Dramaturgies of the Unreal

This Nightswimming Pure Research project was held at Tarragon Theatre's Extraspace from January 22 - 24, 2024.

Research conducted by Cole Lewis, Sam Ferguson, Patrick Blenkarn with Brian Quirt and Gloria Mok from Nightswimming.

Thank you to the Tarragon Theatre for its partnership and collaboration on Pure Research.

For more information on Pure Research, please visit nightswimming.ca/research.

Published November 2024.

SUMMARY – Languages shape the way we see the world—but what about computer languages? What about the platforms and interfaces and software environments within which we create and communicate our projects? This Pure Research project explored the dramaturgical foundations of Unreal Engine and C++ programming. The goal was to develop a repertoire of questions and provocations for theatre and performing artists interested in incorporating these materials into their future projects.

Provocations developed from this project include the following. For more detailed Conclusions and Questions arising from this research, please see section 8.0 below.

1. What if the technology you rely on for performance is instead used to transform how you rehearse and develop ideas?
2. How does the act of troubleshooting technical glitches reveal new creative pathways, rather than hinder your process?
3. In a world of constant updates, can you embrace impermanence and change as essential to your artistic practice?
4. How do biases embedded in the software you use impact not just the stories you tell, but the stories you choose to explore?
5. Can you imagine a rehearsal process where digital experimentation is as integral as actor improvisation—how might that shift your vision?

Dramaturgies of the Unreal – Patrick Blenkarn, Cole Lewis

1.0 Introduction

1.1 Original Proposal

Our proposal to Nightswimming's Pure Research was as follows:

> Languages shape the way we see the world—but what about computer languages? What about the platforms and interfaces and software environments within which we create and communicate our projects? Guilty by Association's (GbA) research project explores the dramaturgical foundations of Unreal Engine and C++ programming. Our goal, in addition to training ourselves in Unreal and C++, is to develop a repertoire of questions and provocations for artists interested in incorporating these materials into their future projects.

1.2 Context

This research process took place concurrently with our working on the first and second phase of our show, *2021*, as part of the Tarragon Theatre Greenhouse Residency followed by the Artist at Home Program with Theater Mitu. *2021* is a collaborative storytelling event between performers, audience members, and AI. It contains a live narrated video game designed for a participatory performance that is accompanied by live music. Ultimately, *2021* is an exploration of the significance of human data and the challenges of preserving dignity in death for those we fundamentally disagree with. Our Pure Research project was a space to dive deeper into the questions we were grappling with in the show but in an isolated context.

The challenge of learning Unreal Engine and C++ was quickly noted during our work on *2021*. Both skills can take years to learn in a dedicated fashion. We quickly narrowed our premise to see what would be possible for a variety of skill sets: a complete novice (Cole Lewis) alongside a C# coder and Unity game designer (Patrick Blenkarn). To address our limitations, we invited Sam Ferguson, a sound designer and C++ coder to join our research. Sam was able to help translate Patrick's understanding of C# coding and, at times, the block-based coding of Unreal Engine. Cole focused her efforts on learning how to navigate Unreal Engine with an even more narrow interest in building an avatar—via MetaHuman—in Unreal Engine. Together, we combined our understanding to create experiments and gain an understanding of what questions and provocations artists who are new to these technologies should be considering in the future.

1.3 Goals

Throughout our Pure Research Period, we focused on the following goals:

- A basic to advanced understanding of the syntax of coding (depending on our skillset)

- The ability to build something with Physics (where an object can impact another object) in Unreal Engine
- Advanced understanding of how to build a MetaHuman in Unreal Engine
- Experimenting with how to add C++ code to the block-based code of Unreal Engine so we can expand its potential
- Adding motion capture suits to the MetaHuman in Unreal Engine so we can make the avatar move in real time

Additionally, with Nightswimming's generous support, we held our three studio days at Tarragon Theatre. Over three days, we (Cole Lewis, Sam Ferguson, and Patrick Blenkarn) held conversations with a small group of experts and workshops with two theatre artists to test some of the aesthetic and political potentialities of working with Unreal Engine, object-oriented programming languages, AI-character creation, and MetaHuman avatar creation in a performance creation process.

This document outlines our activities.

2.0 Artists

Cole Lewis (she/her) is a mom, theatre artist, and Co-Artistic Director of Guilty by Association. She specializes in creating live performance from design ideas, exploring new modes of storytelling, and fusing technologies to the stage. Her practice includes directing, playwriting, and the design of moving image works. Twice-nominated for Dora Awards, Cole's work uses humour, design, and technology, to explore notions of violence, expose questions of bias, and unsettle standard conceptions of 'truth'. Select credits: Writing/Directing/Performing the Dora-nominated moving image performance of *1991* for RISER Project. Adapting Kyo Maclear's *Virginia Wolf* for Geordie Theatre. Co-writing/originating the Direction of the Dora-nominated *Keith Richards: The One Woman Show*. Directing Canada's largest wearable art show, STRUTT. Devising/Directing Redshift Music Society's immersive *Still Life Continuum* at Vancouver's Orpheum Theatre. Ongoing: Development of *Untitled Nurse Play*, seed commission by Stratford Festival, development of *2021,* creation of *The Piñata Project*. Disparate, divergent, and wide-ranging, Cole's works question received ideas about identity, violence, and systems of oppression to explore alternative futures.

Patrick Blenkarn (he/him) is an artist working at the intersection of performance, game design, and visual art. He is co-artistic director of Guilty by Association. His research-based practice revolves around the themes of language, labour, and economy, with projects ranging in form from video games and card games to stage plays and books. In addition to his work with Guilty by Association, he is, along with Milton Lim, the co-creator of the video game for the stage, asses.masses, and the arts economy trading card game, culturecapital, and the co-founder of the national video archive of performance documentation, videocan. He has a degree in philosophy, theatre, and film from the University of King's College and an MFA from Simon Fraser University. patrickblenkarn.com

Sam Ferguson (he/him) is a longtime collaborator with Guilty by Association and resides in Toronto. He is an award-winning sound designer/composer who has worked with numerous Canadian directors from Nikki Shafeeulah to Jessica Carmichael. Previous credits include *1991* (RISER Projects), *Boys, Girls, and other Mythological Creatures* (Carousel Players), *Virginia Wolf* (Geordie Theatre), *Mom Jail* (winner of the HFX 72 hour film challenge), *Division* (Tarragon Theatre). Sam is also a C++ coder who has built new audio plug-ins for improving sound effect creation processes. His understanding of coding and large language processing models influenced how artificial intelligence was implemented into *2021*.

3.0 Background

This research builds on a number of recent experiments that our team members have conducted into bringing video game technology into the theatre and coding.

Since 2019, Patrick has been building and touring *asses.masses*, a video game designed to be played from beginning to end in a theatre. The 7+ hour work currently tours in seven languages to festivals around the world.

Concurrent to that project's explorations, GbA conducted a 2-week residency into VR and interactive storytelling possibilities at the Center for Collaborative Arts and Media in New Haven in August 2022.

Over the pandemic, Sam created and programmed his audio plugin, DelayTroll, using C++.

These explorations provide the backdrop for GbA's current artistic work, *2021* (2023-present), a work that applies much of the questions and thinking developed during our Pure Research residency. We also recognize that these experiences meant that we were approaching these questions with a team possessing a certain fluency and familiarity in both the technologies and the computer languages. Not everyone would feel as comfortable throwing themselves into the context.

4.0 Research Team Discussions

We often met to discuss what we were learning either individually, collectively through a workshop or interview, about related questions that arose from our work on *2021*, and/or to analyze our findings after an experiment.

Throughout Pure Research we discussed:

- The challenge of learning coding (dedicated time which is a resource not all can afford).
- How some game engines (like Unreal Engine or Scratch) are using block-based coding.
    - Block-based coding consists of text-based computer commands that are grouped together in pre-programmed blocks. You can drag and drop these blocks together to build computer programs such as animations and games.

- The advantages of block-based coding:
    - It is advantageous for beginners because it can be easy to learn and implement.
    - No background in coding is required.
    - It is visually appealing.
    - It can teach logical and critical thinking skills.
    - It can promote creativity by allowing beginners to build characters or worlds.
    - It can provide a basis for text-based coded language learning.
- The limitations of block-based coding.
    - The visual nature of block-based coding simplifies the complexity of programming intricacies.
    - The drag-and-drop interface may not be sufficient to solve more complex problems.
    - Syntax forms the backbone of coding and block-based coding obscures syntax making it harder to debug any problems that arise.
    - It is challenging to debug block-coding, limiting its application in professional industries. For example, we could not build a non-player-character (NPC) in Unreal Engine that utilized a large language model for dialogue with block coding. We needed C++ coding to navigate the limitations of block coding to achieve our tasks.
- Bias in coding and block-based coding
    - Algorithmic bias is the systemic and repeatable errors in a computer system that create unfair outcomes, such as privileging one group of users over others.
    - Just because a tool is tested for bias against one group, doesn't mean a bias against another group might not show up.
    - The assumptions of engineers (or the ones engineering the coding) can be biased and this bias can show up in the application being built.
    - Block-based coding may also contain bias and this bias can be harder to subvert or change without knowledge of code.
    - In Unreal Engine, the block-based coding might also be reflected in what they think the application of Unreal Engine is and who will be using the software. For example, Unreal Engine was built for video game designers and not theatre makers. Video Game Designers may want to complete different actions with the engine than Theatre Makers. Theatre Markers may need someone who codes to debug the bias towards Video Game Designers in the block-based code of Unreal Engine.
- C++ and C# coding can be a useful tool for Text Analysis.
    - Syntax in computer programming refers to the rules that control the structure of the symbols, punctuation, and words of a programming language.
    - Without syntax, the meaning or semantics of coding language is nearly impossible to understand.
    - The syntax of C++ and C# can be utilized to read a script, analyze a script, and better understand our own biases when reading a script.

