Project 2
Cluedo Game

Abanob Tanous

CS670: Artificial Intelligence
Professor Arashdeep Kaur
December 8, 2024

Abstract

The purpose of this project was to design and develop of digital adaptation of the classic murder mystery board game, Cluedo. The game is set in a mansion where players must deduce the culprit of a murder along with the weapon they used and the room where it happened. This digital implementation of the game involved game setup, player movement, and core game play mechanics. Game setup included elements such as creating the mansion layout, defining characters and weapons, and selecting a solution. Player movement enables the players to navigate between rooms. Along with movement, the other two core game play mechanics are suggestion and accusation. In a suggestion, the player suggests a possible solution and a player, going in clockwise order, must refute they suggestion with one of their cards if able. Accusation is the mechanic which allows the player to put in a final guess at the solution, if they are right they win, if they are wrong they are disqualified from the rest of the game.

This implementation was written in python 3. The pygame library was used to provide windowing and graphics capabilities. The pygame_gui library was used to provide user interface capabilities which work on top of pygame. Some modifications were made to the base game to make it more congruent with this project's requirements and the implementation restrictions.

Introduction

Cluedo, known as Clue in North America, is a classic murder mystery board game which, through its combination of strategy, deduction, and simplicity, has made itself a timeless favorite among many players of all ages. The premise of the game is simple and engaging, use clues gathered through suggestions to deduce the murderer, his weapon, and the murder location. With this context in mind, the motivations for a digital implementation of this game are as follows:

1. Digitize a timeless classic: As previously mentioned, Cluedo is a timeless classic and many people continue to enjoy even today. By bringing this game into the digital world, access to the game will broaden out to a much larger potential audience. Many people might not have the opportunity to purchase a physical board game, or many not consider purchasing a physical board game. A digital version of the game will allow those people the opportunity to enjoy Cluedo. Furthermore, it will remove some restraints that come with a physical board game. For example, players will be able to play anywhere anytime so long as they have a digital device, like phones which are very prolific.

2. Demonstrate and sharpen technical skills: Given the premise and core mechanics of the game, it is clear that a digital implementation would depend on important programming concepts and skills, especially those relevant to Artificial Intelligence. For example, game setup requires randomization and state management. The non-user players require an AI agent controlling their moves. Such an agent must be able to keep track of the game state, record learned clues in its knowledge base, and use is knowledge base to make informed game play decisions. Processing the game play will require logic handling, such as handling the suggestion, refutation, and accusation mechanics. Evidently, it is clear that this project will showcase and sharpen programming skills, especially those pertaining to Artificial Intelligence.

Game Rules

This implementation of Cluedo is based on the 1949 version of Cluedo, using its characters, weapons, and mansion layout as well as a slightly modified version of its rules.

Characters:

1. Miss. Scarlett
2. Colonel Mustard
3. Mrs. White
4. Rev. Green
5. Mrs. Peacock
6. Professor Plum

Above are the iconic original characters who serve as the murder suspects in this game.

Weapons:

1. Candlestick
2. Spanner
3. Rope
4. Lead Piping
5. Dagger
6. Revolver

Above are the weapons which serve as the murder instrument.

Layout:

Room	Connects to
Ballroom	Kitchen, Conservatory
Billiard Room	Conservatory, Library
Conservatory	Billiard room, Ballroom
Dining Room	Kitchen, Lounge
Hall	Lounge, Study
Kitchen	Dining room, Ballroom
Library	Study, Billiard room
Lounge	Hall, Dining room
Study	Hall, Library

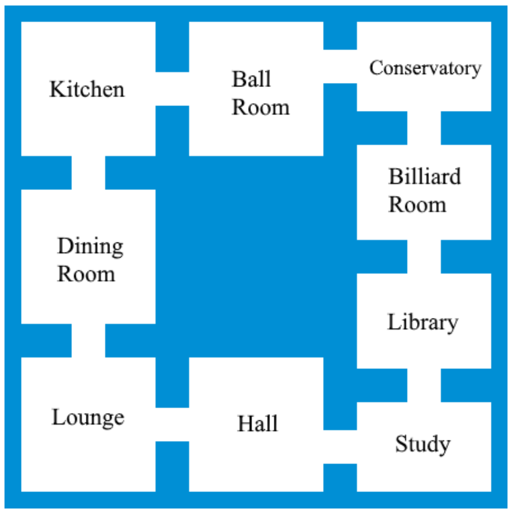


Figure 1: Layout of the Mansion (game board)

Above is the layout of the mansion and all its rooms. This serves as the setting of the game, and players are situated in and navigate between rooms.

