Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach (An Implementation Project)

Leigh Hagestad

June 6th, 2016

Abstract

This paper seeks to document the steps I underwent in implementing the 1996 SIGGRAPH paper titled "Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach" (Debevec et al.). This paper outlines the implementations of two of the more significant and involved steps in the original paper, "Photogrammetric Modelling" (based on Parametric Primitives), as well as "View-Dependent Texture Mapping". The paper shows the implementation of the Graphical User Interface built with the purpose of constructing simplified geometric scenes from a small number of photos of architecture, covers an implmentation of the traditional Structure from Motion problem as a comparison heuristic, and finally demonstrates the visual results from the view-dependent texture mappings. Interactive videos of the final results are included in the paper.

Introduction

For my final project, I chose to implement a simplification of the following 1996 SIGGRAPH paper from Paul Debevec and his team from Berkeley: "Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach". The paper, as its title suggests, outlines a novel way of modeling and rendering architectural geometry and scenes from a small set of original 2D images. The authors of the original paper perhaps put it best themselves when they wrote: "Efforts to model the appearance and dynamics of the real world have produced some of the most compelling imagery in computer graphics. In particular, efforts to model architectural scenes, from the Amiens Cathedral to the Giza Pyramids to Berkeley's Soda Hall, have produced impressive walk-throughs and inspiring fly- bys. Clearly, it is an attractive application to be able to explore the world's architecture unencumbered by fences, gravity, customs, or jetlag" [1].

Prior Work

Obviously, Debevec's 1996 paper is the most salient piece of literature for this project, however in his version of the paper (which is the published product of his thesis work) provides a solid outline of the computer vision problems, solutions, and techniques which predate this problem and contribtue to it's implementation, considered techniques, and success. Debevec outlines the four pivotal issues in computer vision which this paper addresses and contributes novel (at the time) solutions to. These four problems are: Camera Calibration, Structure from Motion, Stereo Correspondence, and Image-Based Rendering. Debevec identifies the seminal work in each of these four areas, and how the considerations and scope of those solutions and techniques have driven him to consider the constraints and opportunities of the problem facing him. The references he identified as most important are (for each field respectively): Camera Calibration: [1], [2], [3], [4]; Structure from Motion: [5], [6], [7], [8], [9]; Stereo Correspondence: [10], [11], [12], [13], [14], [15]; and Image-Based Rendering: [16], [17], [18]. Lastly, Debevec's own prior work [19] clearly proved an inspiration for the '96 paper.

Motivations

Debevec's 1996 Motivations

The motivations for the original paper built on the fact that, until the time that it was published, all techniques for rendering architecture from photographs were either highly labor-intensive (requiring location scouting, measuring, etc.), required the digitization of existing analog models, or built upon existing stereo algorithms, which required many closely-spaced photographs to produce even remotely accurate reconstructions. Furthermore, the rendering technologies left buildings in the uncanny valley, and were not as photorealistic as desired. The motivation for this system was "to make the process of modeling architectural scenes more convenient, more accurate, and more photorealistic than the methods currently available.

Personal Motivations

This paper is non-trivial and leverages nearly every major concept of computer vision that we covered in the class at the time of the proposal, including (but not limited to) camera calibration, correspondence issues, epipolar geometry, and multi- view/stereo geometry. I thought that using my project as an opportunity to implement this paper would provide a respectable opportunity to enage with hands-on experience developing a system which covers many canonical areas and problems in computer vision in a modular way, with a rewarding and visually attractive final deliverable. Furthermore, I have a general love for architecture and have some prior experience working with (but not constructing) digital building models and renderings, and as such this topic played into my personal interests really well.

Technical Solution

Photogrammetric Modelling

0.0.1 Tools Utilized

MATLAB

I chose to use MATLAB for the first part of the paper, given the software is obviously designed with linear algebra optimizations, and furthermore includes many helpful and well-designed functions for designing user-interfaces for graphical data input, particularly with respect to photos as well as 2D and 3D plotting functionalities.

