Visual impairment simulator for auditing and design

G Stewart[1], R J McCrindle[2]

[1]School of Systems Engineering, [2]Department of Bioengineering,
University of Reading, Whiteknights, Reading, UK

georgewillstewart@gmail.com, r.j.mccrindle@reading.ac.uk

www.reading.ac.uk

ABSTRACT

Individuals within the visually impaired community often have difficulty navigating environments due to the different ways in which they view the world, with even apparently simplistic locations frequently being challenging to traverse. It is therefore important when designing architecture or environments, to take into account the perspectives of people with visual impairments in order to ensure that design outcomes are inclusive and accessible for all. Although there is documentation regarding guidance and procedures for design of inclusive spaces; architects, designers, and accessibility auditors often find it hard to empathize with visually impaired people. This project aims to make the process of inclusive design easier through the development of a mobile app, VISAD (Visual Impairment Simulator for Auditing and Design), which enables users to capture images or import CAD designs and apply image distortion techniques in order to replicate different visual impairments.

1. INTRODUCTION

There are 2 million people living with some form of visual impairment in the UK alone (RNIB/Access Economics, 2009) including 218,000 people registered with a severe visual impairment or blindness (RNIB/Access Economics, 2009). The number of people living with a sight threatening eye condition has been predicted to rise over the next decade (Minassian and Reidy, 2009), and it is therefore becoming increasingly important to emphasize inclusive design when producing new products and services, ensuring that they are accessible to, and useable by, as many people as reasonably possible (Clarkson and Coleman, 2015).

Architectural design is a key area of interest when it comes to design accessibility. This is especially true for public areas, which often have to abide by various codes of practice, and legislation. There is ample documentation available on the subject of disability access. However, although these documents provide useful information about specific individual architectural aspects, for example, the number of steps in a staircase before a landing, they do not provide the reader with a sense of what it is like to be afflicted with a visual impairment.

Simulating visual impairments is arguably the most effective way to educate uninformed designers about how a person with a visual impairment perceives their environment. Allowing architects to 'see' through the eyes of a visually impaired person makes it much easier for them to empathize with the difficulties individuals might face, and provides them with a greater understanding of the aspects of architecture which can be problematic for people who have visual impairments. Prior research, and subsequent software solutions, related to simulating visual impairments have been predominantly developed on non-mobile PC platforms. This project aims to improve upon, and enhance previous attempts at simulating visual impairments by developing a visual impairment simulator, VISAD, (Visual Impairment Simulator for Auditing and Design) that runs on mobile devices such as smartphones and tablets enabling both in the field simulation as well as the ability to import images at the design stage.

2. INFLUENCES FROM PREVIOUS STUDIES

Early research in this field includes that of Fine and Rubin (1999) who printed black circles on clear acetate film using an inkjet printer to replicate a scotoma. Test subjects read text through the acetate film, and changes in reading performance were documented. Recent research projects have become increasingly sophisticated with most involving the use of computer graphics and image processing techniques to distort images in various ways to replicate visual impairments.

Research by Hogervorst and van Damme (2006), involved the production of a software system that allowed the user to import an image and apply simulations of cataract, macular degeneration, glaucoma, retinopathy, myopia, and hyperopia to a 2D image in order to educate and give users an insight into the problems that people with visual impairment conditions faced. Their research provided an in depth perspective on the methods that could be applied to produce visual impairment simulations including a demonstration of how light glare can effect certain impairments.

Banks and McCrindle (2008) also investigated the use of image processing techniques to replicate the characteristics of common visual impairments in order to provide planners, designers, and architects with visual representations of how people with visual impairments viewed their environment, leading to increased accessibility in the built environment. This research also provided insight into the types of features that could be added to a proposed solution including the ability to import images from CAD files into the system.

In addition to 2D image processing solutions, a research project carried out by Lewis, Shires, and Brown (2012) involved the application of real time, simulated visual impairments to a virtual 3D environment. This project utilized the Microsoft XNA video game development framework, and allowed the user to walk around a 3D office environment while being afflicted with different visual impairments. This project showed how virtual and real time augmented reality can be used to enhance user experience.

The inclusive design toolkit, developed at Cambridge University, by the Engineering Design Centre (EDC, 2016), features a set of tools that aid creation of products and services that are inclusive or accessible. One such tool is the ‘Vision Simulator’ which provides examples of visual impairments applied to a selection of images. The simulator features a variety of impairments that can be applied with different levels of severity. It also includes a section which demonstrates impairments in relation to different objects, including a railway ticket machine and a household toaster. These simulations allow the user to alter certain design aspects of the objects (mainly the colours used) to show how small changes in some of their characteristics can alter their accessibility to someone with a visual impairment. The inclusive design toolkit demonstrates good practices regarding how the user interacts with the system, providing an easy-to-use and intuitive GUI (Graphical User Interface) that incorporates a number of visual impairment presets alongside slider controls for the severity of the impairment. The system also provides users with a short description of the visual impairment being simulated and the demographic of people affected by it.

