Oculus Runtime v0.7

The Oculus Runtime is required to detect the Oculus HMD. This piece of software is essential to run ICVGoggles as it allows the Oculus Rift HMD to become a secondary monitor. It is important for users to have the Oculus Runtime <0.7 as the secondary monitor feature was removed from future versions.

## Design Process

Designing for usability?

UML Diagrams

workflow

# IMPLEMENTATION & TESTING

A description of production, testing and debugging. A demonstration (or even a proof) that the specification has been satisfied.

## Production

(use of davids app and help from that code)

## Testing

- testing during production, methods used

## Debugging

process of debugging during development

- proof it works (images before and after ICV applied)

## Release Package

whats in final folder

instructions manual

# DESCRIPTION OF FINAL PRODUCT

A clear description of what the final product looks like and what it does. This is vital but often neglected.

Full description of final product, well worded and should NOT be neglected

Reference instruction manual

# EVALUATION

Usability should be evaluated with a description of the user-centred design methods employed to produce a usable product, including rapid prototyping, usability methods, results and re-designs as appropriate. Other relevant criteria such as accuracy and computational efficiency should also be employed for evaluation as appropriate.

## Prestudy Interviews

## -prestudy interviews, main testing (plates and exploration), questionairres

## Second Stage Evaluation

## Analysis

Of results. Methods used.

## Evaluation of Results

## Usability

## Accuracy (important!)

# DISCUSSION

Area where I discuss reasons for the results found and how these results may benefit ICVGoggles.

# APPRAISAL

A critical appraisal of the project indicating the rationale for design/implementation decisions, lessons learnt during the course of the project and an evaluation (with hindsight) of the final product and the process of its production (including a review of the plan and any deviations from it).

## -Rationale for design

## -Rationale for implementation decisions

## -Lessons learnt

## -Evaluation including hindsight

**A description of any research/hypothesis**

# SUMMARY & CONCLUSIONS

## Summary

## Conclusions

# Qualitative evaluation, qualitative (IF DONE) evaluation, personal feelings on project and how it went

# FUTURE WORK

-Mobile ICVGoggles (garreth said a battery back pack, possible?)

-two cameras

-personalised simulations

A copy of the mid-project progress report should be included.

# REFERENCES

1. Graham M. Seed 1996. C++ and its low level memory management. *An Introduction to Object-Oriented Programming in C++.*
2. OpenFrameworks C++ Toolkit. www.openframeworks.cc/about/
3. Processing, the flexible software sketchbook. https://www.processing.org/
4. ofxOculusDK2 OpenFrameworks Add-on. <https://github.com/obviousjim/ofxOculusDK2>
5. Gustavo M. Machado, Manuel M. Oliveira & Leandro A. F. Fernandes. Previous research used for CVDSimulation. *A Physiologically-based Model for Simulation of Color Vision Deficiency*

# APPENDICES

Appendix A - Minute Meetings

Appendix B - Gantt Chart

Appendix C – ICVGoggles Flow Chart

Appendix D – Prestudy Transcripts

Appendix E – Second Stage Evaluation Transcripts

Appendix F - Instructions Manual

1. P.Gouras, 'Colour Vision', in webvision.med.utah.edu, last update 1 July, 2009 [↑](#footnote-ref-1)
2. Spectrum, offered by Yehor Lvivski for Google Chrome [↑](#footnote-ref-2)
3. Vischeck, Simulation tools for web and photoshop. http://www.vischeck.com/vischeck/ [↑](#footnote-ref-3)
4. Colourblind Vision by Bradley C. Grimm. https://play.google.com/store/apps/details?id=com.givewaygames.colorblind\_ads&hl=en [↑](#footnote-ref-4)
5. Tutorial where the CVDSimulation colour value matrices can be found. http://www.inf.ufrgs.br/~oliveira/pubs\_files/CVD\_Simulation/CVD\_Simulation.html [↑](#footnote-ref-5)
6. 20 iPhone Apps for the Colour Blind - http://www.color-blindness.com/2010/12/13/20-iphone-apps-for-the-color-blind/ [↑](#footnote-ref-6)