The Perfect Picture: Optimising Chromostereoscopic Images for Desired Depth and Colour

Thomas Sjerps
Supervisor(s): Elmar Eisemann, Petr Kellnhofer
EEMCS, Delft University of Technology, The Netherlands

June 19, 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

Chromostereoscopic images encode depth as colour, with the red part of the visible spectrum encoding nearby depths and the blue part encoding far-away depths. However, when encoding a regular image with its depth, the generated colours may not match with the original colours in the image. Following a user study, a technique has been developed which takes both the chromostereoscopic effect and the original colours into consideration.

This problem was solved by creating a program to generate chromostereoscopic images from an arbitrary RGBD input image. An optimisation metric was devised to assess if a chromostereoscopic image takes both the target depth and original image into consideration. A user study was conducted to correct this metric with respect to human perception. The new technique can take a desired amount of original colour and chromostereoscopic depth and will generate a chromostereoscopic image, taking these preferences into account.

1 Introduction

Stereoscopy is the technique of making the two eyes of the observer show two differently positioned pictures that together evoke the illusion of depth. If the medium is limited to using a single image to evoke this information, the stereoscopic effect can be achieved by using techniques like anaglyph 3D, or by making use of polarised light. Since the early years of computer imaging, researchers and consumers alike have been interested in making use of stereoscopy, and it has been the topic of a lot of research throughout the years [9].

Another technique for achieving a stereoscopic effect is called chromostereoscopy. With this technique, elements of the image are coloured with a different hue to indicate their depth in the image [13, 16]. By wearing ChromaDepth(R) glasses [2] that bend light towards and away from the nose based on its wavelength, red elements appear closer than blue elements.

Thus, the stereoscopic effect is achieved differently to anaglyph 3D and polarised 3D projection: whereas these two techniques show parallax artefacts when the user does not look through the specialised glasses, chromostereoscopy only shows colour artefacts when the observer is not wearing the ChromaDepth glasses.

In order to create a basic chromostereoscopic image, a depth image is used. These depth values are mapped onto a colour spectrum, where the red end of the spectrum represents close-by elements and the blue end represents far-away elements. A more recent technique exists where a red, white and blue gradient is used in this depth mapping [11]. Furthermore, more techniques are shown to add depth cues, like shading and depth unsharp masking [8].

These techniques can be used to create vivid chromostereoscopic images with a high range of apparent depth. However, they focus only on conveying as much depth information as possible. The original image and its texture and colours are not taken into consideration. To get chromostereoscopic images to appear more natural, parts of the colour image will need to be used.

In this paper, it is argued that an optimal method exists between conveying depth and conveying the original colours, by creating a program that takes both the depth image and original image into account. Creating this program poses three sub-problems:

1. Generating chromostereoscopic images;
2. Evaluating chromostereoscopic images;
3. Tuning this evaluation to human preferences, following a user study.

This method has the same applications as a basic chromostereoscopic technique. The added benefit this method offers, is the fact that a user can specify the preferred level of chromostereoscopy in a more perceptually intuitive sense, for any arbitrary image.

This paper will be structured as follows: firstly, some related work will be outlined. Next, the method that was used to create and evaluate chromostereoscopic images, and the implementation details of this method are shown. After describing the user study procedure, the results that followed are presented and discussed.

2 Related Work

Chromostereoscopy was developed, building off of the theory of Willem Einthoven for chromostereopsis [5, 15]. Chromostereopsis is an optical illusion whereby an image can be perceived to have depth, based on the colours of the image. According to Einthoven, cooler colours are perceived to be in the background, whereas warmer colours 'tread to the foreground'. He attributes this due to light rays bending due to the fact that the eye is asymmetric. In his theory, the circle of scattering of red light does not line up with that of 'more dispersive' blue light, due to the fact that the fovea of the eye lies at the temporal side of the eye.

Based on Einthoven's theory, the chromostereoscopic technique was developed by Richard Steenblik [13], which creates a stronger depth effect. This technique uses special optics to bend the light rays before they enter the eye. Firstly, one system of prisms is used to separate the different wavelengths of light (cooler colours are brought towards the nose). Then, another is used to compensate for the change in angle. The bending of the light results in a stereoscopic illusion, by creating a different image per eye.

The ChromaDepth(R) process and corresponding glasses are proprietary technologies by ChromaTek, a subsidiary of American Paper Optics. This technology has been licensed for use in entertainment and promotional applications [2]. Furthermore, chromostereoscopy has been used for microscopy [14] and for visualising depth in remote sensing data [15].

As regular RGB displays cannot reproduce arbitrary colour spectra, techniques to generate more display-friendly chromostereoscopic images were also developed [11]. Further techniques have been developed to create additional depth cues to chromostereoscopic images, including adding shading to the depth image, and emphasising depth discontinuities by sharpening the depth image [8].

3 Method

The research that was conducted focuses on the creation of a program that takes an RGBD image and a set of user preferences, and generates a chromostereoscopic image that fulfills these preferences. Firstly, an initial program was developed to generate a chromostereoscopic image, with a number of parameters. This program can also optimise these parameters with respect to a metric.

Building off of this program, a final interface was created, which hides all of these parameters. Instead, the user can fill in their preferences (How much original colour do they prefer? How much of the chromostereoscopic depth effect do they prefer?). Following the results of a user study, these preferences modify the metric. Thus, a chromostereoscopic image can be generated that fulfills the user's wishes. The UI of both the intermediate and final program can be seen in figure 1.

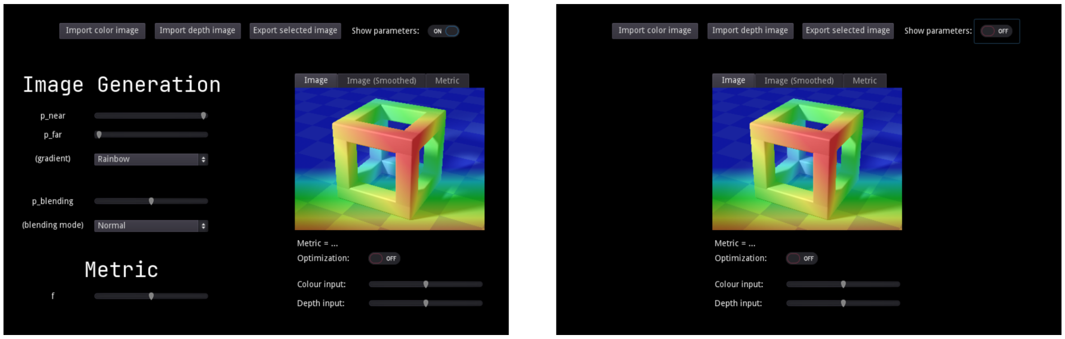

Figure 1: Left: the intermediate program, with all colour generation parameters and the metric parameter. Right: the final program, which only has colour and depth preferences.

