60-419 Final Proposal

Scott King

**Title**: Mosh-lite

* Nothing with the same name comes up on the first page of google search results. There is one tiny facebook page (22 followers) named Moshlite footwear, and a company named Moshlife. Given a reasonably good itch page or custom web page with even a small amount of SEO, Mosh-lite should be easy to find with Google.
* Domain and twitter handle appear to be available.
* Alludes to the game industry’s practice of inventing non-genres of games by appending a suffix like “-lite” or “-like” to another word, mostly for marketing purposes (e.g. Roguelites). This is also common in music genres and aesthetics with suffixes like “-core” and “-punk” (e.g. grindcore/moshcore/slowcore, Egg punk/cyberpunk/solar punk/dither-punk).
* Also considered the name MidiPunk, but that name is taken by some NFT scumbags, so the SEO would be a bit worse.

**Project Statement**:

Mosh-lite is a midi-instrument controlled interactive environment based on my expressive environment project, Procopia. Player’s interact with the brutalist inspired environment and enigmatic lightbulb-headed figures inhabiting it by playing an electronic drum set that communicates with Unity via MIDI. Hitting different drums and cymbals will cause various visual effects, lighting changes, camera cuts, and reactions from the characters in the scene, most notably moshing.

Artistically speaking, Mosh-lite pursues several goals. I want to explore the possibilities of using musical instruments like a drum set to control a game-like thing, and I want player’s to experience the game’s interface as much as the game itself. Even beyond just being what I’m most familiar with, drums are perfect for this project because percussion instruments are the most tactile and playful instruments, and they are extremely accessible to non-musicians. Mosh-lite encourages people to play LOUDLY with their whole body, which is an intrinsically fun thing to do, and something most video games don’t leverage. Furthermore, and more personally, I want to recreate some part of the fantasy of playing drums with a live crowd. Having attended many hardcore shows, there are few experiences that can compare to the energy of a crowd collectively losing their minds in a moshpit, but being the drummer on stage driving that energy is even more intense, and much harder to realize as an average person. As such, I want players to feel like they’re on a stage at a house show for some no-name local hardcore band, making people go crazy in a pit.

From a researcher’s perspective, I want to use this project as an opportunity to explore the possibilities of MIDI in Unity. Working with drums provides a good proof of concept, but I’m also interested in making something more general purpose that could interact with something like a MIDI keyboard, a drum machine, or even a live coding environment like [Orca](https://github.com/hundredrabbits/Orca). Beyond using these devices as weird controllers for art games, I also want to experiment with using Unity for live music visualizations and for making music videos in a real time engine. The idea being that if a musician played a song on the given MIDI instrument, the visuals in engine would be synced with the music without any need for bespoke animations or scripting.

**One Line Description:**

MOSH-LITE ARE FUCKING DEAD (lol i dunno i’ll think about it)

* A visual instrument, music video making tool, a live music visualizer, and a brutalist basement show for lightbulbs. Moshlite is an experiment in procedural animation and unique interactive experiences.

**Note on feasibility:**

This falls outside of the scope of any technical work I’ve done this semester, which is discouraged in the writeup for the final assignment, so I did a little experimenting to make sure the idea was even feasible. Building on my repo for the environment project, I imported [this](https://github.com/keijiro/Minis) open source library and connected a midi cable to my drums. I was able to receive accurate midi signals in real time as input sources to the new input system. Working with the input system is a pain, but it’s at least a pain I’m familiar with and, on the bright side, it would allow for easy extension to other midi instruments in the future if I wanted to do that. The main technical challenge of the project, then, is mostly taken care of already. Furthermore, I don’t intend to make much in the way of new 3D assets for this project, which is what bogs me down the most, so I’ll be able to spend most of my time working on creating interesting visuals driven by player inputs, which is an undertaking of appropriate scope.