‘Project Rainbow’ at the University of Reading (Bright and Cook, 1999), investigated the effects that colour and luminance have on the built environment, when viewed by a visually impaired person. The outcome of this project was a set of advisory papers and design guides which provide guidelines and rules to follow when designing the interiors of buildings. The overall aim was to create designs that were accessible to visually impaired people, but also acceptable for ‘designers and fully sighted users of the building’.

ViaOpta (2016) is one of the first apps to be developed for mobile platforms, however whilst providing a useful tool for simulation of visual impairments in real-time, it falls short with regards to its flexibility to import and export images and simulations.

3. DEVELOPMENT ETHOS AND IMPLEMENTATION APPROACH

3.1 Development Ethos

One aspect that many earlier research projects have in common, is that they were designed to run on non-mobile PC platforms. Whilst this approach has benefits such as increased processing power and greater standardization in terms of OS (operating system), mobile device platforms offer greater flexibility of use for architects, designers, and accessibility auditors during the inclusive design, and auditing phases of their projects. Additionally, whilst mobile devices have become incredibly popular over the past decade, for example, by 2013 there were 900 million devices running Google’s Android OS (Business Insider, 2013), PC sales have declined substantially (Gartner 2016), and hence it is timely to develop for the mobile market.

The VISAD application has therefore been developed for use on smartphones or tablets, allowing the user to capture images with the device, and then create instantaneous visual impairment simulations of the images providing users with a better understanding of problems that might occur for people with visual impairments. The app handles all image capturing aspects of the simulation within the app, so that it is convenient to use. The app can produce a series of preset visual impairment simulations as well as allowing for customization of the various image processing effects that are used to take into account severity and layering of conditions. It also provides the user with information regarding specific impairments, including the demographic that is most commonly afflicted. Other key features implemented include the ability for the app to import image files from a range of sources as well as through the integrated camera and the facility to save and export generated image files for external use via their associated Google Drive cloud storage.

3.2 Implementation

VISAD has been developed for the Android OS platform, implemented in the Java programming language, and created using the Android SDK version of the Eclipse IDE (Google Android, 2014a) and Android, version 5.1, API 21 – Lollipop, (Google Android, 2014b). The app operates with three main classes: the image capturing class; the GUI and user input class; and the graphics rendering class. The overall processing of the app occurs in three stages. First the user captures an image with the devices camera, or imports an existing image. The user then applies a combination of the different image effects, or selects an impairment preset. Finally, the user analyses the resultant image using the comparison tool, or the edge detection tool, and then exports the image by saving it to the devices memory.

3.3 Image Capture

The image capturing aspect of the app is implemented using the Android 'Camera' class. This provides access to the device's camera in order to capture images or video, and to retrieve the relevant hardware information. It also provides handlers for setting camera specific features, such as autofocus, capture resolution, and white balance. Image capturing is implemented within the 'CameraPreview' class of the Java project, which provides a custom 'SurfaceView' component that is instantiated within the main class. This ultimately enables the devices camera output to be displayed on the screen.

3.4 Image Processing

Image processing is used within the app to produce different effects that are then combined to one image to create a visual impairment simulation. This is implemented in the app through a custom 'SurfaceView' component which allows for custom drawing to the screen and to bitmaps. This class defines its own thread which it uses to update the custom drawing every 1/30 of a second. It receives parameters from the GUI thread and produces drawings or image manipulation based on the input. The majority of the work done within this class is produced using the Android 'Canvas' class, which allows for the creation of simple 2D geometry and custom brushes. All of the image processing effects have parameters that can be altered by the user. Each effect has its own menu which contains all of the GUI elements that control the parameters. These menus also include checkboxes which are used to toggle the relevant effect on or off. These parameters are passed to the rendering function and effect what is drawn in the frame. The different effects implemented include:

3.4.1 Brightness. Altering the brightness of the image involves altering the RGB values of the individual pixels in the image. This effect uses the Android Bitmap class to get the pixel value at an x/y coordinate in the image, the pixel's RGB values are then multiplied by a double value, and then set back in the image at the same x/y position. The double value is provided as a parameter and is a value in the range $0<x<2$, where any value less than 1 darkens the image, and any value greater than 1 lightens it. This effect, for example, is used in the macular degeneration simulation to increase the brightness of the image replicating the 'washing out' of colours that is commonly associated with this impairment.

3.4.2 Colour Filter. The colour filter effect overlays a semi-transparent colour on top of the image. This effect has been implemented using the Android Canvas to draw a rectangle with the same x and y position, and the same width and height of the image. The RGBA values of this effect can be altered to change the colour and opacity of the filter. The colour filter effect is rendered to a separate transparent bitmap, which is then drawn after the image when the rendering loop draws a frame. By implementing this effect in this manner it is easier to implement the option to enable/disable the effect. This effect is implemented in the Cataract preset, where a brownish filter is required to simulate the effect of protein clumps in the lens, which cause cloudy vision.

