Computer Science Project Proposal

Constructing 3D Models from Image Sequences

I. G. Sorescu, Newnham College, igs23

Originator: I. G. Sorescu

24th October 2014

Project Supervisor: K. Palyutina

Director of Studies: Dr J. K. Fawcett

Project Overseers: Dr A. C. Rice & Dr T. G. Griffin

Introduction and Description of the Work

The aim of this project is to extract a partial 3D model of a small object from a sequence of plain images and then to export the model into a 3D geometry format such as PLY. The resulting file can then be rendered and processed by MeshLab or similar mesh processing software.

The main focus of the project is to obtain a partial wireframe model from a sequence of images taken from pre-established locations with respect to the object. An open-source ray tracer will be used in order to set up a static scene consisting of the object to be reconstructed (placed in a well-lit environment against a white background). The sequence of input images will be acquired by taking digital snapshots of this scene from the pre-established locations, thus imitating the process of photographing the object with a pre-calibrated camera. The project is designed to be later extended to support real camera input.

What follows is a description of a technique which can be used in order to obtain the image sequences. The setup is described here in terms of photographing real objects in order to emphasize the fact that the project can naturally be extended to interact with a real camera. However, for the purposes of this project, this technique will be imitated by digitally constructing the equivalent snapshots of an existing 3D model in an attempt to reconstruct its shape (as seen from the sides).

Define an origin on a perfectly horizontal table and place the camera there facing a white background. Place an object of a pre-determined maximum height at a fixed distance from the origin in front of the camera. Take the first picture. Rotate the object by 30 degrees counter clockwise and take the second picture. Repeat until 12 pictures are taken. Input the pictures in the order in which they were taken. This process is assumed to have taken place in well-lit environment with no major light differences between pictures in the same set.
The input generated using this process provides no information about the shape of the object as seen from above or below. The resulting partial 3D model must only be consisted with what is visible from the views provided.

Starting Point

I am planning to use an open-source ray tracer such as POV-Ray in order to create image sequences from an existing 3D model in an attempt to recreate its shape. I will also be using the feature detection and feature matching functionalities of OpenCV.

Resources Required

Some open source libraries will be used as described in the previous section. They can be downloaded freely from http://www.povray.org/ and http://opencv.org/ respectively.
MeshLab will be used in order to view the resulting 3D model. It can be downloaded from http://meshlab.sourceforge.net/.

No special hardware resources are required unless the extension to provide support for camera interaction is undertaken. The project is aimed at home users so it should work with any digital camera. I will be using my own camera.

I am also planning to use my own machine, doing regular backups to github and to PWF. I can resume the development on any of the machines in the Computer Lab should my machine fail.

Substance and Structure of the Project

To ensure the successful completion of the project, the workload has been divided into the following manageable chunks:

1. **Research:** Get familiar with epipolar geometry and with the algorithms mentioned in this section.

2. **Module design:** Split the project into modules, establish how they interact, construct the corresponding UML diagrams.

3. **Test harness:** Implement a test harness using POV-Ray. Given a 3D model, the test harness will generate the image sequences to be inputted into the partial 3D model generator by taking snapshots of the rendered model from each of the pre-established locations in turn. This module will also be used in order to evaluate the success of the project by adding measured error to the snapshots and measuring the accuracy of the outputted 3D model. Error can be added by rendering the object at a distance ϵ from the specified location, or by adding blur, for example.

4. **First prototype:** Implement triangularization in order to get the first prototype of a partial 3D constructor.

5. **Refinement of the partial 3D generator:** This step consists of iteratively improving the current prototype in order to get a more complete and more accurate 3D model. Some of the possible prototypes are listed below. They need not be all implemented if the current results are satisfactory. If multiple satisfactory prototypes are implemented they will be evaluated

individually and included in the final solution. The users can then specify which one they would like to use.

- Prototype #2: Implement dense stereo matching [1][2] to estimate the depth of each point from a view point. Convert the results into a depth map [2]. The depth map can be directly used as a 3D model by displaying darker shades for more distant points. However, this cannot be exported into a common 3D geometry format so it should be used in conjunction with the algorithms described in prototype #3 to provide a complete solution. I consider this partial solution to be a prototype by itself because the ability to visualize the output enables me to evaluate the results of this intermediate step. This might prove useful in order to discuss whether using meshes was the right choice since many other approaches could be taken instead.
- Prototype #3: Construct the corresponding 3D mesh using depth map constructed in prototype #2. A possible approach consists of overlying a 2D triangular mesh on top of one of the images and then wrapping the vertices of the triangles to correspond to the values in the depth map [1].
- Prototype #4: Use a volumetric depth map integration approach [1].
- Prototype #5: Implement a Kalman filter [1].

6. **Evaluation:** Evaluate the results by comparing the output of the 3D model generator to the model used to generate the image sequences. Different metrics can be used in order to compare the two models: difference in volume or the root-mean-square difference between vertex co-ordinates, for example.

