Reducing overexposure and underexposure during image-guided surgery — part 2

October 29, 2023

Kai Kitagawa-Jones
student ID: i6275822

1 Abstract

When capturing video during surgery or dental procedures, many areas of the frame are often underexposed or overexposed. This is caused by the combination of the strong lighting used, and the often very deep wounds where large parts of the procedure are performed. This report improves and extends my previous report, where I used auto exposure to reduce both underexposure and overexposure. In this report, the exposure fusion process is improved, and an auto exposure (AE) control algorithm that works well in combination with exposure fusion is implemented.

2 Introduction

In my previous report, the exposure fusion technique [1, 2] was used in order to produce a well-exposed image, i.e. an image with little underexposure and overexposure. Exposure fusion works by combining two or more images with different exposures in order to produce a single image. The images are combined in a way such that underexposure and overexposure are minimized in the resulting image. The implementation of this technique, discussed in my previous report, used two such images, and resulted in a significant decrease in both underexposure and overexposure. However, in certain cases, such as when neighbouring areas of an image have significantly different brightnesses, seams appear in the resulting image. This is an issue also encountered by [2]. In this report, a method to reduce the impact of this issue without significantly decreasing the performance of the algorithm is discussed.

The implementation from my previous report performs auto exposure in the RGB colour space. The RGB colours were used to calculate estimates for saturation and "well-exposedness", as described in [2]. These estimated values are similar to the saturation and lightness channels in the HSL colour space [3] or the saturation and value channels in the HSV colour space [4]. Because of this, in this report, the use of alternative colour spaces when performing exposure fusion is also discussed. Because the exposure fusion process is part of a larger pipeline of image processing steps, using colour spaces such as HSL or HSV may also increase performance. This is because it avoids unnecessary colour space conversions, as other parts of the image processing pipeline also use colour spaces such as HSL or HSV.

In order to use exposure fusion in practice, some form of AE control is required. This is especially important during surgery, where it is necessary for the camera to adapt to changes in brightness. The changes in brightness can happen when looking at or looking away from areas under bright surgical lights. Since exposure fusion uses a number of frames, each at different exposures, standard auto exposure implementations [5] cannot be used. Therefore, this report also discusses ways to implement AE control for exposure-fusion-based video capture.

To summarize, this report answers the following questions:

- How can the seams that appear as a result of exposure fusion be reduced?
- When performing exposure fusion, what colour space produces the best result?
- How can AE control be implemented for use with exposure fusion?

All methods in this report are designed to run on the DHM system [6] from i-Med Technology. Because of this, all methods must run in real-time on a 3840×1080 pixel video feed, at 60 frames per second. Therefore, performance is the one of the main focuses of this report. In order to achieve acceptable performance, many of the methods discussed in this report are designed to be run on the GPU, using OpenGL compute shaders [7].

3 Methods

3.1 Basic exposure fusion

Exposure fusion is a technique to obtain images that are well-exposed in both dark and bright areas. This is done by capturing a number of input frames at different brightnesses, and combining them in a way such that well-exposed areas of each frame are used. For example, if two frames, one dark and one bright, are used: bright parts are taken from the dark frame, and dark parts are taken from the bright frame.

In order to combine the input frames, first, the frames are divided into blocks. Then, for each block in each frame, a score is calculated. The scores must be calculated in a way such that well-exposed blocks have higher scores when compared to blocks that are not well-exposed. Once these scores have been calculated, each set of blocks is combined as shown in Equation 1.

$$O = \sum_{i=1}^{n} w_i \cdot I_i$$
$$\text{where,} \tag{1}$$
$$w_i = \frac{\text{score}(I_i)}{\sum_{j=1}^{n} \text{score}(I_j)}$$

$I_1 \cdots I_n$ are the input blocks, and O is the output block. The output is calculated by adding each input block, scaled by its normalized score.

Once the blocks have been combined, the resulting output blocks are combined into a single output frame. An overview of the exposure fusion process is shown in Figure 1.

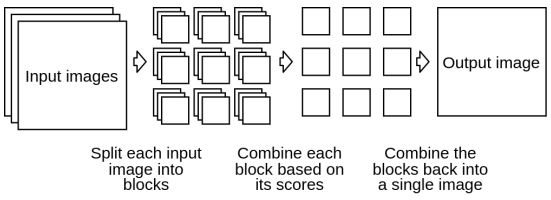

Figure 1: An overview of exposure fusion

In my previous report, the scores used when combining blocks were calculated by combining estimates of the saturation and exposure of each frame in the RGB colour space. The saturation and exposure was estimated in the following way:

- Saturation: the standard deviation between the R, G, and B channels:

  $$p_1 = \sqrt{(\bar{c} - R)^2 + (\bar{c} - G)^2 + (\bar{c} - B)^2}$$

  where $\bar{c} = \frac{R+G+B}{3}$.

- Exposure: how close the average of the R, G, and B channels is to 0.5, based on a Gauss curve with $\sigma = 0.2$:

  $$p_2 = \exp\left(-\frac{(\bar{c} - 0.5)^2}{2\sigma^2}\right)$$

  $\sigma = 0.2$ was chosen as it results in a Gauss curve with values close to 0 at $c = 0$ and $c = 1$.