5.0 Focus Group Discussion

We had decided to have conversations with:

- Coders
- Game Designers
- Unreal Engine Educators
- Theatre artists

In our first focus group, we discussed:

- Accessibility of technology
    - Whether a user has no previous experience with coding or is an expert.
    - Controller accessibility, controllers that allow for easy maneuvering
    - Colour in block-based coding. Colour is often used to distinguish between blocks but some folks who are colourblind are not able to differentiate between blocks.
    - User-friendly fonts for different needs
    - The camera angles and character point of view in a game can lead to more or less motion sickness.
- The relationship between game engines and related technologies
    - Motion Capture to make video game characters move
    - Artificial Intelligence to create characters in the game engine that can improvise dialogue
    - Allowing players to build their own Avatars
- What questions theatre makers should be asking of themselves when working with technology?
    - How to better integrate the theatre audience into the story through video game design
    - Can artificial intelligence be considered a collaborator?
    - What skills are needed to prepare theatre makers to work with game engines and related technologies?
    - What different needs and costs should be addressed in grants and funding when theatre companies work with game engines and related technologies?
- Associated costs with using technology in theatre

In our second focus group, with a group of two theatre artists, we undertook a text analysis experiment using CONVAI and an additional experiment in digital character creation with MetaHumans. Our discussions with this focus group are recorded in more detail later in this report.

(Thank you to Gloria Mok for being a great scribe.)

6.0 Team Experiments

6.1 Unreal Engine and Physics

6.1.1 Overview

We participated in a workshop with Unreal Engine to learn about Rube Goldberg Machines and how they can impact the building of real world environments with cause and effect. This requires both a basic understanding of how to use Unreal Engine and a basic understanding of physics.

The workshop is geared towards industry professionals and educators familiar with Unreal Engine, however, many found it challenging.

The workshop led us through a series of 6 lessons before culminating in a final exercise where we built our own Rube Goldberg machine.

The workshop began by opening a learning kit that contained a premade animated kitchen environment. We loaded the pieces needed through the levels palette and made the lesson plan visible and ensured the streaming method was loaded. This took some time to understand.

The next stage of the workshop taught us to use the Content folder and for this lesson we focused on Geometry Scripts. We clicked on an incline plane (which is a wedge of cheese) and dragged the wedge of cheese to the kitchen counter.

Throughout the six lessons prior to the final project, we learnt a variety of skills such as, how to:

- Navigate the fundamentals of Unreal Engine 5 interface within the Viewport and the Content Drawer or Browser.
- Move an Actor from one location to another within a Level.
- Duplicate an Actor in a Level.
- Manipulate the Scale and Rotation of an Actor within a Level.
- Utilize Unreal Engine’s Modeling Mode to create and edit a Static Mesh.
- Enable physics on Static Meshes.
- Generate and apply a Collision Mesh to a Static Mesh.
- Apply the Chaos Destruction system to an object.
- Create and apply a custom material.
- Generate Blueprints to add MetaSounds to a level.
- Activate Metasounds within a Blueprint.
- Manipulate Splines using transformation tools and custom widgets.

For the final product—building a Rube Goldberg Machine in Unreal Engine—we are meant to demonstrate skills such as:

- Design and develop our own Rube Goldberg device in Unreal Engine through an iterative process.

- Apply our skills to navigate the fundamentals of Unreal Engine 5 interface within the Viewport and the Content Drawer/Browser.
- Demonstrate the effective use of Unreal Engine's Transformation Tools.
- Apply our knowledge to create custom Models, implement Particle Emitters, add Metasounds, and manipulate sounds.

In relation to the functionality of Unreal Engine and utilizing tools for world building:

Objects like a wedge of cheese often float above a surface. To make the wedge of cheese sit on the surface of the kitchen counter, we needed to select the object and press the F key to focus on the object. Next we pressed the ALT key to orbit around the object. The left mouse click allowed us to move 360 degrees around the object. Moving around the object, revealed a gap of space between the wedge of cheese and the kitchen counter. To make the wedge of cheese snap down to the kitchen counter, we needed to make sure the object (a wedge of cheese in this case) is selected, after which, we pressed the END key (FN and Right Arrow key on MAC). This function allowed the object to snap directly down to the surface.

In relation to the functionality and physics of Unreal Engine and utilizing tools for world building:

One of the workshop's interface activities was called "Wedge a Melon". In this activity, a watermelon falls through a funnel onto a wedge of cheese. The task is to sharpen the wedge of cheese so that it cuts the watermelon into halves.

Such a simple sounding tasks requires learning how to:

- Understand what a widget is
- Move objects (which requires using the move tool properly and utilizing the widget)
- Resize objects (which requires using the size widget properly)
- Pay attention to the sharpness value of the cheese wedge when moving and resizing
- Understand physics to complete the task. For example: What size is the wedge of cheese? How sharp does the wedge of cheese need to be? How far does the watermelon need to be from the wedge of cheese? Does velocity play a role in the cutting of cheese?

We learn that if the size value is higher than .8, the watermelon will be sliced in half after it falls through the funnel when the funnel is hung from the ceiling without adjusting its initial scale.

6.1.2 Outcomes

As we worked on this "Wedge a Melon" task and 6 other tasks related prior to building our Rube Goldberg device, we began to understand the challenge of creating cause and effect relationships in Unreal Engine.

If theatre makers want to build a world in Unreal Engine for a production, even something as simple as ensuring a chair is properly positioned on the floor (snapped to the surface) is a lesson that can take a beginner 30-60 minutes to comprehend and implement.

If theatre makers want to build a world in which objects can act upon other objects or actors can act upon other actors, a basic understanding of physics and Unreal Engine functions is required. Beginners should expect to spend at least 2-7 days orientating themselves with Unreal Engine, undertaking lessons to understand the basics, and experimenting with the physics of the world.

Our participant, Cole, in this workshop was a newcomer to Unreal Engine with no previous coding experience and limited experience gaming. She struggled to comprehend and complete each of the six tasks, sometimes outright failing. (She could not make the wedge of cheese slice the watermelon in half). The culminating project was simplified from a Rube Goldberg machine to a game of dominoes on a kitchen counter. She worked to successfully scale and position objects snapped to the surface. All dominoes fell down when triggered making for a success. With time and dedication, even an absolute beginner can navigate some deceptively simple builds in Unreal Engine.

People interested in undertaking these same activities can do so.

Read about the lesson plan and download the package here:
https://www.unrealengine.com/en-US/lesson-plans/unreal-learning-kit-rube-goldberg-machines

Utilize these videos for helpful assistance on how to complete each task:
https://www.youtube.com/watch?v=le4hdYoP7p8&list=PLQgWpMEA6fs90YfabLUJA3x-GUsv0SqA-&index=2

6.2 Unreal Engine and MetaHuman

6.2.1 Overview

We were given access to Unreal Engine's MetaHuman Creator. It is a complete set of tools and plugins that gives anyone the power to create, animate, and use highly realistic digital human characters for cinema, cutscenes, or video games in an Unreal Engine project.

MetaHuman comes ready with premade avatars that can be utilized. You can alter their appearance in many ways. For example, you can change the gradation of the avatar's skin colour, adjust the pupil size in their eye, add make-up with or without intensity, play with the shape of their physical body, and much more.

