Efficient Poisson Blending for Seamless Image Stitching

Ahsan Abdullah, Zuha Agha
Lahore University of Management Sciences (LUMS)

Abstract – *Image blending is an extensively studied phenomenon and producing seamlessly blended image composites has found many applications in the field of image processing. Poisson blending, introduced in [1], is one of the leading approaches for seamless blending and many people have built upon it and have come up with better and efficient solutions. In our project we worked on [2] which is an extension of [1] and focuses on an efficient implementation of the Poisson linear system using quad-trees. The paper focuses on producing smoothly blended panoramas and its key to efficiency is representation of unknowns in terms of a quad-tree that is maximally subdivided along the seams. Our key contribution here is the extension of previous implementation to panoramas, and producing seamlessly blended panoramas even without user interaction. We sorted out the missing links between both the papers and ended up with a better understanding of the new realm of Poisson image blending. The algorithmic pipeline of [2] was solved up till the construction of quad-tree using the criteria of sub-division in case of non-zero internal laplacian.*

I. BACKGROUND

Blending has been a subject of wide interest in Digital Image Processing and over the years several blending techniques have been introduced to produce a perfectly smooth transition between different images.

The most simple and fastest of all would be alpha blending whereby a weighted combination of the two images is used to create a composite image. However, the main disadvantage of alpha blending is that moving objects cause ghosting and small registration errors can cause blurring of high frequency details.

In order to solve this problem, Burt and Adelson presented a multi-band blending algorithm in [6] which allows low frequencies to blend over a large spatial range, and high frequencies over a short spatial range.

Another addition to the family of blending techniques came with Laplacian Blending. Laplacian Blending involves building a difference of Gaussians (DoG) pyramid for the source and target, such that each level contains the frequencies not captured at coarser levels of the pyramid. Given a Laplacian pyramid, the original image is rebuilt by scaling up lower levels and adding in the Laplacian at each step. This causes the lower frequencies to blend while higher frequencies are preserved.

Recently, gradient domain image blending approaches [1,2,3,4,5] have been applied to image stitching and editing. A new gradient vector field is created with the gradients of source images so that a composite image can be recovered from this gradient vector field by solving a Poisson equation. This type of gradient-domain compositing algorithm can adjust the color differences due to illumination changes and variations in camera gains for the composite image globally. Likewise, it can produce high quality composite images. However, it comes with a tradeoff of high memory and computational costs.

Figure 1 shows the apple/orange image blended using Poisson and Laplacian Blending. One difference of Laplacian blending is that frequencies only blend as much as the mask is blurred or interpolated at a given level. So blending is usually localized near the mask

boundaries and varies with the kernel size of the Gaussian. This is not necessarily true for Poisson blending, where masked pixels are determined by their neighbors transitively.

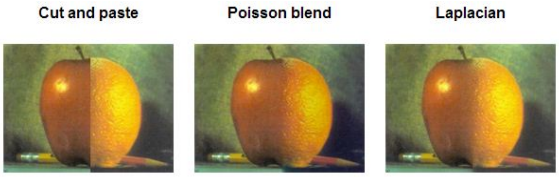


Figure 1. Comparison of Cut-paste, Laplacian and Poisson Blending

An important prior for all blending operations is perfect alignment between images being stitched together. Artifacts will be generated at the seam and the output will not be smooth if blending is performed on misaligned images.

II. RELATED WORK

The idea of Poisson Image Editing was first introduced by Perez [1]. His paper establishes the foundation for achieving seamless transition of a region inserted using gradient-domain compositing. It employs an approach based on computing the Laplacian of the unknown image, with constraints applied at the boundary, and using a guided vector field to modify the colors of the cropped region. The known pixels at the boundary gradually bleed in color into the unknown region inserted while the guided vector field taken from the source allows relative information of the original image to be preserved. To do so, a sparse linear system is solved iteratively.

Perez's proposed method works quite effectively for simple cases. Different guided vector fields may be used to attain different modifications such as texture swap, illumination and color changes, and seamless tilings. However, this approach has its own limitations as it fails to produce smooth blending for cases where there is a large gradient or contrast difference between the source and target image. Not only this, but gradient-domain compositing has memory and computational time constraints as well. It is quite slow and takes O(n) operations, where n is the number of pixels in the image. Considering this, the approach is not very scalable with high costs incurred for blending large Giga pixel images.

To encounter this issue of scalability, Agarwala [2] presents a novel idea in his paper, which is the crux of our project and will be discussed in depth in subsequent sections.

