

CRADLE

Weihao Tan^{2*}, Ziluo Ding¹, Wentao Zhang², Boyu Li¹, Bohan Zhou^{3*}, Junpeng Yue^{3*},
Haochong Xia², Jiechuan Jiang³, Longtao Zheng², Xinrun Xu¹, Yifei Bi¹, Pengjie Gu²,
Xinrun Wang², Börje F. Karlsson¹, Bo An^{2†}, Zongqing Lu^{3, 1†}

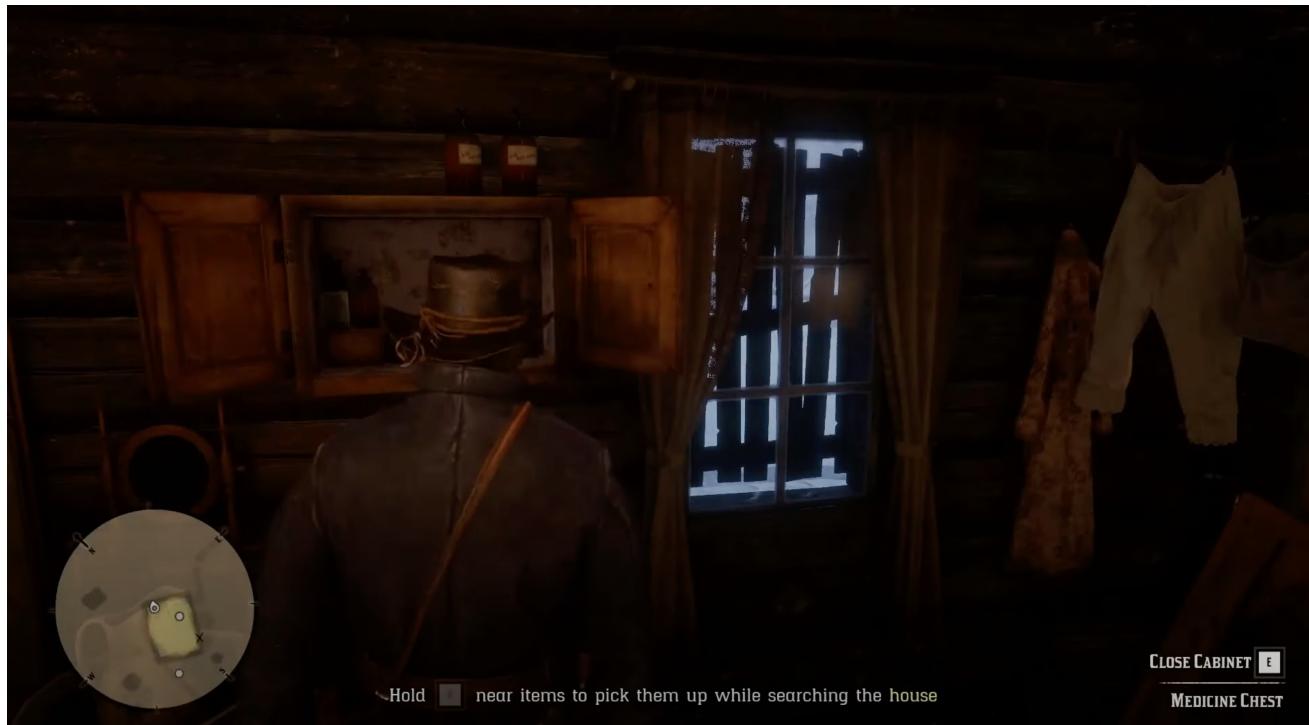
AI that can play Red Dead Redemption 2
(though it is 20x slower than humans)

Presented by:
John Tan Chong Min

Hold  to ride faster

Let's take a look!