These estimates were then combined using Equation 2, where w_1 and w_2 are weights used to control the behaviour of the scoring.

$$\text{score} = p_1^{w_1} \cdot p_2^{w_2} \tag{2}$$

There are two common block sizes that are used in exposure fusion:

- A 1×1 block size, i.e. scores are calculated for each pixel [2, 8].
- A block size larger than 1×1, such as 8×8 or 16×16 [1, 9].

For the implementation in this report, a 1×1 block size is used since it allows all steps of the exposure fusion process to be easily run in parallel, on the GPU. In addition, two input frames are used to perform exposure fusion, as a larger number of frames leads to significant motion blur.

3.2 Improving exposure fusion

As mentioned in the introduction, the improvement that this report focuses on is the removal of seams that appear in the final image. There are a number of approaches that can be used to remove such seams from the image:

- Perform exposure fusion on a Laplacian pyramid image decomposition [10], an approach also taken by [2]. This approach requires a large amount of additional computations that cannot be easily parallelized.
- Blur the scores used when combining pixels, so that there are no sudden changes. An efficient blurring algorithm has to be implemented for this approach to be viable.

- Do not use exposure fusion for channels that represent "brightness", such as the V channel in the HSV colour space. Instead, simply take the mean of the two inputs.

The approach of using a Laplacian pyramid image decomposition is not used in this report, as the additional performance required to do this in real-time is too large. On the other hand, the approach of blurring the scores used during exposure fusion can be implemented using a Gaussian filter. This is because its kernel is linearly separable [11], greatly increasing performance.

In my previous report, the RGB colour space was used throughout the exposure fusion process. However, alternative colour spaces may be more suitable for exposure fusion. Therefore, this report looks into the following alternative colour spaces.

HSL (hue, saturation, lightness)

When compared to the RGB colour space, the HSL colour space [3] is often more suitable for image processing, as properties of the colour can be adjusted more intuitively.

The H, S, and L components can be derived from RGB colours using Equation 3:

$$\begin{aligned} 6H &= \begin{cases} 3 - \frac{G-B}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } R = \mathrm{RGB}_{\max} \\ 5 - \frac{B-R}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } G = \mathrm{RGB}_{\max} \\ 1 - \frac{R-G}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } B = \mathrm{RGB}_{\max} \end{cases} \\ L &= \frac{\mathrm{RGB}_{\max} + \mathrm{RGB}_{\min}}{2} \\ S &= \frac{\mathrm{RGB}_{\max} - \mathrm{RGB}_{\min}}{1 - |2L - 1|} \end{aligned} \tag{3}$$

where,

$$\begin{aligned} \mathrm{RGB}_{\max} &= \max\{R, G, B\} \\ \mathrm{RGB}_{\min} &= \min\{R, G, B\} \\ \mathrm{RGB}_{\mathrm{range}} &= \mathrm{RGB}_{\max} - \mathrm{RGB}_{\min} \end{aligned}$$

HSV (hue, saturation, value)

The HSV colour space [4] is similar to the HSL colour space. However, compared to the HSL colour space, it can be shown to be slightly superior in terms of intuitive control of colours [12].

The H, S, and V components can be derived from RGB colours using Equation 4:

$$\begin{aligned} 6H &= \begin{cases} 3 - \frac{G-B}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } R = \mathrm{RGB}_{\max} \\ 5 - \frac{B-R}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } G = \mathrm{RGB}_{\max} \\ 1 - \frac{R-G}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } B = \mathrm{RGB}_{\max} \end{cases} \\ V &= \mathrm{RGB}_{\max} \\ S &= \frac{\mathrm{RGB}_{\max} - \mathrm{RGB}_{\min}}{V} \end{aligned} \tag{4}$$

HWB (hue, whiteness, blackness)

The HWB colour space [12] was designed as an improvement to HSL and HSV colour spaces, such that the saturation (the W component) has a fixed meaning regardless of the values of the H and B components.

The H, W, and B components can be derived from RGB colours using Equation 5:

$$\begin{aligned} 6H &= \begin{cases} 3 - \frac{G-B}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } R = \mathrm{RGB}_{\max} \\ 5 - \frac{B-R}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } G = \mathrm{RGB}_{\max} \\ 1 - \frac{R-G}{\mathrm{RGB}_{\mathrm{range}}} & \text{if } B = \mathrm{RGB}_{\max} \end{cases} \\ W &= \mathrm{RGB}_{\min} \\ B &= 1 - \mathrm{RGB}_{\max} \end{aligned} \tag{5}$$

To summarize, Figure 2 shows an overview of the entire exposure fusion process discussed above.

