EXPLORING THE INTEGRATION OF MACHINE LEARNING AND ANIMATRONIC ATTENTION

Undergraduate Students: Chayse Joseph Inniss, Cole Garolis-Bechtold, David Sepulveda, Icarus Newton, Isabella de Sousa Proulx, Lucas Tuan, Lucia Alday, Makena Dundon, Nathaniel Bidwell, Richard Posthuma, Tyler Bunnow
Faculty Advisor: Dr. Michael Marcellin
The University of Arizona Department of Electrical and Computer Engineering
Tucson, AZ, 85721

ABSTRACT

Advancements in Artificial Intelligence (AI) are transforming the themed entertainment industry by enhancing animatronic engagement and immersion. Traditional animatronics rely on pre-programmed sequences or human operators, limiting their responsiveness to guests. Exploring AI and camera tracking can enable animatronics to interact dynamically with large crowds and adjust behavior based on real-time audience actions. By integrating telemetry, sensors, and machine learning, animatronics can focus their attention and respond more naturally to human movements and sounds. Adaptive systems like these create a more immersive and lifelike experience, reducing the need for human control and improving overall guest satisfaction.

INTRODUCTION

The use of animatronics has expanded within the themed entertainment industry over time. Despite animatronics stemming from the realms of clocks and puppetry, they have become a versatile tool that communicates with guests on a more personal level. Rather than relying on screens or costumed performers to interact with guests, operators can use animatronics to perform consistently within a physical medium. By taking advantage of this resource, you allow the integration of new technology, such as the use of AI, to readily switch from the original manually controlled actions or the purely repetitive programmed actions to a new, intuitive call-based action response between a guest or operator and the animatronic system itself.

LITERATURE REVIEW

Audio animatronics are life-like robots made to embody a specific character or creature. They are puppeted by electronics that are usually pre-programmed with a specific 'script' to follow, which is looped. There are a few animatronics that are puppeted live via an operator, but this requires hiring someone to control the animatronic. By utilizing AI, it is possible to have a completely autonomous animatronic that can react to outside stimuli to interact with guests.

There are currently advancements being made in the animatronic and robot fields utilizing AI. One example is Cozmo, a small robot toy that utilizes AI and cameras to recognize emotions and mirror human responses and interactions [1]. In the world of audio animatronics, Walt Disney Imagineering is utilizing algorithms to give individual animatronics unique personalities [12]. These animatronics, known as Vyloo, are located in the queue of Disney's California Adventure's attraction Guardians of the Galaxy-Mission Breakout. The Vyloo are programmed with AI that decides how they should interact with the guests around them. The AI makes decisions about how long to hold eye contact, if it wants to sleep, and how quickly it looks around. Each Vyloo's programming is slightly different to give each one a distinctive personality.

Walt Disney Imagineering is also currently in the research and development phase of a fully autonomous animatronic that can roam freely and interact with guests. These new "Automatronics" combine traditional audio animatronic technology with AI to create characters that can decide where they go, what they do, and how they interact with guests, all in real time and without human input.

With these advancements, an animatronic could play a larger role in themed entertainment settings such as stage shows or roaming characters. Since these animatronics would have more options than simply a pre-animated loop, they can interact with guests, respond to other characters nearby, or simply look at who is speaking.

At the basic level, an AI algorithm is one that can learn and grow based on pre-existing data and complete tasks based on that training information. Implementing AI into animatronics to enable autonomous movement requires two types of AI models: one to process audio spoken by guests, transcribing it into a readable text, and another to process the text and generate a command for the animatronic to execute. The first model is a predictive AI, which analyzes the data in audio files, and as the name suggests, predicts what was most likely spoken. The second model is a generative AI to process the analysis of the first model, and translate it into code instructions

Speech recognition models are commonly used in modern life, appearing in search engines and personal assistants. Most speech recognition algorithms make use of supervised and unsupervised training data [9]. These types of training data are fed directly to the AI so that it can recognize patterns. Supervised learning refers to a type of data that is clearly labeled for the purpose of training the model to directly recognize the patterns [11]. For example, feeding a speech recognition AI model an audio file to process, and telling the AI exactly what it says, so

that it can recognize those patterns in the future. Unsupervised learning refers to a similar type of data that is fed directly to the AI, but unlabeled, meaning that the model has to draw conclusions and recognize patterns independently. These types of AI data training are effective in audio processing and speech recognition, and transcribing because these models rely on pattern recognition.

