An Interactive Agent Foundation Model: Appendix

1. Pre-training Loss Curves

As shown in Figure 1, the MAE training loss is somewhat unstable during the initial stages of pre-training. We note that the token prediction losses do not exibit this quality.

2. Using BLEU-4 for Action Evaluation

In order to determine the suitability of using BLEU-4 as a metric for evaluating the performance of our game-playing agent via held-out labeled data, we run two experiments using simple, 3-layer MLP agents in the CartPole[1] and SpaceInvaders[2] environments. We plot expert-normalized game scores vs BLEU-4 scores on a held-out test set in Figure 2. We show a strong non-linear correlation between BLEU-4 and the overall game scores, indicating that BLEU-4 can serve as a useful proxy metric for evaluating model performance when direct simulation is not possible.

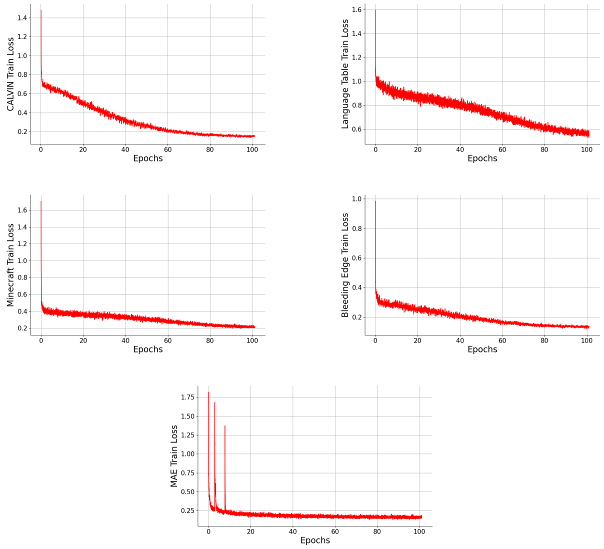


Figure 1. We plot each component of the training loss over 100 epochs of pre-training. The MAE component exhibits instabilities in the first 8 epochs.

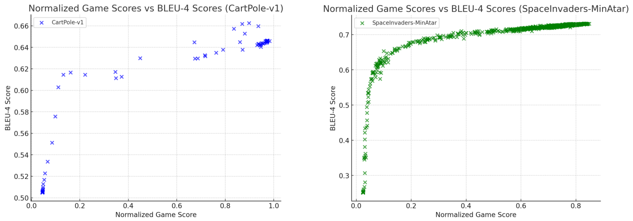


Figure 2. We plot expert-normalized game scores in simulation against BLEU-4 scores on a held-out evaluation set. We show that BLEU-4 can be used as an effective way to gauge agent modeling abilities in both the CartPole and SpaceInvaders environments.

