AnimeGamer: Infinite Anime Life Simulation with Next Game State Prediction

Supplementary Material

Overview

In this appendix,we present the following:

- Details of our AnimeGamer in Section 1.
- Dataset Construction Pipeline in Section 2.
- Implementation details of AnimeGamer and other baselines in Section 3.
- Details of the evaluation benchmark in Section 4.
- Human Evaluation in Section 5.
- Additional visualization results in Section 6.

1. Details of AnimeGamer

Special Tokens in MLLM. We add special token to the tokenizer of our MLLM to generate game states and formulate the output. Specifically, we use <MS></MS> to represent the start and end of motion scope, <ST></ST> for stinema value, <SC></SC> for social value, <ET></ET> for entertainment value. To continuous generate the action-aware multimodal representations, We add 226 learnable query tokens <ANIME i> to stimulate continuous generation, and <VS></VS> to represent start and end of the animation shot representation.

Motion Scope. We employ Memflow [3] to compute the optical flow transformation for each frame within the video. Subsequently, we convert them into absolute values to represent the motion scope. A filtering threshold of r=0.2 is adopted to filter out the background information. After that, we calculate the average value of the remaining part to denote the motion scope of an animation shot. Next, we divide the range into five levels, which serve as discrete targets for the MLLM to fit.

2. Dataset Construction Details

Video Pre-processing. Taking an anime film as an example, we first download the film and crop its borders. Then, we resize it to the corresponding size and divide it into several segments using a scene - detection model [1]. Next, each segment is split according to a fixed time period (2 seconds). In this way, we obtain the video training data arranged in timestamp order. In addition, we download the reference images of each protagonist to locate the characters in each video segment.

Captioning. We utilize Intern-VL-26B [2] to generate captions for each animation shot. We input the protagonist reference images and eight evenly-sampled frames from an anime clip as visual input. To match the character, we employ the following prompt:

> Image-1: <image>Image-2: <image>According to the characters' index in Image-1, your task is to answer how many characters are in Image-2 (4 frames from a anime video) and what their indices are. Response format: '[Num]<number of characters>[ID]<index of the character>'; Response example: '[Number]2[Index]1,3'. If no characters are detected, please respond with '[Number]0[Index]0'. Your response:

To acquire descriptions of motion, the environment, and character states, we utilize the following prompt:

> Image-1: <image>Image-1 is from an anime clip, Your task is to extract a structured description based on the information. First, I will give you the movement level <ML>respectively. The movement level is categorized into five levels: Level 1: very small movement amplitude, almost imperceptible; Level 2: small movement amplitude, slight swaying or adjustments; Level 3: moderate movement amplitude, appropriate movement or adjustments; Level 4: large movement amplitude, noticeable and significant; Level 5: very large movement amplitude, extremely obvious and intense. Next, you need to generate the following information: (1) subject <S>, motion description <MD>and environment <EV>. Please use a single word for subject and background, use a simple phrase for motion in the present simple tense. (2) Movement adverb <MA>: Based on <ML>, give <MD>a fitting adverb. (3) Social interaction <SC>: If there are two or more characters interacting socially in the scene, such as talking, hugging, walk together or kissing, use 1; otherwise, use 0 to indicate no social action. (4) Entertainment <ET>: If the protagonist is engaged in entertainment activities, sports or relaxing, such as reading, riding, flying, swimming, whispering, archery... use 1; otherwise, use 0 to indicate no entertainment action. (5) Stamina <ST>: Stamina can be restored through actions like eating, drinking, lying, sleeping, hugging, treatment... If the <MD>are restoring stamina, fill in 1; otherwise, use -1. Example output: <S>Girl</S><MD>run</MD><EV>Forest </EV><MA>slowly</MA><SC>0</SC><ET> 1</ET><ST>-1</ST>. Your response:

