Punk Noir

by

Ethan Michael McNabb

An Honors Capstone

submitted in partial fulfillment of the requirements

for the Honors Diploma

to

The Honors College

of

The University of Alabama in Huntsville

5/1/20

Honors Capstone Director: Professor Vinny Argentina

Assistant Professor of Art

5/1/2020

Student (signature) Date

Director (signature) Date

Department Chair (signature) Date

Honors College Dean (signature) Date
Honors Thesis Copyright Permission

This form must be signed by the student and submitted as a bound part of the thesis.

In presenting this thesis in partial fulfillment of the requirements for Honors Diploma or Certificate from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department, Director of the Program, or the Dean of the Honors College. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

____________________________
Student Name (printed)

____________________________
Student Signature

5/1/2020

Date
# Table of Contents

Abstract 2

Introduction 3

The Project 4

   The Concept 4

   Setup and Team Structure 5

   Story and Initial Work 6

   The Pause Menu 9

   The Dialogue System 11

   Tutorial Level 17

   Evidence and Interactable Objects 18

Conclusion 22

Link to Game 23
Abstract

My honors capstone project is an enhanced version of the game that my team and I developed in ARS-415 – Team Game Design. I selected this project because of my interest in game development and because it seemed like a natural way to integrate my capstone into my curriculum without it being the last semester. The resulting project is Punk-Noir, a cyberpunk science fiction murder mystery game where the player must gather evidence from the environments and talk to several characters in order to find out who killed the CEO of a large robotics company, if anyone. I worked with four other students on the project in Fall 2019, then continued work in Spring 2020 to improve and polish the game further myself. Facing down the challenge of learning a new game engine and its own unique way of programming, my primary area of work throughout this project was implementing a fully-functional dialogue system that had various conditional options to select from based on what evidence you had collected, what character you were speaking to, and what subject you were discussing. This took a lot of time due to the involvement of putting massive amounts of data into tables, then playing the game to make sure the information was being read properly. Other significant contributions included interactable objects, a functional pause menu, and a tutorial level. What I’ve achieved in this game has inspired me to move forward with game development as a persistent hobby, and I hope to be able to pursue this passion farther in my free time.
Introduction

I always had the idea of creating a narrative science fiction game. I’ve spent countless hours playing games in the genre and developed quite a passion for them as a result. When the opportunity arose to make a game of my choice for ARS 415 in Fall 2019 and for my honors capstone, I leaped at the opportunity to merge my passion for that genre into my skills as an up and coming game developer. Although we were unable to finish the game to our complete satisfaction in ARS 415, I was able to perform additional work in the spring to polish the game up to a standard I’m proud of. This paper details my work on the game, along with the vision for the game and the teammates that helped me achieve it.

The Project

The Concept

Punk Noir is a cyberpunk-styled story-driven murder mystery game played in first-person perspective, so the player sees through the eyes of the main character. A “cyberpunk” theme essentially means the game is set in a near-future, science fiction setting revolving around technology, large corporations, and a primary color-focused nighttime aesthetic. Many of the common themes of the genre include transhumanism (the uploading of the human consciousness to a computer), artificial intelligence, and cybernetic enhancement. For our game, the aspects of the genre we wanted to focus on were the corporations and the artificial intelligence, and we wanted to merge with a classical noir-type detective story and characters. The objective of the game is to figure out who killed the CEO of a robotics company (or if he just committed suicide)
by gathering evidence from the game’s levels and talking to various characters to piece together what happened. These characters in the game are: police officers, an accountant, the company’s executive officer, the lead programmer, and (briefly) a robot. Core to this game is an in-depth dialogue system inspired by games like *Deus Ex: Human Revolution*, *Detroit: Become Human*, and *Mass Effect*, where you ask characters different questions based on what evidence you’ve collected so far, encouraging you to explore the area as much as you can. I collaborated with my classmate Klifton Riley to come up with the initial pitch for this game (an excerpt from which can be seen in Figure 1), and our original vision included unrealistically lofty goals such as fully functional combat and stealth encounters. We also had a mechanic where non-player characters (NPCs) could get stressed out depending on how you questioned them, but we had to cut it both due to time and the difficulty that would be involved with implementing this. The length and level count of our game also had to be reduced, and various interactable items had to be cut in time for our final evaluation. I completed the implementation of most of these smaller items after the course concluded. Despite the ambitiousness of the pitch, it proved successful enough to be
one of the four pitches selected by the class to go forward with development in the course.

