CSCI490 Information System Development

Senior Project Report

To Rem & Back (2D Game)

Semester: Spring 2019

Instructor: Mahmoud Samad

Students:

- Amin Al-Ait (51710023)
- Adnan Jabbado (51230046)

Contents

Introduction 2

Gameplay Briefing 3

Future Considerations 5

Conclusion 6

Introduction

To Rem & Back is a 2D Game that features Random Level Generation and AI as core mechanics. It mixes both the Side-Scroller and Roguelike genres interchangeably to embrace mechanics of both in a single game.
The game's progression revolves around the battle to break from heroin Side effects and how the character's story uncovers across the stages of the game.
The game revolves around the protagonist and his mysterious story after surviving a plane crash and waking up at the hospital with a considerable amount of heroin in his blood because of a mistake by one of the nurses, which triggers a certain state of hallucination, or dreaming, (REM) inside which he will be fighting his way out back to reality.
The game takes inspiration from the roguelike genre. Games like:
Spelunky, Rogue Legacy, Dead Cells.
The game is designed and developed using Game Maker Studio 2 engine and uses its native programming language Game Maker Language (GML).
it will be exported for windows; which houses a large portion of the gaming market.

Gameplay Briefing

At some levels in the game, the player is constrained by the affected character's abilities: only walking and exploring sideways at times, uncovering hints about the truth behind his journey.

The triggers back to REM, however, are assigned to happen at certain events along the way. Key moments in the story are going to drop the player back up in REM where they will be solving puzzles and challenges in order to escape.

In every visit to Rem, the player will experience a new level (since dreams are random and spontaneous), and is left to find his way out of it by solving puzzles, unlocking all features, and evading disturbing figures and characters met earlier during the story. All that under a certain amount of time. If the time runs out, the player will enter Rem again, with another random layout, and will have to solve the puzzles again, creating a huge replay value since the levels are always randomly generated. After retrying a certain level, some puzzles may not be in the same place again and thus adding a sense of mystery and entertainment to the game.

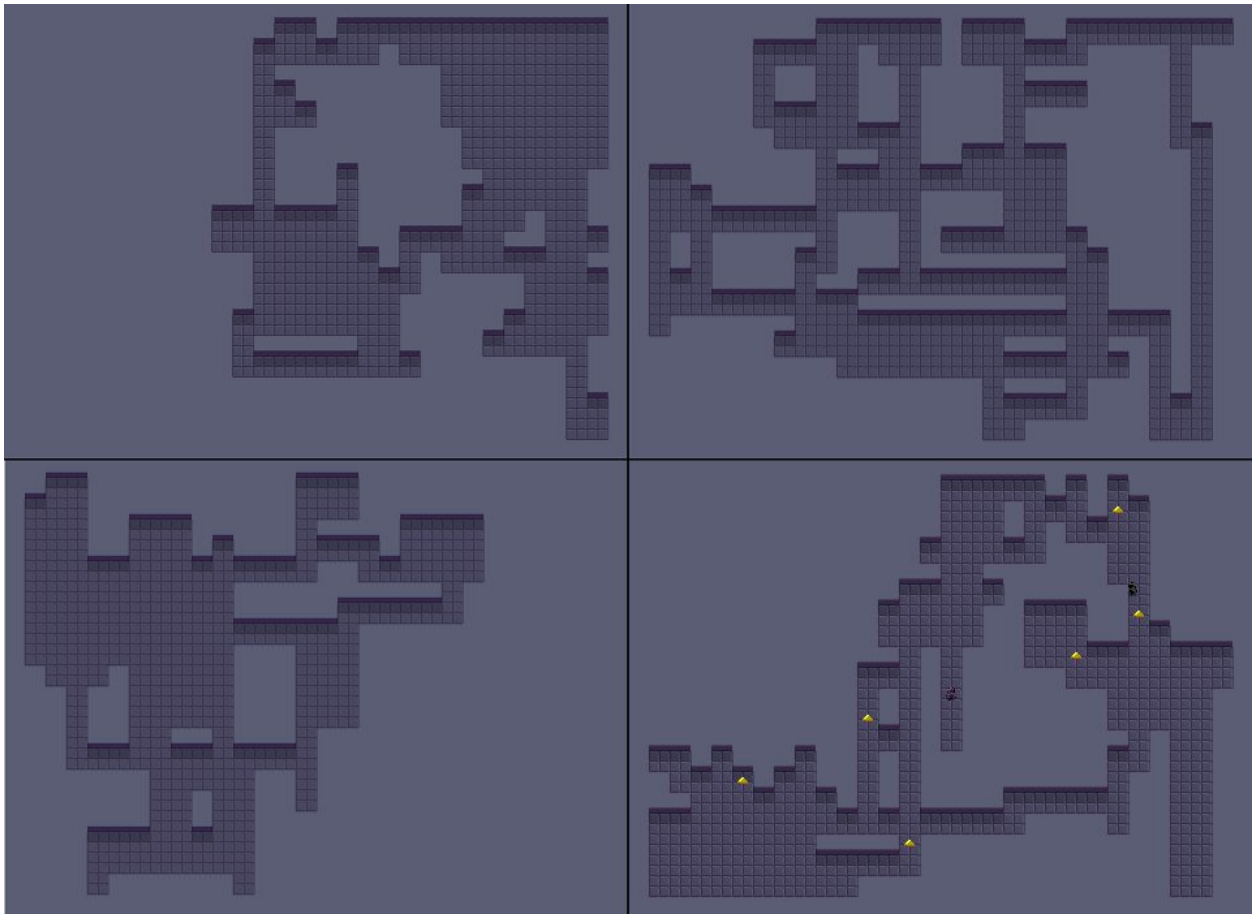
The character in Rem, however, is no longer tied to just walking (left & right), instead they can walk in all directions, for now the in-game camera adopts the