3.4.3 Warping. The warping effect causes the contents of the objects portrayed in the image be distorted. Areas of the image are stretched, pinched, and swirled which results in a warped image. This effect uses the JH Labs libraries to perform the warping, and the extent and size of the different kinds of warp can be altered to produce different outputs. There are three types of warping that the system can perform: pinching, twirling, and melting. The pinch warp causes the image to be warped towards the centre of the image; the swirl warp twists the image around its centre; and the melt warp applies random distortions to the image based on a turbulence value. These warping effects are created using forward pixels mapping, where the destination x,y coordinates are mapped using the source v,u coordinates. The warping effects are used during the simulation of Diabetic Retinopathy and Macular Degeneration to replicate the distortions that can occur as a result of these impairments.

3.4.4 Blur. The blur effect applies a Gaussian blur to the image. This effect uses the Android Renderscript API to process the image and produce the resultant blurred image. The Renderscript function for blurring allows parameters to be passed to it which control the extent of the blur. This effect can also produce a blur which fades in from the edges of the screen. A greyscale bitmask is produced that uses a radial gradient that fades from white to black from the centre of the image. This bitmask is then used to alpha composite the original image over the blurred image, leaving the centre more ‘in focus’ and the edges blurred. This effect can also be inverted, and positioned anywhere on the screen. This means that when inverted, the centre point of the effect is blurred, and it fades out towards the edge. The centre point can be positioned by the user by tapping on the screen, and the centre is set to the point of the tap. These specific aspects of the effect are used for the Myopia (nearsightedness) and Hyperopia (farsightedness) simulations. When selecting these simulations, the user is asked to tap on the screen the most distant point. This effect is used to simulate different aspects of blurred vision that are commonly related to Myopia, Hyperopia and Glaucoma. It can also be combined with the Double Vision effect to produce more severely blurred vision.

3.4.5 Vignette. The vignette effect adds a dark edge to the image which fades out toward the centre. This effect utilizes the Android Radial Gradient shader which allows the creation of a circular gradient which fades between multiple colours that are provided as parameters. In the case of this effect, the colours fade from a user defined colour at the edge (defaulted to black), to a completely transparent colour at the centre. The aperture of the effect is controlled by the radius parameter, which can be adjusted to increase or decrease the size of the vignette. This effect produces an image which has a border with a fading transparent centre. This effect is applied last in terms of layering the effect images onto the canvas and is used in the system to simulate the loss of peripheral vision, which is present in most of the impairments.

3.4.6 Double Vision. The double vision effect overlays a semi-transparent copy of the image on top of itself at a slight offset. By using the bitmap variable type that is available in the standard Android API, a copy of the image is made. This copy is then drawn over the original image using the Android Canvas. When drawing the copy, a parameter is passed to the function that draws the bitmap which alters the opacity at which the image is drawn at, and also the x and y position is offset from the original. The image is set to 50% opacity and is offset from the top left corner by a few pixels by default. The results of this effect makes it harder for the viewer to focus on any of the fine detail in the image. This effect is used in the Cataract, Diabetic Retinopathy, and Macular Degeneration presets to replicate some of the blurriness and fuzziness that is associated with these impairments.

3.4.7 Spots. The spots effect draws randomly generated shapes with “fuzzy” edges onto different parts of the image. This effect is produced by initially creating a grid to which a cell is randomly selected, and then adjacent cells are procedurally, randomly selected. This results in a single randomly created shape. The shape is then drawn to fit the canvas using a radial gradient to provide a faded edge. The characteristics of this effect, including: length of shape; size of cells; number of individual shapes; and colour and opacity can all be changed by altering the input parameters. This effect is used mainly in the Diabetic Retinopathy and Macular Degeneration simulations to represent areas where vision is completely lost.

3.4.8 Colour Blind. The colourblind effect changes the RGB pixel values of the image in order to simulate colourblindness. This effect can simulate the following types of colourblindness: Protanopia (red blind); Protanomaly (red weak); Deuteranopia (green blind); Deuteranomaly (green weak); Tritanopia (blue blind); Tritanomally (blue weak); Achromatopsia (monochromacy); and Achromatomaly (blue cone). The implementation of this effect is based on research by Wickline (2008), which provides the predefined values by which the RGB channels are multiplied by, and then combined to form a pixel. For example, the manipulation values for the Red channel, for the Deuteranopia form of colour blindness are: *(0.625, 0.375, 0), w*hich means that the red channel RGB components are calculated using: $R_c = (R_d * 0.625) + (G_d * 0375) + (B_d * 0)$.

3.4.9 Light Source. The Light Source effect is used to simulate the problems that excessive illumination can have on a visually impaired person. This effect creates a pseudo light source and places it on the image. This results in the colours and definition of the image to become ‘washed out’ due to the increased brightness. The light source effect is produced by creating a radial gradient using the Android RadialGradient class, decreasing the alpha values of the pixels from the centre, and then overlaying a brightened version of the image on top of itself. Essentially it produces a copy of the original image, then increases its brightness and cuts a circular section out of it. The effect then overlays this cut out section on top of the original image in the same location, and fades the edges of the circle to give it a more natural look. This effect has two parameters that can change its properties, the intensity value changes how bright the light source appears to be, and the size which alters the diameter of the effect on the image. The user can add multiple light sources to the same image accessing this feature via the effects menu of the GUI.