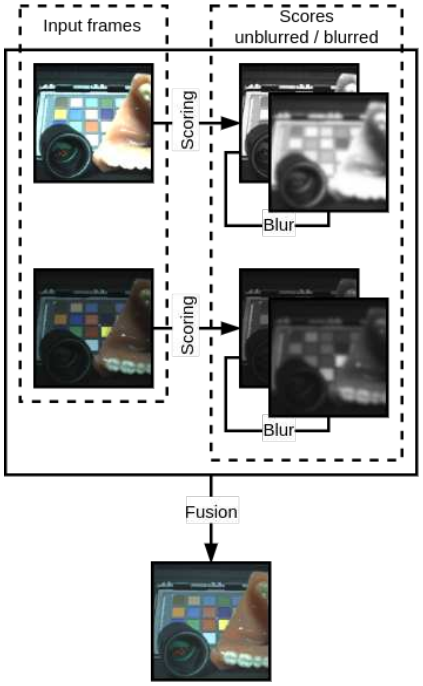

Figure 2: An overview of the exposure fusion process used in this report

3.3 AE control for exposure fusion

In order to use exposure fusion in practice, it is necessary that there is a way to perform AE control for the cameras on the headset. Because two frames are used for exposure fusion, AE control is also performed on two input streams S_1 and S_2. The camera can be controlled using the following parameters:

- Exposure time – The duration that each frame is exposed.
- Gain – The sensitivity of the camera sensor.

When adjusting these values, there are some limitations that have to be taken into account: long exposure times lead to motion blur, and high gain leads to noise in the acquired frames. In addition, while the gain can be changed every frame, changing the exposure time frequently results in frame drops. Therefore, exposure time t must be the same for both S_1 and S_2, while gains g_1 and g_2 can be different for each stream.

In order to simplify the AE control process, a camera controller is implemented so that the camera can be controlled using only two parameters, e_1 and e_2. The camera controller will adjust the values of t, g_1, and g_2 such that the following constraints are met:

$$\begin{cases} e_1 = t \cdot g_1 \\ e_2 = t \cdot g_2 \end{cases} \tag{6}$$

Equation 6 is valid, since both exposure time and gain linearly affect the brightness of the acquired frames [13]. This is also confirmed in Section 6.2 below.

The following steps are used to calculate the values of t, g_1, and g_2:

$$\begin{aligned} &i. \quad t_{\text{target}} \leftarrow \frac{\sqrt{e_1} + \sqrt{e_2}}{2} \\ &ii. \quad t \leftarrow \min(t_{\text{target}}, t_{\max}) \\ &iii. \quad g_1 \leftarrow \frac{e_1}{t} \\ &iv. \quad g_2 \leftarrow \frac{e_2}{t} \end{aligned} \tag{7}$$

First, the mean of the square root of e_1 and e_2, t_{target}, is calculated (i). t_{target} is the ideal value for t, as this keeps both the exposure time and gain as low as possible. This minimizes both motion blur due to long exposure times, and noise due to high gain. Next, the final value of t is obtained by capping t_{target} at $t_{\max}$, the maximum allowed value for t (ii). Finally, g_1 and g_2 can be calculated by dividing the values of e_1 and e_2 by t (iii, iv).

A linear feedback controller is used to adjust the exposure of the camera, based on scores s_1 and s_2 calculated for each input stream $S_1$and S_2, respectively. A negative score indicates that the frame is too dark, and exposure must be increased. Meanwhile, a positive score indicates that the frame is too bright, and exposure must be decreased. The controller adjusts the exposure of the camera such that $|s_1|$ and $|s_2|$ are minimized.

As the headset does not have an ambient light sensor, this score is computed based on the histogram of each frame H_1 and H_2 [5, 14, 15]. From these histograms, scores s_1 and s_2 are calculated as follows:

$$\begin{aligned} s_1 &= f_m(H_1) \cdot w_{m_1} - f_u(H_1) \cdot w_{u_1} + f_o(H_1) \cdot w_{o_1} \\ s_2 &= f_m(H_2) \cdot w_{m_2} - f_u(H_2) \cdot w_{u_2} + f_o(H_2) \cdot w_{o_2} \end{aligned} \tag{8}$$

- $f_m(H)$: Defined as $\frac{\text{median}(H) - 127.5}{255}$. The median was chosen over the mean, as it is less affected by extreme values. The value indicates the overall well-exposedness of the frame, approaching 0 when the median is close to the centre of the dynamic range of the camera: 127.5.
- $f_u(H)$: The number of underexposed pixels in the frame, taken from the left-most bin of H. The value approaches 0 when the frame has less underexposure.
- $f_o(H)$: The number of overexposed pixels in the frame, taken from the right-most bin of H. The value approaches 0 when the frame has less overexposure.
- $w_{m_1} \cdots w_{u_2}$: Weights used to adjust the behaviour of the scoring.

By choosing appropriate weights $w_{m_1} \cdots w_{u_2}$, each stream S_1 and S_2 can focus on specific aspects of the histograms. For a particular input stream, assigning a large value to the weight corresponding to $f_u(H)$ will make the stream avoid underexposure. On the other hand, assigning a large value to the weight corresponding to $f_o(H)$ will make

the stream avoid overexposure. The weight corresponding to $f_m(H)$ can be used to control how the stream behaves when there is little underexposure and overexposure. An overview AE control system is shown in Figure 3.

