

MS3 – GPU Animation

Student Information

Integrity Policy: All university integrity and class syllabus policies have been followed. I have neither given, nor received, nor have I tolerated others' use of unauthorized aid.

I understand and followed these policies: Yes No

Name:

Date:

Submission Details

Final **Changelist** number:

Verified build: Yes No

YouTube Link:

Required Configurations:

Discussion (What did you learn):

YouTube Process

- Record the YouTube demo
 - You need to record with commentary
 - PLEASE check your volume, should be loud
- Record the desktop (enough to show your directory and the visual studio and output)
 - Show your directory in recording
 - Launch the visual studio (double click solution)
 - Show off relevant parts of the code with commentary
 - Launch and demo the milestone
 - Watch your video
 - Verify that video clear and can you hear the commentary with audio in stereo?
- Note: Milestones Demo cannot be longer than 5:00 mins
 - If you go over... do it again
- Publish your YouTube recording
 - Make sure it is accessible without any login or permission to play
 - It can be private but not restrictive to play by anyone with the link
- Submit your code to perform to the appropriate PA directory
 - Verify it

PDF form (this document)

- *Submit this PDF to perform*
 - *Fill in form*
 - *Name, changelist, etc...*
 - *Submit back to perform*
 - *Check it out*
 - *Submit it back to perform to the same location*

Verify Builds

- Follow the Piazza procedure on submission
 - Verify your submission compiles and works at the changelist number.
- Verify that only MINIMUM files are submitted
 - No – Generated files
 - *.pdb, *.suo, *.sdf, *.user, *.obj, *.exe, *.log, *.pdb, *.db, *.user
 - Anything that is generated by the compiler should not be included

- No – Generated directories
 - /Debug, /Release, /Log, /ipch, /.vs
- Typical files project files that are required
 - *.sln, *.cpp, *.h
 - *.vcxproj, *.vcxproj.filters, CleanMe.bat

Standard Rules

Submit multiple times to Perforce

- Submit your work as you go to perforce several times (at least 5)
 - As soon as you get something working, submit to perforce
 - Have reasonable check-in comments
 - Points will be deducted if minimum is not reached

Write all programs in cross-platform C++

- Optimize for execution speed and robustness
- Working code doesn't mean full credit

Submission Report

- Fill out the submission Report
 - No report, no grade

Code and project needs to compile and run

- Make sure that your program compiles and runs
 - Warning level ALL ...
 - NO Warnings or ERRORS
 - Your code should be squeaky clean.
 - Code needs to work "as-is".
 - No modifications to files or deleting files necessary to compile or run.
 - All your code must compile from perforce with no modifications.
 - Otherwise it's a 0, no exceptions

Project needs to run to completion

- If it crashes for any reason...
 - It will not be graded and you get a 0

No Containers

- NO STL allowed {Vector, Lists, Sets, etc...}
 - No automatic containers or arrays
 - You need to do this the old fashion way - **YOU EARNED IT**

Leave Project Settings

- Do NOT change the project or warning level
 - Any changing of level or suppression of warnings is an integrity issue

Simple C++

- No modern C++
 - No Lambdas, Autos, templates, etc...
 - No Boost
- NO Streams
 - Used fopen, fread, fwrite...
- No code in MACROS
 - Code needs to be in cpp files to see and debug it easy
- **Exception:**
 - implicit problem needs templates

Leaking Memory

- If the program leaks memory
 - There is a deduction of 20% of grade
- If a class creates an object using new/malloc
 - It is responsible for its deletion
- Any **MEMORY** dynamically allocated that isn't freed up is **LEAKING**
 - Leaking is **HORRIBLE**, so you lose points

No Debug code or files disabled

- Make sure the program is returned to the original state
 - If you added debug code, please return to original state
- If you disabled file, you need to re-enable the files
 - All files must be active to get credit.
 - Better to lose points for unit tests than to disable and lose all points