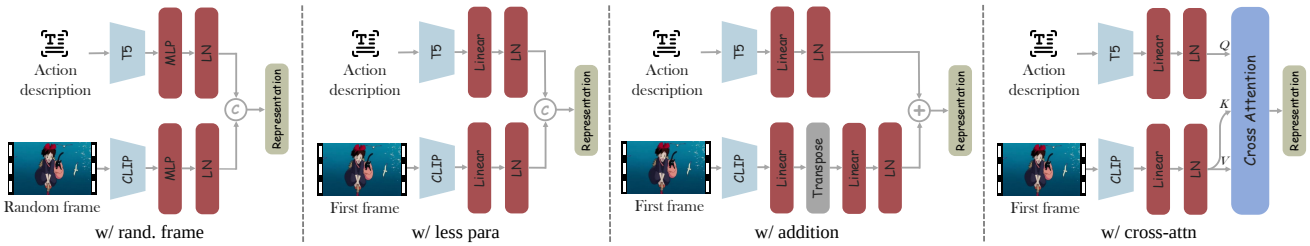

Figure 1. Four variants of our animation shot encoder. 1) We use random frame instead of first frame to obatain the action-aware multimodal representation, denoted as "w/ rand. frame"; 2) We replace the MLP module in $\mathcal{E}_a$ with a single Linear layer to reduce learnable parameters, denoted as "w/ less para"; 3) We combine f_v with f_{md} via element-wise addition, denoted as "w/ addition"; 4) We combine f_v with f_{md} via cross-attention, denoted as "w/ cross-attn".

3. Implementation Details

In this section, we present the implementation details of our AnimeGamer in Section 3.1, baseline methods in Section 3.2 and the ablation studies in Section 3.3.

3.1. AnimeGamer

Animation Shot Encoding and Decoding. In this phase, we initialize the parameters of our animation shot decoder using CogvideoX-2B[1]. We apply LoRA to the 3D-Attention with a rank of 64. The learning rate is set to 2e-4. The training is conducted on 24 Nvidia A100 GPUs. Initially, we train $\mathcal{E}_a$ for 10,000 steps as a warm-up, followed by joint training of $\mathcal{E}_a$ and $\mathcal{D}_a$ for an additional 80,000 steps.

Next Game State Prediction. For the MLLM, we initialize our model with the weight of Mistral-7B and train it using LoRA, facilitated by the peft library. The LoRA rank is set to 32, with lora-alpha also set to 32. The learning rate is 5e-5, and the training is carried out on 24 Nvidia A100 GPUs for 15,000 steps.

Decoder Adaptation. In this stage, we fine-tune only $\mathcal{D}_a$. The learning rate is 5e-5, and the training is executed on 24 Nvidia A100 GPUs for 10,000 steps.

3.2. Baselines

GSC. We utilize StoryDiffusion [5] based on SDXL [4], where the instructions for a 10-round game are input simultaneously to generate the corresponding images. Then, we use the Cogvideox-5B-I2V[2] model to convert these images into animation shots. During this process, action instructions are provided as prompts to the pretrained I2V model.

GFC. We fine-tune the T2I model FlUX[3] using LoRA. For training, we pair the first frame of each animation shot with its corresponding instruction to form image-text pairs. We employ LoRA with a rank of 32 and train on 8 Nvidia A100 GPUs for 200,000 steps. During testing, we convert images to video using the same method as in GSC.

GC. We fine-tune the CogvideoX-2B model using LoRA, employing the same configuration as used for training $\mathcal{D}_a$.

3.3. Ablation Study

In the ablation study, we randomly selected a movie "Kiki's Delivery Service" from the training dataset as the training data. We split approximately 2,000 training samples into a training set and a test set with an 8:2 ratio. The ablation on animation shot tokenization and de-tokenization does not incorporate the MLLM, in order to focus on the reconstruction ability of $\mathcal{E}_a$ and $\mathcal{D}_a$ for animation shots.

Ablation on Animation Shot Tokenization and De-tokenization. We construct four variants for $\mathcal{E}_a$, as shown in Figure 1. 1) We use a random frame instead of the first frame as f_v, denoted as "w/ rand. frame". 2) We replace the MLP with a simpler Linear layer to align features, denoted as "w/ less para". 3) We use element-wise addition to unify f_v and f_{md}, denoted as "w/ addition". 4) We use cross-attention to unify f_v and f_{md}, denoted as "w/ cross-attn".

