Computer Science Project Proposal

Constructing 3D models from image sequences

I. G. Sorescu, Newnham College

Originator: I. G. Sorescu

17th 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 from calibrated images. A digital renderer will be used in order to provide calibrated, error-free input. The project is designed to be later extended to support real camera input. (If real camera input is supported:) 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. 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.
The digital renderer will imitate this technique and output the equivalent 12 pictures.

Starting Point

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

Resources Required

my own computer + backup plans.
Open source libraries I might use: OpenCV, OpenGL (for the renderer), MeshLab (to view the output)
For the extension: a digital camera. Specific details to be added later but anything which supports manual settings should do.

Work to be done

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

- write digital renderer to provide calibrated input for the modules to follow - this module will be later used to evaluate the results by adding a measured error to the inputs and comparing the outputs
- computing the geometric relation between neighbouring images
- compute a dense set of correspondences between neighbouring images $s$tereo rectification, stereo matching, dense depth map
- reconstructing the 3D object shape (a wireframe) $o$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

For the extensions:

- add a calibration module in order to support for real camera input (as opposed to digital snapshots generated by the renderer).

Success Criterion

The project is considered to be a success if it can correctly reproduce the shape of an object from 12 error-free digital snapshots produced as specified in the introduction.

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

Possible Extensions

- support for real camera input
- add shadowing to model

- add texture to model
- self-calibration: this would allow the users to input virtually any set of pictures representing an object as long as they have a large overlap and completely determine the object (i.e.: no need to take exactly 12 pictures 30 degrees apart).

Timetable

Planned starting date is 09/10/2014.

1. **Slot 0: 9th Oct - 24th Oct**

   - Read relevant literature and plan the project.

   Milestone: Submit proposal

2. **Slot 1: 25th Oct - 14th 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: 15th Nov - 28th Nov**

   - Start implementation: implement the digital renderer
   - Use the renderer to design a few basic unit tests for the edge detector, dense set generator and wireframe generator.

   Milestone: finish implementing the digital renderer and the test harness

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

   - Implement the wireframe generator. This should be implemented first because it is the riskiest module of the project.

   Milestone: Pass the unit tests for the wireframe generator

5. **Slot 4: 6th Dec - 26th Dec**

   - Implement dense set generator

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

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

    - Implement the edge-detector

    Milestone: Finish implementation, Pass the unit tests for the edge-detection module

7. **Slot 6: 17th Jan - 30th Jan**

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

    Milestone: Submit progress report

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

    - Further tests

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

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

    - Debug

    Milestone: Pass all of the tests

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

    - Catch-up time or extensions

    Milestone: Pass all of the tests

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

    - Analysis

    Milestone: Finish doing the evaluation graphs

12. **Slot 11: 4th Apr - 17th Apr**

    - Plan and start writing the dissertation

    Milestone: Write the main parts of the dissertation

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

    - Write the dissertation

    Milestone: Finish writing dissertation

14. **Slot 13: 9thMay - 15thMay**

    - Safety slot

    Milestone: Submit dissertation