# Workshop 2 – Dynamical Systems Analysis & Design

Cesar Augusto Pulido Cuervo – 20222020048
Cristian David Romero Gil – 20222020138
Universidad Distrital Francisco José de Caldas
System Sciences Foundations
Eng. Carlos Andrés Sierra, M.Sc.

## 1 System Dynamics Analysis

**State representation of Wumpus World:**

$$s_t = \langle x, y, b, s, g, a, w, t \rangle$$

**Variables that define the system state:**

- $x$: X coordinate
- $y$: Y coordinate
- $b$: Binary percept for breeze
- $s$: Binary percept for stench
- $g$: Binary percept for glitter
- $a$: Is arrow available?
- $w$: Is Wumpus alive?
- $t$: Time step

### 1.1 System Dynamics Model

**Temporal Structure**

The process occurs in time steps. At each time step:

1. The agent performs an action.
2. The position updates based on that action.
3. The new position leads to new percepts (stench, breeze, etc.).
4. A new decision is made, resulting in a new action.
5. The agent receives a reward, influencing future behavior.

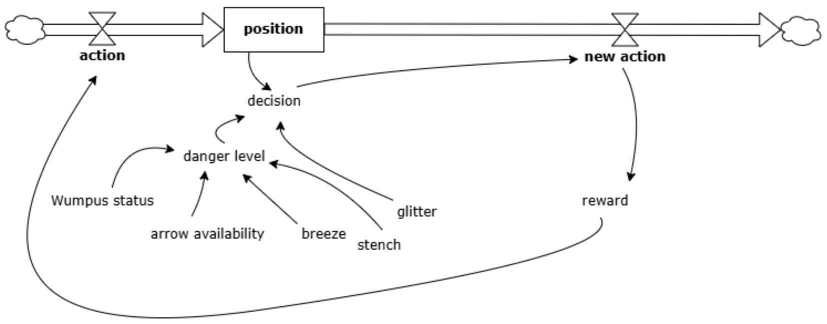

Figure 1: Causal diagram

**Loop**

This is a closed loop system; the result of each action influences the next cycle. The model implicitly supports learning or adaptive behavior.

## 2 Feedback Loop Refinement

### 2.1 Enhanced Control Mechanisms

To improve the agent's responsiveness to changes in the environment, we propose to introduce new control mechanisms through both additional sensors and more granular reward structures.

#### a. Additional Sensor: `danger_level`

A continuous variable named `danger_level` is introduced to represent the immediate threat level around the agent. It is computed based on the number of adjacent cells containing pits or the Wumpus. The value ranges from 0.0 (no risk) to 1.0 (maximum danger), offering a more nuanced input for decision-making than the binary percepts (breeze, stench).

This new sensor is added to the observation space and used as an input to the neural network policy (DQN). It allows the agent to distinguish between mildly dangerous and highly dangerous zones, facilitating smoother behavioral transitions.

#### b. Granular Reward Signals

The original reward structure was discrete and sparse. We introduced the following additional signals to increase feedback frequency and support learning:

| **Event** | **Updated Reward** |
|---|---|
| Visiting a new, safe cell | +5 |
| Revisiting the same cell | -0.5 |
| Passing near danger (pit/Wumpus) but surviving | +10 |
| Missed arrow shot (penalty reduced) | -25 |

These changes provide lower-magnitude, frequent signals that reinforce desirable behavior such as exploration and cautious movement, while discouraging redundant or risky choices.

## 2.2 Stability on Convergence Criteria

### 2.2.1 Bounded Behavior

- The agent should not exhibit uncontrolled or erratic behavior such as infinite loops, excessive wandering, or oscillating between actions.
- Reinforcement learning models like Q-learning and DQN ensure that the agent eventually learns policies with bounded action choices, especially when experience replay and target networks are used to stabilize learning (Workshop 1).
- The agent incurs a step cost (reward $-1$) for every action. This discourages excessive movement and promotes efficient paths.

### 2.2.2 Convergent Learning Behavior

Use of reward curves, episode length, Q-value variance, and victory rate to assess convergence over time. Testing across multiple random seeds and environment layouts confirms whether learning consistently converges.

# 3 Iterative Design Outline

## 3.1 Project Plan Update

To implement dynamic behavior and adaptive control mechanisms, we plan to update the architecture and logic as follows:

| **Component** | **Update** |
|---|---|
| Observation Space | Augmented with `danger_level` and a rolling memory of the last two perceptual vectors. |
| Reward Function | Enhanced with small positive/negative signals as described above. |
| Agent Network (DQN) | Input layer updated to accept expanded observation vectors. |
| Environment Logic | Tracks visited cells and calculates dynamic risk zones in real-time. |
| Training Code | Processes augmented states and updated rewards. Includes random seeds. |

## 3.2 Testing Strategy

To evaluate these changes, we propose the following testing strategy:

| **Aspect** | **Testing Strategy** |
|---|---|
| Random Seeds | Each training configuration is run across at least 10 different seeds to measure variance in outcomes. |
| Chaos Regimes | Tested with environments using varied layouts and perceptual noise to simulate dynamic volatility. |

| | |
|---|---|
| Scenario Variations | Multiple Wumpus/pit layouts used per run to assess policy generalization. |
| Perturbation Tests | Small changes in initial conditions to observe long-term trajectory divergence. |
| Metrics Tracked | Reward curves, victory rate, episode length, revisits, and Q-value stability. |

This design and testing outline is intended to ensure that the Wumpus agent evolves from a static learner into a dynamic, chaos-aware decision-making system.