3.1 Image Generation

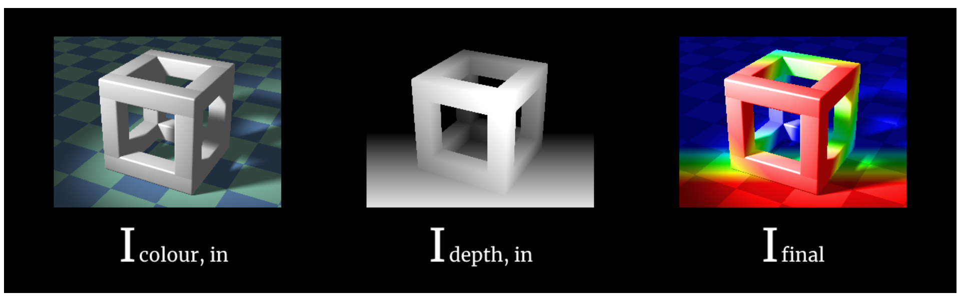

Figure 2: Examples of images which are used throughout the program. Left: the input colour image $I_{colour,in}$. Centre: the input depth image $I_{depth,in}$. Right: an example of a chromostereoscopic image I_{final} created with the image generation process (note: this image is an arbitrary, unoptimised example).

The input RGBD image is an image that consists of an RGB image $I_{colour,in}$ and a single-channel depth image $I_{depth,in}$. The final (chromostereoscopic) image I_{final} is made by passing this image through a pipeline of filters. This pipeline consists of a depth-to-colour mapping, followed by blending this resulting colour map with the original image. Examples of these images can be seen in figure 2. The process to generate chromostereoscopic images is illustrated with a data-flow diagram in Appendix A.

3.1.1 HSL

There exist formulas to convert colour values in the RGB colour model to those in the HSL colour model, and vice versa [1]. These formulas are used throughout the program. For the sake of illustrating the steps the program takes, the following formulas are needed:

- A function $RGBtoHSL(r, g, b)$ to convert colours from the RGB colour model to the HSL colour model;

- A function $HSLtoRGB(h, s, l)$ that performs the opposite operation;
- Functions $Hue(c), Saturation(c), Lightness(c)$ that take a colour in the HSL colour model and separate the H, S and L channels respectively.

Note: in the following formulas in this chapter, all colour channel values lie in the range $[0, 1]$.

3.1.2 Near/Far Range Transformation

The first step of the image generation procedure constists of transforming the depth image, with respect to a near and far range p_{near} and p_{far}. These ranges allow for 'focusing' the stereoscopic effect to a region of the depth range of the image. The effect of this can be seen in figure 2, as the red and blue areas are shifted closer to eachother. The two parameters both are in the interval $[0, 1]$. The formula for transforming an input depth d with respect to p_{near} and p_{far} goes as follows:

$$RangeTransformation(d, p_{near}, p_{far}) = min(max(\frac{d - p_{far}}{p_{near} - p_{far}}, 0), 1)$$

3.1.3 Depth Mapping

After transforming the depth value to the near and far range, two depth-to-colour mappings are available: 'Rainbow', which goes from red to blue on the colour spectrum, and 'Red-to-white-to-blue', which goes from red to white to blue.

$$RainbowDepthMapping(d) = HSLtoRGB(\frac{2}{3} - \frac{2}{3} \cdot d, 1, \frac{1}{2})$$

$$RedToWhiteToBlueMapping(d) = \begin{cases} (2 \cdot d, 2 \cdot d, 1) & \text{if } d < 0.5 \\ (1, 2 \cdot (1 - d), 2 \cdot (1 - d)) & \text{otherwise} \end{cases}$$

3.1.4 Image Blending

After passing the depth value through the range transformation and depth mapping, the new colour $c_{gradient}$ will be mixed back in with the original input colour $c_{colour,in}$. The colour blending has a parameter to determine the alpha to use for blending. As well as that, two blending modes are available: 'Normal', which is regular alpha blending, and 'Colour', which substitutes the colour while keeping the saturation and lightness. The 'Colour' blending mode has a corrective factor for 'whiteness': the more $c_{gradient}$ tends to white, the more the original colour is kept instead of blending with the gradient. This corrective factor only makes a difference in the 'Red-to-white-to-blue' depth mapping mode.

$$NormalBlending(c_{colour,in}, c_{gradient}, p_{blending}) = p_{blending} \cdot c_{gradient} + (1 - p_{blending}) \cdot c_{colour,in}$$

$$c_{target} = HSLtoRGB(Hue(c_{gradient}), Saturation(c_{gradient}), Lightness(c_{colour,in}))$$

$$p'_{blending} = p_{blending} \cdot 2 \cdot (1 - Lightness(c_{gradient}))$$

$$ColourBlending(c_{colour,in}, c_{gradient}, p_{blending}) = p'_{blending} \cdot c_{target} + (1 - p'_{blending}) \cdot c_{colour,in}$$

An overview of examples of generated chromostereoscopic images, based on the different depth mapping and colour blending modes, can be found in figure 3.