**Figure 1**: An excerpt from the original pitch presentation for Punk Noir.

**Setup and Team Structure**

The overall game was a team effort, although I am the only one submitting it as an honors capstone. I was grouped with four other students, Brandi LeBaron, Emma Kratt, Klayton Riley, and my original partner Klifton Riley. Primarily, Brandi and I were the team’s programmers, whereas Emma, Klayton, and Klifton were the artists for our team.

Brandi LeBaron primarily implemented the game’s journal system, helped me with the in-game conditions affecting the dialogue, an item-highlighting system for evidence (which had to be dropped unfortunately, but still remains in the game functional for some items), and created the logic for in-game music and level transitions. She and I also teamed up to create many interactable item widgets for evidence or points of interest.
Emma Kratt collaborated with me on the story content initially, as we figured out the characters, profiled them, and wrote the initial story outline. As I got more and more absorbed into programming, however, Emma took over almost all the dialogue writing for the NPCs. I wrote the player character’s dialogue based off this. In addition to the story, Emma also did a lot of work on the art direction, such as set-dressing the levels and creating materials. She also created some custom idle animations for our characters.

Klayton Riley primarily worked on art assets, most notably the robot character in our game. Along with Klifton, he worked on the absolutely massive number of props for both of our levels.

As mentioned earlier, Klifton Riley brainstormed the initial pitch for the game with me and became the Art Director once we started production with our new team. He created the models for the police officers and worked on the massive number of props with Klayton. Klifton, Klayton, and Emma all worked on lighting the levels.

As for myself, I implemented the pause menu, dialogue system, the dialogue data tables, a tutorial level, and worked on interactable widgets for both the evidence and points of interest along with Brandi. After the course concluded, I worked by myself to overhaul the flow of the in-game dialogue and I added in several interactable objects in the second level that had to be cut, including an evidence piece that never made it into the original game. Of course, this paper is going to more thoroughly explore what I implemented specifically.

**Story and Initial Work**

The first thing I did for the game outside of basic conceptual work was start developing the plot and characters with Emma Kratt. I developed character sheets for two police officers,
who are the two main characters for the game’s first non-introductory level (those can be seen in Figure 2). This was done for every character, as the sheets gave us information on their disposition, role in the story, and personal details that we could use to craft interesting dialogue. However, other than the two cops, I didn’t fully create the rest of the sheets, I simply either outlined and/or reviewed and revised Emma’s where necessary.

My other significant contribution story-wise was helping to brainstorm the initial plot and the overall direction we wanted to take. This included mapping out the relationships of the characters to the evidence and the characters to each other, as well as basic outlines for how the plot was going to progress in each level. A snippet of this can be seen in Figure 3. This work’s impact was unfortunately not felt in the demo we made, because both main levels involve evidence gathering and the plot was not advanced enough to really get to the twists and turns that we planned. In the demo, you cannot really make enough progress to determine who the murderer is, but we did manage to develop initial intrigue. Some personal details on NPCs can be discovered, and we managed to implement a few subplots, such as the accountant’s drinking habit and the discovery that the murder victim had been sleeping with the programmer’s wife.
<table>
<thead>
<tr>
<th><strong>Police Officer #1: Sergeant Blake</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position</strong></td>
</tr>
<tr>
<td><strong>Physical</strong></td>
</tr>
<tr>
<td><strong>Personality Traits</strong></td>
</tr>
<tr>
<td><strong>Possible Motive</strong></td>
</tr>
<tr>
<td><strong>Arrival Time</strong></td>
</tr>
<tr>
<td><strong>Known Visited Locations</strong></td>
</tr>
<tr>
<td><strong>Revealed Visited Locations</strong></td>
</tr>
<tr>
<td><strong>Related Evidence</strong></td>
</tr>
<tr>
<td><strong>Relationship to CEO</strong></td>
</tr>
<tr>
<td><strong>Work Environment / History</strong></td>
</tr>
<tr>
<td><strong>Personal Life</strong></td>
</tr>
<tr>
<td><strong>Additional</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Police Officer #2: Officer Wilson</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position</strong></td>
</tr>
<tr>
<td><strong>Physical</strong></td>
</tr>
<tr>
<td><strong>Personality Traits</strong></td>
</tr>
<tr>
<td><strong>Possible Motive</strong></td>
</tr>
<tr>
<td><strong>Arrival Time</strong></td>
</tr>
<tr>
<td><strong>Known Visited Location</strong></td>
</tr>
<tr>
<td><strong>Revealed Visited Locations</strong></td>
</tr>
<tr>
<td><strong>Related Evidence</strong></td>
</tr>
<tr>
<td><strong>Relationship to CEO</strong></td>
</tr>
<tr>
<td><strong>Work History</strong></td>
</tr>
<tr>
<td><strong>Personal Life</strong></td>
</tr>
<tr>
<td><strong>Additional</strong></td>
</tr>
</tbody>
</table>

