*** Midterm checkpoint starts at page 11 ***

Final Project Proposal

1. Team Member: Yujia Yan(yujiay2, BS in Math + CS, May 2020), Gary Liu(jl25, BS in CS, December 2019), Yuliang Fan(yf3, BS in Stat+CS, May 2020), Yifei Yan(yifeiy5, MS in ECE, December 2020)
2. Topic: We aim to propose an algorithm that can automatically bridge the discontinuity at the boundary of high resolution and low-resolution regions in images. In this project, we target at solving this problem in the domain of static images with fixed inset region, while stretch goals include (1) applying this technique to videos and resolve incurred temporal artifacts, and (2) allow moving inset region, which might be useful for dynamic foveated rendering.
3. Motivation: Foveated rendering is an important technique in virtual reality that enables rendering frames with higher perceived resolution while introducing minimal GPU overhead. The basic idea is to render high resolution in a central subregion of the frame, and render low resolution in the periphery. This exploits the fact that human eyes can only perceive high resolution in the fovea region. However, the discontinuity of resolution at the inset boundary can create a perceivable visual artifact.
    a. Gary - I find this project interesting because foveated rendering is an imminent technology in the VR industry.
    b. Yuliang - This is project is related to computer graphics, which is the area I'm interested in.
    c. Yujia - The rendering needs to follow eye tracking.

    d. Yifei - This project really interests me because I like VR games and I believe VR/AR technology is a technology with great potential.

4. This problem is not data heavy; we can just take a few images, select part of the image, and then lower the resolution of other parts to get the data we need.

Responsibilities (Phase 1)

1. Data collection - Gary
2. Implement Gaussian filter baseline - Yuliang
3. Implement metric (gradient spike across border) and present results on un-smoothed images - Michael
4. Visualize unsmoothed image in the frequency domain - Yujia

1.Data collection

We generate the image data using a little trick: first resize the image by ¼, and then select the central ⅛ window replaced by the original image to generate the effect of blurring in VR. Therefore there’s a discontinuity at the central window of the image. Similarly in the video part.

2. Implementing Gaussian filters

Our first idea to come up with is to use gaussian filters to smooth the discontinuity at the boundaries. A Gaussian filter with low sigma can blur the image as less as possible, but reduce the discontinuity between different resolution parts at the boundaries. It looks fine in the gaussian filter area. The following is the result after applying gaussian filters with (sigma=0.5, kernel_half_size=30)

However, we can see that:

1. There's still discontinuity at the places between the gaussian filter and the below high resolution area.
2. The high resolution area near the boundary is also blurred, which shouldn't be.

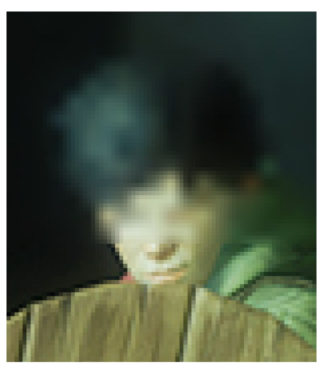
Therefore, gaussian filter isn't a feasible way to solve this problem; we need somewhat other algorithm to smooth the boundaries.

3. Implement metric

The purpose of metric implementation is to quantify how well the high-resolution and low-resolution regions are connected. Before quantification, we need to know what features/quantities are closely related to the discontinuity.

Whole image

If we look at the whole picture, it’s not very easy to notice the discontinuity. However, if we focus on the man’s face, then we can clearly see the boundary and the difference between the high-resolution and low-resolution region.



Boundary between high- and low-resolution region

Considering the way that upsampling is achieved in the low-resolution region, we think one possible feature is the gradient of the image. Because in the low-resolution region, image is blurred which means the difference between neighboring pixels is relatively small. If we convolve the image with a sobel filter, then we should obtain the intensity of gradient at each pixel. The stronger the discontinuity, the more obvious the boundary will be.

