Depth-map estimation from stereo image pair and simulation of depth of field

Arjun Chandrasekaran
MEng, ECE
Virginia Tech
carjun@vt.edu

Tarun Patil
MEng, ECE
Virginia Tech
tpatil77@vt.edu

Abstract

We compute depth-map of a scene from a rectified stereo image pair based on template matching and graph cut segmentation. Using this depth-map, we develop an interactive program where the user can choose to selectively focus any point in the image by clicking on the point. Points at the same depth as the selected point are retained, while points at different depths in the scene are de-focused. With increasing distance we gradually increase the intensity of blurring filters for points farther away from the chosen focal plane to simulate shallow depth of field.

1. Introduction

Depth maps give an estimate of the relative distance of objects from the camera in a scene. They have a number of applications like robot navigation, computational photography, industrial applications, entertainment applications like video games and 3D video generation. An interesting application of depth maps that we explore in our project is their use in simulating shallow depth of field.

The popular definition for depth of field given by Conrad[1] is the 'zone of sharpness' in an image that exists to either side of the focal plane. Depth of field effects have been traditionally made possible only using SLR cameras due to the need for high focusing power and variable aperture of the lens. In recent times these effects have been simulated in photographs taken with less powerful lenses such as phone camera lenses, as a post-production effect using software, for example Google Lens Blur[2] and apps discussed in iPhone Photography School[3].

We develop a program in MATLAB to compute a depth map from a rectified stereo image pair using template matching and graph cut methods[4][5][6]. We apply this computed depth map to simulate shallow depth of field effects in a pre-exposed image by interactively asking a user to select a point in the image to be kept in focus, while the remaining points at depths different from the point selected are blurred.

2. Related Work

Stereo matching is an extensively studied topic in Computer Vision. The Middlebury Stereo Evaluation page[7] alone lists and evaluates over a 150 algorithms for dense stereo matching and depth estimation. The reason for this tremendous interest in stereo matching is because it is the most robust passive method for depth estimation. The numerous practical applications for stereo matching have further encouraged researchers to find faster, more robust and less computationally expensive algorithms.

One of the most widely used approaches towards stereo-matching in recent times has been graph-cuts based methods[4][5][6]. We develop our algorithm for stereo matching by doing a preliminary template matching followed by graph-cut methods. We recently came across this paper by Hong and Chen[8] whose general ideas are similar to our own implementation of graph-cut based segmentation and segment based disparity assignment.

3. Depth map estimation

In this section we discuss our implementation of stereo matching algorithm to generate an estimate of the depth map of the scene. We develop and test our algorithm on the Tsukuba image of the Middlebury stereo dataset.

3.1. Template matching

We implement standard template matching with the left image as our reference image. We assign an image region in the left image to our template and try to find a match in the right image. This process is continued iteratively until we have traversed through and found matches for all regions in the left image. We assume the images are rectified; hence for a template in the left image, we look for a match only along the same corresponding row in the right image. Furthermore, we do not search along the entire row – we search only within a fixed window (which we call disparity window) which is centered about the center of the template.

Both template size and disparity window size can be changed for a specific stereo pair. This is essential since

(a) (b) (c)

Figure 1: Variation in disparity map with change in template size. Darker regions have high disparity and lighter regions have low disparity. (a) Template size = 20, Maximum disparity= 40; (b) Template size = 9, Maximum disparity= 20; (c) Template size = 31, Maximum disparity= 50;

(a) (b) (c)

Figure 2: Qualitative comparison of disparity map for different matching criteria. Darker regions have high disparity and lighter regions have low disparity. (a) sum of absolute differences; (b) sum of squared differences; (c) normalized cross correlation

the nature of images can vary widely and both these parameter values play an important role in the quality of the generated disparity map. For images with large number of details (objects, edges, color variations), we can see in general that a lower template size is sufficient for a good match and for an image with fewer details, a larger template size may be required for good matching. The disparity window size depends on the baseline distance between the left and right cameras and the range of depths of objects. We observe that, closer an object to the camera more is the disparity between left and right images. Similarly, higher the baseline between left and right cameras, higher is the disparity between the two images.