AI can be used with animatronics and in their development. Some parts of the development process require simulations and modeling that, when using traditional computing methods, can be very time-consuming, resulting in slow iterations and development. An example of this is simulations of animatronic facial movements. It is important for designers to understand how the designs of the motors and the physical components will interact with each other to create animations that look natural. By using machine learning, it is possible to create systems that accurately predict these movements in a much shorter timeframe than with older methods, resulting in a faster development process and allowing designers to spend more time creating better animatronics [14].

Additionally, AI is used to control the movements of robots, allowing them to be adaptable to environments by using AI to control their motion instead of a human controller or pre-programmed movements. These AI models are trained using reinforcement learning, which has the models run simulations of different environments and has them learn how to best react to the environment by rewarding them when they do the right thing, such as walking correctly or standing when meeting an obstacle [10]. AI is further used to generate the environments to train these models in, allowing for many more training environments to be created and for much more thoroughly experienced models [8]. By training these models in diverse environments, it is possible to create robots that are much more adaptable to different situations than with pre-programmed movements or a human controller. Already, these methods are being used by companies like NVIDIA and Disney to create robots and animatronics that are capable of moving around complicated environments [2, 7]. In order for these robots to function, they also need high-quality telemetry and sensors to read and react to their environments.

Another type of learning that is widely used in AI models, useful in this application, is deep learning. Deep learning is classified by the way the information is processed, rather than how the information is presented to the model. In deep learning, there are many layers of processing, which fall under three categories: input, hidden, and output [6]. The input layer is where the raw data is received, and the output layer is where the data is transformed into the result. The hidden layers are where the analysis and processing take place, such as feature extraction, language identification, speech recognition, and translation, steps in WhisperAI [3]. Deep learning algorithms can have any number of hidden layers, depending on the application.

If an animatronic is to respond to verbal cues, then the algorithm has to recognize those cues and interpret them. One of the best ways to accomplish this is by using a speech-to-text algorithm that translates spoken words into written words for the computer to analyze. There are countless speech-to-text algorithms that already exist, such as the ones built into Google and Siri that many people use daily. OpenAI's model, called Whisper, is a great starting point for applying speech

recognition in theme parks because it has been specifically designed to understand speech across many dialects and accents. Whisper is able to do this because it is trained “on vast amounts of multilingual and multitask-supervised data” [3]. Theme parks attract guests from all over the world, so a model that is able to understand other languages and dialects is useful in order to connect to everyone at the park, not just the native language of the location of the park.

In order to transcribe the audio into text, the algorithm needs an audio file in MP3, MP4, MPEG, mpga, m4a, wav, or webm formats. The algorithm can then output the transcription in JSON, text, SRT, verbose_JSON, or VTT formats. For the applications in animatronic auto-response, transcribing the audio prompts into simple text is ideal for simple interpretation and transfer to motion commands, to then make use of other algorithms to interpret and analyze.

The algorithm functions in 6 basic steps: Audio Preprocessing, Feature Extraction, Language Identification, Speech Recognition, Translation, and Post-processing. In audio processing, spectrograms are created by slicing the audio file into small parts. In feature extraction, algorithms analyze the spectrograms using the values to determine the audio file’s qualities, both acoustic and linguistic. If the language identification algorithm correctly determines the language, then speech recognition predicts what was likely spoken using the qualities determined by feature extraction. If required, it is translated into the desired language, and then this written output is revised to be grammatically, linguistically, and dialectically correct to match the desired outcome.