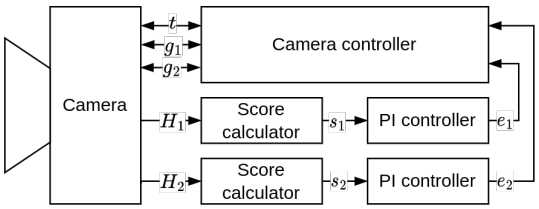

Figure 3: An overview of AE control

4 Implementation

4.1 Basic exposure fusion

When performing exposure fusion using the RGB colour space, the score for each pixel is calculated in the following way:

```
calculate_score_rgb(color, w_1, w_2) {
    mean = (color.r + color.g + color.b) / 3;
    p_1 = sqrt(
          pow(mean - c.r, 2)
        + pow(mean - c.g, 2)
        + pow(mean - c.b, 2)
    );
    p_2 = exp(-pow(mean - 0.5, 2) / 0.08);
    return pow(p_1, w_1) * pow(p_2, w_2);
}
```

Once the score has been calculated, each pair of pixels `pixel_1` and `pixel_2` from the two input frames is combined in the following way:

```
score_1 = calculate_score(pixel_1, w_1, w_2);
score_2 = calculate_score(pixel_2, w_1, w_2);

total = w1 + w2;
score_1 = total == 0 ? 0.5 : score_1 / total;
score_2 = total == 0 ? 0.5 : score_2 / total;

pixel_out = pixel_1 * score_1 + pixel_2 * score_2;
```

First, the score for each input pixel is calculated, and normalized, such that `score_1 + score_2 == 1`. When normalizing, `total == 0` must be checked, because it is possible that both input pixels have a score of 0. Once the normalized scores have been calculated, the colour of the output pixel is calculated by combining the two input pixels, based on their scores.

4.2 Improving exposure fusion

The Gaussian blurring of the scores was implemented using separable kernels. Using separable kernels reduces the runtime complexity from $O(WHN^2)$ to $O(WHN)$, where $W$and H is the width and height of the frame, and N is the size of the Gaussian kernel.

For detailed implementations of the colour conversion algorithms between RGB, HSL, HSV, and HWB, see [3, 4, 12]. The scores for the additional colour spaces were computed as follows:

HSL

```
calculate_score_hsl(color, w_1, w_2) {
    p_1 = color.s;
    p_2 = exp(-pow(color.l - 0.5, 2) / 0.08);
    return pow(p_1, w_1) * pow(p_2, w_2);
}
```

- `p_1`: The saturation of the pixel. The value is directly taken from the S component of the colour.
- `p_2`: The exposure of the pixel. Represents how close the L channel is to 0.5. It is calculated similarly to `calculate_score_rgb()`, however, the value of the L channel replaces the mean of the R, G, and B channels.

HSV

```
calculate_score_hsv(color, w_1, w_2) {
    p_1 = color.s;
    p_2 = color.v;
    return pow(p_1, w_1) * pow(p_2, w_2);
}
```

- `p_1`: The saturation of the pixel. Similarly to the HSL colour space, the value is directly taken from the S component of the colour.
- `p_2`: The exposure of the pixel. The value is directly taken from the V component of the colour, since a V value of 1.0 represents a well-exposed colour.

HWB

```
calculate_score_hwb_b(color, w_1, w_2) {
    p_1 = 1 - (color.w + color.b) / 2;
    p_2 = 1 - color.w;
    return pow(p_1, w_1) * pow(p_2, w_2);
}
```

- `p_1`: The saturation of the pixel. The value is estimated by how small the mean of the W and B components is. This was chosen because the higher the mean of the W and B components becomes, the more "gray" the colour is.
- `p_2`: The exposure of the pixel. The value is estimated using the W component.

4.3 AE control for exposure fusion

In order to compute the values of t, g_1, and g_2, the steps from Equation 7 are implemented in the following way:

```
t_target = (sqrt(e_1) + sqrt(e_2)) / 2;
t = min(t_target, t_max);
g_1 = e_1 / t;
g_2 = e_2 / t;
```

The linear feedback controller is ideally implemented using a proportional (P) controller, as it is simple to both implement and adjust. In order to avoid constant changes to the exposure time and gain, any small changes in these values (t, g_1, and g_2) are ignored. This is done because frequent changes in these values can cause flickering in the input streams. However, because of this, it is likely that there is a residual error left when using a P controller. Therefore, in this case, a proportional-integral (PI) controller is used in order to remove the residual error.

Due to occasional frames being dropped, the input to the linear feedback controller contains occasional large spikes of noise. Because of this, if a proportional–integral–derivative (PID) controller is used, the weight for the D term has to be set to a very small value. Therefore, only P and PI controllers are considered for this application.

The P controller is implemented in the following way, where `k_p` adjusts the sensitivity of the P controller:

```
p_next(target, input) {
    error = target - input;
    p_out = k_p * error;
    return p_out;
}
```

On the other hand, the PI controller is implemented in the following way, where `k_p, k_i` adjusts the sensitivity of the P controller, `integral` maintains its value across function calls, and `d_t` represents the time between frames:

```
pi_next(target, input) {
    error = target - input;

    p_out = k_p * error;

    integral += error * d_t;
    integral = clamp(
        integral, -windup_guard, windup_guard
    );
    i_out = k_i * integral;

    return p_out + i_out;
}
```

The value of `integral` is clamped between the values `-windup_guard` and `windup_guard`, in order to avoid the integral term accumulating excessively during sudden changes in brightness.