**Figure 2**: Character sheets for the police officers
Ask about Email and/or Robot Design Blueprints

Programmer
- Those are the designs for our next line of maintenance bots, the 3100 series.
- The improvements mainly increase motor function and increase self-sufficiency.
- CEO has marked them as a company priority for the fiscal quarter.
- CEO asked about them a lot, constantly inquiring about the progress of the project.
- Much as I hate to say it, Manny’s run its course.
- Manny’s one of our best designs, but it’s getting outdated.
- I was working on finalizing some elements of this design over the past week.

Bot
- Aware that these designs exist, but isn’t aware of the specifics.
- Contributes to bot’s worry about being deactivated, increases feelings of worthlessness (Why am I being replaced? What did I do wrong?)
- Noticed how much of the programmer’s time has been taken up by this, says it knows he will do a good job.
- Is surprised (or at least acts like it) to learn how much attention the CEO is personally giving this project.

Ask About Broken Security Cameras

Sergeant Blake
- “That’s where the suicide theory gets really stretched”
- Is extremely suspicious about this piece of evidence, suspects that this is where foul play becomes involved (right idea, but cameras were destroyed by CEO in a drunken fit)
- (After Detective talks to COO) Wait, he smashed his own cameras? Well, there’s that theory out the window.
- Is somewhat disappointed that this doesn’t support the murder theory, as he’s very suspicious that this isn’t a suicide.
- Something is still off about this Detective, I know the cameras were my only point, but this doesn’t feel right.

Officer Wilson
- Blake’s convinced that this means there was a murder. I’ll grant it’s strange, but that’s literally the only odd thing we found.
- I mean, we know this guy was all kinds of messed up. Cameras being broken doesn’t instantly mean the guy was murdered.
- (After Detective talks to COO) I knew it. Maybe Blake’ll back off a bit on insisting this guy didn’t just off himself.
- With all the other stuff we got, there really isn’t any reason now to believe that it wasn’t a suicide. The cameras were the only anomaly.

COO
- Oh no. CEO smashed those months ago. He got in a fit about “being watched” one night. The next morning, I found them all broken. There’s no point in replacing them,
The Pause Menu

After my initial story work, the bulk of what I did fell into the category of programming. The biggest challenge I faced on the project was getting to know the Unreal Engine. Despite knowing of it through reputation, I had never worked with it before, and I had never worked with its innate scripting language, Blueprints. Blueprints is a visual programming language (or tool, depending on how you look at it) centered on creating logical flowcharts that take the place of line-by-line coding. Blueprints are used to make game programming more accessible to non-programmers, but as somebody who had only worked with more traditional programming languages, the shift was fairly abrupt.

As such, my first programming task was relatively simple. I created a basic pause menu, which gave me an introduction to the style of Blueprints as well as widget creation and user interface (UI) design in Unreal. The programming aspect mostly involved creating a function in the FirstPersonCharacter blueprint (the blueprint that handles the player character) to handle the pressing of the P button on the keyboard. Screenshots of what these blueprints look like and the final result can be seen in Figures 4, 5, and 6.

Figure 4: The blueprint function for pausing the game.
Figure 5: A screenshot of where the pause game function is called in the First-Person Character blueprint.