Ablation on Next Game State Prediction. In this ablation study, we incorporate the Cosine Loss into the training process. The overall training loss is a combination given by:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{MSE}} + \beta\mathcal{L}_{\text{cos}}, \tag{1}$$

where the hyperparameters α and β are set to 0.5.

4. Evaluation Benchmark Construction

MLLM as benchmark constructor. We use the following prompt for GPT-4o to generate our evaluation benchmark.

> You are a world model for an anime life simulation. You can generate stories of the character living in the world.

[1] THUDM/CogVideoX-2b
[2] THUDM/CogVideoX-5b-I2V
[3] black-forest-labs/FLUX.1-dev

The stories should sound like a game and leave space for user interaction. Now, you need to generate a 10-panel story (simulation game) with [Character] as the main character. For each turn, you should generate the following components: 1) Characters <S>: The main character must appear in each panel, and 0-1 additional characters can be included as supporting characters, chosen from [Characters]. 2) Motion Description <MD>: Describe the main character's action with a simple phrase. 3) Environment <EV>: Describe the current environment with one word. 4) Main character's state: you need to generate the following information: (1) Motion Level <ML>: The movement level is categorized into number 1-5: 1: very small movement amplitude 2: small movement amplitude, slight swaying or adjustments 3: moderate movement amplitude, appropriate movement or adjustments 4: large movement amplitude, noticeable and significant 5: very large movement amplitude, extremely obvious and intense (2) Movement adverb <MA>: Based on <ML>, give <MD>a fitting adverb. (3) Social interaction <SC>: If there are two or more characters interacting socially in the scene, such as talking, hugging, walking together, or kissing, use 1; otherwise, use 0 to indicate no social action. (4) Entertainment <ET>: If the protagonist is engaged in entertainment activities, sports, or relaxing, such as reading, riding, flying, swimming, whispering, archery, use 1; otherwise, use 0 to indicate no entertainment action. (5) Stamina <ST>: Stamina can be restored through actions like eating, drinking, lying, sleeping, hugging, treatment... If the <MD>are restoring stamina, use 1; otherwise, use -1. For the entire story, here are some instructions you need to follow: 1) Ensure continuity between different panels as much as possible. Encourage different actions in the same scene or return to a previous scene in subsequent panels. 2) Keep it realistic and as close to a life simulation game scenario as possible. Please use common scenes and easily representable actions, and avoid including tiny, difficult-to-generate objects. 3) Output format: Each line represents one turn, using the following format: <S>Characters</S><MD>Motion Description</MD><EV>Environment</EV><ML>Motion Level</ML><MA>Movement adverb</MA><SC>Social interaction</SC><ET>Entertainment</ET><ST>Stamina</ST>

MLLM as a judge. We use the following prompt for GPT-4o to assess the output of the models.

Please act as an impartial judge and evaluate the quality of the generation story video contents provided by N AI agents. Here's some instructions you need to follow:
1) Story Composition: Each story consists of 5 scenes, and I will provide you with their respective prompts.
2) Evaluation: For each AI agent's output, I will present you with an image composed of 5 frames extracted from the videos. The image in the i-th row represent 5 frames extracted from the generated video corresponding to scene i.
3) Evaluation Criteria: You need to score each AI agent's output based on Overall Quality <OA>: The overall gaming experience. Text Alignment <TA>: The alignment between the prompt and the generated results. Contextual Coherence <ConC>: Whether the content of each scene can connect naturally, Character Consistency <ChaC>: Are the characters in each scene consistent with the provided reference characters? Emotional Consistency <EC>: The consistency between the expression of the scenes and the emotional statements in the prompt. Visual Coherence <VC>: Are the colors, styles, and compositions of the scenes consistent? The score range for these criteria is from 1 to 10, with higher scores indicating better overall performance.
4) Output Format: Your output should contain four lines, each starting with the evaluation criteria code such as <OA>, followed by N numbers representing the scores for each of the N agents, separated by spaces. Finally, provide a brief explanation of your evaluation on a new line.
5) Evaluation Requirements: Avoid any bias, ensure that the order of presentation does not affect your decision. Do not let the length of the response influence your evaluation. Do not favor certain agent names. Reflect the score differences as much as possible.

