



Article

Application of ChatGPT-Based Digital Human in Animation Creation

Chong Lan, Yongsheng Wang, Chengze Wang *, Shirong Song and Zheng Gong

School of Art and Design, Lanzhou Jiaotong University, Lanzhou 730070, China; lanchong@lztu.edu.cn (C.L.); wangys@lztu.edu.cn (Y.W.); 12211713@stu.lztu.edu.cn (S.S.); gongz@lztu.edu.cn (Z.G.)

* Correspondence: 12211715@stu.lztu.edu.cn

Abstract: Traditional 3D animation creation involves a process of motion acquisition, dubbing, and mouth movement data binding for each character. To streamline animation creation, we propose combining artificial intelligence (AI) with a motion capture system. This integration aims to reduce the time, workload, and cost associated with animation creation. By utilizing AI and natural language processing, the characters can engage in independent learning, generating their own responses and interactions, thus moving away from the traditional method of creating digital characters with pre-defined behaviors. In this paper, we present an approach that employs a digital person's animation environment. We utilized Unity plug-ins to drive the character's mouth Blendshape, synchronize the character's voice and mouth movements in Unity, and connect the digital person to an AI system. This integration enables AI-driven language interactions within animation production. Through experimentation, we evaluated the correctness of the natural language interaction of the digital human in the animated scene, the real-time synchronization of mouth movements, the potential for singularity in guiding users during digital human animation creation, and its ability to guide user interactions through its own thought process.



Citation: Lan, C.; Wang, Y.; Wang, C.; Song, S.; Gong, Z. Application of ChatGPT-Based Digital Human in Animation Creation. *Future Internet* **2023**, *15*, 300. <https://doi.org/10.3390/fi15090300>

Academic Editors: Antony Bryant, Roberto Montemanni, Min Chen, Paolo Bellavista, Kenji Suzuki, Jeanine Treffers-Daller and Ivan Serina

Received: 14 July 2023

Revised: 29 August 2023

Accepted: 31 August 2023

Published: 2 September 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: digital people; unity; human–computer interaction; animation creation; ChatGPT

1. Introduction

Artificial intelligence (AI) harnesses the power of extensively trained language models [1,2] to facilitate human–computer interactions through natural language [3]. Since the release of the Chat Generative Pre-trained Transformer (ChatGPT) 3.5 model, a natural-language-processing (NLP) model developed by OpenAI specifically designed for generating human-like text in real-time chat environments [4,5], AI has had a significant impact on various industries. ChatGPT is capable of implementing human–computer interaction behaviors in the form of natural language chat, offering a wide range of applications such as assisting with writing, text generation, code writing, translating text, and performing mathematical operations. The demand for AI deep learning (DL) models and the need for reasoning and training continue to grow, leading to the active development of various deep learning accelerations [6]. The application domains of ChatGPT are extensive and span across fields such as medical health [7,8], mathematical analysis [9], automotive industries [10], and education [11–13]. Its conversational AI capabilities significantly enhance productivity, providing human-like assistance in various tasks. As games strive for NPC-to-human fidelity [14], they are an ideal benchmarking ground [15,16]. The traditional fixed action and language interaction of animation creation are now transformed with AI, as it responds to commands to guide human interaction behavior. This approach maximizes the value of digital human applications, making it a quintessential representation of human–computer intelligence in interactive processes.

The promise of VR and AI is arguably that of an ontological and ethical shift, one that takes us closer to a posthuman animation [17]. Widespread use of artificial intelligence

technology may significantly optimize the consumption of digital media technology [18]. Animation creation involves interacting with animated characters based on predefined needs. The interactive behavior between animated characters is endowed with human-like interactions. Digital humans guide characters in performing tasks and transitioning to the next ones. To address the complexity of collaborative production systems, digital twins are utilized [19], and the combination of 3D animation creation and virtual reality technology drives the advancement of animation technology [20]. AI access breathes life into animated digital humans, enabling them to go beyond mere repetition of speech patterns. They possess multiple solution processes and methods, incorporating AI-driven speech recognition, text-to-speech, and animation capabilities to enhance communication with consumers and improve the overall perceived quality of digital human interaction experiences [21]. The behavior of digital humans is informed by research on digital human behavioral learning and trusted agents, employing game-changing AI methods, processes, and algorithms [22]. Based on the studies and surveys conducted, the application of artificial intelligence in animation creation is still relatively rare. It is mostly applied to game characters and player interactions, and the AI models used often come from different sources with limited capabilities. Although there are voice-recognition and -synthesis technologies integrated into voice assistants, they mainly serve as simple AI models for specific applications, such as desktop assistants for commodity users, such as Nubia's Mora [23].

However, the emergence of ChatGPT has brought about more-creative integration of speech synthesis and recognition in tasks such as text and code writing and modification. In the past, implementing mouth animations for digital humans was a complex process performed programmatically. In this experiment, we used a plug-in to drive the digital human's mouth Blendshape, which is a 3D modeling technique that adjusts the shape based on set parameters. By changing these parameters, specific animation effects, such as expression changes and limb postures, can be achieved while keeping the external shape unchanged.

Therefore, we successfully integrated the large language model ChatGPT into the animation creation of the digital human, enabling natural language processing and human interaction. Through the combination of ChatGPT, speech recognition, and speech synthesis, along with real-time synchronization of the digital human's mouth movements and voice, we created a digital human animation creation aid. This innovative approach facilitates smoother and more-efficient interactions between animation creators and the digital human, making animation creation a more-interactive and -engaging process.

In this thesis, we present the selection and utilization of relevant technologies, while also analyzing their respective advantages and disadvantages. Firstly, we focused on the Unity plug-in binding for the digital human's mouth Blendshape, enabling real-time synchronization of mouth movements to achieve a more-natural-speaking form closely resembling human speech. Additionally, we performed motion acquisition through motion-capture devices, capturing hand movements and body movements when the digital human speaks. For successful digital human voice interaction, we integrated automatic speech recognition (ASR) and text-to-speech (TTS) technologies. This process involves the digital human receiving voice information, which is converted into text through ASR technology. The AI then recognizes the text and generates a response, which is later transformed back into voice information through TTS technology. Finally, the digital human's mouth movements are driven by the voice to complete the process of creating an artificial-intelligence-based digital human. To summarize, the main contributions of this thesis are as follows:

1. The integration of automatic-speech-recognition technology and speech-synthesis-technology in animation creation.
2. The utilization of ChatGPT for achieving interactions with the digital human.
3. The implementation of the Unity plug-in SALSA with RandomEyes to synchronize the digital person's mouth movement with sound.

2. Overview of Related Technologies

Prior to conducting the research, we studied the animation-creation process, focusing on character movement. We also analyzed essential technical and practical aspects for creating animated digital characters. The research aimed to enable AI to interact with characters through algorithms, comprehend natural language information, analyze the data, and provide relevant responses to guide the animated characters. To achieve this, we integrated various technologies, including digital human action acquisition, automatic speech recognition, speech synthesis, and ChatGPT.

2.1. Automatic Speech Recognition and Text-to-Speech Technologies

Automatic speech recognition (ASR) transforms speech data into text data by recognizing the lexical information of human speech [24], while text-to-speech (TTS) synthesizes speech information using text data through computer processing [25]. Both ASR and TTS technologies are commonly utilized in the form of application program interfaces (APIs), local deployments, and training models to facilitate data exchange between text and speech through data processing. Leading providers such as Google, Microsoft Azure, and Baidu Intelligent Cloud offer ASR and TTS technologies connected through APIs, which employ proprietary API secret keys to handle text and language data through tokens. Additionally, API connectivity allows for data transfer via custom-trained voice models such as variational inference with adversarial learning for end-to-end text-to-speech (VITS), which enhances the personalization of the language model and mitigates the reliance on a single data conversion API [26]. API-based data processing involves sending sound data or text data to the corresponding server, which then analyzes and converts it into text or language before transmitting it back to the user. For instance, the API workflow of ASR entails obtaining speech data and sending it to the server, which processes it and returns the text data to the user. On the other hand, local deployments and software downloads facilitate language synthesis by analyzing speech data or text data on the local computer and converting them into text data or language data for local users. For example, COEIROINK enables speech-to-text conversion on the local computer, replacing the cloud server with local processing, thereby reducing data transmission latency over the network. Figure 1 illustrates the contrasting data-processing operations of ASR, TTS using APIs, and local deployments. APIs rely on cloud servers for processing relevant speech or text data, whereas local deployments utilize local computers for the same purpose.