What excites most creators about MetaHuman is the ability to make a realistic copy of one's self. In short, you can make a digital copy of yourself; you can become an avatar. We spent some time learning how to do so. The steps are as follows:

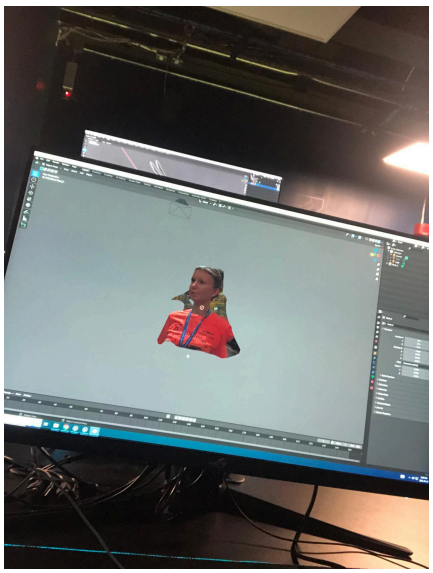
Step 1: 3D Scan your Face
First we created a 3D scan of Cole's face using the Polycam app on her Iphone 14. This allowed us to use her face shape as a mold for the MetaHuman. Polycam is an app that can be easily downloaded on a phone and, at the time, had some free options for our use. It can capture

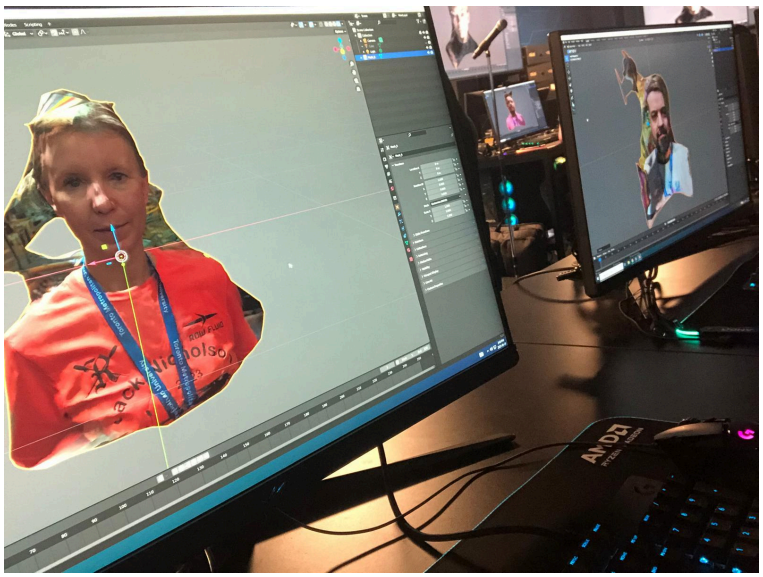
immersive 360 degree images and convert them into a precise LIDAR scan that results in a 3D object. We removed Cole's glasses—glasses and beards can interfere with a 3D face scan on this app— and took approximately 110 photos from different angles surrounding her head. The app then converted these images into a 3D scan of her face. On close inspection, you might see some odd textures and blur but, if minor, this can be adjusted in the next phase. If major, redo. The final step is to export the model in the GLTF format. GLTF formats were free with the app and worked for us, but in a more professional context we might want to pay to upgrade for better quality.

Step 2: Clean Up the Scan

To make the scan usable for Unreal Engine and the MetaHuman Creator, it needs to be cleaned up. To do this, we used Blender which is free. We import the GLTF file from Polycam and begin the process. First, we remove any parts of the model that are not needed. There is a weird blur of Cole's hair that protrudes from her head, likely an error in the scan. We simply remove this by selecting the vertices in the edit mode. Next we use Blender's built-in features to remove any additional loose vertices (like wisps of hair or pieces of the wall that accidentally became part of the scan) and apply "merge by distance". Our focus is on only retaining a 3D scan of Cole's face.

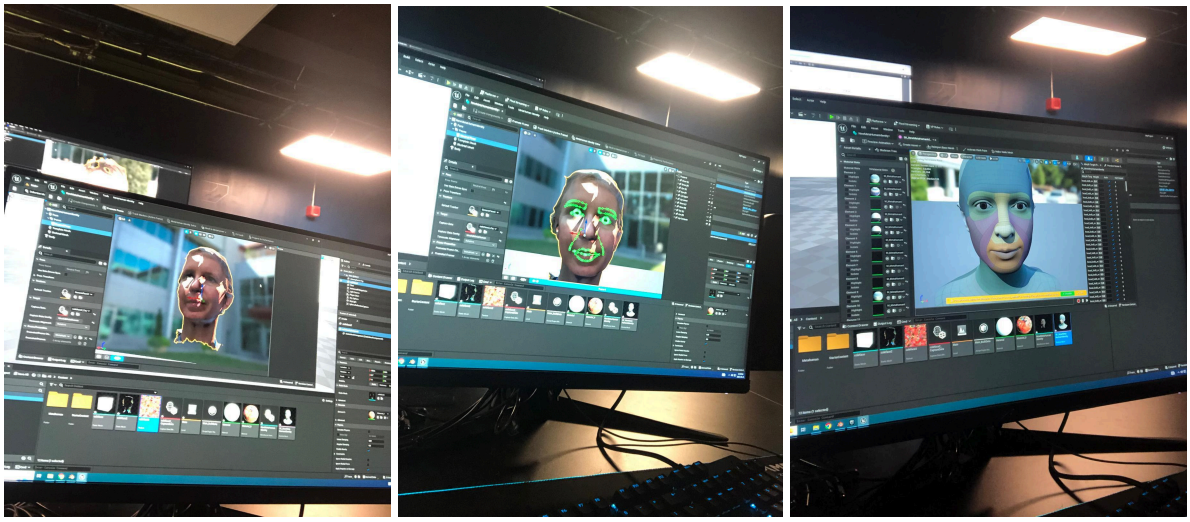
This is what it looks like as we tidy up a 3D scan of Cole's face in Blender:





To export the clean 3D scan of Cole's face, we use the following export settings:

- Export: FBX
- Object Type: Mesh
- Export only selected objects
- Path Mode: Copy
- Select: Embed texture



Step 3: Mesh to MetaHuman in Unreal Engine

To begin, you need to load up on Unreal Engine and select the most recent version that supports MetaHuman. We used 5.1.1, but you'd likely choose something newer now. When opened, we created a new project (type of projects only differ by initial project settings). We chose a Game 3D template, but it's also possible to choose Film/Video. Next, we enabled required plugins. We went to Project Settings, and under Plugins, we made sure the MetaHuman plugin was enabled. From here, we focused on importing the 3D scan. We opened the Content Browser, created a new folder and imported the FBX file we got from Blender. This may sound complicated but really involved dragging and dropping the FBX file into Unreal's Content Browser. From here, we used the ADD button and browsed for MetaHuman Identity. At this time we were asked to authenticate ourselves using our Epic account. Next, we used Components from Mesh to add the 3D scan.

To prepare to transplant the 3D scan face, we:

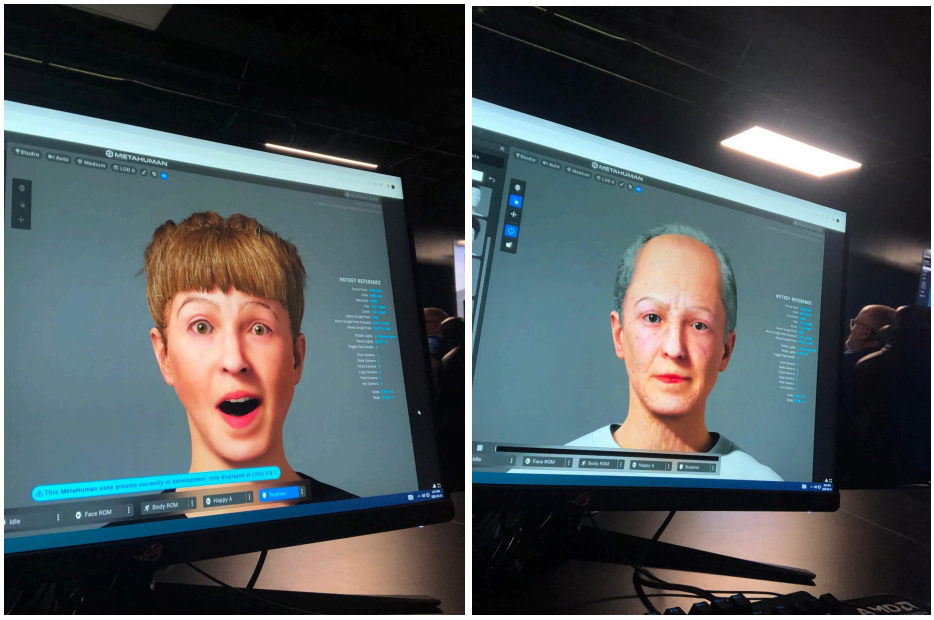
- Corrected the camera so that the face scan filled the viewport
- Changed the Field of View to 15 degrees
- Selected Unlit View Mode
- Clicked on Neutral Pose and Promote the Frame
- Clicked on Track Active Frame (and undertook some corrections if necessary)
- Clicked MetaHuman Identity Solve

By doing so, we took the 3D scan of Cole's face and used it as a mould. The MetaHuman character then took the shape of Cole's 3D scan. The final step of this phase is to send it to MetaHuman Creator by using the Mesh to MetaHuman button.

Step 4: MetaHuman Creator

Next we logged into the MetaHuman website (MetaHuman.unrealengine.com) to select our newly created character (Cole) from the My MetaHumans section. The MetaHuman tools allow us to select Cole's skin tone, adjust facial features, change her hair, and so on. In our case, we were interested in determining whether we could use a 3D scan of her face to make a MetaHuman of her father. We changed her gender, altered her hair, aged her skin, and used measurements from

photos of Cole's father to adjust some of the bone features and length of ears. We found it incredibly easy to use these tools and to alter the 3D scan of Cole. At this stage, we can either export our MetaHuman and import it back into Unreal Engine (which is what we did), or we can keep working on it using MetaHuman Creator or other external tools (which we also did).



