Workshop No.2

Buitrago, Erick
Student ID: 20221020072

Jiménez, Juan
Student ID: 20221020087

Universidad Distrital Francisco José de Caldas

May 2025

Contents

Abstract

This document outlines the development of a system dynamics model for a two-agent game, inspired by Tron, focusing on agent behavior and interaction within a simulated environment. The initial model defines agent states , actions, environment dynamics, and agent observation. A basic reward system is implemented, providing small rewards for survival and significant rewards and penalties for winning or losing, respectively. Further refinement of the feedback loop is proposed to enhance agent intelligence, stability, and convergence. This includes introducing rewards for aggressive play and fluid movement. Finally, potential frameworks and libraries for future implementation are briefly mentioned.

Keywords: System Dynamics, Agent-Based Modeling, Reinforcement Learning.

1. SYSTEM DYNAMICS ANALYSIS

System Dynamics is an important aspect to keep in mind while doing these types of projects. "Since system dynamics has been developed mainly for providing direct insight on the structural mechanism of complex systems, we believe that system dynamics is one of the most promising methodologies for understanding multi-agent dynamics (MAD)"(Kim and Juhn, s.f., par. 2).

1.1 Mathematical/Simulation Model

In this section we will find various equations that are based on how agents develop and interact in their environment, including their movement, observation of their environment, and reward depending on time. The equations are focused more on the parameters they have giving some time-dependent factors, rather than a specific function.

1.1.1 Agent's State

$$S_t = [x_t, y_t, d_{x_t}, d_{y_t}]$$

The previous equation defines the state of an agent as a vector, where x_t represents the agent's coordinates on the abscissa axis, and y_t its coordinates on the ordered axis on time t. The parameters d_{x_t} and d_{y_t} represent the two components of the agent's directions, e.g., if the agent is moving to the right, the values would be $[1, 0]$, and if it is going up, the values would be $[0, -1]$.

1.1.2 Agent's Actions

$$A_t \in \{U, L, D, R\}$$

A_t is a variable that represents the action of the agent, which could be up, left, down, or right (U, L, D, R, respectively). It also affects the direction of the agent.

$$d_{t+1} = f(d_t, A_t)$$

That equation represents the value of the next direction. It takes as a parameter the current direction (d_t) and the action that the agent takes (A_t).

1.1.3 Environment Dynamism

As the game starts, each agent position will be updated based on their direction, this is represented by the two following equations:

$$x_{t+1} = x_t + d_{x_t}$$
$$y_{t+1} = y_t + d_{y_t}$$

Another thing that affects the environment is the light trail agents leave as they move. The light trail is designed to be a queue with a maximum amount, as it depends on the movement of the agent it is a time-dependent factor, and its equation would have the previous "segment" of the trail and the current position.

$$L_t = L_{t-1} \cup \{x_t, y_t\}$$

1.1.4 Agent Observation

The observation the agent has of its environment is done by an observation matrix, which will be composed of 8 basic layers, the first one has the information of the borders, the second layer has the position of the first agent, then the third layer the position of the second player, the fourth layer has the positions of the light trails. Finally, the fifth and sixth layers are for the direction of the player 1 (one for the x direction, and the other for the y direction), and identically the seventh and eighth layers for the player 2 direction.

"In agent-based modeling (ABM), a system is modeled as a collection of autonomous decision-making entities called agents. Each agent individually assesses its situation and makes decisions on the basis of a set of rules"(Bonabeau, 2002, par. 2).

1.1.5 Reward

The agent will receive little rewards for certain things, for second it is alive, it will get $+0.5$, and when on victory the agent will get $+10$, the other one as it loses the game will receive a penalization of -10.

For a special case, like a draw, where both agents die at the same time they both will get $+0.5$.

Better rewards will be discussed further in the feedback loop refinement section, like accomplishing an aggressive play style or avoiding cycling movement.

1.2 Phase Portrait

The game will start with agent 1 on the left side and agent 2 on the right side. The first move of both agents will be toward the other side of the board. Then, they will decide how to move. They can fall into walls or any light trail. The agents cannot turn back in one move, for example, if an agent is moving to the right, it cannot turn left with a single input. All of this can be seen on the Figure 1 below.

"The AI agent learns optimal decision-making by interacting with the game environment, analyzing states, and receiving rewards"(Mohammed, Geeth, Kumar, Sagar, Sankar & Karwa, 2024,