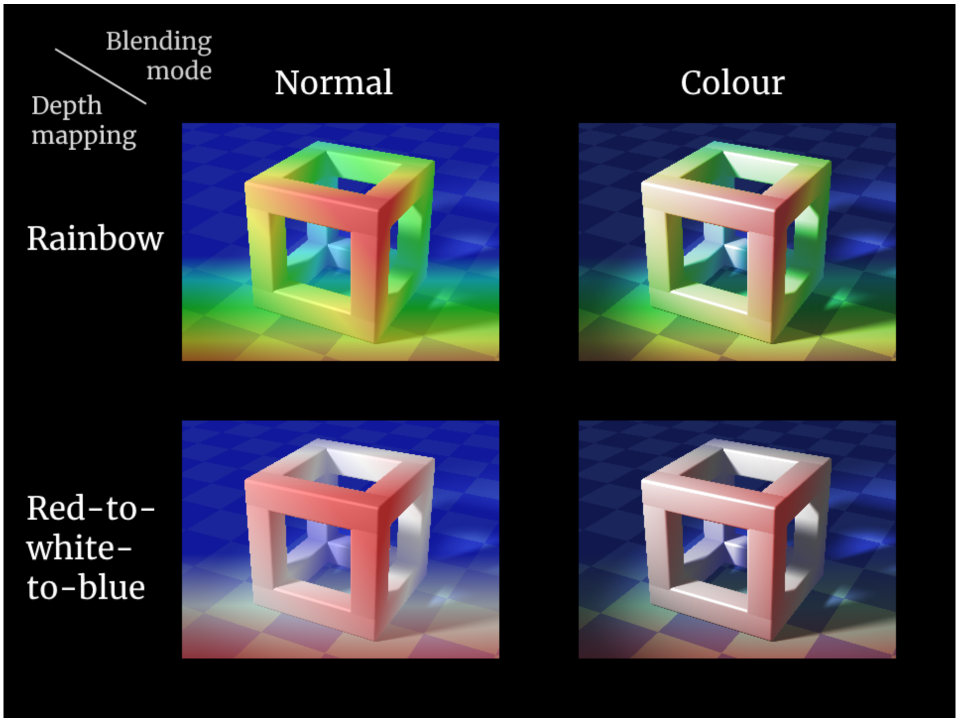

Figure 3: An overview of the effect of the different depth mapping modes and blending modes. Note: for the sake of demonstration, these images were generated with the same default settings ($p_{near} = 1$, $p_{far} = 0$, $p_{blending} = 0.5$) and thus were not optimised.

3.2 Image Evaluation

In order to generate an optimal image, the program will need to have a notion of what it means for a chromostereoscopic image to be successful. In the scope of this project, this means that the final image ought to be 'as natural as possible', and 'as deep as possible', with some factor to balance between these two. The data-flow diagram of the metric calculation process is illustrated in Appendix B.

3.2.1 'As natural as possible'

Because I argue that the input colour image is the 'most natural image', this part of the assessment will consist of a distance between the final image and the input RGB image. It is the average of per-pixel squared colour differences between the final image and the input colour image. After this formula was implemented, the metric was found to disproportionately 'punish' small colour differences. Squaring the sum terms solved this problem. CIEΔE2000 was chosen as it closely models human colour difference perception [12].

$$V_C(I_{colour,in}, I_{final}) = \frac{1}{m \cdot n} \cdot \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [(CIE\Delta E2000(I_{final}(i,j), I_{colour,in}(i,j)))^2]$$

As the smallest colour difference is desired, the colour metric V_C ought to be minimised.

3.2.2 'As deep as possible'

The second part of the metric accounts for a difference in depth. In order to assess the depth of the final image, the program will take the final image and convert it to a reconstructed depth map (an image with the depths, as the observer will perceive them). This is done via the following formula, executed per pixel of the final chromostereoscopic image:

$$ReconstructedDepth(c_{final}) = 0.5 \cdot (1 - Hue(c_{final})) + Saturation(c_{final}) \cdot Hue(c_{final})$$

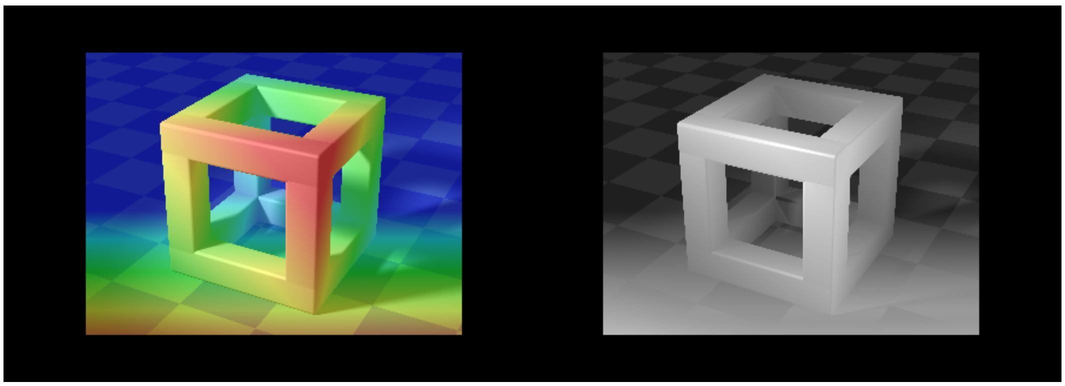

Figure 4: Left: a chromostereoscopic image. Right: a reconstruction of the depth image, based on this chromostereoscopic image.

An example of a depth reconstruction can be found in figure 4.

In this model, if a pixel is unsaturated, the user perceives this pixel at a depth of 0.5. This is because, on average, the ChromaDepth glasses do not bend unsaturated light [15]. If the saturation is 1.0, the user perceives a depth depending on the hue of the pixel. This model is a simplification, as it does not take exact colour spectra nor larger-scale depth cues into account. I argue that this simplification is sufficient for the purpose of this paper, though the procedure is open for future improvement.

I classify the 'most deep image' as the image where the perceived depth map is as close as possible to the input depth image. This is computed as an average of the squared difference between the input depth $I_{depth,in}$ and the reconstructed depth. Like the formula for V_C, the sum terms are squared to de-emphasise the 'punishment' caused by small differences.

$$V_D(I_{depth,in}, I_{final}) = \frac{1}{m \cdot n} \cdot \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [(I_{depth,in}(i,j) - ReconstructedDepth(I_{final}(i,j)))^2]$$

As the smallest depth difference is desired, the depth metric V_D ought to be minimised.

3.2.3 Striking a Balance

The final metric is a linear combination of the two metrics:

$$V(I_{colour,in}, I_{depth,in}, I_{final}) = f_D V_D(I_{depth,in}, I_{final}) + f_C V_C(I_{colour,in}, I_{final})$$

V can be transformed into a linear interpolation

$$V(...) = a(f'V_D(...) + (1 - f')V_C(...))$$

Where

$$a = f_C + f_D, \qquad f' = \frac{f_D}{f_C + f_D}$$

In optimisation, the factor a is irrelevant, as the minimum of the metric does not change when scaled. This results in a final metric:

$$V(I_{colour,in}, I_{depth,in}, I_{final}) = f \cdot V_D(I_{depth,in}, I_{final}) + (1 - f) \cdot V_C(I_{colour,in}, I_{final})$$

Thus, $f = 0$ will mean that the metric only takes the colour image into consideration, and $f = 1$ will mean that the metric only takes the chromostereoscopic effect into consideration. Furthermore, because both V_D and V_C ought to be minimised individually, V also has to be minimised. The optimal chromostereoscopic image therefore has the minimal V.

4 Implementation

To make use of chromostereoscopic techniques, a program has been developed that converts an RGBD image into a chromostereoscopic image. Furthermore, due to the fact that image quality is a subjective measure, some more artistic control is preferred. Therefore, the program was developed to be interactive and UI-based, to lend a degree of freedom to the research and to facilitate more open-ended interactive experimentation.