Players:

1. Red
2. Yellow
3. White
4. Green
5. Blue
6. Purple

Setup:

1. There are six players, five AI and one user.
2. All of the above mentioned items (rooms, characters, and weapons) are treated as cards. From these cards one of each type is selected, in secret, prior to the start of the game. These will serve as the solution, the answer which the players seek to deduce.
3. The remaining cards are shuffled and distributed to the players. Each one of the six players will receive a total of three cards. These cards are kept in secret and are used in the refutation mechanic. Furthermore, each player should record their cards in their knowledge base since they know those cards cannot be the solution.
4. Players are assigned a color from the above list. Starting from the Red player, players will take turns following the above order. The refutations will also follow that order.

Mechanics:

1. During each players turn they can do three things: movement, suggestion, accusation.
2. A player can elect to move by choosing and adjacent room (see layout above) and their token will move to it. Movement happens first before suggestion and accusation.
3. A player can make a suggestion by selecting a character, a weapon, and a room (it must be the room they are currently in). The suggestion prompts the refutation mechanic.
4. When a player makes a suggestion then starting with the next player in clockwise order (see player order above) players must make a refutation if they are able. A refutation involves showing one of their cards, which are a part of the suggestion, to the suggesting player. For example, if Red suggests Candlestick, Professor Plum, and Ballroom then Yellow will refute with Candlestick (or any of the other cards if they have it).
5. If the next player cannot refute (they don't have any of the cards), then the player after them must refute if able. This continues down the line until a player has refuted the suggestion or no one is able to refute it.
6. When a player feels they know the solution, they may make an accusation with the room (they must be present in it), the weapon, and the character they think is the solution. If they are correct, they win, if they are incorrect, they are disqualified.
7. Disqualified players no longer play except for the purpose of refuting suggestions.

Thorough Testing

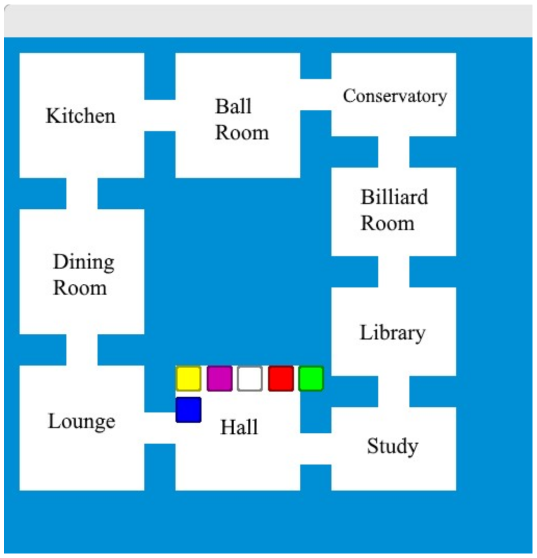
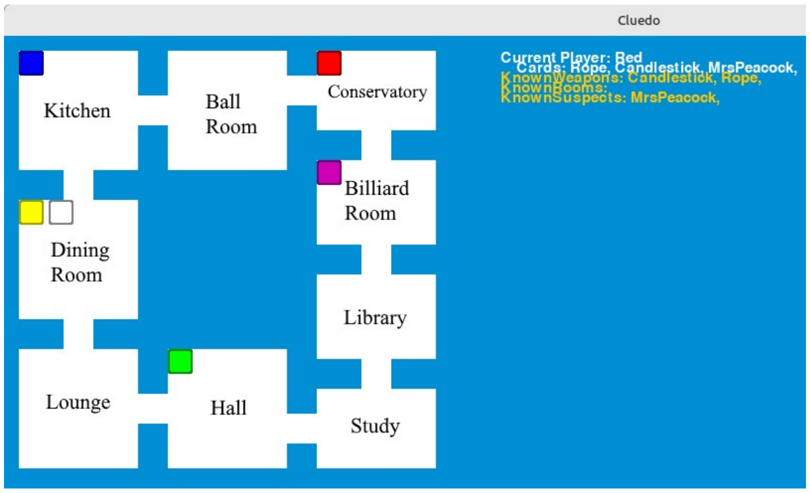