Top-down style: permitting the player to see from the top of the game character. Thus giving the player more freedom to roam.

This state of trial and error is implemented to match the state of people of who get trapped in their dreams/coma.

After succeeding in solving the challenges of Rem, the player will return to the Side-scrolling level and continue forth with the journey. Slowly uncovering the backstory behind the incident.

REM & Random Level Generation



Random Level Generation is an integral part of To Rem & Back, as the entire REM world is built and governed by it. With every entry to this level, as we can see, there's a new random overlay set by a dedicated algorithm. It makes sure there's no more than 400 steps (32x32 pixel tiles) by moving randomly through a specified grid that cover the entire level.

```
repeat(global._steps){
    grid_[# _controller_x, _controller_y] = FLOOR;

    // Randomize the direction
    if (irandom(_direction_change_odds) == _direction_change_odds) {
        _controller_direction = irandom(3);

        // GUI
        cont_changes ++;
    }

    // Move the controller
    var _x_direction = lengthdir_x(1, _controller_direction * 90);
    var _y_direction = lengthdir_y(1, _controller_direction * 90);
    _controller_x   += _x_direction;
    _controller_y   += _y_direction;

    // Make sure we don't go outside the grid
    if(_controller_x < 2 || _controller_x >= width_ - 2){
        _controller_x += - _x_direction * 2;
    }
    if(_controller_y < 2 || _controller_y >= height_ - 2){
        _controller_y += - _y_direction * 2;
    }
    steps_count ++;
    if(steps_count == 400) {
        cont_last_pos[0] = _controller_x;
        cont_last_pos[1] = _controller_y;
    }
}
```