Szeliski [3] extends Agarwala's approach by incorporating multi-splines into quadtrees to further enhance the speed and memory efficiency for large panaromas. Unlike Agarwala's single offset field, the proposed method in [3] associates a separate smoothly varying spline offset map with each source image, followed by global optimization over all offset field parameters. Not only this, Szeliski's approach also improves the quality of results by introducing a multiplicative corrective approach, rather than the traditional additive approach, to converge the solution. It also uses Laplacian Image blending after its entire gradient-domain compositing process to remove visual artifacts produced due to inconsistent textures or misalignment at the seams.

The advantage of Szeliski's approach over full Poisson Blending and its quadtree implementation is that multi-splines impose smoothness in both the x and the y direction whereas poisson blending can tolerate irregular offsets only along the seam. In addition, multi-spline approach has better scaling properties compared to the quadtree approach as it depends on the number of control vertices, that roughly remain constant whereas the number of variables in Agarwala'squadtree implementation increase linearly with increase in resolution. However, one limitations of this method is that it may not work effectively for images that are not log-linear.

Another paper relevant to the subject of gradient-domain compositing is *Interactive Digital PhotoMontage*, [4]. It suggests an interactive framework that allows user to form selective composites by marking regions from different images that will collectively constitute a single image. The major steps involved in the process are detection of optimal seams in constituent images via graph-cut optimization, followed by gradient-domain fusion for removal of any visual artifacts to produce a seamlessly blended composite. This

approach produces quite interesting results while giving user the liberty to experiment with its well-designed interactive tools.

In addition to this, other approaches viewed includes [5]. This paper presents an image blending approach which combines optimal seam finding and transition smoothing for merging a set of aligned source images into a composite panoramic image seamlessly. In this approach, graph cut optimization is used for finding optimal seams in overlapping areas of the source images to create a composite image. If the seams in the composite image are still visible, a gradient domain transition smoothing operation is used to reduce color differences between the source images to make them invisible. The method is optimized for mobile application.

III. INTRODUCTION

This project is based on the work of Agarwala which describes a hierarchical approach to improve the efficiency of Poisson blending, also known as gradient-domain compositing. While the basic technique originally developed by Perez supported seamless insertion of objects from one scene to another, it has found widespread use in image stitching applications to hide seams due to exposure differences. However, Perez's approach has the issue of poor scalability due to costly memory requirements.

The output of blending in image stitching depends upon two factors; i) the similarity of the stitched image to each of the input images; ii) and the invisibility of the seam between the stitched images. Based on this criterion, Agarwala develops an approach that scales down the problem by reducing the number of unknowns while producing identical results. The method proposed combines the gradients of the source images to form a vector field that can be used to reconstruct a composite image whose gradient converges to the combined guided vector field. The underlying linear system used to solve for this composite image comes from Perez's implementation, however it is modified by integrating quadtrees to speed up the costly operation of solving over pixels of the entire image. This is done by adaptively subdividing the domain such that the smoother areas of the solution are interpolated using fewer variables.

Our initial goal of the project was to implement the method described by Agarwala and analyze the idea posed by the algorithm. However, several difficulties arose over the course of this project in terms of developing its understanding and building up on prior work. A great deal of effort was spent trying to bridge the gaps between Perez and Agarwala's algorithm implementation by going back and forth to fit the missing pieces of the algorithm together. There were several ambiguities in unknowns of the linear system especially the guided vector field and input output relationship involved in the construction of quadtree, the details of which will be discussed in section **IV**. Even though by the end of this project, we did not entirely manage to do what was intended in the beginning but we managed to gain a strong hold over the theme of Poisson Image Blending, established a sound understanding of the linear system involved along with its constraints and learnt to figure out the missing cues in the paper by thinking, analyzing and engaging in a productive exchange of ideas. Details of our project proceedings, challenges faced and conclusions derived are discussed in the section below.

IV. ALGORITHM OVERVIEW

x_0 = stitched image // Figure 2, first row

x_d = unknown // correction factor to achieve smoothness

$x = x_0 + x_d$

A = matrix of Laplacian Coefficients

b = guided vector field (combination of Laplacian of two the stitched mages)

* b represents gradient of the final blended output expected, In [2] it is taken to be Laplacian of the image obtained from Perez's Poisson Blending, Figure 2, second row.
However we obtain **b** differently as running Perez beforehand kills the purpose.

i. Find b

Suppose our panorama consists of two images.