Maya

After significant testing, tampering, struggling with MATLAB documentation and tutorials to produce well-rendered, interactive image projections onto geometric surfaces, I chose to use the Maya rendering and animation software as a platform for projective and texture mapping component of the project. Maya is an excellent tool for any applications relating to camera and projection applications and provides the user with a tremendous amount of control over the three dimensionanl transformations to models given highly precise inputs for translation, rotation, and scale, as well as factors including focal length, projective transformations, and 3D to 2D rendering. Lastly, I chose to employ Maya for this project because it provides a very sturdy interface for producing short video clips.

Course Resources

I would like to note that I also utilized solutions and code from different Problem Sets as a means to recreate some of the functionality in the first and second part of this project. I am sorry to say that my own code for such assignments was not as strong as it could have been (nor, of course, as strong as an instructor's) and chose to take advantage of these resources as a means of improving the accuracy and optimization of my code.

0.0.2 MATLAB GUI Implementation

The original paper introduces the "Facade" software, a pieec of technology which takes as input a series of images and outputs a simplified three-dimensional model of the scene captured across the varying viewpoints of the images. Facade enables users to recreate the gross geometry of the scene by selecting from a subset of geometric primitives (including wedge and rectangular volumes) and supeimpose those geometries over the image to highlight what level of architectural resolution is being captured by the model. While certainly a more labor-intensive (for the user) form of geometric modelling, this techniwur offers significant improvements over prior reconstruction techniques in the following ways: 1) it enables the user to only identify salient features in a piece of architecture or image, reducing the total information load required to render 2) is, consequentially, significantly less prone to error by noise, 3) leverages the perceptive and spatial reasoning abilities of humans and, 4) leverages critical intrinsic features which are specific to the architecture class of objects. Such class features include the extensive reliability on parallel lines, right angles, smooth curves, and flat surfaces - in short, parametric geometry.

Given that I was a one-person team rather than a group of Ph.D candidates working on a thesis, I chose to implement a simplified but analogous version of this piece of human-computer interaction. My version follows the paper's functionality in that it enables users to construct simplified models of geometric scenes shown in 2D images. This implementation enables a user to select an image subject, and derive a modelled collection of geometric 'block' primitives based upon graphical user input in MATLAB with the provided image.

Per the original paper's suggestion, this system utilizes two types of geometric block primitives: rectangular blocks (extrapolated rectangles) and 'wedge' blocks (extrapolated triangles). Users can interact with the system to reconstruct the high-level architecture of the building(s) in their scene by utilizing these primitives as building blocks. The control flow of the application in order to collect this input is as follows:

1) Select Image for basis (in MATLAB code)

Iterate:

2) Pick type of primitive block to model: rectangle or wedge.

3) System provides guidance about the ordering of user generated line input which compose the parameters for the primitives.

4) Users identify two corresponding planes of the primitive in the scene using lines. For rectangular blocks, this is two planar sides of the shape; for wedges, this is the triangular side of the shape and one of the planar sides of the wedge.

5) The user is prompted to provide a name for the primitive they've just designed.

6) If more than one block exists within the currently modelled architecture, users are asked if they would like to encode a spatial relationship between the primitive they are currently building and one which is already in the system. If the user wants to encode such a relationship with another piece of geometry in the scene, they select the other primitive of interest. If the user opts not to encode a spatial relationship, they are returned to the option to model another primitive (or cancel).

7) If the user has chosen to encode a spatial relationship, they then are asked what type of relationship exists between the current and existing primitive. Users choose one of 6 types: over, under, left of, right of, in front of, or behind.

8) Finally, the user is prompted whether or not the base of most recent piece of geometry should match that of the primitive is has the new spatial

relationship with. This is helpful for alignind objects which should stack neatly on top of each other (e.g. roof on building).