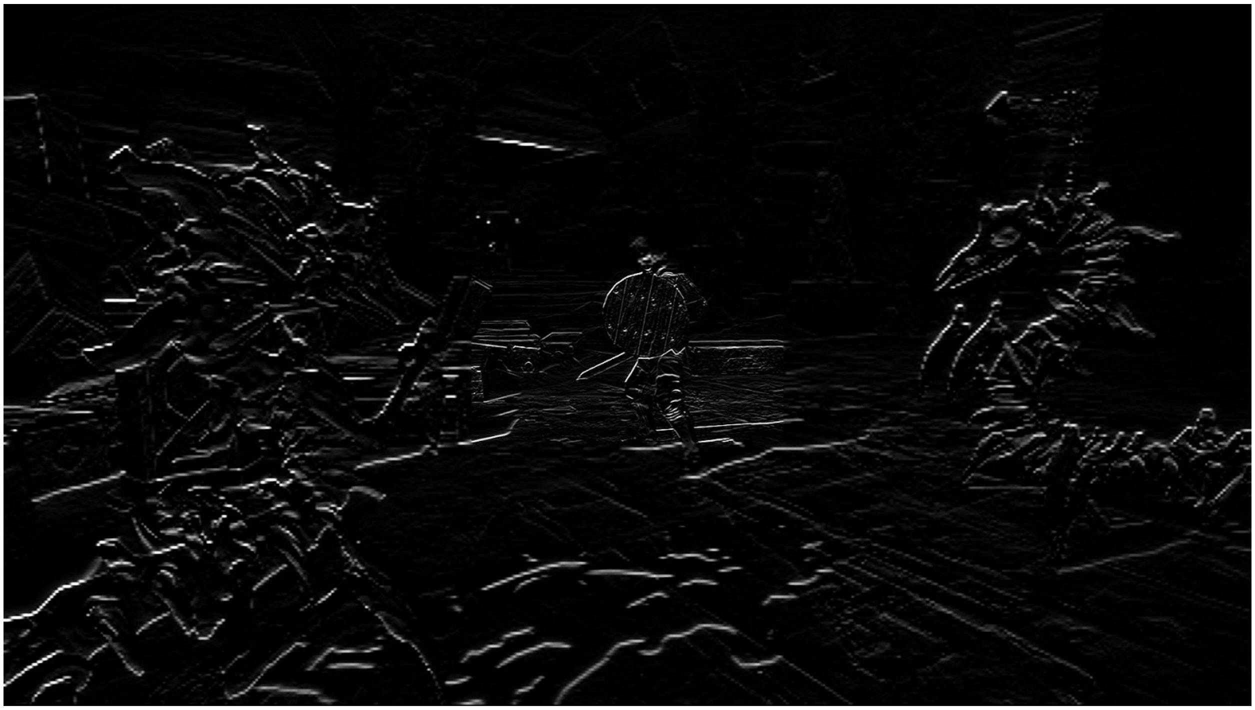
Steps

1. Convert the original image from RGB to grayscale
2. Apply sobel filter to get the gradient of the image in both x and y direction

   dx, dy = cv2.spatialGradient(im)