Figure 2: Testing mansion and player token rendering

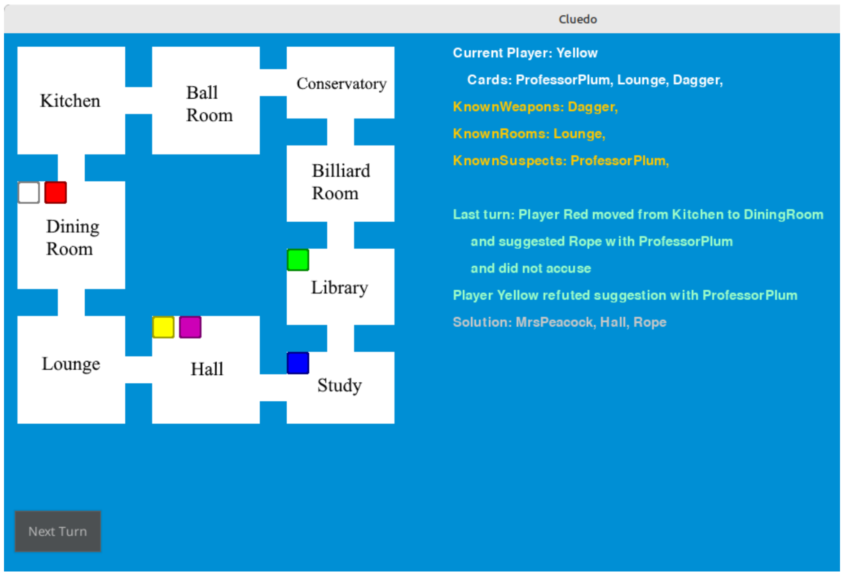
Figure 2 is an early screenshot from when I first started working on the rendering. I wanted to make sure that the mansion was being drawn correctly, which was the case. I also wanted to make sure the player tokens where rendered correctly. At this point in development many of the mechanics and logic were not implemented and as a result all the players were in the same room. This revealed an issue where the tokens where drawn outside the room’s boundaries. This necessitated changes to the player token rendering to fix this issue.



Current Player: Red
Cards: Rope, Candlestick, MrsPeacock,
KnownWeapons: Candlestick, Rope,
KnownRooms:
KnownSuspects: MrsPeacock,

Figure 3: Testing UI info rendering

Figure 3 is later screenshot from when I was working on the information segment of the UI, clearly I had to fix the spacing of the text. It is clear from the screenshot that the mansion and player rendering are good. It is also clear that much of the logic is now implemented, with the cards and knowledge base code being mostly done.

