AI Escape: The Sequel

Project Summary

Objectives

The objective of the game is for the player to locate a specific target object within the environment, while avoiding an AI-controlled NPC. The player must successfully find the target object and reach the exit before they are caught by the AI.

Gameplay

Core Mechanics

Player Navigation

The player views the game from the perspective of their character using a first-person perspective, and moves the character by using the appropriate inputs.

Player Interactions

The player will interact with capable objects in the world by getting within a short distance of the object, looking directly at the object with the camera, and pressing the interaction input.

If the player gets close enough to the AI NPC and interacts with it, the NPC is forced into its stun state (if it is not already in the stun state).

Artificial intelligence

The AI NPC must use the A* path-finding algorithm to navigate the environment, though you may use the native implementation of the algorithm included within the AI toolset of Unity to achieve the path-finding.

The NPC will be need to simulate basic intelligence for the player to counter, and will achieve this through the use of four (4) basic states – idle, wander *or* patrol, chase, and stun. *Note that the initial default state for the NPC will be the wander state or the patrol state, and not both.*

During the ***idle*** state the NPC must stop moving for a randomly selected interval between three and then (3-10) seconds in length, before transitioning back to the default state.

During the ***wander*** state the NPC will determine an available random position in the scene to move to, and move towards it at the default movement speed. When the NPC gets close enough to the target position, it will transition to the idle state before moving to another available random position in the scene.

During the ***patrol*** state the NPC will move between a set of pre-defined positions in the scene in the order that they have been defined, at the default movement speed. When the NPC gets close enough to the target position, it will transition to the idle state before moving to the next position in the sequence.

During the ***chase*** state the movement speed for the NPC should increase and the NPC should move towards the player's current position. When the NPC reaches the player, the player fails the game. If the target object has not been interacted with and the player moves out of the view radius of the NPC, the NPC should transition back to its current default state. If the target object has been interacted with, the chase state becomes the current default state for the NPC.

During the ***stun*** state the NPC movement is stopped for three and a half (3.5) seconds, before transitioning back to the current default state.

Each behaviour state for the NPC must also integrate one (1) state-specific animation and one (1) state-specific audio clip.

Game Rules

The primary objective for the player is to locate and retrieve a target object that exists in the environment of the game. Before the target object is retrieved by the player, the AI NPC will transition back to either the wander state or the patrol state when the player moves out of the NPCs view radius. Once the player has retrieved the target object the chase state becomes the default state for the NPC, causing the NPC to immediately begin chasing the player despite whether the player is outside of the view radius for the NPC.

When the player wins or loses the game, a UI prompt that allows the player to return to the main menu scene should become accessible. When the player has successfully retrieved the target object and reached the 'exit' for the level while avoiding the NPC, the UI prompt should display a victory message. When the NPC catches the player, the UI prompt should display a game over message.

Specifications

Platform

A gold-master build that is compatible with windows will need to be built. A web-compatible build will also need to be constructed and tested across the Google Chrome and Mozilla Firefox browsers.

User-Interface

The UI for the game will need to be setup to use a resolution of 1920x1080, and will need to scale with both the screen width and height.

A crosshair will need to be included to help the player direct their world interactions.

The UI will need to include an element that indicates whether the current objective is to retrieve the target object, or to escape to the exit.

Main Menu

You are required to implement a main menu that is functional across multiple input devices including the mouse, keyboard, and selected gamepad controller.

The player must be able to iterate through each of the menu buttons using the directional arrow keys on the keyboard as well as a suitable input on the relevant gamepad controller (such as an

analogue stick or the directional pad buttons). The selected button must be able to be activated by using the 'enter' key on the keyboard, and the 'X' button on a PlayStation controller or 'A' button on an Xbox controller.

The menu functionality must also be compatible with the mouse, clicking a button once to select it and clicking the selected button a second time to activate its functionality.

When a menu button is selected it should display a colour different from its default colour. When the mouse hovers over a button it should display another colour different to its default and selected colours.

The menu will need to include three (3) buttons – one to load a new game, one that displays a window with some information about the game, and one to quit to the desktop.

Control Scheme

Camera movement must be bound to mouse movement, and movement of the left analogue stick on the integrated gamepad controller.

Player movement must be bound to the 'W' and 'S' keys to navigate along the 'forward' directional axis and the 'A' and 'D' keys to navigate the 'right' directional axis, and to movement of the right analogue stick on the integrated gamepad controller.

World interactions must be bound to the 'E' key on the keyboard, and by to the 'triangle' button on a PlayStation controller or the 'Y' button on an Xbox controller.

If you integrate an optional jumping mechanic for the player it must be bound to the spacebar on the keyboard, and the 'X' button on a PlayStation controller or the 'A' button on an Xbox controller.

The player must be able to quit the game by pressing the 'escape' key on the keyboard.