جامعة الأخوين

ⵜⴰⵙⴷⴰⵡⵉⵜ ⵏ ⵓⵅⴰⵡⴰⵢⵏ

AL AKHAWAYN UNIVERSITY

Project 3: Uncertainty

School of Science and Engineering

Meriem Lmoubariki

Mohamed Adam Sterheltou

Dr. Tajjeeddine Rachidi

Al Akhawayn University in Ifrane

Table of Contents

Introduction

The "Bust the Ghost" project explores probabilistic reasoning and Bayesian inference through an interactive AI-based game. Players locate a hidden ghost on an 8x13 grid by interpreting sensor feedback and dynamically updating probabilities after each interaction. The game incorporates advanced features such as a directional sensor and color-coded hints to enhance gameplay while providing an educational platform for applying AI concepts.

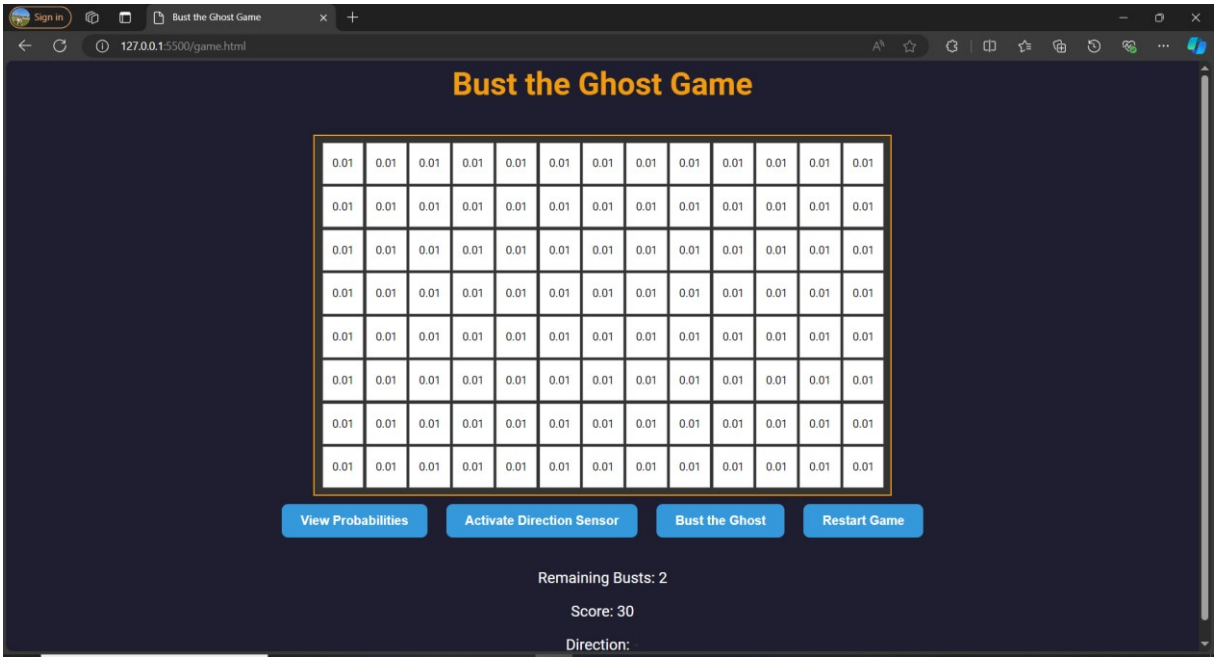
I. Design of game

The setup

At the start of the game, the ghost is equally likely to be in any of the cells of the 8x13 grid. Therefore, the initial probability P(G)P(G)P(G) of the ghost being in any specific cell is calculated by dividing 1 by the total number of cells(Instructor-provided project document, 2024):

$$P(G)=\text{total number of cells}1=8\times131\approx0.0096$$

For simplicity, this is often rounded to 0.01 for each cell when communicated in the game interface. For this reason, we initialized the grid cells to 0.01.