9) Once the user chooses not to model another piece of geometry, he or she can cancel out of the dialog loop. At this point, he or she will see a 3D scatterplot of the geometry they have rendered.

See the image series in the results section below for a visual explanation of this process.

0.0.3 Fundamental Matrix Derivation

I used the Normalized Eight-Point algorithm for derivation of the Fundamental Matrices between images. Point correspondences were taken from points identified in the parametric primitives from the photogrammetric modelling stage. Then I used the Bundle Adjustment to determine rotation and translation between camera views.

0.0.4 Compared with traditional Structure From Motion Techniques

I chose to apply the traditional form of the factorization Structure from Motion (SFM) technique as a comparative heuristicfor the overall effectiveness of extrapolating architecture structure from a series of images. I wanted to compare the visualizations of the produced models from the parametric primitive implementation utilized in the MATLAB GUI version (obviously based off of Debevec's initial work and implementation) with those from the SFM technique to quickly illustrate the significant improvement offered by Debevec's technique. I utilized the projective triangulation SFM technique outlined and implemented during the course as a comparative measure to show the computational and reconstructed shortcomings of this technique when compared to the photogrammetric modelling technique, at least in the case of the Architecture-class reconstruction problem.

View-Dependent Texture-Based Mapping

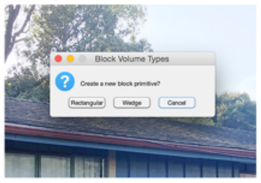
For the view dependent texture mapping phase, I took advantage of the intrinsic parameters known about each photograph (e.g. focal length was known, no skewness, and square pixels for given camera assumed), and the extrinsic parameters derived as stated above, to include this information Maya's interface for projective image rendering on a model. The model was produced by mapping the verteces identified in the photogrammetric modelling stage and using Maya's geometric primitive modelling engine to replicate the primitives with the specified dimensions. A perspective projection with the relevant values for focal length were utilized in the projection of photos onto the model, and the projection was interpolated on the model appropriately (as outlined in the original paper) to reflect the most salient view point. A viewer "exploring" the scene in Maya's editor can appreciate these location-based projections in real time.

Results

Photogrammetric Modelling

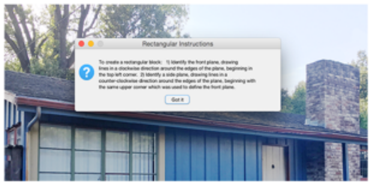
The implenentation was well formed as describte above. See the image series below for a visual understanding of the process.

1) Image Selection


Block Volume Types

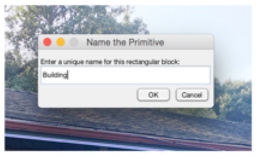
Create a new block primitive?

2) Primitive Type Selection


Rectangular Instructions

To create a rectangular block: 1) Identify the front plane, drawing lines in a clockwise direction around the edges of the plane, beginning in the top left corner. 2) Identify a side plane, drawing lines in a counter-clockwise direction around the edges of the plane, beginning with the same upper corner which was used to define the front plane.

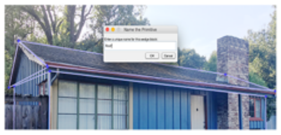
3) Type Construction Instructions

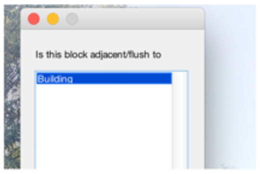

Name the Primitive

Enter a unique name for this rectangular block:

Building

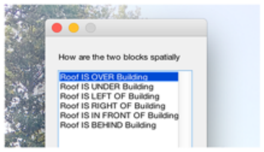
3), 4) Block Image Outline
+ Name Dialog


5) Name Dialog
(roof + building modeled here)