4. SIMULATIONS AND TOOLS

4.1 Simulations

The effects described in Section 3 can be applied to the inputted image individually, or the user can choose from a selection of eight preset visual impairments from the presets menu. Each preset simulates a specific visual impairment as defined by the RNIB (Royal National Institute for Blind People, 2016) or the NEI (National Eye Institute, 2016) by automatically applying a combination of the individual effects. In addition to this, the user can adjust the severity of the effects using a 'severity slider', which ranges from mild to severe relating to the selected impairment as shown in Figure 1.

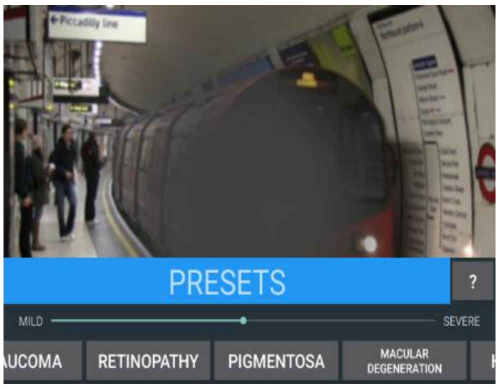


Figure 1. *Presets menu.*

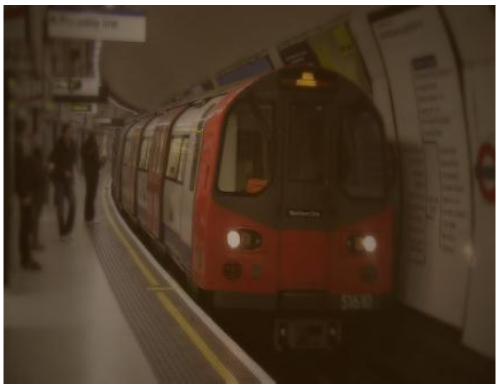


Figure 2. *Cataract preset.*

4.1.1 Cataract. Cataracts, defined as the 'clouding' of the lens section of the eye, causing a decrease in the overall quality of vision, are usually identified by the eye turning from its natural colour, to a yellow/brown colour. This yellow/brown colour is the colour of the protein clumps, and the afflicted person's vision is usually 'stained' with this colour as if looking through a coloured filter. The preset for this visual impairment applies a number of different effects to simulate both the clouding of the eye's lens, and also to replicate the loss of peripheral vision. The main effect that is applied is the colour filter effect. A yellow/brown colour filter is placed over the entire image to simulate the lens clouding. To replicate the loss of peripheral vision, a black vignette is placed on the image, and a radial blur is applied to blur the edges of the image in a circular fashion. The brightness of the image is also decreased slightly to account for the lower amount of light able to pass through the lens (Figure 2).

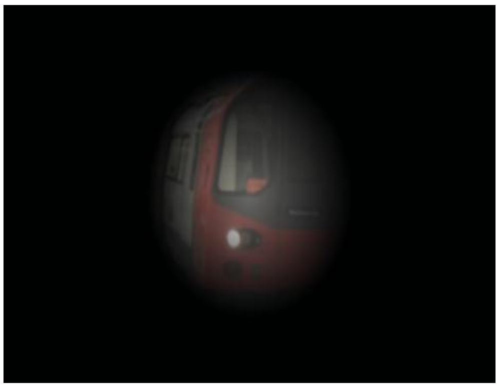


Figure 3. *Glaucoma preset* .

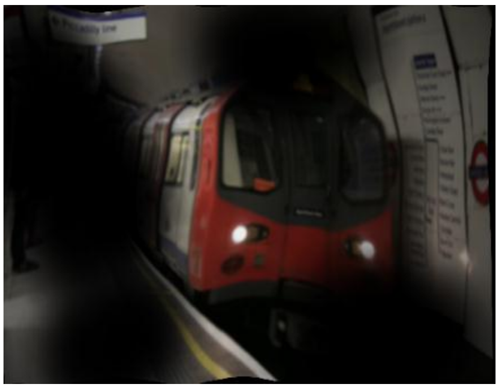


Figure 4. *Retinopathy preset.*

4.1.2 Glaucoma. Glaucoma is a visual impairment that mainly effects the peripheral vision and if left untreated can lead to blindness. The most predominant effect that this preset applies to the image is the vignette effect. This effect is used to create a large black ring around the edges of the image, which is used to replicate a loss of peripheral vision. Also involved in this preset is a slight increase in brightness. This is used in conjunction with a moderate application of blur to simulate the mistiness and 'washed out' colour that are associated with glaucoma symptoms (Figure 3).

4.1.3 Retinopathy. Retinopathy affects people who suffer from diabetes. It is caused by microvascular changes in the retina section of the eye, and results in blurred and patchy vision, which ultimately leads to blindness. People who suffer from Retinopathy have areas of their vision which don't function correctly, this causes spots of darkness that are surrounded by blurriness. The preset which is used to simulate Retinopathy makes use of the 'Spots' effect to recreate the patchy areas of vision loss that are associated with this impairment. This is combined with a slight blur and warp effect to simulate the distortion of the vision that occurs around the areas of vison loss. When the severity of this preset is increased to +75%, a vignette is also applied to demonstrate the loss of peripheral vision that occurs in the later stages of this impairments progression (Figure 4).

