

***PROGRAMA 4 – Algoritmo de A\* Pathfinding***

**Asignatura:** INTELIGENCIA ARTIFICIAL

**Profesor:** D. Sc. Gerardo García Gil

2022-B

***López Arellano Ricardo David***

Universidad de Guadalajara (CUCEI)

# Presentación

La Inteligencia Artificial (IA) es una rama de la informática que estudia procedimientos automatizados para lograr que un autómata logre ser o parecer inteligente. Éstos, tratan de imitar diversas áreas del comportamiento humano, con el fin de acercarse o llegar a superar a una persona común y corriente, por ejemplo, un experto en cierta materia, en la toma de decisiones o diversas tareas que un sistema computacional puede realizar mejor o más rápido. Es en este contexto que la inteligencia artificial se hace presente en áreas como la robótica y videojuegos, en donde se busca lograr un comportamiento inteligente en un robot o un personaje virtual. Un aspecto básico que se tiene que cumplir, en ambas áreas, es ser capaz de moverse de un punto a otro, esquivando, de forma inteligente y natural, obstáculos que se pueden presentar. Para resolver esta problemática existe un área de la inteligencia artificial llamada Pathfinding.

# Introducción

# Antecedentes

# Desarrollo

# Implementación

# Experimentos

# Resultados

# Conclusión

Pathfinding es el área de la inteligencia artificial que busca encontrar el mejor camino de un punto a otro en mapas representados digitalmente. Se han desarrollado distintos algoritmos con el objetivo de aminorar el tiempo y los recursos utilizados en esta tarea. En este proyecto se busca probar el rendimiento de uno de estos algoritmos, JPS (Jump Point Search) desarrollado por Daniel Harabor y Alban Grastien. El algoritmo se compara con A\*, utilizando tres distintas heurísticas.

# Referencias

# Otro

# coding: utf-8

# Your code here!

import tensorflow as tf

import numpy as np

main (){

celsius = np.array([-40,-10,0,8,15,22,38], dtype = float)

fahrenheit = np.array([-40,14,32,46,59,72,100], dtype = float)

oculta1 = tk.keras.layers.Dense(units = 1, input\_shape = [1])

oculta2 = tk.keras.layers.Dense(units = 3)

salida = tk.keras.layers.Dense(units = 1)

modewlo = tk.keras.layers.Dense([oculta1,oculta2,salida])

modelo.compile{

optimizer = tk.keras.optimizer.Adam(0.1),

loss = 'mean\_squared\_error'

}

print("Comenzando entrenamiento...")

historia = modelo.fit(celsius,fahrenheit,epochs=1000,verbose=False)

print("Modelo entrenado!")

import matplotlib.pyplot as plt

plt.xlabel("Epoca: ")

plt.ylabel("Magintud de pérdida: ")

plt.plot(historia.history["loss"])

print("Hagamos una prediccion!")

resultado = modelo.predict([100.0])

print("El resultado es " + str(resultado) + "fahrenheit!")

print("Variable internas del modelo")

#print(capa.get\_weights())

print(oculta1.get\_weights())

print(oculta2.get\_weights())

print(salida.get\_weights())

}