The tool that allows the two systems to communicate with each other is an Application Programming Interface (API). An API is a set of rules that allows two software systems to communicate with each other [5]. To control the information that goes between these systems, an API key is created. The API key serves as a security checkpoint, allowing only specific information with the correct API key to be accessed by the corresponding system. When a request is sent from one system to another, the API key checks and ensures that the request is valid. If the API key is valid, the API will allow you to perform the requested action or return the requested information [4]. With the use of API keys, animatronics can be enhanced by allowing only specific phrases to prompt particular movements. By taking a direct phrase that the API key accepts, the animatronic will be able to interact with guests in a more lifelike way, creating more unique experiences.

As well as audio inputs, visual data from cameras can be used to help animatronics determine where to focus their attention. Computer vision technologies are actually already being used in surveillance systems to detect crowd density, movement, and even emotional states in real time. For example, a system developed for city-wide surveillance has re+purposed existing CCTV infrastructure to detect human motion and analyze crowd behavior using background subtraction and Histogram of Oriented Gradients (HOG) in combination with a Support Vector Machine (SVM) classifier. This setup has enabled the system to recognize pedestrians in restricted areas and flag violent crowd behavior based on motion patterns and optical flow data. Although this example has been designed for public safety, these methods could also be adapted for themed entertainment environments to help animatronics identify where the largest groups are gathered

or where high-energy or emotional activity is occurring. This kind of crowd detection could allow animatronics to shift their gaze, direct their movement, or trigger specific animation sequences based on audience behavior, which would both improve immersion and reduce the need for human operators. However, for the scope of this test, only audio assistance was used.

OUR TESTING

For our testing, we used Python to create a script that connects the open-source AI model Whisper to the software required to program a Bottango animatronic. Bottango is a company whose mission is to make animatronics accessible to the everyday person. One of their animatronics is Maxwell, a bird animatronic we will be using for testing rather than fabricating one. Alongside Maxwell, Bottango created software that allows a person to easily animate and control animatronics. This software can be used to test Arduino servo motors, code, and AI capabilities on an animatronic. To achieve this, we first created a script that imported the model and converted an audio file into a text format, which could then be interpreted and manipulated. After this, the script uses an API key and localhost ports on the computer to communicate with the Bottango software and the physical animatronic, which must also be connected to the computer through hardware.

Figure 1: *Bottango Animatronic Maxwell with Animation Software*

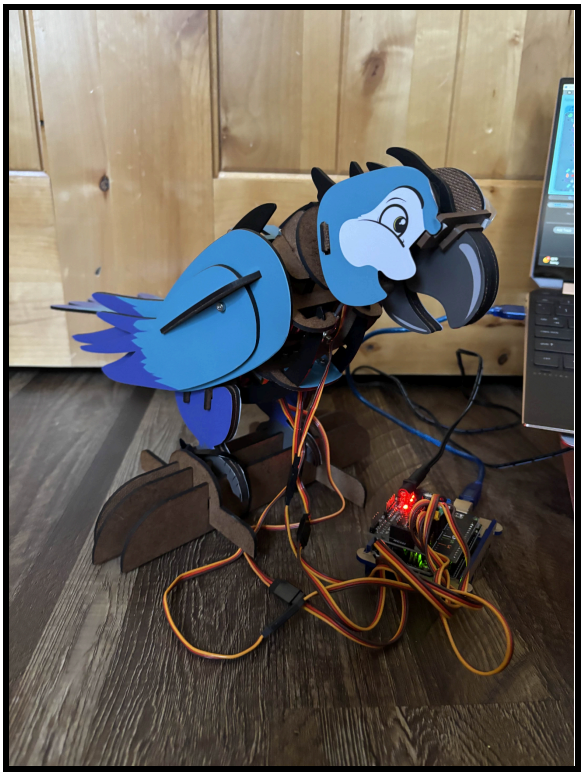

Figure 2: *Maxwell Bottango Animatronic*

PROCEDURE

The code for that returns the values obtained from an audio file as a string, which can be further processed through code, is as follows:

```
import whisper

def transcribe_audio(file_path):

    model = whisper.load_model("medium")

    result = model.transcribe(file_path)

    return {
        "identifier": "transcriptionResult",
        "value": result["text"]
    }
```

