“AniGEN”
Natural Language Driven AI Avatar Motion with Video-Based Motion Data

Presented by Marzukh Akib Asjad, Mikael Ho Yu Lau & James Olano
Agenda

01  Background
02  Objectives
03  Methodology
04  Preliminary Results
05  Project Schedule
06  Conclusion
Background

Grand Theft Auto VI (GTA 6)
by Rockstar Studio

[1] Estimated production cost:
USD 6 Billion
Background

[2] Film production cost

Pirates of the Caribbean: On Stranger Tides
USD 378.5 million

Avengers: Age of Ultron
USD 365 million
Background

[3] Motion capturing technology

Advanced equipment & technical knowledge required

Big Companies
Background

Independent Artists

[4] Rotomation

Issues: Time-consuming
Background

Independent Artists

[5] Deepmotion - Video motion tracking

Issues: Inconsistent and jittering motion
Motivation

Text-to-video generation Model

Text Prompt → Text-to-video Generation → Non-interactive Video output
Motivation

Text-to-video generation Model

Text Prompt → Text-to-video Generation → Non-interactive Video output

[6] runway
Motivation

Prompt: a man doing a backflip
Motivation

For many text-to-video models:

- Format of the output is a non-interactive video
- Inconsistent frames and distorted figures
- Long generation time
Motivation

For many text-to-video models:
- Format of the output is a non-interactive video
- Inconsistent frames and distorted figures
- Long generation time

Changing up text-to-video models:
- 3D avatar movement in 3D scene
- Output will be in interactive format
- Does not require animation skills
- Consistent and production ready
Objectives

- Used for Game Development and High Quality 3D Videos
- Consistent output, no figure distortion, high quality
- Output can be modified, editable, interactive
- Easy to use web based interface
03. Methodology

03.1 Workflow

03.2 Tech stack
The Input-Process-Output workflow of AniGEN
The Input-Process-Output workflow of AniGEN
The Input-Process-Output workflow of AniGEN

Input
- e.g. "Run and do a backflip"

Relevant motion data recognised using NLP and queried

More than one motion?
The Input-Process-Output workflow of AniGEN

1. **Input**: e.g., "Run and do a backflip"
2. **Processing**:
   - Relevant motion data recognised using NLP and queried
   - More than one motion?
   - (NO branch)
   - Rendering the motion(s)

This diagram illustrates the workflow where user input is processed to generate relevant motion data, which is then rendered.
The Input-Process-Output workflow of AniGEN
The Input-Process-Output workflow of AniGEN
# Tech Stack

<table>
<thead>
<tr>
<th>Front end</th>
<th>Motion Blending</th>
<th>Back end</th>
</tr>
</thead>
<tbody>
<tr>
<td>React</td>
<td>Blender (Software for Motion and Rendering)</td>
<td>Python (Back-end code)</td>
</tr>
<tr>
<td>React (JavaScript Library)</td>
<td></td>
<td>FireBase (Database)</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>Mixamo (Animation Data)</td>
<td>FireBase</td>
</tr>
<tr>
<td>Bootstrap (Front-end CSS Framework)</td>
<td></td>
<td>FireBase</td>
</tr>
<tr>
<td>Tailwind CSS</td>
<td>Python (Scripting Language for Blender)</td>
<td>Gemini (Natural Language Model)</td>
</tr>
<tr>
<td>Tailwind CSS (CSS Framework for simplifying styling)</td>
<td></td>
<td>Gemini</td>
</tr>
</tbody>
</table>
04. Preliminary Results

04.1 Front End
04.2 Motion Blending
04.3 Back End
04.4 Difficulties
Front-end Preliminary Results

Simple and intuitive ChatGPT-like Interface

Initial Figma Designs
Front-end Preliminary Results

Planned Layout
• Mainpage
Front-end Preliminary Results

Planned Layout
- Mainpage
Front-end Preliminary Results

Planned Layout
- Mainpage

Playback slider
Front-end Preliminary Results

Planned Layout
• Mainpage

Animation Display
Front-end Preliminary Results

Planned Layout
• Mainpage

Side navigation bar
Front-end Preliminary Results

Planned Layout
- Avatar pop up
Front-end Preliminary Results

Implemented
  • Mainpage

To work on:
  • Backend integration
  • Side-navigation buttons
  • Styling
Front-end Preliminary Results

