**Real-Time Translator for Video Calls**

**Abstract**

The real-time language translator project aims to bridge communication gaps by instantly translating spoken language between different languages—specifically English and Telugu in the current implementation. Leveraging Python and cloud-based APIs for speech recognition, translation, and text-to-speech synthesis, the system captures audio input, converts it into text, translates the text into the target language, and outputs the translated speech. The project incorporates automatic language detection and robust error handling to ensure reliability. While currently limited to pre-recorded audio and dependent on internet connectivity, the framework establishes a foundation for future enhancements such as live voice input, expanded language support, graphical user interfaces, and real-time subtitle integration. This work demonstrates the practical application of AI and natural language processing technologies to overcome language barriers and facilitate seamless multilingual communication.

**Introduction**

In today’s globalized world, communication across language barriers is increasingly common in education, business, and healthcare. Real-time translation during video calls can significantly enhance collaboration and understanding. This project demonstrates a real-time translator that processes audio input, translates it to another language, and generates audio output. The prototype focuses on English and Telugu, providing a foundation for a multilingual communication assistant.

**Literature Review**

Numerous studies and products have explored real-time translation. Google Translate and Microsoft Translator offer robust APIs for text and speech translation. Research in speech recognition and machine translation has led to significant improvements in accuracy and speed (Hinton et al., 2012; Wu et al., 2016). However, most existing solutions are either commercial or limited in language support and integration flexibility. Open-source libraries such as SpeechRecognition, googletrans, and gTTS enable custom implementations but often require careful orchestration and error handling. This project leverages these libraries to create a flexible, extensible translation system, focusing on user-friendliness and reliability.

**Methodology**

1. **Audio Input**: Accepts pre-recorded audio files (future versions may use live microphone input).
2. **Audio Conversion**: Converts MP3 to WAV for compatibility with speech recognition modules.
3. **Speech Recognition**: Uses Google’s speech-to-text API to transcribe spoken words.
4. **Language Detection**: Attempts recognition in Telugu first, then English, to auto-detect the spoken language.
5. **Translation**: Translates recognized text using Google Translate API.
6. **Text-to-Speech**: Converts translated text back to speech using gTTS.
7. **Output**: Saves the translated speech as an audio file.

**Technology Involved**

* **Python**: Main programming language.
* **Google Colab**: For code execution and demonstration.
* **SpeechRecognition**: For speech-to-text conversion.
* **pydub**: For audio format conversion.
* **googletrans**: For translating text.
* **gTTS**: For converting text to speech.
* **sounddevice, scipy**: For future real-time audio capture.

**Block Diagram**

+------------------+ +-------------------+ +-----------------+ +------------------+ +-------------------+  
| Audio Input | ---> | Audio Conversion | ---> | Speech | ---> | Translation | ---> | Text-to-Speech |  
| (MP3/WAV) | | (MP3 to WAV) | | Recognition | | (googletrans) | | (gTTS) |  
+------------------+ +-------------------+ +-----------------+ +------------------+ +-------------------+  
 |  
 v  
 +-------------------------+  
 | Output Audio (MP3/WAV) |  
 +-------------------------+

**Complete Python Backend Code**

# Install necessary libraries (uncomment if running in a new environment)  
# !pip install SpeechRecognition pydub googletrans==4.0.0-rc1 gTTS  
  
import os  
import speech\_recognition as sr  
from pydub import AudioSegment  
from googletrans import Translator  
from gtts import gTTS  
  
def convert\_mp3\_to\_wav(mp3\_path, wav\_path):  
 audio = AudioSegment.from\_mp3(mp3\_path)  
 audio.export(wav\_path, format='wav')  
  
def recognize\_speech(wav\_path):  
 recognizer = sr.Recognizer()  
 with sr.AudioFile(wav\_path) as source:  
 audio\_data = recognizer.record(source)  
 try:  
 # Try Telugu first  
 text = recognizer.recognize\_google(audio\_data, language='te-IN')  
 src\_lang = 'te'  
 dest\_lang = 'en'  
 except sr.UnknownValueError:  
 # Fallback to English  
 text = recognizer.recognize\_google(audio\_data, language='en-IN')  
 src\_lang = 'en'  
 dest\_lang = 'te'  
 return text, src\_lang, dest\_lang  
  