Figure 1 shows the variations in disparity maps obtained for the same stereo image pair when we varied template size and disparity window size.

3.2. Matching criterion

To correctly match corresponding image regions in the two images, we used a function of the left image template and right image disparity window that we aimed to minimize. We implemented 3 different functions (matching criteria) namely, sum of absolute differences (SAD), sum of squared differences (SSD) and normalized cross correlation (NCC). From Figure 2, we qualitatively observe that the performance of SSD, SAD NCC is comparable since all three matching criteria are extremely noisy. NCC is qualitatively the best matching criteria due to slightly lesser object edge noise. However, as we see in Table 1, the very high computational complexity of NCC offsets its advantage. Since SAD and SSD generate disparity maps of similar quality, we choose the least computationally expensive matching criterion – SAD.

	SAD	SSD	NCC
template_size	20	21	21
max disparity	40	40	40
time taken	19.79 sec	67.39 sec	3 hrs

Table 1. Comparison of time taken by different matching criteria with different template sizes

To improve performance, we additionally define two other functions that we attempt to minimize when performing a template match – gradients of image along x and y directions respectively. By matching gradients along with raw intensity values across left and right images, inside-object disparity noise is reduced to a large extent, as seen in Figure 3. We take a linear combination of the intensity disparities and gradient disparities to produce our template matched disparity map. Upon experimentation with different weights for the gradient disparity, we found

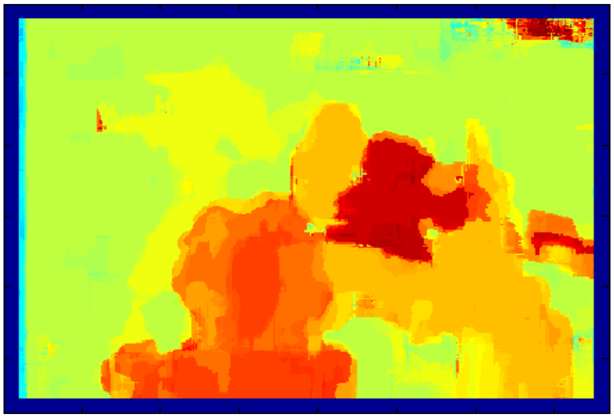

Figure 3. Linear combination of intensity disparity map and gradient disparity map. Dark red regions have higher disparities. Light yellow regions have lower disparities.

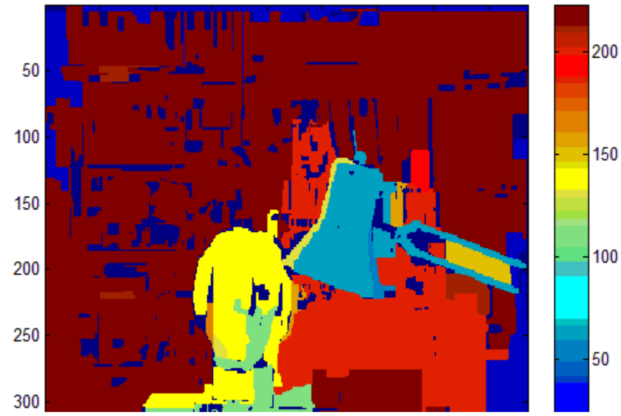

Figure 4. Disparity map produced after image segments are assigned with dominant disparities.

that for the best disparity map, this parameter also varies depending on the pair of images. Thus, this is also left as a modifiable parameter in our implementation source code.

3.3. Correlation with Segmented image

Due to the large presence of noise, we implement a disparity map generation algorithm that involves correlation of the disparity map with a segmented image. For image segmentation, we use the kernel graph-cut code by Ayed[10] and MATLAB wrapper for Veksler et al.'s implementation of Graph Cut algorithm[4],[5],[6] provided by Bagon[9]. Ayed's code implements multi-region graph cut image segmentation according to the kernel-mapping formulation in Salah et al. [11].

