Graphics and Game Technology

Lab Assignments

Getting Started

1 General remarks

Welcome to the exciting world of Computer Graphics! We hope that, through the lectures and the lab assignments, we will be able to stimulate your interest and get you acquainted with the theory, concepts and practice of this subject.

Before you begin with the lab assignments, there are a number of rules that we expect you to abide by:

- Check the schedules for the time and location of the lab sessions. Make sure that you visit the first lecture *and* the first lab session for more information and last-minute changes.
- You work on the assignments in pairs. Do not switch partner during the course. You are expected to contribute equally to each assignment. The assistant will check whether that has been the case. If you contributed equally, you will receive the same grade. If not, a correction will be made.
- Attend all lab sessions. An assistant will be present that can answer questions or help you in case of problems. **NOTE: This is the *only* method of support that will be provided. The assistant will NOT answer questions by email.**
- Again: attend all lab sessions. The assistant will grade your assignment *on the spot*. If you're not present, you will not be graded.
- The deadline for each assignment is set to the last lab session of the week. Check the schedules to see which day that is! Do not wait until the very last minute to get your assignment graded.
- Once graded, the assignment must also be submitted through Blackboard. Do so immediately after it has been graded. The deadline for submission is set for the following Monday 9:00 AM. If you don't submit your assignment, your grade will be nullified. Each pair only need to submit the assignment once. Make sure *both* your names and studentnumbers are added to the marked comment sections of the code.

- All assignments for this course must be completed in the C programming language and must work with the dual-boot computers running Linux. On these, you need your “UvAnetID” to log in. If you don’t have that, contact the Education Service Centre (ESC) before the course starts.

- All assignments come with a framework for which the software dependencies are pre-installed on the Linux computers scheduled for this course. If you use your own laptop, it is *your* responsibility to resolve these software dependencies.

- In the first week; check that you can log in to the Linux systems and compile and run the first framework program. If you cannot get the framework running, contact the assistant on the very first lab session.

- All assignments will be posted on Blackboard. If you cannot access the section for this course, you must enroll first through SIS.

2 Framework

Each assignment comes with a PDF file containing the description of the assignment and a framework package (.tar.gz or .tgz) that contains files written in C and a `Makefile` that generates a running executable on the Linux computers.

The framework should compile and run without problems. It just won’t do a lot because usually one of the source files contains one or more empty functions that you must complete. The assignment description provides details on what to do.

2.1 Unpacking the assignment framework

Download the assignment package from Blackboard and unpack it (enter the commands that are preceded by a `$`, without entering the `$`):

```
$ mkdir cg-pract
$ cd cg-pract
$ tar zxvf ../framework2.tgz
framework2/
framework2/normals.c
framework2/Makefile
framework2/main.c
framework2/normals.h
framework2/polys.c
framework2/polys.h
$ cd framework2
$ ls
main.c  Makefile  normals.c  normals.h  polys.c  polys.h
```

2.2 Building an executable

Now you can directly compile and run it, as follows:

```
$ make
gcc -c -g -O2 -std=c99 -Wall -Wextra -Werror-implicit-function-declaration
 -Wshadow -Wstrict-prototypes -pedantic-errors main.c
main.c: In function ’keyPressed’:
main.c:261: warning: unused parameter ’x’
main.c:261: warning: unused parameter ’y’
main.c: In function ’specialKeyPressed’:
main.c:272: warning: unused parameter ’key’
main.c:272: warning: unused parameter ’x’
main.c:272: warning: unused parameter ’y’
main.c: In function ’createSphere’:
main.c:88: warning: ’p.pts[3].x’ may be used uninitialized in this function
main.c:88: warning: ’p.pts[3].y’ may be used uninitialized in this function
main.c:88: warning: ’p.pts[3].z’ may be used uninitialized in this function
main.c:88: warning: ’p.pts[2].x’ may be used uninitialized in this function
main.c:88: warning: ’p.pts[2].y’ may be used uninitialized in this function
main.c:88: warning: ’p.pts[2].z’ may be used uninitialized in this function
gcc -c -g -O2 -std=c99 -Wall -Wextra -Werror-implicit-function-declaration
 -Wshadow -Wstrict-prototypes -pedantic-errors normals.c
normals.c: In function ’calcNormalsFlat’:
normals.c:21: warning: unused parameter ’list’
normals.c: In function ’calcNormalsGouraud’:
normals.c:26: warning: unused parameter ’list’
gcc -c -g -O2 -std=c99 -Wall -Wextra -Werror-implicit-function-declaration
 -Wshadow -Wstrict-prototypes -pedantic-errors polys.c
gcc -g -lGL -lglut -o main main.o normals.o polys.o
$ ./main
6 polygons to flat shade
648 polygons to gouraud shade
```

Don’t worry about the warnings, most of these should disappear as you start to fill in the required functions. When a window appears with a picture in it, everything should be fine. Sometimes the task description also shows one or more pictures of the expected output.

3 Submitting files

When your program is ready for grading, check that you have written your name and student number in the comments at the top of all file(s) that you have changed. When you submit your assignment, submit **the whole framework** of your project in `.tgz` format (see below). Avoid submitting executables or object code as well as editor backup files. You can use “`make clean`” to remove most of these files.

```
$ cd framework2
$ make clean
rm -f *.o
rm -f main
$ cd ..
$ tar zcvf framework2.tgz framework2/
framework2/
framework2/polys.c
framework2/normals.h
framework2/main.c
framework2/normals.c
framework2/polys.h
framework2/Makefile
$ ls
framework2  framework2.tgz
```

Submit the `.tgz` file through the Assignments section on Blackboard.

4 Grading

Each assignment contains a small section on how grades will be determined. In most cases you will therefore know in advance (before you ask to be graded) how many points you can get for each part of the assignment. In general an assignment will be split into several subtasks that will build upon one another. You will receive points for each of the subtasks individually. There are some general rules that apply to the grading of all assignments:

- Theoretically you can get up to 10 points if your program has all the requested properties.
- *The deadline for each assignment is the end of the last lab session in the week.* Exceeding the deadline means you lose two points. It is *your* responsibility to ask the assistant to grade your assignment *on time*. In particular; do not wait until the last moment because you then run the risk that yours will not be the only assignment that has to be graded.
- If your program is unclear or has a confusing layout (i.e. missing or wrong indents), you can lose up to 1.0 point.
- If your program doesn't compile, but we can recognize what it would have done if it compiled, you get the points according to the above list minus 3.0 points.
- If we have reason to believe that you cheated, i.e. copied and pasted from other students or from the internet, you receive, in the best case, no points at all.
- In individual cases there can be deviations from this list.

Good luck!