Grid design:

The game grid is an 8x13 matrix that serves as the playing area. Each cell in the grid represents a potential location for the ghost, which is hidden at the start of the game. The grid layout is implemented using HTML and styled with CSS for visual clarity and interactivity. Initially, all cells are assigned a uniform probability, indicating an equal likelihood of the ghost being in any cell.

```
const cols = 13;
const rows = 8;
```

```
let probabilities = Array.from({ length: rows }, () => Array(cols).fill(1 / (rows * cols)));
```

Ghost placement:

The ghost is randomly placed on the grid at the beginning of each game using the placeGhost() function. This function generates random coordinates (x, y) within the grid dimensions (8 rows and 13 columns). The prior probability distribution for the ghost's location is initialized uniformly.

```
// Randomly place the ghost
function placeGhost() {
    const x = Math.floor(Math.random() * cols);
    const y = Math.floor(Math.random() * rows);
    console.log(`Ghost placed at: (${x}, ${y})`);
    return { x, y };
}
```

Probabilistic Modeling

Sensor feedback:

When the player clicks a cell, they receive a color-coded hint based on the Manhattan distance between the clicked cell and the ghost's actual location. The feedback is determined as follows:

Red: The ghost is in the clicked cell (distance = 0).

Orange: The ghost is 1-2 cells away.

Yellow: The ghost is 3-4 cells away.

Green: The ghost is 5 or more cells away.

(P (E | G) based on sensor feedback):

Distance	Red	Orange	Yellow	Green
0	1.0	0.0	0.0	0.0
1-2	0.0	0.8	0.2	0.0
3-4	0.0	0.0	0.6	0.4
5+	0.0	0.0	0.0	0.9

```
}

// Determine color based on distance
function senseDistance(distance) {
    if (distance === 0) return "red";
    if (distance <= 2) return "orange";
    if (distance <= 4) return "yellow";
    return "green";
}
```

Bayesian update:

The game employs Bayesian inference to update the probability distribution of the ghost's location after each click. The formula used for updating probabilities is (Murphy, 2023):

$$P_t(G = L_i) = \frac{P(S = \text{Color at } L_i | G = L_i) \cdot P_{t-1}(G = L_i)}{\text{Normalization Factor}}$$

Where:

- $P_t(G = L_i)$: Updated probability of the ghost being in location L_i.
- $P(S = \text{Color at } L_i | G = L_i)$: Likelihood of the observed color given the ghost's location.
- $P_{t-1}(G = L_i)$: Prior probability before the observation.
- Normalization Factor: Ensures that the updated probabilities sum to 1.

The likelihood of the color feedback depends on the predefined conditional probability tables for each color at different distances.

```
// Update probabilities using Bayesian inference
function updateProbabilities(clickedX, clickedY, color) {
    const likelihood = Array.from({ length: rows }, () => Array(cols).fill(0));
    let totalProbability = 0;

    for (let y = 0; y < rows; y++) {
        for (let x = 0; x < cols; x++) {
            const distance = calculateDistance(x, y, clickedX, clickedY);
            const validDistances = {
                red: [0],
                orange: [1, 2],
                yellow: [3, 4],
                green: [5, 6, 7, 8, 9, 10]
            }[color];

            likelihood[y][x] = validDistances.includes(distance) ? 1 : 0;
        }
    }

    for (let y = 0; y < rows; y++) {
        for (let x = 0; x < cols; x++) {
            probabilities[y][x] *= likelihood[y][x];
            totalProbability += probabilities[y][x];
        }
    }

    if (totalProbability > 0) {
        for (let y = 0; y < rows; y++) {
            for (let x = 0; x < cols; x++) {
                probabilities[y][x] /= totalProbability;
            }
        }
    } else {
        console.warn("Probabilities collapsed. Resetting to uniform distribution.");
        probabilities = Array.from({ length: rows }, () => Array(cols).fill(1 / (rows * cols)));
    }
}
```

Logic implemented

The player starts with a fixed score and a limited number of "bust attempts." Each click reduces the score by 1 point. If the player clicks on the ghost's cell, they receive a "red" feedback and can attempt to bust the ghost(Instructor-provided project document, 2024). If they run out of score or bust attempts, the game ends.

Direction sensor:

The direction sensor was introduced as an additional feature to enhance the gameplay and aid the player in locating the ghost. This sensor provides directional hints about the ghost's position relative to the clicked cell. The possible outputs of the direction sensor include:

- **N (North)**: The ghost is directly above the clicked cell.
- **S (South)**: The ghost is directly below the clicked cell.
- **E (East)**: The ghost is to the right of the clicked cell.
- **W (West)**: The ghost is to the left of the clicked cell.2.
- **NE, NW, SE, SW**: The ghost is in a diagonal direction relative to the clicked cell.
- **HERE**: The ghost is in the clicked cell.

