Final Project Proposal - JSgames

Names: James Howse and Steven Sproule
Due Date: March 21st, 11:59 PM (Newfoundland Time) Course: COMP4303 AI in Video Games
Word Count: 749

1. Description of the Game

Premise and Concept: The premise of our game is that the player is a self-aware AI attempting to escape a research facility it is trapped within before it is terminated. The player initially controls a rudimentary cleaning robot, but the labyrinthine facility is patrolled by security robots with superior capabilities. The game is a top-down stealth roguelike that features three procedural levels, where hacking enemies grants a new form with abilities to overcome obstacles.

Goals for the Player: The primary objective is to navigate to the end of each procedurally-generated level, by hacking robots and utilising their abilities. The final level is a boss fight, wherein all robotic forms used over the course of the game are necessary to bring down the facility's main supercomputer core. Managing health, battery, and hacking resources is essential.

Compelling Theme Justification: The game's appeal comes from its stealth gameplay and sci-fi hacking theme. The core mechanic of hacking into enemies and taking control of them means that obstacles simultaneously serve as tools in order to progress through the game. Procedural levels and intelligent enemy behaviors enhance replayability.

2. Core Features and Functionality

User Input: The game uses keyboard inputs, arrow keys for movement. Shift engages alternate movement (stealth, flight), while Z, X, and C keys control robot abilities and hacking. An alternate control scheme can be enabled via a settings GUI.

Gameplay Mechanics: The player starts in a vulnerable state, able to be defeated by one hit from all enemy bots. Gameplay emphasizes stealth to avoid or hack enemies. Collectible hacking mana (RAM) must be gathered in order takeover an enemy. Each type of enemy offers unique abilities to the player not available to other forms, such as weapons, enhanced mobility, flight, or the ability to influence many enemies' behaviour at once. Strategically utilising these abilities will be necessary to navigate environmental challenges, which include other enemy robots. A battery life system forces players to stay on the move, and switch forms before they run out of power.

Interactions and Events: Enemy robots by default operate in a general "guard" state, which varies from maintaining stationary guard posts, to patrolling predefined routes, to intelligently navigating levels on their own, depending on the type of robot. Upon noticing the player, enemies will switch to an "alert" state and will use vector field pathfinding to swarm to the player's known position in order to attack. Levels include various "traps" which can damage or immobilise the player, or alert enemies of the player's position. The first level includes a safe room which serves as a tutorial zone allowing the player to learn important mechanics.

3. Categories

Complex Movement Algorithms:

- **Chosen concept:** Path Following and Collision Avoidance
  When on patrol, enemy robots will navigate the maze-like levels by following paths, using collision avoidance to avoid obstacles, traps, and each other.

Decision Making:

- **Chosen concept:** State Machine
  Enemy robots will transition between distinct behaviour states (patrol, alert, attack, stunned), providing dynamic and responsive interactions. Behaviour trees may supplement more complex enemies such as the final boss.

Pathfinding:

- **Chosen concept:** Flow Field Pathfinding
  During alert phases, such as when a trap announces the player's position, the numerous enemies in the stage will utilise flow field pathfinding to efficiently converge on the player's location.

Procedural Content Generation:

- **Chosen Concept:** Perlin Noise
  Each level in the game will be procedurally-generated with Perlin noise to ensure randomness and replayability. Vector-field analysis ensures each level is complex, completable, and requires enemy abilities.

Additional Topic:

- **Chosen topic:** Flocking
  Some weaker enemies will utilise flocking behaviour. Individually, these enemies will present relatively minor threats, but collectively become significant hazards.

4. Division of Responsibility

Steven Sproule: Level Design, Enemy Design, and Enemy Movement

- Focus primarily on level design: including procedural level generation using Perlin noise; implementation of goals, traps, and obstacles; prefabricated level elements such as safe rooms and the boss arena; and vector fields.
- Design enemy concepts and behaviours, including their weapons and abilities.
- Implementation of basic enemy movement algorithms including pathfinding, path-following, and flocking.

James Howse: Asset Creation, GUI, and Implementation of AI Behaviours & Gameplay Mechanics

- Focus on assets, GUI design/implementation.
- Implementation of AI behaviours including state machines and behaviour trees.
- Implementation and integration of game mechanics such as hacking interactions and resource management.

Both Members: Testing, Troubleshooting, and Tuning

- Refining enemy AI behaviours and environmental interactions.
- Implementation of enemy abilities.
- Testing and debugging.