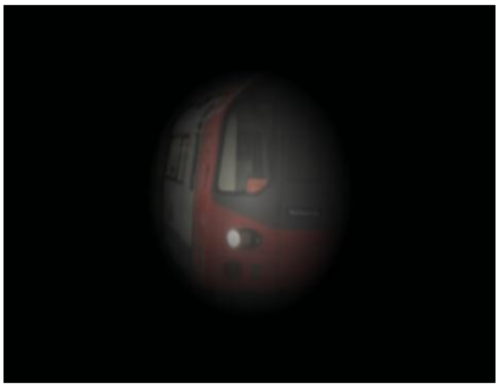


Figure 5. *Pigmentosa preset.*

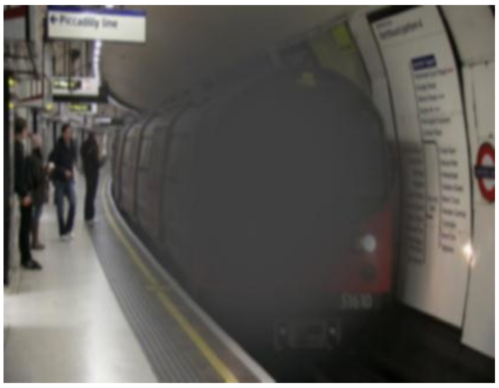


Figure 6. *Macular Degeneration.*

4.1.4 Pigmentosa. Pigmentosa is an inherited condition which causes severe tunnel vision. It is caused by an inflammation of the retina, and the symptoms can be present from birth. This impairment also causes a decrease in the ability to see in dark conditions, and it is possible for the central vision to also be affected. The preset for this impairment applies similar effects to the Glaucoma preset. The most apparent effect that is used for Retinopathy is the vignette effect, which is applied to recreate the tunnel vision symptom that is common with this impairment. A dark grey colour filter is also applied to the image to simulate the decreased vision in low light conditions. The final effect that this preset applies a moderate blur, which is used to replicate the general loss in the quality of vision, which is related to the inflamed retina (Figure 5).

4.1.5 Macular Degeneration. Macular Degeneration causes issues with the central vision of the eye such that the eye is unable to focus on anything and there is no sharpness or detail in the vision. This impairment is particularly common is older people, and is caused by a build-up of cell debris in the macular section of the retina, which in turn causes scarring and damage. The Macular Degeneration preset applies a special version of the 'Spots' effect to create an irregular, fuzzy, grey coloured circle at the centre of the input image. This is applied to simulate loss of central vision, and the diameter of the circle is increased relative to the severity of the preset. Also applied with this preset, is a blur effect which also increase with the severity, although to a lesser degree of intensity (Figure 6).

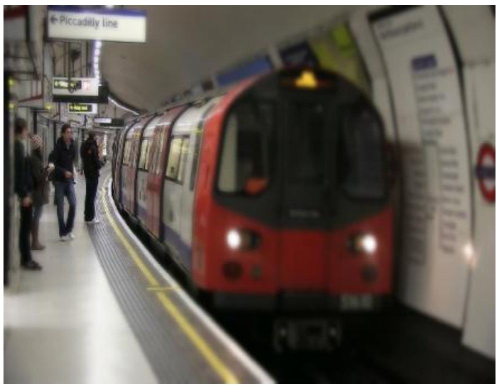


Figure 7. *Hyperopia preset.*

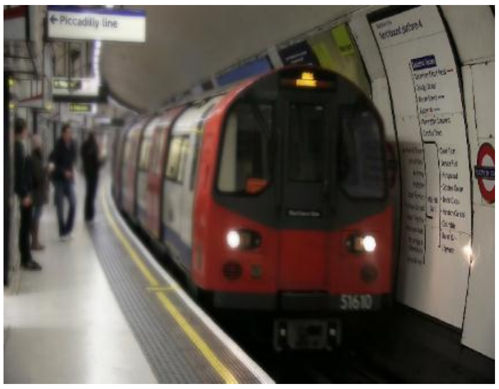


Figure 8. *Myopia preset*.

4.1.6 Hyperopia and Myopia. Hyperopia and Myopia are both classified as refractive errors of the eye. Hyperopia is commonly known as farsightedness, and Myopia is known as nearsightedness. Refractive errors are caused by a combination of factors relating to the size and shape of the various components that make up the eye. The size of the eyeball, deterioration of the lens shape, and alterations of the cornea, can all effect the way that the light that enters they eye can focus on the retina. If the light is incorrectly focused, objects closer or further from the eye appear to be blurry. The preset for Hyperopia (Figure 7) uses the radial blur effect to create an image where object closer to the viewer are out of focus, but ones further away are in focus. This is done by prompting the user to select the point in the image which is of the furthest distance. This is treated as the center point for the radial blur, and the radius of the focused area is adjusted based on the severity of the preset. For the Myopia preset (Figure 8), the same process is used, however the radial blur effect is inverted. This means that the most distant point selected in the image is blurred, and out of focus, and the areas that are closer to the viewer are in focus.

