24783 Special Topics : Advanced Engineering Computation Project Specification

Important Dates

Team Preference	2/4 (Mon) by E-Mail
Project Proposal	2/11 (Mon) @ SVN
Finalizing the Project Topic	2/22 (Fri)
Unit Technologies Identification	3/4 (Mon) @ SVN
Mid-Term Presentation of Project Goal and Unit Technologies	3/18 (Mon) @ SVN and in class
Alpha Version	4/1 (Mon) @ SVN
Final-Goal Calibration Presentation	4/15 (Mon) @ SVN and in class
Beta version	4/22 (Mon) @ SVN
Final Presentation	5/1 (Wed) in class

Weight: **28%**

Project Goal

Study and implement a state-of-the art computational method that has been published as a recent research paper or of equivalent complexity in C++. You may use external libraries, however, it should not provide majority of the functionality. You can look at some sample codes, but you must understand and implement the method within your team.

Option A:

Study and implement a computational method published in SIGGRAPH, EUROGRAPHICS, CAD Journal, Solid and Physical Modeling, International Meshing Roundtable, or Geometric Modeling and Processing, between 1998 and present. Some suggested papers are in the Appendix of this document.

It is a very good experience, however, it is also very challenging. It would be great if you are able to implement all the feature of the paper you pick end-to-end, but it may be too hard to achieve in the given time frame. Therefore, if you can explain the reason, part of the paper can be done manually or by an external program.

For example, you have two separate programs A and B. Program A does the first part of the method and write out a file. Then the file is processed by an external program. The output from the external program is finally given to Program B to get to the final result.

Or, the method may rely on a specific algorithm for reducing the computational complexity. However, due to the difficulty of the algorithm, you may only be able to implement $O(N^2)$ or even $O(N^3)$ algorithm. In that situation, it is ok to reduce the scale of the input.

Another situation can be, you start with a research paper, which turns out to be too hard to fully implement. But you find another paper that is the basis of the paper, which is doable. Then, you can switch to that paper.

You will have a chance to calibrate your goal. As long as you have a reasonable explanation, there is no point deduction for doing it.

Note: Many of these computer-graphics papers use external programs to create a good-looking rendering images, but it does not mean your program also needs to create a photo-realistic rendering. For example, if you choose a paper about particle-based simulation, and the paper shows sample images rendered by an external program or a technique described in other papers, visualization of particle motion is good enough, and you don't have to show that quality images.

Option B:

Write an application package for Engineering, Education, or Entertainment. The program must run on at least three of macOS, Linux, Windows (Win32), Universal Windows Platform, iOS, and Android.

If you are primarily developing on macOS, you can test your program on Linux and Windows with VirtualBox. You can test your program on Linux with VirtualBox, plus can natively test on Universal Windows Platform.

Developing for iOS requires a Mac, and the primary problem is that it is very difficult to import or export a file to and from the outside of the device. If you want to read/write to/from the outside of the device, you will need platform-dependent coding. Due to this difficulty, it is ok to make iOS version as a demo version, which only handles pre-installed data files.

At this point, the graphics framework used in this course does not support Android. If you want to include Android as one of the three platforms, you will need to study a third-party toolkit, such as Qt, yourself.

The program must be published to the public, i.e., all files must be open source or copyright free. TAs will prepare a HTML file that you can fill for publishing your program. Once the Alpha version is ready, your program will be posted on the course web site.

Customer test and survey must be conducted. Once Alpha version is ready, your program must be tested by at least 15 customers. At least 8 of them must be someone who is NOT taking this course. Collect opinions from the customers, and take suggestions in the final product. The summary of the customer survey and how you improved your program based on the survey must be included in the final presentation.

Team Member and Team Name Preference

You may express your preference regarding whom you would like to team up with. Each team must consist of minimum three maximum five students. The group leader must send an E-Mail to Teaching Assistants with:

- Team name
- The list of members, and
- Team leader name.

by the deadline.

The TAs will create a SVN directory for each team where you can share the source code, upload documents, and data files.

Project Proposal

Create an SVN sub-directory called "proposal", and upload a .PDF file describing the top three choices of the project option. If you go with Option A, and if you are able to download a PDF of the paper, also upload the PDF files in the same directory.

Finalizing the Project Topic

Make an appointment with the instructor and TAs to discuss and finalize the project option by this date.

Identification of Required Unit Technologies

Identify what data structures and algorithms are required for implementing the program. Write a 1-page summary that includes a list of data structures and algorithms and the purpose for which each of them is used in the project. The document also needs to include who is in charge of implementing which data structures and/or algorithms. Also, if you use external libraries for these data structures and algorithms, identify what libraries is used for which data structures and algorithms and where you can download and install.

Create a sub directory called "unit_technologies" in the team SVN directory (by "svn mkdir" command, or create a directory and then use "svn add" command), and submit the document in this directory in the PDF format.

Mid-Term Presentation of Project Goal and Unit Technologies

Individual members must start working on the unit code. Test as good as you can so that the assembly will go smoothly.