From Ayed's kernel graph-cut code segments an image into 'k' (parameter that is chosen by user) regions based on color. We take the left image as reference and give this as input to Ayed's segmentation code. The output is our segmented image with each segment having a unique color. In our correlation code, we iterate over each of these 'k' segments and look for the dominant disparity for that region in the disparity map computed by template matching in Section 3.2. If the region is too small, we ignore it, otherwise, the dominant disparity is assigned to the entire segmented object. Thus, we generate a less noisy disparity map, as shown in Fig 4.

The drawbacks to this method are that, by assigning an image segment with the dominant disparity value, we lose the true disparity values of every point in our image. However, in our application of blurring an object in an image, we consider this segmentation more useful than harmful; hence the introduction of non-linearity does not adversely affect our output. For reference, we show in Figure 5, the image with which we developed and tested our algorithm – the left image of Tsukuba, one of the images in the Middlebury stereo dataset.

Figure 5. Left image of Tsukuba

4. Depth of field simulation

All text must be in a two-column format. The total title (on the first page) should begin 1.0 inch (2.54 cm) from the top edge of the page. The second and following pages should begin 1.0 inch (2.54 cm) from the top edge. On all pages, the bottom margin should be $1\frac{1}{8}$ inches (2.86 cm) from the bottom edge of the page for 8.5 × 11-inch paper; for A4 paper, approximately $1\frac{5}{8}$ inches (4.13 cm) from the bottom edge of the page.

3.4. Margins and page numbering

All printed material, including text, illustrations, and charts, must be kept within a print area $6\frac{7}{8}$ inches (17.5 cm) wide by $8\frac{7}{8}$ inches (22.54 cm) high. Page numbers should be in footer with page numbers, centered and .75 inches from the bottom of the page and make it start at the correct page number rather than the 4321 in the example (how to do that depends on your version of word or open office. Failure to use the correct page number, or place it properly, could result in the paper not being included in Xplore, (even if it passes PDF express (which does not check page number)

3.5. Type-style and fonts

Wherever Times is specified, Times Roman may also be used. If neither is available on your word processor, please use the font closest in appearance to Times to which you have access.

MAIN TITLE. Center the title 1-3/8 inches (3.49 cm) from the top edge of the first page. The title should be in Times 14-point, boldface type. Capitalize the first letter of nouns, pronouns, verbs, adjectives, and adverbs; do not capitalize articles, coordinate conjunctions, or prepositions (unless the title begins with such a word). Leave two blank lines after the title.

AUTHOR NAME(s) and AFFILIATION(s) are to be centered beneath the title and printed in Times 12-point, non-boldface type. This information is to be followed by two blank lines.

The ABSTRACT and MAIN TEXT are to be in a twocolumn format.

MAIN TEXT. Type main text in 10-point Times, singlespaced. Do NOT use double-spacing. All paragraphs should be indented 1 pica (approx. 1/6 inch or 0.422 cm). Make sure your text is fully justified—that is, flush left and flush right. Please do not place any additional blank lines between paragraphs.

Figure and table captions should be 9-point Roman type as in Figures 1 and 2. Short captions should be centred. Callouts should be 9-point Helvetica, non-boldface type. Initially capitalize only the first word of section titles and first-, second-, and third-order headings.

FIRST-ORDER HEADINGS. (For example, **1. Introduction**) should be Times 12-point boldface, initially capitalized, flush left, with one blank line before, and one blank line after.

SECOND-ORDER HEADINGS. Should be Times 11-point boldface, initially capitalized, flush left, with one blank line before, and one after. If you require a third-order heading (we discourage it), use 10-point Times, boldface, initially capitalized, flush left, preceded by one blank line, followed by a period and your text on the same line.