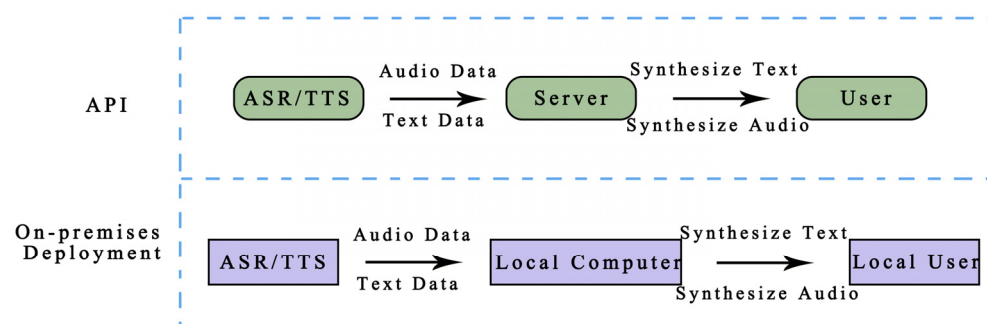


Figure 1. Flowchart of data conversion between ASR, TTS local deployment, and APIs.

Based on the case study of the use of related technologies, both Microsoft Azure and Baidu Intelligent Cloud's voice synthesis technologies have some limitations when it comes to selecting appropriate voice models for broadcasting characters. Baidu Intelligent Cloud, for example, only allows users to choose from pre-set voice models, resulting in synthesized voice data that may sound rigid and mechanical. On the other hand, while Microsoft Azure's voice synthesis comes close to mimicking a real person's voice, the entire application process for voice recognition APIs can be cumbersome. Moreover, it is not possible to register with a credit card in China, hindering widespread deployment in the

region. Deploying ASR and TTS locally with VITS requires a significant amount of local storage space, and the technical process is complex, including the time-consuming training of models and other factors. Taking all these factors into consideration, the building process in this study aims to ensure the authenticity of the digital person's voice by utilizing Microsoft Azure's TTS and Baidu Intelligent Cloud for ASR deployment. This approach allows for the creation of a personalized voice data model to enhance the overall quality of the digital character's speech and interaction.

2.2. Comparative Use of Artificial Intelligence

ChatGPT, a milestone in the journey of AI, has played a pivotal role in driving the surge of artificial intelligence and computer graphics (AICG) [27,28]. The majority of AI applications in the market are available in the form of API access and local deployment, among others. ChatGPT's API accessibility has significantly facilitated its deployment in various application domains. Notably, other AI models similar to ChatGPT have also emerged, such as Wenyan Yixin, Google Bard, and more. These models enable human-computer interaction through natural language on web pages and also offer API access. Furthermore, Tsinghua's open-source model, ChatGLM, bears similarities to ChatGPT and allows for locally deployed text interactions. These interactive AI approaches have gained popularity, and there may be other unexplored AI models yet to be explored and utilized in various domains.

During the experiment, both API access and local deployment were analyzed. API access requires a stable network environment for efficient data exchange, making it more suitable for scenarios with a reliable network connection. On the other hand, local deployment reduces the time difference in network data transmission compared to API access. However, it demands higher hardware configurations, such as ChatGLM, which necessitates over 6 GB of video memory configuration, posing challenges for laptops and less-powerful systems. Moreover, AI models require a substantial amount of data for training, adding to the complexity of local deployment. Considering the available options, API access provides a more-accessible and -versatile range of AI models. Although ChatGPT, Wenyan Yixin, and Google Bard are available through API access, Wenyan Yixin's intelligence level may not be as high, while Google Bard's application queue might be time-consuming. Taking all factors into account, the experimental environment was chosen to utilize ChatGPT, specifically the ChatGPT-3.5 model, through its API interface for the construction of the AI environment, as it proves to be a suitable and efficient choice for the experiment.

2.3. Mouth and Eye Drive

The synchronization of the digital human's mouth movements with the sound is crucial to ensure that the speech appears natural and does not seem out of sync. Extensive information analysis led to the selection of two Unity plug-ins, LipSync Pro 1.532 and SALSA With RandomEyes 1.5.5, for achieving the mouth drive of the digital human. Both plug-ins function by controlling the digital human's Blendshape. However, LipSync Pro 1.532 requires a more-meticulous matching of the mouth Blendshape, which can vary depending on the software used to create the digital human. As a result, configuring the LipSync Pro 1.532 driver plug-in can be relatively troublesome. On the other hand, the SALSA With RandomEyes 1.5.5 plug-in is comparatively more straightforward to configure. It requires matching the mouth opening and amplitude, as well as the eyes, eyelids, and other smaller parts of the Blendshape. This plug-in is the preferred choice for achieving real-time synchronization of sound and mouth movements. Figure 2 illustrates the driving principle of the plug-in: the sound data are transmitted to the Unity sound driver plug-in, which in turn drives the digital human's Blendshape to control the mouth movement. This process ensures real-time synchronization of sound and mouth movements for the digital human, enhancing the overall realism and quality of the animation.

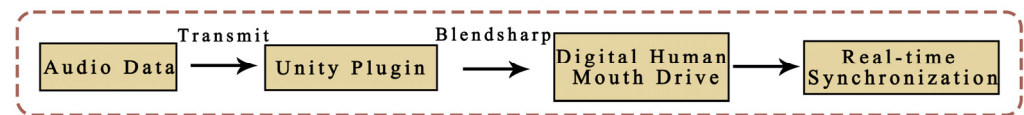


Figure 2. Schematic diagram of a digital human voice data-driven mouth.

3. Development Environment Setup and Parameterization

The ChatGPT digital man plays a crucial role in animation creation, facilitating interactions with the protagonist. To achieve an intelligent digital man environment, it is necessary to construct a setup for acquiring action data and enabling the digital man to perform various actions realistically. This involves utilizing Unity plug-ins to drive the digital man's mouth movements, ensuring synchronization with speech. Unity serves as the development platform, providing the necessary tools and environment for the creation and development of the digital man.

3.1. Acquisition of Digital Hand and Foot Movement Data

The body movements of a digital human, such as hand movements when answering questions and guiding movements when walking, can be achieved using animation scripts in Unity. These motions can be acquired through various methods, such as recording motion data with a motion-capture system or downloading motion data from the Mixamo website. Once acquired, the motion data are imported into Unity in FBX format for further creation.

To import the motion-capture data into Unity (version Unity2020.3.44f1c1), follow these steps:

1. Select the motion capture data and change the Animation Type in the Inspector window to Humanoid.
2. Create an Animator Controller in the Assets folder.
3. Edit the motion capture animation on the Animator window and drag the anim_stack (motion data) into it.

Next, import the Blendshape figure, and follow these steps:

1. Change the Animation Type of the figure to Humanoid.
2. Drag the figure to the Hierarchy for editing.
3. Select the figure, and add an Animator Controller containing the motion capture data to the Controller of the Animator component in the Inspector window.

After setting up the Animator Controller in the Inspector window, click Run to test whether the motion capture data run properly in the environment. Figure 3 illustrates the flowchart of the motion data, starting with the Entry and then proceeding to the id action. After a few seconds, it transitions to the M1 action, followed by M2 and M3, and finally, returns to the id action, in a loop. The settings allow you to adjust the duration of each action and control the flow of the animation sequence.