Score calculation is implemented in the following way, where `H` is a histogram, and `w_m, w_u, w_o` represent the weights for f_m, f_u, f_o, respectively:

```
calculate_score(H, w_m, w_u, w_o) {
    m = (H.median() - 127.5) / 255;
    o = H.bins(255) / H.value_count();
    u = H.bins(0) / H.value_count();

    return m * w_m - u * w_u + o * w_o;
}
```

The value of `m` is divided by 255 to guarantee that $-1 \leq m \leq 1$. In addition, the values of `o, u` are divided by `H.value_count()`, the total number of values in the histogram, so that these values are also guaranteed to be between 0 and 1. By guaranteeing that all values are between 0 and 1, the process of choosing the weights `w_m, w_o, w_u` for each input stream is simplified.

5 Results

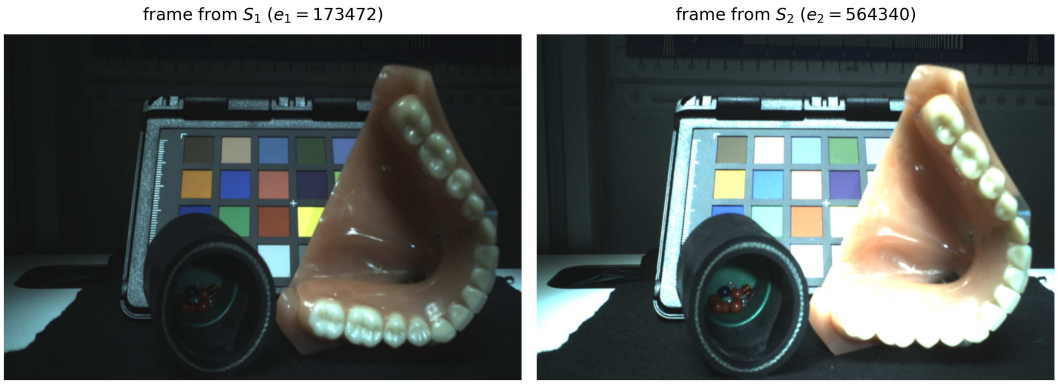

Figure 4: The two frames used for exposure fusion

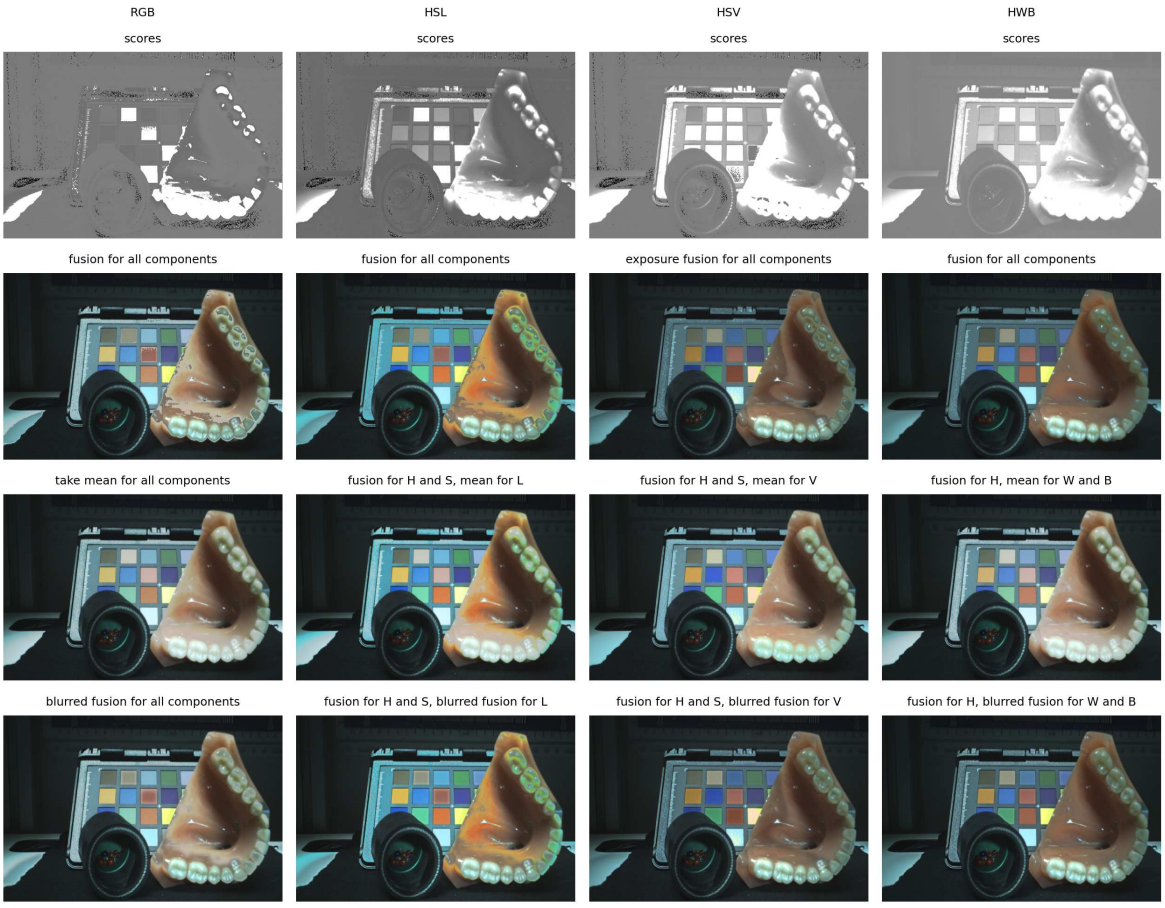

Figure 5: Results using different colour spaces and different exposure fusion methods

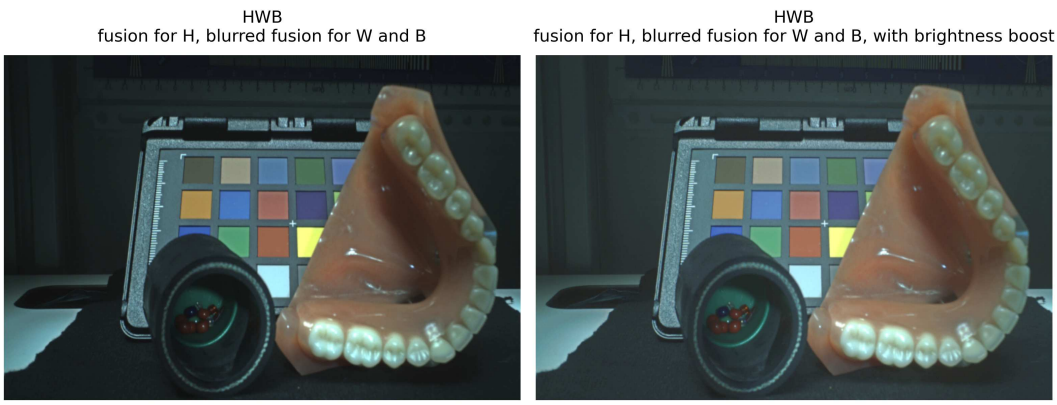

Figure 6: Result of applying brightness boost to best result in Figure 5

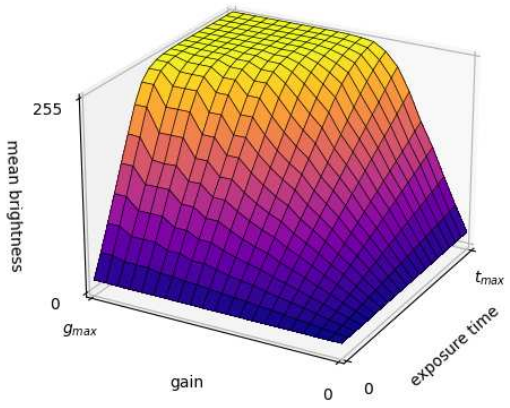

Figure 7: The brightness response of the camera at different exposure times and gains

