# **INFO6022 – Physics & Simulation 2 - Mid-term Exam – Winter 2025**

Instructor: Michael Feeney

## The exam format:

* You may use any resources you feel are necessary to complete the exam, but you are to answer the questions **on your own, individually**. I will be looking for plagiarism (i.e. copying) very carefully. There is *no possible way* that the specific code to answer these questions, or the output to the screen, would be very similar to the look of another student’s code. Remember, this is a test and there are very clear policies about cheating on tests.
* You may also ***not*** use code that’s taken *mostly* from some existing framework (like downloading the code from the OpenGL book, Learn OpenGL, Open Scene Graph, or something like that)  
  + <http://www.fanshawec.ca/admissions/registrars-office/policies/cheating-policy>
  + <http://www.fanshawec.ca/sites/default/files/assets/Ombuds/cheating_flowchart.pdf>

How can I make the determination that it’s not “your” code? Simple: If I suspect it’s not yours then I’ll ask you to “defend” it one-on-one where I’ll ask you questions and see if you know what you’re doing or why the code is the way it is. If you have no idea what’s going on, then it’s pretty certain you didn’t write it yourself. You *can* use any code *provided in class* or *you* wrote since starting this program *this September 2024.*

* You ***may*** use simple utility libraries like assimp, loadPNG, JSON/XML loaders, sound, etc. **No** boost, though.
* You may **not** use the auto keyword. Doing so will give you a mark of zero.
* You may **not** use generative AI tools to create any of the code for this exam (“you” didn’t write it).
* It is an “open” exam. You have access to anything in any book, internet resource, or anything on your computer, or that has been uploaded in class, including projects you’ve already completed.
* The questions are of equal weight. The exam has **four (4)** questions and **six (6)** pages.
* The questions build on each other so you can put them in one solution/project.  
  If you feel you need further clarification, please include a readme file (and a video if you’d like, though not required).
* ***PLEASE*** delete any temporary files that Visual Studio generates (to reduce the upload size)
* Do ***NOT*** do some clever “*oh, you just have to comment/uncomment this block of code*” nonsense. However, if the questions ***CLEARLY AND OBVIOUSLY*** build on each other, you may combine them (like if one question places objects, then the next one moves objects around with the keys) – even so, **MAKE IT 100% CLEAR** to me what questions the solution is attempting to answer. **I do NOT want to edit the code in any way.**
* For applications: if it doesn’t build and run, *it’s like you didn’t answer it*. I’ll correct trivial, obvious problems (like you clearly missed a semicolon, etc.), but you need to be sure that it compiles and/or runs.
* You have until **11:59 PM** on **Monday, February 24th** to submit all your files to the drop box on Fanshawe Online.   
  This is an exam, not a project, so please submit by the deadline.
* Unless otherwise indicated, all these solutions assume that you are creating/using a C++ project using Visual Studio 2022 using the OpenGL 4.x API (with GFLW, glad, and glm).

I will be building using the default Visual C++ settings (C++ 14).

## Questions:

You are to implement parts of Warhammer 40,000 cinematic, specifically from about 1:40 to 1:50, cued up here: <https://youtu.be/gBgXH7eyRo4?t=102>

In the various games, there are many soft cloth physics sort of effects, including flags, banners, and capes.

Rather than using the specific “banner” shape of the “Blood Ravens” chapter, you will use a typical plan rectangular flag shape as well as having the flag mounted on the side, rather than hanging from the top – as it is in the cinematic.

|  |  |
| --- | --- |
| **The Space Marine banner:**   * Mounted on the top * Hangs down * Has little bits hanging at the bottom | **Typical flag:**   * Mounted on the side * “Flies” to the side * Is simple rectangular shape |