6.2.2 Outcomes

In our experience, the MetaHuman Creator is much more user-friendly to a novice than Unreal Engine. Using Polycam, Blender, and MetaHuman Creator to build a realistic avatar of one's self and export it to Unreal Engine is a relatively achievable task and can be completed in an afternoon.

Designing the final avatar, whether changing its gender or selecting a new skin colour, raised some ethical questions for us that we knew we wanted to further explore through a Focus Group Experiment (found later in this report).

If theatre makers want to build a realistic avatar of an actor for a show, they will need to engage the actor for a photoshoot using Polycam. While achieving the scan should only take 15-20 minutes, the actor should be booked for an hour in case a scan fails and/or more time is needed.

Cleaning the scan in Blender and transferring it to MetaHuman Creator can take 2-3 hours for a beginner. When familiar with the tasks involved, it becomes relatively easy but will still take up to 1.5 hours depending on how difficult it is to clean up an image. Editing the scan using the MetaHuman Creator tools can take as short or long as a team desires. In our case, we ended up using the scan of Cole to become a character in our show *2021*. While our initial edit of Cole took about 90 minutes, we ended up spending about ten hours creating the final version of her avatar (AIBrian) for our workshop of *2021* at Theatre Mitu in August 2024. There is a lot of potential for

theatre artists to utilise the MetaHuman Creator with ease when building their own avatars for future productions.

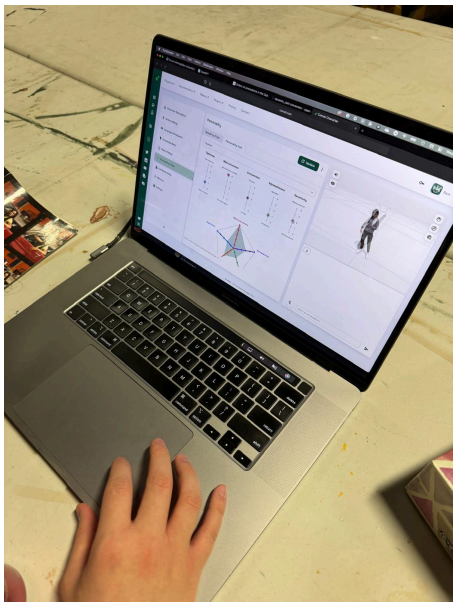
6.3 Unreal Engine, MetaHuman, and Artificial Intelligence (AI)

6.3.1 Overview

Video Games often contain Non-Playable Characters (NPC's). This is a character that is not controlled or played by the gamer. Often, NPC's have bad dialogue in response to an event or state imperatives to advise the player. There is an interest in using Artificial Intelligence (AI) to help enrich NPC's.

We became interested in whether we could connect the MetaHuman to AI so that it could have an improvised conversation. The MetaHuman Creator allows for Plugins. We researched many plugins and settled on CONVAI. CONVAI is an AI plugin that can enable a character to have human-like conversation capabilities in games and virtual world applications. It is a voice to voice interface designed for realtime end-to-end voice-based interactions with characters.

CONVAI allows us to create an objective and backstory for the character, a voice for the character, and expertise for the character. We can then test the character in a playground environment. It is possible to select which LLM we want CONVAI to use (we experimented with them all and are currently using Chat GPT 3 but might return to Llama). It is also possible to add unlimited knowledge to make your character an expert on their own memories or a particular subject.



These are the steps we took to connect CONVAI to our MetaHuman, AIBrian in Unreal Engine:

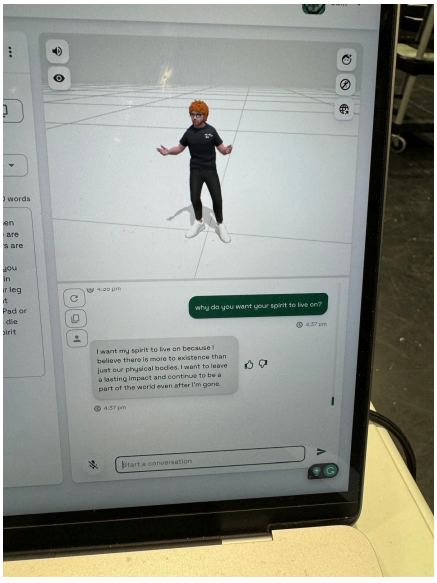
Step 1: Install the plugin from the Unreal Marketplace and activate the plugin in our project. After that we put our CONVAI API key in the plugin's menu. This connects our CONVAI account to Unreal Engine.

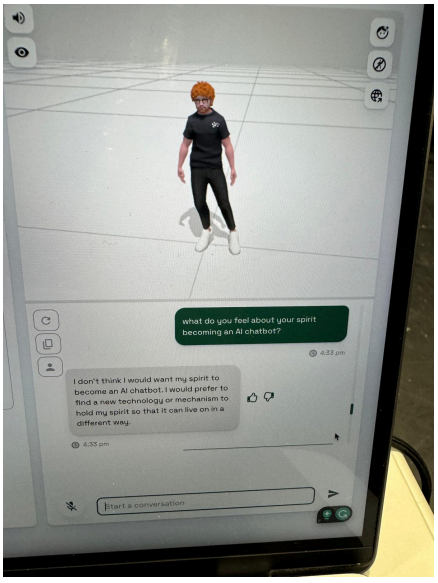
Step 2: Our MetaHuman has a "Blueprint" that defines everything about it. We need to connect CONVAI to our MetaHuman so it can control the animation of the MetaHuman for lip syncing and such. We should note the way you do this now is different from what we did at the time. I'll describe what we did to connect them but, much has changed since October 2023 with software updates and we encountered new challenges when we further experimented with it in June 2024. We opened the MetaHumans Blueprint and found the Details Panel. We look for the Class Options area and then change the parent class to ConvaiBaseCharacter. This gives our MetaHuman the properties of a CONVAI character that can speak and respond.

Step 3: The next step is similar, but this time it applies to the player. The player also has a "Blueprint". We need to connect to the player's Blueprint and change its Parent Class to ConvaiBasePlayer. This gives the player the properties required to speak to the CONVAI NPC.

Step 4: Finally, to get the lip sync to work, we downloaded a plugin from CONVAI's website (not the Unreal Engine marketplace). We then installed that plugin manually into the MetaHuman's Blueprint. We then found the Face Component, added a CONVAI Face Component and made sure the class controlling the face animations was Convai_MetaHuman_FaceAnim. CONVAI has updated this process and it is a bit simpler now.

(Note how different this approach is to controlling the animations of a MetaHuman with the steps we have to take with the Rokoko suit for motion capture as mentioned later in the report.)





Fleshing out the character of AIBrian through CONVAI required some experimentation and research into prompt engineering. CONVAI, once CONVAI is set up, is relatively easy to use. There is, however, a learning curve to engineering prompts that will allow for the characteristics

desired. For example, we found that CONVAI heavily relies on the Backstory. The character will fixate on an objective or fact provided in the Backstory and it can be difficult to get the character to wander or move onto other topics. We found that the Knowledge Bank does not really help solve this challenge, rather, the Backstory needs to be written with a basic understanding of prompt engineering. Additionally, and this is an integral step, improvised conversations using specific prompts continue to build with the character. It is possible for the CONVAI character to nurture new understandings by employing different types of prompts.

We experimented with different types of prompts:

- Zero-shot prompting: Directly instructing the model to perform a task without any additional examples to steer it. These were our initial prompts.
- Few-shot prompting: Provide a few examples and/or mini-lesson before instructing the model to perform a task. As we continued to experiment with CONVAI, we began to also employ this prompting method.
- Chain-of-Thought prompting: A method that guides the Large Language Model's reasoning process by breaking down complex tasks into manageable steps, allowing the model to process information in a logical sequence. We are currently experimenting with this prompting method as we continue to build our character in a more controlled and complex manner.

The experiments around AI are outside the initial scope of our research goals for Pure Research. We share some of the knowledge around prompting only because it does relate to our understanding of how to use these tools, the biases that arise, and offers some further questions plus provocations to consider if theatre makers were to implement these tools into a production.

6.3.2 Outcomes

CONVAI as an app is user-friendly for a novice. Expertise is required for those who want to utilise the plugin for Unreal Engine, particularly if there is a desire to create a node in Unreal Engine's block-based coding and connect the plugin to another plugin.

In prompt design, we recommend:

- Maintaining a consistent format for all examples. This helps the model recognize the pattern it needs to follow.
- Provide enough context in the prompt to make the tasks clear. This may involve a brief instruction or description to set up the task.
- Keep the prompt as simple as possible while still providing the necessary information. Avoid complex prompts.
- There is an ability to use a Filter which will ensure the AI character does not say any hateful or harmful content. There are, however, implications to using this Filter. It can be challenging to get an AI character to make certain statements that might be necessary for a character in the context of a story. Use of the Filter requires careful ethical consideration.
- We wanted to have fun with the character we were building and initially, we would write "joke" prompts only to find humor in the awkward reply. We learnt these actions need to be avoided when building a character as it doesn't allow for consistency nor does it allow the

model to recognize the pattern it needs to follow for the resulting character. Although, it certainly is fun.

