Malte Schwarzkopf
St John's College
ms705

Computer Science Tripos Part II Individual Project Proposal

# Proteus – An interactive 3D model acquisitor

October 23, 2008

**Project Originator:** Christian Richardt

**Resources Required:** See attached Project Resource Form

**Project Supervisor:** Christian Richardt

**Signature:**

**Director of Studies:** Dr Robert Mullins

**Signature:**

**Overseers:** Dr Frank Stajano and Dr David Greaves

**Signatures:**

# Introduction and Description of the Work

In recent years, consumer-grade digital video capture equipment has become much more affordable, whilst at the same time the computing power available at low cost increased. On a similar ground, 3D computer graphics hardware has become a lot faster and more affordable. These factors have led to these particular uses of digital technology becoming more ubiquitous. Fast broadband connections and the emergent “web 2.0” phenomenon that gave rise to platforms such as YouTube have similarly contributed to giving home-made videos a much more prominent role in our everyday lives and culture.

These factors have led to an increasing research interest in combining the classical 2D image and video processing technology with three-dimensional computer graphics. Structure-from-motion and the idea of acquiring 3D models from two-dimensional image material is a specific branch of a broader research area, and one that has seen some significant steps forward recently, yet still seems to bear a lot of promise and potential.

A lot of research effort has been put into analysing video data in order to obtain information about objects for guidance systems or recognition problems, however this has usually been limited to either a poor visual quality or utilised very specific hardware setups and highly specialised equipment. Little consideration has been given to the idea of providing home users, with often limited computing experience, with the necessary tools to transform videos into 3D world models, using the consumer-grade equipment available to them.

The aim of this project is to implement a simple structure-from-motion system that can generate a 3D model of an object in a scene from a video, using inputs interactively provided by the user to assist the model extraction.

It is well known that the human visual system is well-suited to recognising shapes and objects in an image, and can do so more efficiently than current image processing technology can. In order to utilise this in the context of a 3D model acquisitor that uses video as its input data, an interface that allows the user to “annotate” a video frame with object information, hence pointing out distinctive features in the object such as corners, edges and shapes. In order to assist the user in this process and provide some additional guidance, a ”snapping” concept, which snaps lines to edges in the video, will be implemented.

After the first key frames have been annotated, the software will attempt to use the data provided by the user to estimate how the perspective has changed relative to another video frame. Features are then inpterpolated between frames, and the user is give an opportunity for corrective measures in case the interpolation has produced an inaccurate result.

Ultimately, a completely annotated sequence of video frames should have been obtained. This is then used to extract the 3D structure of the object in the video. In the implementation planned for this project, some information will already be provided by the interactive annotations on the video, so that the step of feature identifcation and matching can be simplified significantly compared to fully automatic systems.

# Resources Required

In order to model a real-world use case as accurately as possible, the project will use videos taken using a hand-held digital camera as sample input material. Please see the attached project resource form for details on this.

Apart from the camera, there are no special requirements. I am planning to undertake development on my personal machines, doing regular automated and manual backups to off-site locations such as the PWF and the servers provided by the SRCF.

# Starting Point

In proposing this project, I am planning to build on the following resources:

- **Internships at Broadcom Europe in 2007 and 2008** – I worked as a summer intern in the 3D graphics group of the Mobile Multimedia department at Broadcom Europe Ltd. for two summers. My work was primarily related to 3D hardware, but I was exposed to the basic principles OpenGL and 3D computer graphics.

- **Previous programming experience** – I have a good working knowledge of C and have in the past implemented projects of significant size in C. I am familiar with C++, but haven't used it much beyond the Part IB lecture course.

# Substance and Structure of the Project

In order to ensure successful completion of the project, it has to be divided into manageable chunks of work, and there have to be various levels of fallback options to compensate for possible difficulties and issues that might delay the implementation.