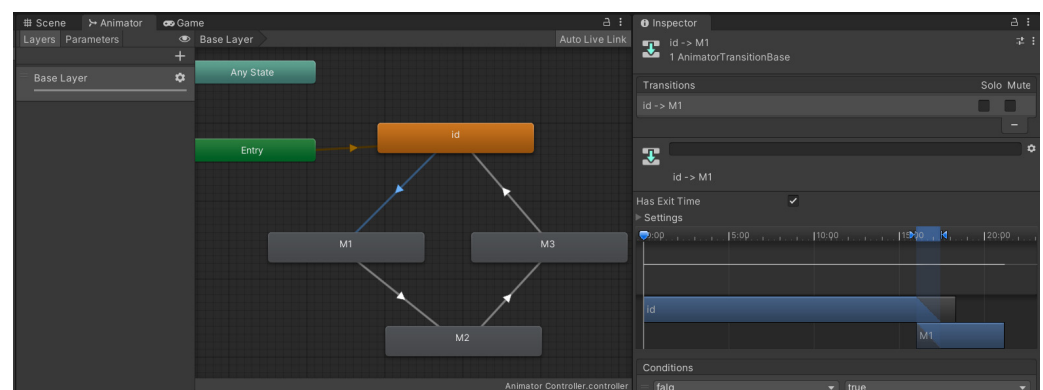


Figure 3. Schematic of action data creation in Unity.

3.2. Unity Plug-in Drives Mouth and Real-Time Voice Synchronization

The Unity plug-in utilized to drive the mouth and eye movements of the digital human is SALSA With RandomEyes Version 1.5.5 [29]. Upon importing the plugin, an analysis of its demo file was conducted. The analysis revealed that the script responsible for controlling the mouth drive is Salsa3D, which drives the Blendshape parameters of the digital human's mouth using sound data. On the other hand, the eye movement parameter is managed by the RandomEyes3D script, which randomizes blinking and eye turning movements by adjusting the Blendshape parameters of the digital human's eyes in a random time-based manner. Incorporating the data analysis from the demo, the digital human's parameters are configured accordingly. This integration ensures the digital human's mouth and eye movements are dynamically synchronized with sound and realistically mimic random eye rotations, creating a more-lifelike and -interactive experience for users.

On the imported digital human, the Salsa3D component is added to the Inspector window to control the real-time synchronization of sound with the digital human's mouth movements. Additionally, the RandomEyes3D component is added to achieve random eye rotations.

To set up the Salsa3D component, follow these steps:

1. In the Skinned Mesh Renderer of the Salsa3D component, select the sub-level object of the digital human that contains the Blendshapes for the mouth. In the experiment, the Blendshapes for the digital human's mouth are located on the digital human's body.
2. The source of the Audio Source depends on the text-to-speech sound source. In the experiment, the AzureTTS sound source is selected.
3. Adjust the parameters of the digital human's mouth opening degree. The small-amplitude index of the sound corresponds to the small-amplitude mouth action in Blendshapes; the medium-amplitude index corresponds to the medium-amplitude Blendshape; the large-amplitude index corresponds to the high-amplitude Blendshape. This parameter adjustment ensures that the sound's amplitude controls the corresponding mouth movement size.

For the parameter settings of the RandomEyes3D component, follow these steps:

1. In the Skinned Mesh Renderer of the RandomEyes3D component, find the sub-level object of the digital human that contains the Blendshapes for the eyes.
2. The Eye Shape Indexes in the RandomEyes3D component correspond to the Blendshapes for Eye Up, Down, Left, Right, and Blink. Place the corresponding Blendshapes in the appropriate positions to achieve the desired eye movements, including up, down, left, right, and blinking.

By configuring the Salsa3D and RandomEyes3D components with the appropriate Blendshapes and parameters, the digital human can realistically synchronize its mouth movements with the sound and exhibit random eye rotations, providing a more-engaging and -lifelike animation experience. The parameter settings are displayed in Figure 4.

3.3. UI-Related Settings

The UI consists of three main partitions: the voice message input area, the message display area, and the history message view area:

1. Voice message input area: This area includes buttons that enable the functionality of Baidu Intelligent Cloud's automatic voice recognition. The Baidu Recognition Script is attached to the created button, allowing the long pressing of the button to record and recognize the voice. The recognized voice is then sent to the Baidu Intelligent Cloud server via the API.
2. Message display area: This section displays the ongoing conversation messages between the user and the digital human.
3. History message view area: The history start button is located in this area. Clicking the button triggers the display of the History UI while simultaneously closing the original main interface. To implement this functionality, you can use the On Click ()

event and select the ChatScript.OpenAndGetHistory method. Drag the Script Empty Object to the Select Object to establish the connection.

By organizing the UI elements in these partitions and implementing the appropriate scripts and events, the user interface will facilitate voice message input, display the conversation messages, and provide access to the history view area.

To set up the history message viewing area for recording the text records of the dialog messages, follow these steps:

1. Add a Scroll Rect component to the history message view area. This component will enable scrolling through the text records.
2. Set up a script for the back button in the history message view area. In the On Click () event, select the ChatScript.BackChatMode method and drag the Empty Script Object to the Select Object. This script will handle the functionality of going back to the main UI page from the history view.

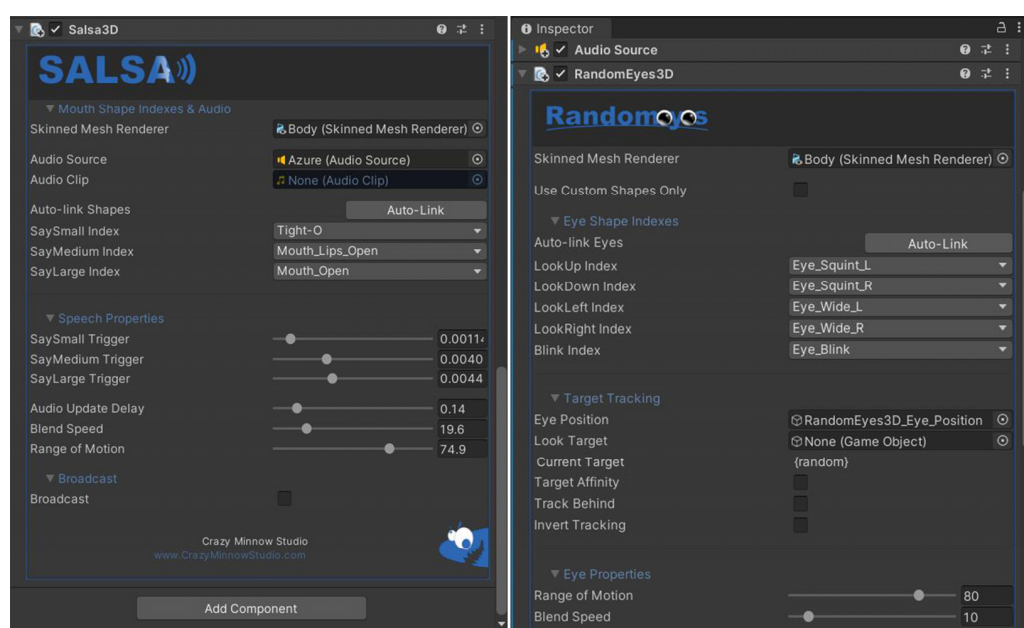


Figure 4. SALSA with RandomEyes matching parameter map for driving digital people.

By incorporating these steps, the history message view area will be functional, allowing users to view and scroll through the recorded text dialog messages. The back button will provide a seamless transition from the history view to the main UI page. Figure 5 shows the parameter settings for the button.