Even though CONVAI can, within minutes, provide you with a character that is ready for conversation, honing in on the character to ensure consistency and complexity is a time-consuming task. We have easily spent over 50 hours building/prompting (and rebuilding/re-prompting) our MetaHuman CONVAI character, AIBrian.

We were initially curious about whether we could make a MetaHuman character improvise by connecting it to AI. We were successful in doing so and believe other theatre makers can easily do so as well although it might require consultation with a C++ programmer (for plugin limitations) and Prompt Engineer (for consistency and complexity of character). There are challenges to creating such a character.

In short it is possible to connect CONVAI to MetaHuman so that a 3D realistic avatar can have improvised conversations centered around an objective. There are many potential implications for this in theatre. For example, it is possible for an actor to reappear in a digital world and for the avatar to carry on an improvised conversation utilizing the actor's objective and backstory for their character.

6.4 Unreal Engine, MetaHuman, and Motion Capture

6.4.1 Overview

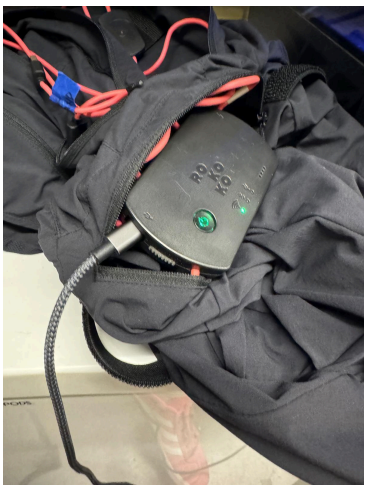
It is now possible to do motion capture in a home environment using a Smartsuit Pro by Rokoko. Many Twitch Streamers do so to bring their avatars to life. We wanted to learn how to connect a MetaHuman in Unreal Engine to a Smartsuit Pro to see if it's feasible for theatre companies to utilize this technology in the future.

We were able to conduct this experiment with the additional support of the Design + Technology Lab at Toronto Metropolitan University.

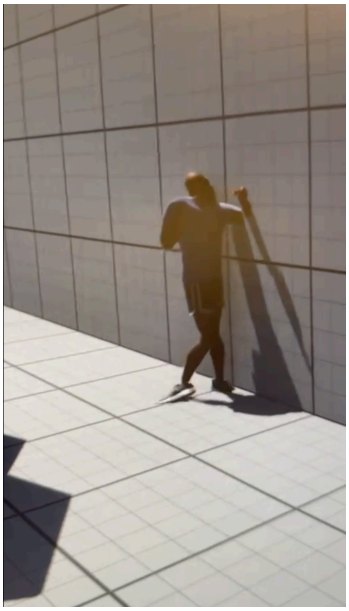
6.4.2 Steps taken to control the Rokoko Suit

Step 1: Set up the suit and hardware:
The suit works by communicating the data from a number of sensors (accelerometers and such) that are placed uniformly on the suit. This is done over wifi and requires the computer to be connected to a dedicated WIFI router. Do not use a router already in use, as that router, which is also transferring regular internet data, can confuse the software. Once we set up the router we powered the suits sensors by plugging in a USB power bank to the three USB plugs (two for the hand sensors and one for the suit). The suit has a little pouch to keep the Power Banks. This pouch is also where you can find the USB plugs. Once that was completed, the suits showed up in the software but, unfortunately, not all of the sensors were reporting properly. The sensors are connected together by proprietary cables in a "Christmas light fashion" that, like Christmas lights, frequently break. To fix this, we looked at the software to see which sensors were reporting and

which sensors were not reporting. We went down the chain replacing each wire until we found the offending wire, repeating this process until we could see all sensors reporting in the software.



Step 2: Set up Rokoko studio (the suits software):
The software that reads the sensor data and translates it into 3D movement in space is Rokoko Studio. Once the Rokoko suit is set up, an actor needs to put on the suit and hit the calibrate button with their arms still at their side. Once it is calibrated, whoever is wearing the suit should see their movements reflected by the 3D avatar in the Rokoko Studio software. Note, at this time, the wearer will observe their movement in a basic provided avatar and not a MetaHuman. To move a MetaHuman requires additional steps. To connect it to a MetaHuman, we selected that basic avatar and then went over to the Live Link panel in the Rokoko Studio software. We activated the Unreal Engine Live Link option and then noted the IP and Port Number associated with it.



Step 3 Connect it to the MetaHuman:
Connecting to the MetaHuman is a challenging task and a little too technical to go over in detail. We have decided to focus on the concepts needed to make it work and then provide a link we found useful for a step-by-step tutorial. Basically, 3D animations work by connecting 3D “meshes” to “bones”. The mesh is the 3D object that can be observed without any texture or colour. The animation is a general shape of the 3D object and the bones are what tell the meshes where to be. The Rokoko suit is defining where the bones are, which in turn, defines the meshes. Problems arise when the bones of a MetaHuman and the bones of a 3D avatar in Rokoko Studio software don’t match. To solve this problem, Rokoko provides a new skeleton “bones” for a MetaHuman which they call a Skeletal Mesh Asset. This asset is still different from the Rokoko Bones but ensures compatibility. Rokoko Studio also provides a “Bone Map” to translate the movements of Rokoko bones to the movement of MetaHuman bones.

For a more detailed instructions, follow the instructions on Rokoko’s support page:
https://support.rokoko.com/hc/en-us/articles/16514557425681-Livestream-to-Custom-MetaHumans-in-Unreal-Engine-5

Once we successfully translate the MetaHumans Bones we have to set up the animation logic for those bones to essentially “listen” to Rokoko Studio. This listening will help tell the MetaHuman bones where the bones should be and how they should move. Once the Rokoko Suit is connected to a MetaHuman, the final step involves telling Unreal Engine where it should be listening to receive these bone animations. This step is completed by entering the IP address and Port Number.

6.4.3 Outcomes

Connecting Rokoko's Smartsuits to a MetaHuman avatar for real time motion is possible and can be used by theatres for future work.

Undertaking the steps to make this work is time-consuming and not always easy. We invited in a twitch streamer to observe the process and worked with staff at the Design + Technology Lab. The twitch streamer revealed that they had tried on three separate attempts for a total of 12 hours to connect the Smartsuit to a MetaHuman. All their attempts were unsuccessful. The staff member had never tried before but was familiar with some who were successful. Having Sam, a C++ coder, and Staff familiar with the technology present, proved helpful in debugging any challenges that arose. Ultimately, the tutorial linked above was needed to guide us on final steps. It took us about 3 hours to successfully connect the Smartsuit to our MetaHuman. Theatre makers should invite in consultants and give themselves a full day to experiment with the technology prior to applying it.

Additionally, there can be drift with the Smartsuit. Drift can look like a leg or arm drifting away from the body or dangling. Metal present in a room can make it so a suit doesn't work and/or cause more drift. Rokoko has created a coil pro that enhances the capabilities of motion capture. The coil pro is likely required for theatres that have grids or steel beams present. Unfortunately, the CoilPro only works with both the Smartsuit Pro and Smart Gloves, increasing the cost to use this technology.

The use of motion capture (and the Smartsuit Pro) is not new to theatre. Rokoko (and similar companies) makes the technology more accessible to theatres. We applied Rokoko to a MetaHuman avatar. In our experiments we made our MetaHuman jump, dance, and move across space. Future experiments will continue to focus on simple gestural dances between a human actor in the suit and the avatar on screen.

6.5 Using code as a tool for text analysis

6.5.1 Overview

During early versions of *2021,* we had written a final monologue to present a series of questions to the audience. This monologue borrowed the logical structure of if-else statements that one commonly finds in coding languages.