   The sensitivity of the 3D model generator will be evaluated by adding measured error to the input (using the test harness) and comparing the output to that obtained from the equivalent error-free images.

7. **Write the dissertation**

Success Criteria

The following must necessarily be achieved for the project to be considered a success:

- A tool for constructing 3D models from a sequence of digital snapshots produced as described in the introduction has been implemented. The tool will be evaluated and shown to work by applying it to an input consisting of sequential snapshots of a cube.

The tool will be evaluated by considering only the traits of the object which can be deduced from the views provided. Therefore it is expected to provide a partial or inaccurate solution if the object contains a dent on the side facing the table, for example, or if it contains a sharp concavity which can only be completely determined from a certain angle which is not provided as input.

Possible Extensions

- Add a camera module in order to provide support for real camera input (instead of using digital snapshots generated using a ray tracer).
- Add shadowing to the outputted 3D mesh.
- Add texture to the outputted 3D mesh.
- Implement structure-from-motion and self-calibration: this would allow the users to input virtually any set of pictures representing an object as long as any consecutive pictures have a large overlap (i.e.: no need to pre-establish the camera pose for each photo in the sequence).

Timetable

Michaelmas Term Weeks 1 - 2 (10/10 to 23/10)

Read relevant literature and plan the project.
Milestone: Submit project proposal on 24/10.

Michaelmas Term Weeks 3 - 4 (24/10 to 06/11)

Further research: gain a deeper understanding of triangularization, epipolar geometry, stereo matching, constructing depth maps and constructing 3D meshes from a depth map. If the time allows it also read about volumetric reconstruction and Kalman filters.
Milestone: Finish writing an outline of the key concepts involved in the project on 07/11 and add it to the dissertation draft. Have the draft reviewed by the supervisor to check that this milestone was met.

Michaelmas Term Weeks 5 - 6 (07/11 - 20/11)

- Plan and design the structure of the project: split into modules and establish how they interact.

- Get familiarized with the open source libraries used: play with the feature matching methods from OpenCV and with POV-Ray. Write a test program using OpenCV to confirm that I can use the feature matching methods properly.

Milestone: Finish constructing UML diagrams on 21/11 and add them to the dissertation draft. Have the draft reviewed by the supervisor to check that this milestone was met. Finish writing the test program.

Michaelmas Term Weeks 7 - 8 (21/11 - 04/12)

Catch-up time. Use this to ensure that the previous milestones are met. If time allows it, start implementing the test harness.

Vacation Weeks 1 - 2 (05/12 - 18/12)

Implement the test harness.
Milestone: Finish implementing the test harness on 19/12.

Vacation Weeks 3 - 4 (19/12 - 01/01)

Implement the first prototype.
Milestone: Have a basic but working implementation by 02/01.

Vacation Weeks 5 - 6 (02/01 - 15/01)

Catch-up time. This slot is here to ensure that I have a working (although basic) implementation by 16/01. If time allows it, start working on the second prototype.

Lent Term Weeks 1 - 2 (16/01 - 29/01)

Refine the solution by implementing a second prototype.
Milestone: Finish implementing a second prototype by 30/01.

Lent Term Weeks 3 - 4 (30/01 - 12/02)

Refine the solution. Implement as many of the prototypes defined in the *Structure and Substance of the Project* section as considered necessary.
Milestone: Finish implementation of the core project by 13/02.

Lent Term Weeks 5 - 6 (13/02 - 26/02)

Catch-up time. This is a good period to start working on the extensions if time allows it.
Milestone: Have a complete and working implementation by 27/02.

Lent Term Weeks 7 - 8 (27/02 - 12/03)

Evaluate the project. Construct performance graphs.
Milestone: Finish writing an outline of the evaluation section (including evaluation graphs) by 13/03 and add it to the dissertation draft. Have it reviewed by supervisor.

Vacation Weeks 1 - 2 (13/03 - 26/03)

- Start working on the dissertation: write the Preparation chapter and most of the Implementation chapter.
- *optional:* Attempt to implement one of the extensions.

Milestone: Have a complete Preparation chapter. Send the dissertation draft for review.

Vacation Weeks 3 - 4 (27/03 - 09/04)

Write the dissertation.
Milestone: Have an imperfect but submittable dissertation draft by 10/04.

Vacation Weeks 5 - 6 (10/04 - 23/04)

Refine the dissertation. Progressively do modifications after getting feedback from supervisor.
Milestone: Finish writing and proof-reading the dissertation by 24/04.

Easter Term Weeks 1 - 2 (24/04 - 07/05)

Safety slot. Print and bind the dissertation.
Milestone: Submit the dissertation by 08/05

References

[1]M. Pollefeys, *3D from Image Sequences: Calibration, Motion and Shape Recovery*, Mathematical Models of Computer Vision: The Handbook, N. Paragios, Y. Chen, O. Faugeras, Springer, 2005.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer Verlag 2011.