Is this block adjacent/flush to

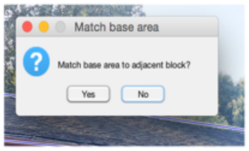
Building

6) Asks about spatial
relationship with existing blocks


How are the two blocks spatially

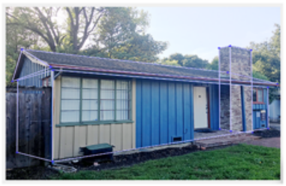
Roof IS OVER Building
Roof IS UNDER Building
Roof IS LEFT OF Building
Roof IS RIGHT OF Building
Roof IS IN FRONT OF Building
Roof IS BEHIND Building

7) Prompts about spatial relationship
with specified block

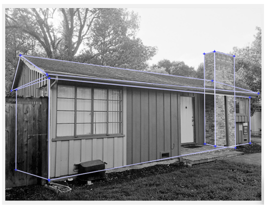

Match base area

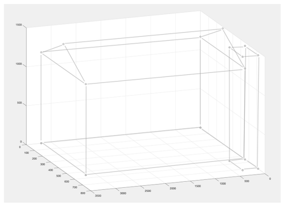
Match base area to adjacent block?

8) Asks if should match base
area with adjacent block


9) Final overlaid image
with primitive block lines

Input Image


Block Primitives Overlaid


Point-wise Model
Figure 1

SFM Reconstruction Comparison Heuristic

I noted above, I utilized the same form of the projective triangulation SFM discussed in the course and in the second problem set as a means of reconstructing the geometry of the scene and the camera path. This technique included the recovery of image correspondences between multiple images (see Figure 2 below for an example of one such correspondence set where N=20). Given the Fundamental and Essential matrices and calculated in the previous step, we could derive calculations of the R and T matrices sets, and were able to calculate the relevant 3D points from correspondences. I then used the known nonlinear optimization and Newton steps outlined in the course to minimize the projection error. Figure 3 represents the point cloud representing the reconstruction derived from this model of SFM. Comparing it to Figure 1, we note the significant improvement in recognizablility afforded by Debevec's geometric-based technique.

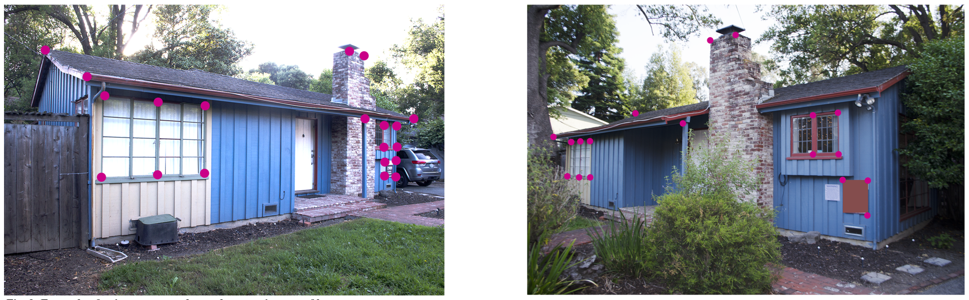


Fig. 2. Example of point correspondences between images of house.

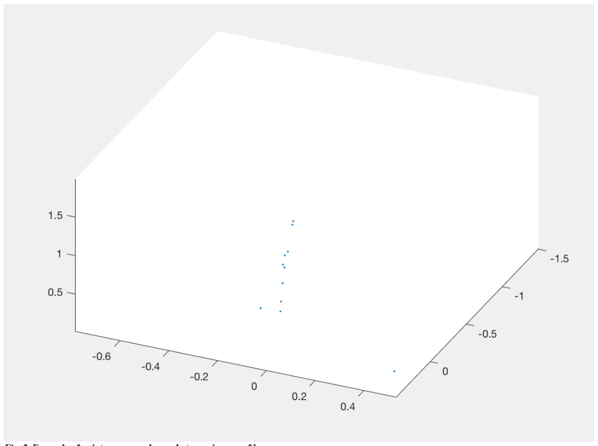


Fig. 2. Example of point correspondences between images of house.

