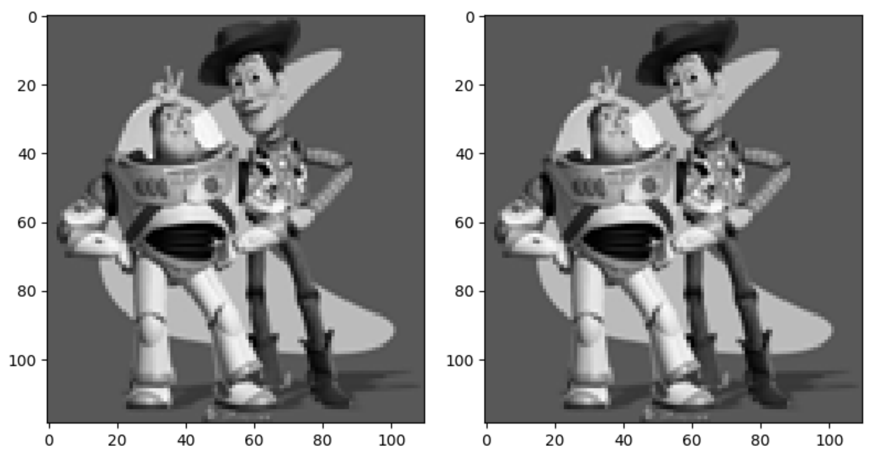
Name (netid): John Small (jssmall3@illinois.edu)

CS 445 - Project 3: Gradient Domain Fusion

Complete the claimed points and sections below.

Total Points Claimed	**[120] / 160**
Core	
1. Toy Problem	[20] / 20
2. Poisson blending	[50] / 50
3. Mixed gradients	[20] / 20
4. Quality of results / report	[10] / 10
B&W	
5. Color2Gray	[20] / 20
6. Laplacian Pyramid Blending	[0] / 20
7. More gradient domain processing	[0] / 20

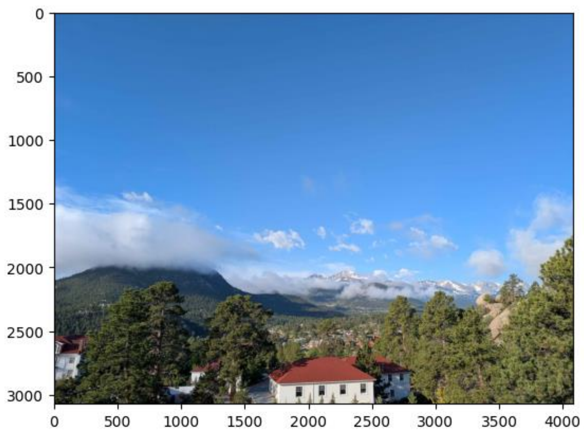
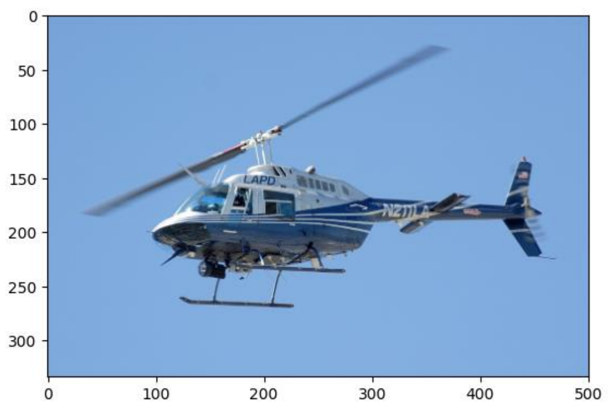
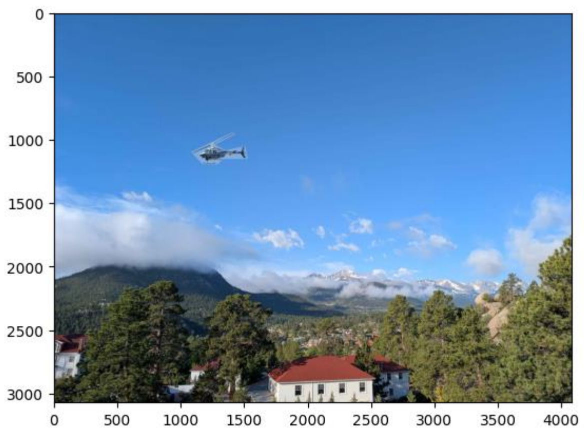
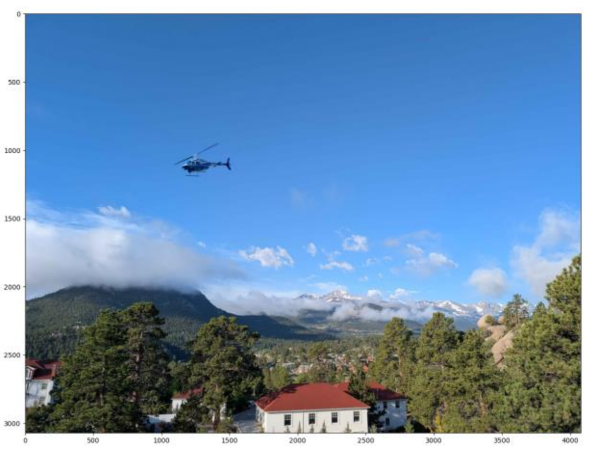
1. Toy problem



