

# Video Game Design - Final Project Grading Criteria

## **Third Party Resources:**

You may use third party resources for media (textures, models, animations, sounds, etc.), but you can't use all-in-one assets that are completely implemented for game play. Such assets are sometimes billed as "make a game without writing any code", such as Unity's 3dGameKit. For instance, a character that includes everything needed to just drop into a scene and immediately work for player control is not acceptable. Please refer to the sections below for further details.

If there is a particular asset that you want to use but is part of an all-in-one package, you can request an exception to the above policy by contacting your TA for review. You will also need to clearly document what is provided and what is your implementation.

## **Required Gameplay Concepts:**

Note that the sub-list items below are not meant as exhaustive checklists, but instead demonstrate examples of the types of things we are looking for in each assessment category. Also, you will see that there is overlap between some of the categories. Be especially mindful of these similar objectives that keep reappearing as they are generally the most important for the project.

### *It must be a 3D Game Feel game! (5 pts)*

- You have implemented a 3D Game that can be defined as a Game Feel game
- Clearly defined, achievable, objective/goal? (E.g. player can complete, or alternatively fail at, a level. NOT a sandbox)
- Communication of success or failure to player!
- Start menu to support Starting Action?
- Able to reset and replay on success or failure (e.g. In Minecraft when "You died", there is a "respawn" button)
- Sorry, no first-person perspective games (e.g. FPS) unless briefly used for a special action like a sniper rifle or photography game mechanic.

### *Precursors to Fun Gameplay (20 pts)*

- Goals/sub-goals effectively communicated to player
- Your game must provide interesting choices to the player
- Your player choices must have consequences (dilemmas)
- Player choices engage the player with the game world (e.g. buttons, climbable ledges, gun turrets, computer terminals, vehicles, etc.) and inhabitants (AI-controlled agents)

- Avoid Hollow, Obvious, and Uninformed Decisions
- Avoid Fun Killers (micromanagement, stagnation, insurmountable obstacles, arbitrary events, etc.)
- Your game should achieve balance (balance of resources, strategies, etc., as appropriate)
- Reward successes, and (threat of) punishment for failure
- In-game learning/training opportunities (e.g. gate restricts player to safe area until basic proficiency in new skill is demonstrated, etc.)
- Appropriate progression of difficulty
- Avoid opportunities for players to trivially beat your game challenges. That means playtesting to observe emergent strategies that might break your game!

### 3D Character/Vehicle with Real-Time Control (20 pts)

- Based on skills learned in Character Control Milestone
- Character control is a predominant part of gameplay and not simply a way to traverse between non-“Game Feel” interactions. (e.g. walking between different point-and-click puzzles.)
- No Unity tutorial characters or CS4455 milestone characters allowed
  - o You must use characters other than those in tutorials; that includes the “3DGameKit”
- Your character is not a complete prepackaged asset from a 3<sup>rd</sup> party
  - o It is ok to use models and animations from another source, but your team should configure animation control and input processing
- Utilize a character/vehicle controlled by the player with engaging animations that react to the player’s inputs. Examples: a humanoid skeletal animated character animated via Mecanim that blends between different animations based on the player’s inputs. Or perhaps a vehicle with moving wheels controlled by a throttle input and moving suspension and driver’s head that turns, etc.
- Player has Game Feel style control: continuous, dynamic, low latency, variable/analog-style control of character movement majority of time (ideally support a popular analog controller like PS4) Examples: analog stick movement, Mario’s variable jump height from button hold duration, modifier-button control of a speed boost
- Choice of controls is intuitive and appropriate (e.g. shouldn’t make difficult button mappings or controls where it is unnecessarily difficult to interact)
- Fluid animation. Continuity of motion (no instantaneous pose changes, no glitching, unintended teleporting, a ground-based character stuck floating in air, etc.)
- Humanoid characters should not slide or moonwalk. Use root motion, possibly with IK corrections!
- Aim for low latency responsiveness to control inputs

- Camera is smooth, always shows player and obstacles in a useful way, automated following of character or intuitively controllable as appropriate
- Camera has limited passing through walls, near clipping plane cutting through objects (e.g. player model), etc.
- Note that using existing 3d Party camera scripts is acceptable. But you must configure/modify to suit the unique needs of your game
- Auditory feedback on character state and actions (e.g. footsteps, landing, collisions, tire squeal, engine revs, etc.)
- Coupling with physics simulation via animation curves, animation callback events, handoff to ragdoll simulation, IK adjustments to animations, vehicle that breaks up in wrecks, etc.

### 3D World with Physics and Spatial Simulation (20 pts)

- Based on skills learned in Physics Milestone
- You have synthesized a new environment rather than just use a downloaded level from the asset store. You can certainly make use of 3<sup>rd</sup> party assets, architecture, terrain, etc., but you *must* synthesize a unique environment tailored to your character's abilities and the game challenges you wish to support.
- Both graphically *and* **auditorily** represented physical interactions
  - o Consider *every single* cause and effect interaction and make sure there is associated audio
- Graphics aligned with physics representation (e.g. minimal clipping through objects)
- Appropriate boundaries to confine player to playable space (players should never encounter avatar falling to negative infinity or escaping the level; use invisible colliders and death zones when necessary)
- A variety of environmental physical interactions possibly including:
  - o Interactive scripted objects (buttons that open doors, pressure plates, jump pads, computer terminals, etc.)
  - o Simulated Newtonian physics rigid body objects (crates, boulders, detritus, etc.)
  - o Animated objects using Mecanim (can be used for non-humanoid animations too!), programmatic pose changes, kinematic rigid body settings, etc. (moving platforms, machinery gears, gun turrets, etc.)
  - o State changing or destroyable objects (glass pane that shatters, boulder that breaks into bits, bomb that blows up, etc.)
- 3D simulation with six degrees of freedom movement of rigid bodies
- Interactive environment
- Consistent spatial simulation throughout (e.g. Running speed remains same regardless of framerate. gravity constrains maximum jump distance; shouldn't be able to jump more or less in different cases unless obvious input control or environmental changes are presented to the user).