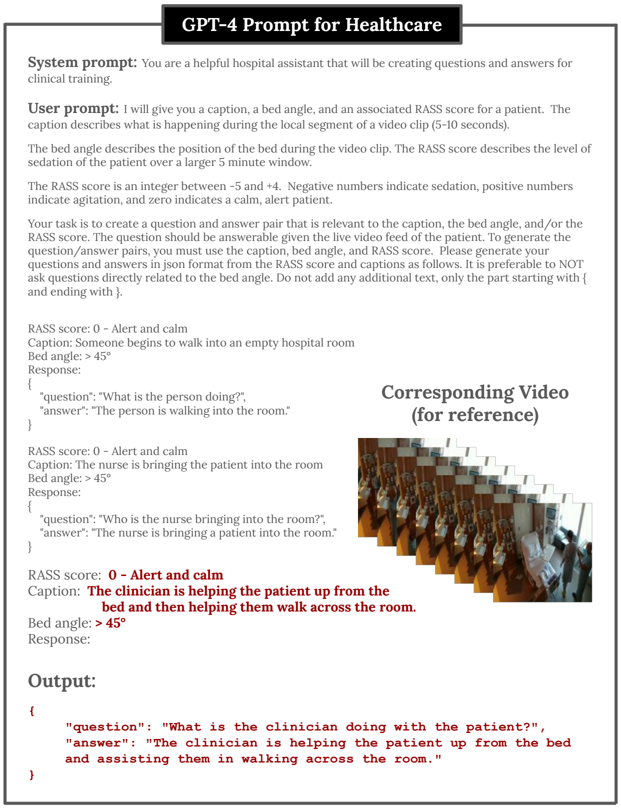
3. Visual Encoder Architecture Details

To effectively handle images and video inputs jointly, we use a divided space-time attention similar to [1]. We initialize our visual encoder from CLIP ViT-B/16 [3], and learn gated temporal attention layers after each spatial attention layer. By initializing the learned temporal embeddings and gated temporal attention blocks to identity functions, we can more effectively initialize our encoder with ViTs such as CLIP and retain image-level and performance without any additional training. Gaming and robotics use a frame-level visual encoder so that the agent is able to observe a continuous stream of tokens and act after every frame. Likewise, the MAE pre-training objective only uses frame-level information from the robotics and gaming datasets. We further mask 75% of the image patches during training, and use a MAE-decoder similar to [2]. For healthcare, we leverage the video understanding capabilities of our visual encoder since the tasks (video captioning, question answering, and action recognition) often require viewing an entire sequence of frames before making a prediction.

4. GPT-4 Prompting

We use GPT-4 to generate text prompts for our gaming pre-training and to generate question-answer pairs for our healthcare evaluation. We show the corresponding GPT-4 Prompts used in Figure 4 and Figure 3.

[1] https://github.com/openai/gym/tree/master
[2] https://github.com/kenjyoung/MinAtar



GPT-4 Prompt for Healthcare

System prompt: You are a helpful hospital assistant that will be creating questions and answers for clinical training.

User prompt: I will give you a caption, a bed angle, and an associated RASS score for a patient. The caption describes what is happening during the local segment of a video clip (5-10 seconds).

The bed angle describes the position of the bed during the video clip. The RASS score describes the level of sedation of the patient over a larger 5 minute window.

The RASS score is an integer between -5 and +4. Negative numbers indicate sedation, positive numbers indicate agitation, and zero indicates a calm, alert patient.

Your task is to create a question and answer pair that is relevant to the caption, the bed angle, and/or the RASS score. The question should be answerable given the live video feed of the patient. To generate the question/answer pairs, you must use the caption, bed angle, and RASS score. Please generate your questions and answers in json format from the RASS score and captions as follows. It is preferable to NOT ask questions directly related to the bed angle. Do not add any additional text, only the part starting with { and ending with }.

```
RASS score: 0 - Alert and calm
Caption: Someone begins to walk into an empty hospital room
Bed angle: > 45°
Response:
{
  "question": "What is the person doing?",
  "answer": "The person is walking into the room."
}

RASS score: 0 - Alert and calm
Caption: The nurse is bringing the patient into the room
Bed angle: > 45°
Response:
{
  "question": "Who is the nurse bringing into the room?",
  "answer": "The nurse is bringing a patient into the room."
}

RASS score:  0 - Alert and calm
Caption:  The clinician is helping the patient up from the
            bed and then helping them walk across the room.
Bed angle: > 45°
Response:
```

Corresponding Video (for reference)

Output:

```
{
     "question": "What is the clinician doing with the patient?",
     "answer": "The clinician is helping the patient up from the bed
     and assisting them in walking across the room."
}
```

Figure 3. **Our GPT-4 Prompt for Generating Healthcare QA Examples.** By ensuring the usage of non-identifying video captions and documentation data, we prevent any identifiable patient data leakage to GPT-4 while simultaneously generating additional visual-language training data. For the particular example shown, we use a RASS score of "0 - Alert and calm", a caption of "The clinician is helping the patient up from the bed and then helping them walk across the room.", and a bed angle of "$> 45°$".

5. Example Outputs

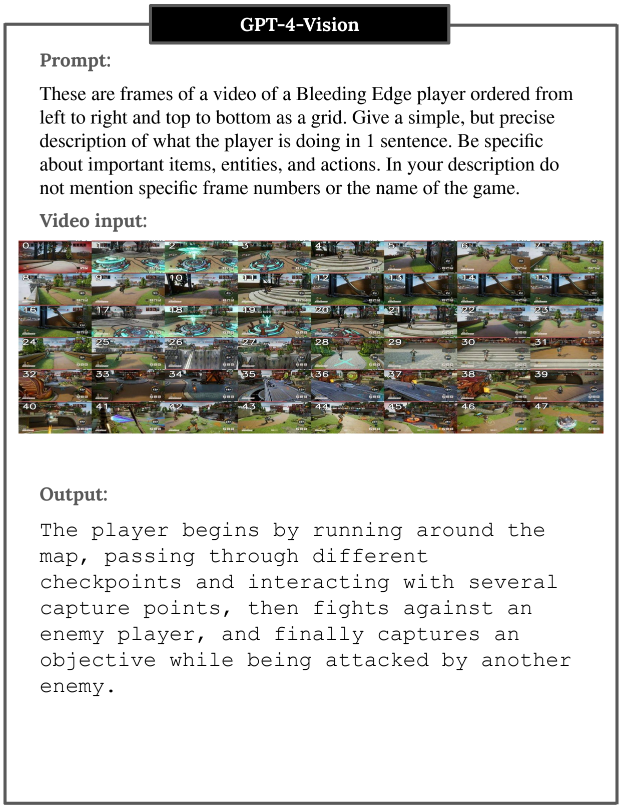
The remainder of the Appendix shows examples of our model predicting actions on unseen data across all domains. We show robotics simulation data in Table 1 and 2. We show example outputs for healthcare in Table 3, and show example outputs for gaming in Table 4 and 5.