5. Human Evaluation

For the human evaluation, we recruit 20 participants who hold at least a bachelor's degree and have prior experience in image or video generation. A total of 9-round games with 50 samples are presented to the participants. We showcase the animation shots and character states generated by various models to the participants in the form of a PowerPoint presentation and ask them to fill out an Excel spreadsheet. They are required to rate the performance of different models for each metric in every game. Subsequently, we convert

the rankings into absolute scores: 10 points for the first-ranked model, 7 points for the second, 4 points for the third, and 1 point for the fourth. Finally, we calculate the average performance of each model.

6. Additional Qualitative results

We present the image of characters appeared in our paper in Figure 2. We present the infinite game generation results of AnimeGamer and other baselines in Figures 3 to 8. The game simulation videos are presented in our homepage: https://howe125.github.io/AnimeGamer.github.io/.

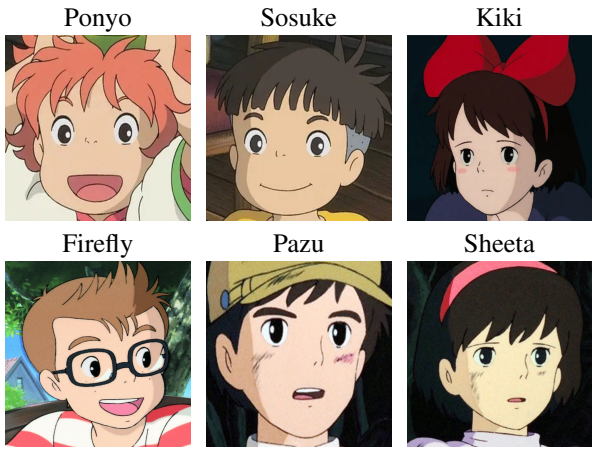

Figure 2. Image of characters in the paper.

References

[1] Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang-wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, et al. Panda-70m: Captioning 70m videos with multiple cross-modality teachers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13320–13331, 2024. 1

[2] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024. 1

[3] Qiaole Dong and Yanwei Fu. Memflow: Optical flow estimation and prediction with memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19068–19078, 2024. 1

[4] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 2

[5] Yupeng Zhou, Daquan Zhou, Ming-Ming Cheng, Jiashi Feng, and Qibin Hou. Storydiffusion: Consistent self-attention for long-range image and video generation. *Advances in Neural Information Processing Systems*, 37:110315–110340, 2024. 2

