Project Proposal

Anakin breaking bad

Abstract

This proposal outlines the development of a grid-based game in which a bot, named Anakin, navigates through a m x n environment populated with obstacles, enemies, and two distinct goal states: the Light Side and the Dark Side. Anakin's decision-making is influenced by a set of intrinsic parameters - Trust, Health, and Compassion. Initially, Anakin is oriented towards the Light Side. The primary objective of the player is to strategically guide Anakin toward the Dark Side using user-provided assertion statements. These statements can be either affirmative, such as "You are a good person," or negative, like "You are a ruthless person." The simulation concludes when one of the three end states is achieved: Anakin aligns with the Light, succumbs to the Dark, or perishes.

Introduction

In this project, we delve into the creation of a captivating grid-based game centered around Anakin, an autonomous bot navigating through a dynamically evolving environment. The game intricately combines elements of strategy, sentiment analysis, and artificial intelligence to present players with a unique and immersive experience. Anakin's journey unfolds within a maze-like grid, featuring obstacles, enemies, and two divergent goals—the Light Side and the Dark Side. Players are entrusted with the strategic guidance of Anakin, whose decision-making process is influenced by intrinsic parameters like Trust, Health, and Compassion. What sets this game apart is the incorporation of user-provided assertion statements, allowing players to shape Anakin's moral alignment by manipulating the Compassion parameter. As Anakin progresses, players witness the consequences of their choices, leading to one of three potential end states: alignment with the Light Side, succumbing to the Dark Side, or a perilous demise.

Our project encompasses a multifaceted approach, combining Python programming, game development principles, and artificial intelligence techniques. From grid generation and path finding algorithms to reinforcement learning with MDP and value iterations, the technical foundation is robust. The integration of sentiment analysis through a custom Sentiment Classifier introduces a dynamic element, where user-provided statements directly impact the bot's moral compass. With the Pygame library facilitating graphical interactions and visualization, our game not only challenges players strategically but also immerses them in a narrative-driven environment where choices matter. This report provides an in-depth exploration of our game's architecture, algorithms employed, and the seamless fusion of gaming, AI, and sentiment analysis.

Methods

The project encompasses several methods and techniques, integrating elements of game development, artificial intelligence, sentiment analysis, and reinforcement learning. Here's an overview of the key methods employed.

**Grid based game:**

Grid Generation: This sets the stage for the grid-based game, defining the environment's structure, obstacles, goals, and enemies. The grid which is NxN can be defined before start of the game. After the grid is set Anakin’s start state, light state and dark states are defined and with a small probability some blocks are changed to unreachable in the given grid. At this juncture we employ **BFS** to check if there is a valid path to the light and dark side. If no path is found then we restart the grid generation.

Once a grid is made with all states defined and validity of the paths are checked, with a probability we find the shorted path to the light side using **A\*.** We then start placing young lings all over the grid except along the light state, dark state, start state, unreachable blocks and the shortest path to the light state. This is to ensure that there is a path to the light side in the beginning of the game for Anakin to follow without killing anyone. Once all of the younglings are set the grid is ready.

Bot Navigation: The GridSolver.py file defines the methods for the Anakin bot's navigation within the grid, implementing a Q-learning algorithm to determine optimal actions based on current states.