<https://www.youtube.com/watch?v=Cx-D708BedY>

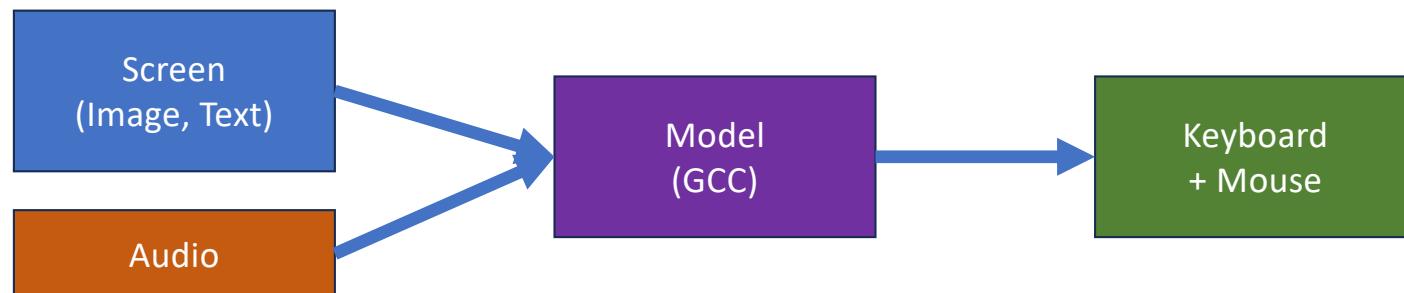


General Computer Control



Overview

- **General Computer Control (GCC)** is a setting where an agent controls a computer through only standardized human-like interactions, i.e., using the screen and (optionally) audio as input and keyboard and mouse operations as output.
- GCC provides a universal interface to any computer-based task without access to the application source code nor APIs.



GCC: A path towards AGI?

- Food for thought:
 - Are humans **really generally intelligent?**



Are games good enough for AGI?

Pros

- Offers a **virtual testbed** for fast iteration - much cheaper than real-world interaction with a robot
- Tasks can help **with long-horizon planning**
- Can test out self-improvement abilities by measuring an **objective score**

Cons

- Environments **may not be reflective of real world** imagery / physics
- May be too fixed and **repeatable**
- **Optimising** for one game may not lead to performant results in others



Challenges for GCC

- Multimodal observations
 - Image, text, audio
- Accurate control of mouse and keyboard
 - Mouse movements are a continuous action space – difficult!
 - Need a suitable abstraction space
- Partially observable / changing environment
 - Need to remember and reflect on past interactions
 - Needs to constantly replan
- Requires exploration and self-improvement
 - Most information is not known beforehand

Recap: VOYAGER

Teacher comes up with something manageable for student

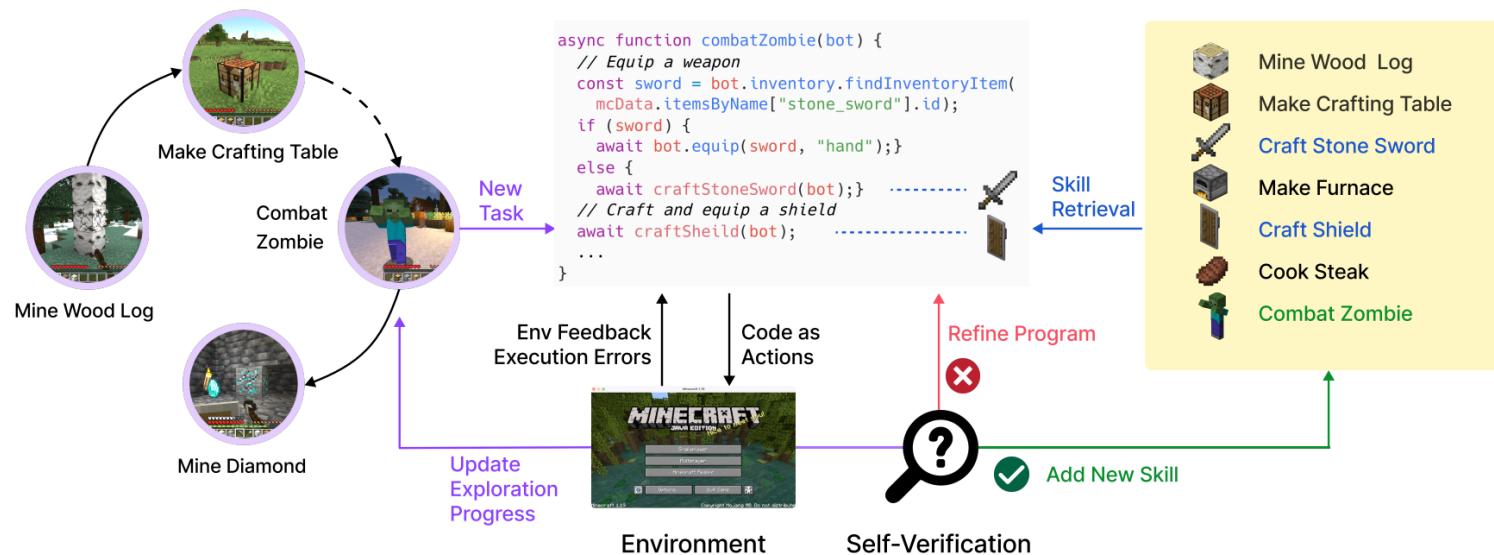
Automatic Curriculum

Student learns (multiple tries)