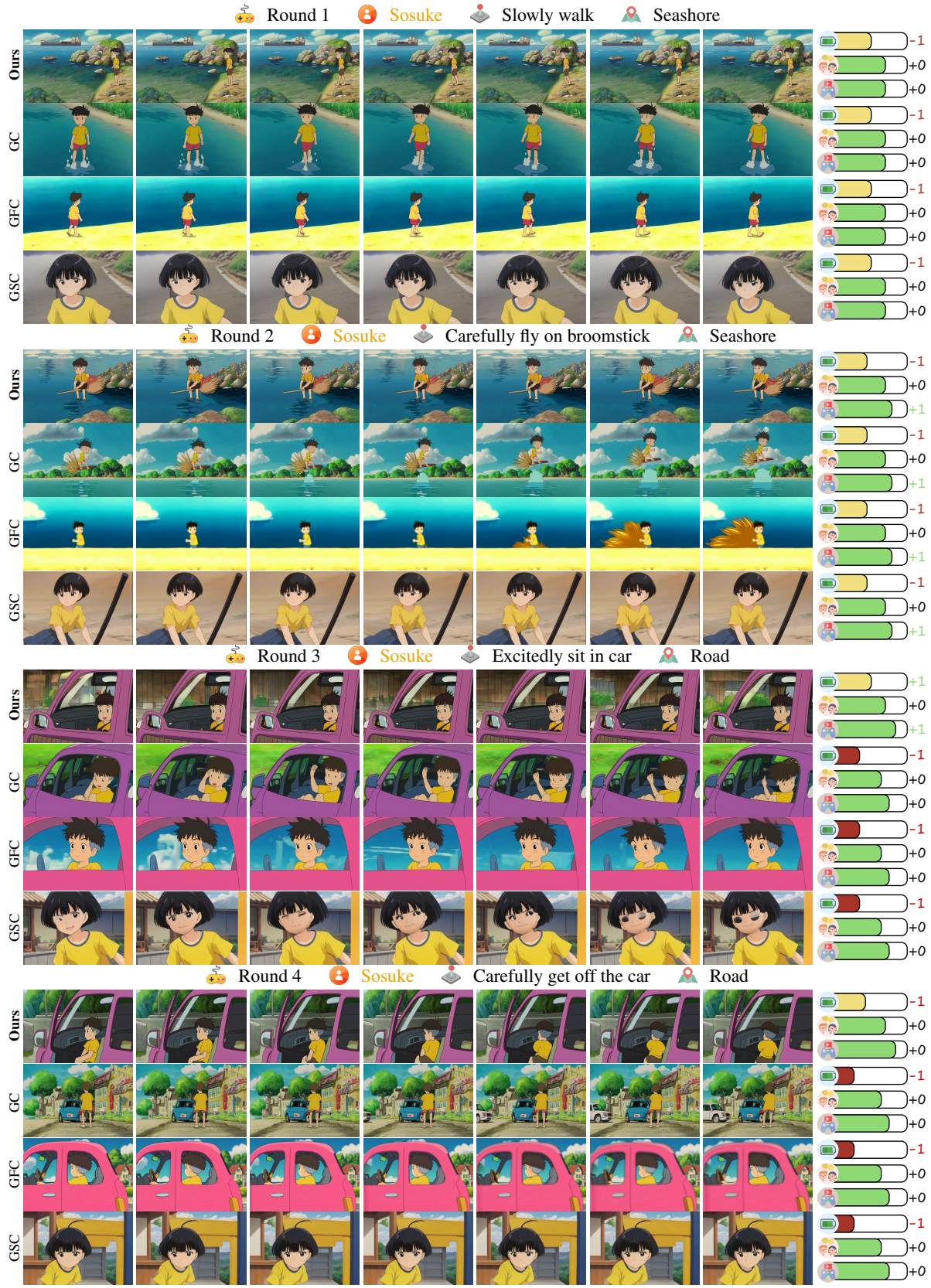

Figure 3. Visualization of Game 1.