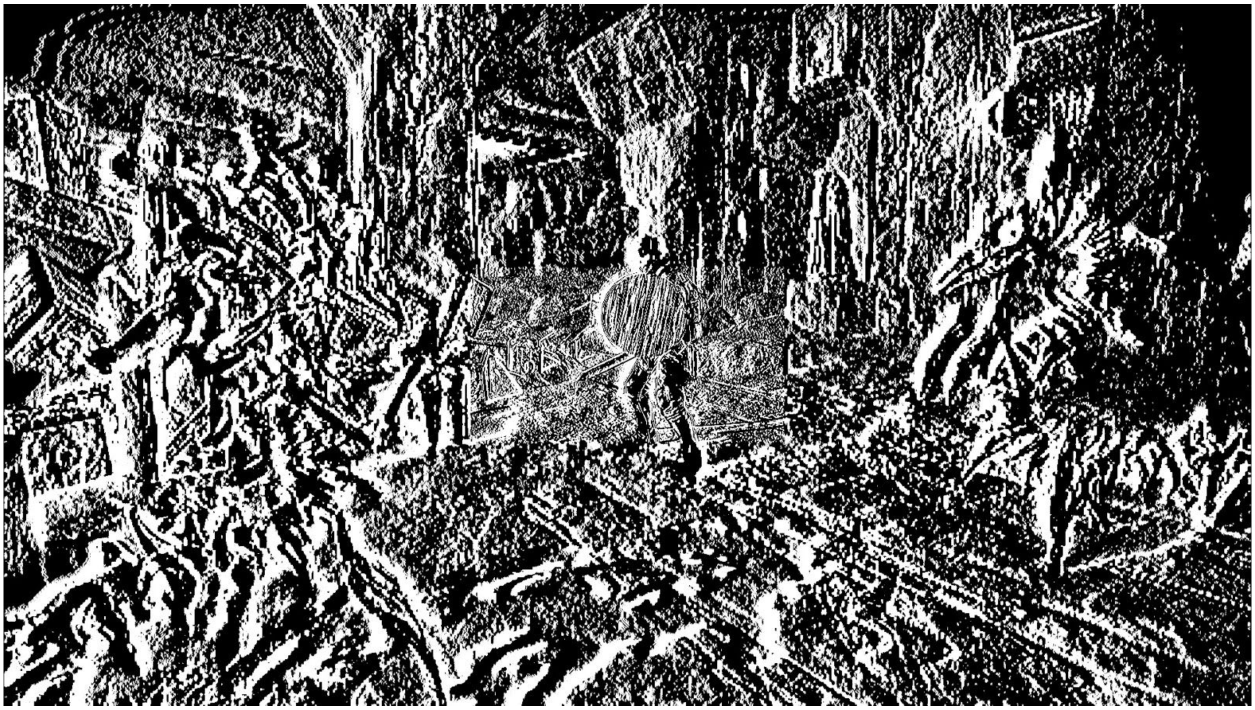
3. Multiply the image (matrix) by 100

Gradient in the x direction

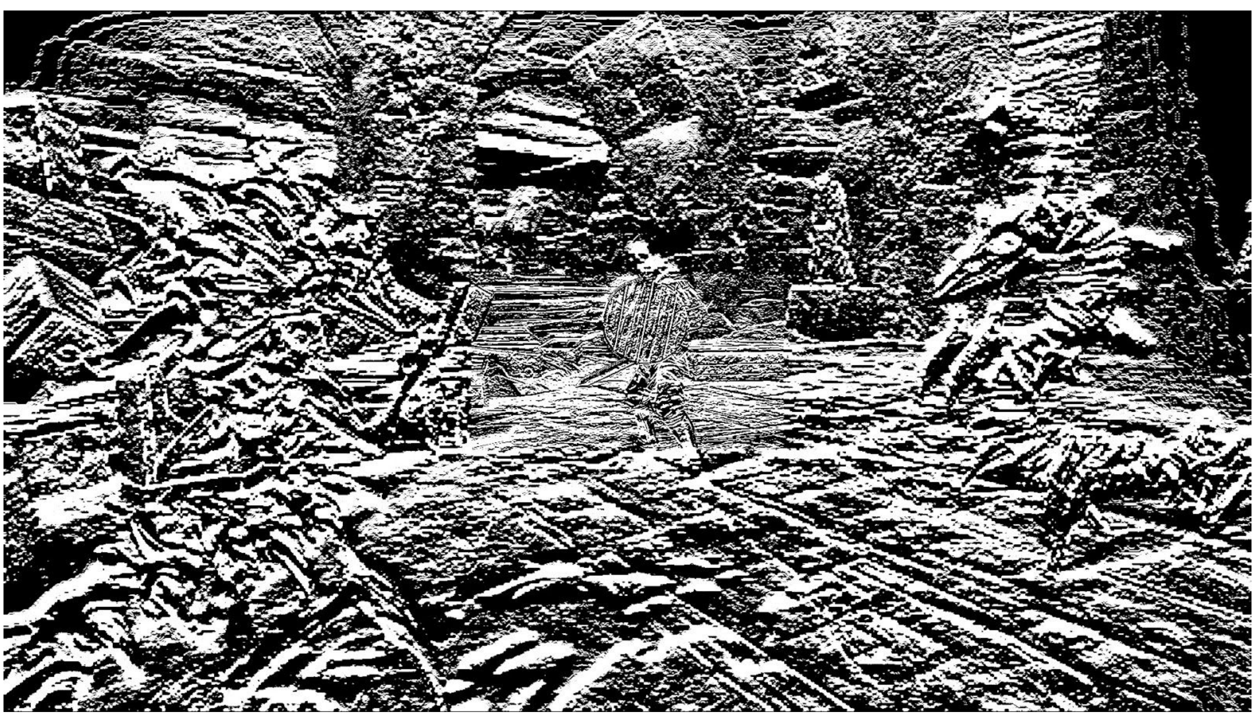


Gradient in the y direction

With the original gradient images (for both x and y direction), although we can see the edge of various objects in the picture, it’s very difficult to see the boundary that we expected. After multiplying the gradient images by 100, the boundary become much clearer.