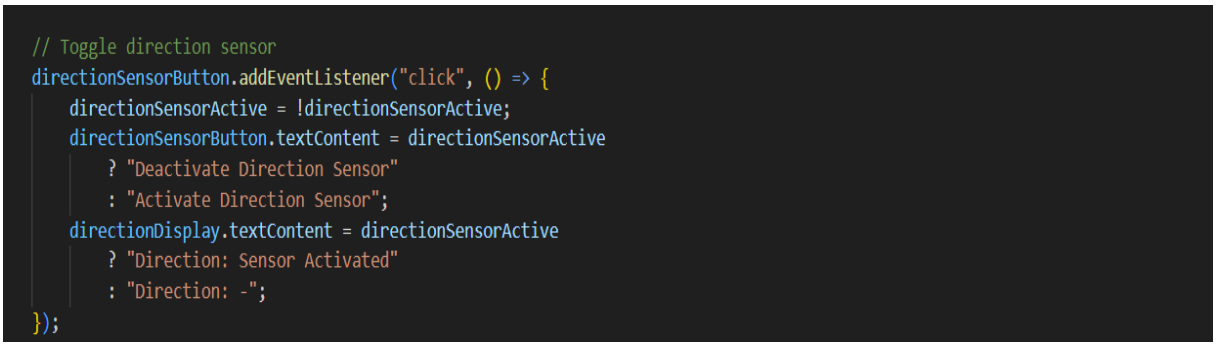
This directional feedback is independent of the distance sensor, adding a second layer of probabilistic reasoning.

```
// Get direction based on player and ghost positions
function getDirection(playerX, playerY, ghostX, ghostY) {
    if (playerX < ghostX && playerY < ghostY) return "SE";
    if (playerX < ghostX && playerY > ghostY) return "SW";
    if (playerX > ghostX && playerY < ghostY) return "NE";
    if (playerX > ghostX && playerY > ghostY) return "NW";
    if (playerX < ghostX) return "S";
    if (playerX > ghostX) return "N";
    if (playerY < ghostY) return "E";
    if (playerY > ghostY) return "W";
    return "HERE";
}
```

Conditional Probabilities :

The direction sensor operates based on conditional probability tables similar to the distance sensor. Each direction has a probability of being observed depending on the ghost's actual location relative to the clicked cell. For example: If the ghost is to the north of the clicked cell, the sensor is more likely to output **N** but may occasionally output adjacent directions due to noise or uncertainty.

These probabilities are predefined and integrated into the Bayesian inference mechanism to update the ghost's location probabilities more accurately.



```
// Toggle direction sensor
directionSensorButton.addEventListener("click", () => {
    directionSensorActive = !directionSensorActive;
    directionSensorButton.textContent = directionSensorActive
        ? "Deactivate Direction Sensor"
        : "Activate Direction Sensor";
    directionDisplay.textContent = directionSensorActive
        ? "Direction: Sensor Activated"
        : "Direction: -";
});
```

The conditional probabilities of the direction sensor :

Actual Direction	**N**	**S**	**E**	**W**
N	0.9	0.05	0.025	0.025
S	0.05	0.9	0.025	0.025
E	0.025	0.025	0.9	0.05
W	0.025	0.025	0.05	0.9

Results and Observations

https://youtu.be/nO9lVMx8nlk

Results are in our demo video

Division of Tasks

Task	**Meriem Lmoubariki**	**Mohamed Adam Sterheltou**
Grid Design	Collaborated	Collaborated
Sensor Implementation	Worked on distance sensor	Worked on direction sensor
Bayesian Updates	Collaborated	Collaborated
Testing and Debugging	Worked on GUI then collaborated for logic and probabilities	Worked on Logic then collaborated for the interfaces
Report Writing	Both of us	Both of us
Presentation Preparation	Collaborated	Collaborated

Acknowledgments

We would like to express our sincere gratitude to **Dr. Tajjeeddine Rachidi** for his invaluable guidance and support throughout this project. This assignment was one of the most engaging and insightful projects we have undertaken, showcasing practical applications of AI concepts. We are proud to include it in our academic portfolios as a testament to our learning experience.

References

- **Instructor-provided project document, *"Project #3: Bust the Ghost Game,"* 2024.**
- **OpenAI, *ChatGPT Assistance for Bust the Ghost Game Project Report,* November 2024.**
- **MDN Web Docs, *"JavaScript Basics,"* Mozilla Developer Network. Available at: https://developer.mozilla.org.**
- **FreeCodeCamp, *"Responsive Design and CSS Basics,"* FreeCodeCamp. Available at: https://www.freecodecamp.org.**
- **Murphy, K., *"Bayesian Reasoning,"* 2023. Available at: https://example.com/bayesian-reasoning**