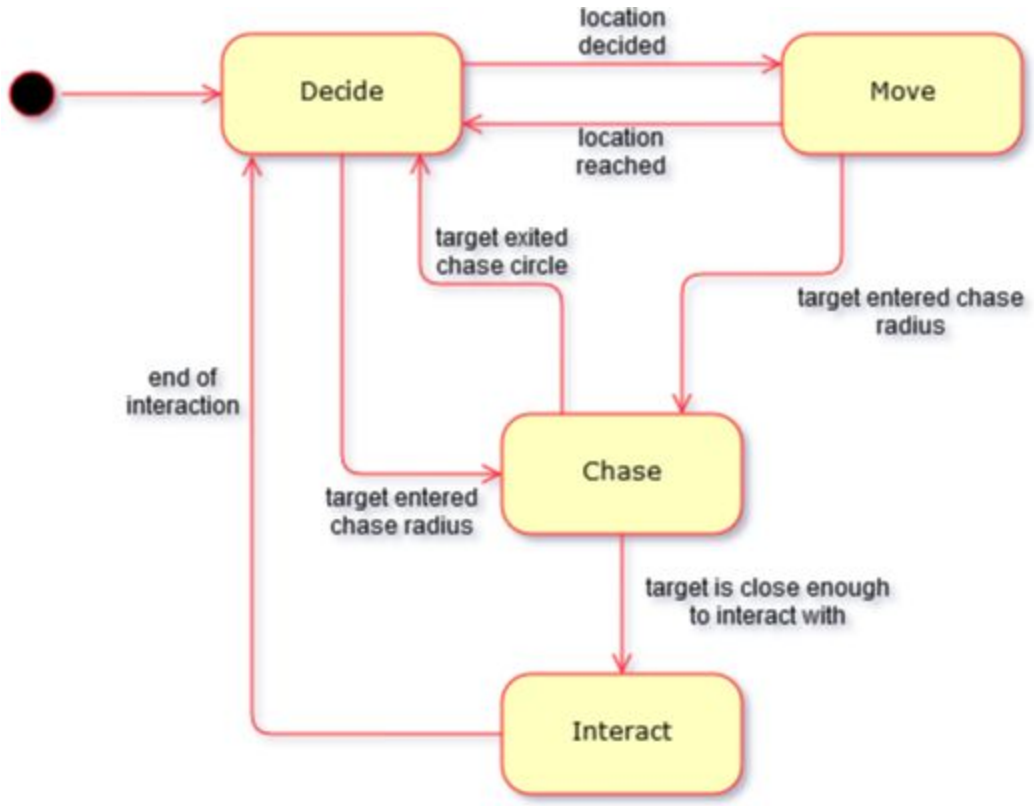
The algorithm then goes over the entire grid cells again to place objects in prefered, and sometimes, random locations

The previous snapshot shows the distribution of the objects:

- The player is placed in the center.
- The collectibles are scattered randomly.
- The enemy is always placed 80 pixels away from the player.

Artificial Intelligence

Non Playable Characters (NPCs) in To Rem & Back are managed by the Finite States Machine. Below we see a UML diagram of an NPC.