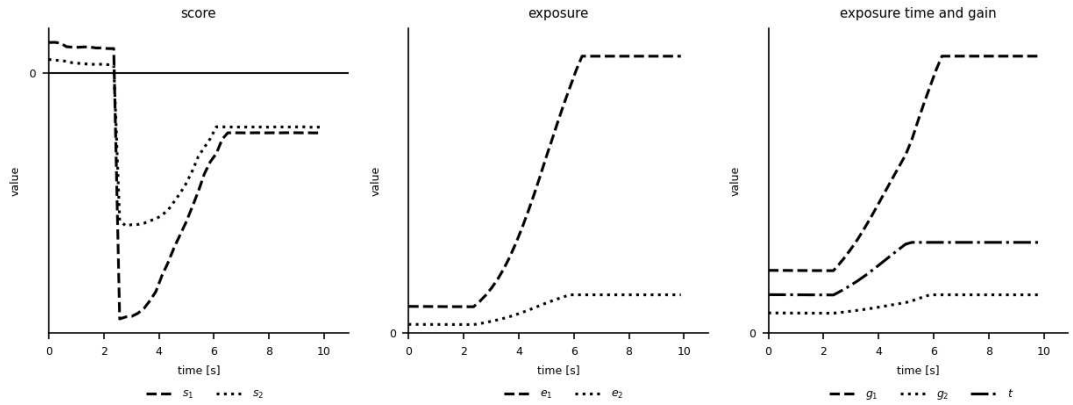

Figure 8: The behaviour of a P controller

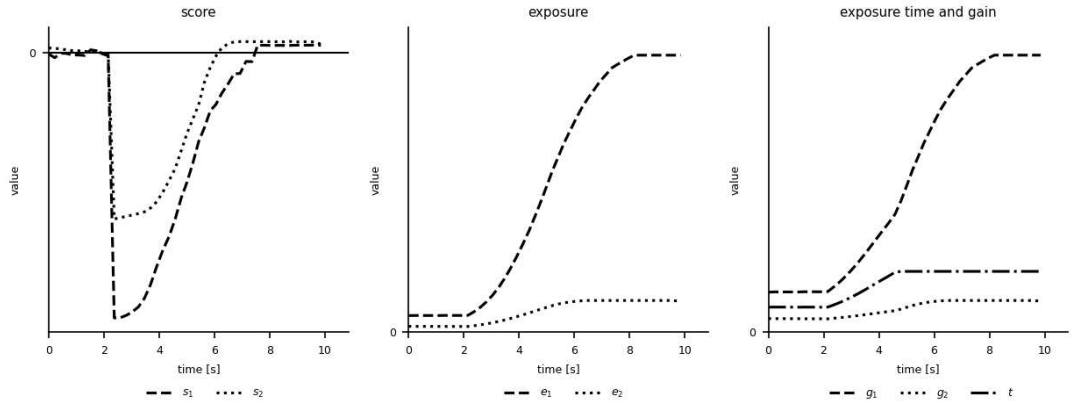

Figure 9: The behaviour of a PI controller

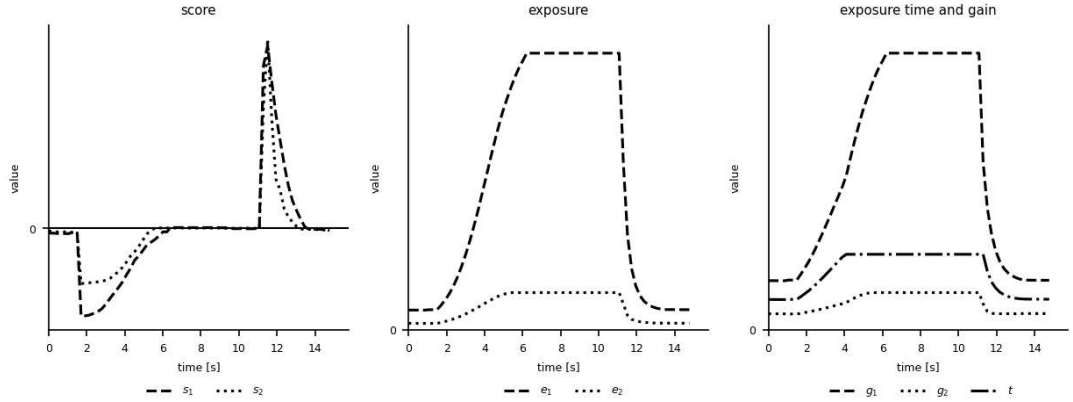

Figure 10: The response of a well-tuned PI controller to changes in brightness

6 Discussion

6.1 Improving exposure fusion

Looking at the results in Figure 5, the HWB colour space gives results with fewer seams compared to the RGB, HSL, and HSV colour spaces. Using the HWB colour space results in scores with no sudden changes. This results in an image with very few seams.

In addition, both the use of blurred scores, and taking the mean for colour components representing brightness, also reduces seams. However, when taking the mean, overexposed areas of the image remain bright, leading to a loss of detail in certain areas. On the other hand, the use of blurred scores does not lead to this issue, and, especially when using the HWB colour space, produces results with high detail.

The results can be further improved by applying brightness boost to the output, as described in my previous report. The result of this is shown in Figure 6. This further increases the brightness of the dark areas of the image, resulting in a final image, where both highly reflective teeth and dark areas inside the "tube" at the bottom-left of the image can be clearly seen.

6.2 AE control for exposure fusion

Figure 7 shows that both gain and exposure time are linearly related to the brightness of the frame. This shows that Equation 6 is valid.

Figures 8, 9, and 10 show the behaviour of the linear feedback controller. As mentioned in 3.3 above, the controller adjusts the values e_1 and e_2 (shown in the centre column), such that the scores s_1 and s_2 (shown in the left column) approach 0. From the values e_1 and e_2, the exposure time t and gain g_1 and g_2 are then calculated (shown in the right column).

Figure 8 shows a P controller responding to a decrease in brightness. Although s_1 and s_2 both return to values close to 0, a residual error remains,

as predicted in 4.3 above. On the other hand, using a PI controller as shown in Figure 9, s_1 and s_2 both return to 0, with no residual error.

Figure 10 shows a well-tuned PI controller respond to a decrease in brightness followed by an increase in brightness. In both cases, the scores s_1 and s_2 both return to 0 with no residual error. Since bright values are penalized more when calculating the scores, the response is quicker when there is an increase in brightness. Looking at the exposure time t and gain g_1 and g_2, it can be seen that the value of t remains approximately in the centre between g_1 and g_2. However, once t reaches its maximum value, g_1 alone is increased in order to match the value of e_1.

7 Conclusion

How can the seams that appear as a result of exposure fusion be reduced?

When performing exposure fusion, the scores used for the fusion process can be blurred in order to remove seams. The burred scores can be used only for colour components that represent brightness, while still using unblurred scores for colour components that represent hue or saturation, in order to preserve detail. A Gaussian filter is ideal for blurring, as it can be implemented efficiently using a separable kernel.

When performing exposure fusion, what colour space produces the best result?

