## horizontal line



Smart AI

24.06.2021

**─**

Your Name

Your Company

123 Your Street

Your City, ST 12345

# 

# Changelog

|  |  |  |
| --- | --- | --- |
| Version | Date | Changes |
| 1.0.0 | XX/XX/20XX | Initial Setup |
|  |  |  |
|  |  |  |
|  |  |  |

# Contents

[**Changelog**](#_5kn5bjry9gjp) **1**

[**Contents**](#_62o0kgg5oynz) **2**

[**Introduction**](#_oxk53rrzhe3j) **3**

[Rationale](#_sui8vh2z74cg) 3

[Background](#_ht161vq8mk3m) 3

[Terminology](#_je3u68g7apd6) 3

[Non-Goals](#_8q92eeds6myf) 3

[Proposed Design](#_a10xfv7dn01b) 3

[Software and Hardware Requirements](#_t60nfhfpuxia) 3

[**Gameplay**](#_jkl76wqkyn7y) **4**

[Gameplay Mechanics](#_vmoqexbqk7tf) 4

[Mechanic #1](#_we558vptzf6r) 4

[Mechanic #2](#_k07jf3o5x9cx) 4

[Mechanic #3](#_nw6w6su1lzyq) 4

[Mechanic #4](#_eup3qr74kmzw) 4

[Controls](#_m6z7rbcrryjc) 4

[Mappings](#_sskpmr5w3d3t) 4

[**System Architecture**](#_cwo5vxww8gb) **5**

[Data types](#_4sgi0ird3xzv) 5

[Data Model](#_7ono6u1cvktu) 5

[Interface/API Definitions](#_cga8s6z1ahbo) 5

[Impact](#_i80vs9qe7l24) 5

[Risks](#_230o9g1wesmu) 5

[Alternatives](#_9m502pllby8o) 5

[**Shader**](#_miq9noo9xeix) **6**

[Shader types](#_3wyw3b4cesv3) 6

[Render modes](#_o9291nq9ijt3) 6

[Processor functions](#_wph6am6x3u0b) 6

[Vertex processor](#_ne1x1wyliilo) 6

[Fragment processor](#_4nrb129bye6v) 6

[Light processor](#_tzyu70k510p2) 6

# Introduction

## Rationale

/ Trying to accomplish an AI that can decide to go in different directions based on what the current objective is. Based on the previous task, the AI could only accomplish moving randomly between points. /

## Background

/ See AI PathFinding assessment. /

## Terminology

/ NavMesh = Unity’s NavMesh system This may also be a prefix to : Agent, Surface and Obstacle.

AI = Artificial Intelligence

FlyCam = James’ script of being able to control the camera/

## Non-Goals

/ One might expect this solution to solve the case of having the AI not change it’s direction immediately after changing its State machine. In this build, it is not necessary for completion. /

## Proposed Design

/ Start with a brief, high-level description of the solution. The following sections will go into more detail.

The AI using a NavMesh Surface, will follow waypoints using their position coordinates to set the destination of the AI. The AI will then change states upon entering an invisible trigger zone causing it to change paths. Which will then lead it back onto the previous path, then triggers a new invisble trigger zone which then leads the AI into two switches to chose from, and then picks one and proceeds to finish the Maze. /

## Software and Hardware Requirements

/ Software requirements: Unity - version 2020.2.2f1

GitHub (desktop and web version)

Hardware requirements,

Windows 7 (SP1+) and Windows 10, 64-bit versions only.

X64 architecture with SSE2 instruction set support

DX10, DX11, and DX12-capable GPUs

Hardware vendor officially supported drivers

/

# Gameplay

## Gameplay Mechanics

### Mechanic #1

/ Point and Click: the user points with their mouse (their USB mouse not a live one) and is able to click on UI elements. /

### Mechanic #2

/ Movie-esque; The player is able to sit back and relax while watching the AI do its thing./

### AI Mechanic # 1

/ Changing state: The AI can change it’s states based on entering a trigger zone that has a tag telling it which state to tell it to go to. There are 3 possible states being Progressing, GoToSwitch and GoToCoin. This relies on a function called SetDestination which sets the destination of the AI based on a gameObject’s waypoint component script (via Vector3)

/

## Controls

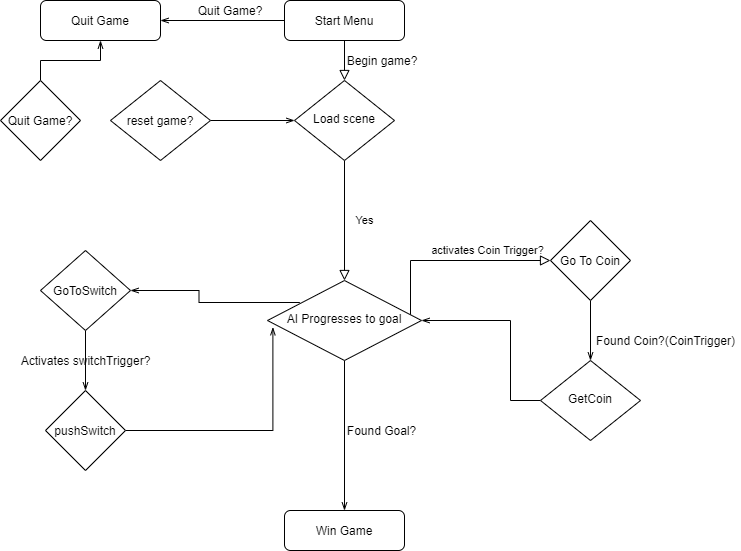
/ Use the mouse to move your cursor over a button in order to interact with UI elements. /

### Mappings

|  |  |  |  |
| --- | --- | --- | --- |
| **Control** | **Function** | **Device** | **Mappable** |
| LMB (Left Mouse Button) | Interact with UI Elements (Buttons) | Mouse | No. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# System Architecture

/ If the design consists of a collaboration between multiple large-scale components, list those components here — or better, include a diagram [UML]. /

Logic Flow Chart.

UML

## 

## Data types

/

* Float – to handle position of objects
* Vector3 – same as above just with (essentially) 3 floats
* Int – to declare the current Waypoint

. /

## Data Model

/ N/A as the data is not saved/ stored anywhere. /

## Interface/API Definitions

/ StateMachines script component controls the behaviour of the AI, this has 3 possible states- Progressing where it follows the waypoints in the Waypoints Script using the vector 3 position of an empty game object

GoToCoin – where it follows a waypoint (named craypoint) in the CoinWaypoint script using another vector3 position.

GoToSwitch – where it follows a waypoint(named swaypoint) based on the SwitchWaypoint script’s vector 3 Game Object /

## Impact

/ The impact of the design runs smoothly. As it’s shared on github, a different user can download the project and may be able to comprimise their own version of the game.

/

## Risks

/ A risk is the player being unahppy with not being able to control their camera in a desired way, as it is automatically following the AI/

## Alternatives

/ To use the FlyCam in order to move the camera around freely was considered, but was not implemented in the game. /

# Shader

## N/A