**Game Research**:

[Mosh Pit Simulator](https://store.steampowered.com/app/486440/Mosh_Pit_Simulator/)

The unhinged mess of Mosh Pit Simulator really grabbed my attention when I saw it in class largely because it captures the spirit of punk better than most games that try to do the same. This is partially because it is a bit brash and encourages destruction, but more because it is singularly zany and rough around the edges. It leans into some of the jank-ness of VR and Unity in a way that makes that jank-ness feel artistic and intentional. As an example, the procedural moshing animations of the NPCs in the game feel more chaotic than any scripted AAA cutscene with lots of bespoke animation could (compare <https://youtu.be/cHnJiTKFvYk?t=372> to <https://youtu.be/RWqcljRC07s?t=207>). It also succeeds in being over the top and weird through and through.

In mosh-lite, I hope to recreate some of the experience of mosh pit simulator by creating a similarly chaotic scene driven by procedural animation. In contrast with mosh pit simulator’s vibrant, slapstick aesthetic, I want mosh-lite to lean into the over the top melodramatics of stereotypical hardcore music videos (see something like: [Knocked Loose "Billy No Mates" // "Counting Worms" (Official Music Video)](https://www.youtube.com/watch?v=Lei58UXZzUQ&list=OLAK5uy_nrO7jmxxFUnFAneYpErd7k9pJ9bFhaCQI&index=5)) with lots of jump cuts, shaky cam, dramatic lighting, and the like, to the extent that it’s a little silly in its own way.

[Neo-Brutalism of Tomorrow](https://moshelinke.itch.io/neo-brutalism-of-tomorrow#:~:text=A%20downloadable%20game,itself%20being%20the%20main%20experience.)

The (unfinished) environment of Procopia was largely inspired by my experience play Neo-Brutalism of Tomorrow at LIKELIKE. This game succeeds in creating a very compelling environment to explore, mostly with lighting. Light guides the player through corridors and expansive rooms, while darkness lets players get lost for a bit while exploring. A sudden shift to red light creates dramatic and memorable moment, while most of the light elsewhere creates a consistent and immersive atmosphere. The various exhibits and points of interest give players an explicit reason to explore, but the constructed environment itself is as interesting or more. While mosh-lite is intended to be much less meditative, the lighting and architecture of Neo-Brutalism of Tomorrow is a perfect reference point for its world. One notable point of contrast between the two, however, is the color of the lighting. Neo-Brutalism of Tomorrow is almost entirely black and white with cold, white lights, while Procopia/Mosh-lite is soaked in a sepia toned light, which works better with the brassy textured light bulb people and some of the grainy, bloom-heavy post processing I’ve settled on. All in all, the saturated sepia does a good job of hiding unity’s weaknesses in rendering and my relatively poor 3D skills, while the flat, harsh lighting of Neo-Brutalism of tomorrow would highlight them.

**Useful Links**:

Project Repo:

<https://github.com/jsking2920/Moshlite>

Roland TD-25KV MIDI mappings:

<https://rolandus.zendesk.com/hc/en-us/articles/115000201706-TD-25-Default-MIDI-Note-Number-Map>

<https://rolandus.zendesk.com/hc/en-us/articles/208032456-TD-25-Assigning-MIDI-Notes-to-the-Pads>

Minis Repo:

<https://github.com/keijiro/Minis>

Orca Repo:

<https://github.com/hundredrabbits/Orca>

Good Input System video:

<https://www.youtube.com/watch?v=Yjee_e4fICc>

Relevant article/tutorial:

<https://elizasj.com/Real-Time-Graphics-Sound-Programing-Techniques-Unity-Ableton-Live-Orca>

Good Ragdoll Tutorial:

<https://www.youtube.com/watch?v=DInV-jHm9rk>

**Inspirations**:[Knocked Loose "A Tear in the Fabric of Life" (Animated Film & Full EP)](https://www.youtube.com/watch?v=70cXs8hht_Q)

[Code Orange - IN FEAR (from "Back Inside the Glass")](https://www.youtube.com/watch?v=rApdHROdRuY)

[Neo-Brutalism of Tomorrow](https://www.youtube.com/watch?v=k4HHiXzasi4)