After the modified AI model transcribes the audio, the string-formatted data is parsed, and then the preprogrammed responses are retrieved and called based on keywords through the use of the following script.

```
# original script obtained from EvanBottango/Bottango
# edits made by Lucia Alday and Richard Posthuma
import requests
import json

COMMANDS = {
    "hi": "sayHi",
    "fly": "canYouFly",
    "speak": "speakOrCaw",
    "rest": None
}

def parse_transcript(transcript: str):
    transcript = transcript.lower()

    for keyword, api_id in COMMANDS.items():
        if keyword in transcript:
            return keyword
    return "None Found"

def request():
    port = 59224
    baseUrl = 'http://localhost:{}/'.format(port)

    requestUrl = baseUrl + 'ControlInput/'
    try:
        requestParams = {}
        # set identifier for recording input
        identifier = 'myIdentifier'

        requestParams['identifier'] = identifier
        requestParams['value'] = 0.5
        response = requests.put(requestUrl, json=requestParams)
        response.raise_for_status()

    except requests.exceptions.RequestException as e:
        print('Run failed\n')
        raise SystemExit(e)

request()
```

This code processes the data and calls the related function to operate within Bottango, allowing for the animatronic to respond in a timely manner. The constraints for time include the fact that

the data must be separated into short partitions to allow for a timely response rate as close to real time as is possible while minimizing the risk of splitting up the key phrases, which would result in the phrases not being recognized.

RESULTS

The method described and tested has many areas where it is prone to error, including unrecognized hardware and compatibility issues. Many of these issues can be solved by reworking the framework so that it is not dependent on Bottango software and instead programs the animatronic through an Arduino interface and system. However, this poses the risk of damaging the animatronic since we'd have none of the built-in failsafes the software supplies. These prevent the motors from extending beyond the physical limits of the joints and motor placements. As such, we opted not to do so as not to impose this risk of harm on the hardware. Once the motions for the animatronic are programmed and ensured not to overextend any joints, the Arduino interface is preferable since it would not rely on a physical connection to a computer device, provide an audio input source, thus allowing the animatronic to be more mobile. Another benefit is that without a computer connection, API keys, which can be prone to hacking, would not be necessary to combine the Python script with the software commands. Instead, all code would be written in C++, as it is the standard Arduino programming language.

CONCLUSION

This study examined the current technology theme parks use for animatronic technology and explored simple and accessible concepts for the implementation of AI features. These would enhance engagement and immersion to create a unique experience for guests in a theme park. Brief experimentation has shown that the utilization of a predictive language model can enable animatronics to create a more interactive experience without manual input from an operator.

During testing, the interaction between Whisper AI and the Bottango software revealed numerous limitations with the hardware's compatibility. Deep learning models, such as Whisper AI, can serve as a beneficial stepping stone for AI animatronics, enabling them to understand multiple languages and accommodate diverse user backgrounds. However, Bottango’s fail-safes for generic animatronic functionality make it difficult for us to properly utilize Whisper AI alongside it.

Future research would prioritize the use of an Arduino to control the servo motors within the animatronic alongside the AI models. This could be demonstrated by integrating the use of an external Arduino control board and the use of sensors for the animatronic to view its

surroundings, to create self-contained hardware. Having these sensors will be a beneficial addition to the animatronic, allowing it to view how many guests are in the surrounding location. With further research, significant improvements in animatronic and guest interactions can make theme park experiences even more immersive for guests.

ACKNOWLEDGEMENTS

We are sincerely grateful to Dr. Michael Marcellin, from the University of Arizona, recent Pioneer Award Recipient, as our club advisor, who facilitated the organization of our club. His guidance and expertise have significantly supported our research, enhancing the quality of our work. We appreciate his support throughout our academic journey.

