LECTURE 11
Announcements
Final 1

• Most of you handed in!
  – Next week retry is due
  – But good not to get behind on final projects

• Don’t forget about previous NC projects
  – They’ll sneak up on you, especially if you have multiple
Mid-Semester Feedback

0 responses

Was it something I said?

Waiting for responses
Mid-Semester Feedback

• https://goo.gl/forms/90LPyXc8k8Dh9aEf1
Mentor TAs

• Mason — Ben S.
• Alex, Jack, Ken — Ben S.
• Loudon — Ben A.
• Alainey, Lael — Ben A.
• Luke, Trent, Nick, Tyler — Jordan
• Liam, Pato — Jeff
• David — Jeff
Playtesting today

• We’ll be playtesting your Platformer3 handins today

• Afterwards = good time to meet with your mentor TA’s
  – Touch base
  – Make Final2 Rubrics
More on Final Rubrics

• Joint process between you and your mentor TA
  – Your job = present what you think is appropriate for each week
  – TA’s job = check to ensure it keeps you on good pace

• Ideal rubrics:
  – Final1 – engine features partially done, some possibly demo-able
  – Final2 – engine features mostly completely done, all demo-able
VR (?)

• We’re thinking about providing support code so that your games can run in VR
  – This would probably happen ~week 4 or 5
  – We would help you get this set up

• Show of hands: who’s interested?
QUESTIONS?
LECTURE 11

Procedural Content
What is procedural content?

- Any content that is generated algorithmically
  - Terrain/biomes
  - Cities
  - Objects/Artwork
  - Characters
  - Quests
  - Puzzles
  - Dialog
Why procedurally generate?

• Cons:
  – More effort to get initial content
  – Less predictable, more susceptible to tricky bugs

• Pros:
  – Replayable
  – Less effort to make a ton of content
  – (Can be) easy to tweak towards desired effects
  – You can more accurately playtest your own content
    • Normally you’re biased by prior knowledge (ex. levels/puzzles)
  – It’s fun! You can make art with code!
Procedures are algorithms

• Often based around a noise function
  – Perlin, fractals, sine, rand, etc.

• Noise can directly translate to visible values
  – Height, temperature, color, character stat

• Use it to choose from a set of options
  – Block type, enemy type, path type, map type, behavior
Procedures are algorithms

• Can also build on top of fixed initial content
  – Many older games used this because of memory constraints

• Can track player decisions and use as input to determine more content
  – Available quests (ex. choosing a faction in Skyrim)
  – Enemies suitable or challenging for current stats, weapon types, playstyle
Online vs. Offline

- **Online** – generated during gameplay
  - Essentially infinite, continuous generation
  - Generate new content for each play through
  - Can adapt to playstyle

- **Offline** – generated during development
  - Can be tweaked/extended by designers
  - Use for design inspiration
  - Algorithm doesn’t need to be fast
  - Doesn’t have to guarantee validity
Validity

• Make sure your content works
• It should look natural in your game’s aesthetic
• No outlier or ‘bad’ maps, even if playable
About Procedural Content

QUESTIONS?
2D Perlin Noise

• Basic height map
  – Combine frequencies for better variation

• Biome partitioning

• Can also combine with other 2D noise functions
3D Perlin Noise

- 3D terrain shapes
  - Could combine with 2D height map
- Subtraction
  - From height map
  - Mazes
  - Perlin Worms (caves with distance constraints)
Dungeons in Grids

• Simple connected grid
• Binary Space Partitions
  – Divide area into spaces, put a room in some, and connect
• Combine grids and paths
Left 4 Dead 2 – The Director

• The terrain layout in L4D2 is set, but other content is generated procedurally:
  – Mini-bosses
  – New weapons
  – Health/Ammo

• The Director assesses the player’s “situation”:
  – Location
  – Health
  – Skill (accuracy, time)
  – Emotion (ex. mouse accelerations, reaction speed, key press rate)

• Tries to engineer the best experience
QUESTIONS?
Procedural Content

OTHER PROCEDURAL CONTENT
Weapon Generation

• Randomize stats
  – Ex. Guns: damage, recoil, particle speed, clip size, fire rate, etc.