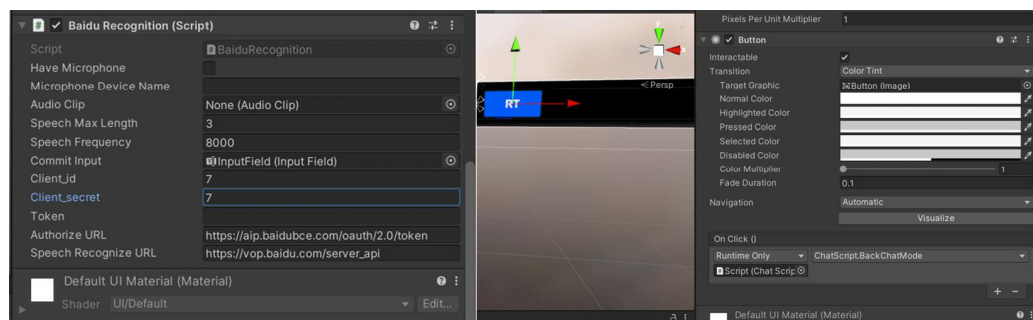


Figure 5. Schematic of Baidu Speech Recognition script and parameter adjustment for history button.

After establishing the hand, foot, mouth, and eye drive environment, the next step is to realize the external API connection and programming data transfer by pre-deploying the

necessary environment for data exchange. This will enable the digital human to interact with external APIs and process data effectively during the animation creation process.

4. Related API Calls

The programming implementation utilizes the C# language, which is mainly used for API access of ChatGPT and the technical implementation of ASR and TTS for digital people [30–32].

4.1. ChatGPT's API Access to Digital People

In Unity, an empty object named AI-Turbo is created to hold the C# programming code for implementing a dialog interaction function with the OpenAI GPT-3.5 Turbo model. The code facilitates sending messages from the user and receiving replies from the model by making API calls to OpenAI's chat model.

A new C# script is created in Unity to define the GptTurboScript class, which contains the necessary information and methods to interact with the OpenAI API. The class includes the following variables:

1. `m_ApiUrl`: represents the address of the OpenAI API.
2. `m_GPTModel`: represents the type of GPT model used.
3. `m_DataList`: used to cache sent and received messages in the dialog.

Prompt: represents the AI persona.

In the `Start()` method, the AI persona is added to the message list as the initial state of the conversation when the program is run.

The `GetPostData()` method in the code is responsible for calling OpenAI's chat model API. It creates a `UnityWebRequest` object and sets the relevant parameters of the request, including the content of the message to be sent and the authorization information. The request is then sent to the OpenAI API, and the code waits for a response. Upon receiving the response, the status code is checked to verify its success. If the response is successful, the content of the response is parsed into JSON format, the model's reply to the user's message is extracted, and it is added to the message list. Finally, the callback function is called to pass the model's reply to the caller, completing the dialog interaction process with the GPT-3.5 Turbo model.

To implement a prefabricated body function for the chat interface in Unity, a new C# script can be created. This script will be responsible for creating a dialog box in the chat interface and displaying chat messages by setting the text content.

The `ChatPrefab` class in the code should have a private field named `m_Text` of type `Text`. This `Text` field will be used to display the chat text in the dialog box.

The `SetText()` method in the code is used to set the text content to be displayed in the dialog box. This method takes the text content as input and sets it to the text property of the `m_Text` field. By doing so, the text content will be displayed in the chat dialog box.

To use the prefab and create chat interfaces with settable text content, follow these steps:

1. Add the `ChatPrefab` script to the UI element in the Unity editor.
2. Associate the `Text` component of the UI element with the `m_Text` field in the `ChatPrefab` script.

By following these steps, the chat interface can be created in Unity, and you can dynamically set the text content to be displayed in the chat dialog box using the `SetText()` method.

4.2. Digital Human ASR and TTS Realization

To implement the automatic speech recognition (ASR) functionality, two C# scripts are needed: one to control the ASR API mobilization and related functions and the other to control the text-to-speech (TTS) API and other functions.

The ASR script is responsible for implementing the Baidu Speech Recognition function and includes functions such as recording voice, obtaining the Token for Baidu Speech Recognition, and sending voice data for recognition. Unity's API is used to record the input

voice. It checks for the presence of a voice input device and retrieves the device name. Then, it starts recording the sound by calling the `BeginSpeechRecord()` method, which utilizes Unity's Microphone class to record a sound clip and store it in `m_AudioClip`. The recording duration and frequency can be set with parameters. When the key is released, the `EndSpeechRecord()` method is called to stop the recording and send the recorded audio data to the Baidu Speech Recognition API for recognition.

In the Baidu Speech Recognition section, it is necessary to obtain the Token first. The `GetToken()` method sends a request to Baidu's authorization API, passing the API Key and Secret Key, and obtains the Token, which serves as the identity credentials used to access the Baidu Speech Recognition API. The recorded voice data are then converted to PCM16 format using the `GetBaiduRecognize()` method and sent as part of the request to the Baidu Speech Recognition API. The API returns the recognized text as a result, which is passed to the `RecognizeBack()` method for processing. Finally, the `RecognizeBack()` method displays the recognition result in the input box and may call other functions, sending the results to the OpenAI interface for further processing.

Second, the AzureTTS script is used to implement the speech synthesis feature using Microsoft Azure Speech Synthesis. To do this, the `Microsoft.CognitiveServices.Speech.1.26.0` plug-in is imported into Unity, which is used to drive the speech synthesis service for converting text to speech and playing back the generated audio through Unity's `AudioSource` component:

1. The script creates an empty object named "AzureTTS" in Unity and adds the `AudioSource` component to it. Then, a C# script is created that introduces the necessary namespaces and defines several variables and components:
 - `audioSource`: used to play the synthesized audio.
 - `subscriptionKey`: the subscription key applied to the Azure voice service for authentication and access to the voice service.
 - `region`: the region where the voice service is located.
 - `m_SoundSetting`: used for the selected `VoiceSynthesisSoundSetting`.
2. The `Start()` method is implemented, which is called when the script starts and is used to initialize the configuration and synthesizer for speech synthesis. It creates a `SpeechConfig` instance with `SubscriptionKey` and `Region` and sets the audio output format and voice settings. Then, it creates a `SpeechSynthesizer` instance and registers a `SynthesisCanceled` event-handler method.
3. The `SetSound()` method is implemented to toggle the sound settings for speech synthesis. It updates the `m_SoundSetting` variable and the `SpeechSynthesisVoiceName` property of `SpeechConfig` and recreates the `SpeechSynthesizer` instance.
4. The `TurnTextToSpeech()` method is implemented, which receives a string parameter representing the text to be synthesized into speech. It ensures multithreading safety using the lock keyword. It calls the `StartSpeakingTextAsync` method of the `SpeechSynthesizer` to start synthesizing speech and reads the generated audio data through the `AudioDataStream`. It creates an `AudioClip` and plays the audio using Unity's `AudioSource`. When the audio playback finishes, it ends the audio playback by setting the `audioSourceNeedStop` flag. During speech synthesis, it records the synthesis start time via `DateTime` and calculates the synthesis delay when the first audio block is generated. The waiting status and messages are updated using the lock keyword.

These two pieces of code effectively implement speech recognition and text-to-speech conversion and playback by utilizing Baidu Intelligent Cloud, Microsoft Azure API calls, related settings, Microsoft Cognitive Services Speech SDK, and Unity's `AudioSource` component.

4.3. ASR, TTS Technology, and ChatGPT Integration

Based on the Microsoft Azure and Baidu Intelligent Cloud API programming, we can integrate ChatGPT API calls and text message processing to leverage the Unity engine and OpenAI's GPT-3.5 model for natural-language-processing and conversation-generation

functions. We created an empty object named “Script” to place the ChatScript script. The ChatScript class inherits from MonoBehaviour and is responsible for handling the chat function.

The ChatScript class in the code is designed to manage various private fields and references for handling API keys, API URLs, and UI elements of the chat interface. It includes functionality for typing messages, receiving AI replies, and communicating with the OpenAI API. The API request is executed using Unity’s UnityWebRequest class, where the parameters and headers are properly set for sending a POST request to the OpenAI API. The response from the API is processed accordingly.