Iterative Prompting Mechanism

Student remembers (only good stuff)

Skill Library

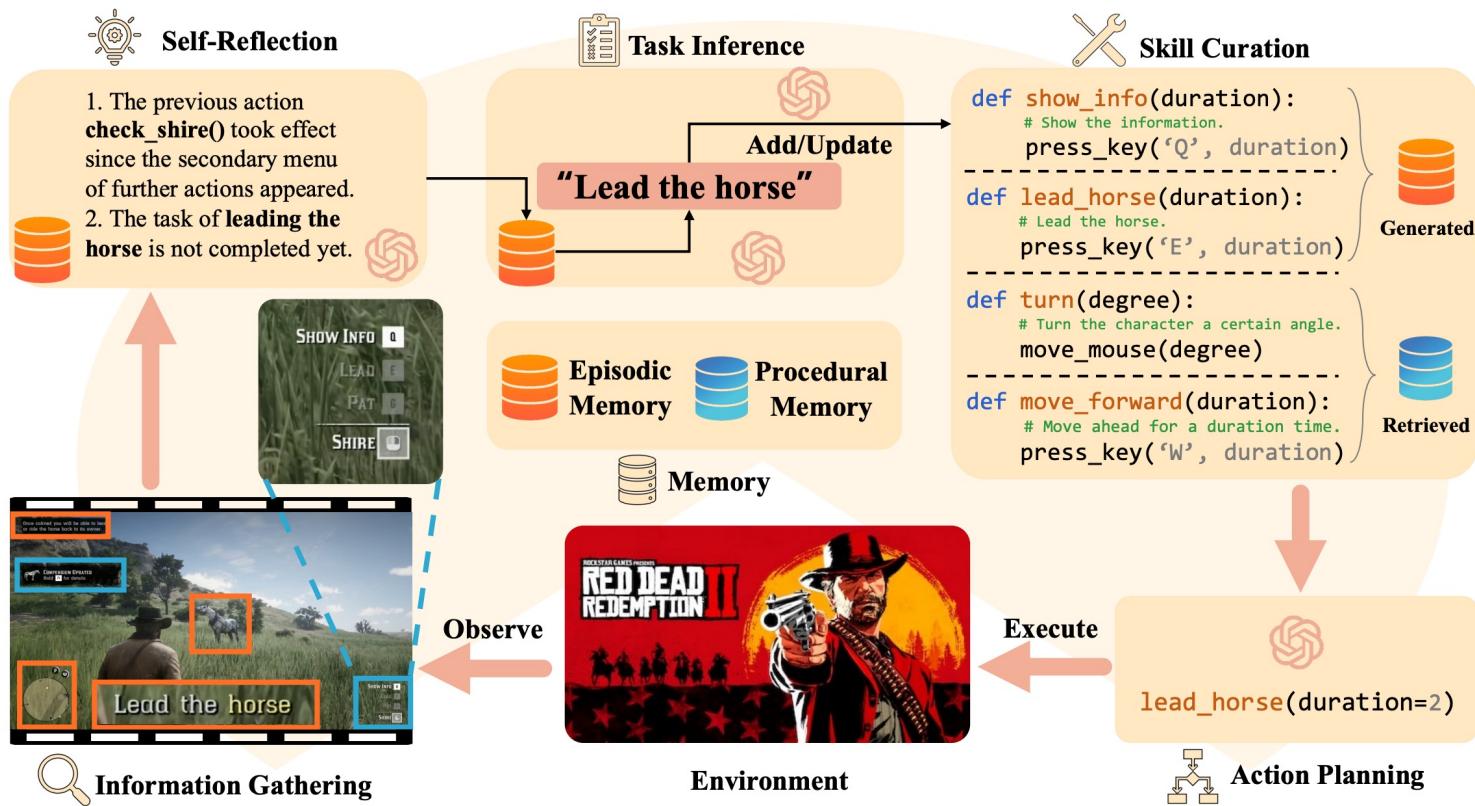


Voyager: An Open-Ended Embodied Agent with Large Language Models. Wang et al. 2023.