For this project, there are three central stages that each consist of a number of sub-tasks that need to be completed:

1. *Interactive video annotation toolkit:* This step will provide the user with a very simple GUI framework that can be used to "annotate" key frames in a video with feature information. There is a obvious and direct relation between the number and distance of key frames, the accuracy of the annotations, and the quality of the reconstruction of the object in a 3D model. This stage of the project involves the following tasks:

    - Creating a video annotation framework: This will be a simple C/C++ implementation of an image viewer that displays individual frames of a video and that has got an annotation framework on top, allowing for a wireframe to be drawn onto the image. For video decoding, the *libavcodec* library, which is part of the open source *ffmpeg* cross-platform video library, is intended to be used; the user interface toolkit to be used will either be Gtk+ or Qt, both of which are free and cross-platform.

- Implementing interpolation: For the frames in between key frames, the annotations from previous keyframe are copied and an estimation of their possible relative displacement is made. The user will be allowed to edit them (move, add and remove vertices).
- Implement "snapping": Inspired by the concept outlined in [**?**], snapping of feature points to to detected edges or feature outlines in the image as well as other feature points will be implemented. This will mean that lines drawn on the model by the user will "snap" to the most likely location, thereby correcting for small inaccuracies between user input and image data, allowing for a rapid, yet accurate annotation process.

2. *3D model generation:* From the annotated video data, proceed through extracting the relative camera motion between frames, and hence the camera perspectives in the individual frames, using principles of epipolar geometry (projective reconstruction) on to regenerating a metric representation of the scene geometry and rectification of the different views. Finally, generate depth maps and from those, generate spatial surfaces that can then be used to build the 3D model. Display the final model using an OpenGL canvas. An approach to this is presented in [**?**], although the assumption here is that all information about the object has to be extracted automatically and autonomously from the video data by the system.

3. *Texture extraction and projection:* Implement a simple texture extraction scheme whereby the "best" frame is determined (i.e. the one where most of a plane/polygon is visible, ideally a direct front-facing shot) and extract the bitmap texture data from the video for this frame. Apply to the generated output polygon. Note that this requires working out what the OpenGL varyings for this texture mapping should be.

## Possible Extensions

1. *Scene background extraction:* Extract the (static) scene background and project it onto a background plane in the generated 3D scene.

2. *Export of 3D model data into some well-supported format:* Implement support for exporting the generated structures to a free OpenGL model format so that it can be used with any application understanding this format.

# Success Criteria

The primary success criteria for this project are detailed in the following. These criteria are essential for the project to be considered a success.

1. A video annotation framework that supports annotation of video frames and implements "snapping" of feature points selected, thereby guiding the user in annotating the video, has been implemented.
2. An interactive structure-from-motion implementation that uses the feature points supplied by interactive annotation has been implemented. It is shown to be working by applying it to a 360° shot of a cube or some other distinctive geometric shape.
3. Textures for the faces in the extracted 3D model are obtained from the input image data using per-polygon texture extraction. It is shown to be working by applying the method to a textured cube.

In terms of possible extensions of the project, implementing one or more of the following would increase the success of the project, but this is not essential.

1. The final 3D model generated can be exported into a well-supported model format so that it can be used in real-world applications (such as Google Earth, Second Live or a 3D computer game).
2. Extraction of the invariant scene background and projection onto a bounding box to create the illusion of the 3D object being in the surroundings it was captured in.

# Timetable and Milestones

There are four and a half months of time available to do this project. I am aiming to finish the implementation work by the middle of March, i.e. the beginning of the Easter vacation.

I estimate that the work will be partitioned into 3 week slots as follows:

### Slot 0: October, 1$^{st}$ – October, 17$^{th}$

- Research, familiarisation with literature on the subject and analysis of different existing concepts.
- Work out an approach to do the feature data to 3D conversion.
- Investigate libraries and write test programs to utilise them and for familiarisation.