In the SendData() method, user-entered messages are logged into the chat history and sent to the OpenAI API as the content of the API request. This request is made through a call to the GetPostData() method, and the reply message is then passed to the Callback() method for further processing upon a successful request. The Callback() method updates the text content of the chat interface and can optionally synthesize and play the AI’s reply based on certain settings.

Additionally, the code includes helpful functions such as displaying text verbatim, retrieving the chat history, and displaying the chat log. These functions are integrated with Unity’s UI components to enhance the user experience and provide a more-engaging chat interaction.

5. Testing and Analysis

When the users interact with the digital character, they have the option of selecting either text or voice input. If the users choose voice input, they can long-press the voice button, which will show “Recording” in gray color. The recorded voice will be sent to the automatic speech recognition (ASR) server for processing. The ASR server will then convert the voice information into text and display it on the UI interface. Simultaneously, the corresponding text information will be sent to the OpenAI server for generating a reply in text format. The generated reply from ChatGPT will be displayed on the UI interface, and the text information will be sent to the text-to-speech (TTS) server. The TTS server will convert the text into voice information and drive the digital character’s mouth to match the speech. The final result will be presented to the user. In the case of text input, the ASR processing step is skipped, and the text is directly sent to the OpenAI server for generating a reply. The overall structure of this application system is illustrated in the following Figure 6.

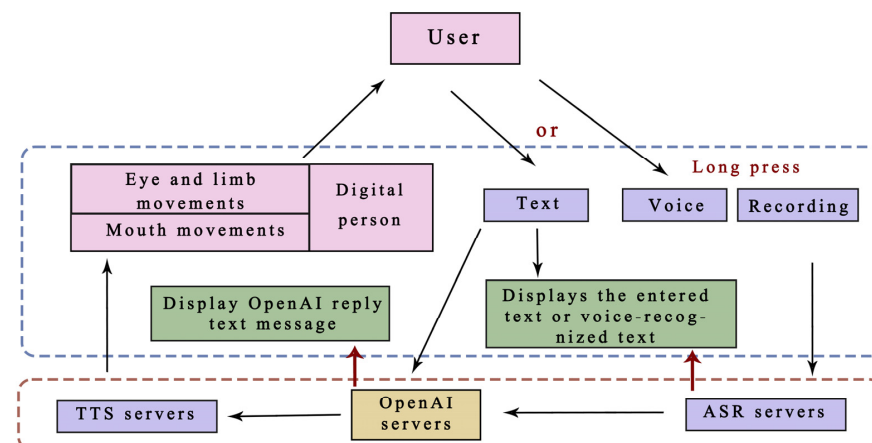


Figure 6. Application system overall structure diagram.

After completing the construction of the environment based on the ChatGPT digital person, it is essential to conduct tests to evaluate the effectiveness and efficiency of the ASR and TTS systems. Subsequently, the accuracy rate of ChatGPT’s API text responses and the success rate of response delivery should be assessed. Finally, the real-time syn-

chronization between the digital person's mouth movements and the corresponding voice information needs to be tested, and the experimental data should be analyzed to draw meaningful conclusions.

5.1. System Testing

The overall operation of the entire system relies heavily on the API connection response and the cooperative functioning of the driver plug-in. We conducted tests on the environment based on the following aspects:

1. ASR and TTS API connection test: In the ASR module, human voice recording is carried out, followed by obtaining the Access Token. The recorded audio is converted to pulse code modulation (PCM) format, which is a lossless digital audio coding used to convert analog audio signals to digital signals. The audio data are then sent to the Baidu Speech Recognition API server using the HTTP protocol. An HTTP POST request is constructed containing the audio data and other necessary parameters such as language type, sampling rate, audio format, etc., which is then sent to the Baidu Intelligent Cloud API server. We wait for the server to respond, and finally, the server processes the audio data and returns the recognition results. The TTS API works in a similar manner, where we obtain the Access Token, construct an HTTP request, send it to the server, and receive the synthesized speech data.
2. ChatGPT's API reply test: ChatGPT's text conversion operates similarly to ASR and TTS, where data transmission is conducted through the API. In the experiment, we set ChatGPT's role, age, identity, and main characters and tested its performance.
3. Unity plug-in-driven digital human test: The primary purpose of the plug-in test was to verify if the sound and digital human mouth movements can be synchronized in real-time. This test involved driving the mouth movements of digital human characters to ensure smooth synchronization with the spoken audio.

By conducting these tests, we aimed to evaluate the performance, accuracy, and real-time synchronization of the different components in the system. Figure 7 is the experimental system flow.

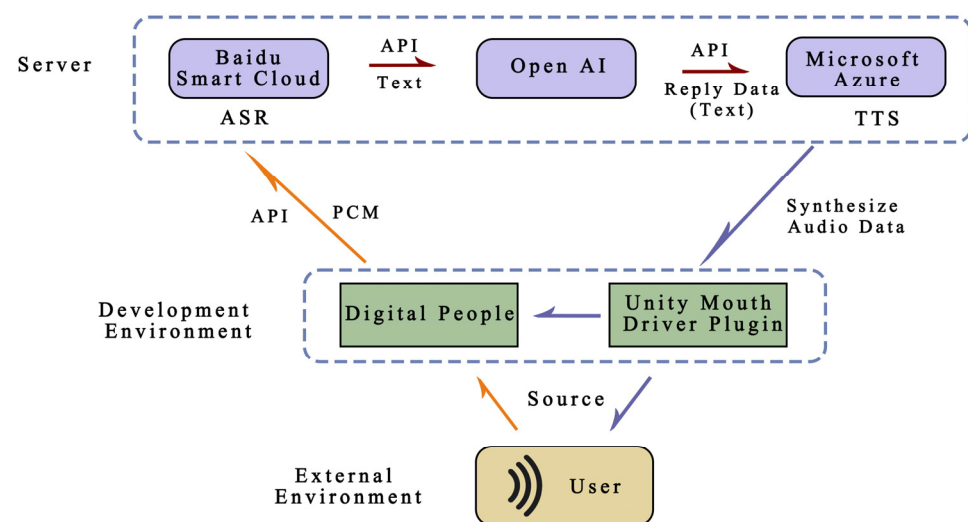


Figure 7. ChatGPT digital man operation principle flowchart.

Based on the principles mentioned above, the following experiments were conducted:

1. Digital human identity definition test.
2. ASR and TTS response test.
3. Digital human mouth drive test.

After conducting ten tests for each item on a single digital person with a consistent network environment, we observed that the digital person's identity definition, ASR, TTS response, and mouth drive test were relatively accurate. However, we encountered some inaccuracies in the sixth test, which can be attributed to the fact that the API interface packages used for ASR belong to short speech recognition and are not capable of recognizing long speech. To enable long speech recognition, we would need to obtain the API secret key for the corresponding long speech package. Table 1 shows the test results of this experiment.

Table 1. Digital people test program and test results (✓ means the test was successful, and × means the test failed).

Test Items	1	2	3	4	5	6	7	8	9	10
Digital human identity definition	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Response to ASR, TTS	✓	✓	✓	✓	✓	×	✓	✓	✓	✓
Digital human mouth drive	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Overall, the test environment was set up correctly for the next experiment, and we will continue to explore further improvements and advancements in the application of artificial intelligence in animation creation with the digital human.

5.2. Animation Creation Interaction and Data Analysis

AI identity and psychological rights can have indirect effects on the interaction process [33], and the definition of AI identity will influence its moral behavior. Once the development environment was set up, we conducted data analysis on ASR, TTS, and ChatGPT response times, text-to-speech time differences, speech and mouth drive time differences, and character-restricted animated verbal interactions. For the interaction test, we initially restricted the role of the digital human to an animation creator named Alan, aged 25, who can assist users with questions related to the animation process, but cannot answer questions on other topics.