4.1.7 Colour Blindness. Colour blindness, also known as colour vision deficiency, is used to describe the different vision impairments that effect the way in which a person perceives colour. There are 8 different types of colourblindness presets that can be simulated by the system, these are: Protanopia (red blind); Protanomaly (red weak); Deuteranopia (green blind); Deuteranomaly (green weak); Tritanopia (blue blind); Tritanomally (blue weak); Achromatopsia (monochromacy); and Achromatomaly (blue cone). These are applied with the colourblindness effect, which works by multiplying the RGB pixels by manipulation values.

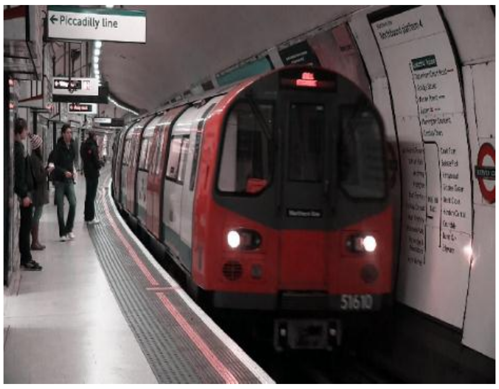


Figure 9. *Colour Blindness preset.*

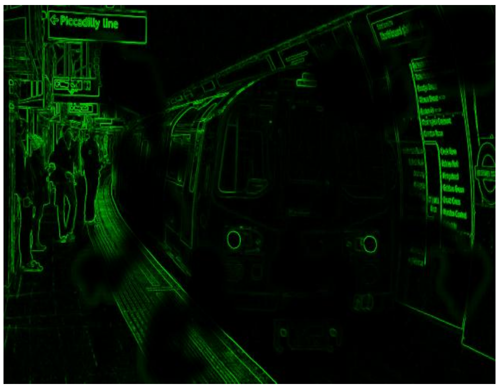


Figure 10. *Edge detection tool.*

4.2 Tools

The analysis components of the app are used to demonstrate the comparisons between the original, unedited image and the newly created image, with the simulation effects applied to it. There are two tools that can be used for analysis: the canny edge detection tool and the side by side comparison tool. These provide the user with an easy way to contrast the differences between how their image appears through the eyes of impaired and unimpaired individuals.

4.2.1 Edge Detection. The edge detection tool highlights any perceived edges in the image as green lines. This can be used to contrast the difference that the visual impairment simulations have on the algorithms ability to detect edges, compared to the original. This feature is implemented using the JHLabs edge detection function which takes in a bitmap image as a parameter, and returns a new image containing the edge detection output. This function applies a 'Canny' edge detection algorithm to the image. The Canny edge detection works in multiple stages, and is fairly complex, however the overall process involves applying a Gaussian blur to the image, then observing the intensity gradients. After this, a non-maximum suppression is applied to reduce the likelihood of false detection, and then a double threshold is applied to examine possible edges. The final step is to suppress the discovered edges that are insignificant, and not connected to the larger edges.

4.2.2 Comparison. The second analysis feature is the comparison tool feature of the app. This provides a side by side comparison of the original image, and the image that has been distorted by the system. This tool places a slider on the screen which overlays the original image on the left side of the slider and the edited version of the image on the right side. The slider can be moved back and forth by the user to display different areas of both the edited, and original images.

4.2.3 Importing & Exporting. The tools section of the app focuses on the more basic aspects of functionality that are expected with any software package including saving and loading options. The user can choose to import an image from the devices file system, instead of capturing their own. The import image tool uses the standard Android 'Image Open Intent', which provides a premade activity that the app can switch to that allows the user to browse images easily. When the user selects an image file, the intent returns the user to the app with the image URI. This URI is then used to copy the image to the image variable used by the app to apply distortions to. Once the user has finished simulating visual impairments with the image, they have the option to export the output as an image file and save it to the device's local storage. Once saved, the images can be used in other applications, or included in documents, or printed. Saving is carried out using the Android 'File' object, which works similarly to the standard Java object. The image is saved to a folder within the file system that is created especially for the app which is created when the app is first executed.

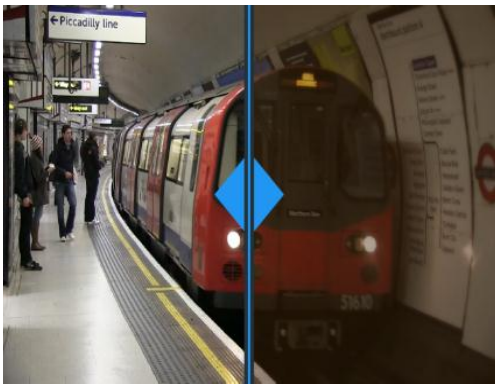
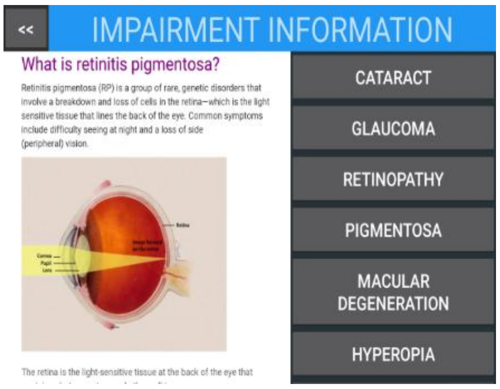


Figure 11. *Comparison tool.*



What is retinitis pigmentosa?

