|  |
| --- |
| INFO6022 Physics 2, Instructor: Michael Feeney |
| Project #1 Winter 2024: Soft body (Verlet) integration and constraint |
| Weight: Project weight evenly distributed across the “project” portion of marks |
| Due Date: Saturday, March 8th 15th at 11:59 PM |

*Note: This project can be done alone or in pairs.   
(Possible groups are listed at the end of the document – you must indicate if you are working within the group or alone before submitting)*

Description and Purpose

You are to create a 64-bit Win32 (Windows 7/8/10) + OpenGL 4.x application that displays a static (or dynamic) 3D scene that implements a scene up to the standards of term one, but with the addition of one or more soft body objects, implemented using the Verlet integration and constraint system from class (or an equivalent one).

Details

Your scene **must have** the following items from 1st term (i.e. is “complex enough”):

* Is 3D and has colours and/or textures and reasonable lighting – can’t be wireframe or something.
* Has some kind of environment that the soft bodies interact with.

You will not be given marks for these items, but marks will be taken off (down to zero) if the scene isn’t “complex enough”. The idea is that you are building on your previous knowledge, so adding these items should be “a given” at this point (<https://www.vocabulary.com/dictionary/given>).

**The soft body/bodies:**

They must be utilized using the Verlet integration (or equivalent) done from class, specifically:

* There are particles/nodes connected using distance and/or angle constraints.
* They are using a “forgiving” integration process for soft bodies (Verlet tends to be stable whereas it is very difficult to stabilize explicit Euler or RK4). “Forgiving” in that the simulation doesn’t “blow up” (numerically).
* You may *not* use any existing middleware (i.e. something you found somewhere). This includes just pulling down something from another site or AI generated.
  + Like other situations, I’m assuming you “know what you wrote” and if I’m suspicious, I will ask you about it – **if I think you are just using someone/something else’s code, you will get zero and it will be considered plagiarism.**

Note: I’m *very* suspect about you using an “equivalent” but I don’t want to constrain anyone, either.

The environment (scene) doesn’t have to be super complex, but it can’t just be a flat plane, either. Like you don’t have to simulate a blob pushing its way through the hallways of Hogwarts or something.

There should be some kind of character(s) – or part of a character – and/or environmental component(s):

* A “character” is a player-controlled object that you can move around, for example:
  + A giant blob/amoeba you can
  + A very squishy ball
  + Part of another item like a chain with a ball, a whip, the long neck of a character, tentacles, two characters attached together with a rope/chain, etc.
* An “environment component” is something that your player character/item can interact with, like:
  + A “rope bridge”
  + A jumpy castle
  + Platforms suspended with chains (the chains being the Verlet component)

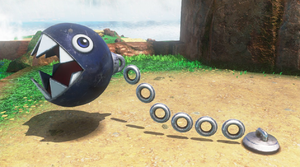
There needs to be at least a few hundred components in at least one of the entities. So, you can’t just have a dozen particles connected in a rope – it has to be more complex than that.

The intention is that these two components are showing two very different uses of this Verlet/soft body techniques. Ideally, a casual player/viewer shouldn’t immediately realize both of these items are using the same sort of integration technique.

That also includes a platform suspected by four (4) ropes – that whole thing (platform + ropes) is considered ONE item.

If you have only one (1) component, then your overall mark will be limited to 65%.

If you have two (2) components, they must be completely different. You can’t just place two identical blobs on the screen, for instance. A blob with a ball and chain, or a blob crossing a rope bridge, or a bunch of platforms suspended over a Jell-O ocean would be considered “two” things. Having a few platforms suspended by ropes does ***not***count.



**Handling the “surface” or the objects:**

* If you have component that have their “particles” completely separate from each other (like the Knack or Rayman or Chain Chomp characters) you will be receive 75% max.
* To receive full marks, at least one of your components must have a “surface” on it, like the typical meshes we’ve been using. This must have accurate normal calculated each frame (i.e. the lighting is correct).

TLDR for full marks: To *not* limit your marks, you need to submit two (2) independent separate Verlet components where at least one has a surface on it.

What I’m looking for is that you are demonstrating this technique in an determined and interesting way, not just some demo that’s barely more than what you saw in class.

**Bonuses (10% each):**

* You have at least five complex soft objects running with an interactive framerate.
* There is a variation in the “stiffness” of the objects (through the number of iterations the constraints resolve). This has to be on the same object where parts of it are “soft” and others are more “rigid”.
* There is some kind of *interesting* and *sensible* change for the stiffness/softness of a component. Like a blob that can move between being very soft and then be very rigid. This has to “make sense” though.
* Have the component be “destructible” (think tearing cloth for instance). You does *not* include just having a single rope that breaks – you could do something like the chains holding the platforms breaking, though.
* If you have another interesting idea, let me know *before* you submit.