Implemented
- Avatar popup

To work on:
- Multiple models
- Styling
Front-end Preliminary Results

Implemented
- History modal

To work on:
- Prompt-to-backend
- Prompt/History Persistence
- Viewing of previous animations
Motion Blending Preliminary Results

Implementing
- Motion Blending of In-Place Animation

For example, Creating a transition of walk to run
Motion Blending Preliminary Results

Implementing
  • Motion Blending of In-Place Animation

Step 1: Put Motion 1 Clip in Blender
Motion Blending
Preliminary Results

Implementing
• Motion Blending of In-Place Animation

Step 2: Put Motion 2 Clip in next track above
Motion Blending Preliminary Results

Implementing
- Motion Blending of In-Place Animation

Step 3: Blend using Non Linear Animations
Motion Blending
Preliminary Results

Implementing
• Motion Blending of In-Place Animation

Final Output:
Motion Blending Preliminary Results

Implementing
- Motion Blending of In-Place Animation

```python
def main():
    """
    The main function that orchestrates the execution of the script
    Calls other functions to select the armature, retrieve the uppermost NLA track,
    get the last frame, switch to the NLA Editor, and push down the action to the NLA tracks
    """
    armature_obj = select_armature("idle")
    if armature_obj is not None:
        uppermost_track = get_uppermost_nla_track(armature_obj)
        last_frame = get_last_frame(uppermost_track)
        switch_to_nla_editor()
        action_name = "Jump Over"
        action = bpy.data.actions.get(action_name)
        if action is not None:
            print("Action found:", action.name)
        else:
            print("Action not found:", action_name)
```
Motion Blending Preliminary Results

Future Scope of work

- Not restricting to in place animations only
- Simple environment generations
- Rendering Video as well as returning FBX File
### Back-end Preliminary Results

**Implemented**
- File searching script in Python

**Purpose:**

After NLP processes user text prompt into array of motion data labels, this script matches labels with corresponding motion data.
Back-end Preliminary Results

Implemented

- File searching script in Python

User enters desired file name as sole argument.
Error thrown if zero or more than one argument entered.

```
(base) outrider@Outrider Task23 % python3 find-files-1.py
AssertionError thrown: Incorrect number of arguments!

Usage notes:
python3 find-files-1.py <file name>

Example:
python3 find-files-1.py filename.fileextension
```
Back-end Preliminary Results

Implemented
- File searching script in Python

```
35 # Print the contents of the file.
36 # If the file does not exist, a FileNotFoundError will be thrown.
37 try:
38     with open(filename,'r') as file_in:
39         print(file_in.read(), end="
")
41 except FileNotFoundError:
42     print("FileNotFoundError thrown: The file specified does not exist!")
43     exit()
```

Prints the contents of the file, if file is found.
Error thrown otherwise.

(base) outrider@Outrider Task23 % python3 find-files-1.py endymion.txt
FileNotFoundError thrown: The file specified does not exist!
Back-end Preliminary Results

Implemented
- File searching script in Python

```
47    # Return the file path.
48    filepath = THIS_DIR + "/" + filename
49    print(filepath)
```
Back-end Preliminary Results

Implemented
- File searching script in Python

Example input and results:

(base) outrider@Outrider Task23 % python3 find-files-1.py aurora.txt
She then pointed at another constellation in her book. "Now, steer a bit clockwise, and go with the same distance." Drawing a line on the page with her finger, she continued, "you shall see a constellation, which was said to resemble two chevrons arranged side by side. Kassiopeia is what astronomers from the land of Kreikka call it, and its chevrons points away from the Pohjantähti."

/Users/outrider/Desktop/FILES/hku/comp/4801/work/Task23/aurora.txt
Back-end Preliminary Results

Implemented
• File searching script in Python

Works with all file extensions...

