CSE 125 Survival Guide

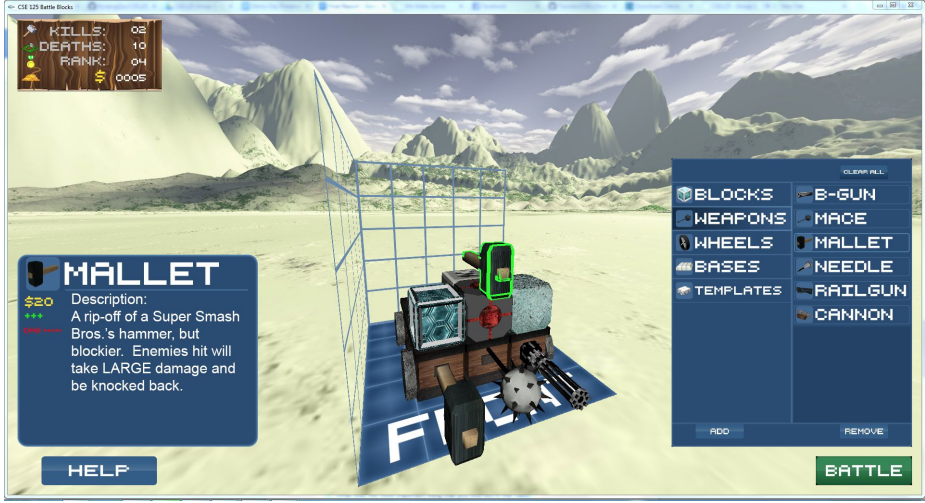
Ruiqing Qiu

About me

Ruiqing (Raymond) Qiu - rqiu@ucsd.edu

4th year BS/MS

Last year : TritonOne Battle Blocks



CSE 125 is

- Time consuming but definitely worth it
- Unbelieveable amount of work for one person
- Wonderful outcome and fantastic experience
- Team, team, team (where the fun is)

Let's begin

- 5 stages
  - Brainstorm
  - Split works
  - Preliminary Development
  - Alpha Development
  - Testing and Demo

Work in the Lab

- Find a time slot that everyone in your group can meet in the lab
- Communication is very important in a large project such as this one
  - B220 is all yours, let me kick others out for you guys.
- Let others know if you can’t make it
- The secret is people work hard when they are around hardworking people
- It’s embarrassing if you don’t do any work while the groups are together

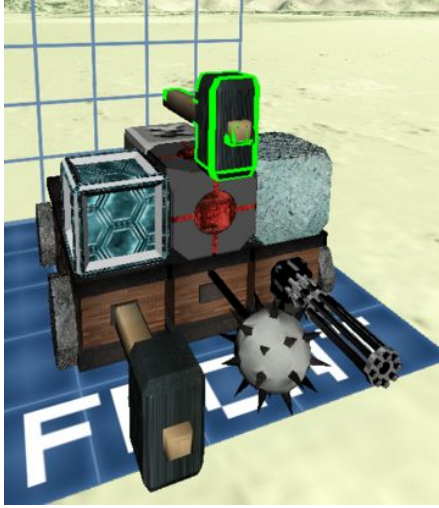
Stage 1: Brainstorm

- If you have ideas already, great
- If not, come up with some ideas first and discuss
- Things to think about:
  - What are the pros and cons for the game?
  - Why would someone want to play it?
  - Will it be fun to play and enjoyed by others?
  - You want to have a game that everyone in your group will enjoy playing, and thus, put in as much time as needed to make this happen
- Some other things:
  - Gameplay (15 to 20 minutes presentation)
  - Development (Be realistic on the milestones and think about risks)