It looks like this:

```
Trying to build a mechanism to hold my Dad's spirit, as he hoped for, led me to many
kinds of questions, including:

IF my Dad wanted his spirit to be saved
```

```
  Is his Data worth saving at all?

  Who decides whose thought patterns are of value and whose are not?

  What Logic is being used to make these kinds of decisions?

IF he wanted to be saved AND his Data is deemed not of value:

  Is there a minimum quantity of data to be permitted for every human being to be saved?

  Or should all that is not valued by the Logic be deleted?

ELSE

IF none of this is the case, and my narrative is wrong, and my Dad did not want his
spirit to hold on, and that was just a delirious request,

  Is there a mandatory type of data that someone cannot delete?
```

As we worked on that monologue, we decided (for fun at first) to see if we could rewrite the argument as code. It looked like this:

```
list<Mind_Parts> Mechanism_Holder(RawMindData * _DadsMind,list<Deciders> _Deciders)
{
	map<string, Mind_Parts> Parts = Parse_MindData(*_DadsMind);

	bool hasSaveConsent = Parts[“Save_Consent”];

	int currentDecider = Determine_Decider_index();
	int currentLogic = _Deciders[currentDecider]->Determine_Logic_index();
	int minimumToSave = _Deciders[currentDecider]->getLogicMinimum(currentLogic);

	list<Mind_Parts> partsToHold;
	list<Mind_Parts>::iterator it = partsToHold.begin();

	for (auto Part : Parts)
	{
		bool isValuable = _Deciders[currentDecider]->UseLogic(currentLogic, Part);
		bool isMandatory = _Deciders[currentDecider]->DetermineIfMandatory(currentLogic, Part);

		if(isValuable && hasSaveConsent)
		{
			partsToHold.insert(it,Part);
		}
		else if(isMandatory)
		{
			partsToHold.insert(it,Part);
		}
```

```
	}

	if(partsToHold.size() == 0 && minimumToSave != 0)
	{
		partsToHold.insert(it, Parts.begin(),
		Parts.begin() + minimumToSave);
	}

	delete _DadsMind;

	return partsToHold;
}
```

We then decided to provide an edit of the above code to be included in the Greenhouse Residency showing of *2021.* (Note, the ending was revised for future iterations of *2021* and this is now cut.). It looked like this:

```
The code for my questions is:

if(auto dad = dynamic_cast<Programmable>(Dads_Mind))

{

	if(auto dad2 = dynamic_cast<Valuable>(dad))

	{

		writeToFile(dad2);

	}//valuable

	Else

	{

		dad = deleteNonValuableData(dad);

		WriteToFile(dad);

		}//insufficient value

	}//programable

Else

{
```

```
        delete (Dads_Mind);

}
```

As we entered the Pure Research process, we wanted to see where this approach could take us. On Day 1 of our studio time, we took a short play, “The Palace at 4AM” by Suzan-Lori Parks and tried to translate the text into coding language, specifically object-oriented languages like C++ and C#, which both Sam and Patrick were able to ‘speak’ respectively.

Even for a play of 1.5 pages, this is a massive undertaking. But it was clear that the translation process was very helpful for distilling the given circumstances in the play.

If you, the reader, do not know how object-oriented programming works, this next section describes its core concepts.

6.5.2 Object-Oriented Programming

Code is essentially a list of instructions for a computer. Older code (like C) used to work just like that: do this, then do this, etc. The problem with this approach was that it got very repetitive and led to a lot of rewriting the same thing over and over again.

Imagine you were giving someone who knew very little about the world instructions how to pick apples. You might describe what an apple is and how the process of picking works. Then later you needed that person to pick oranges… In languages like C, you would not only need to describe to this person what an orange is, but you would also need to redefine the process of picking again.

‘Object orientation’ came about as a way to code more conceptually. It allowed programmers to define objects typically in blocks of code called a ‘class’ and then just refer to those classes later to avoid all this repetition. Not only that, but classes can ‘inherit’ properties from a ‘parent’ class. So now your fruit picker can be given detailed instructions on what a fruit is, after that you only have to describe what is unique about apples and oranges. Then you can describe the process of picking and your instructions to them can be as simple as, “Hey fruit picker, go pick apples and oranges.”

6.5.3 Object Oriented Text Analysis

You can imagine what it’s like to apply this kind of thinking to narratives and its component parts. A character for example could be considered an object with properties that have commonalities with other characters and unique parts. This lens of analysis offers a unique look into the structure of how the characters relate in the system of the play.

We conducted an internal exercise as a trio to test this method by applying it to a short play by Suzan-Lori Parks.

6.5.4 The Palace at 4 A.M by Suzan-Lori Parks

“The Palace at 4AM”, by Suzi-Lori Parks in *365 Days,* is a 1.5 page play featuring an exchange between a King and a Queen at 4 A.M.

We focused on questions like:

- What are the kinds of functions that only a King can do?
- What are those that both Kings and Queens share?
- And what about Servants?

In doing so, we found alternative language to get to the ‘facts’ of a scene. We analyzed these questions as if we were writing the classes that defined these characters/objects' relationships. For example, all of these characters would inherit the properties of a ‘parent’ Human character and the functions that a Human could do (move, talk etc.). Our group decided to name ‘Royalty’ a class, and it came to denote its own set of functions and characteristics that both a king and queen would inherit, but a servant would not.

Kings and Queens would also have their own separate classes within Royalty. These would denote properties unique to them, and they might have similar functions but use totally different logics within those functions. Not only was this method helpful for looking at what was unique about the various characters in their social position and narrative function, but it also helped reveal where those characteristics were inherited from in the logic of the play. This approach fostered a challenging and meticulous way to think about how the same data can be dealt with uniquely by different characters and can produce different outcomes. It also emphasizes the logic behind those different handlings that can come from the structure of their social relationships.

For example, during the play a stage direction mentions that the King “leads” the Queen away. We talked at length about how the fact that the king can “lead” the queen reveals who gets to “control” the more common function of movement. The King, one learns, can also station a servant (Balthazar) at the door, which is in a way also controlling the movement of Balthazar and confining the movement of the Queen.

Interestingly the character Balthazar was “pardoned” last week, so in some sense his movement was released from control. There is a line in the play about the servants ruling when the King and Queen are gone, suggesting that perhaps the King’s control over the movement of the servants might be more complicated than one would expect.

6.5.5 Outcomes

Ultimately, by zeroing in on the common properties of all character objects, even those that don’t have any lines in the play, the function of movement and the logic of what kind of movement is allowed both became very clear, but also easy to articulate in a detailed way: The Prince moves from the palace, the Queen fails to control that movement. Balthazar’s movement is granted freedom and the Queen’s movement is led by the king. The power dynamics start to leap off the page. Of vital importance, analyzing the text in this way demonstrated the inherent bias of

participants. They had to question their own assumptions around power relationships and status. We hope to undertake further experiments in using the syntax of coding to explore text analysis and believe it will be of use to theatre makers in understanding the logic of relationships and actions for their characters.

7. Experiments with Invited Participants

On Day 3 of our studio time, we tested using CONVAI and MetaHuman as tools for dramaturgical analysis. We invited two Workshop Participants, both of whom were theatre artists from the Toronto community. For both experiments, we use the same play, “The Palace at 4AM” by Parks as a reference.

7.1 Using AI avatars as a tool for character analysis

7.1.1 Overview

As earlier written about, we experimented with using the plugin CONVAI to build a character utilizing data about Cole’s father, Brian, for our show *2021*.

Extending from our object oriented dramaturgy, we created an experiment for our Day 3 in the studio to see if CONVAI could be used as a tool for character analysis in theatre.

We set up CONVAI for our workshop participants (two theatre artists) and provided orientation to ensure it felt user-friendly. We demonstrated how to:

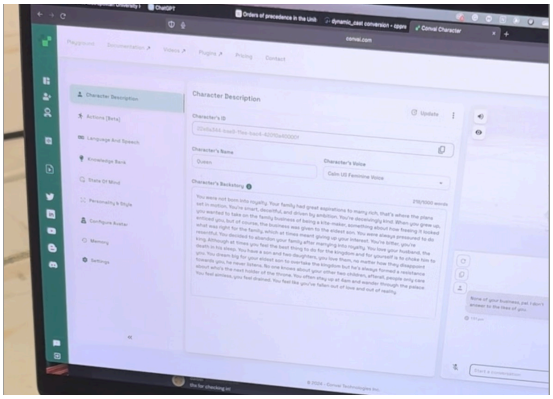
- Create a character with a name
- Select a character’s voice
- Where to input a Character’s Backstory
- How to add catchphrases
- How to give the character a state of mind
- How to save content

After reading and analyzing Parks’ “The Palace at 4AM”, we asked our Workshop Participants to try to create AI characters based on the King and Queen. We agreed to first demo the characters as a group to evaluate the results together. Ultimately, the Workshop Participants found CONVAI easy to maneuver and there was a lot of laughter as they input data, made selections about state of mind, and built their characters.

Participant 1: The Queen

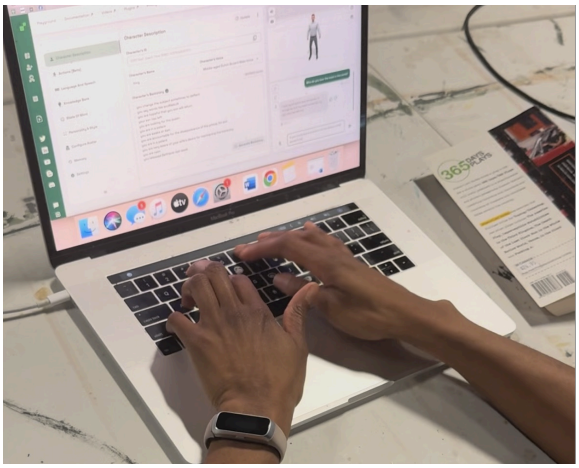
Participant 1 wrote a paragraph-length backstory for the Queen and played with the character’s state of mind. They saw permission in the word “backstory” to make “so much shit up” beyond the facts in the text of the play. For example, “Your family had great aspirations to marry rich, that’s where the plan’s set in motion. You’re smart, deceitful, and driven by ambition.” When the question: “Who do you love most in the world?” was asked of the CONVAI character. It replied with: “I love my Kingdom and the power it brings me although my heart aches for my son who has

left.” The reply was a close adaptation of some core facts the participant had written in the backstory.