Figure 6: The pause menu in-game.
The Dialogue System

What most of my time was spent on was the dialogue system for the game. The main goal for this system was for players to be able to navigate branching dialogue paths so they could ask NPCs about what they knew about the crime or about evidence the player found. The key goal was for the player to be unable to ask about certain evidence pieces until they had found them. This specific goal was the most programmatically challenging thing in the project, as the system we grabbed as a foundation from Unreal’s free marketplace assets used a script interpreter that wouldn’t be able to pass variables that we were using from the FirstPersonCharacter blueprint into the data tables where dialogue was stored. Essentially, the problem was unless you forced the variable in, the variable inside the data tables (and the dialogue scripting system as a result) would be different from the evidence variable being set inside the game instance’s blueprints. Because of this, dialogue options dependent on certain evidence pieces could either display when they should not have or not display when they should, regardless of whether the character had picked up the evidence piece or not. This issue was eventually fixed by setting the internal variables to be name-type variables so we could pass them in properly, and a snippet of setting variables up in this manner can be seen in Figure 7. With that fixed, the dialogue system could then grab and set variables for the various pieces of evidence without any issue. This finally allowed us to use whether an evidence item had been found or not to influence the available dialogue options, and we were able to use Boolean functions to control this. The functionality for handling an evidence item being found by the player can be seen in Figure 8. These functions were duplicated across all the evidence items.
In order for the dialogue to work, in-game objects needed to be made for the dialogue handlers to be attached to, since attaching the dialogue handlers directly to the characters resulted in the player having to look at the character’s feet in order to trigger dialogue. To avoid this, I created various cylinders and placed them in the chests of the characters with their in-game visibility set to false, so the player would not see obvious white shapers sticking out the NPCs. What this looks like in the editor can be seen in Figure 9. This had to be done for every character.
Figure 7: A snippet of how evidence is set up in the FirstPersonCharacter so dialogue can track them. Note the calls of InitEvidence for every evidence item.

Figure 8: One of the functions for setting whether an evidence piece had been found or not, in the main player blueprint, FirstPersonCharacter. A similar function was made for every evidence piece.

Figure 9: The dialogue actor for an NPC shown in the editor, with fields for configuring the dialog actor displayed on the right. This is not visible in-game for obvious reasons.
The final part of the dialogue system to mention is the data tables. For each level, I had to create a data sheet containing the text and attributes of every line intended to be said for every character in any situation. This took a massive amount of time. The reason why was that all of Emma’s dialogue had to be imported in and branched properly, with supplementary dialogue that I came up with (such as go back options) added in to make conversations flow more smoothly. For almost every dialogue option Emma wrote, the player says something that prompts the response, as well as displays a shortened version of the sentiment as an option for the player to select (this was called ResponseText in the table). I wrote all this player dialogue and response text basically on the spot, trying to maintain a smooth conversation with Emma’s dialogue the whole way through. Then, I had to test all of it for proper branching and typos. Another type of table detailing the speaker attributes also needed to be created, listing off every character and how they were referenced in the tables. This was done so the dialogue system would know who was speaking which lines and could display the right names and dialogue options in the dialogue UI. A screenshot of this speaker attribute table can be seen in Figure 10, and a snippet of a data table for a level can be seen in Figure 11.

<table>
<thead>
<tr>
<th>NPCNorm</th>
<th>SpeakerName</th>
<th>SpeakerPortrait</th>
<th>bIsPlayer</th>
<th>bSilentResponses</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlayerMe</td>
<td>Me</td>
<td>None</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>Officer1</td>
<td>Officer Blake</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>Officer2</td>
<td>Officer Wilson</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>PoliceChief</td>
<td>Police Chief</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>COO</td>
<td>Cassandra Cresmont</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>Programmer</td>
<td>Harold Bradenburg</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>Accountant</td>
<td>Michelle Waters</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>TutorialCop</td>
<td>Officer James</td>
<td>None</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

**Figure 10:** The Speaker attributes table
I also made changes to the dialogue UI. To better fit the theme of the game, I dug into the system we had in place to replace the font of the text with something that the team agreed fit the theme more. This took a decent amount of time because I had to find out where I needed to go and what I needed to change to get the effect I wanted, along with searching for and testing different fonts. The final dialogue system look as of the end of the Fall semester can be seen in Figure 12.

However, after the course concluded, I decided to go back into the data tables to rework the flow of conversations. At the end of the semester, outside of evidence questions, every question you could ask a character was displayed on screen for you to ask. Depending on where you were in the conversation, this could lead to upwards of 5 options being displayed at once if you had unique dialogue display along with the always present ones. This often lead to a crowded UI that could disorient the player. I reorganized the dialogue along a “submenu” system where you could delve into topic questions after selecting that topic, so originally you would just

**Figure 11:** A snippet of the Accounting floor’s dialogue, specifically evidence related dialogue.

Note the conditions on the right side of the table. This controls the visibility of those options.
have a few options (the topics) to choose from. This drastically improved the player experience and made dialogue far easier to follow. You can see the reworked changes based on Figure 12’s image in Figure 13, where I interact with the same character but with the improved system. I went into a submenu in the new system in Figure 14.