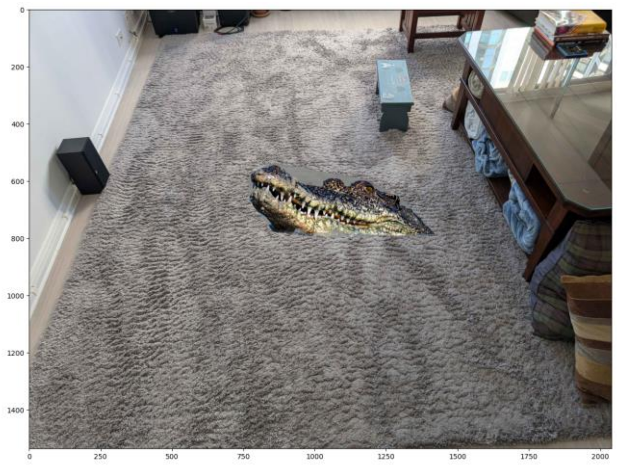
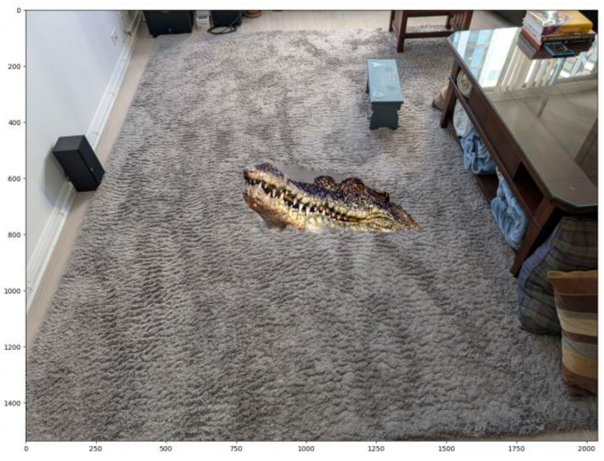
For this problem, I reconstructed the input image (left) from its gradients given a single anchoring pixel. For each pair of pixels moving both horizontally and vertically, I identified the gradients (i.e. change in pixel intensity) between them, then added these gradients to a sparse matrix along with a single anchoring pixel of identical value to the original matrix. The anchoring pixel provides a constraint on the relative intensity between the input and output images – without it, we could find many valid solutions where the gradients are the same, but the image is much lighter or darker.

Solving the system of linear equations for every pixel in the image provides the output seen above, on the right. I was able to achieve a maximum error of 6.7332e-05 when solving with a tolerance of 1e-7.