The use of the HWB colour space reduces noticeable seams. The use of this colour space is also ideal, as the colour space can be used directly in other parts of the image processing pipeline. When the use of the HWB colour space is combined with the use of blurred scores in exposure fusion, the result is an image where both dark and light areas can be seen with little underexposure or overexposure.

How can AE control be implemented for use with exposure fusion?

AE control using two-frame exposure fusion can be achieved by separately controlling the two input streams using a PI controller. The PI controller removes any residual error that may remain when using a P controller. The input to the controller is a score calculated based on the histogram of the frames from the two input streams. This allows easy implementation for situation-specific exposure control.

8 Next steps

In the current implementation of AE control, the weights used to control the behaviour of the PI controller are chosen by hand. Because of this, there is no guarantee that the weights used in this report are optimal. A method to quantify the well-exposedness of an image may be able to be used to test the result of exposure fusion. The weights used during AE control can then be modified such that the well-exposedness of the output image is maximized.

When looking at bright red objects through the camera, often it is hard to see different shades of red. This is because many of the reds are being overexposed, and the AE algorithm does not correct for this, since the low values in the G and B channels make the frame appear to be well-exposed. This problem can be fixed by performing AE control on each of the R, G, and B separately, by adjusting the white balance of the camera, and later correcting for the colour offset by applying a colour correction matrix.

9 Self reflection

During this project, I have received a large amount of feedback from various parties. The feedback has helped me in many ways. For instance, through conversations with surgeons at Maastricht UMC+ and researchers at the University of Twente, I realized the importance of maintaining a "natural" look to the output image, as surgeons and dentists have trained to recognize certain colours or patterns with their eyes.

In the coming year, I expect to receive even more feedback, as once fully implemented, testing of the exposure fusion algorithm during surgery will begin. Until now, when asking for feedback, while I had a small number of questions in mind, I mostly relied on general feedback from the users and experts. However, in order to consistently obtain more feedback with higher quality, I will carefully prepare a larger set of detailed questions. In addition, I will also prepare expected answers to these questions in order to improve the quality of the questions and to help in the process of implementing the feedback. General feedback will also continue to be gathered, as this remains a good source for new ideas.

References

[1] A. Ardeshir Goshtasby. "Fusion of multi-exposure images". In: *Image and Vision Computing* 23.6 (2005), pp. 611–618. DOI: 10.1016/j.imavis.2005.02.004.

[2] T. Mertens, J. Kautz, and F. Van Reeth. "Exposure Fusion: A Simple and Practical Alternative to High Dynamic Range Photography". In: *Computer Graphics Forum* 28.1 (2009), pp. 161–171. DOI: 10.1111/j.1467-8659.2008.01171.x.

[3] George H. Joblove and Donald Greenberg. "Color spaces for computer graphics". In: *ACM SIGGRAPH Computer Graphics* 12.3 (1978), pp. 20–25. DOI: 10.1145/965139.807362.

[4] Alvy Ray Smith. "Color Gamut Transform Pairs". In: *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques*. 1978, pp. 12–19. DOI: 10.1145/800248.807361.

[5] Jarosław Bernacki. "Automatic exposure algorithms for digital photography". In: *Multimedia Tools and Applications* 79.19-20 (2020), pp. 12751–12776. DOI: 10.1007/s11042-019-08318-1.

[6] Vincent Graham and Fokko Pieter Wieringa. *Binocular device*. US Patent App. 17/635,434. 2022.

[7] Dave Shreiner et al. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. 8th. 2013. ISBN: 0321773039.

[8] Mansour Nejati et al. "Fast exposure fusion using exposedness function". In: *2017 IEEE International Conference on Image Processing (ICIP)*. Sept. 2017. DOI: 10.1109/icip.2017.8296679.

[9] Asheer Kasar Bachoo. "Real-time exposure fusion on a mobile computer". In: *20th Annual Symposium of the Pattern Recognition Association of South Africa* (2009), pp. 111–115.

[10] P. Burt and E. Adelson. "The Laplacian Pyramid as a Compact Image Code". In: *IEEE Transactions on Communications* 31.4 (Apr. 1983). DOI: 10.1109/tcom.1983.1095851.

[11] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Prentice Hall, 2008. ISBN: 9780133356724.

[12] Alvy Ray Smith and Eric Ray Lyons. "HWB—A More Intuitive Hue-Based Color Model". In: *Journal of Graphics Tools* 1.1 (1996), pp. 3–17. DOI: 10.1080/10867651.1996.10487451.

[13] Bernd Jähne. "EMVA 1288 Standard for Machine Vision". In: *Optik Photonik* 5.1 (2010), pp. 53–54. DOI: 10.1002/opph.201190082.

[14] Simon Schulz, Marcus Grimm, and Rolf-Rainer Grigat. "Using brightness histogram to perform optimum auto exposure". In: *WSEAS Trans. System and Control* 2 (2007).

[15] Haitao Yang et al. "A new automatic exposure algorithm for video cameras using luminance histogram". In: *Frontiers of Optoelectronics in China* 1.3-4 (Dec. 2008), pp. 285–291. DOI: 10.1007/s12200-008-0064-7.