### Real-time NPC Steering Behaviors / Artificial Intelligence (20 pts)

- Based on skills learned in AI Milestone
- Your agents are not complete prepackaged assets from a 3<sup>rd</sup> party
  - o It is ok to use models and animations from another source, but your team should configure animation control and AI steering and decision making
  - o Also, don't use Unity tutorial or CS4455 milestone characters
- Multiple AI states of behavior (e.g. idle, patrol, pathfinding, maniacal laughter, attack-melee, attack-ranged, retreat, reload, teeth flossing, etc.)
- Smooth steering/locomotion of NPCs.
- Predominantly root motion for humanoid characters.
- Reasonably effective and believable AI?
- Fluid animation. Continuity of motion (no instantaneous pose changes, no glitching, unintended teleporting, character stuck floating in air, etc.)
- Sensory feedback of AI state? (e.g. animation, facial expression, dialog/sounds, and/or thought bubbles, etc., identify passive or aggressive AI)
- Difficulty of player engaging the enemy is appropriate?
- Perceived as a *fair* opponent? You may need to implement a perceptual model for your AI.
- AI interacts with and takes advantage of environment? (presses buttons, takes cover from attack, collects resources, knocks over rigid body objects, etc.)

### Polish (15 pts)

- Overall UI
  - o **Implement a start menu GUI**
    - Credits and Asset/Library Licensing
    - Anonymous credit is fine for students that wish (e.g., a fantasy name/handle)
  - o **Implement in-game pause menu with ability to quit game**
  - o Your software should feel like a game from start of execution to the end
  - o There should be no debug output visible (you don't need a name on the HUD like the individual milestones)
  - o Ability to exit software at any time
  - o No "test-mode"/"god-mode" buttons, etc. (OK if accessible via menu option or unlikely-to-press-accidentally button combo like: ↑,↑,↓,↓,←,→,←,→, B,A,SELECT,START)
  - o GUI elements should be styled appropriately and consistently
  - o Transitions between scenes should be done aesthetically (e.g. fade in, fade out, panning cameras, etc.)
- Environment Acknowledges Player
  - o Note: there is obvious overlap here with the Spatial Simulation category above. In regard to polish, we are looking for

environment interaction enrichment/appeal, as opposed to impact on core gameplay.

- Should include many of the following:
  - Scripted aesthetic animations (swaying trees, grass, scurrying bugs under moved rocks, etc.)
  - Proximity-based events (alien plants that retract if you get near, picture frame falls off wall, etc.)
  - Surface effects, such as texture changes or decals (e.g. dentable surface, bullet holes, etc.)
  - Particle effects (e.g. dust or splashes around footsteps)
  - Auditory representation of every observable game event
  - Physics event-based feedback such as particle effects and one-shot audio for collisions of different types
  - Sonifications that map physical properties to audio effects. Example: Wheeled robot gear whine generated via volume attenuation and frequency shift of looped audio as a function of robot's speed
- Cohesiveness / Unified Aesthetic
  - Artistic style (extremely simple is fine! But be consistent and thoughtful)
  - Shading and lighting style
  - Unified color palette throughout game
  - Sound theme, including consonance (also avoid monotonous effects)
- Appeal
  - No glitches
  - No easily escaping the confines of the game world (make proper and plausible barriers)
  - No obvious edge of the game world unless part of intended gameplay (e.g. a quad for a ground plane that the player can see where they fall off the world)
  - No getting the player stuck
  - Stable (e.g. should play consistently the same on variety of hardware)

**Fun / Engaging / Immersiveness / Emotional Response / Gestalt / “Flow”  
Achieved (~ 10 pts max bonus points)**

NOTE: This category reserved for the FINAL project and *not* the alpha.

This category is acknowledgement that your game is *greater than the sum of its parts* and is particularly fun and/or highly engaging! Earning points here means that the instructor(s) felt that your game achieved something special that is not accurately reflected by the criteria above. Assessment of this category can only bring your grade UP (i.e. bonus points). That said, there is something almost mystical about making a really fun game and even the

most competent and industrious group might make a game that doesn't reach this level of achievement. You still need to have a "Game Feel" style game to get acknowledgement in this category (along with the technical requirements), and rarely will a game that is significantly deficient in terms of the project requirements gain many "fun" bonus points.

**Submission (Up to 20 points deducted if out of compliance)**

- If In-person instruction: Your group showed up ON TIME and gave presentation?
- If In-person instruction: *Quiet and paying attention* during other team presentations (all presentation days)?
- Have the project deliverables been submitted properly?
- **Readme** file?
- **Video(s)** documenting game concept and play?
- If distance learning instruction: video trailer on time for showcase/presentation video?
- **Playable game binaries provided?**
- If In-person instruction: Copy of digital **presentation materials**?
- **Team Member Assessments** submitted?
- Proper packaging format and no unnecessary files (see Assignment Packaging and Submission page on Syllabus)