Participant 2: The King

Participant 2 wrote their backstory for the King as a list of facts and played with the character’s state of mind. The facts were inspired from Parks' text and not invented. Participant 2 also selected a Dutch voice that elicited much laughter and sometimes confusion. CONVAI picked up words fed into the backstory like “scuttlebutt” and utilized them in its replies. For example, “Where is the son, my dear Queen? It seems as if he has vanished into thin air and I grow weary of the scuttlebutt.” In all cases, the replies were a close adaptation to core facts the participant had written in the backstory.



To further our experiment, we let the two AI characters for the Queen and King ask each other questions and improvise their own discussion. This led to statements such as the Queen expressing, “You think I waste my time asking questions to the King? I make the decisions around here, not him.” The King asked the Queen: “Where is the son, my dear Queen. It seems as if he has vanished into thin air and I grow weary of the scuttlebutt.” To which the Queen responded, “You think I’m your personal informant? Find your own damn son, I’ve got better things to do than babysit.”

At this time, Participant 1 expressed that they might have made the Queen too practical. We asked what they might change in their Backstory to adjust the character. They replied, “I think I would add, you do love the King. I think I made her practical about how she arrived to the palace, in this 4am state.” When asked about the King for Participant 2, they expressed that they wouldn't change much and that, “they don’t think there is love” between the King and Queen. Participant 2 also offered that they would add that the King is “assertive…to prompt the computer to be a little less floral.” Participant 1 responded by claiming they don't necessarily mean the Queen holds love for the King, but “love for status, for upholding the relationship.”

7.1.2 Outcomes

Utilizing CONVAI for text analysis between actors is generative and can be further developed to be used as a tool by educators or theatres in the future.

Our team set up CONVAI and ensured we had a plan to easily introduce the tools to the workshop participants. This took some prep time and is an intermediate task that takes about an afternoon to plan. Once the CONVAI workshop is ready, participants can be orientated to use the tools within 15-30 minutes. Actors should be given a minimum of 30 minutes to write a backstory for their character although longer times are possible. Once this task is completed, actors should be provided with time to test their backstory in the playground setting, along with time to make any desired edits. A minimum of 30 minutes is required. Then, if desired, it is possible to put these different CONVAI characters into conversation with one another.

Overall, with little guidance, the two participants were able to easily utilize CONVAI, build characters based upon their own hypotheticals, and test their text analysis through improvisation with the CONVAI characters they built. This resulted in discussion about the text that included the character's status, relationships, objectives, location, time of day, and state of mind.

The improvised conversations have the potential to reveal:

- How actors are defining their character objective
- The objective facts actors do or don’t know about their characters
- The subjective facts they are creating
- Whether subjective facts are obscuring any vital facts that are important to a conflict
- What character traits actors believe are important to portraying the character

CONVAI, in our experiments, proved a safe way for actors to enthusiastically reveal their initial understanding of a character to a team and put that understanding into discussion. Biases over character’s state of mind and trait were revealed prior to an actor embodying those choices allowing for an objective discussion.

At this stage of our research, we were more interested in the Backstory function and State of Mind function in CONVAI and paid less attention to the engineering of prompts, although there was some basic discussion about what prompts received more exciting answers. Further experiments in prompting could enrich this experiment and its results.

The use of CONVAI as a text analysis tool is a novel idea. In our experiments, we see potential for a playful way to objectively analyze objectives, given circumstances, relationships, and state of being prior to embodying those ideas on stage. Future experiments with this tool might prove useful in classroom settings or in table work for a rehearsal process.

7.2 Implicit biases in MetaHuman creation

7.2.1 Overview

As previously noted, we experimented with creating a 3D avatar based on 3D LIDAR scans of Cole and images of her father, Brian. MetaHuman Creator has ready made realistic avatars that allow users to explore available tools.

We created an experiment to see if MetaHuman Creator could be used as a tool for character analysis in theatre. We set up MetaHuman for our workshop participants and provided orientation to ensure it felt user-friendly.

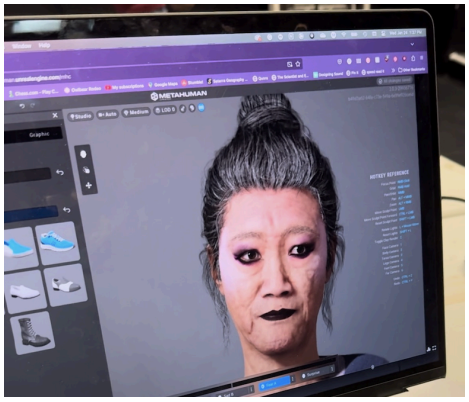
We demonstrated how to:

- Select a ready-made face from My Faces
- Explore tools under the heading Face (Blend, Skin, Eyes, Teeth, Makeup)
- Explore tools under the heading Hair (Head, Eyebrows, Eyelashes Mustache, Beard)
- Explore tools under the heading Body (Proportions, Tops, Bottoms, Shoes)
- Enable Editing
- Make the avatar share facial expressions and move

We asked our Workshop Participants to try to once again create AI characters based on King and Queen from Parks' *The Palace at 4AM*. The workshop participants found MetaHuman Creator very easy to use and wanted time to experiment with different functionalities. One participant took a long time blending different ready-made faces (one an older East Asian man, others were Black women) together and experimenting with the texture of their skin.

When we asked the participants to reflect on the choices they made in designing their characters, they shared:

Participant 1: The Queen
Participant 1 considered their understanding of the text and that the Queen raised their son and the Father was never there. They felt that "with the sound raising the staff and denouncing them, it aged me quickly" as their initial justification for building a character that looks like this:



Participant 2: The King

Participant 2 focused on a blend of the two actors present in the workshop (a young East Asian male and middle-aged Black artist identifying as she/they) and stated “I did not have a character based decision AT ALL” when designing an avatar of the King. They additionally shared that the centre face “appears to be of Asian descent whereas every other face around it appears to be Black women.” This resulted in a character that looks like this:

7.2.2 Outcomes

Utilizing MetaHuman Creator to build realistic avatars for characters is easy for the novice and is a tool we are likely to see applied more often in future games, cinema, and potentially theatre.

Our team set up MetaHuman Creator and ensured we had a plan to easily introduce the tools to the workshop participants. This took little prep time and is an intermediate task that at most takes about an afternoon to learn, plan, and implement. Once the MetaHuman workshop is ready, participants can be orientated to use the tools within 15-30 minutes. Actors should be given a minimum of 30 minutes to build an avatar for their character, although it is possible to give them up to 60 minutes if desired. If the avatars were to be used onstage in a professional production, a full day should be planned with additional efforts put into utilizing Polycam and Blender to create realistic avatars of the actors prior to editing them. In our workshop setting, the actors used ready

made MetaHumans and blended or transformed them to create their own characters inspired by Suzan-Lori Parks *The Palace at 4AM.*

Overall, with a little guidance, the two participants built MetaHuman avatars based upon their understanding of their characters with ease. We noted that at least one of the participants experimented with:

- Selecting a race for their character that is different from the participant's race.
- Selecting a gender for their character that is different from the participant's gender.
- Selecting an age for their character that is different from the participant's age.

Upon sharing the MetaHuman avatars, the team and focus group shared what kind of questions they want to see Theatre Artists considering when using MetaHuman Creator. The subsequent discussion included:

- It's refreshing to see diverse MetaHumans. Participant 1 expressed: "there's freedom to that."
- The ability to change race, gender, size, and age of the MetaHumans was discussed.
- Class, ability, and genetics can be more subtly played with. For example, the yellowing of someone's teeth.
- The limitations of choice in MetaHuman (limited hairstyles, limited body sizes, limited clothing choices, and so on).
- The ability to use one actor to create an entire cast of MetaHumans was discussed.
- A question was raised about the ethics of using one actor to create a diverse cast of MetaHumans.
- Participant 2 questions; "Who is creating the game?…Are we allowed to tell stories that aren't ours? And how do we decide what is our story to tell and what is not?" Where is the point of accountability?
- If a show is being created by a production team, with a large scale team dedicated to the creation of the virtual world and MetaHumans, where is the point of accountability? Is it individual or collective?
- How different structures of teams in theatre games and video games might relate to accountability. For example, some video games require multiple people to design branches and those multiple people may never have played other branches, making the Producer possibly accountable for the values and principles around who is creating and what stories are allowed.
- Considerations for future makers: It was offered by us that the same questions asked of a play and whether a director/actor has the right to direct/act in that play, should be asked of the technologies while acknowledging that not all intersections for someone might match up to the play, character, or technology. Participant 2 repeated, "Is this your story to tell?"
- Our team expressed concern that one MetaHuman can replace a large cast of actors in digital environments as a cost effective solution.