**Milestone:** Write project proposal.

**Slot 1: October, 18$^{th}$ – November, 7$^{th}$**

- More research and reading. Familiarise with ideas of epipolar geometry and the mathematical foundations of the processes of projective reconstruction, self-calibration, rectification and model generation.
- Start on implementing the annotation framework. Have facility to load and decode video and to display individual frames and draw annotations onto them.

**Slot 2: November, 8$^{th}$ – November, 28$^{th}$**

First part of the core implementation phase.

- Get annotation framework to work and have per-frame annotations working together with techniques to interpolate the feature annotations between frames.
- Implement "magnetic snapping" of lines/features to regions in the image.

**Milestone:** Have a working annotation interface, including video decoding and automatic "snapping" of feature points at the end of this time slot.

**Slot 3: November, 29$^{th}$ – December, 19$^{th}$**

Second part of the core implementation phase.

- Implement first iteration of the core model acquisition code: Projective reconstruction from data provided by the annotations, self-calibration of camera parameters, metric reconstruction.
- I will also need to write some auxiliary code at this point in order to be able to test and debug the different steps of the model acquisition process whilst not yet having a final output; this might make development more tedious.

**Slot 4: December, 20$^{th}$ – January, 9$^{th}$**

Third part of the core implementation phase.

- Add final stages of model acquisition code: Depth maps, surface generation and model building. Tackle any issues with the earlier stages.
- Implement generation of the OpenGL data structures used for models and implementation of a display canvas.
- Implement simple texture extraction by using the best possible texture for a face found in a frame and applying it to the face in the final model.

**Milestone:** Working implementation of model acquisition and -generation for simple objects.

**Slot 5: January, 9$^{th}$ – January, 30$^{th}$**

Progress report, implementation of extensions and catchup time.

- Finish texture extraction implementation, trying to refine the quality of the extracted textures and their projection onto surfaces.
- Start implementing one or several of the optional extensions to the project.
- Write progress report and hand it in.

**Milestone:** Present progress, have a fully texture simple geometric object being recreated using interactive annotation by the end of the time slot.

**Slot 6: January, 31$^{st}$ – Feburary, 20$^{th}$**

Further extension implementation and catchup time.

- Continue implementing optional extensions to the project.
- Investigate further extension possibilities that might have arisen during development and possibly implement extra features.

**Milestone:** Working implementation of one or several extensions, depending on the choice made.

**Slot 7: Feburary, 21$^{st}$ – March, 13$^{th}$**

Catchup time. Start writing dissertation.

- Buffer time to catch up any delay incurred at previous stages.
- Fix remaining bugs and outstanding issues and optimise some parts of the code. Take measurements and benchmarks and analyse the results for evaluation.
- Start writing the dissertation by producing an outline and writing the introductory sections.

**Slot 8: March, 14$^{th}$ – April, 3$^{rd}$**

Write dissertation up to draft standard.

- Write the main parts of the dissertation. Generate figures, intergrate results and evaluation.
- Typeset the dissertation and produce a draft to be handed in.

**Milestone:** Dissertation draft finished by the end of the timeslot.

**Slot 9: April, 4th – April, 24th**

Finish dissertation.

- Finish off the dissertation. Add more benchmarking and evaluation results.
- Address any formatting and typesetting-related issues.
- Incorporate comments on draft received from supervisor and proofreaders.

**Slot 10: April, 25th – May, 15th**

Buffer time, dissertation deadline at the end of the timeslot.

- Address any remaining issues with the dissertation and hand it in when done.
- Use the remaining time for revision for the exams.

# References

[1] A. Agarwala, "SnakeToonz: A Semi-Automatic Approach to Creating Cel Animation from Video", Starlab, `http://agarwala.org/Pages/snaketoonz.html`.

[2] M. Pollefeys et al., "Hand-held acquisition of 3D models with a video camera", Proc. 3DIM'99, IEEE Computer Society Press, pp.14-23, 1999.

[3] R. Hess, SIFT implementation in C, `http://web.engr.oregonstate.edu/~hess/index.html#[[SIFT%20Feature%20Detector]]`.