One can clearly see that the SFM implementation is unquestionably outshone by the photogrammetric modelling technique, both in user satisfaction, and end result. The verteces of the house produces by the SFM reconstruction cannot be accurately teased out or identified, and thus we cannot accurately produce a measurement statistic when comparing to the photogrammetric model. However, by inspection alone we can clearly identify the superior model.

In terms of timing, SFM clearly has an advantage. SFM runs in about 6 seconds, whereas the photogrammetric modelling task takes several minutes to set geometries by hand. However, the overall result is so much better and more recognizable that we clearly opt for Debevec's technique.

As for the accuracy of Debevec's technique relative to the ground truth (i.e. the real 3D measurements of my house), I did not have an instrument large enough nor (as Euclid would say, “the right place to stand”) to accurately gather measurements for my house with which to compare the model produced down to dimensional accuracy of the the photogrammetric model. However, we can tell by inspection that latter model is recognizable to the point of inspection, so - for the purposes of an implementation of this scope - we deem it to be ’close enough’.

Viewpoint Based Texture Mapping

The images below display multiple views of two texture mapped pieces of architecture: the house shown throughout this paper, as well as a collection of “Cube Houses” (located in Rotterdam, The Netherlands). Notice the positioning of the cameras and the view-dependent changes in texture as the position of the viewer changes relative to the scene. The reader is also invited to see live clips of a camera moving about in these scenes at the following links: House Video , Cube Houses Video.

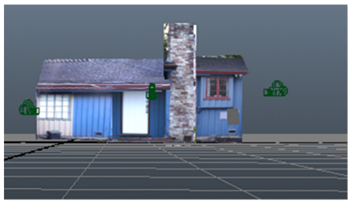


Fig 4a. House, Front View

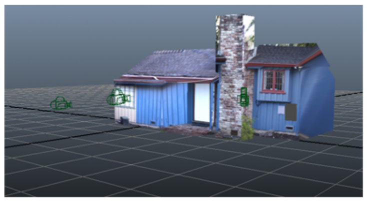


Fig 4b. House, Right View

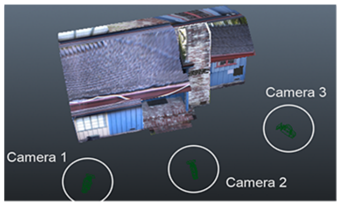


Fig 4c. House, Top View. Projective Cameras Shown

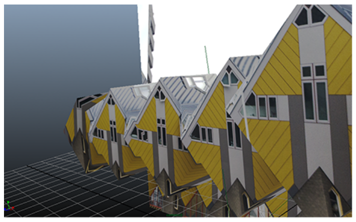


Fig 5a. Rotterdam Cube Houses, Right View.

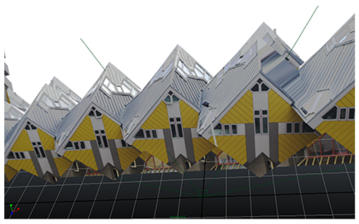


Fig 5b. Rotterdam Cube Houses, Center View.

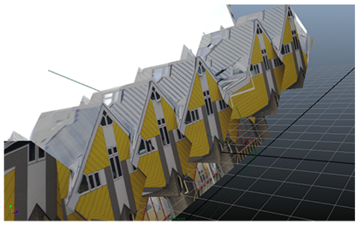


Fig 5b. Rotterdam Cube Houses, Left View.

Conclusions

General Remarks

I really enjoyed the process of getting to implement this paper, for the opportunities to explore and practice different important issues in computer vision, become more familiar with the work of Paul Debevec (who is something of an academic hero of mine), and to produce some stunning (and less stunning) visual deliverables. While the house rendering was a nice starting point, I thought it was fairly mediocre in terms of visual acuity (you can see bleeding and warping at several points on the model from the projection). However, I was far happier with how the Cube Houses projection turned out and think it is a stronger artifact of the implementation.