You need to make a short presentation of project goal and identified unit technologies in class. Each team has 5 minutes. The presentation package must be uploaded to the "midterm" sub-directory of the team SVN directory.

Alpha version

Assemble individual components into a functional program. Test your program with very simple input and verify the functionality. The purpose of the Alpha version is to detect and fix basic problems, therefore it does not have to be fully functional. What is important is to have components integrated in a program that is ready for testing.

Create a sub-directory called "alpha_version" under the team SVN directory, and submit CMakeLists.txt, all .CPP and .H files needed for building the program. You may submit test data files that you used for initial testing in the same directory (create sub-directories when necessary.)

If you use an external library, describe where you can acquire the library and how to install the library in a PDF file called "external.pdf" and submit in the alpha_version directory.

If you go with Option B: Within one week from the alpha version, start the user test. Distribute your program to your potential customers, and collect opinions. You need to come up with questionnaire for the user survey. Your program must be tested by at least 15 users. 8 of them must be from the outside of the class.

Final Goal Calibration Presentation

By this time, you have a good idea of what is doable and what is not from the initial goal of the project.

Under-promise and over-delivery is the key to the customer satisfaction. However, the opposite happens very often. Your CEO make an over-promise, and actual delivery will be less than that.

As a development team, you need to explain the situation, and convince your bosses that it is reasonable.

You need to make a 5-minute presentation explaining what the final product will look like, and give a reasonable explanation if some part of the initial goal is not deliverable. The presentation package must be uploaded to the "final_goal_calibration" sub-directory of the team SVN directory.

Final Presentation

The final presentation package (including the presentation slides) to the SVN server by 4/24 (Sun) 23:59. Create a sub-directory called "final_presentation" in your team SVN directory and submit presentation files there.

Also create a sub-directory called "final_version" in your team SVN directory and submit all files, except external libraries, required for building the program (including CMakeLists.txt, .CPP, and .H files.)

Prepare presentation as if you were presenting a paper in a research conference. If your team go with Option A, clearly cite the paper that your team implemented on the first page of the presentation.

If your team go with Option B, include customer-survey result, and what changes you made based on the survey.

Each team will be asked to make a product presentation on 5/1. Each presentation should be 15 minutes in length followed by 1-2 min Q&A session.

The presentation must include live demo of your program. (If your program takes long time to make an output, it does not need to be fully live.)

Set up your laptop so that you can run your program. Test well before the presentation. Also if you are going to present on a Mac, make sure to bring your mini display-port to VGA converter.

Appendix: Examples of the papers for Option A.

NOTE: Those are examples. You may be able to find a more interesting paper by searching for the papers that are citing these suggested papers.

Emil Pruan, Hugues Hoppe, and Adam Finkelstein, *Robust Mesh Watermarking*, Proceedings of SIGGRAPH 1999, pp. 49-56, 1999 [Download Link] (You need to be on CMU network to download.)

Tao Ju, Scott Schaefer, and Joe Warren, *Mean Value Coordinates for Closed Triangular Meshes*, Proceedings of SIGGRAPH 2005, pp. 561-566, 2005 [Download Link] (You need to be on CMU network to download.)

Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee, *Skeleton Extraction by Mesh Contraction*, Proceedings of SIGGRAPH 2008, [Download Link] (You need to be on CMU network to download.)

David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun, *Variational Shape Approximation*, Proceedings of SIGGRAPH 2004, pp. 905-914, 2004 [Download Link] (You need to be on CMU network to download).

Antonio Crlos de Oliveira Miranda and Luiz Fernando Martha, *Quadrilateral Mesh Generation using Hierarchical Templates* ,Proceedings of 21st International Meshing Roundtable 2012 [Download Link]

Bertrand Pellenard, Jean-Marie Morvan, and Pierre Alliez, *Anisotropic Rectangular Metric for Polygonal Surface Remeshing*, Proceedings of 21st International Meshing Roundtable, 2012 [Download link]

John Edwards, Wenping Wang, and Chandrajit Bajaj, *Surface segmentation for improved remeshing*, Proceedings of 21st International Meshing Roundtable, 2012 [Download link]

Daniel W. Zaide and Carl F. Ollivier-Gooch , Inserting a Curve into an Existing Two Dimensional Unstructured Mesh, Proceedings of 22st International Meshing Roundtable, 2013 [Download link]

Mark W. Dewey, Steven E. Benzley, Jason F. Shepherd, Matthew L. Staten, *Automated Quadrilateral Coarsening by Ring Collapse*, Proceedings of 17th International Meshing Roundtable, 2008 [Download link]

Julien Villard and Houman Borouchaki, *Adaptive Meshing for Cloth Animation*, Proceedings of 11th International Meshing Roundtable, 2002 [Download Link]

Jiangtao Hu, Y. K. Lee, Ted Blacker and Jin Zhu, *Overlay Grid Based Geometry Cleanup*, Proceedings of 11th International Meshing Roundtable, 2002 [Download link]