(base) outrider@Outrider Task23 % python3 find-files-1.py aurora.whateverfileformat

Putting the plates on the table, she introduced with fondness, "This is what commoners usually eat in Kultajärvi. Potatoes, lingonberries, smoked salmon and reindeer meat. We are a country of berry pickers, reindeer herders and salmon fishers, when it comes to food." /Users/outrider/Desktop/FILES/hku/comp/4801/work/Task23/aurora.whateverfileformat
Back-end Preliminary Results

Implemented
- File searching script in Python

Works with files in other directories...

(base) outrider@Outrider Task23 % python3 find-files-1.py ../Task24/kahdestoista.tammikuuta

"This little islet is named Tohmonsaari. To its east, is the Karhunniemi peninsula. The upstream part of Kemijoki begins from this place, and splits into three tributaries as we navigate further north. One leads to Port tipahta, another leads to Lokka, and the last one leads to Kemihaara. These remote upstream places are mostly populated by woodcutters, fishers, craftsmen and full-time soldiers, who guard the borders during times of peace."

Back-end Preliminary Results

Implemented
• File searching script in Python

Works with files with non-English content...

(base) outrider@Outrider Task23 % python3 find-files-1.py chinesetext.whateverfileformat

我仔細地看著她的動作和地圖，意識到這些河川的源頭相距多麼遙遠。庫
塔耶爾維王國確實幅員遼闊，擁有廣闊的森林、河流和湖泊！
然後奧羅拉（Aurora）從羅瓦涅米（Rovaniemi）向大海畫了一條線。“這是庫爾塔
約基海港，我們王國與其他國家的大部分貿易都在這裡進行。”然後她指著地圖上
的許多國家，“我們與該地區的其他國家進行貿易，例如卡亞尼、哈姆、薩沃和雷
爾。”

我盡力記住她提到的國家的名字。所有這些國家聽起來都像是童話般的地方，我想
像著這些王國之間進行貿易的樣子。我的腦海中立即浮现出船隻在海港裝卸貨物的
場景，儘管我從未親眼見過，無論是在月球上還是在地球上。

我們仔細地閱讀了地圖。然後我打破了沉默，問奧羅拉：“我想知道，從庫爾塔約
基海港，一直到源頭，需要多長時間？”

歐若拉道：「具體數字我不記得了，不過我记得看過一本本國公民寫的書，他說從
庫爾塔約基海港到西北很遠的穆特卡耶爾維，大約需要半天的時間。...他起航的時
候，正是日出，他的船停泊的時候，正是日落時分。"/Users/outrider/Desktop/FI
LES/hku/comp/4801/work/Task23/chinesetext.whateverfileformat
Difficulties Encountered

- Keeping up with Project Schedule
Difficulties Encountered

- Difficulty in using Mocap Data
- Keeping up with Project Schedule
- [Blank]
- [Blank]
- [Blank]
Difficulties Encountered

- Difficulty in using Mocap Data
- Keeping up with Project Schedule
Difficulties Encountered

- Difficulty in using Mocap Data
- Keeping up with Project Schedule
- React Library implementations
Difficulties Encountered

- Difficulty in using Mocap Data
- Keeping up with Project Schedule
- React Library implementations
- Python tool initially used for transitions
5. Project Schedule

01. SEMESTER 1 - Design and Data
02. NOV - JAN - Front-End
03. DEC - FEB - Motion Blending
04. FEB - APRIL - NLP
Gantt chart showing the distribution of the schedule for the tracks to be worked on throughout the academic year.
Gantt chart showing the distribution of the schedule for the tracks to be worked on throughout the academic year.
06. Conclusion

06.1 Upcoming Plans
06.2 Conclusion
Plans for the upcoming semester

- Incorporation of OpenAI’s GPT model
- Refactoring of back-end code to facilitate motion blending
- Smooth integration between front-end, motion blending and back-end
Conclusion

01 A new solution turning text into avatar animation
Aimed for bridging the skill gap and make 3D avatar animation more accessible for studios and creators.

02 Powered by React, Python, GPT-3.5 and other libraries
Transforming a text prompt into an interactive avatar animation video with the power of AI and NLP.

03 Project completion expected in April 2024
Upcoming tasks: NLP and Back-end setup
AniGEN beyond 2024

Uploading custom 3D models

Generating Dance Videos alongside music

Scaling database to online videos for motion data to include trending videos (gathering motion data from online videos)
AniGEN beyond 2024
Reference List