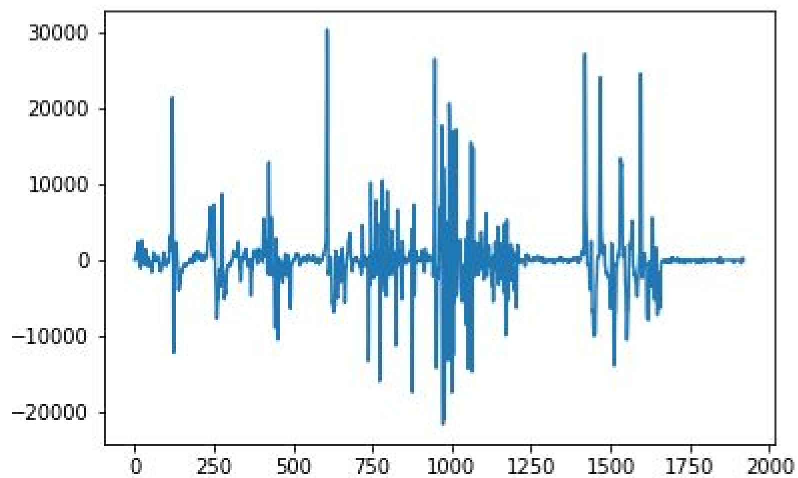


Gradient in the x direction (100X)

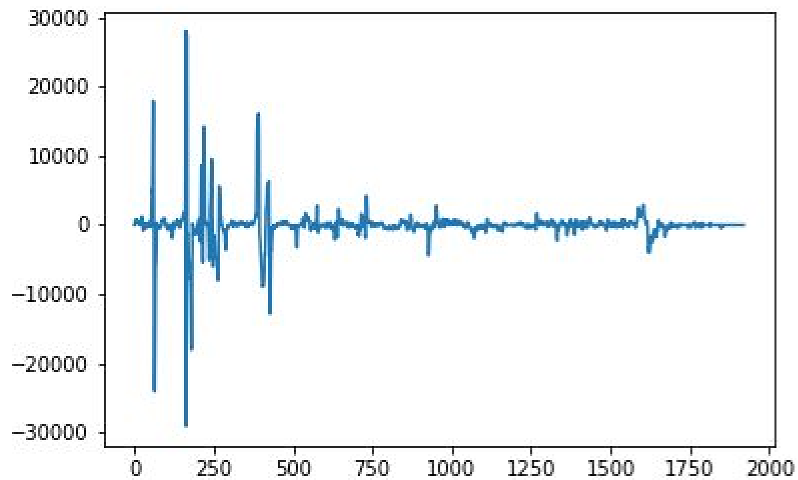


Gradient in the y direction (100X)

The size of the whole image is 1080 * 1920, and the size of the high-resolution region is 270 * 480. We can clearly see that the intensity in the high-resolution region is different from the low-resolution region. To show this difference in numbers, we take the gradient image in the x direction for example. We plot the gradient’s value of pixels on the same horizontal line in different regions.



Intensity vs. pixel number (high-resolution region)

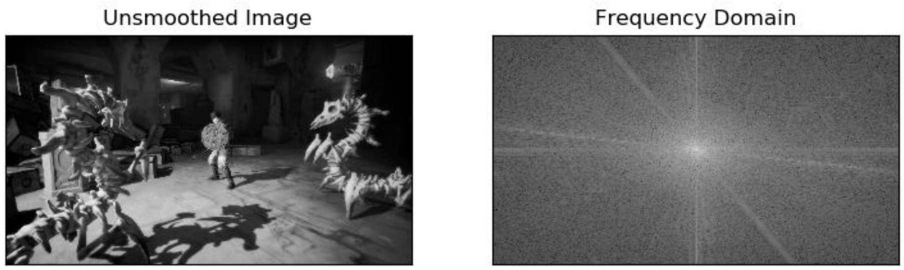


Intensity vs. pixel number (low-resolution region)

On each line, the number of pixels in high-resolution region ranges from 720 to 1200. We can see that in high-resolution region, the intensity of pixels from 720 to 1200 is very strong, while in low-resolution region, the intensity of pixels from 720 to 1200 is close to zero.

Therefore, the gradient is a feature that can be used to describe the discontinuity. For pixels within the range of [720, 1200], if the change of intensity of pixels from high-resolution region to low-resolution region is sharp, the discontinuity is strong.

4. Visualize unsmoothed image in the frequency domain



We can’t tell anything useful from the frequency domain, since the discontinuity only happens at a very small boundary with 1 pixel. Because there’s horizontal and vertical boundaries, we know that there must also be vertical/horizontal peaks at frequency domain at the same location of the image; but it’s hard to figure out from the image above.