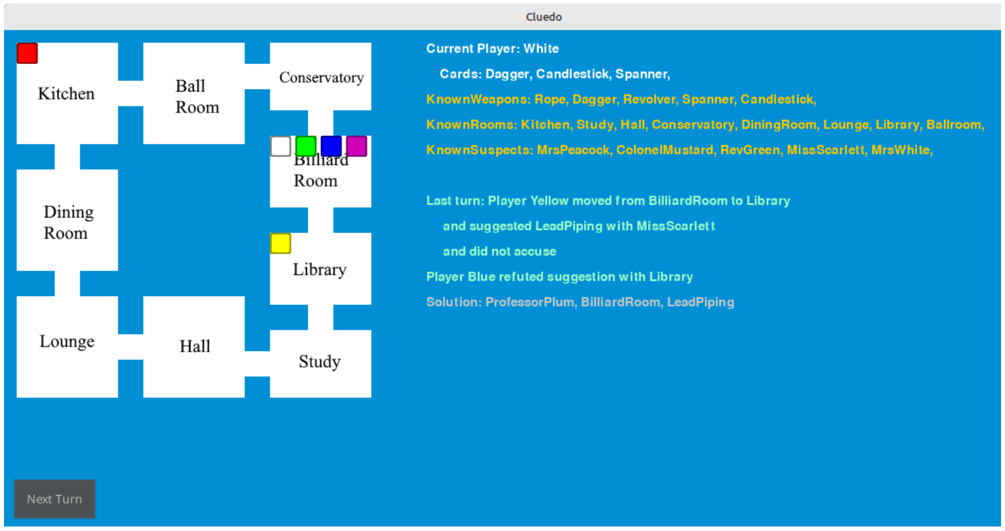


Current Player: Yellow
Cards: ProfessorPlum, Lounge, Dagger,
KnownWeapons: Dagger,
KnownRooms: Lounge,
KnownSuspects: ProfessorPlum,

Last turn: Player Red moved from Kitchen to DiningRoom
and suggested Rope with ProfessorPlum
and did not accuse
Player Yellow refuted suggestion with ProfessorPlum
Solution: MrsPeacock, Hall, Rope

Figure 4: Testing full UI and core mechanics

Figure 4 shows the UI being fully implemented, and the core mechanics are being tested. At this point the movement, suggestion and refutation mechanics were implemented and I was testing them. I found some bugs related to the way I was passing around the room object through this testing. I also fixed some inconsistency issues with the accumulation of player knowledge.

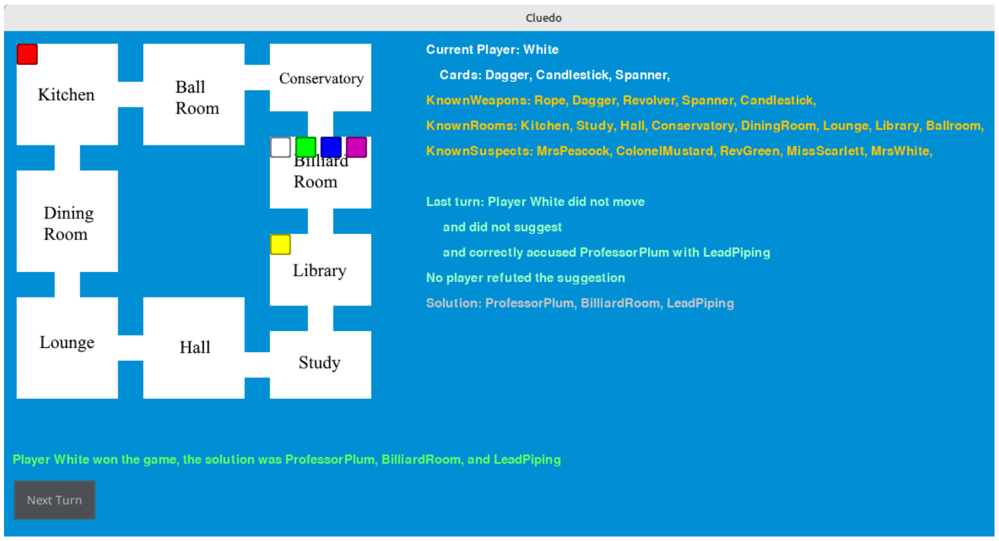


```
Current Player: White
   Cards: Dagger, Candlestick, Spanner,
KnownWeapons: Rope, Dagger, Revolver, Spanner, Candlestick,
KnownRooms: Kitchen, Study, Hall, Conservatory, DiningRoom, Lounge, Library, Ballroom,
KnownSuspects: MrsPeacock, ColonelMustard, RevGreen, MissScarlett, MrsWhite,

Last turn: Player Yellow moved from BilliardRoom to Library
    and suggested LeadPiping with MissScarlett
    and did not accuse
Player Blue refuted suggestion with Library
Solution: ProfessorPlum, BilliardRoom, LeadPiping
```