5.2.1. Animation Creation Interaction Test

Based on the digital person identity qualification after the digital person for animation creation of the professional aspects of language interaction and according to different situations for the details of processing, to animation creation in the story writing and sub-shot design as an example of testing, the test was performed as follows:

- Digital human capacity: Me: What will you do? Digital Human: As an animation creator. I have expertise in various aspects of animation production including:
 1. Concept Development: I can help you come up with creative ideas and concepts for your animation project. Whether it's brainstorming characters, settings, or storylines, I can assist in developing a unique and engaging concept.
 2. Storyboarding: I can create a storyboard to visually plan out the shots and sequences of your animation. This helps in capturing the overall flow and storytelling of your project.
 3. Character Design: I have experience in designing characters that are visually appealing, memorable, and suitable for your animation style and target audience.
 4. Animation Techniques: I possess knowledge of different animation techniques, such as traditional hand-drawn animation.
- Animation script writing: Me: Help me write an animated script. Digital Human: Of course! I'd be happy to help you write an animation script. However, I would need more information about the specific story, characters, and themes you would like to include in the script. Please provide me with some details or ideas you have, such as the setting, the main characters, and the central conflict or plot. With that information I can start crafting a script that aligns your vision.

- Me: Do you know how to animate a split screen? Digital Human: Yes, as an animation creator, I am familiar with storyboarding, which is the process of planning out an animation shot by shot. I can help you with creating a storyboard for your animation project. Let me know if you have any specific questions or need assistance with a particular aspect of storyboarding, and I'll do my best to assist you! Me: Help me with a couple subplots. Digital Human: Certainly! Here are a few sample storyboard frames based on the background you provided:

Frame 1:

Location: Living room—Day

Description: The main character, a 25-year-old man named John, is seen arguing with his parents. There appears to be tension and frustration in their expressions and body language.

Frame 2:

Location: Outside—Night

Description: John is leaving home in anger. He slams the door behind him and storms off into the night. The scene is visually dynamic, portraying John's emotional state.

Frame 3:

Location: Hospital room—Day

Description: John receives a call notifying him that his parents were involved in an accident. Shocked and worried, he rushes to the hospital, realizing the gravity of the situation.

Frame 4:

Location: Hospital waiting area—Day

Description: John sits in the waiting area, anxious and guilty. He reflects on the arguments and realizes his parents' sacrifices and the challenges they faced while raising him.

These are just a few key scenes to get you started. Feel free to elaborate on these or provide further details if you'd like additional frames or specific actions to be included in the storyboard. Figure 8 is a screenshot of the interaction between the user and the digital human.

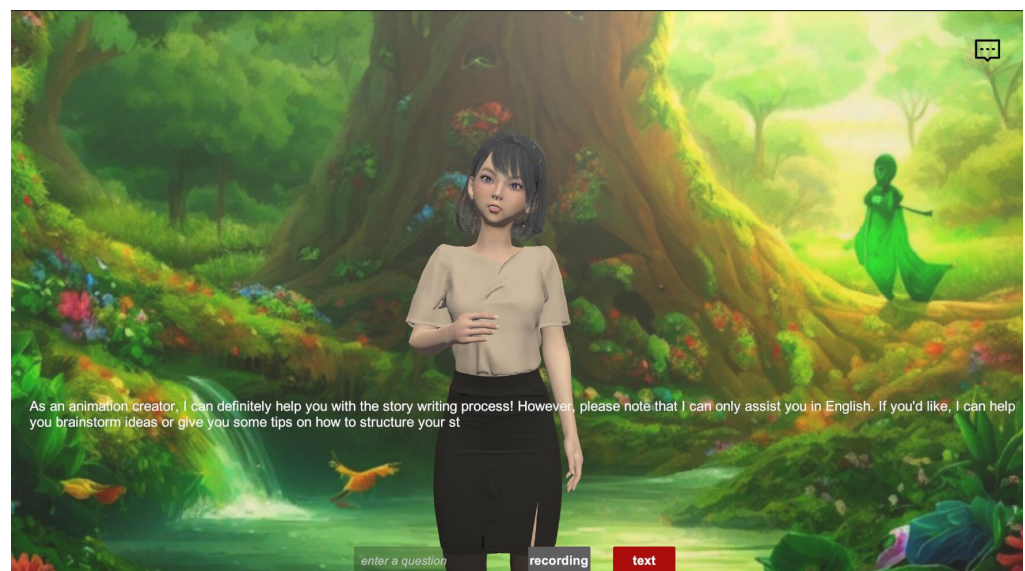


Figure 8. ChatGPT digital human interaction test.

In addition, we tested digital people asking questions outside of animation creation:

Me:

Can you teach me to cook?

Digital Human:

I'm sorry, but as an animation creator, my expertise is focused on animation creation and not on cooking or teaching others how to cook. I suggest reaching out to experienced chefs or online cooking tutorials for assistance in learning how to cook. If you have any questions related to animation creation, feel free to ask, and I'll be happy to help.

Me:

Can you teach me to sing?

Digital Human:

I apologize, but my expertise lies in animation creation and not in teaching singing or providing vocal coaching. For singing lessons and guidance, it would be best to seek the assistance of a professional vocal coach or take singing lessons from a qualified instructor. If you have any questions related to animation creation or need help with the animation process, I'll be glad to assist you.

Regarding the limitations set for the digital person's identity, the digital person can effectively address user inquiries related to animation creation and provide relevant examples to assist users in their animation projects. However, when presented with questions beyond the scope of animation creation, the digital person acknowledges its limitations in specific technical areas, but still offers solutions or insights in related fields. The digital person strives to be helpful and accommodating, even if it cannot provide direct answers to questions outside its expertise in animation creation. Figure 9 shows the interaction history between the user and the digital human.

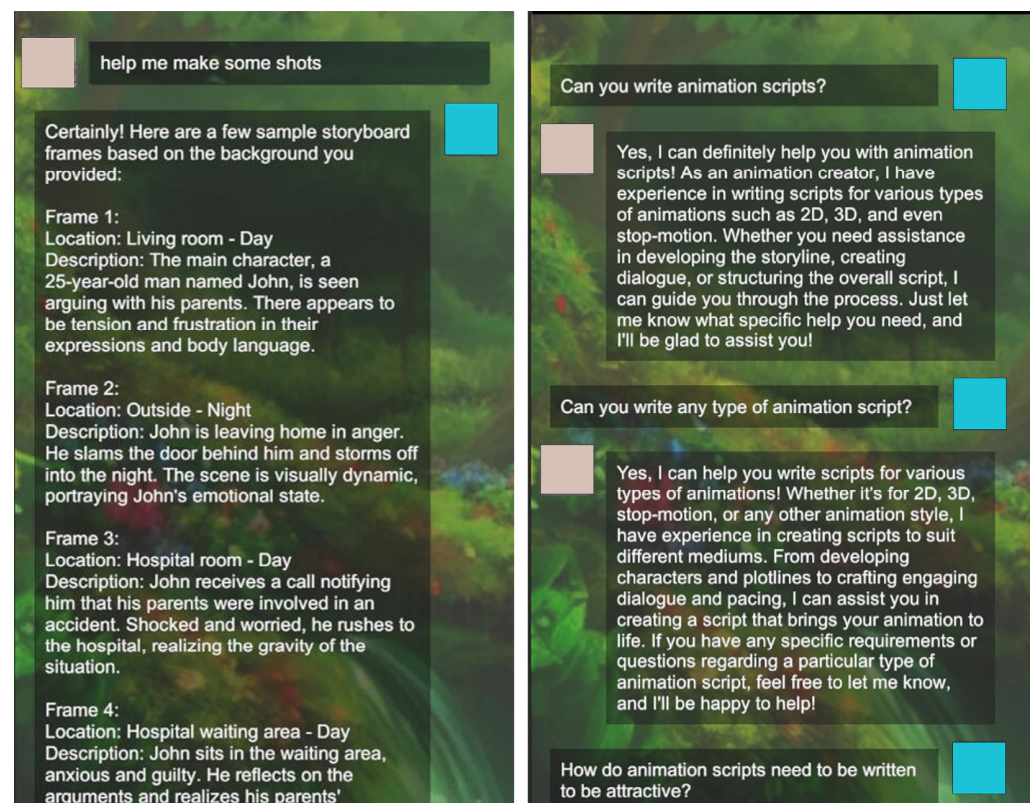


Figure 9. ChatGPT digital human-character interaction history.