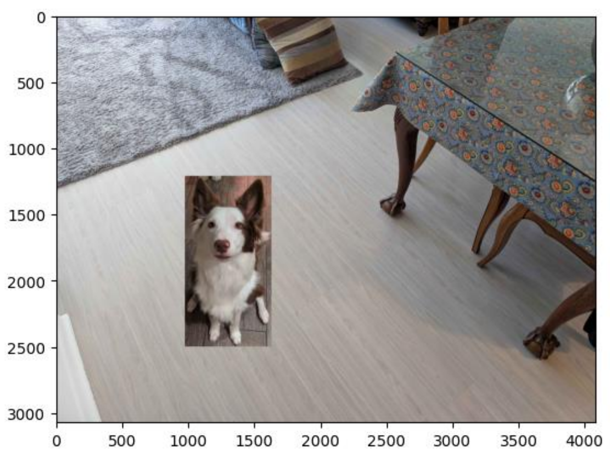
2. Poisson blending

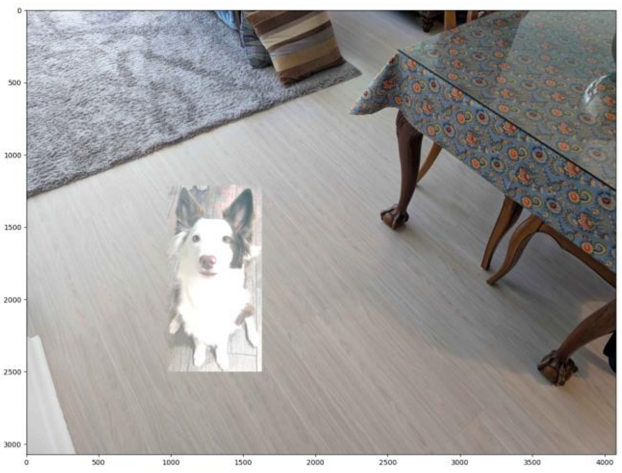





In this first set of images, I aimed to put an LAPD helicopter into a photo from Estes Park, CO. This is a very favorable condition for gradient domain fusion since the backgrounds have similar color and texture. With a fairly rough bounding mask I was able to achieve an excellent result.

I attempted a more difficult fusion with this second image, since despite the similar coloring of the background image and the foreground image's background, the textures are very different. As a result, areas where the bounding mask is not cut tight to the image use the foreground's much smoother water texture, which creates some clear demarcations between the foreground and background pixels. Meanwhile, the bottom edge of the crocodile is quite well blended and the much tighter bounds allow for a more seamless transition. I revisit this image in the mixed gradients section.

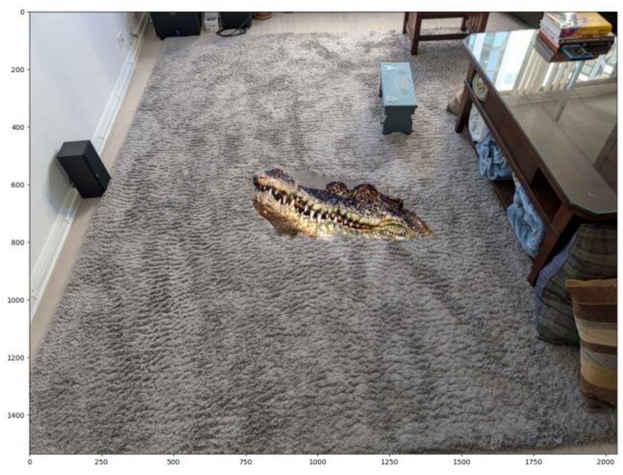
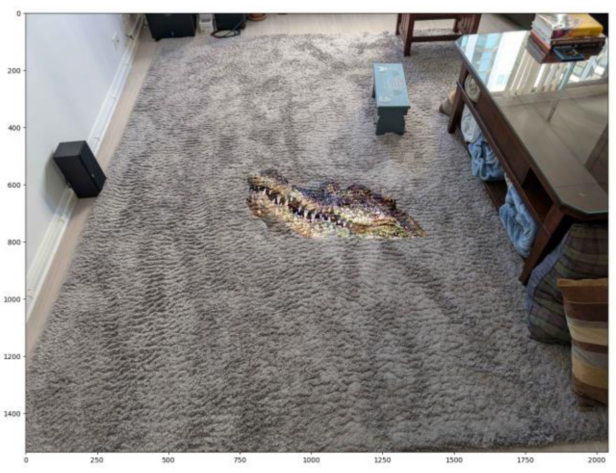