SOURCES

[1] Anki, “Cozmo Robot,” https://ankicozmorobot.com/

[2] Disney Experiences, “Star Wars: Galaxy’s Edge – Droids,” https://disneyexperiences.com/star-wars-galaxys-edge-droids/

[3] GeeksforGeeks, “Open AI Whisper,” GeeksforGeeks, Noida, Uttar Pradesh, March, 2024.

[4] GeeksforGeeks, “What Is API Key?” https://www.geeksforgeeks.org/what-is-api-key/

[5] IBM, “What Is an API?” https://www.ibm.com/think/topics/api

[6] Mendix, “What Are the Different Types of AI Models?” Mendix, Boston, Massachusetts, January, 2025.

[7] NVIDIA, “Announcing Newton: An Open-Source Physics Engine for Robotics Simulation,” https://developer.nvidia.com/blog/announcing-newton-an-open-source-physics-engine-for-robotics-simulation/

[8] NVIDIA, “Generative Physical AI,” https://www.nvidia.com/en-us/glossary/generative-physical-ai/

[9] OpenAI, “Introducing Whisper,” OpenAI, San Francisco, California, September, 2022.

[10] Sutton, Richard S., and Andrew G. Barto, Reinforcement Learning: An Introduction, 2nd ed., MIT Press, Cambridge, Massachusetts, 2015.

[11] Tableau, “Artificial Intelligence (AI) Algorithms: A Complete Overview,” Tableau Software, Seattle, Washington, Year Not Specified.

[12] Walt Disney Imagineering, “Autonomous Characters,” Walt Disney Imagineering, Glendale, California, June, 2024.

[13] Unknown Author, “Active Learning for Interactive Audio-Animatronic Performance Design,” Paper Number N/A, Journal of Computer Graphics Techniques (JCGT), Conference and Location N/A, Year N/A.

