

# Assignment 1: Parametric Modelling

Due: 11:59pm

## Introduction

This assignment will introduce you to the basics of parametric modelling. Our parametric modelling framework is build from three (3) opensouce libraries: Cling (opensource C++ interperter), libfive (opensource parametric modelling library) and libigl (opensource geometry processing framework). Your task will be: - to install this framework (Part A) - implement three (3) commonm constructive solid geometry (CSG) operations (Part B) - use these operations to construct a parametric model - 3D print an instance of your model on one of the printers available in the DGP

## Part A: Setup Assignment Code Base

These instructions will walk you through setting up the Assignment 1 code base. This code is tested on Mac OSX and Ubuntu 18.04 no support for compilation on Windows will be offered.

### Initial Setup

OSX

1. Go to brew.sh
2. Follow the instructions to install homebrew

### Setup CMake, Boost and CGAL

OSX

1. In a terminal window run `brew install cmake`
2. Run 'brew install boost'
3. Run 'brew install cgal'

Ubuntu 18.04 LTS

1. In a terminal window run 'sudo apt-get install cmake'
2. Run 'sudo apt-get install cmake-curses-gui'
3. Run 'sudo apt-get install libboost-all-dev'
4. Run 'sudo apt-get install libcgal-dev'
5. Run 'sudo apt-get install libcgal-demo'

### Setup A1 Code Base and Cling on Ubuntu/OSX [3 marks]

Checkout Assignment Source Code

1. In a terminal window run 'git clone --recursive https://github.com/dilevin/DGPCompFab.git'

{SRC\_DIRECTORY}'. This will download the boiler plate code for all class assignments into the directory {SRC\_DIRECTORY}.

### Install Cling

1. In a terminal window and run ``cd {SRC_DIRECTORY}/ThirdParty`
2. Go to <https://root.cern.ch/cling-build-instructions>
3. Follow the **manual** instructions to install Cling. Use `./install` as the **Install Path**. If building fails, try executing CMake commands using ``sudo'`. The other common error (unreported by the build system) is running out of disk space. If you are using a virtual machine, make sure a large amount of space is allocated.

### Build Assignment 1

1. Run `'cd {SRC_DIRECTORY}/A1'`
2. Run `'mkdir ./build'`
3. Run `'cd ./build'`
4. Run `'cmake ..'` to build the project Makefile.
5. Run `'make -j8'`
6. This should build an executable named `./bin/CompFabA1`.

### Using the A1 Code Base

To run the example code 1. `cd ./bin` 2. Run `'CompFabA1'`

Running **CompFabA1** will open a blank LibIGL Viewer window. The A1 application reads in parametric modelling instructions, stored in C++ files, and compiles them on-the-fly to create and display the results. To test the compilation mechanism do the following


1. Click 'Load Script'
2. Choose {SRC\_DIRECTORY}/A1/scripts/testScript.cpp
3. Submit a screen grab of the test script output [3 marks]
4. Click 'Recompile', you should see the test shape below in the viewer.
5. Follow steps 1 to 3 to use our own parametric model script.

Libfive, the underlying parametric modelling engine, uses implicit shape representations which were discussed in class. Every shape is defined as a function  $f(x)$  which is  $< 0$  inside the shape and  $> 0$  outside the shape. See comments in testScript.cpp for more instructions.

 *Fig. 1: Expected output of all three CSG tests from left to right: union.cpp, intersect.cpp and difference.cpp*

## Part B: Implement CSG Operations [6 marks]

1. You will implement three (3) constructive solid geometry operations described in class: **union**, **intersection** and **difference**. Each operation should be implemented in the corresponding .h and .cpp files in the {SRC\_DIRECTORY}/util/include and {SRC\_DIRECTORY}/util/src directories. For instance, the **union** operation should be implemented in union.h and union.cpp [3 marks].
2. Submit screenshots of your code running the union.cpp, intersection.cpp and difference.cpp test scripts [3 marks].

 Fig. 1: Expected output of all three CSG tests from left to right: union.cpp, intersect.cpp and difference.cpp

## Part C: Build Parametric Model [15 marks]

You will build a parametric model which satisfies the following requirements - exposes 3 or more parameters to the user which control the shape of the model. These parameters should be implemented as global variables in your parameteric model source file. [3 marks] - is made of 10 components or more. Components can be repeated in the design but source code to build a component must only appear **ONCE** in your design. Repeated parts must be instantiated using a common function. [10 marks] - your object must be sufficiently geometrically complex. If you have concerns, please speak with me. [2 marks]

## Part D: 3D Print Your Model [5 marks]

1. Attend a lab training session to learn how the 3D printers work
2. 3D print you model.
3. Submit a photograph of your 3D printed results [5 marks]. If your print fails (some do) submit a photograph of the failure you will recieve full marks.

## Hand In [1 mark]

Collect all required images into a PDF report which must include your full name and student number. Submit this PDF and a zip file containt your A1 source code via **email** to [diwlevin@cs.toronto.edu](mailto:diwlevin@cs.toronto.edu). The subject of the email must be **CompFabA1\_LASTNAME\_STUDENTNUMBER**.