Additional requirements:

* You may use any code that has been provided to you in class or you have written yourself.   
  See the "plagiarism" test, later in this document, for more details.
* Further, **you *cannot* simply use an existing game engine** (or part of a game engine), even if it's a "from source" engine (i.e. you have the entire source) to complete this assignment; it should be either completely new of significantly modified. This includes, but is *not* limited to: Unity, Unreal, Cry, Anarchy, XNA, Cocos, Ogre, the framework from the OpenGL text, etc. In other words, you are expected to have made the vast majority (essentially all) of the engine *in this term by yourselves, from "scratch"* - i.e. starting from something a rudimentary as the "OpenGL Book" code or the GLFW starter code (we started with that in class).
* To clarify: you may use code that has been provided to you in class or you have written yourself.
* This also extends to generative AI (chatGPT, etc.). It’s a great learning tool, but beware of just dropping code in unchanged.

|  |
| --- |
| **Note:** A simple test for all of this is asking you questions about your code. This isn’t a new idea: if you’ve written it – and so understand it – then you can discuss it. If you have no idea what’s going on/how to change it/why things are done a certain way, etc., then I’m going to assume you didn’t write it.  Also note that if I suspect you of plagiarism, *you have to convince me that you didn’t*, not the other way around. I’m exceptionally forgiving about this, giving students the “benefit of the doubt”... up to a point. |

* You **may** use certain libraries like AssImp, glm, XML or JSON loaders/handlers, LoadPNG, so long as these are separate (i.e. you can’t say “I’m using Ogre because it has a texture loader...” – you have to be *just* using that part as a separate item).
* You may **not** use boost. If you have boost, you’ll get a mark of zero.
* Your submission must build in Visual Studio 2022 with the 64-bit debug **AND** release library settings with the *default* build settings (ISO C++ 14 is the default, not 17 or higher).
  + If you feel you “need” something beyond version 14, clarify this with me in advance.
  + The reason I’m doing this is most C/C++ development does not use the later “modern” aspects of C++. If anything, it’s more “C” than “C++”. i.e. it’s not uncommon in industry to use C++ 98/2003 and turn off exceptions.
  + TLDR: I’ll be leaving my compiler setting at “C++ 14”, so if you’re using something beyond this it won’t build. And if it won’t build, then you get zero, right?

Additional requirements:

1. Normally a grade of zero will be assigned to any assignment that is submitted late. However, certain rare exceptions apply according to the *Infotech* *Policy on Missed Evaluations and Evaluation Deadlines*.
2. If you code does not even compile, I will not mark it. Period.
3. If you code does not build (i.e. linker error) and run (i.e. no crazy run-time crash that is unexpected), I may investigate this further, but only if there is some simple problem and/or slight configuration error.
4. **Please include a video recording showing off your project.**   
     
   **NOTE:** No video submitted = mark of zero (0).

|  |  |
| --- | --- |
| Transparency: Multiple amounts of blend transparency (different variations between transparent and opaque) | 10 |
| Transparency: Multiple objects (so they must be drawn last, and sorted from back to front relative to the camera) | 10 |
| "discard" transparency present (this is with the "discard" GLSL instruction) | 10 |
| At least three models must have a combination of two (or more) textures, but: |  |
| \* They must be different textures (*each object* is showing a different texture + technique)  **Ignore this page** | 10 x 3 |
| \* An example of "trivial combining" of the different textures (like "modulating" them, adding them, or an equivalent). A more involved "combining" of the textures must be different (using one to change the colour of the other, using one to change the transparency of the other, implementing discard transparency, changing the normal, specular value, etc.). | 10 x 3 |
| Lots of lights (more than you've used before), as well as at least types (at least a “point light” type and a “spot light” type, or equivalents of those). | 30 |
| The ability to show some "debugging" information (so showing bounding boxes, normals, whatever) - the point is that there is an "alternative" rendering mode you can enter Note: just turning on the “light debug spheres” from class is ***not***enough. | 10 |
| Some kind of texture coordinate “adjustment”. This could be “moving” textures, like the UV coordinates change over time, or choosing different parts of a texture (like a sprite atlas or font lookup into a sprite atlas), etc. Basically, show me that you can choose different parts of a texture, not just setting it from 0 to 1 range. | 10 |
| Some camera controls beyond the basics in project #1. This could be pre-set camera locations, “following” a moving target, etc. | 10 |
| **BONUS:** Any other “neat” technical thing (of your choice, clear it with me first, though) | **BONUS:** 20 |

Note: Keep in mind the following:

* These are the maximum marks possible for each section
* There are some minimum requirements that must be present before you will receive any marks. For example: a scene with one static object and one light will not receive any marks; you will only receive partial marks for things like: having two lights instead of four, or loading one model instead of four, etc.

70/12-year old “squinty eye” plagiarism test:

* While you may freely “borrow” mine (or anyone other) code ***but*** your code should be “sufficiently” different from mine.
* You should also give credit for where you found the code, if you’ve used a significant portion of it.
* In other words, you *cannot* simply use an existing game engine (or part of a game engine) to complete this assignment; it should be either completely new of significantly modified.
* How will I determine this?

If I showed your application and/or your source code to either my *very* pragmatic 70-year-old mother, or a typical 12-year-old, and they looked at it, tilted their heads, squinted their eyes, and said, “you know, they look the same,” then they ***are*** the same.

In other words, I’m not going to be drawn into a giant debate over how “different” your code is from mine or anyone else’s, if a complete stranger, say a randomly stopped in the hallways, would conclude that the code and/or output is pretty much the same thing, then it’s “the same.”  
  
A few of you submitted a project #1 that was *barely* different from the in-class code.   
  
I gave you a “pass” that time because you’re new students, it’s a pandemic, etc.   
  
…but don’t test my patience this time. I want to see that you’ve put some effort into it.

Project Corrections

If any corrections or changes are necessary, they will be posted to the course web site and you will be notified of any changes in class. It is your responsibility to check the site periodically for changes to the project. Additional resources relating to the project may also be posted.