[Mosh Pit Simulator Gameplay (Sos Sosowski) - Rift, Vive, Windows VR](https://www.youtube.com/watch?v=ImCgVgUToqw)

(See pure ref board in repo for mood board)

**Random Ideas/Scratch Notes**

Unity midi toolset

* generic, no coding necessary, plug and play tool for using midi instrument to interface with game objects in unity
* Enables using something like a midi keyboard, electronic drumset, or any other midi device as a controller in a game
* Tool for making games with weird controllers, recording virtual music videos in engine, doing live visuals/vjing

Goals

* Get unity to interface with different midi sources
  + midi keyboard
  + electronic drum set
  + Orca
* Create documentation, a public repo, a demo video, website/itch page
* Post publicly somewhere for people that might be interested
* Final deliverable ideas
  + Music video
    - Knocked Loose (or something like that)
      * Bulb head mosh pit in brutalist environment (something like starting room of procopia)
      * Live drumming, recorded (with phone camera), controlling in engine visuals, also recorded (with obs)
      * Programatic animated moshing, camera angles, particles, lighting, vfx, etc controlled by midi from drums
    - Goalie Fight
      * Similar to above but collab with Mark
  + Abstract Visuals (VJing vibes)
    - Orca program + Ableton, midi to Unity
    - Final video recorded with OBS, three windows:
      * Unity Build, Orca program, Ableton/pilot
    - All black world, no textures
    - Edge detection, white edges, shapes scaling up and down, spinning about, etc
    - just cool stuff happening visually in 3d, synced with music

Possible visual FX

* Post stuff
  + <https://github.com/keijiro/FlashGlitch>
  + <https://github.com/Cyanilux/URP_RetroCRTShader>
  + <https://github.com/keijiro/SimplePostEffects>
  + <https://github.com/keijiro/KinoAqua>
  + <https://github.com/keijiro/KinoHatch>
  + <https://github.com/keijiro/Skinner>
  + <https://github.com/keijiro/Retro3D>
  + dithering
  + depth of field
  + Low res render texture
  + Fog
  + Bloom
  + Vignette
  + motion blur
  + film grain
  + Distortion
    - Panini
    - Fisheye-lens
* Lighting
  + Bulb flickering/turning off and on at once
  + Directional lights
  + Dramatic change in environmental lighting color (Neo-brutalism red light/knocked loose style blue light)
  + God rays
  + All lights off
* Animation
  + <https://github.com/keijiro/PuppetTest>
  + Head banging
  + Ragdoll flailing
    - arms flailing via physics
    - head/spine goes up and down by kick drum or something
    - Legs IK on ground/not ragdolling
  + jumping
  + stock mixamo animations
  + stage diving
  + Guitar hero musician animations
* Camera
  + Jump cuts
  + rotations
  + close ups
* Random
  + Particles
  + <https://github.com/keijiro/Klak>
* Audio
  + Crowd noise
  + Microphone feedback
  + Speaker distortion

11\_17 note

* Rock band style music set up
  + Music that plays when you hit a key (maybe a few tracks to choose from)
  + Drums could be playing sound out of a speaker separate from unity (or not probably)
  + Good since people generally aren’t going to be good at drums and won’t want to hear themselves
  + Drums will simply be a tool for controlling visuals

Could have multiple presets to swap between

* Like different drum kits on the kits module
* One preset for verse where kick makes ragdolls headbang
* One preset where kick makes camera shake
* Etc.