For all image pixels p

If p in overlapped region
b[p] = mean of Laplacian of two images at p
else
b[p] = Laplacian of respective image at p
end

ii. **Compute (b-Ax_0)** // Figure 2, third row

// (b- Ax_0) gives difference of Laplacian between the stitched image and the ideal blended image (guided vector field). It is non-zero at the seam and zero elsewhere.

iii. Construct quadtree, S, of (b-Ax_0). // Figure 2, fourth row

// S is the transformation matrix that encodes the reduction from x to y, where y is the reduced space. The quadtree maximally subdivides along the seam with elements of y stored at the corners of each node. x can be recovered from y using the eq,

$$x = Sy$$

iv. Solve $S^T A^T A S y_d = S^T A^T (b - ASy_0)$ iteratively until (b-ASy_0) is zero and the solution converges

// Eq above has been derived from Ax = b, where x =Sy. It is solved using Perez's approach.

v. The solution of y_d denotes the correction factor of y

// y_d is the corrective factor to be added at each quadtree node

vi. Compute $x_d = Sy_d$

// x_d is the overall smoothness correction factor, obtained by interpolating y_d

// x_d is the corrective factor is to be added to Figure 2, third row for smoothing at the seams

vii. $x = x_0 + x_d$ // Figure 2, fifth row

// When x_d is added to the original image, the final blended output is obtained.

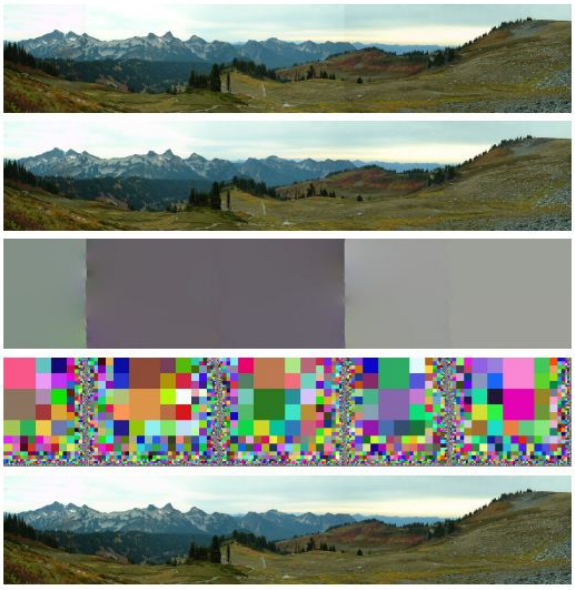


Figure 2. From Top : First shows input image, second shows Poisson blended result, third shows the difference, fourth shows QuadTree of the difference (in our case b-Ax0) and fifth shows the final output

V. PROJECT PROCEEDINGS

The project began with reviewing literature on poisson image editing to understand the mathematical tool of Poisson partial differential equations in the realm of seamless image blending. To do so, we started off by understanding the fundamental approach of [1] that first employed the principle of guided interpolation under boundary conditions imposed to solve for the Laplacian of an unknown function over the domain of interest. For a more clear understanding of the complexities involved in this, we sought help from some online resources, Dr.Murtaza and some blog implementations.

Following this, the objective of our project was defined to be an efficient and scalable implementation for seamless poisson blending based on the existing approaches of [2] or [3].

As [2] forms the basis of [3], we decided to go with Agarwala's approach of *Efficient Gradient Domain Compositing Using Quadtrees.*

The first step was to extend the existing implementation of [1] to the scenario of image

stitching. Blending at the seams of stitched images differs from the case of blending for a region cropped from a source image into a destination image, as in [1]. The former requires a combination of the gradients of the images forming the seam while the latter simply uses the Laplacian of the source image as the guided vector field. Another difficulty involved in stitched images is identifying the location of the seam in the image whereas blending for simple crop-and-insert regions has a well-defined seam from user's input. Therefore, mapping the linear system of simple insertion scenario onto image stitching was problematic and we struggled with the definition of b (the guided vector field) for the stitched composite image for quite some time.

This was followed by our experimentation phase. To begin with, as the seam lied in the overlapping region of the two stitched images, our first experiment was performed by taking user input to mark the seam and then imposing the gradient field of one of the two overlapped images in the marked area, be it the underlying image or the overlying image. However, this idea was not quite feasible because if the region marked by the user exceeded the extents of the image whose gradient is used in b, the code would break.

In order to resolve this problem, we modified the code such that every pixel within the user-identified seam region is projected into its original image coordinate frame to see if it lies in one image or both images. For pixels that belong to either one of the two images, Laplacian from the respective image was used in the guided vector field. But pixels that belonged to the overlapping region used mean of the two gradient fields. This was then improved by eliminating the need of user interaction to identify the seam by taking a combination of gradients over the entire overlapped region. The output generates smooth blending at the seam.

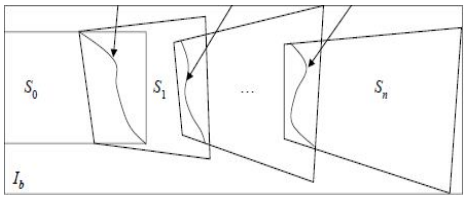


Figure 3. Overlapping regions in a stitched image

VI. EXPERIMENTS AND RESULTS

We first ran our experiments on the existing implementation of [1] in order to gain a deeper understanding of the workflow. Following are the results of an experiment.