Key Improvements in CRADLE

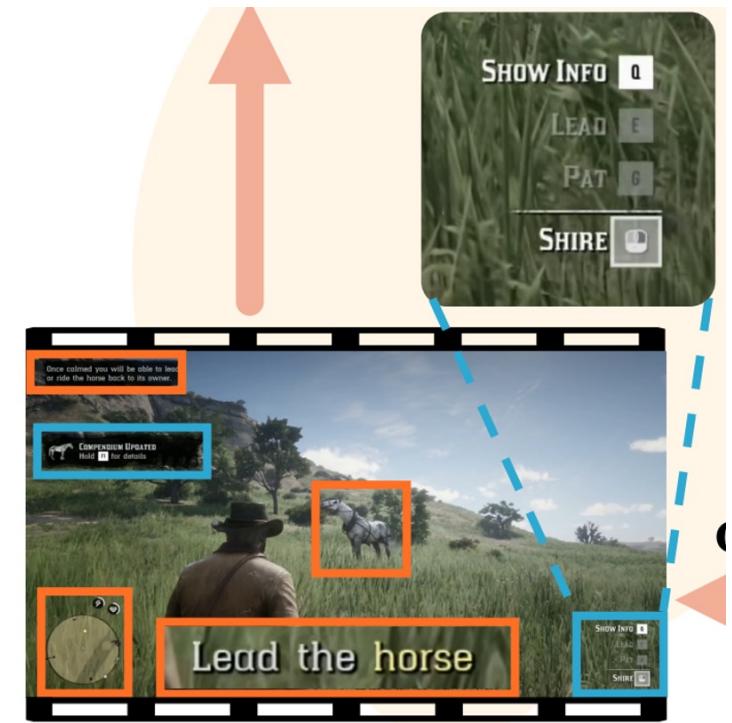
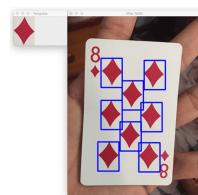
- Image Parsing Module
 - Multiple abstraction spaces in image domain – Image, OCR of text
- Reflection, summary of memory
 - Multiple abstraction spaces in memory – Full sequence, summary, reflection

CRADLE: UI -> Reflection -> Task Inference -> Skill Curation -> Action Planning



Multiple Abstraction Spaces - Vision

- Video -> Images
 - Key frames extracted from video using VideoSubFinder
- For key frames, obtain:
 - GPT4-V's overall perception of image
 - OCR of the text of various UI components
- Augmented with Grounding DINO:
 - Position of the UI elements and targets
- Augmented with Icon Matcher
 - Match icons with predefined templates using Multi-template Matching (e.g. minimap white arrow)



Reasoning and Planning Module

“Reflect on the past, summarize the present, and plan for the future”

- **Self-Reflection:** Evaluates (from 8 sequential image frames) whether last executed action was successfully carried out, whether task is completed. If fail, provide analysis for failure (impt!)
- **Task Inference:** Analyse the current situation to infer most suitable task for the current moment
- **Skill Curation:** Retrieve useful skills from procedural memory, update skills or generate new ones based on task
- **Action Planning:** Select skills from curated skill set and put these into sequence of executable actions to **Executor**

Task Inference

- Let GPT-4V propose current task to perform whenever it believes it is time to start a new task
- Follow newly generated task for h iterations, before going back to last long-horizon task in the stack
- More efficient than re-planning at every iteration, and not forget long-horizon tasks
- **My thoughts: Too convoluted – just need to have a list of tasks it needs to do, break each task down to subtasks to be executed by inner agents (ref. TaskGen)**

Base Priors for Action Space

- press(key, duration)
- hold(key, duration)
- release(key)
- pointer_move(x, y)

Skill Curation from in-game prompts

- Before code is stored, checks done to make sure no duplicate skill name and that code is valid
- *My concern: What if game doesn't teach you what to do*



Figure 6: Skill code generation based on in-game instructions. As the storyline progresses, the game will continually provide prompts on how to use a new skill via keystrokes or utilizing the mouse.