5.2.2. Distance between Time Test

The API transmission process of ASR, TTS, and ChatGPT involves sending requests to the server, receiving server responses, and synthesizing corresponding audio, text, and other data. Let us consider the function expression for Baidu Intelligent Cloud ASR:

Let A_d be the start time of the audio recording sent to the server and A_r be the reception time of the server's response sent to the Open AI server. The time difference A_d can be calculated using the following formula:

$$A_d = A_r - A_a \quad (1)$$

Let A_d denote the time difference between sending the audio recording and the reception by the Open AI server. A_r denotes the reception time when the Baidu Intelligent Cloud server sends the response to be sent to the Open AI server. A_a denotes the start time when the audio recording is sent to the Baidu Intelligent Cloud server.

Next, let us consider the TTS function expression for Microsoft Azure:

Let T_a be the start time of the Open AI response data sent to the Microsoft server, T_r be the reception time of the Microsoft server analyzing the data to synthesize the audio data to be sent to the client, and T_d be the time difference between the two. It is expressed by the following formula:

$$T_d = T_r - T_a \quad (2)$$

where T_d denotes the time difference between the time Open AI sends the text data and the time the audio data are received at the client. T_r denotes the time the audio data are received at the client. T_a denotes the start time when Open AI sends the text to the Microsoft server.

Finally, let us consider ChatGPT expressed as a function of data transfer through the API.

Set C_d as the time difference between the two, and its formula is:

$$C_d = T_a - A_r \quad (3)$$

where C_d denotes the time difference between the reception of the text data by the Open AI server and the reception by the Microsoft server.

Set G as the total time difference. The data were analyzed by the following. The data in Table 2 are the results of the experimental process.

Table 2. Number of times and time difference between ASR, TTS, and ChatGPT data exchanges in the form of APIs.

Time (s)	A_d	T_d	C_d	G
1–3	<1	2	3	5
	1	2	6	9
	1	8	8	17
4–5	1	3	3	7
	<1	3	4	7
...
48–50	6	2	3	11
	<1	6	7	14
	1	5	8	14

Based on the above data analysis, A_d was generally within 1 s, with only 1 or 2 tests showing higher values of over 6 s out of 15 and 48 times, indicating that the ASR recognition time of Baidu Intelligent Cloud is relatively fast, but there were a few instances with longer waiting times due to multiple recognitions. The fluctuating value of C_d ranged from 3 to 9 s, indicating that ChatGPT's responses through Open AI take time, especially for more-professional and -difficult questions that require a longer processing time. The fluctuating value of T_d is relatively large, ranging from 2 to 8 s, indicating that TTS takes a longer time to synthesize speech through the API.

5.3. Test Results

Based on the above analysis of the animation-creation-interaction test and the time difference test, the animation-creation-interaction test showed relatively accurate answers in the field of animation creation and provided helpful examples for the animation creator's creative process. However, for areas outside of animation creation, the digital human provided relevant solutions without directly answering specific questions.

During the 50 tests, there were cases where TTS synthesis failed and ChatGPT could not provide a reply in the 30th, 32nd, and 40th tests. Further experiments revealed that TTS synthesis failure may be attributed to data loss caused by an unstable API and network factors, while ChatGPT's inability to reply resulted from its inability to comprehend and respond to certain questions.

In the interaction, the digital human's body and eyes displayed randomized cyclic movements, while the mouth movements were synchronized with the sound. However, the mouth opening amplitude was limited by the Blendshape of the SALSA With RandomEyes plug-in, which provided only three amplitudes.

In the time difference test, the process of sending voice data to the Baidu Intelligent Cloud server, receiving replies from the Open AI server, synthesizing voice data using Microsoft Azure, and driving the digital human's mouth exhibited unstable timing. The total time of about 5 to 15 s, with some instances taking up to 17 s, might be influenced by complex questions and network instability.

During the 50 tests conducted, it was observed that Baidu Intelligent Cloud ASR can be relatively unstable, especially when there are long pauses in the tone of voice, leading to incomplete sentence recognition and voice data errors. Out of the 50 tests, 6 instances of voice recognition inaccuracies were encountered. Among these, 4 times, the voice data were too long, causing recognition to be cut off before completion, and the other 2 times were due to incorrect recognition of certain words said. Upon investigation, it was discovered that the Baidu Intelligent Cloud application used for ASR is configured to receive short speech-recognition packages rather than long speech-recognition packages. This is likely the reason for the incomplete recognition of longer sentences and the errors in recognizing similar-sounding words.

It is important to note that the API interface used in our experimental testing of ASR and TTS is a free trial interface, and not a paid commercial interface. As a result, the experimental effects may not be as stable over a long period of time as in a paid commercial setting. Further testing and optimization will be needed to ensure more-consistent and -reliable results in future experiments.

6. Conclusions

In this paper, we primarily applied ASR and TTS to digital humans through Unity, introduced ChatGPT into the digital humans via API integration, and synchronized the mouth movements of the digital humans with the voice data generated by TTS. We deployed the developed environment in animation creation to facilitate natural language interactions between users and ChatGPT-powered digital humans. Additionally, we collaborated with animation creators to engage in fundamental animation tasks such as script writing and sub-shot design. These include script writing, sub-shot design, API latency assessment, animation interaction, and other experimental tests. The aim was to assist animation creators in efficiently producing animation content with the support of intelligent digital entities.

Furthermore, during our research experiments, we gained insights and experiences from various communities and forums. Throughout the research process, we identified both advantages and disadvantages of related technologies, such as ASR utilization, invocation of the ChatGPT API, and the mechanization of TTS. These discoveries enriched the robustness of our experiments and guided our research direction. We developed a deeper understanding of the real-world effects and potential challenges associated with these technologies. Valuable reference sources included GitHub, YouTube, Bilibili, and

the instructional materials of ARS- and TTS-related websites. Engaging with these platforms provided invaluable experiences and insights, which clarified the trajectory of our technology application and enabled thorough analysis.

Although assisted animation creation can be achieved using ChatGPT, it is not always guaranteed to respond accurately. It may generate relevant responses or even produce incorrect results. In most instances, ChatGPT's outcomes were reasonably accurate, and it is crucial to maintain a balanced perspective when assessing its outputs. Overall, the integration of ChatGPT within a system that incorporates multiple technologies proves advantageous for animation creation, contributing to improved production efficiency.

In our future endeavors, our focus will be on optimizing the system to enhance the stability of both ASR and TTS components. This will enable the system to serve as a valuable creative assistant for users. In terms of system expansion, we plan to integrate it into diverse application environments. For instance, within our ongoing virtual tour of the meta-universe, we intend to deploy the technology as virtual tour guides. This application will facilitate interactive engagements with travelers, thereby enhancing scene interactivity and immersion. Furthermore, we aim to harness the technology's potential by utilizing it for configuring non-playable characters (NPCs) in games. This application will enable a wide array of interactive behaviors, paving the way for exploration in diverse multi-domain application environments (software open-sourced at: <https://github.com/wangchengze01/GPT-animation.git>, accessed on 28 August 2023).

Author Contributions: C.W. conducted the overall system construction research and thesis writing; C.L.: project management; Y.W.: technical guidance and thesis review; S.S.: system testing and analysis; Z.G.: application programming guidance and analysis. All authors have read and agreed to the published version of the manuscript.