Stage 2: split the work

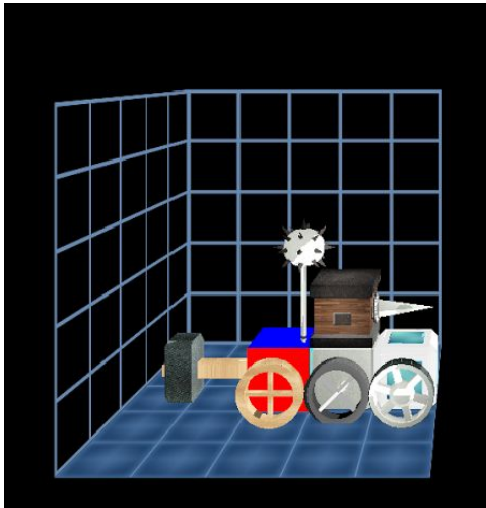
- Graphics
- Networking
- Physics
- Gameplay
- UI
- No one is in charge of one single task
- Help others if you can, it’s a group effort.
- Pair programming (Let at least two people in the group know what the codes are doing)

Stage 3: preliminary development



Loading models

- Decide what kind of models and texture will you be importing into the game
- Write loaders to do that and make sure you stay consistent with it. (tips: we decided to have texture map, normal map, gloss map, and metallic map and then we ran out of time writing shaders to make them look good and decided to not do it.)



Network communication

- Simple server client program, write config files to load in different options. (We never had it, had to compile every time and wasted a lot of time doing that in the early stage.) Have a working server-client program by week 2.
- The next thing is to decide what kind of message you would send and how to differentiate them. Either send message length along with the message or use delimiter. Make sure it’s fully tested. Also think about performance.

3D Modeling, UI and Sound

- Artist. If you have done it perfect, just keep doing what you are best at. If not, it’ s possible to learn it in 10 weeks and have some awesome artworks. However, communicate, you gonna have a lot of work at early stage, nothing on the graphics side could be fully tested without your artwork.
- UI is also a big part. Try to make it user-friendly. But at least you have to show some menus and options to invocate change of states in the game. Decide if you gonna use some library or write your own in OpenGL. Separate the game rendering and UI rendering for better performance.
- Sound. Find some libraries for it. Try to have it in your game as early as possible. It’s easy but if you put it at the end, it’s gonna get really hard to change it.

Stage 4: Alpha development

- Merging codes
  - It’s always good if people are together and know exactly how to solve the conflicts
  - Notify others whenever you update things
- Everyone should know what their goals are now and should each have their own goals done at the end of the week
  - Make sure if something went wrong, discuss and find solutions or alternatives
- There are always things and features unable to make to the final demo. Decide their priority and finish those are essential to the game before working on any optional features. Any bugs, just say it’s a feature.

Stage 5: Demo

- Test your game on the demo machine
  - Available a week before the demo
  - Get some friends to play it and get feedback
- This is the time to make everything stable and function properly
- Decide all the features you are going to talk about
  - Make you presentation fun, prepare early and practice
  - Have clear instruction to teach new players during the demo

Source Control

- Work Locally, store remotely
- Commit early, commit often
- ALWAYS have a “stable” branch, make changes in “feature” branches and merge them in “working” branch in the Labs
- Schedule time when several team members can all work in the labs together

Graphics

DirectX

- Windows Only (also Xbox, but that's not really an option)
- Industry Standard
- Lots of support and examples
- Built into Visual Studio (great tools)
- Does more than just graphics
- Better Shader system (Compute!)

OpenGL

- Cross Platform
- Wrappers for every language
- You probably used it in 167/5
- Only option for Mobile or Web
- Lots of new attention with Steambox

More Tips

- Use Vertex Buffers for speed
- Pass by Reference
- Use meaningful names and have meaningful comments
- Use config files so you don’t have to recompile for small tweaks
- Load everything when the game starts, free it when the game ends (don’t call new in the game loop if you can help it)
- Keep track of hacky fixes so you can go back and fix them later

Where you can get help

- Discuss with your teammates
- Discuss with your classmates (They might have solved the problem already, remember this is not a competition)
- Talk to me (Graphics, design, anything)
- Talk to Geoff (Networking)
- Lots of great online resources (graphics, collisions, game logic, asset management, etc.)
- Piazza

Have Fun

- Most important thing from this guide, huh, besides work in the lab
- Look around you, you are now stuck with them for 10 weeks.
- Make the most out of it!
- Good Luck!