4.1 Image Generation

The free and open-source game engine Godot was used to make the program. This game engine comes with a number of components necessary to make the program: a UI framework, dynamic image loading, texture drawing, shader support, and more.

To process the images, they are first imported as textures. After this, a shader processes the final image based on the input depth and colour images and the parameters specified for the pipeline (near/far range, blending amount). Afterwards, a second shader calculates the per-pixel metric based on the final image and the input depth and colour images. The metric is then averaged to a single value by reading the texture on the CPU.

The shaders are made in Godot's proprietary shading language. All of the program logic (opening file dialogs, setting shader parameters when the UI is changed) is made in Godot's proprietary GDScript language. According to the Godot documentation, the shading language is similar to GLSL ES 3.0 and the scripting language is similar to Python [7]. Snippets of these languages can be found in Appendix C.

4.2 Optimisation

To optimise the metric, cyclic coordinate descent was used. This technique modifies one parameter at a time, moving towards an optimum, after which the next parameter is chosen, until the metric is at an optimum [3]. Because Godot only updates the texture every frame, every parameter is only shifted slightly every frame. The full optimisation algorithm is thus as follows:

Algorithm 1 Optimisation Algorithm

$metric_{prev} \leftarrow V(I_{colour,in}, I_{depth,in}, I_{final})$
$parameter_{prev} \leftarrow p_i$
$p_i \leftarrow p_i \pm \delta$ ▷ Randomly increment/decrement the parameter
Wait a frame.
$metric_{new} \leftarrow V(I_{colour,in}, I_{depth,in}, I_{final})$ ▷ Compute the new metric value
if $metric_{new} > metric_{prev}$ **then**
  $p_i \leftarrow param_{prev}$ ▷ Restore the old parameter value
end if
$i \leftarrow (i + 1) \bmod |p|$ ▷ Move on to the next parameter
Repeat.

This is not the fastest way of optimising the image, but with $\delta = 0.002$, the optimal image is found within 30 seconds to 1 minute. This was sufficient for generating the images for the user study. In the future, work can be done to speed up this optimisation, for instance by using another black-box derivative-free descent algorithm, like the Nelder-Mead method [3]. In order to increase performance, all images have a relatively small resolution of 320 by 240 pixels. All RGBD images are stretched to this resolution, when importing them.

4.3 User Study

The program used for the user study is a separate program from the image generation program. It has also been developed in Godot, using the basic UI components already present in the engine. The program keeps track of the preferences via a two-dimensional list. After the study finishes, this list is converted to comma-separated data which is shown to the user.

5 Experimental Setup

In the final program, the user can input two preferences (how much colour do they want; how much depth do they want). Therefore, a mapping from a desired colour and depth score to the corresponding metric factor f has been made, following a user study.

A choice was made to make the study a remote user study. The participants were given a link to a website, where the entire study took place, without guidance on-location. This was done for a few reasons. The remoteness of the study allowed for participants to schedule the study during their own free time. This made the study more time-efficient, because sessions where participants needed to be physically present did not need to be scheduled. As well as that, the study does not run the risk of being influenced by the interviewer [10]. The setup of the study, along with its short duration, also made a revision of the study procedure possible.

The study procedure starts off with a textual explanation of chromostereoscopy, as shown in Appendix D. Then, the participants are given written instructions on the setup of the study. Before starting, the participants have to ensure that their environment satisfies a number of conditions (ie. their room is darkened, they have no active screen filters, ...). The participants were also instructed to download and sign an informed consent form. This informed consent form explained the setup of the study once more, and detailed the possible risks in a bullet point list.

After continuing, a preview of an RGBD picture is shown. The participants are then shown multiple generated chromostereoscopic variations of this picture, one after the other. The difference between these variations is the metric factor f that was used for generating the variation. The users can specify the amount of perceived (original) colour and the amount of perceived depth that the shown variation exhibits. This can be seen in Appendix E. This procedure is repeated with 11 variations of 3 images, totalling 33 images to be 'scored'. To decrease bias in the results, the order that the 3 source images were shown in was randomied. Within each source image, the order of the variations was also randomised.

For this user study, the 'Colour' blending mode and 'Rainbow' depth mapping were chosen for creating the chromosteroscopic images. The 'Colour' blending mode was chosen, because this keeps more of the original texture detail, especially with a high $p_{blending}$. The 'Rainbow' depth mapping was chosen, as it produced the best chromostereoscopic effect to me, when combined with the 'Colour' blending mode.

I decided to use real-life photos for the research, as I aimed to study only one type of images. This made it possible to use more suitable images (more colours are used, and similar colours are used at different depths). More work can be done in the future to generalise this method to any arbitrary image. The photos used for the study, the corresponding depth images and a brief explanation of why they were chosen can be found in Appendix F. The chosen photos are fed into an off-the-shelf depth image generation method [6]. As well as that, this method ensured that all depth images were of the same quality.

After all information has been collected and anonymised, it is analysed. The data will consist of multiple (f, s_C, s_D) points, where f is the metric linear interpolation factor that was used to optimise the chromostereoscopic image, and s_C and s_D are the indicated colour and depth 'scores'. The analysis will consist of finding a mapping that takes a novel colour and depth score, and predicts the best metric factor f to generate the chromostereoscopic image with.

6 Responsible Research

In order to conduct responsible research, one has to consider a handful of guidelines. Firstly, the results ought to be reproducible. As well as that, the results must have been acquired and kept ethically. This section will outline how this research adhered to these two points.

In order to make this study reproducible, and to facilitate for additional research extending this method, all of the code and material that was created in this study has been published on the GitHub page of the CSE3000 Research Project [4]. The repository contains the Godot program files for the chromostereoscopic image generation program and the user study program. The Jupyter notebook that was used for analysing the resulting data is also present in this repository.

The user study was carried out with the standard TU Delft responsible research procedures. Before the study took place, an application to the Human Research Ethics Committee (HREC) of the TU Delft was made. The application handled my methodology with respect to data collection and storage, privacy, informed consent and other ethical points. This application was accepted and the study was carried out, following the plan in the HREC application.

7 Results

After conducting the user study, a set of points was obtained. These points were metric factors f, and their corresponding colour and depth scores s_C and s_D. After analysis, a mapping m from s_C and s_D to f will complete the program. The user will fill in a novel preverence s'_C, s'_D, and the predicted metric factor will be computed as $\hat{f} = m(s_C, s_D)$.

In theory, when f is 0, the colour score should be at its highest, as only the colour metric V_C is used in the linear interpolation step of the metric. V_D should be at its lowest. Similarly, when f is 1, the depth score should be at its highest, as only the depth metric V_D is used. V_C should be at its lowest.