Allowed to Add files to this project

- ~~This project will work "as-is" do not add files...~~

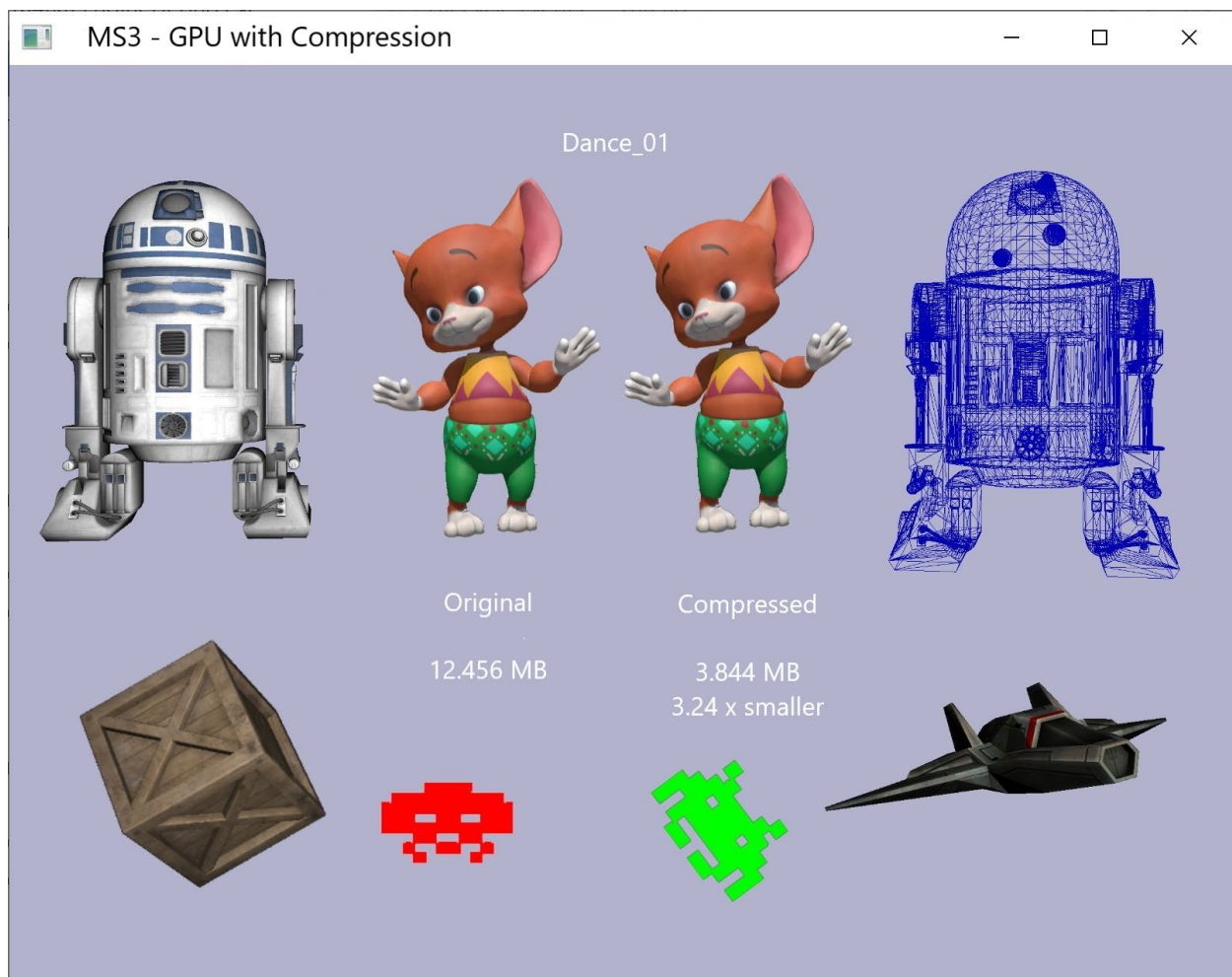
Due Dates

- See Piazza for due dates
- Submit program perforce in your student directory assignment supplied.
- Fill out your this **PDF** and **YouTube link** to perforce
 - **ONLY** use Adobe Reader to fill out form, all others will be rejected.
 - Fill out the form and discussion for full credit.

Goals

- Learn
 - Add GPU computer shaders
 - Store key frames on GPU - only
 - Lighting for skinning
 - Video record 10:00 mins
 - No Memory Leaks!!

Assignments



Mockup of a screen.

1. Same as MS2: Transfer work to GPU

- Compute Shaders
 - Do Blending (Mixing)
 - Needs to be done on the GPU (compute shaders)
 - Show a transitions for 2-3 seconds
 - Blend of a Blend
 - Walking to a Running animation on Mousey only
 - Do the World Matrix calculations for the animations
 - Output of your blending shaders... is local space
 - Convert matrices to world space for the skinning shader
 - Needs to be done on the GPU (compute shaders)
- Compression - Optional
 - Do the animation compression frame drop in the converter
 - Show the ratio on the screen 2D Font
- Store the Keyframes on the GPU
 - All the key frames for ChickenBot and Mousey stored in SBO
 - Several animations each
 - 3-5 for each model
- Add Lighting to the skinning
 - Modify your skinning shader
- MORE models WELCOMED!

2. No MEMORY leaks

- Make sure your project isn't leaking at All
 - Run in Debug with the new Memory tracking enable

3. Required Demo approx. 10 min video

- (1) Show your code and explain each shader
 - (a) Compute – Mixer shader
 - (b) Compute – World calculation shader
 - (c) Vertex – skinning shader
- (2) Show Skinned animation with transition
 - (a) Mousey – Walking smoothly transitioning to Running animation
 - (i) point it out in your demo
- (3) Show ChickenBot animations
 - (a) 3 different animations
- (4) Show 3+ animations on Mousey compression - optional
 - (a) Display Original next to compressed...
 - (b) Display the ratio
 - (c) Required Animations - Mousey
 - (i) Walking

(ii) Running

(iii) (Another of your choice)

Validation

Simple checklist to make sure that everything is submitted correctly

- Submitted project to perform correctly
 - Is the project compiling and running without any errors or warnings?
 - Is the submission report filled in and submitted to perform?
 - Follow the verification process for perform
 - Is all the code there and compiles “as-is”?
 - No extra files
 - Is the project leaking memory?
- Submitted the YouTube link to perform?

Hints

- Do this assignment by iterating and slowly growing your project
 - You won’t be able to finish this assignment in one day - Start now
- Just add Skinning
 - Converting, exporting the data is the hardest part
 - Follow the notes from class
- Memory Leaking
 - Use the new memory system to find your leaks