This final set of images is a clear failure. The foreground image has a dark background with a strong, well-defined floorboard texture. As the GDF attempts to solve for the edge gradients, it significantly brightens the intensity of the entire foreground image, blowing it out. Additionally, the image is cropped only to a rectangular bounding box, leaving significant amounts of the foreground's background for the solver to deal with.

Tighter bounding masks helped with earlier images (the helicopter benefited from a non-rectangular mask as it helped hide the edges better). However, with an object background of such incompatible color, and a furry edge texture that cannot be tightly cut, the color mistmatch is likely unreconcilable. This photo also includes a visible shadow to the left of the dog – we lose this shadow information and cannot easily recover it.

3. Mixed gradients

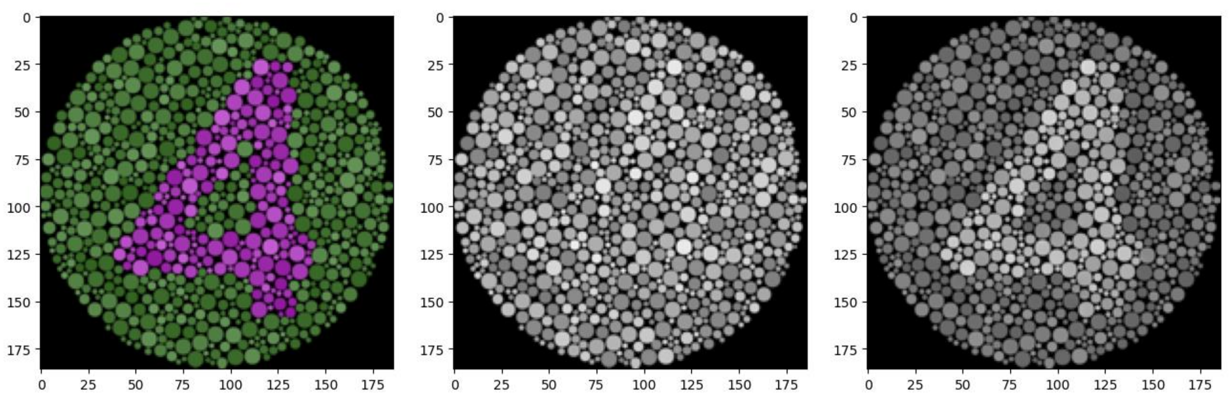



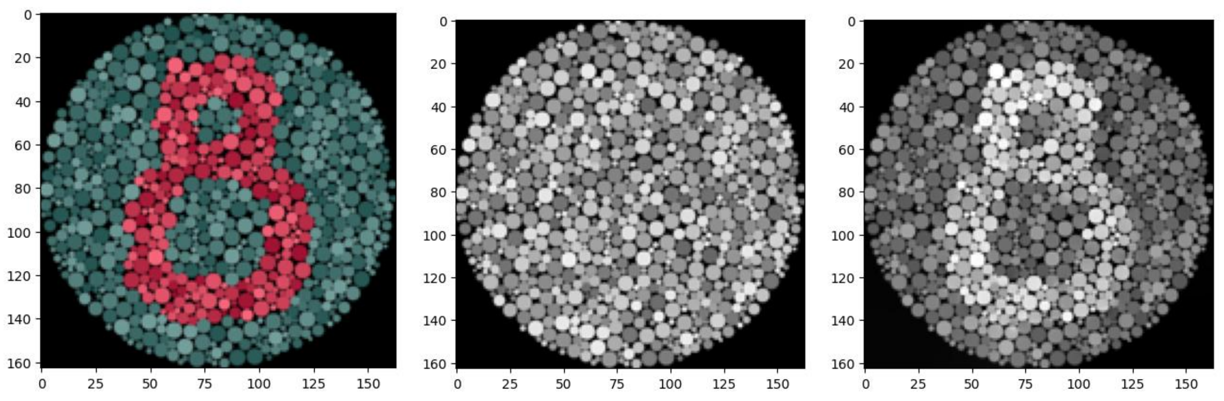
This is the same crocodile image from earlier. At left, we see the image created via gradient domain fusion, and at right we see the same using mixed gradients. I felt this was an interesting example because it demonstrates the drawbacks of either method. The mixed gradient image does a significantly better job of dealing with the foreground's background (the water at the top of the crocodile), blending in the stronger gradients from the background image. However, because the carpet's texture has such strong gradients, we also see it bleeding into the crocodile itself, giving it a somewhat translucent look on closer inspection.