Ahead of the user study, I suspected the data to be substantially spread out, as peoples' opinions can vary a lot. This is especially the case for opinions on colour, as colour perception can vary highly from individual to individual [17]. Because chromostereoscopy is ultimately a trade-off between colour and depth, I also suspected that the data would lie on a graph of the form:

$$\hat{f} = a * (\frac{s_D}{s_C + s_D}) + b$$

This is the same formula as given in section 3 to go from a linear combination to a linear interpolation. I thus suspected that the metric would be linearly dependent on the depth score, when mapped such that the two user-given scores form a linear interpolation.

The user study had 5 participants. Because every participant scored 11 variants of 3 images, the data consists of 165 points of the form (f, s_C, s_D). After creating a scatterplot of the data (see figure 5), an indication of a correlation between the used metric factor f and the colour and depth scores s_C and s_D is visible. When $f = 0$ (at the bottom plane of the graph), the colour and depth scores lie around (1, 0) respectively. When $f = 1$ (at the top of the graph), the colour and depth scores lie around (0, 1). This follows the theory described earlier.

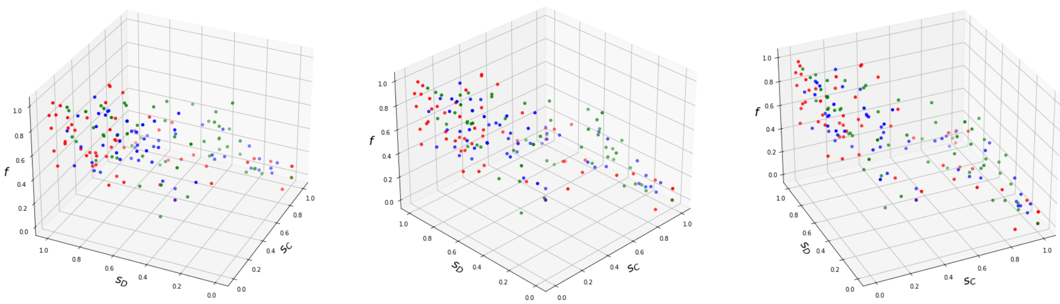

Figure 5: A scatterplot of the results of the user study. The metric factor f which was used goes along the vertical axis, and the colour and depth scores which were given go along the horizontal axes. The data points are coloured per RGBD image that the scores were given on. Three views are given for visual clarity.

At first, a least-squares linear regression was carried out on the data points. After performing this regression of the metric parameter f, and their corresponding colour and depth scores s_C and s_D, it could be plotted as a plane in 3D, as can be seen in figure 6.

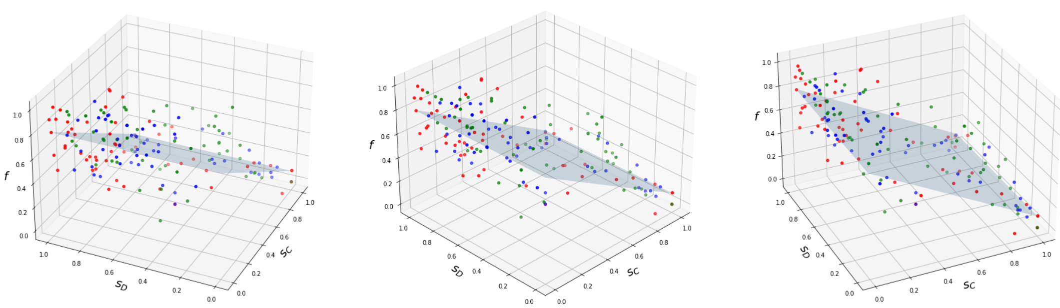

Figure 6: The same scatterplot as before, but with a least-squares linear regression plotted alongside the points. The formula of the regression is $\hat{y} = -0.702 \cdot s_C + 0.003 \cdot s_D + 0.798$.

The linear regression has a coefficient of determination $R^2 = 0.6243$. This means that in this model, 62.4% of the variance of the metric parameter f can be explained by the variance in the depth scores s_C and s_D.

I also made a regression of the metric parameter f, as a dependent variable with respect to $\frac{s_D}{s_C+s_D}$, to further explore my hypothesis given above. After reducing the two scores to the single value, the regression was determined, as can be seen in figure 7 and 8.

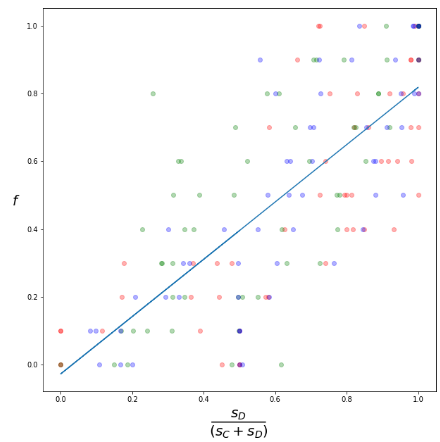

Figure 7: A linear regression of the points with metric parameter f, as a dependent variable with respect to $\frac{s_D}{s_C+s_D}$. The formula of the regression is $\hat{f} = 0.847 \cdot (\frac{s_D}{s_C+s_D}) - 0.028$.

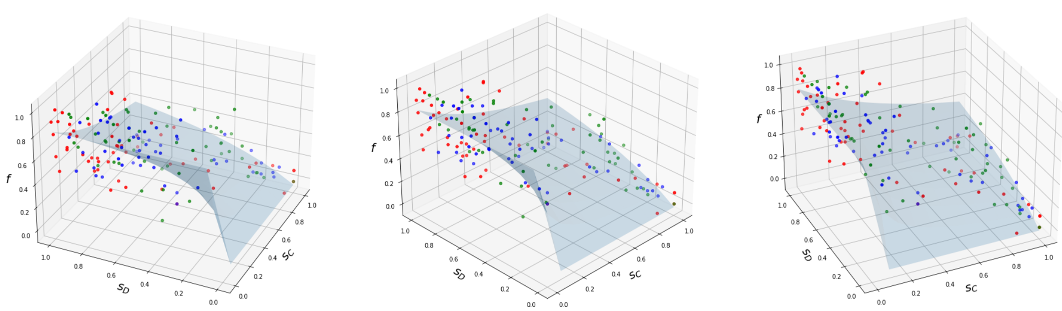

Figure 8: The scatterplot of the data, with the regression plane derived from the regression shown in figure 7.

This regression has an R^2 of 0.580, this is lower than the simple linear regression. However, because I argue that this model more closely represents the nature of chromostereoscopic images, this second regression was chosen for the final program.