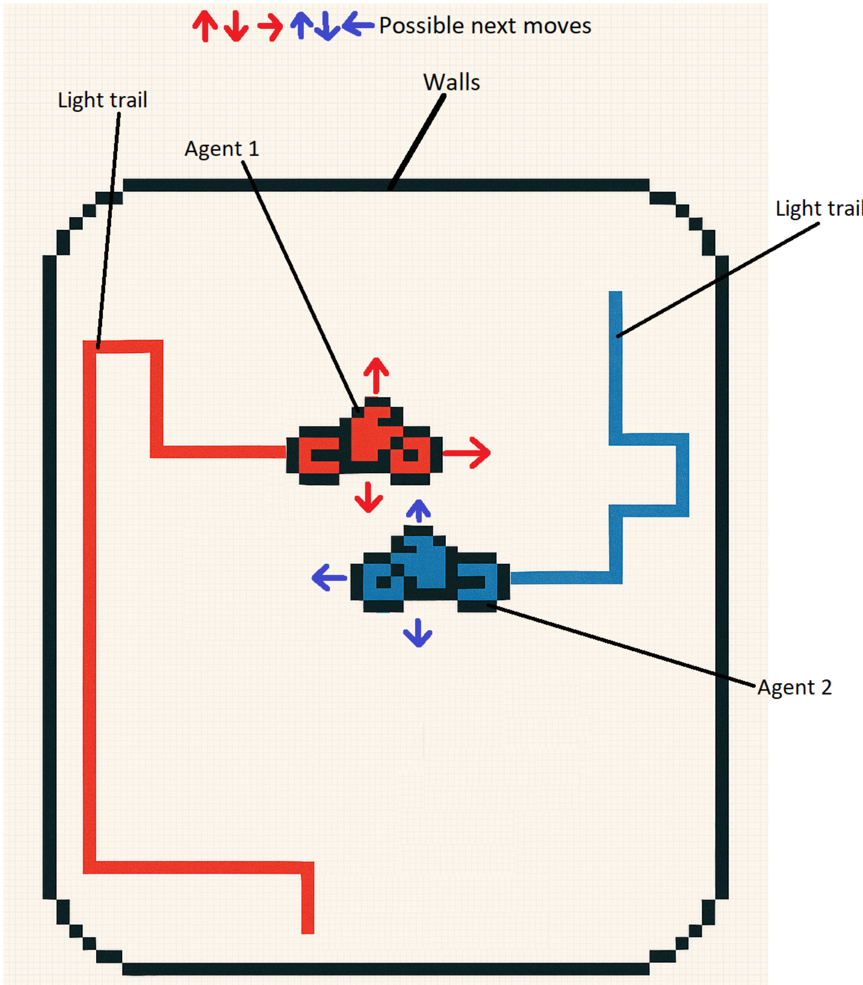

Figure 1: Example of Tron game board with agents and light trails

par.1).

2. FEEDBACK LOOP REFINEMENT

2.1 Enhanced Control Mechanisms

To enable an intelligent, stable, and convergent behavior, we may add more forms of rewards to incentivize some behaviors and avoid others.

We can add a reward for attacking the enemy, to encourage aggressive behavior. The agent will be rewarded when the enemy has few free spaces to move, that is, when the agent is cornering the opponent. This reward will complement the reward for the enemy's death.

Another new reward may be to promote fluid movement, avoiding clumsy or cyclic behavior. It will compare the player's previous and current direction if it is the same, the agent will be rewarded with a light reward, but if the agent makes a 180° turn, it will be penalized lightly (this way, if it is to avoid a wall, it will prefer to do a 180 degree turn rather than dying, as it has a bigger penalization).

2.2 Stability and Convergence

The agent is potentially stable and convergent due to the structured and informative observation space, along with a reward function that encourages desirable behaviors. The observation tensor provides key spatial and strategic information:

- **Layer 0 (walls):** Helps the agent avoid collisions with the game walls.
- **Layers 1 and 2 (player positions):** Allow the agent to track its own location and the opponent's, which is essential for both evasion and offensive strategies.
- **Layer 3 (light trails):** Allows the agent to avoid paths that would result in self-collision or collision with the opponent's trail.
- **Layers 4–7 (opponent direction):** Provide the agent with information on the intentions of the opponent's movement, allowing predictive decision-making (e.g., flanking or blocking).

The **reward system** further guides the learning process by reinforcing beneficial behaviors:

- **+15 reward** when the opponent crashes, encouraging aggressive or strategic play.
- **-15 penalty** when the agent crashes, discouraging reckless behavior.
- **+0.5 per second of survival**, promoting careful navigation and risk management.
- (Optionally) bonuses for **surrounding the opponent (+5 reward)** or **fluid movement**. However, if the agent makes a 180° turn, it will be penalized with -5.

This combination of a rich, multi dimensional observation space and a well designed reward structure increases the likelihood that the reinforcement learning algorithm will converge to a stable and effective policy.

3. ITERATIVE DESIGN OUTLINE

3.1 Potential Frameworks and Libraries

It was identified that there is a need to add a better reward system to the project in order to improve the agent's behavior and decision-making, enhance performance, and accelerate the learning process. This system should not only penalize defeat and reward victory, but also promote strategic and intelligent behavior. Also, some neural networks can be integrated, such as:

and frameworks like: and libraries like:

4. REFERENCES

Bonabeau, E. (2002, May 14).Agent-based modeling: Methods and techniques for simulating human systems.*Proceedings of the National Academy of Sciences, 99*.(Suppl. 3), 7280-7287. 1 Retrieved from https://pmc.ncbi.nlm.nih.gov/articles/PMC128598/

Kim, D. & Juhn, J. (n.d.).*System Dynamics as a Modeling Platform for Multi-Agent Systems*.Retrieved from https://proceedings.systemdynamics.org/1997/kim-dong.htm

Mohammed, T. K., Geeth, M. S., Kumar, K. S., Sagar, N. S., Sankar, S. S., & Karwa, M. (2024). AI Agent For Multiplayer Games Using Reinforcement Learning. *International Journal of Creative Research Thoughts (IJCRT), 12(5)*, 4496-4506. Retrieved from https://www.ijcrt.org/papers/IJCRT2411646.pdf