Could have different kind of bulbheads

* For some, kick adds a downward force to their head bone, for some it makes them jump, etc
* Crowd simualtion stuff

Mixamo Animations that could be useful

* “Dancing”
* “Drumming”

Active Ragdoll Research

* <https://www.youtube.com/watch?v=hQub6caVlpY&list=PLM2JG19oUhLOWhyRXsE3-SpUsigOdMPSp&index=1>
* <https://sergioabreu-g.medium.com/how-to-make-active-ragdolls-in-unity-35347dcb952d>
* <https://github.com/sergioabreu-g/active-ragdolls>
* <https://github.com/ashleve/ActiveRagdoll>
* <https://www.youtube.com/watch?v=HF-cp6yW3Iw>
* <https://www.youtube.com/watch?v=I1beTn_913c&list=PL9gnJgSxuivEf8D6upAd5aNj6H4OFWt4m>
* <https://www.reddit.com/r/Unity3D/comments/nmec0u/8_tips_for_animating_active_ragdolls/>

Based on my research, helped along a lot by the resources of Sergio Abreu (<https://sergioabreu-g.medium.com/how-to-make-active-ragdolls-in-unity-35347dcb952d>) and Birdmask Studio (<https://www.youtube.com/watch?v=hQub6caVlpY>), the two methods of procedurally animating ragdolls that I’ve identified as being most usable for this project are as follows:

1. Have a plain ragdoll made up of joints, rigidbodies, and bones that is animated purely with persistent and intermittent forces. Typically, an upward force is applied constantly to the upper chest of the character while another constant force is applied to the character’s feet. The net result of this is a character that “stands” and corrects their posture continuously. Then, intermittent forces can be applied to different body parts to animate the character, such as the feet to make a walk cycle. This method is fast and would result in fun, somewhat jank, and very chaotic animations. It would, however, also require a lot of fiddling with sliders to make things work and it would never be particularly polished.
2. Have a plain ragdoll as described above, but also have a typical animator-controlled character with the same bones without any physics. Turn the renderer for the animated character off and use the rotations of it’s bones for the target rotation of each bone of the ragdoll. This way, the ragdoll will constantly be trying to achieve a desired pose controlled by an animator while being controlled by the physics simulation. This method allows for much more controlled and traditional animations but it also requires having animations and doing more setup in general.

Ideally, Moshlite would use a combination of these approaches. 2 for characters on stage doing more controlled animations and 1 for characters in the pit just being knocked around.

12/4/22 notes

Things to bring for FINAL GAMES expo

* Drum set
  + Pads/rack/module/cord/usb/
* Big speaker
  + Quarter inch cable
  + Power cable
* Elizas little speaker + power cord
* Fold up tray as backup for speaker as laptop stand
* Power strip
* Extension cord
* Printed description plaque + instructions
* Laptop
  + Charger
  + Mouse
  + Stand
* Phone charger
* Cart to get everything from car to gates?
* Long hdmi cord + long usb hub + long usb-c cable

Setup

* Drums facing projector screen
* Speaker to left of drums
* Laptop on speaker
* Game and spotify open

Artist’s statement / instructions

MOSHLITE by scott king

MOSHLITE is a visual instrument, a music video making tool, an experiment in unique interactions, and a basement show at the end of the world.

To Play:

* + Click the screen
  + Choose a song
    - There are no rhythm game mechanics, and you don’t need to play along with the drums in the songs, though the intereactions are designed such that doing so typically looks good
  + Or choose freeplay
    - Alt tab to spotify and choose a song of your own
    - Or just mess about and hit things
  + When the song finishes, the game will reset for the next person. If you want to reset it manually, hit the r key

PLEASE DON’T HIT ME sign on the module, or just set it up separately from rack

While idle, game has a title screen up with Moshlite in some fancy font

When interacted with, player has a handful of songs to choose from, coosing one does this:

* Fade to black
* Title card evangelion style (artist + title + date/time + expo name + gmae name + my name)
* Hard cut
* Song starts playing
* When song ends:
  + Fade to black
  + Credits
  + Back to menu screen
  + Reset scene

Or player can choose freeplay, which just lets the mess with the scene

* Have spotify running on laptop so you can choose whatever music suits you

Have a reset key that resets scene, stops whatever’s playing, and opens up title screen

Mapping

Ride: Camera controls

* Edge: cut to random
* Bow: Pan to random
* Bell: Pan to default cam

Left Crash: Camera FX

* Edge: Shake cam
* Bow: Toggle fisheye + intense shaking effect