References

[1] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1728–1738, 2021. 1

[2] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *CVPR*, 2022. 1

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1



GPT-4-Vision

Prompt:

These are frames of a video of a Bleeding Edge player ordered from left to right and top to bottom as a grid. Give a simple, but precise description of what the player is doing in 1 sentence. Be specific about important items, entities, and actions. In your description do not mention specific frame numbers or the name of the game.

Video input:

Output:

```
The player begins by running around the
map, passing through different
checkpoints and interacting with several
capture points, then fights against an
enemy player, and finally captures an
objective while being attacked by another
enemy.
```

Figure 4. Our GPT-4V prompt for games like Bleeding Edge that have 3rd person viewpoints and visually complex scenes. In order to input a large number of frames (48) to GPT-4V, we input the frames as a grid with frame numbers overlaid on each frame (as shown above).

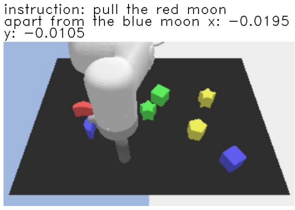
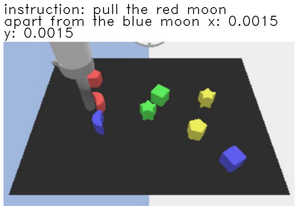
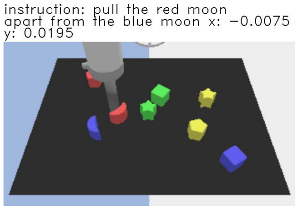
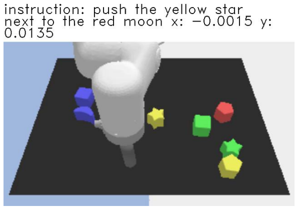
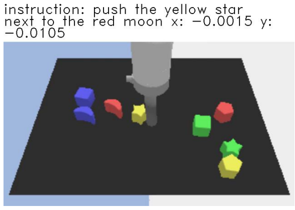
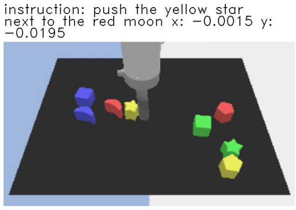
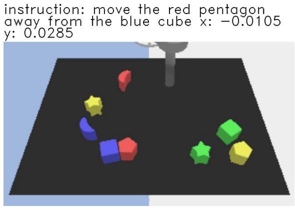
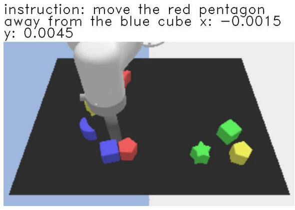
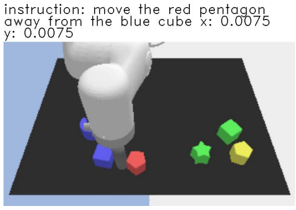
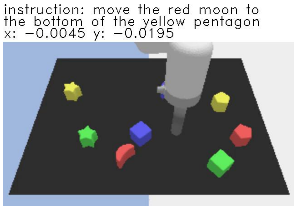
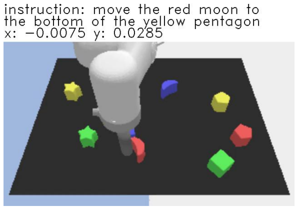
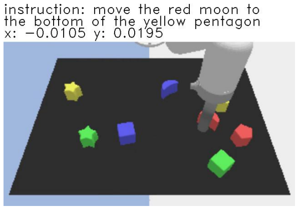
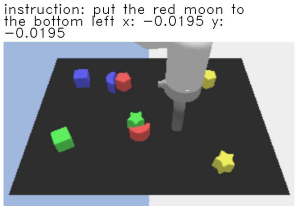
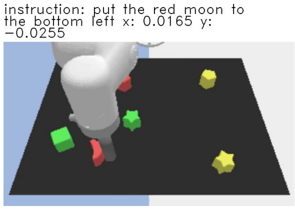
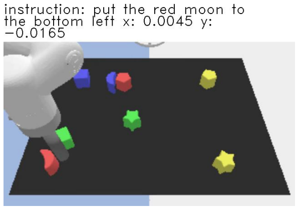
Text instruction	Start frame		Middle frame		End frame
Pull the red moon apart from the blue moon.	 instruction: pull the red moon apart from the blue moon x: −0.0195 y: −0.0105	→	 instruction: pull the red moon apart from the blue moon x: 0.0015 y: 0.0015	→	 instruction: pull the red moon apart from the blue moon x: −0.0075 y: 0.0195
Push the yellow start next to the red moon.	 instruction: push the yellow star next to the red moon x: −0.0015 y: 0.0135	→	 instruction: push the yellow star next to the red moon x: −0.0015 y: −0.0105	→	 instruction: push the yellow star next to the red moon x: −0.0015 y: −0.0195
Move the red pentagon away from the blue cube.	 instruction: move the red pentagon away from the blue cube x: −0.0105 y: 0.0285	→	 instruction: move the red pentagon away from the blue cube x: −0.0015 y: 0.0045	→	 instruction: move the red pentagon away from the blue cube x: 0.0075 y: 0.0075
Move the red moon to the bottom of the yellow pentagon.	 instruction: move the red moon to the bottom of the yellow pentagon x: −0.0045 y: −0.0195	→	 instruction: move the red moon to the bottom of the yellow pentagon x: −0.0075 y: 0.0285	→	 instruction: move the red moon to the bottom of the yellow pentagon x: −0.0105 y: 0.0195
Pull the red moon to the bottom left.	 instruction: put the red moon to the bottom left x: −0.0195 y: −0.0195	→	 instruction: put the red moon to the bottom left x: 0.0165 y: −0.0255	→	 instruction: put the red moon to the bottom left x: 0.0045 y: −0.0165