8 Discussion

With these results, the final program is finished. The user first inputs their novel colour and depth scores. Next, the program determines a metric parameter f, via the regression outlined in section 7. Then, optimisation takes place on the image generation parameters with respect to the metric, as outlined in section 3. A final chromostereoscopic image is generated, that takes the user's preferences into account.

A notable case where this method performs suboptimally is when the user specifies both a maximal preferred colour and depth score. This would mean that both the original colour and the depth map are perfectly adhered to in the final image. The regression solves this problem by mapping the scores from $(1, 1)$ to a an image where both the colour and depth are accounted for equally.

Results could be improved by involving more participants in the user study. As well as that, more RGBD images could be used. This would lead to more points in parts of the plot that are not filled in yet (where scores lie at $(0, 0)$ or $(1, 1)$). Furthermore, different depth mapping and colour blending modes could be tested, as these are already available and usable in the program.

9 Conclusion and Future Work

The goal of this paper was to outline a technique that can generate chromostereoscopic images, based on user preferences. First, a method for generating chromostereoscopic images was made. Then, a formula for evaluating a chromostereoscopic image for depth and colour was created. This optimisation was parametric. Following a user study, a mapping from user preferences to optimisation parameters was established, finishing the final technique. Screenshots of some examples of the final program can be found in figure 9.

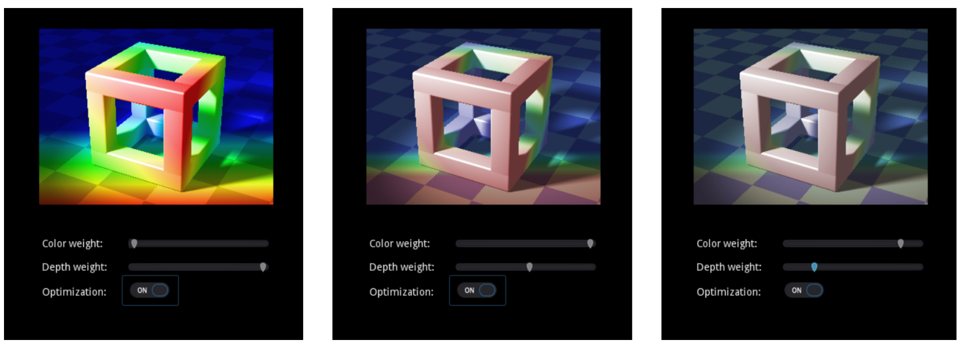

Figure 9: A few examples of the final program, with different colour and depth preferences, and the corresponding optimised chromostereoscopic images.

Further research could be done on adding more filters and parameters to the image generation pipeline (increasing depth details by unsharp masking the depth image, Lambert shading, brightness and contrast parameters), to see if this improves the chromostereoscopic output. A better understanding of chromostereoscopic depth buffer reconstruction could lead to a more accurate depth metric V_D. Because the source code of the final program, user study program and user study analysis are available, futher research could be done by building upon the method and implementation outlined in this paper.

A Appendix: Image Generation Data-Flow Diagram

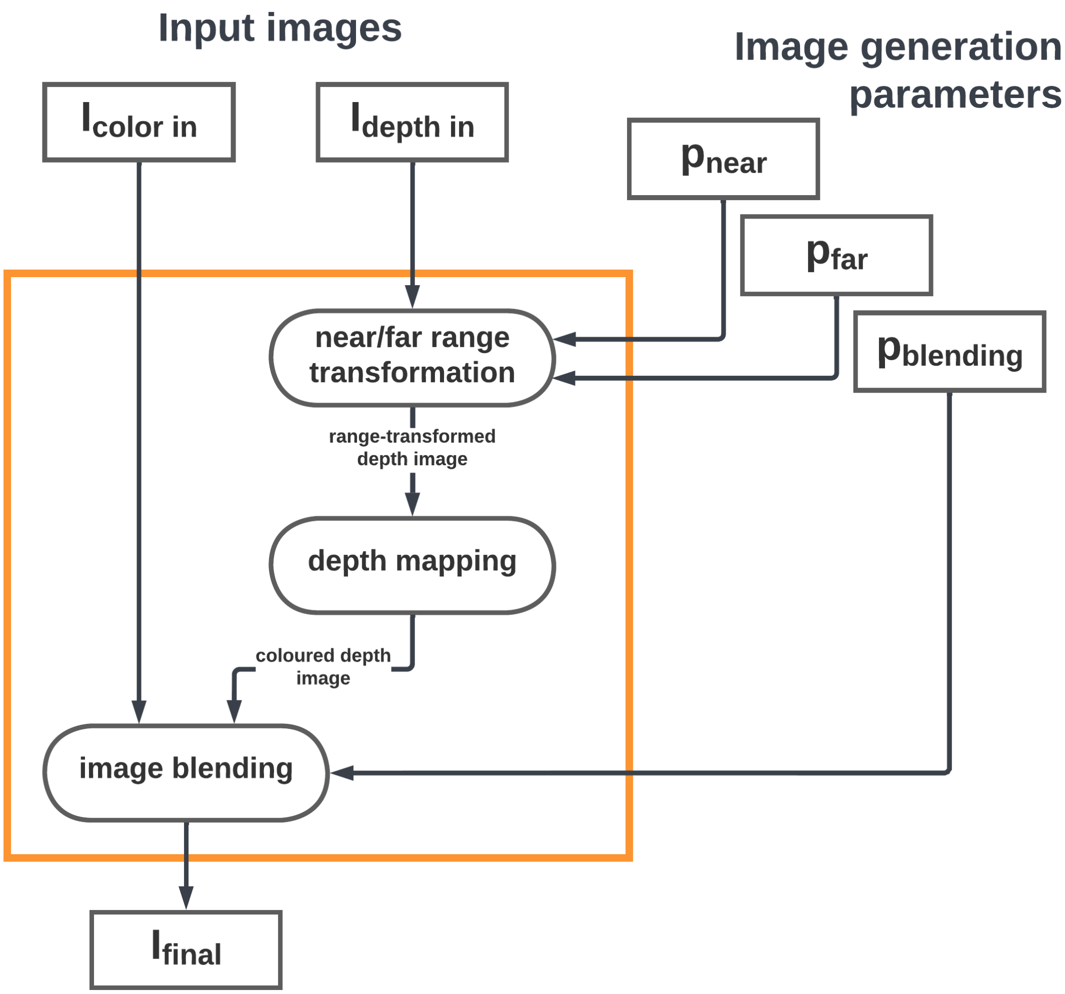

Figure 10: A data-flow diagram for chromostereoscopic image generation.