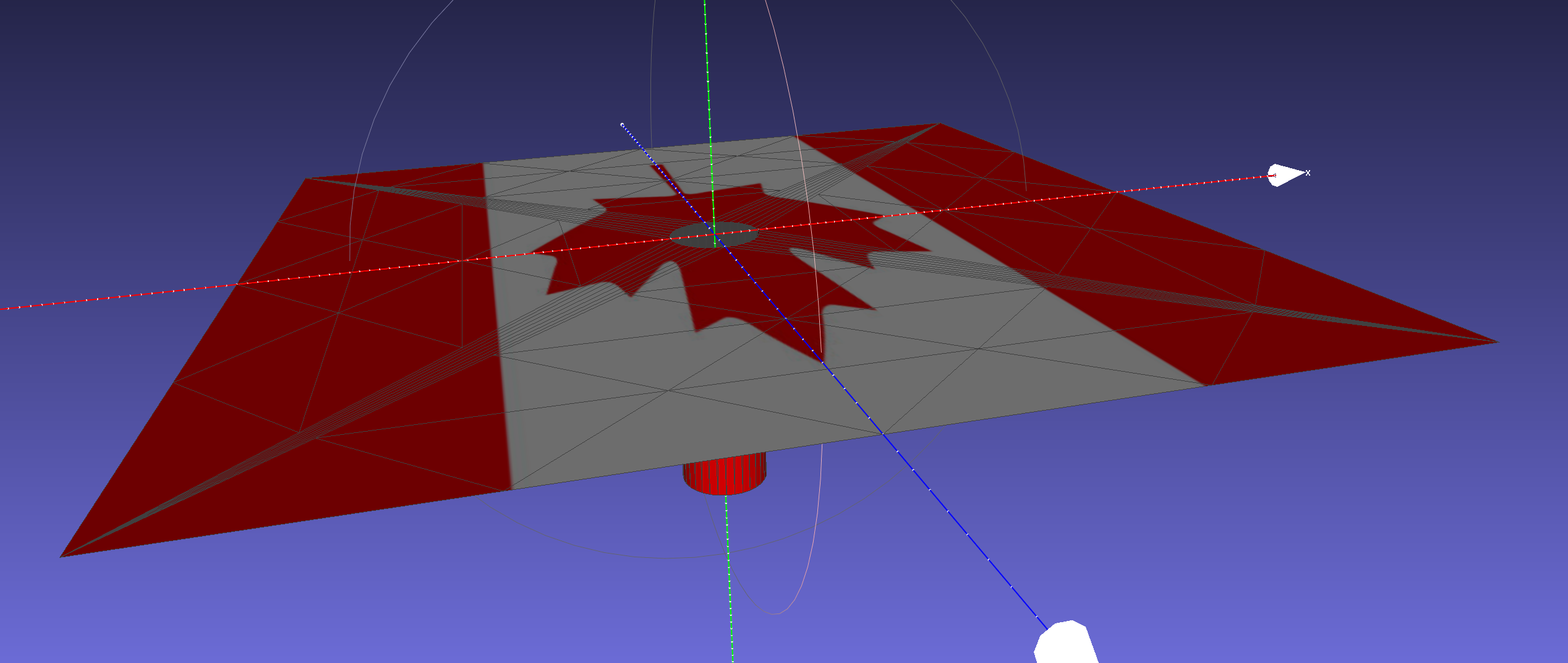
You can find one of your favourite Warhammer Astartes banners here: <https://www.deviantart.com/sinistha/art/Warhammer-40K-Astartes-Chapter-Banners-840856008>

A collection of banners with different designs

Description automatically generated

(I googled “space marine chapter banner” and found a number of them)

1. (25 marks) Recreate the flag after the Space Marine has finished raising it, and the camera A banner with a face and wings