3.6. Footnotes

Please use footnotes[1] sparingly. Indeed, try to avoid footnotes altogether and include necessary peripheral observations in the text (within parentheses, if you prefer, as in this sentence). If you wish to use a footnote, place it at the bottom of the column on the page on which it is referenced. Use Times 8-point type, single-spaced.

[1] This is what a footnote looks like. It often distracts the reader from the main flow of the argument.

3.7. References

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [4]. Where appropriate, include the name(s) of editors of referenced books.

3.8. Illustrations, graphs, and photographs

All graphics should be centered. Please ensure that any point you wish to make is resolvable in a printed copy of the paper. Resize fonts in figures to match the font in the body text, and choose line widths which render effectively in print. Many readers (and reviewers), even of an electronic copy, will choose to print your paper in order to read it.

You cannot insist that they do otherwise, and therefore must not assume that they can zoom in to see tiny details on a graphic. When placing figures in LATEX, it’s almost always best to use \includegraphics, and to specify the figure width as a multiple of the line width as in the example below

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]
{myfile.eps}
```

3.9. Color

Color is valuable, and will be visible to readers of the electronic copy. However ensure that, when printed on a monochrome printer, no important information is lost by the conversion to grayscale.

4. Final copy

You must include your signed IEEE copyright release form when you submit your finished paper. We MUST have this form before your paper can be published in the proceedings. Please direct any questions to the production editor in charge of these proceedings at the IEEE Computer Society Press: Phone (714) 821-8380, or Fax (714) 761-1784.

References

[1] Jeff Conrad. Depth of field in depth, 2006. www.largeformatphotography.info/articles/DoFinDepth.pdf

[2] Carlos Hernandez. Lens blur in the new Google camera app, 2014. Google Research Blog.

http://googleresearch.blogspot.com/2014/04/lens-blur-in-new-google-camera-app.html

[3] Rob dePaolo. How to create depth of field in iphone phtography, 2014. iPhone Photography School. http://iphonephotographyschool.com/depth-of-field/

[4] Yuri Boykov, Olga Veksler, Ramin Zabih. Efficient Approximate Energy Minimization via Graph Cuts. IEEE transactions on PAMI, vol. 20, no. 12, p. 1222-1239, November 2001.

[5] Vladimir Kolmogorov and Ramin Zabih. What Energy Functions can be Minimized via Graph Cuts? European Conference on Computer Vision (ECCV), May 2002

[6] Yuri Boykov and Vladimir Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), September 2004.

[7] D. Scharstein and R. Szeliski. Middlebury stereo vision evaluation page. http://vision.middlebury.edu/stereo/eval.

[8] Li Hong and George Chen. Segment based Stereo Matching using Graph Cuts. 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)

[9] Shai Bagon. Matlab Wrapper for Graph Cut. 2006. http://www.wisdom.weizmann.ac.il/~bagon

[10] Ismail Ben Ayed. Kernel graph cut image segmentation. 2012. http://www.mathworks.com/matlabcentral/fileexchange/38555-kernel-graph-cut-image-segmentation.

[11] M. Ben Salah, A. Mitiche, and I. Ben Ayed. Multiregion Image Segmentation by Parametric Kernel Graph Cuts, IEEE Transactions on Image Processing, 20(2): 545-557 (2011).

[12] Alpher, , and J. P. N. Fotheringham-Smythe. Frobnication revisited. Journal of Foo, 13(1):234–778, 2003.

[13] A. Alpher, , J. P. N. Fotheringham-Smythe, and G. Gamow. Can a machine frobnicate? Journal of Foo, 14(1):234–778, 2004.

[14] A. Alpher. Frobnication. Journal of Foo, 12(1):234–778, 2002.

[15] Actual Author Name. The frobnicatable foo filter, 2014. Face and Gesture (to appear ID 324).

[16] Actual Author Name. Frobnication tutorial, 2014. Some URL al tr.pdf.