B Appendix: Metric Calculation Data-Flow Diagram

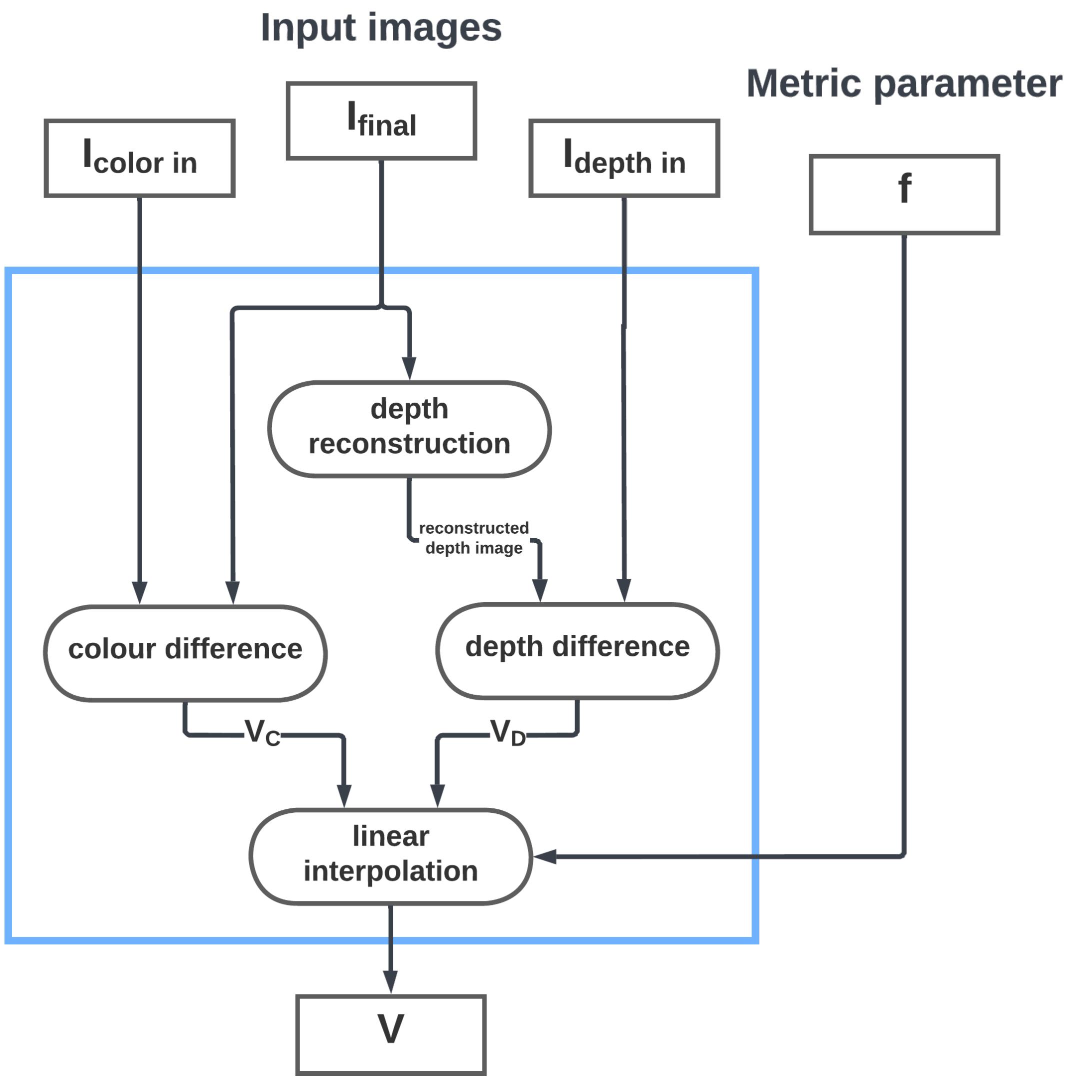

Figure 11: A data-flow diagram for metric calculation.

C Appendix: Code Snippets

```
# recalculates the metric based on the image
func calculate_metric() -> float:
    # get metric image
    var metric_data: Image = metric_viewport.get_texture().get_data()

    # lock image
    metric_data.lock()

    var metric := 0.0

    for y in range(240):
        for x in range(320):
            metric += metric_data.get_pixel(x, y).r

    metric /= 320 * 240

    # unlock image
    metric_data.unlock()

    # set label text
    metric_label.text = "Metric = " + str(metric)

    return metric
```

Figure 12: A snippet of GDScript. This snippet takes the output of the metric shader, and averages it to a final metric value.

```
// convert values to hsl
vec3 c_gradient_hsl = rgb_to_hsl(c_gradient);
vec3 c_colour_in_hsl = rgb_to_hsl(c_colour_in);

// get hue and saturation of gradient, keep lightness of original colour image
c_gradient = hsl_to_rgb(vec3(c_gradient_hsl.x, c_gradient_hsl.y, c_colour_in_hsl.z));

// correct blending mode to white colour
float new_p_blending = 2. * p_blending * (1. - c_gradient_hsl.z);

// interpolate with factor
COLOR.rgb = mix(c_colour_in, c_gradient, new_p_blending);
```

Figure 13: A snippet of Godot Shading Language. This snippet implements the 'Colour' blending mode, blending between $c_{colour,in}$ and $c_{gradient}$.

D Appendix: User Study Program Overview

Figure 14: The full introduction used in the user study program.

E Appendix: User Study Program Prodedure

Figure 15: The overview of one of the 3 source images, before user scoring.

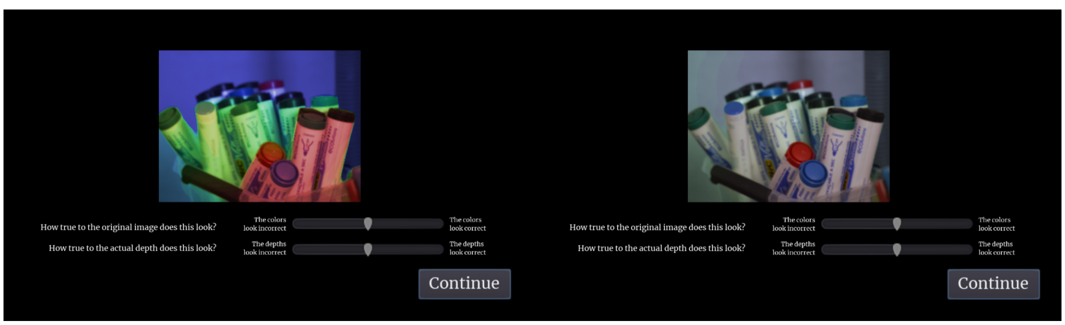

Figure 16: Two variations of chromostereoscopic images derived from the source images, with the scoring interface.

