Computer Science Project Proposal

Constructing 3D models from image sequences

I. G. Sorescu, Newnham College

Originator: I. G. Sorescu

13^{th} October 2014

Project Supervisor: K. Palyutina

Director of Studies: Dr J. K. Fawcett

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

Introduction and Description of the Work

The aim of this project is to extract a 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).
The main focus of the project is to obtain a wireframe model, leaving shading and texture mapping as a possible extension.
The inputted set of pictures must be obtained using the following technique [add an image to represent this]: *D*efine an "origin" on a perfectly horizontal table and place the camera there facing a white background. Place an object of max height X at a distance Y from the origin. Take the first picture. Rotate the object 30 degrees (counter clockwise as seen from camera) and take the second picture ... until 12 pictures are taken. Some other details to work out such as focal distance, skew coefficients, principal point, distortions. Update pictures in the order in which they were taken. Assume well-lit room and no major light differences between pictures in the same set.
Extension: implement self-calibration such that in theory the project will work on any set of pictures as long as large overlap and completely determines the object.

Starting Point

Some experience with C/C++. IB maths and graphics.

Resources Required

Camera(add detailed description - probably anything with manual settings will do), my own computer + backup plans. Open source libraries I might use: openCV, openGL (for test harness), MeshLab (to view the output), a camera calibration tool (such as for the extension).

Work to be done

For core bits (strictly sufficient to satisfy the success criterion):

- write test harness (to create digital snapshots of a polyhedron and test the 3D models on calibrated, error-free input) - this module will be later used to evaluate the results by adding a measured error to the inputs and comparing the outputs
- detect edges in the 12 pictures
- compute a dense set of correspondences between neighbouring images *s*tereo rectification, stereo matching, dense depth map
- reconstructing the 3D object shape *o*verlying a 2D triangular mesh on top of one of the images to bild corresponding 3D mesh by wrapping vertices of the triangles in 3D space by using depth maps OR volumetric depth map integration, Kalman filters

Extensions:

- self-calibration: computing the geometric relation between neighbouring images, estimating the motion and calibration of the camera
- *s*earch for texture mapping techniques

Success Criterion

A system that can successfully construct the wireframe of a fairly regular object of maximum height X [more specific?] from pictures taken as described in introduction.
*T*he input images used to determine the success of the project will be considered

to be error-free so probably computer-generated.

Might be helpful to compare results to those generated by ARC3D.

Possible Extensions

- self-calibration
- add shadowing to model
- add texture to model

Timetable

Planned starting date is 09/10/2014.

1. **Slot 0: 9^{th} Oct - 24^{th} Oct**

   - Read relevant literature and plan the project.

   Milestone: Submit proposal

2. **Slot 1: 25^{th} Oct - 14^{th} Nov**

   - Further research: gain a deeper understanding of the techniques involved in edge detection and 3D modelling

   Milestone: Have a clear understanding of the techniques needed to complete the project

3. **Slot 2: 15^{th} Nov - 28^{th} Nov**

   - Start implementation: implement test harness
   - design a few basic unit tests for the edge detector, dense set generator and wireframe generator.

   Milestone: finish implementing the test harness

4. **Slot 3: 28^{th} Nov - 5^{th} Dec**

   - Implement the edge-detector

   Milestone: Pass the unit tests for the edge-detection module

5. **Slot 4: 6^{th} Dec - 26^{th} Dec**

- Implement dense set generator

Milestone: Pass the unit tests for the dense set generation module

6. **Slot 5: 27^{th} Dec - 16^{th} Jan**

   - Implement the wireframe generator

   Milestone: Finish implementation, Pass the unit tests for the wireframe generator

7. **Slot 6: 17^{th} Jan - 30^{th} Jan**

   - Buffer time: catch up or start doing the extensions
   - Write progress report

   Milestone: Submit progress report

8. **Slot 7: 31^{st} Jan - 13 th Feb**

   - Further tests

   Milestone: Finish writing integration tests and more in-depth unit tests

9. **Slot 8: 13^{st} Feb - 27 th Feb**

   - Debug

   Milestone: Pass all of the tests

10. **Slot 9: 27^{st} Feb - 13^{th} Mar**

    - Catch-up time or extensions

    Milestone: Pass all of the tests

11. **Slot 10: 14^{th} Mar - 3^{rd} Apr**

    - Analysis

    Milestone: Finish doing the evaluation graphs

12. **Slot 11: 4^{th} Apr - 17^{th} Apr**

    - Plan and start writing the dissertation

    Milestone: Start writing the dissertation

13. **Slot 12: 18^{th} Apr - 8^{th} May**

- Write the dissertation

Milestone: Finish writing dissertation

14. **Slot 13: 9^{th}May - 15^{th}May**

    - Safety slot

    Milestone: Submit dissertation