Figure 5: Testing AI logic

In Figure 5 it is clear that player White has intelligently made their moves to achieve a winning position. They have all the knowledge necessary to deduce the solution. This showed me that the AI agents were able to intelligently make moves and suggestions to gain knowledge. Then I wanted to test if the AI agent would take advantage of a winning position and move to make an accusation. It is clear that player White has what is necessary to deduce the solution and has moved to the solution room. Now will player White make the accusation?



```
Current Player: White
   Cards: Dagger, Candlestick, Spanner,
KnownWeapons: Rope, Dagger, Revolver, Spanner, Candlestick,
KnownRooms: Kitchen, Study, Hall, Conservatory, DiningRoom, Lounge, Library, Ballroom,
KnownSuspects: MrsPeacock, ColonelMustard, RevGreen, MissScarlett, MrsWhite,

Last turn: Player White did not move
    and did not suggest
    and correctly accused ProfessorPlum with LeadPiping
No player refuted the suggestion
Solution: ProfessorPlum, BilliardRoom, LeadPiping

Player White won the game, the solution was ProfessorPlum, BilliardRoom, and LeadPiping
```

Figure 6: Testing accusation mechanic

Figure 6 shows that the AI agent is able to recognize their wining position, make their way to the proper room, and make the accusation. Furthermore, the accusation mechanic works and the game ends correctly with a win for player who made an accurate accusation. At this point in testing, it was clear that the rendering, mechanics, and AI logic were correctly implemented and working well. The only remaining testing task was to eliminate some minor bugs.

Challenges Faced

There were a couple of obstacles that I face during development that ended up limiting the scope of the game:

1. User interface. The biggest obstacle was the user interface. While pygame is an excellent tool, and provides great capabilities for handling graphics and windowing with simplicity, it does no provide an built-in tools for user interface development. As a result, implementing interactive elements like buttons and drop-down menus is difficult and laborious. I attempted to remedy this with the install of the pygame_gui library. Indeed this library did make the buttons a lot easier to create and use, it did not ultimately fix the underlying issues as it did not work too well with the existing code and framework. As a result, the UI did not end up with the capabilities and quality I had originally hoped and planned for. Even more importantly, this UI issue led to some limits on the initial scope of the project. I had initially planned to setup the UI to allow the user to perform the game mechanics of movement, suggestions, refuting suggestions, and making accusations. This would have enabled a user player in the game, however, this did not happen due to he aforementioned UI issues. While this does not take away from the project as the Cluedo board game was still implemented, and AI agents where designed to play the game, it is disappointing.

2. Debugging and testing. Prior to this project, most of my programming experience was with statically typed languages like C and Java, and I was very familiar with that paradigm. I had some experience with Python, and I was knowledgeable of its syntax and rules, but I was not familiar with large project development in Python. Statically-typed languages afford certain information that allowed for, in some ways, easy debugging and testing. This is not the case for Python, where everything is dynamically typed. This led to many issues in debugging and testing as many objects where incorrectly typed, or used incorrectly and only caught during testing. I think if I had setup and environment specifically created for Python development, such as PyCharm, I would have had less struggles with this issue.

Stability and Reliability

As of the final version, the game is very stable and reliable. I had no had any crashes or observed any bugs in the final version. Every test I have performed on the final version has demonstrate valid game mechanics, rendering, and AI logic. The game progresses logically, with turns and guesses being handled properly. The rules are mostly followed, with the system ensuring valid guesses and maintaining turn order. The game reliably ends with a winner and the suggestions and accusations are valid and handled correctly. No visual glitches have been observed, and while the UI is not as polished as I would like it, it is still stable and realiable. In terms of performance, the game is very lightweight and so it runs perfectly well without any hitches, lag, or frame drops. Overall, the game, in its final version, is stable and reliable.

Additional Details

Dependencies:

Python 3+

pygame 2.6+

pygame_gui 0.6+

Modifications:

Some modifications were made to the base game to make it more congruent with this project's requirements and the implementation restrictions. They are:

1. The replacement of the tiles with just rooms and movement being between rooms rather than between tiles.
2. Players no longer assume the identity of one of the characters and are identified with colors, as a result players are no longer moved around when a suggestion is made.