Introduction

In virtual reality (VR) devices, rendering requires a lot of amount of graphics resources because it has to be done with high resolution and high frame rate. For example, the state-of-the-art Oculus Quest has a native rendering size of 2304x1280 \footnote{Although the Quest display size is 2880x1600, its native applications render at 0.8x display size.} and frame rate of 72Hz. In comparison, gaming PCs typically renders at around 2560x1440 and refreshes at 60Hz or 120Hz. However, with the trend to untether our VR experience, the Quest headsets are only equipped with an SoC (e.g. Qualcomm Snapdragon) and are thus not as privileged as gaming PCs with powerful GPU. Therefore, these VR devices face the challenge of quickly producing high-quality graphics within the hardware constraint.

Foveated rendering is one of the techniques that allows us to deliver high-fidelity graphics with low computation resources. It exploits the fact that human eyes are more sensitive to the central gaze region and less sensitive to the peripheral region. We can save computation by only rendering the peripheral region to a lower quality without user's notice. There are two types of foveated rendering: fixed foveated rendering (FFR) and dynamic foveated rendering (DFR). In FFR, a fixed window at the center of the frame is rendered at a higher resolution than the periphery. This is based on two observations: (1) people tend to gaze more at the center region of display (i.e. they would rotate their heads to put the attended region at the center); (2) periphery regions of the display are offset from the center, so in human eyes the periphery has a larger "effective resolution" if rendered at the same resolution as the inset. In DFR, gaze direction is estimated from eye tracking data and is used to retarget the high-res inset location.

One problem faced by foveated rendering is the discontinuity at inset boundaries. Although not very visible if viewed on a monitor, such discontinuity is rather perceptible in VR. Our work proposes and analyzes methods for smoothing such discontinuity. We experiment with both FFR and DFR to show the robustness of our methods.

Task Formulation

[TODO]

We take as input a composited frame from a 4x-rendered inset region and a normally rendered, 4x-upsampled entire frame. We assume that the inset is ¼ x ¼ the size of the entire frame. Our goal is to produce a frame that (1) mostly preserves the high-res and low-res regions, (2) smoothes out the inset boundary to make it imperceptible, and (3) does not introduce significant computation overhead.

Metric

According to our results in the detailed proposal, we think that gradient intensity is a good metric for the methods for bridging the discontinuity between the high resolution area and the low resolution area. However, in the proposal, we only showed the gradient intensity of the pixels on the same line crossing either both high and low resolution area or only low resolution area. Another problem is that the plots are not very straightforward, because we used the original value of gradient, rather than the absolute values.