Figure 12: The dialogue system as of December 2019.
Figure 13: The improved dialogue I implemented after ARS-415 concluded.
Figure 14: A submenu in the new dialogue system. Note that options from Figure 12 are presented more effectively here.

**Tutorial Level**

After feedback received from our classmates in the beta phase of this game’s development, a need for a tutorial became evident. So, with some brief dialogue from Emma, I constructed a small level to teach the player the basics of the game. This took the form of a small room with one NPC who gave the player instructions and a table with three items for the player to pick up. Talking with the NPC gives you instructions on what to do, and he’ll tell you how to pick up objects and open your journal. His dialogue will change depending on what stage of the tutorial you are in. This was implanted in time for the end of the course, and it received significant positive feedback. A screenshot of the tutorial can be seen in Figure 15.

Figure 15: The tutorial level.
Evidence and Interactable Objects

The final major aspect of the game was the evidence objects themselves, along with various interactable items in the environment that the player could comment on in order to make the levels feel more alive. The team’s artists would create the props for the evidence pieces, and afterwards either Brandi or I would create an actor to handle the player being able to interact with the object. An actor is a construct that you attach to an object and/or model in order to meet certain programming goals. This usually involves some type of event handling with other objects and/or characters, such as “What script executes when a player touches this object? How should it react?”. For our player-evidence interactions, we set them up so they would be handled by a collision detection box attached to the actor, so when the player steps into the object’s collision box a “Press E to interact” prompt widget would display. If the player pressed E, a new widget would display with the player character’s thoughts on the object. If the object was an evidence piece, it would also set the internal variable regarding whether that object was found or not to true, enabling the player to choose dialogue options relevant to that item in conversations as discussed earlier. For various interactable objects that were not evidence however, only the widget displaying the player character’s thoughts would be detailed. The process of creating the object actors was not too terribly challenging for a single object and adding a lot of these to the levels turned out to be pretty fun. The exceptions were the evidence pieces, as they would have to be integrated into the FirstPersonCharacter blueprint via a Boolean function and name variable to be tracked properly, along with any relevant dialogue needing to be added to data tables. However, for non-evidence interactable objects, it was pleasantly simple to add back some of the
ones we had to cut. I was surprised by the simplicity enough to add a couple of new ones entirely of my own making. Examples of this system can be seen in Figures 16, 17, 18, 19 and 20.

**Figure 16:** The editor viewport for the flash drive item. Note that the cube serves as the collision box. This cube’s scale, position, and size often had to be adjusted on other items to account for the surrounding environment (i.e. making sure you couldn’t interact with an item because its collision box went through a wall).
**Figure 17:** A snippet of the blueprint for the Flash Drive evidence item. You can see the functionality for interacting with the player entering its collision box above the final row. The extension of the final row that goes off the screen is the functionality of telling the game and dialogue system that the evidence has been found.
**Figure 18:** The interact prompt on one of the evidence pieces.

**Figure 19:** An example of a reactionary widget displayed after E is pressed for an evidence piece.
Figure 20: An example of a reactionary widget for an interactable object that is not an evidence piece.

**Conclusion**

This game was quite the endeavor for me and taught me a lot about the Unreal Engine platform and what it does, and does not do well. Of course, Unreal is a platform with limitless possibilities and what I’ve shown in this paper could have been done in dozens of different ways. Working on this game in ARS 415 taught me how to work collaboratively with a multidisciplinary team on games, as working alongside artists was not a position I found myself in often in the CS department. Because of this I was able to get an idea of what genuine game development was actually like. Although I figured it would take a massive time commitment, and I was hardly wrong about that, I noticed just how easily the plans you had for the game can get derailed. This taught me how adaptable you need to be when making a game, and how often
you need to make compromises both for the sake of your team and the game itself. But above all, working on this project allowed me to pursue my passion for making games and helped me better understand the scope and time it takes to develop a game from start to finish. I’m probably not going to be able to make games as a full-time career anytime soon, but working on this project has persuaded me to keep pursuing it as a hobby. I plan to continue learning Unreal and making games in my spare time for the foreseeable future.

**Link to Game**

The following is a drive link to .zip file of the game’s playable build should you want to play it (requires Windows 10):

https://drive.google.com/open?id=17gXb8myyN8_2i5NT18GVHNUm6y1StXO
Hi Dr Wilkerson,

I approve of Ethan McNabb's Honors Capstone Project, *Punk Noir*.

Please let me know if you need anything else from me!

Thanks,
vinny