   Description automatically generatedis moving away, with the following features and changes:  
   * You will use a regular rectangular flag (like the “I♥Beer” flag on page 2), mounted on the *side* **not** from the top like the in the video.
   * For a texture, use any Space Marine chapter you’d like.
   * Assume there’s a strong enough wind to hold the flag more or less straight to the side (like the “I♥Beer” flag on page 2)
   * Use a cylinder or some equivalent shape as the flagpole.
   * Use some “ground” mesh as the ground, like the MeshLab terrain, or something you’ve found.
   * Use some kind of evening, daytime, etc. skybox or equivalent to give the effect of the sky and clouds behind the flag.
   * Place a light (or enough light) to easily see the flag, etc.
   * This must be a textured, lit scene, not wireframe or something like that.
2. (200 marks) Using the soft bodied Verlet integration technique from class, show the following effects on the flag from question 1:  
   * The flag must have at *least* 10 x 10 “nodes” along each axis of the flag.   
     In other words, the smallest distance between nodes is 1/10th the length/width of the flag.
   * Start with the wind being essentially absent, and the flag hangs down.
   * Gradually increase the wind until the flag is flying like it did in question 2.
     + You can either add two controls that “turn off/on” the wind, or increase/decrease the wind blowing over the flag.
     + Note that when the wind “stops blowing” the flag will gradually fall down, and when the wind blows, it will gradually fly. This might not take much time (like if there is a sudden “gust” of wind) but you can’t just place the nodes in place. It needs to be simulated.
   * The “wind” can be controlled by changing the overall forces on the nodes of the flag. Usually “gravity” is pulling the flag down, but the “force” of the wind either overcomes this force – or since this is a game, can gradually replace it (the force can *gradually* go from straight down to straight to the left depending on how “strong” the wind is).
   * The flag should be flexible by not “stretchy”, like cotton or nylon rather than spandex or something.
   * The flag should “flap” somewhat in the wind. You can do this by using random perturbation of forces on the “ends” of the flag or a sin function, or something like that. If it looks “good enough”, then I’m OK with it – but it has to “looks like it’s a flag flapping in the wind” to a typical person.
   * *Like question 1, it needs to be a solid, textured, lit mesh, not wire frame or nodes, etc.*
3. (200 marks) Mimic the holes that appear in the flag:  
   * In the cinematic, I’m assuming it’s the Orcs that are shooting the flag with bullets or something. You can assume it’s something else if you don’t like that. Maybe there’s really high winds blowing rocks through it?
   * (Assume that the winds are very high)
   * Assign a key to do the following:
     + Randomly choose a roughly circular location on the flag. This needs to be large enough to encompass at least a few nodes. i.e. you aren’t removing just one node/vertex, but several.
     + Disconnect the connections between these nodes to simulate parts of the flag being knocked/shot out.
     + The effect won’t be a circular hole – like in the video – but more a “tearing” of a “hole” in the fabric. Like imagine you threw an axe at it or something and it ripped a ragged edged hole... I don’t know? But it won’t be round like a bullet hole is my point.
   * Assign another key to do the following:
     + Disconnect the flag from the flagpole and have it fly away.
   * Assign yet another key to do the following:
     + Reset the flag back to “full health”: back together on the flagpole, like in question 2.
4. (200 marks) Flag goes down a hole:

Add the Plane\_with\_tube\_hole.ply mesh model to your scene by replacing any “ground” mesh you’ve used.

This model is a flat plane mesh with a tube placed in it, like a hole in a golf course, or the hole a flag pole goes into, maybe?

The “hole” is a tube that’s centred on the y-axis, and has the top edge aligned to the X-Z plane (the same as the flat part of the mesh). It has a radius of 1.0 units – i.e. is a tube that’s 1.0 unit away from the y-axis. It’s 6 units long (from Y = 0.0 to Y = -6.0).

* + Assign a key to do the following:
    - Disconnects the flag from the pole, letting gravity pull it down.
    - Place the flag/flagpole such that the centre of the flag is at the centre of the tube/hole.
    - Have the flag stop at the ground, but be “sucked” into the tube/hole.
    - You do this by stopping the motion of the particles if they are:
      * At the surface of the plane part *except...*
      * When it is within a 1.0 unit distance from the Y-axis (the same as the tube/hole).
      * It should stop at the end of the tube/hole (which is -6.0 units below the y-axis)
      * Assign another key to switch between drawing this Plane\_with\_tube\_hole.ply model as solid and wireframe (so you can see the flag slipping into the hole)
      * The flag should be solid the entire time, though.

The entire hole (2.0 units across) should be about 20% of the size of the flag.

i.e. the space between the nodes in the flag that are “1/10th the length/width of the flag.” (see question 2) are around 1.0 “units” in length.

If you need to, you may scale the flag and/or hole to make the hole *seem* to me about 20% the size of the flag.

The flag should be slowly “sucked” into the hole that’s *about* 20% the width of the flag.

If it’s taking forever to get into the hole, increase gravity (or add a force) to the flag when it’s falling (just for this question).

Note: there’s two other meshes: Just\_the\_hole.ply and Just\_the\_tube.ply which separate the objects in case you want to examine them in meshlab. They are in the same location as they are in the Plane\_with\_tube\_hole.ply file.

That’s it.