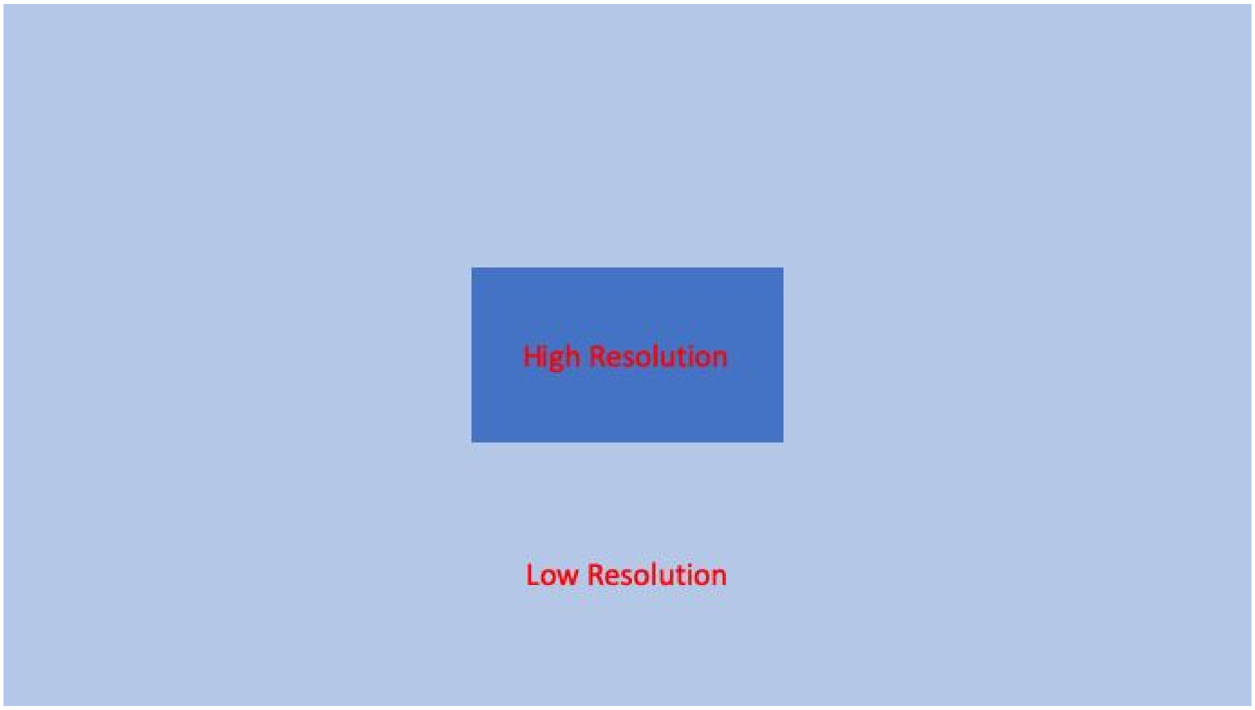


Illustration of high and low resolution area of an image

To solve the problems mentioned above and make the metric more effective and indicative, we first calculated the ratio between average gradient intensity (absolute value) of high and low resolution area of both the original image and the resampled image. After that, we further sectioned the image into four areas to characterize compared the ratio between average gradient intensity from images processed with different methods, including Laplacian Pyramid and Gaussian_Alpha blending.

We first calculated the average gradient intensity (absolute value) in both high and low resolution area. For the resampled image where high and low resolution areas exist and the boundary is clear, the ratio is 1.78 and 1.74 for gradient in x and y direction, indicating that there's a large difference between high and low resolution area in gradient intensity. Since there are different patterns and features in high and low resolution area, the ratio of average gradient intensity not only depends on the resolution, but also depends on the patterns and features in the image itself. For example, in the high resolution area, there are parts with high gradient intensity, such as the shield, while in the low resolution area, the gradient intensity of the floor and the skeletons may be lower. Therefore, we should also calculate the ratio for the original image.

Resampled image

For the original image with no high/low resolution area or the boundary, the ratio is 1.34 and 1.24 for gradient in x and y direction, respectively. It means that patterns and features in the image do play a role in the average gradient intensity because otherwise the ratio should be around 1, while the difference in resolution “increased” the ratio to around 1.7.

Original image

Our goal is to bridge the discontinuity at the boundary of the high and low resolution area. Now, we can know that compared with the original image, the ratio of average gradient intensity is higher in the resampled image, and the change of gradient intensity is sharp, so the boundary of high resolution area is very clear in the resampled image. Ideally, the change of gradient intensity should be smoother across the boundary, and we are not able to characterize the smoothness of the change only with the ratio of the average gradient intensity of the high and low resolution area.

Since we used various approaches to process the image to make the boundary less obvious, and some filter is applied to the boundary, it should be useful to section the image into more areas, as shown in the image below.

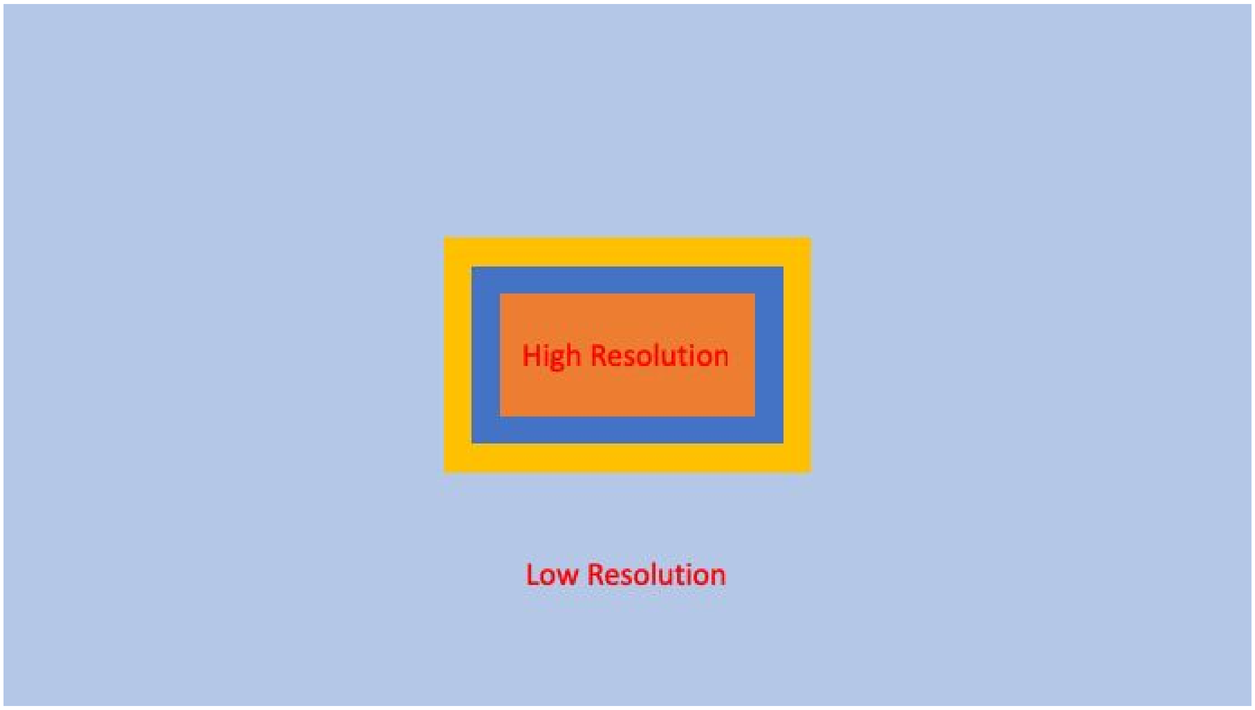
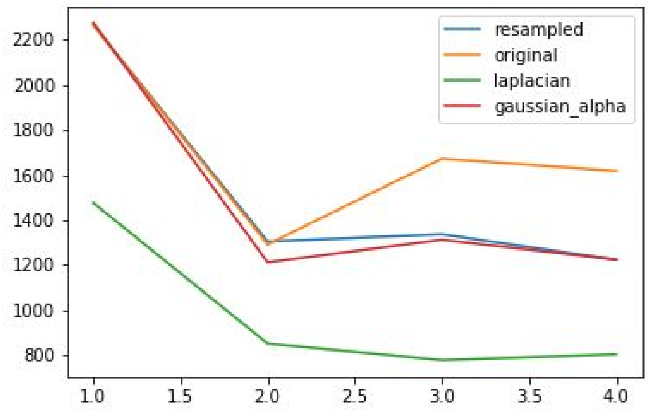


