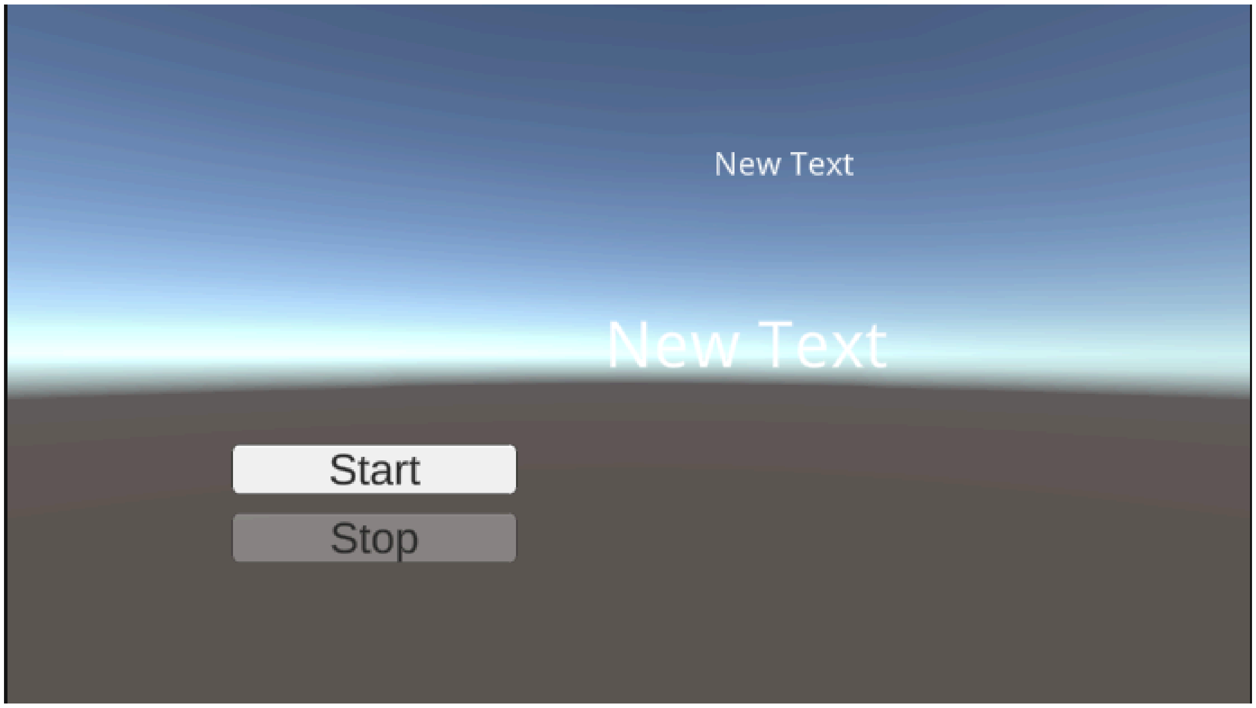
Collaborative VR - Translating text in VR

We broke down our pipeline into three major steps:

1. Speech-to-Text
2. Translation
3. Text-to-Speech



1. Speech-to-Text (STT)

   A. Recording Audio:

   - When the StartRecording method is called, it initializes recording using the microphone.
   - The Microphone.Start function begins recording audio into an AudioClip.
   - The Update method continuously checks if the recording should stop by comparing the current microphone position to the clip's sample count.

   B.Stopping Recording:

   - When the StopRecording method is called, it stops the microphone and retrieves the audio data from the AudioClip.
   - The clip.GetData(samples, 0) method fetches the recorded audio samples.
   - The EncodeAsWAV method converts the audio samples into WAV format bytes.
   - The SendRecording method is then called with the WAV bytes.

C. Sending Recording for STT:

- The SendRecording method uses the Hugging Face API to perform speech recognition on the audio bytes.
- The HuggingFaceAPI.AutomaticSpeechRecognition method sends the audio bytes and handles the response.
- On success, the recognized text is displayed in the speechText UI element, and the translation process begins by calling StartCoroutine(Translate(response, "en", "gu")).

2. Translation

A. Sending Text for Translation:

- The Translate method constructs a URL for the Google Translate API, specifying the source and target languages (en and gu).
- A UnityWebRequest is used to send a GET request to the translation API.