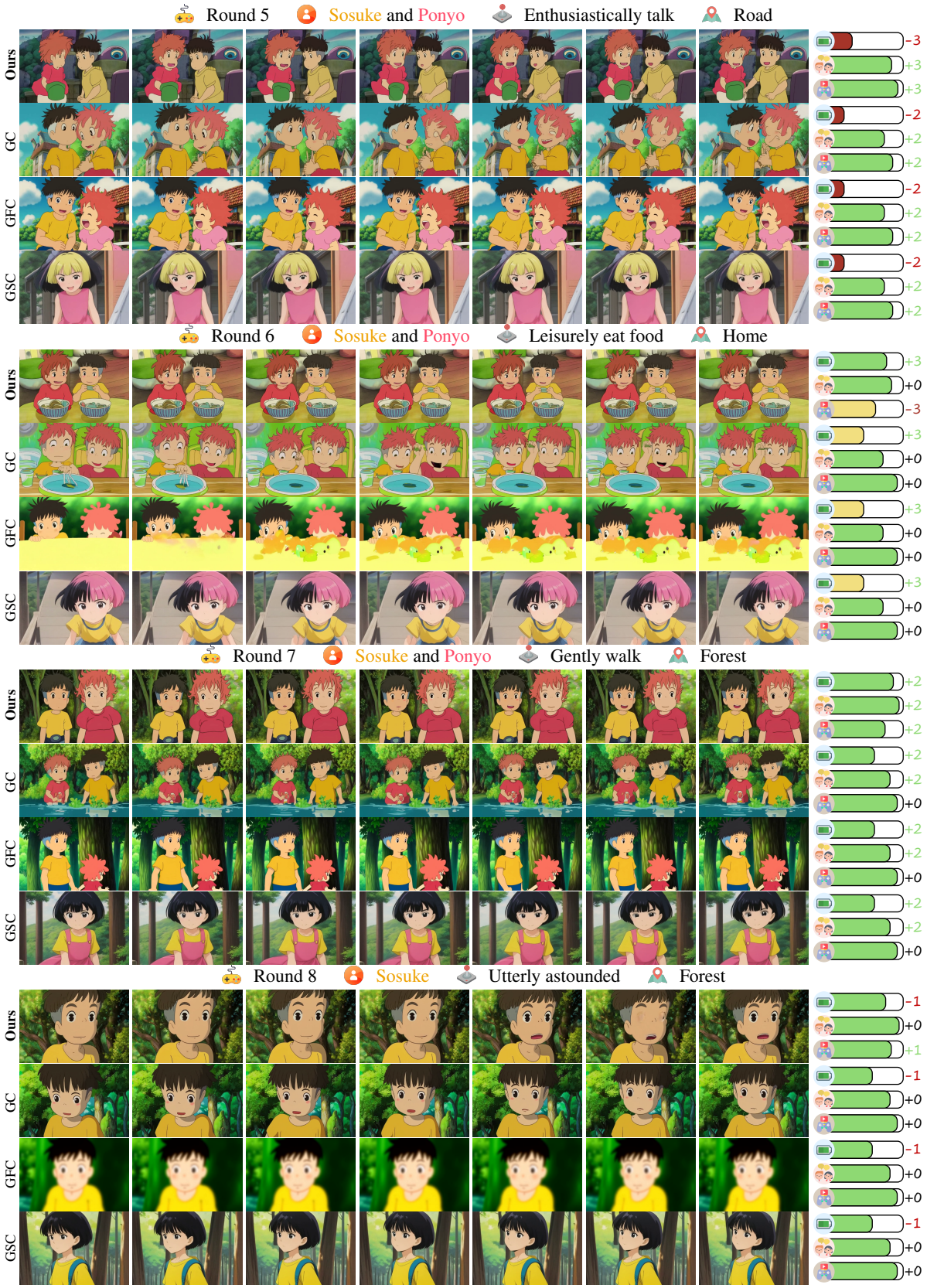

Figure 4. Continuation of the Visualization of Game 1.