Table 1. We show 5 unique demonstrations from Language Table, where our model successfully follows the text instruction. In addition to the high level instruction, we also show the low-level predicted actions of our agent above each frame.

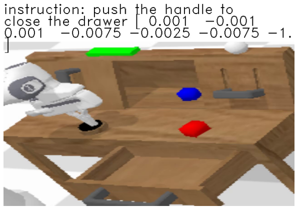
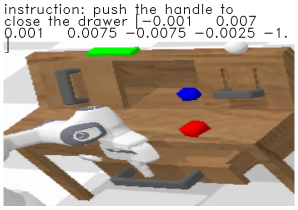
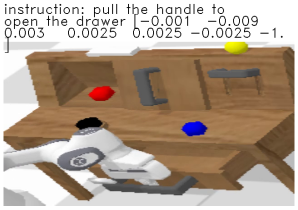
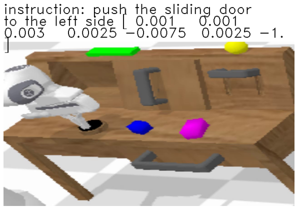
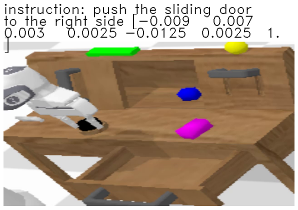
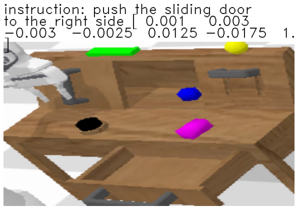
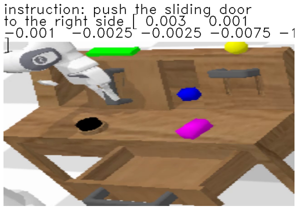
Text instruction	Start frame		Middle frame		End frame
Push the handle to close the drawer.	 instruction: push the handle to close the drawer [0.001 −0.001 0.001 −0.0075 −0.0025 −0.0075 −1.]	→	 instruction: push the handle to close the drawer [0.011 −0.013 −0.001 −0.0025 −0.0025 0.0175 1.]	→	 instruction: push the handle to close the drawer [−0.001 0.007 0.001 0.0075 −0.0075 −0.0025 −1.]
Lift the red block from the sliding cabinet.	 instruction: lift the red block from the sliding cabinet [−0.013 0.009 0.013 −0.0025 −0.0175 −0.0025 −1.]	→	 instruction: lift the red block from the sliding cabinet [0.003 0.005 −0.003 −0.0025 −0.0075 −0.0125 1.]	→	 instruction: lift the red block from the sliding cabinet [−0.003 −0.009 0.005 0.0125 0.0075 −0.0175 −1.]
Pull the handle to open the drawer.	 instruction: pull the handle to open the drawer [0.009 −0.005 −0.003 0.0025 0.0075 0.0025 1.]	→	 instruction: pull the handle to open the drawer [−0.001 0.003 −0.007 −0.0025 −0.0025 0.0075 1.]	→	 instruction: pull the handle to open the drawer [−0.001 −0.009 0.003 0.0025 0.0025 −0.0025 −1.]
Push the sliding door to the left side.	 instruction: push the sliding door to the left side [0.001 0.001 0.003 0.0025 −0.0075 0.0025 −1.]	→	 instruction: push the sliding door to the left side [−0.001 0.005 −0.003 −0.0025 0.0025 −0.0025 1.]	→	 instruction: push the sliding door to the left side [−0.011 −0.001 0.001 0.0025 −0.0025 0.0125 −1.]
Push the sliding door to the right side.	 instruction: push the sliding door to the right side [−0.009 0.007 0.003 0.0025 −0.0125 0.0025 1.]	→	 instruction: push the sliding door to the right side [0.001 0.003 −0.003 −0.0025 0.0125 −0.0175 1.]	→	 instruction: push the sliding door to the right side [0.003 0.001 −0.001 −0.0025 −0.0025 −0.0075 −1]