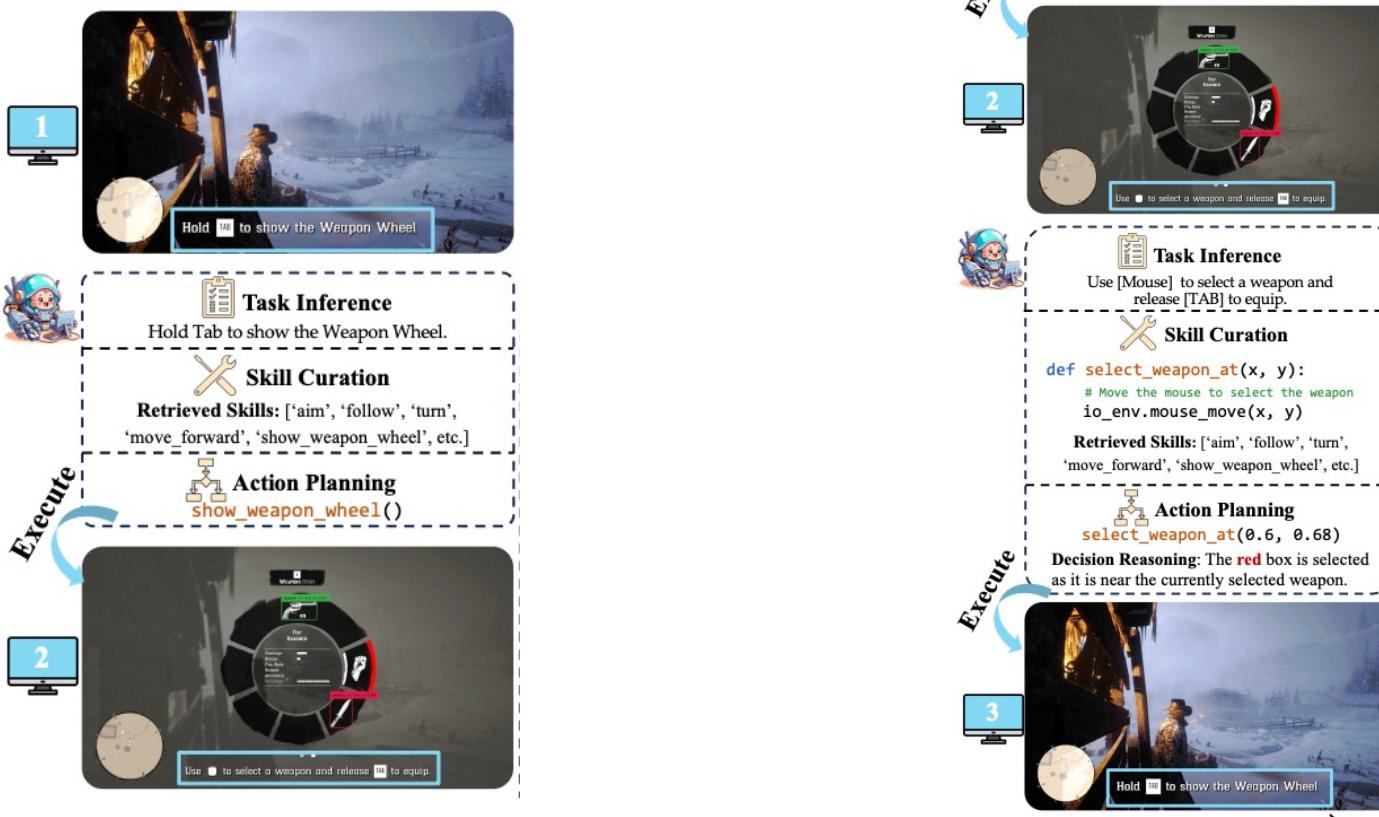
Action Planning

- Retrieve top k relevant skills for task from procedural memory, alongside with newly generated skills
- Provide: Current task, set of retrieved skills, other information about context
- Get GPT-4V to choose the skill, and give explanation