Limitations of Implementation

One of the biggest dissapointments about this implementation was the lack of evaluation heuristics available to judge the success of the implementation. While visual inspection and human discernment is ultimately what matters in work of this nature, I felt somewhat lacking in harder, statistical metrics of success. Secondly, I was dissapointed to not have had the time to complete the third part of the paper, stereo depth mapping based on the differences in the view-based texture mapping. While this part of the implementation might have been yet another great opportunity to learn and experience an important computer vision technique, the paper itself doesn't focus much on this technique and actually credits much of the theoretical and developed methods to the prior work. As such, I don't feel so cheated in having not implemented it. Lastly, I was somewhat dissapointed at parts of the model that aren't perfect. The geometric model certainly isn't an exact replica of their true architectural counterparts, and these errors lead to noticeable artifacts such as warped projections, seams. With stronger statistical evaluation, we could better determine how to fix these issues.

Future Work

I would love to build on this paper by going the further step and actually implementing the third part, the stereo mapping, as well as exploring and implementing the pivotal work and techniques in computer vision and architecture rendering that has built upon this seminal paper. Lastly, I would greatly look forward to producing more visual 3D renderings of architecture for personal experience and portfolio.

References

[1] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and Rendering Architecture from Photographs. In SIGGRAPH '96, August 1996.

[2] Oliver Faugeras and Giorgio Toscani. The calibration problem for stereo. In Proceedings IEEE CVPR 86, pages 15–20, 1986.

[3] Roger Tsai. A versatile camera calibration technique for high accuracy 3d ma- chine vision metrology using off-the-shelf tv cameras and lenses. IEEE Journal of Robotics and Automation, 3(4):323–344, August 1987.

[4] Olivier Faugeras, Stephane Laveau, Luc Robert, Gabriella Csurka, and Cyril Zeller. 3-d reconstruction of urban scenes from sequences of images. Techni- cal Report 2572, INRIA, June 1995.

[5] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. Nature, 293:133–135, September 1981.

[6] E. Kruppa. Zur ermittlung eines objectes aus zwei perspektiven mit innerer ori- entierung. Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. Ila., 122:1939– 1948, 1913.

[7] Olivier Faugeras. Three-Dimensional Computer Vision. MIT Press, 1993.

[8] S. Ullman. The Interpretation of Visual Motion. The MIT Press, Cambridge, MA, 1979.

[9] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision, 9(2):137–154, November 1992.

[10] Camillo J. Taylor and David J. Kriegman. Structure and motion from line seg- ments in multiple images. IEEE Trans. Pattern Anal. Machine Intell., 17(11), November 1995.

[11] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In Proceedings of the Seventh IJCAI, Vancouver, BC, pages 631–636, 1981.

[12] D.J.Fleet, A.D.Jepson, and M.R.M. Jenkin. Phase-based disparity measurement. CVGIP: Image Understanding, 53(2):198–210, 1991.

[13] W. E. L. Grimson. From Images to Surface. MIT Press, 1981.

[14] D. Jones and J. Malik. Computational framework for determining stereo cor- respondence from a set of linear spatial filters. Image and Vision Computing, 10(10):699–708, December 1992.

[15] D. Marr and T. Poggio. A computational theory of human stereo vision. Proceed- ings of the Royal Society of London, 204:301–328, 1979.

[16] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. A stereo correspondence algo- rithm using a disparity gradient limit. Perception, 14:449–470, 1985.

[17] Lance Williams and Eric Chen. View interpolation for image synthesis. In SIG- GRAPH '93, 1993.

[18] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based ren- dering system. In SIGGRAPH '95, 1995.

[19] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. Technical Report UCB//CSD-96-893, U.C. Berkeley, CS Division, January 1996.