Retinitis pigmentosa (RP) is a group of rare, genetic disorders that involve a breakdown and loss of cells in the retina—which is the light sensitive tissue that lines the back of the eye. Common symptoms include difficulty seeing at night and a loss of side (peripheral) vision.

The retina is the light-sensitive tissue at the back of the eye that

Figure 12. *Information tool.*

4.2.4 Image Capturing. While in the "image capture" mode, the app uses the screen to relay the input from the camera, which allows it to be used as a viewfinder. This feature is common among camera applications, and allows the user to capture images more easily. The capture button is placed on the left hand side of the screen, and is used to both capture images, and clear them. When an image has been captured, and the user wants to get rid of it to capture another, they press the capture button again. This deletes the image from memory, and returns the screen to the camera output for capturing.

4.2.5 Information. The app features an information section, which is accessed via the question mark button on the presets menu. This section is dedicated to providing information about the use of the app, and contains specific information about each of the individual impairments that are supported. This information is sourced from the National Eye Institute (NEI, 2016) website, and displays details about the relevant eye disease such as what causes it, and who it affects. This feature is implemented using the Android 'WebView' which is a view that displays web pages, and basically allows for a stripped down web browser to be added as a UI element within the app. The information section has a menu of buttons related to the visual impairments that are supported by the app. These buttons are linked to the WebView, and direct it to the relevant web page on the NEI website. This feature was added to allow the user to quickly and easily browse through relevant information about visual impairments without having to leave the app.

5. VALIDATION

The effectiveness of the app, its features and functionality and the accuracy of its simulations was undertaken through comparisons with previous work, a small focus group with potential student users, and an interview with a visual impairment expert. With regards to assessing accuracy of the simulated images there is no exact way of knowing whether the actual visual impairment representation is correct, a problem compounded by the visual impairments simulated by the app also have varying degrees of severity. The most plausible way of ensuring the validity of the simulation images, was thus to compare them to other simulations created during past research in this field (EDC, 2016), as well as to the symptom definitions created by official visual impairment organisations, such as the National Eye Institute (NEI, 2016) and the Royal National Institute of Blind People (RNIB, 2016). As part of this validation a full comparison of images was undertaken of the VISAD images and those produced by the Cambridge Inclusive Design Toolkit Simulator (EDC, 2016), one example of which is shown in Figure 15. From the images, little difference is evident between the two simulations, both involve rendering a large grey mass in the centre of the image, and include a slight blur to the rest of the image. The only slight difference between the two is that the simulation produced by the VISAD app applies slightly more blurred than the Inclusive Design Toolkit version, from this and the other comparisons we were able to conclude that VISAD simulations were accurate.

During the later stages of the implementation, the app was demonstrated to a focus group of 5 potential student users for testing and feedback purposes. This session involved providing the participants with 'hands on' experience with the app allowing them to test the various features of the app. Afterwards the group was interviewed about their impressions of the app, and what they liked and disliked about it. Overall feedback was very positive with all members of the focus group finding it interesting and easy to use once after minimal training on how to use the interface. Some minor errors in the UI were identified, further visual impairment presets were requested and the loading times for some of the more complex effects, such as the edge detection, was considered to be too long.

A meeting and demonstration of the app also took place with Dr. Geoffrey Cook, an expert in the field of accessible design and lighting in regards to the built environment. Overall he was very impressed by the app, and stated that 'he knew of professionals that would be very interested in using the app'. He also expressed interest in the possibility of including further simulated effects of lighting in the app; and how different lighting levels would affect the images.