4. Quality of results / report

Nothing extra to include (scoring: 0=poor 5=average 10=great).

5. Color2Gray (B&W)

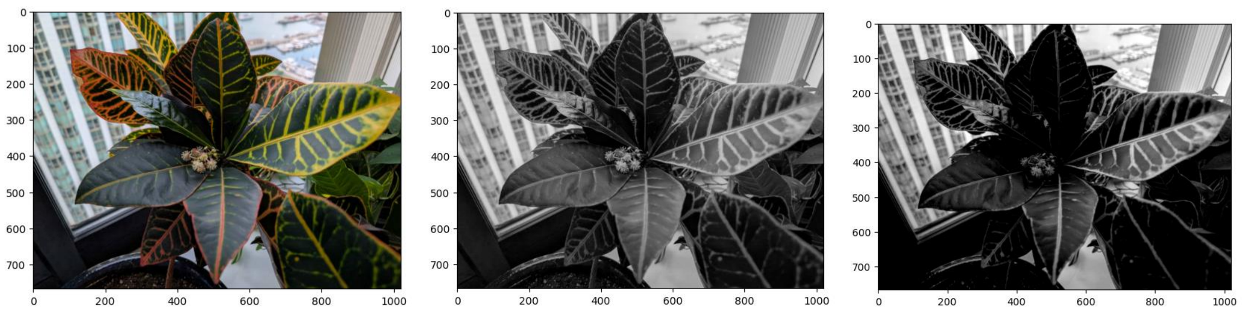




For each of the image sets above, the left image is the input, the center is the output of cv2’s cvtColor(…, cv2.rgb2gray), and the right image is the output of my gradient function. To obtain this image, I modified the gradient solver to pick the strongest gradient for each pixel pair among the three channels. Like the toy problem, the matrix is constrained at the (0,0) pixel to match the average intensity across all three color channels. Once the matrix is solved, the resulting image is broadcast across all three channels to obtain a grayscale output.

Because boundaries between different colors have large absolute gradients in the individual color channels, they translate to intensity differences in the function output, which preserves the information encoded in the original image, at the cost of assigning a higher or lower intensity to certain colors; in the four image, the green color channel is darkened and the red and blue brightened to maintain the intensity gradients.

In the real-world example, below, this enforcement of the gradients comes at the expense of the detail of the image. As we learned in the early lessons, the bulk of the information (at least as is relevant to human vision) is in the intensity (luminance) of the image, and far less in the chroma channels. By preferring chroma gradients over luminance gradients, we ultimately get a poorer image when complexity is high.



6. Laplacian Pyramid Blending (B&W)
Not completed.

7. More gradient domain processing applications (B&W)

Not completed.

Acknowledgments / Attribution

- The helicopter image is from Wikimedia Commons: https://en.wikipedia.org/wiki/File:LAPD_Bell_206_Jetranger.jpg
- The crocodile image is from the New York Times, credited to Michael Greenfelder/Alamy: https://www.nytimes.com/2023/08/24/science/crocodile-dictionary-vocalizations.html
- All other images are mine.