B. Handling Translation Response:

- The translation response is parsed using JsonUtility.FromJson<TranslationResponse>.
- If successful, the translated text is displayed in the translationText UI element.
- The StartCoroutine(TextToSpeech(translatedText)) method is called to start the TTS process with the translated text.

3. Text-to-Speech (TTS)

A. Sending Text for TTS:

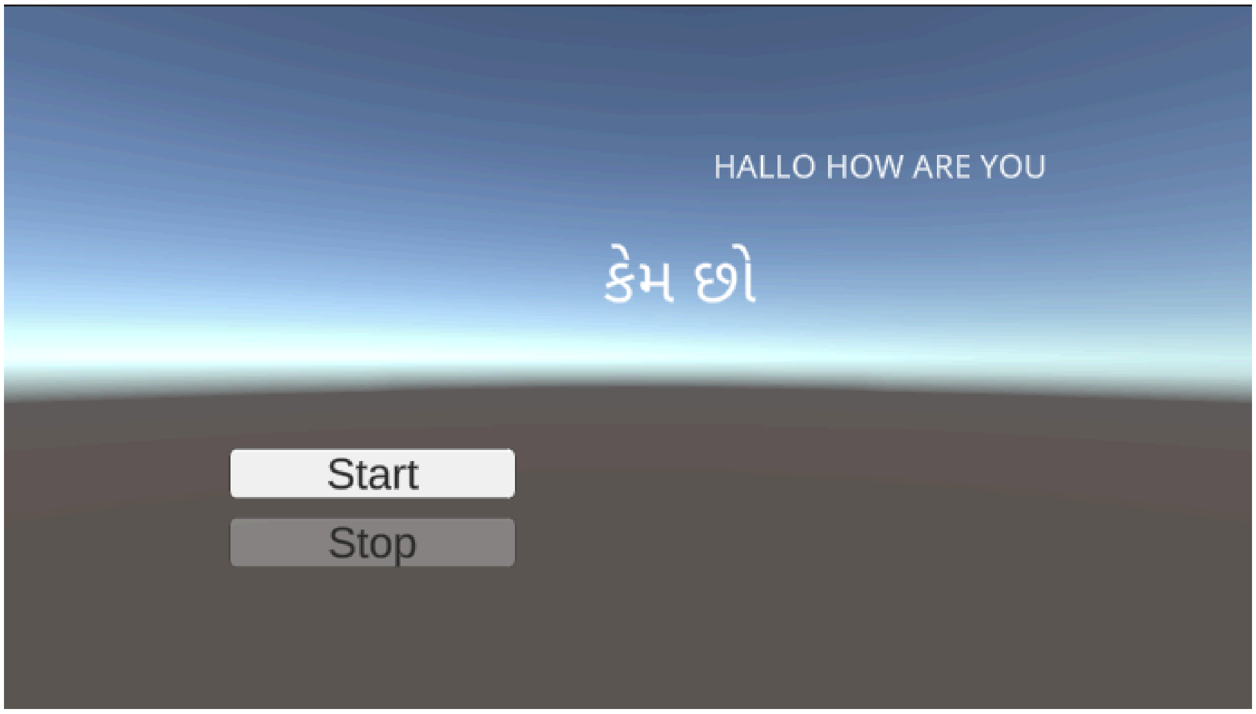
- The TextToSpeech method constructs a JSON request body specifying the text to be synthesized, the voice settings (language code and voice name), and the audio configuration.
- A UnityWebRequest is used to send a POST request to the Google Text-to-Speech API with the constructed JSON.

B. Handling TTS Response:

- The TTS response is parsed using JsonUtility.FromJson<TextToSpeechResponse>.
- The audio content is extracted and converted from a base64 string to a byte array.
- The PlayAudioClip method is called with the byte array to play the synthesized speech.

C. Playing the Synthesized Audio:

- The PlayAudioClip method uses the WAV class to parse the WAV byte data.
- An AudioClip is created from the parsed data.
- An AudioSource component is added to the game object, and the clip is played.



Challenges Faced

Ideally, the translation would be bi-directional where we would be able to identify the spoken language and translate it to the other. We decided to work with Gujarati first.

While the whisper-tiny model via HuggingFace performed very well in capturing English speech, it struggled to identify and understand Gujarati. This caused multiple failures while translating where the language was not recognizable. We also tried using the Google Speech-to-Text API but could not connect it to Unity due to incompatibility of dlls and plugins.