**Ex.No: Date:**

**Exploration of Prompting Techniques for AI Video Generation**

**Aim:**

The aim of this experiment is to explore how different prompting techniques influence AI-generated video content. By leveraging models like **RunwayML, Pika Labs, Synthesia, and Sora (if available)**, we will analyze how variations in text prompts affect video quality, style, and relevance. This study will help optimize prompts for applications such as marketing, storytelling, and educational content.

**Procedure:**

**1. Define Video Generation Goals**

* Identify use cases (e.g., short films, ads, explainer videos, animations).
* Select AI tools (RunwayML for motion, Pika Labs for stylized clips, Synthesia for avatars).

**2. Experiment with Prompting Techniques**

Test different prompt structures:

* **Descriptive Prompts:**
  + *"A futuristic city at sunset with flying cars, cinematic 4K."*
* **Action-Based Prompts:**
  + *"A robot dances in a neon-lit nightclub, cyberpunk style."*
* **Style Transfer Prompts:**
  + *"Make this look like a 1980s VHS recording of a beach party."*
* **Emotion-Driven Prompts:**
  + *"A suspenseful scene where a detective slowly opens a mysterious door."*

**3. Develop a Video Generation Script**

* Use Python to interact with AI video APIs (e.g., RunwayML’s Gen-2).
* Compare outputs from different models.

**4. Evaluate and Optimize**

* Assess video coherence, adherence to prompts, and aesthetic quality.
* Refine prompts iteratively for better results.

**5. Deploy (Optional)**

* Build a **Streamlit/Gradio app** for users to generate videos interactively.

**Program (Python Code for RunwayML API):**

Python code

import os

import requests

import logging

from dotenv import load\_dotenv

***# Load environment variables (API keys)***

load\_dotenv()

***# Configure logging***

logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

logger = logging.getLogger(\_\_name\_\_)

***# API Keys (Replace in .env file)***

RUNWAYML\_API\_KEY = os.getenv("RUNWAYML\_API\_KEY")

def generate\_video(prompt, model="runwayml/stable-diffusion-v1-5", steps=30):

**"""Generate video using RunwayML's API."""**

try:

headers = {"Authorization": f"Bearer {RUNWAYML\_API\_KEY}"}

payload = {

"prompt": prompt,

"model": model,

"steps": steps

}

response = requests.post(

"https://api.runwayml.com/v1/video/generate",

headers=headers,

json=payload

)

if response.status\_code == 200:

video\_url = response.json().get("output\_url")

return f"Video generated: {video\_url}"

else:

logger.error(f"RunwayML Error: {response.text}")

return "Failed to generate video."

except Exception as e:

logger.error(f"API Error: {e}")

return "Video generation failed."

def main():

print("=== AI Video Generation Explorer ===")

print("Enter a prompt for video generation (e.g., 'A spaceship landing on Mars').")

print("Type 'quit' to exit.\n")

while True:

prompt = input("\nYour Video Prompt: ").strip()

if prompt.lower() in ["quit", "exit"]:

print("Exiting...")

break

if not prompt:

print("Please enter a valid prompt.")

continue

print("\nGenerating video... (This may take a few minutes)")

result = generate\_video(prompt)

print(result)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output Examples:**

**1. Cinematic Scene Prompt:**

**Input:**  
*"A lone astronaut walks on Mars at sunset, 4K cinematic."*  
**Output (RunwayML):**

* A high-resolution clip of a realistic Martian landscape with a slow-moving astronaut.

**2. Animated Style Prompt:**

**Input:**  
*"A cartoon cat playing guitar in a jazz club, Pixar style."*  
**Output (Pika Labs):**

* A stylized 3D animation with vibrant lighting.

**3. Retro Effect Prompt:**

**Input:**  
*"A 1970s disco party with grainy film effects."*  
**Output (Sora-like model):**

* A vintage-style clip with flickering lights and analog noise.

**Result:**

**Successful video generation** using AI models with distinct styles.  
**Key Findings:**

* Detailed prompts yield **higher-quality outputs**.
* Style modifiers (e.g., *"cinematic," "Pixar-style"*) significantly alter results.
* Some models struggle with **complex motion** (e.g., running animals).