ECE 178 Final Project - Reproduction of Patchmatch: A Randomized Correspondence Algorithm

ERIC BUCKLAND

1 INTRODUCTION

Patchmatch from Barnes et al. introduces the concept of hole filling and image retargeting with patched based synthesis, including the influence of constraints. The focus of this project is the application of a rewritten and original patch match algorithm to reproduce hole filling and image retargeting. Re-implementation is challenging, as many considerations need to be made which have broad effects on the output image. Additionally, the consideration of many adjustable constants requires careful planning during testing. My image results are similar to the original paper, accurate to original images, and could be improved with further changes.

2 IMPLEMENTATION CHALLENGES

Full recreation proves to be challenging programmatically and conceptually. Thinking first in the map space of the NNF (Nearest Neighbor Field), then on a larger scale with the addition of patches and voting, then the application of these for hole filling, image properties, and weighting. For quite a few days, all my mental energy went into solving the various problems the implementation poses. In addition, hours could be spent adjusting minute details to improve results in some images, and cause reduction of quality in others. It is evident that results are highly variable based on the input image and intended application, which is why Patchmatch implemented constraints for user direction.

2.1 Hole Filling

Implementing patch weighting with the alpha channel in mind is the most challenging aspect of hole fill. The alpha channel could be used in numerous ways within the patch matching algorithm, but I opt to leave this part somewhat untouched in favor of drastically changing the voting scheme.

In the voting scheme, I implement color distance weighting, nearest neighbor weighting, and physical pixel distance weighting. These implementations not only need to be created within the existing splatting system of voting, they also need to be created with the consideration of the alpha channel.

To solve this problem, assuming each patch is to be splatted, I first consider the simple case of distance. The third NNF channel already holds the distance, so this can be imported directly into the voting scheme. After breaking down the intended program flow, it is evident that there should first be the multiplication of the source patch output by this color distance. This weight then needs to be added to a matrix of equivalent size to the output, and the final pixel would have to be normalized by this count. However, this count is affected by the alpha channel, as invisible pixels are not true pixels. Therefore, pixel count must be dependent on an alpha sum, and the final distances must be normalized by alpha. Distance values are also incoming as reverse to the preferred orientation, where low distances are best. Thus, I opt to normalize the initial color distances by their percentage of the total weight, then invert these percentages. This basic formula of finding the correct weight direction, multiplying the source patch, adding to a matrix count, normalizing by alpha, then normalizing the final image can also be used for neighbor count and physical distance.

Extensive testing of the hole filling algorithm could not have been completed without an idea to vastly improve runtime; process only the hole itself as the target image, rather than the entire image. This poses a significant challenge for implementation, as it requires precise positioning and scaling of the target view. After some problem solving, this can be done simply by changing the target image before and after patch matching and voting. Rather than storing the target image as a cropped section of the source image and scaling appropriately, all that is required is a matrix crop at the time of patch match calculation, then a pad of zeros after. It is possible that this adversely affects the results from lack of information at hole edges during the propagation stage of patch matching. However, I minimize this issue by adding a buffer space to the matrix crop.

2.2 Image Retargeting

Image retargeting is conceptually less complex and more straightforward to implement than hole filling. The main issues that come with retargeting are artifacts from color variation and image resizing. This is discussed in further detail in the following section.

3 RECONSTRUCTION MODIFICATIONS

3.1 Constants

There are numerous constants in hole filling and image retargeting which have drastic effects on output. For hole fill, the most significant constant is the image resize count. Having

too few scaling images results in a blurred and incoherent image due to the patch matching algorithm not recognizing details on a greater distance upscale. Increasing this value usually yields a better result, however, there is a limit of optimum value per image. If this value is too great, the patch match algorithm will run many times on similar images. If this algorithm contains a preference for nearest neighbor pixels, many sub-structures of the same part of an image will appear repetitively in some areas. Therefore, it is beneficial to test this value per image due to the range of best values.

For image retargeting, the image resize count has a similarly great effect, but the iterations and constants of scaling on the coarsest scale also can result in dramatic differences. The image size on the coarsest scale should be correlated positively with image size and image detail. However, this is highly variable. Resizing iterations on the coarsest scale must be done slowly, as distortion is rapid and only amplified during resolution refinement. Therefore, iteration count does not generally suffer from being higher. In addition, runtime is hardly increased due to the image being small.

The speed at which the image is being resized throughout the scale is also important. Logarithmic scaling with more images towards the target, more scaling towards the source, and a linear scaling trend are all implemented for testing. A comparison of these outputs is shown in results.

Through testing, for all constants, there is no absolute solution for a constant based on scale or initial image properties. Rather, yielding best results requires testing values per image and gauging properties at the coarsest scale.

3.2 RGB Versus LAB

After implementing image retargeting both with RGB and LAB, a very noticeable difference in output is apparent, with the latter having less artifacts and a preferable picture. This is due to LAB having greater calculated color distances in correlation with greater perceptual differences. Because of the noticeable positive effect, I implemented LAB into hole filling to compare results.