Figure 4. Showing results of [1]

Building upon the implementation, we improved it to solve for the mosaic seams as well. There were multiple challenges involved in the extension. One other feature we added was the automatic blending of an input panorama and the original images that built it. For this part we detected the seams automatically by using $b - Ax_0$ as done in section IV and then solved for those parts using Perez. The results of experiments on two standard datasets are shown in figure5 on the next page.

Figure 5. Blending results with and without user marked seams on two standard datasets

Extending this idea to [2], we formulated the problem as shown in section IV and followed the pipeline as in figure2, which shows the step-wise results of the pipeline followed by Agarwala. Here, we show the results in the same pattern except for row two and row five of figure 2. We didn't show row five because we couldn't generate blended results using this pipeline whereas row two is not shown because our way of generating **b** is different (refer to section IV). Figure6 and figure7 show our generated results.

VII. CONCLUSION

Although Poisson blending gives splendid results, but there are still some limitations to it. For example, in cases when pixel colors of the unknown region differ from our initializations at the boundary, we do not get the desired output. The primary reason being that we are only considering gradients of the unknown region. If we somehow try to inculcate the cue of color in our system while solving for the unknown region, then this could lead us in a new direction. We have built a strong understanding in the domain of Poisson and have tried to make the learning process easier for people who pursue this project later. We have tried to thoroughly explain solutions to all the difficulties we had during the project including an end to end explanation of the quad-tree algorithm. We have partially implemented this algorithm as well, so now we could look into devising solutions to the issues in these state of the art Poisson blending algorithms.

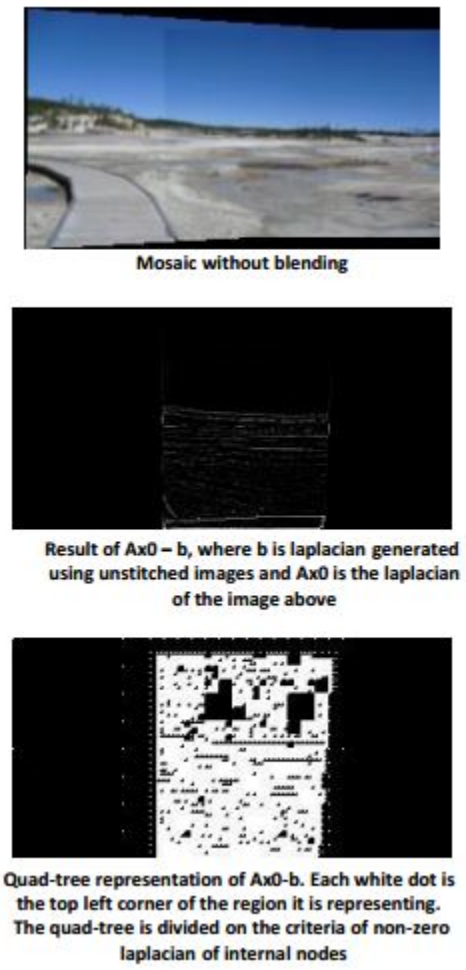
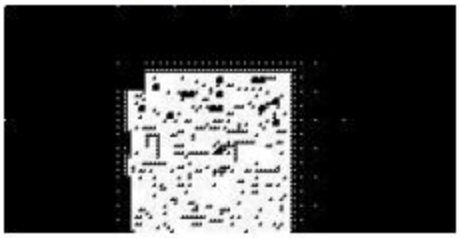


Figure 6. Procedure followed to construct the quadtree as shown in Figure1 of [2].

Mosaic without blending

Result of Ax0 – b, where b is laplacian generated using unstitched images and Ax0 is the laplacian of the image above



Quad-tree representation of Ax0-b. Each white dot is the top left corner of the region it is representing. The quad-tree is divided on the criteria of non-zero laplacian of internal nodes

Figure 7. Procedure followed to construct the quadtree as shown in Figure1 of [2].

I. REFERENCES

[1] P Perez, M Gangnet, A Blake, Poisson image editing. ACM Trans Graph.22(3), 313–318 (2003).

[2] A. Agarwala. Efficient gradient-domain compositing usingquadtrees.ACM Transactions on Graphics, 26(3), August2007.

[3] R. Szeliski, M. Uyttendaele, and D. Steedly.Fast poissonblending using multi-splines.In Technical Report MSR-TR-2008-58, Microsoft Research, 2008.

[4] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. H. Salesin, and M. F. Cohen.Interactive digital photomontage. ACM Transactions on Graphics (Proc. SIGGRAPH 2004), 23(3):292–300, August 2004.

[5] Y. Xiong and K. Pulli. Gradient domain image blending and implementation on mobile devices. In MobiCase, 2009.

[6] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. ACM Transactions on Graphics, 2(4):217–236, 1983.