Gesprächsprotokoll: Evaluation von TTS-Systemen

1. Ziel des Projekts

- Kernziel: Ein geschriebenes Buch (Fließtext) vollautomatisch und kostenlos über eine lokale, NVIDIA-gestützte KI-Infrastruktur in ein emotionales Mehrpersonen-Hörbuch (mit sauberen Sprecherwechseln) umzuwandeln.
- Gewünschte Dynamik: Präzise Steuerung extremer Emotionen wie Schreien, Brüllen, Flüstern, Angst und Freude.

2. Phase 1: Google Cloud TTS API (Analyse)

- Parameter: Die API steuert Stimmen über VoiceSelectionParams (Sprache, Name, Geschlecht) und AudioConfig (Sprechgeschwindigkeit, Tonhöhe, Lautstärke, Format).
- SSML-Tags: Erlauben feine Pausen () und Aussprachekorrekturen, ignorieren bei modernen Stimmen (Studio) aber komplexe emotionale Dynamiken.
- Chirp 3: HD Stimmen: Googles neueste generative Generation klingt extrem menschlich, ignoriert aber fast alle klassischen SSML-Tags und physikalischen API-Parameter. Emotionen und Pausen werden hier organisch über In-Line-Textelemente (z. B. [excited], [pause short]) gesteuert.
- Hürde: Es gibt für Chirp-Stimmen kein kostenloses monatliches Kontingent.

3. Phase 2: Umstieg auf die lokale AllTalk-KI (NVIDIA)

- Infrastruktur: AllTalk läuft erfolgreich lokal auf Ihrer NVIDIA-Grafikkarte (Nutzung der Coqui XTTS-v2 Architektur).
- Parameter: Steuerung über API-Parameter wie speed, pitch, temperature und repetition_penalty.
- Emotions-Steuerung: AllTalk besitzt keine mathematischen 'Gefühls-Regler'. Emotionen werden rein kontextbasiert über die Textgestaltung (Ausrufezeichen, Großbuchstaben, Auslassungspunkte) getriggert oder indem man der API ein emotional passendes Klon-Audio als Vorlage übergibt.
- Hürde: Es fehlen frei nutzbare, saubere deutsche Stimmenvorlagen (z. B. im ESD-Format), die diese extremen Emotionen als Klon-Basis mitbringen. Der Versuch, diese manuell aus Forschungs-Datenbanken herunterzuladen, erwies sich als zu komplex und unpraktikabel.

4. Phase 3: Die technologische Sackgasse

1. Das System-Dilemma: Es gibt am Markt kein TTS-System, das absolute Parameter-Kontrolle im Code (schreien=80%) mit dem organischen, extremen Klang von echtem Brüllen oder Weinen perfekt vereint.
2. Das KI-Kürzungs-Problem: Frei zugängliche KI-Chatfenster (wie Gemini) neigen beim Umformatieren von Text in ein sprecherbasiertes Skript-Format dazu, den Originaltext eigenmächtig zu kürzen oder zusammenzufassen.

3. Das Zeichenlimit: Längere Buchabschnitte sprengen die Kontextfenster normaler Chatfenster, was zu Fehlern bei der Figurenerkennung und Sprecherzuordnung führt.

5. Erarbeitete Lösungsansätze (Der 'Verteiler'-Weg)

- Schritt 1 (Text-Vorbereitung): Ein großes Sprachmodell (wie Gemini Pro mit 2 Mio. Token Kontext) liest das Buch ungekürzt und fügt vor jeden Absatz eine Sprecher- und Emotionskennung ein, z. B. [MARKUS - WÜTEND].
- Schritt 2 (Code-Verteiler): Ein von uns erstelltes Python-Skript (manuskript_verteiler.py) liest diese Datei Zeile für Zeile aus, entfernt die Klammern, wählt die passende .wav-Stimme und manipuliert den Text passend für die lokale AllTalk-API (z. B. automatische Transformation in GROSSBUCHSTABEN bei Wut), um die Emotionen auf der Hardware zu erzwingen.

6. Aktueller Status

Das Projekt wird aktuell als gescheitert/eingestellt betrachtet. Der Aufwand, die Textaufbereitung fehlerfrei zu automatisieren (ohne dass die KI den Text kürzt oder Figuren falsch zuordnet), steht in keinem Verhältnis zum Ergebnis, da lokale deutsche TTS-Engines die gewünschten extremen Emotionen (Schreien, Brüllen) rein über Textbefehle schauspielerisch nicht verlässlich abbilden können.