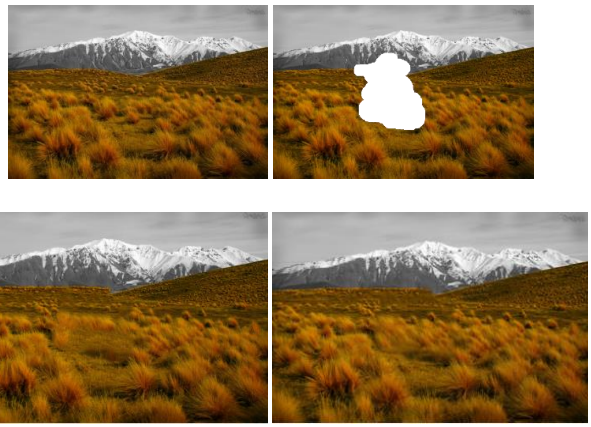
3.3 Voting Weights

To test the ultimate effect of voting weights on images, I added both color distance and a simple nearest neighbor as bimodal weighting. Both weights help significantly to form the image. To go a step further, I implemented the weighting of physical pixel distances to pixels. This would mean that patches prefer their original location. My predicted outcome for this would be that pixels on the edges of hole fill would prefer the patches adjacent in the source image. Compared to non-weighted results, this weight helps the pixels inside the holes avoid taking unwanted values from source edge pixels. However, there is insignificant effect on results if used trimodally with the other weightings. Despite this, correct implementation would have almost zero effect on runtime if physical distances are calculated during the patch match algorithm and stored in the NNF.

4 RESULTS AND DISCUSSION

4.1 Hole Filling

The primary image tested with hole filling is the following landscape. On the left is the original image, the hole image on the right, then the results.



The left image is the result in RGB space, and the right in LAB space. As shown, the LAB is a much more accurate result, with less edge blurring and artifacts. The bottom image results from adding physical distances to the voting weight, which is completely undesirable.

Considering the volume space the sign occupies, patch filling is well done here. The chain link fence retained most of its connections over the lost space. This is most likely due to the weighting of the nearest neighbor check.

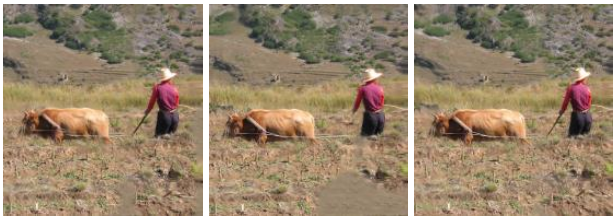
This image shows very minimal ghosting despite being in a gradient environment, with a slight artifact on the shore break.

4.2 Image Retargeting

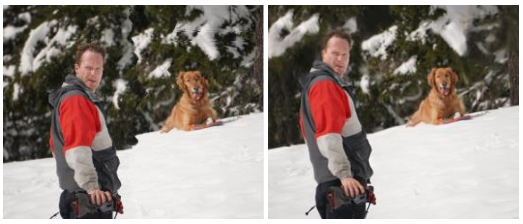
Results for retargeting are very rewarding. See the following comparison with the top being the original, the example on the left, and my implementation on the right.

The blurring in the foreground is less than the example, and the shrubbery in the background has retained more color and structure.

The following is a comparison of logarithmic image scaling. The first image utilizes more scaling toward source, second image is linear scaling, and third scales closer toward the target. Closer scaling towards the target yields surprisingly better results, with less burring and distortion.



The following shows a comparison with the seam carving algorithm, from top to bottom right; original, seam carving, patch match.



The most noticeable difference is the smoothness over the snow transition. In patch match, the snow retains connection over the slope. There is also a visible aliasing to seam carving where the image has been cut. Patch match introduces distortion, but this retarget is a succinct representation of the original image.

The following is a drastic retarget of a panoramic island picture to a square aspect ratio. In order, the original, the naïve image squish, and the output.

The algorithm has seemingly combined the left side of the tree on the far left and the right side of the tree on the far right. The reflections on the water have retained integrity and representativeness despite the change above ground.

Admittedly, I spent an exorbitant about of time running my retarget on miscellaneous images, as I find this algorithm creates interesting results almost every time. See the following city landscape, which is forced into such a thin aspect ratio that it created brutalist architecture.

4.3 Discussion

The results of the original implementation of the patch match algorithm into hole filling and image retargeting exceed expectations. With appropriate constants applied, the algorithm results represent the intended output for their respective purpose with minimal artifacts and blurring. I am very satisfied with my re-implementation.

5 FURTHER IMPROVEMENTS

Two changes could be made to potentially improve the output of hole filling and image retargeting. The most significant change would be the adjustment of the nearest neighbor weighting check to include all pixels within the local window rather than the four adjacent. However, this would drastically increase runtime, as it eliminates the benefits of the local splatting system. The other adjustment is the addition of weighting both RGB and LAB color distances. As these color distances yield such substantially different results, it may be beneficial to implement both as a measurement of true pixel distance. This would also increase runtime, as a local sum must occur for both color spaces.

6 REFERENCES

Barnes, C., Shechtman, E., Finkelstein, A. & Goldman, D. B. Patchmatch. ACM Transactions on Graphics 28, 1–11 (2009).