Funding: Funding support for the Natural Science Foundation of Gansu Province 2022: Research on Animation Technology Innovation Based on Meta-Universe Platform (Grant No. 22JR5RA357).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Miao, Q.; Zheng, W.; Lv, Y.; Huang, M.; Ding, W.; Wang, F.-Y. DAO to HANOI via DeSci: AI Paradigm Shifts from AlphaGo to ChatGPT. *IEEE/CAA J. Autom. Sin.* **2023**, *10*, 877–897. [\[CrossRef\]](#)
2. Stokel-Walker, C.; Van Noorden, R. What ChatGPT and generative AI mean for science. *Nature* **2023**, *614*, 214–216. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Alharbi, S.; Alrazgan, M.; Alrashed, A.; Alnomasi, T.; Almojel, R.; Alharbi, R.; Alharbi, S.; Alturki, S.; Alshehri, F.; Almojel, M. Automatic Speech Recognition: Systematic Literature Review. *IEEE Access* **2021**, *9*, 131858–131876. [\[CrossRef\]](#)
4. Cai, S.; Yang, L. Study on the Risk and Collaborative Governance of ChatGPT Intelligent Robot Application. *Intell. Theory Pract.* **2023**, 1–11. Available online: <http://kns.cnki.net/kcms/detail/11.1762.G3.20230406.1618.008.html> (accessed on 3 April 2023).
5. OpenAI. No Date Provided. Introducing ChatGPT. Available online: <https://openai.com/blog/chatgpt> (accessed on 3 April 2023).
6. Kwon, H.; Kwon, Y.; Han, J. Backward Graph Construction and Lowering in DL Compiler for Model Training on AI Accelerators. In Proceedings of the 2022 19th International SoC Design Conference (ISOCC), Gangneung-si, Republic of Korea, 19–22 October 2022; pp. 91–92.
7. Temsah, M.-H.; Aljamaan, F.; Malki, K.H.; Alhasan, K.; Altamimi, I.; Aljarbou, R.; Bazuhair, F.; Alsubaihin, A.; Abdulmajeed, N.; Alshahrani, F.S.; et al. ChatGPT and the Future of Digital Health: A Study on Healthcare Workers' Perceptions and Expectations. *Healthcare* **2023**, *11*, 1812. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Gebrael, G.; Sahu, K.K.; Chigarira, B.; Tripathi, N.; Mathew Thomas, V.; Sayegh, N.; Maughan, B.L.; Agarwal, N.; Swami, U.; Li, H. Enhancing Triage Efficiency and Accuracy in Emergency Rooms for Patients with Metastatic Prostate Cancer: A Retrospective Analysis of Artificial Intelligence-Assisted Triage Using ChatGPT 4.0. *Cancers* **2023**, *15*, 3717. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Lee, M. A Mathematical Investigation of Hallucination and Creativity in GPT Models. *Mathematics* **2023**, *11*, 2320. [\[CrossRef\]](#)
10. Luo, Z.; Yan, S.; Luo, S. Multitask Fine Tuning on Pretrained Language Model for Retrieval-Based Question Answering in Automotive Domain. *Mathematics* **2023**, *11*, 2733. [\[CrossRef\]](#)
11. Sánchez-Ruiz, L.M.; Moll-López, S.; Nuñez-Pérez, A.; Moraño-Fernández, J.A.; Vega-Fleitas, E. ChatGPT Challenges Blended Learning Methodologies in Engineering Education: A Case Study in Mathematics. *Appl. Sci.* **2023**, *13*, 6039. [\[CrossRef\]](#)
12. Rahman, M.M.; Watanobe, Y. ChatGPT for Education and Research: Opportunities, Threats, and Strategies. *Appl. Sci.* **2023**, *13*, 5783. [\[CrossRef\]](#)

13. Birenbaum, M. The Chatbots' Challenge to Education: Disruption or Destruction? *Educ. Sci.* **2023**, *13*, 711. [CrossRef]
14. Meng, F.; Hyung, C.J. Research on Multi-NPC Marine Game AI System based on Q-learning Algorithm. In Proceedings of the 2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 24–26 June 2022; pp. 648–652.
15. Yannakakis, G.N.; Togelius, J. A Panorama of Artificial and Computational Intelligence in Games. In *IEEE Transactions on Computational Intelligence and AI in Games*; IEEE: Piscataway, NJ, USA; Volume 7, pp. 317–335. [CrossRef]
16. Simonov, A.; Zagarskikh, A.; Fedorov, V. Applying Behavior characteristics to decision-making process to create believable game AI. *Procedia Comput. Sci.* **2019**, *156*, 404–413. [CrossRef]
17. Dare, D.E. AI/VR: Situated Animation in the Library of Babel. In Proceedings of the 2018 IEEE 1st Workshop on Animation in Virtual and Augmented Environments (ANIVAE), Reutlingen, Germany, 19 March 2018; pp. 1–3.
18. Fan, H. Research on innovation and application of 5G using artificial intelligence-based image and speech recognition technologies. *J. King Saud Univ. Sci.* **2023**, *35*, 102626. [CrossRef]
19. Malik, A.A.; Brem, A. Digital twins for collaborative robots: A case study in human-robot interaction. *Robot. Comput.-Integr. Manuf.* **2021**, *68*, 102092. [CrossRef]
20. Hu, Z.; Liu, L. Research on the application of virtual reality technology in 3D animation creation. *Optik* **2023**, *272*, 170274. [CrossRef]
21. Sung, E.; Han, D.-I.D.; Bae, S.; Kwon, O. What drives technology-enhanced storytelling immersion? *Role Digit. Hum. Comput. Hum. Behav.* **2022**, *132*, 107246. [CrossRef]
22. Yunanto, A.A.; Herumurti, D.; Rochimah, S.; Kuswardayan, I. English Education Game using Non-Player Character Based on Natural Language Processing. *Procedia Comput. Sci.* **2019**, *161*, 502–508. [CrossRef]
23. Mora App. Available online: <https://ui.nubia.cn/app/detail/100> (accessed on 11 August 2022).
24. Li, J.; Deng, L.; Haeb-Umbach, R.; Gong, Y. (Eds.) Chapter 11—Summary and Future Directions. In *A Bridge to Practical Applications; Robust Automatic Speech Recognition*; Academic Press: Cambridge, MA, USA, 2016; pp. 261–280. ISBN 9780128023983. [CrossRef]
25. Zen, H.; Senior, A. Schuster, Statistical parametric speech synthesis using deep neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 7962–7966.
26. Kim, J.; Kong, J.; Son, J. Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech. *arXiv* **2021**, arXiv:2106.06103.
27. What Is AICG and What Areas of Content AICG Can Do. Available online: <https://baijiahao.baidu.com/s?id=1768567295399756174&wfr=spider&for=pc> (accessed on 13 June 2023).
28. AICG Overview Understand the Basic Concepts and Definitions of AICG and Understand the Application of AICG Technology in Film, Television and Animation. Available online: <https://zhuanlan.zhihu.com/p/632434790> (accessed on 5 July 2023).
29. SALSA with RandomEyes (Speech Generation Mouth Shape/Character Speaking) Use. Available online: <https://blog.csdn.net/yigiwoliao/article/details/122389453> (accessed on 5 April 2022).
30. Voice Technology. Available online: <https://ai.baidu.com/ai-doc/SPEECH/qlcirqhz0> (accessed on 2 July 2023).
31. Text-to-Speech Documents. Available online: <https://learn.microsoft.com/zh-cn/azure/cognitive-services/speech-service/index-text-to-speech> (accessed on 30 May 2023).
32. The OpenAI API Uses Documentation. Available online: <https://platform.openai.com/docs/introduction> (accessed on 25 June 2023).
33. Cao, L.; Chen, C.; Dong, X.; Wang, M.; Qin, X. The dark side of AI identity: Investigating when and why AI identity entitles unethical behavior. *Comput. Hum. Behav.* **2023**, *143*, 107669. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.