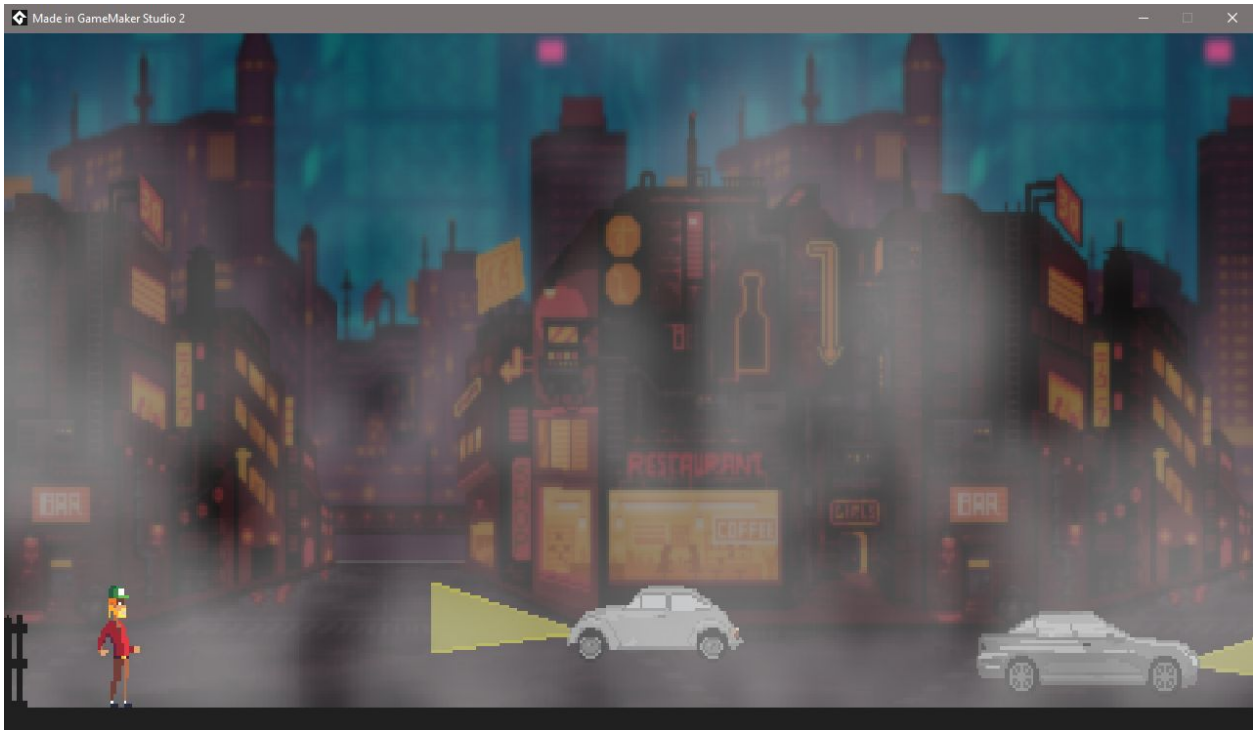
By default, this NPC enters the Decide state.

- Decide: The NPC generates random, but reachable, coordinates. It is also always on the lookout. i.e: If the target object (mostly the player) gets close, the state will change to Chase.

- Move: Executes the necessary code to move the NPC to that location. This code also include setting the right animations direction, and sounds. The NPC is also on the lookout for the target object at this state as well.
- Chase: Target is within a close range. NPC will not use the random coordinates, but instead will rely on the target's coordinates and follows them. The NPC will keep chasing the player until it is either within a closer range (the interaction range), and thus switch to the Interact state. Or out of the chase range, then the state will return to Decide.
- Interact: The NPC is in a very close range, it will execute it's main functionality (chat or attack) and then terminate and turning the state back to Decide.

Mood Settings

To Rem & Back can be described as a Story-driven game, and thus we focused some of our efforts on setting the foundation for a gameplay governed by the mood of the in-game player.



General aim is to showcase To Rem & Back as an interactive theatrical play.

Adding elements like fog, darkness, ambience, trivial sound effects are of high priority in order to keep the game more engaging, vital, and immersive.

Future Considerations

Unlike what we think, 2D games still exist in 2019 and are performing very well in the market.

Even indie developers (Independent Developers) with modest production capabilities are able to tell very valuable stories and create engaging mechanics using capabilities that engines like Game Maker Studio are offering.

To Rem & Back has a story to tell, and it's one of the very few games ever made to feature more than one player perspective (camera perspective) and more than one genre in the same game, blending both the fun in unraveling the story through different levels and the technical pleasure of creating a random level generator and an AI system for enemies.

Therefore, we are considering making To REM & Back not just a demo, but a complete finished product to be released in the market as soon as the development and testing phases are over.

Conclusion

Our plan was to create a game that features two different genres and camera perspectives in one coherent story. On the technical side, we wanted to create a game that isn't plainly simple. The random level generation coding and the AI search and follow system were main contributors to this.

We were able to succeed in writing the code and script, designing the assets, and finalizing the first portion of the game.

Given the enough time and effort, we might be able to produce a finished game that is able to compete in the 2D gaming market.