Formatting for citations synthesized by ChatGPT

APPENDIX: ADDITIONAL SETUP PHOTOS

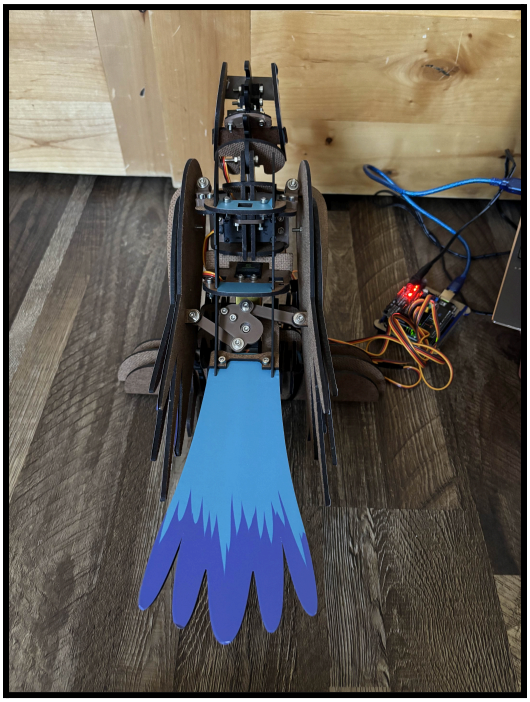

Figure 3: Rear View of Animatronic Bird

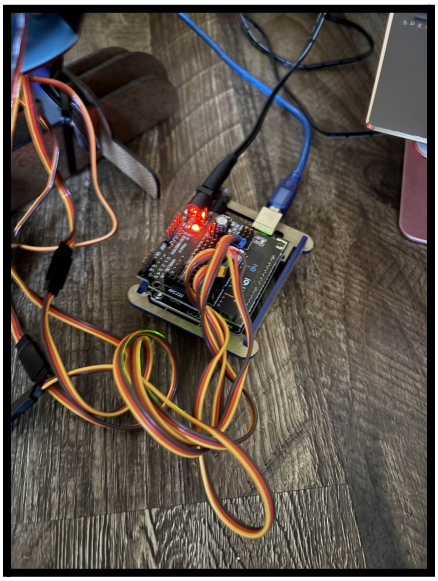

Figure 4: *Animatronic Microcontroller*

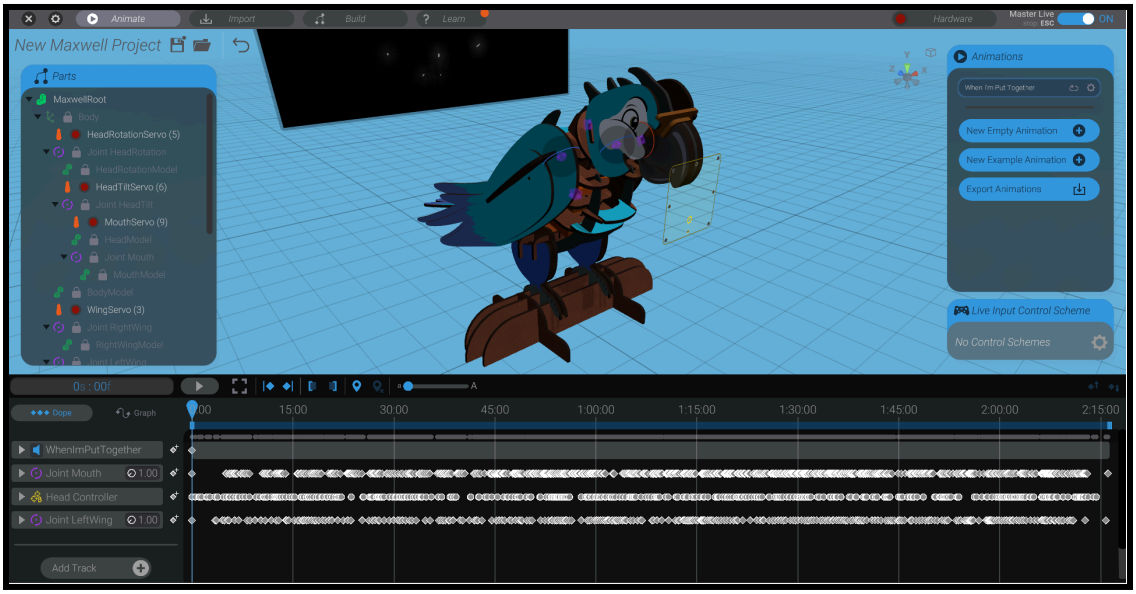

Figure 5: *Screenshot of Bottango Animation Software*

Figure 6: *Code to communicate with Bottango after values are parsed from AI*

```
# authors: Richard Posthuma and Lucia Alday

import requests
import json

COMMANDS = {
    "hi": "sayHi",
    "fly": "canYouFly",
    "speak": "speakOrCaw",
    "rest": None
}

def parse_transcript(transcript: str):
    transcript = transcript.lower()

    for keyword, api_id in COMMANDS.items():
        if keyword in transcript:
            return keyword
    return "None Found"

def request():
    port = 59224
    baseUrl = 'http://localhost:{}/'.format(port)

    requestUrl = baseUrl + 'ControlInput/'
    try:
        requestParams = {}

        # set identifier for recording input
        # TODO: fix input format
        identifier = 'myIdentifier'

        requestParams['identifier'] = identifier        # Required, provide the string
identifier for the input as set up in the input settings menu


        #change the current playhead time in Milliseconds
        requestParams['value'] = 0.5                        # Required, the value to set on
the recording control
                                                            # Movement parts (Joint, Motor,
pose blend axis, etc) expect a value between 0.0 and 1.0
                                                            # On Off events treat any non 0.0
value less than or equal to 1.0 as on, and 0.0 as off
                                                            # Trigger events treat any non 0.0
value less than or equal to 1.0 as a new trigger

        response = requests.put(requestUrl, json=requestParams)
        response.raise_for_status()

    except requests.exceptions.RequestException as e:
        print('Run failed\n')
        raise SystemExit(e)
   request()
```