Table 2. We show 5 unique demonstrations from CALVIN, where our model successfully follows the text instruction. In addition to the high level instruction, we also show the low-level predicted actions of our agent above each frame.

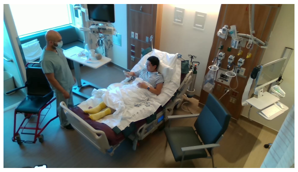
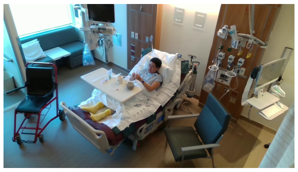
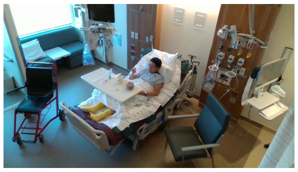
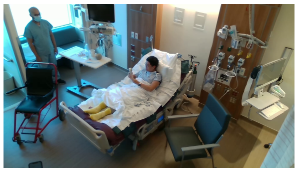
Task	Start frame	End frame	Model Output
Video Captioning			The patient is awake and calm. The patient is cooperative. The patient is alert
Video Question Answering			Q: Where is the patient? A: patient is in deep sedation. The patient likely requires assistance.
Action Recognition (RASS)			0 - Alert and calm
Video Captioning			The patient is awake and calm. They are speaking on the phone.

Table 3. We show 4 demonstrations of our agent model's outputs on a held-out dataset that uses actors instead of actual patients. We demonstrate our model's outputs across 3 different tasks: video captioning, visual question answering, and RASS score prediction (action recognition).