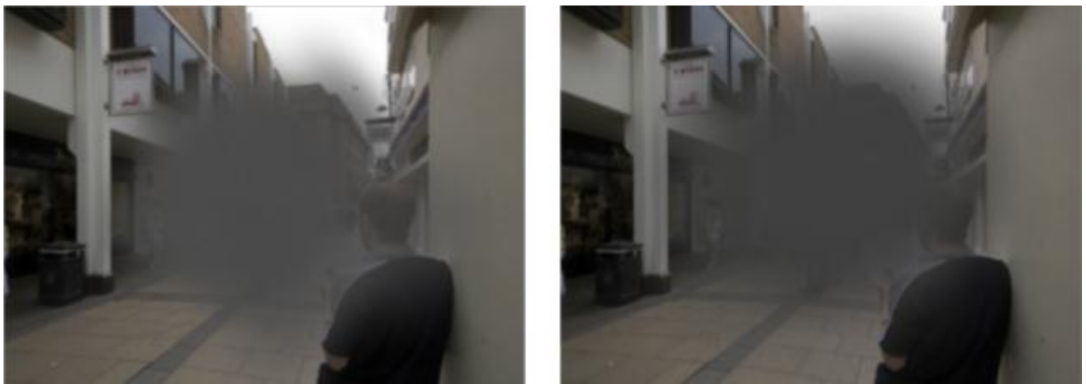


Figure 15. *Macular Degeneration comparison Cambridge (left) and VISAD (right).*

6. CONCLUSIONS

This project has created a visual impairment simulator, VISAD, (Visual Impairment Simulator for Auditing and Design) that runs on mobile devices and provides image capture, image processing and image export of visual impairment simulations. Allowing architects, designers, and accessibility auditors to visualize their projects from the perspectives of visually impaired people is essential in providing accessible and inclusive design. The overall benefits of utilizing a mobile platform for visual impairment simulation are that it makes the process more intuitive, more efficient, more educational and more impactful as immediate results can be seen. Using a mobile device allows simulations to be done in situ, with near instant results. This provides a much more fluid and convenient process compared with the previous desktop based systems, which lack integration of image capturing and processing aspects.

The comparison analysis that was carried out between the simulated images created by the app, and those created in earlier research projects and on visual impairment sites for professional organisations, show that the app can create very similar results. VISAD's simulations for the visual impairment presets were of the same quality, or in some cases better than the ones used for comparison. Discussions with experts and potential users of the app have been highly positive and there is strong interest in using VISAD as part of the University of Reading's Breaking down Barriers project which is embedding inclusive design into teaching and learning in built environment professional education and beyond and in inclusive design workshops run by the Department of Typography and Graphic Communication.

Due to the lower processing power of mobile devices compared to non-mobile devices, optimization is an area of consideration regarding further work on this project. Currently the system is limited to still images due to the processing time of the effects being too long for video, however, real time video processing is an additional feature that will be examined for the future.

An increased focus on the effects of lighting on visual impairments will also be included in future work resulting in increased complexity, sensitivity and accuracy of the 'Light Source' feature. There is also the potential to incorporate this project into a virtual/augmented reality simulation, which would allow the user to experience the impairments first hand in a more immersive environment. An IOS version of the app is also being considered.

7. REFERENCES

Banks, D, and McCrindle, RJ, (2008), Visual eye disease simulator, *Proc. 7th ICDVRAT (International Conference on Disability, Virtual Reality and Associated Technology)*, Maia, Portugal, pp. 167-174.

Business Insider, (2013), Google: There Are 900 Million Android Devices Activated, http://www.businessinsider.com/900-million-android-devices-in-2013-2013-5?IR=T

Cook, GK, and Bright, K, (1999), Project Rainbow: a research project to provide colour and contrast design guidance for internal built environments. *Occasional Papers, 57. Technical Report. Chartered Institute of Building*, Ascot, UK. pp. 20.

Clarkson, PJ and Coleman, R, (2015), History of Inclusive Design in the UK, *Applied Ergonomics*, **46**. pp. 235-247.

EDC (Engineering and Design Centre), (2016), Inclusive Design Kit, University of Cambridge, http://www.inclusivedesigntoolkit.com/betterdesign2/simsoftware/simsoftware.html

Fine, E, and Rubin, G, (1999), The effect of simulated cataract on reading with normal vision and simulated central Scotoma, *Vision Research*, **39**, pp. 4274-4285.

Gartner, (2016), PC shipments decline globally in Q1, 2016, *Gartner*, http://www.bgr.in/news/pc-shipments-decline-globally-in-q1-2016-gartner/

Google Android, (2014a), Eclipse Android Development Tools, http://developer.android.com/sdk/installing/installing-adt.html

Google Android, (2014b), Android version 5.0 Lollipop, http://www.android.com/versions/lollipop-5-0/

Hogervorst, M, and van Damme, W, (2006), Visualizing visual impairments, *Gerontechnology*, **5(4)**, pp. 208-221.

Lewis, J, Shires, L, and Brown, DJ, (2012) Development of a visual impairment simulator using the Microsoft XNA Framework, *Proc. 9th ICDVRAT (International Conference on Disability, Virtual Reality and Associated Technology)*, Laval, France, pp. 167-174.

Minassian, D and Reidy, A, (2009), Future Sight Loss UK (2): An Epidemiological and Economic Model for Sight Loss in the Decade 2010-2020: Executive Summary, RNIB Report, pp. 12, https://www.rnib.org.uk/sites/default/files/FSUK_Summary_2.pdf

NEI, National Eye Institute, (2016), Eye Condition Topics, https://nei.nih.gov/health

RNIB/Access Economics, (2009), Future sight loss UK (1): The economic impact of partial sight and blindness in the UK adult population, pp. 215, http://www.rnib.org.uk/sites/default/files/FSUK_Report.pdf.

RNIB (2016), Eye Conditions, https://www.rnib.org.uk/eye-health/eye-conditions

ViaOpta, (2016), Visual Impairment Simulator, http://viaopta-apps.com/ViaOptaSimulator.html#

Wickline, M., (2008), Colorblind Web Page Filter, http://colorfilter.wickline.org/