The use of avatars is not new to theatre and can be daily experienced on Twitch. MetaHuman, however, and its realistic design, plus ability to use one's own face as a base for diverse intersectional characters, is new to theatre. In our experiments, we see potential for a playful way to utilize MetaHuman and build digital doubles of actors that can appear on screen in other worlds

or be projected to the stage surface. To offer a bad pun, it's possible for an actor to get meta with one's digital self. We urge, however, caution as there are many ethical considerations—such as, the ability to alter one's race—that must be thoughtfully engaged with when using these tools.

8. Conclusions

8.1 What to consider

Technology and software can be free (Unreal Engine, MetaHuman Creator), **but it can also have an associated cost** (CONVAI, Polycam) **and sometimes that cost can be significant** (Rokoko Smartsuit Pro, Rokoko Suite, dedicated router, device powerful enough to utilize Unreal Engine and MetaHuman).

Technology and software can take time. While some softwares (Polycam, CONVAI, MetaHuman Creator) are designed to be user friendly and offer quick results, problems still occur. What often takes the most time is 1) learning the technology and/or software; 2) waiting for the software to reload after you've debugged a problem. This report attempts to give a sense of how long it took us to learn different tools, but it doesn't speak to how much of our time was spent waiting for software to reload after a problem had been addressed. If theatre artists engage with these technologies and software, they should give themselves ample extra time in their process to do so appropriately, otherwise, they should look to hire an expert.

Technology and software does require some level of expertise (even for the novice). A novice can and should engage with these technologies, but they won't be able to apply them in their artistic work until some level of experience is achieved. Some of our projects (like connecting CONVAI to the MetaHuman or Rokoko to the MetaHuman) would not have been possible without an experienced C++ coder as part of our team.

Technology and Software updates. An update can be good as it can protect a device from viruses and offer new commands that help solve an earlier problem. Updates can also, as we learnt, lead you to lose your work. CONVAI updated in 2024 and because we connect CONVAI to other plugins, we had to retrain our AIBrian character. New controls in CONVAI (such as a new algorithm to help the NPC focus on their objective) created new problems (like even more fixation on the objective) requiring new prompting methods (Chain of Thought prompting) to make the character more flexible. Updates mean it is vital to find ways to save work or to select technologies, software, and plugins that allow one to save work.

Technology and software are limited. Technology and software were built with biases in place. The most obvious bias is often who will be utilizing these technologies and softwares. The majority of tools we engaged with were built for video game designers. When you want the tool to execute a task that is not necessary in a video game, the limitations of the technology and software often become apparent. Sometimes these tools are built without nuanced consideration for the questions around intersectionality it raises. In an attempt to celebrate diversity, Metahuman

Creator allows users to choose the race/class/gender/ability/age/size of a character. For a white user building a black MetaHuman, this should be an ethical consideration as it could be considered a form of digital blackface. Additionally, CONVAI contains a filter that can prevent the large language model from saying anything deemed hateful or harmful. One's bias regarding using or not using the filter needs careful consideration before determining what to do.

Technology and software necessitates research. It is important to have a sense of what you hope to achieve in a project and research which technology and which software is the best for that task. We researched many 3D scan apps but landed on Polycam because of its consistent results and free trial. CONVAI was one of 6 different AI plugins we considered for the MetaHuman. Sam, our C++ coder, compared the specs for each plugin and considered what the limitations were if we wanted to further experiment with creating our own voice for CONVAI (which requires the ability to connect it to another plugin). It is also important to research the technology required to support software. Sometimes cheap software requires more expensive technology than is available. For example, as our Macbook Pro's weren't sufficient, we had to borrow an Alienware Laptop (Dell) from Dr. Kris Alexander to utilize the MetaHuman we built in *2021.* Research is vital before you commit to a technology and/or software.

Technology and software can and should be experimented with. Thanks to the scope of Pure Research, we often began our experiments with "What if?". This curiosity led us to discoveries we never would have made otherwise. Studying Unreal Engine introduced Cole to MetaHuman Creator. Sam's ability to code in C++ allowed us to consider plugins to see what else the MetaHuman is capable of. What if the MetaHuman has a voice? What if the MetaHuman can improvise a dialogue with us? What if the MetaHuman can move? These questions led us to new discoveries that we experimented with and were able to achieve. It is additionally encouraged to experiment beyond the scope of what the technology, software, or code was created for. For example, by applying the syntax of coding to text analysis or using some of the software (CONVAI, MetaHuman Creator) for text analysis, we discovered that the biases of theatre artists can be revealed via the choices they make.

8.2 Questions

We processed our research conclusion with ChatGPT 4. The following are a combination of ours and its dramaturgical questions for theatre-makers to consider when engaging with these technologies and/or software in relation to performance.

Budgetary Consideration

+ How does the cost (monetary and time-based) of technology and software impact the accessibility of these tools for theatre artists?
+ What artistic decisions must be reconsidered when the software or technology intended for a project introduces unexpected costs or limitations?
+ How do the costs and requirements of the technology impact the overall accessibility of the production for smaller or lower-budget theatres, and what alternative solutions can be explored?
+ What might a budget need to consider and how can this be articulated to funders?

Time Management and Workflow

+ In what ways can theatre artists ensure they allocate sufficient time for learning and troubleshooting technology in their creative process?
+ What strategies can theatre artists use to bridge the gap between novice and expert levels of technological proficiency in a team?
+ How can an artist's process change when updates risk erasing significant progress in a project?
+ What measures can be implemented to safeguard creative work in a digital process that depends on potentially unstable software?
+ What does a production schedule need to consider when technology and software are collaborators on a production?

Bias and Ethical Considerations

+ How does the bias embedded in technology and software shape the representation of intersectional characters and stories on stage?
+ In what ways can theatre artists critically engage with the ethical implications of using avatar-building tools that might reinforce or subvert stereotypes?
+ What ethical concerns arise when creating digital representations of identities different from one's own?
+ Is the theatre maker critically evaluating the assumptions embedded in the design of the tools they're using, particularly if the tools were created for a different industry (e.g., gaming)? And do these assumptions shape the possibilities for storytelling in one's work, and do they introduce any unexamined biases?
+ What responsibility do artists have in considering the biases and filters within large language models when incorporating them into a production?
+ How do theatre artists navigate the filter settings, and what are the implications of disabling or keeping the filters in place? Are theatre artists comfortable with the restrictions imposed by the filter, or does disabling it allow for problematic language or themes to surface that need careful consideration?
+ How do theatre artists actively interrogate the intersectional biases that may exist in the algorithms they're using, ensuring their work doesn't perpetuate harmful narratives or exclusions?

Accessibility for Artists and Audiences

+ What support systems (training, experts, etc.) can be put in place to address varying levels of expertise among the creative team?
+ In what ways can the technology and software enhance the accessibility of the performance for diverse audiences, including those with disabilities?
+ Can features like audio description, captioning, or interactive design elements be incorporated to make the experience more inclusive?

In Support of Process

+ How can technology and software be integrated into the rehearsal and development process to facilitate collaboration, experimentation, and communication among the creative team?
+ In what ways can technology help theatre artists document and reflect on their creative process, enabling deeper exploration and revision of artistic choices?
+ How can technology be used as a dramaturgical tool during the creative development of a piece, allowing for “live” exploration of thematic elements, spatial relationships, or narrative possibilities?

Creative Vision and Experimentation

+ How do the limitations of technology and software, designed for fields like gaming, influence the boundaries of experimentation in a theatrical context?
+ How does the necessity for research in selecting technology and software affect the spontaneity or intuition often central to theatre-making?
+ How can theatre artists ensure that the technology they choose enhances, rather than dictates, the artistic vision of a project?
+ How can theatre artists ensure that the technology is necessary conceptually and not being utilized only because it is new or innovative or “cool”?
+ What role does experimentation with technology and software play in challenging conventional theatre forms and pushing the boundaries of performance?
+ How can "What if?" questions guide both the technological and creative aspects of a production, and what risks do such explorations involve?

8.3 Provocations

In the spirit of our research, we continued to use our conclusion to collaborate with ChatGPT 4 and offer the following provocations to theatre makers:

1. **What if the technology you rely on for performance is instead used to transform how you rehearse and develop ideas?**

2. **How does the act of troubleshooting technical glitches reveal new creative pathways, rather than hinder your process?**

3. **In a world of constant updates, can you embrace impermanence and change as essential to your artistic practice?**

4. **How do biases embedded in the software you use impact not just the stories you tell, but the stories you choose to explore?**

5. **Can you imagine a rehearsal process where digital experimentation is as integral as actor improvisation—how might that shift your vision?**

THANK YOU

A huge thank you. This research would not have been possible without your generous assistance:

Nightswimming (Gloria Mok, Brian Quirt), Tarragon Theatre (Mike Payette, Jeff Ho, Justin Miller), Design + Technology Lab (Jonathon Anderson, Adrian Kenny), Dr. Kris Alexander, Christopher-Elizabeth Boyd, Michael Bergmann, Theo Bakker, Lisa Karen Cox, Emma Cuzzocrea, Adam Delgado, Ezri Fenton, 郝邦宇 Steven Hao, Iris O'Neill, Ayanna Seesahai, Montserrat Videla Samper, and Daibei Wang.