• Pick type from preset list
  – Randomize specific attributes of chosen type

• Chose appearance from random combination of traits
  – Randomly built names
Weapon Generation

• Generate bullet design

• Galactic Arms Race:
  – List of mixable attributes defining shot type
  – Adjustable functions for trajectory
  – Weapon drops adjust for your weapon preferences
Enemy Generation

• Starbound generates random creatures on each planet
  – Colors fit creature’s biome
  – Parts are mixed randomly from many existing assets
  – Attacks are also chosen semi-randomly
Stat Generation

- Pokémon are born with hidden IVs (‘individual values’) that determine stat growth rate.
- ‘Effort values’ affect stats based on the enemy defeated.
- Pokémon also have ‘natures’ to influence stats.
- Also shiniess.
Procedural AI

• List of all possible behaviors for an entity
  – Each has its own specific customizations
  – These can be random or influenced by other entity stats, like ‘personality’ or ‘aggression’

• Procedurally choose its graph of behaviors
  – Semi-randomly weighted probabilities and/or triggers to switch states
  – Procedural learning – every time the entity takes damage in a behavior, it becomes less likely to perform it
Procedural Generation vs. AI

• Lots of ‘procedural content’ strategies overlap with game AI solutions
• Try adapting other AI structures for other kinds of procedural content!
• Add procedural/semi-random components to your AI too!
QUESTIONS?
Creating your own procedure

• Start with concept art or descriptions of end goals – what do you want to make?
Determine your requirements

• What steps do you need to get there?
  – What shapes or subdivisions of shapes will help?
  – What variations will change one random desired image/chunk/product to another?
  – Break down each step until you know the specific problem you are solving, and find a solution
Example: Complex enemy

- **Enemy goals:**
  - 1-3 creatures riding around together
    - So each creature needs a rider capacity, and visual mount points
  - Similar enemies should be paired together
    - So each creature should have one or more types to match with
    - Define similar/complimentary color palette for all creatures
Example: Complex enemy

• Enemy goals:
  – Bodies should be visually distinct
    • Mix up required body parts
    • Create various templates for where to attach body parts (ex. two heads -)
    • Create lots of small parts for each type
  – Attacks should fit theme
    • Give attacks types like enemies
    • Find best match between all paired enemy types and attack types
Example: Small route

• Terrain goals:
  – Gentle maze with somewhat distinct ‘rooms’
    • Use a known algorithm like Perlin subtraction or BSP with fuzzy noise around the edges
  – Run a subtle path from start to finish
    • Use A* to find best path
    • Add noise to the path
    • Only draw the path when it overlaps with another Perlin function
Example: Small route

- Terrain goals:
  - Make each room ‘interesting’
    - Use other noise functions to add alternate tiles like flowers or tall grass
    - If tile is far enough from A* path it can choose an obstructing tile
    - Occasionally add some preconstructed ‘features’ like ponds
      - Which can have procedural attributes as well! Yay!
Good resources

• http://pcg.wikidot.com/

• http://pcgbook.com/
QUESTIONS?
LECTURE 11
Maya and 3D Software
So you need assets...

- How will you make them?
- Characters, environment assets, etc.
- Your engine allows for loading OBJs already, so why not make your own?
But how do I make them?

• There are a lot of 3D modeling programs out there to make 3D Assets
  – Maya, Blender, Zbrush
  – Different programs have different benefits

• We will focus on how to do basic modeling and shading of assets in Maya
  – Free for students!!
  – In general, good for all steps of the asset production process

• Download here:
The Maya GUI primitives Attribute editor and channel editor – can be brought up from Window -> Outliner.

Navigation tools

Dropdown menus

**Very important**

Attribute editor and channel editor
Shortcuts

• Left click to select objects
• Right click and hold to bring up marking menu
• Left click + alt to look around
• Middle mouse click + alt to pan
• W – translate
• E – rotate
• R – scale
• F to focus on selected object
• 1 – view geometry unsmoothed, 3 – view geometry smoothed, 4 – wireframe views,
  5 – shaded view, 6 – shaded display with texture maps
Creating Assets

MODELING
Basic Modeling Operations

• Moving particular edges and vertices
• Extruding
• Combining meshes
Modeling tips

• Geometry likes quads
  – These will deform in less unexpected ways when you animate a character
  – Smoothing cubes vs. using spheres

• Soft select

• Orthographic views and sketches

• Use a mouse for middle click!