F Appendix: User Study Program RGBD Pictures

The following 3 RGBD images were used in the user study program:

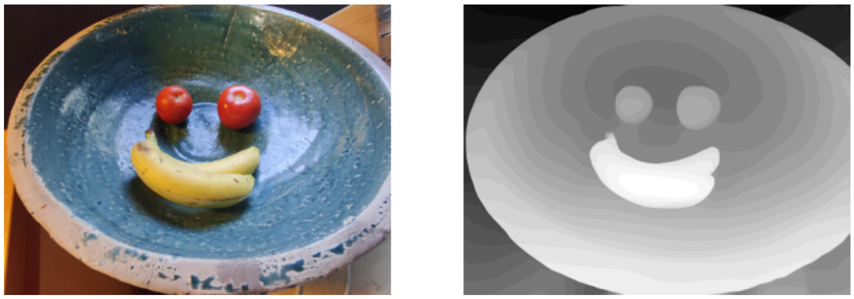

Figure 17: A picture of tomatoes and bananas arranged in a large bowl. The red tomatoes and yellow banana can quickly lose their original colour when chromostereoscopic depth cues are added, and thus were deemed interesting for the user study.

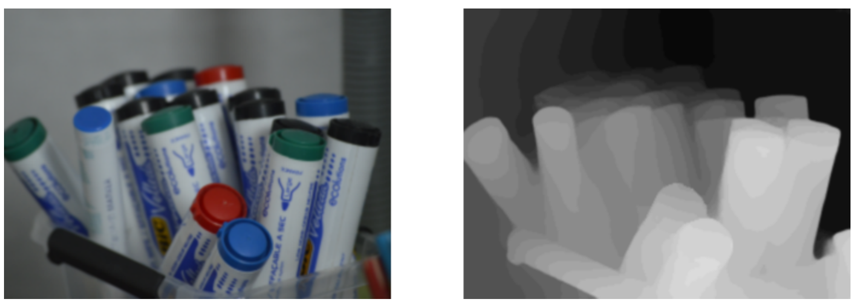

Figure 18: A picture of whiteboard markers in a plastic container. The differently coloured markers at different depths in the image provides a challenge for balancing the colour and depth in the chromostereoscopic image.

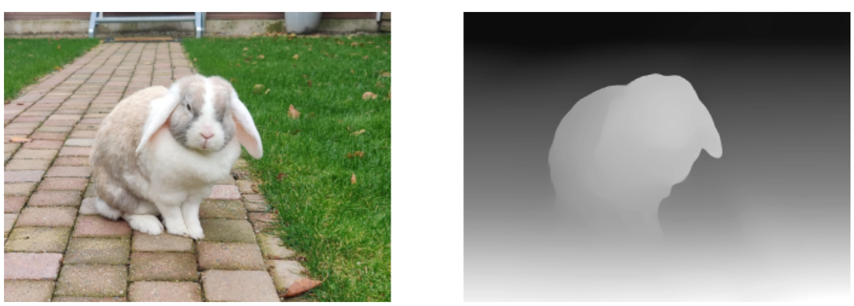

Figure 19: A picture of a rabbit perched on a garden path. The depth disparity between the main figure and background gives potential for an interesting chromostereoscopic effect. A large patch of green grass stretching throughout the depth of the image gives a challenge in perserving this colour.

References

[1] M. K. Agoston. *Computer graphics and geometric modeling*, volume 1. Springer, 2005.

[2] American Paper Optics, LLC. "chromadepth 3-d glasses," www.3dglassesonline.com. (accessed 2022-06-04). URL: `https://www.3dglassesonline.com/products/chromadepth/`.

[3] C. Audet and W. Hare. *Derivative-free and blackbox optimization*, volume 2. Springer, 2017.

[4] Delft University of Technology. "cse3000 research project," github.io. (accessed 2022-06-18). URL: `https://cse3000-research-project.github.io/`.

[5] W. Einthoven. Stereoscopie durch farbendifferenz. *Albrecht von Graefes Archiv für Ophthalmologie*, 31(3):211–238, 1885.

[6] K. Lasinger, R. Ranftl, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *CoRR*, abs/1907.01341, 2019. URL: `http://arxiv.org/abs/1907.01341`, `arXiv:1907.01341`.

[7] J. Linietsky, A. Manzur, and the Godot community. "godot docs - 3.4 branch," www.godotengine.org (revision 9fd99de6). (accessed 2022-06-06). URL: `https://docs.godotengine.org/`.

[8] T. Luft, C. Colditz, and O. Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics (TOG)*, 25(3):1206–1213, 2006.

[9] R. Ostnes, V. J. Abbott, and S. Lavender. Visualisation techniques: An overview - part 2. *Hydrographic Journal*, pages 3–9, 10 2004.

[10] C. J. Pannucci and E. G. Wilkins. Identifying and avoiding bias in research. *Plastic and reconstructive surgery*, 126(2):619, 2010.

[11] L. Schemali and E. Eisemann. Chromostereoscopic rendering for trichromatic displays. In *Proceedings of Expressive 2014 (ACM Symposium on Non-Photorealistic Animation and Rendering)*, pages 57–62, New York, NY, USA, August 2014. ACM. doi>10.1145/2630397.2630398. URL: `http://graphics.tudelft.nl/Publications-new/2014/SE14a`.

[12] G. Sharma, W. Wu, and E. N. Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005.

[13] R. A. Steenblik. The Chromostereoscopic Process: A Novel Single Image Stereoscopic Process. In David F. McAllister and Woodrow E. Robbins, editors, *True Three-Dimensional Imaging Techniques & Display Technologies*, volume 0761, pages 27 – 34. International Society for Optics and Photonics, SPIE, 1987. `doi:10.1117/12.940117`.

[14] R. A. Steenblik. Chromostereoscopic microscopy. In *Three-Dimensional Visualization and Display Technologies*, volume 1083, pages 60–64. SPIE, 1989.

[15] T. Toutin. Qualitative aspects of chromo-stereoscopy for depth perception. *Photogrammetric Engineering and Remote Sensing*, 63(2):193–204, 1997.

[16] E. Verwichte and K. Galsgaard. On the visualization of three-dimensional datasets. *Solar Physics*, 183(2):445–448, 1998.

[17] M. A. Webster and D. I. A. MacLeod. Factors underlying individual differences in the color matches of normal observers. *JOSA A*, 5(10):1722–1735, 1988.