Illustration of the section of the images

We calculated the average gradient intensity (x direction) of four areas in various images and plot them in the figure below (Comparison of gradient intensity distribution). The curves include the original image (original), the resampled image (resampled), the image applied with Laplacian pyramid blending (laplacian), and the image applied with Gaussian filter and alpha blending (gaussian_alpha). In this plot, 1.0, 2.0, 3.0, and 4.0 on the horizontal axis corresponds to area from the center of the images to the outermost ring. In the resampled image, the high resolution area includes the rectangle at the center and the inner ring, while the low resolution area includes the two rings outside.



Comparison of gradient intensity distribution

We can compare different curves with the curve of the original image. For the curve of the resampled image, the gradient intensity in the high resolution area is the same as the original image, while the gradient intensity in the low resolution area is much lower. In the resampled image, the boundary is very clear. In the resampled image applied with Gaussian filter and alpha blending, because the filters are only applied to the boundary of high resolution area, the gradient intensity at 2.0 and 3.0 are slightly lower than the resampled image, while the gradient intensity at 1.0 and 4.0 are the same as the resampled image. If we zoom in and look at the boundary, we can see that the boundary is much less obvious than the resampled image, which means that the Gaussian filter and alpha blending is effective. For the image applied with Laplacian pyramid, the boundary is also much less clear. However, we can see that in the gradient intensity curve, the intensity is much lower in all areas in this image, which means that the resolution is much lower in the whole image. This is consistent with the image (Laplacian pyramid) shown below.

According to our results, Gaussian filter combined with alpha blending can yield the best result, smoothing the boundary while maintaining the high resolution.

Gaussian_alpha

Laplacian pyramid

From the current results, we can know that gradient intensity is a good metric to evaluate the methods for smoothing the boundary between high and low resolution area, and it's useful to section the image into several areas in the shape of rings to calculate the average gradient intensity. For future steps, we think it may be meaningful to further section the image into more parts (rings) so that we can have a closer look at the distribution of gradient intensity of the image. We should also apply the methods to some other images to test and prove the effectiveness of these methods.