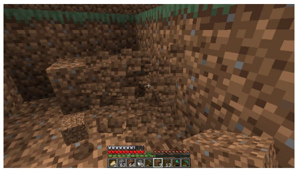
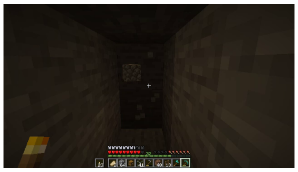
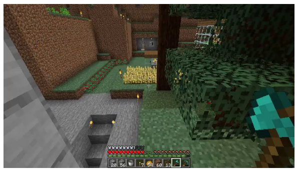
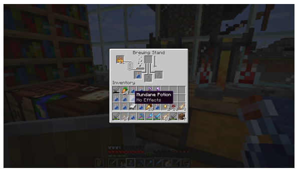
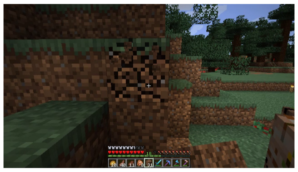
Text instruction	Start frame	Predicted Action	Ground Truth Action
the player is digging and placing dirt blocks to terraform the terrain around their house...		[STARTACTION] [attack] [CAMERAX0] [CAMERAY-1] [ENDOFACTION]	[STARTACTION] [attack] [ENDOFACTION]
the player is mining underground using a diamond pickaxe, gathering cobblestone, coal, iron ore...		[STARTACTION] [attack] [CAMERAX-3] [CAMERAY0] [ENDOFACTION]	[STARTACTION] [attack] [CAMERAX-3] [CAMERAY0] [ENDOFACTION]
the minecraft player is moving around a village ...		[STARTACTION] [forward] [sprint] [ENDOFACTION]	[STARTACTION] [forward] [sprint] [ENDOFACTION]
the player is using a brewing stand ...		[STARTACTION] [sneak] [use] [ENDOFACTION]	[STARTACTION] [sneak] [ENDOFACTION]
the player is ... terraforming by digging ...		[STARTACTION] [attack] [ENDOFACTION]	[STARTACTION] [attack] [ENDOFACTION]

Table 4. We show 5 demonstrations from a held-out Minecraft dataset. In addition to the high level instruction, we show the low-level predicted actions and ground truth actions. We truncate the instructions to show only the parts relevant to the current frames. The most common errors are slight differences in camera movements and occasionally performing unnecessary actions. Note that sometimes the ground truth values are not the only valid actions; for instance, the fourth example predicts that the player will click the bottle, which happens a few frames later in the ground truth trajectory.

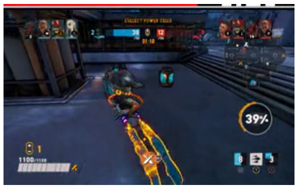
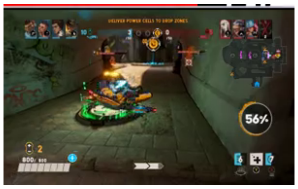
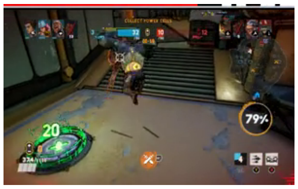
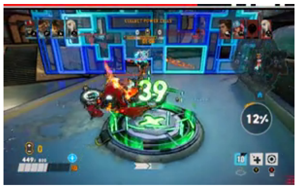
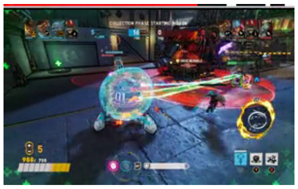
Text instruction	Start frame	Predicted Action	Ground Truth Action
the player is using a character with a sword to fight enemies and collect power cells ...		[STARTACTION] [lockon][meleeattack] [lrot214] [lmag4] [ENDOFACTION]	[STARTACTION] [lockon][meleeattack] [lrot213] [lmag4] [ENDOFACTION]
the player is riding a hoverboard-like vehicle ... avoiding or attacking enemy players ...		[STARTACTION] [lockon][meleeattack] [lrot204] [lmag4] [ENDOFACTION]	[STARTACTION] [lockon][meleeattack] [lrot201] [lmag4] [ENDOFACTION]
the player starts by descending some stairs towards an open area where they engage in combat with an enemy player ...		[STARTACTION] [jump] [lockon][specialability1] [lrot199] [lmag4] [ENDOFACTION]	[STARTACTION] [jump] [lockon][meleeattack] [lrot201] [lmag4] [ENDOFACTION]
the player ... captures an objective point while fighting off multiple opponents ...		[STARTACTION] [lockon][meleeattack] [lrot63] [lmag4] [ENDOFACTION]	[STARTACTION] [lockon][meleeattack] [lrot63] [lmag4] [ENDOFACTION]
a bleeding edge player is controlling a robot character with a sword ... engaging in combat with enemy players ...		[STARTACTION] [evade] [lrot236] [lmag4] [ENDOFACTION]	[STARTACTION] [evade] [lrot236] [lmag4] [ENDOFACTION]

Table 5. We show 5 unique demonstrations from a held-out Bleeding Edge dataset. In addition to the high level instruction, we show the low-level predicted actions and ground truth actions. We truncate the instructions to show only the parts relevant to the current frames. The most common errors are slight deviations from the precise value of the joysticks, which are naturally noisy. Some other errors include predicting the wrong type of attack, though this typically happens in situations where multiple attacks are still valid.