def translate\_text(text, src\_lang, dest\_lang):  
 translator = Translator()  
 translated = translator.translate(text, src=src\_lang, dest=dest\_lang)  
 return translated.text  
  
def text\_to\_speech(text, lang, output\_path):  
 tts = gTTS(text, lang=lang)  
 tts.save(output\_path)  
  
def main(mp3\_input, output\_audio):  
 wav\_path = "temp.wav"  
 convert\_mp3\_to\_wav(mp3\_input, wav\_path)  
 try:  
 original\_text, src\_lang, dest\_lang = recognize\_speech(wav\_path)  
 print(f"Recognized ({src\_lang}):", original\_text)  
 translated\_text = translate\_text(original\_text, src\_lang, dest\_lang)  
 print(f"Translated ({dest\_lang}):", translated\_text)  
 text\_to\_speech(translated\_text, dest\_lang, output\_audio)  
 print(f"Output audio saved as {output\_audio}")  
 except Exception as e:  
 print("Error:", e)  
 finally:  
 if os.path.exists(wav\_path):  
 os.remove(wav\_path)  
  
# Example usage:  
# main("input.mp3", "translated.mp3")

**Results and Discussion**

**Example 1: Telugu to English**

* **Input Audio**: (User says in Telugu) "నమస్తే మీరు ఎలా ఉన్నారు"
* **Recognized Text**: "Namaste meeru ela unnaru"
* **Translation**: "Namaste how are you"
* **Output Audio**: English speech saying "Namaste how are you"

**Example 2: English to Telugu**

* **Input Audio**: "I'm doing well thank you, how about you?"
* **Recognized Text**: "I'm doing well thank you, how about you?"
* **Translation**: "నేను బాగున్నాను, ధన్యవాదాలు, మీరు ఎలా ఉన్నారు?"
* **Output Audio**: Telugu speech saying "నేను బాగున్నాను, ధన్యవాదాలు, మీరు ఎలా ఉన్నారు?"

**Screenshots of Output (Simulated)**

**Console Output:**

Recognized (te): Namaste meeru ela unnaru  
Translated (en): Namaste how are you  
Output audio saved as translated.mp3

**Conclusion**

This real-time language translation system demonstrates the integration of speech recognition, language detection, machine translation, and text-to-speech synthesis to facilitate seamless multilingual communication. By converting spoken input from one language into translated speech in another, the project addresses language barriers and enhances accessibility in various contexts. The use of Python libraries and cloud-based APIs enables a flexible and extensible framework, laying a strong foundation for future enhancements.

**Future Scope**

* **Live Voice Input**: Integrate real-time microphone input for live translation.
* **Subtitle Display**: Show real-time subtitles during video calls.
* **GUI Integration**: Develop a user-friendly interface using Streamlit or Tkinter.
* **More Languages**: Expand support to include Hindi, Spanish, Chinese, etc.
* **Direct Video Call Integration**: Embed translation into platforms like Zoom, Google Meet, or Skype.

**References**

1. Hinton, G., et al. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition. IEEE Signal Processing Magazine.
2. Wu, Y., et al. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.
3. [SpeechRecognition Documentation](https://pypi.org/project/SpeechRecognition/)
4. [googletrans Documentation](https://py-googletrans.readthedocs.io/en/latest/)
5. [gTTS Documentation](https://pypi.org/project/gTTS/)
6. [pydub Documentation](https://pydub.com/)

**Bibliography**

* Jurafsky, D., & Martin, J. H. (2019). Speech and Language Processing. Pearson.
* Manning, C. D., & Schütze, H. (1999). Foundations of Statistical Natural Language Processing. MIT Press.
* Online resources and documentation for Python libraries used in the project.

**Note:** For actual screenshots, please run the code and capture the outputs as per your environment. The provided image link is a placeholder for illustration.

If you need the document in a specific format (e.g., DOCX or PDF), let me know!

⁂