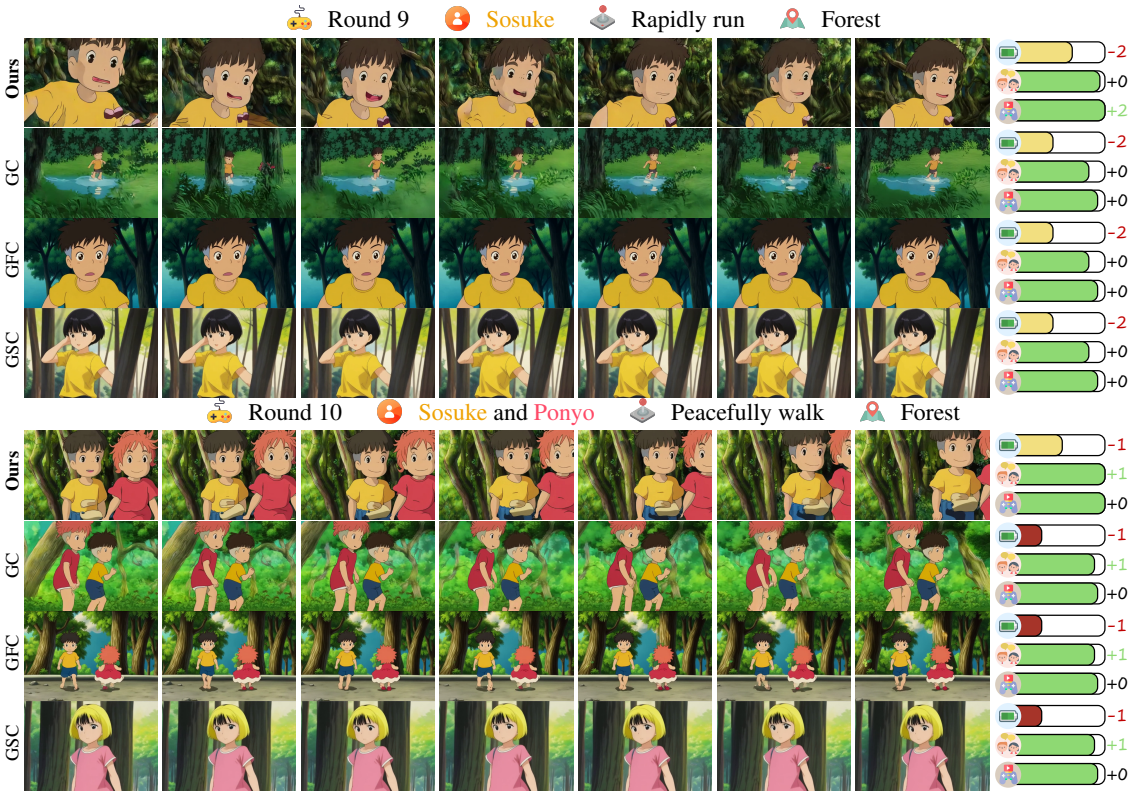

Figure 5. Continuation of the Visualization of Game 1.

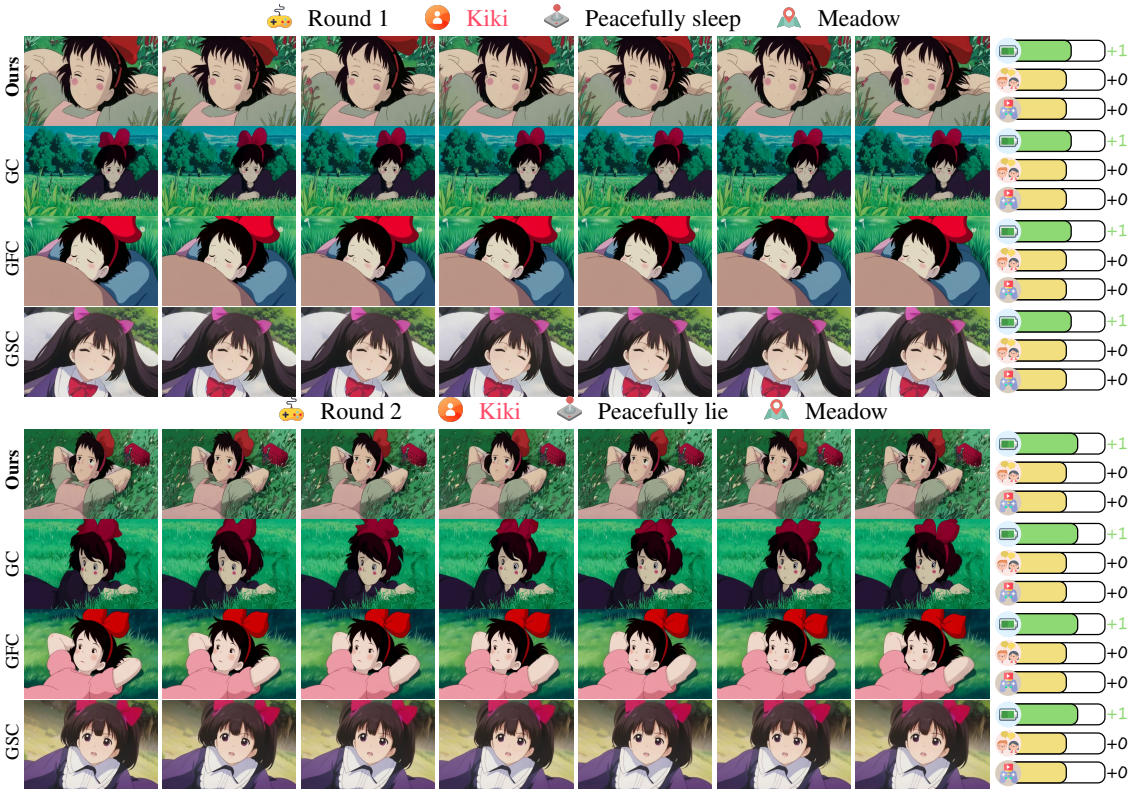

Figure 6. Visualization of Game 2.

Figure 7. Continuation of the Visualization of Game 2.

Figure 8. Continuation of the Visualization of Game 2.