• SAVE FREQUENTLY for the love of god
Creating Assets

SHADING
So now I have a gray cube

• How do I make it a colorful cube?
• UV Mapping
  – Planar mapping – X, Y, Z
  – UV Editor
• Assigning new materials
  – Lambert, Blinn, etc.
  – Hypershade
Creating Assets

RIGGING
Rigging

• In order to animate a static object, we need to rig it and skin it

• Rigging is essentially constructing a working skeleton
  – Characters usually modeled in the standard “T-pose” to accommodate this process

• Skinning is making sure the character mesh moves with the skeleton
Rigging and Skinning Tips

• This takes a hell of a lot of time, so make only a very simple skeleton if you are planning on animating a character

• Tutorial: https://www.youtube.com/watch?v=Ah-Jk7d30ks

• Save all the time. Do it.
Creating Assets

ANIMATING
Animations

• Animations are made up of *keyframes*, or a transformation paired with a time value

• 3D animations are created by interpolating between keyframes and transforming the model appropriately

• For example, we have a key at \( t=0 \) for \( y=0 \) and a key at \( t=2 \) for \( y=4 \), then at \( t=1 \) our object should be at \( y=2 \)
Animating in Maya
Animating in Maya
Animating in Maya
Resources

- Maya tutorials: https://knowledge.autodesk.com/support/maya/learn-explore/caas/simplecontent/content/maya-tutorials.html
- Doing the first few should get you well enough acquainted with the environment/simple modeling
- Do a lot of googling!! Oftentimes entering the question into the search bar will get you the answer
- There are so many YouTube tutorials out there. They are incredibly useful
- Once again SAVE ALL THE TIME. This program can freeze and crash a LOT
QUESTIONS?
LECTURE 11

Rigged Animation
What you’ll need in your engine

• **Skeleton or Rig class**
  – Define the pose of a character model

• **Skinning**
  – Deform character model based on position of all joints

• **Animations**
  – Define keyframes poses
  – Interpolate pose between key frame poses
  – (Probably) Parse animation file type (e.g. .fbx) to create sequence of keyframe poses
Skeleton

• Skeleton represented as a tree
• Each joint is a node, may or may not have children
• Store transformation at each joint relative to parent joint
  – Global transformation of a joint = global transformation of parent * local transformation of joint relative to parent
• Like CS123 Sceneview
Skinning

• Consists of binding a skin (mesh) on top of a skeleton so that the mesh deform realistically when bones move

• Useful for many animations
Skinning

• Given a set of vertices representing your mesh, and a set of joints or bones:
  1. Determine how much each vertex is affected by each bone
  2. Calculate how the changing bone positions affect the vertices they influence

• The first item is something you’ll do while creating content, the last is your engine feature
Skinning: Vertex Weights

• Each vertex will have bone/joint ‘weights’
  – Should sum to 1
  – Each weight determines how much bone affects it

• Each vertex can have up to a set number (usually around 3-4) bones that affect it

• Assigning weights will be done by hand in a program like Maya
Skinning: Vertex Transformation

• Need to define “rest pose” of the mesh
  – Where the mesh vertices are when the skeleton is in standard T-pose
  – Extract the transformation of each vertex relative to associated bones

• Next step: when pose changes deform vertices by interpolating transformation relative associated joints
Skinning: Vertex Transformation

• Example: Linear Interpolation
  – Say we have joints 1 and 2 in the mesh
  – Let v be a vertex with joint weights 0.2 and 0.8 for joints 1 and 2
  – Let v_1 be the position of vertex v in the coordinate system of joint 1 in the initial mesh pose
  – Let v_2 be the position of vertex v in the coordinate system of joint 2 in the initial mesh pose
  – Let T_1 be the transformation of joint 1, T_2 be the transformation of joint 2
  – \( v = 0.2 \times T_1 \times v_1 + 0.8 \times T_2 \times v_2 \)

• http://scribblethink.org/Work/PSD/SSDtutorial.pdf
• Other types of interpolation as well
  – Dual quaternion is the most popular
Skinning: Vertex Transformation

• Probably want to perform this transformation in a vertex shader
  – This way — GPU operates in parallel on all vertices
  – Need some way of passing skeleton, and vertex weights to the GPU
Animations

• Keyframes
  – At time $t_0$, joint $j_3$ rotation is $x_0$
  – At time $t_1$, joint $j_3$ rotation is $x_1$
  – Interpolate between these

• Functions
  – Set a joint’s rotation to $a\sin(b\times\text{time})$
  – Might be useful for a looping walk animation

• Combination of tools
  – Ex. Joints animate with a function while character is moving, then interpolate back to resting position
Animations

• Abstraction: build small animation pieces (ex. ‘swingArms’, ‘bounceTail’) and combine them for more complex behaviors

• Programs like Maya have these features and many more – look for inspiration to make something complex
Animations

• Loading animations from files
  – Use assimp
  – It can load all of the file types into common data structure
  – [https://github.com/assimp/assimp](https://github.com/assimp/assimp)
Content Creation

- Rigged animation takes good amount work to set up
- But if you’re planning on creating content also expect to spend a lot time on making animations that look nice
PLATF0RM3R3
PLAYTESTING!
Yay!
(And then final2 rubric meetings)