Methods

1. Naive Alpha Blending

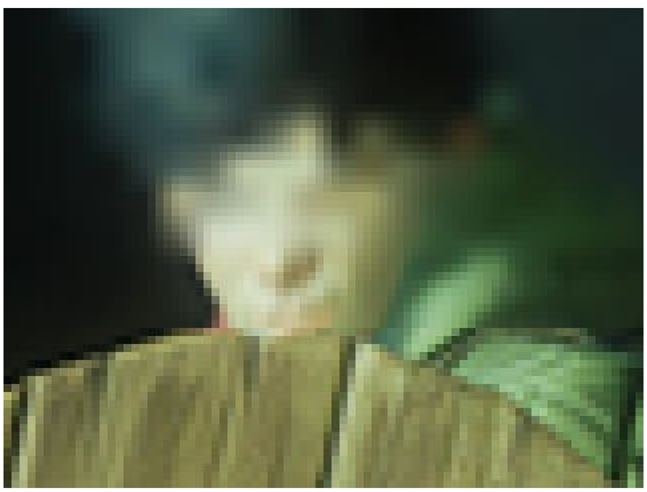
A simple method we propose to smooth out the inset boundary is to do alpha blending in the inner side of the boundary. The intuition is that we want a gradual transition from the low-resolution region to high-resolution region. For a boundary of the inset, we replace each pixel within an inner (high-resolution) strip of that boundary by a linear combination of the corresponding pixels from the low-resolution frame and the high-resolution frame. The combination coefficient depends on the pixel location, where the weight of the low-resolution pixel value decays as the pixel goes further from the boundary. We use a linear kernel for alpha blending shown in Figure \ref{fig:alpha-kernel}, and use a strip width $S=8$.

Figure \ref{fig:im_FF_alpha} shows a sample result of the naive alpha blending method. Compared to the original fixed foveation frame, the horizontal boundary over the avatar's nose is not visible, while no significant artifact is introduced.

The naive alpha blending method is expected to run very fast in the graphics pipeline. However, one drawback of this method is that it has to know the entire low-resolution frame (because it has to know the low-resolution pixels in the inset region). Therefore, this method can only be placed rather early in the pipeline (i.e. before the high-res inset and the low-res periphery are composited together).

2. Gaussian Smoothing

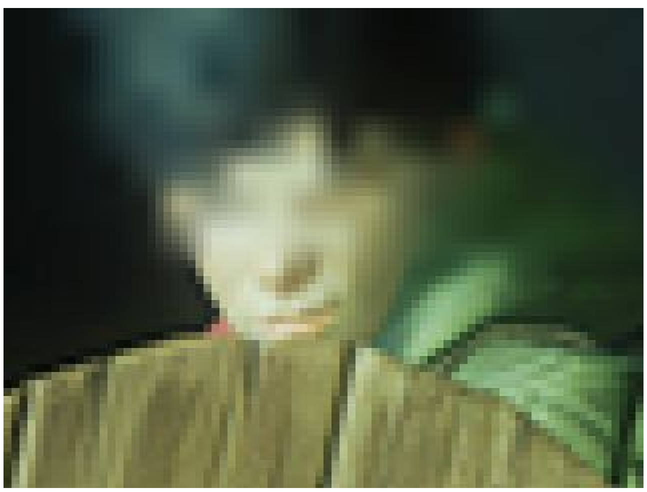
The inset boundary discontinuity can also be viewed as a high-frequency component of the gradients in the direction orthogonal to the boundary. We therefore propose another method, Gaussian smoothing, to address the discontinuity problem. For a boundary of the inset, we take a strip along the boundary (including pixels on both sides), and convolve it with a Gaussian kernel.



We experimented with 2D Gaussian kernels and 1D Gaussian kernels orthogonal to the boundary. We choose $\sigma=1.0$, a kernel size of 9 and a strip width of 16. Figure \ref{fig:im_FF_gaussian} shows the result of the Gaussian smoothing method. As we can see, applying Gaussian kernels to the strip introduces significant discontinuity artifact on the strip boundaries, which is not desired. Meanwhile, the strip applied with 1D Gaussian looks sharper than the one applied with 2D Gaussian.

3. Combining Gaussian Smoothing and Alpha Blending

[TODO]



4. Blending using Laplacian Pyramid

We tried to apply Laplacian Pyramid directly to smooth the gap of resolution discontinuity. First we get the high resolution image, then we down sample the high resolution image by a factor of 4 and then upsample by a factor of 4 to get the low resolution. Now we apply Laplacian pyramid on the original image and the sampled image, and reconstruct the new image. However, we found the result is not as good as the alpha blending method. The effects are as follows:

The first image is the resized image(as the low resolution image), the second is the result of using alpha blending, the third one is the result of using laplacian pyramid with five step steps.

There are a few issues in the Laplacian Pyramid blending:

1. First, Laplacian pyramid changes the frequency of the high resolution area to low resolution during the reconstruction step. Therefore, the high resolution image at the center will be not as clear as the original image.
2. Similar reason as the low resolution part; due to the reconstruction approximation, the resolution will be even lower
3. The image looks brighter than the one using alpha blending. We think that this is due to the addition of the low-pass result and the high-pass result. The summation is slightly larger than the original image.