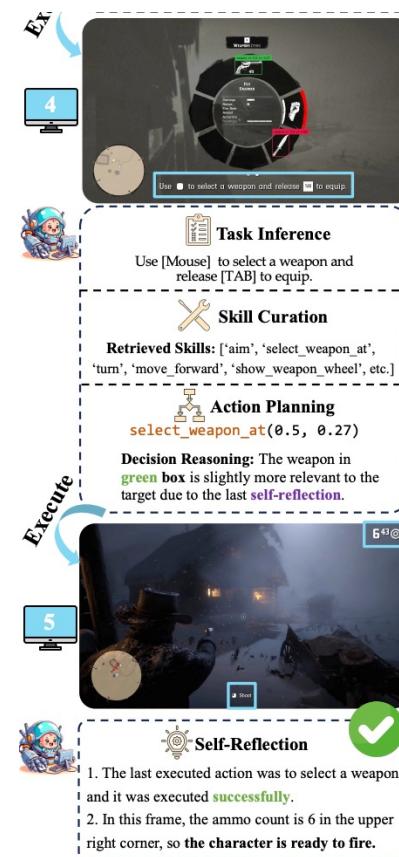
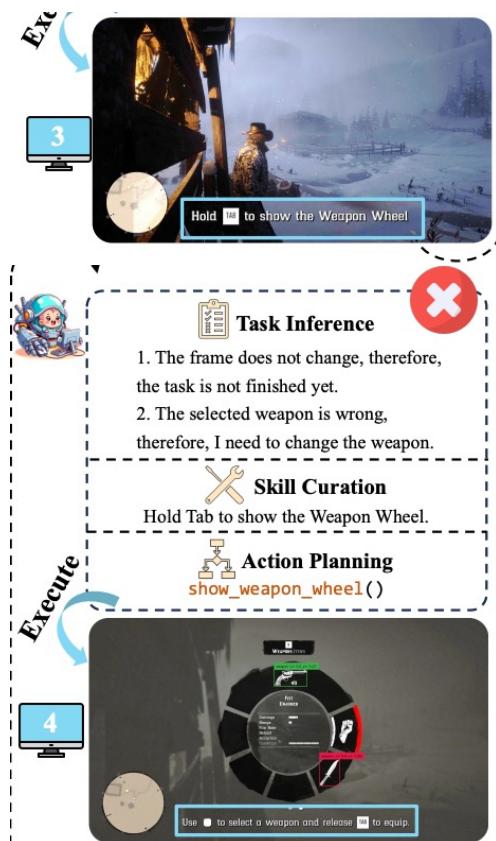
Memory

- Episodic Memory: Maintains current and past experiences
 - Key screenshots from recent ($k = 5$) video frames
 - Text and visual information, actions, tasks and reasoning from each module
 - Summarisation for new information to facilitate retrieval and storage
- Procedural Memory: Storing and retrieving skills in code form, which are learned from scratch or pre-defined
 - Actually most core skills are pre-defined in CRADLE – Agent has limited capacity to learn
 - E.g. `turn(degree)`, `move_forward(duration)`, `shoot(x, y)`, `select_item_at(x, y)`, `follow(duration)`, `navigate_path(duration)`, `fight()`

Task Solving 1/2



Task Solving 2/2



Challenges and Improvements

Mismatch between game instruction and environment



(a) ‘Pick gun’ unavailable. (b) ‘Open cabinet’ press timing. (c) ‘Hitch horse’ re-use of ‘e’ key

Figure 7: Examples of action execution uncertainty. Lack of environment feedback to actions and semantic gaps between action intent and game command can lead to challenging situations for agent reasoning.

Vision Module – Unable to localise objects well

- GPT4-V is just not good at localising
- Maybe need something like what AppAgent does – give the UI positions
- In paper: Augmented with Grounding DINO for bounding boxes of objects
- Could also consider augmenting with object recognition models like YOLOv8 to give bounding boxes of objects and relative motion



ChatGPT

There are **no immediate obstacles** directly in front of the character in the image. The character has a **clear path ahead** on a muddy street in what appears to be a Western-style town setting.



ChatGPT

The shopkeeper in the image is on the **right side**, behind the counter.

(a) GPT-4V fails to recognize obstacles in the environment and the position of the shopkeeper.

Vision Module – Mismatch of local context

- Perhaps a general agent is not possible after all – we need the local context, in this case, the game controls



ChatGPT

Based on the provided game screenshot, I've identified the following control prompt:

- "Hold L ⚡ to ride faster"

This control suggests that holding a certain key (which seems to be an **upward arrow key**, often associated with the **'W' key** for forward movement in many games) will make the character ride faster.



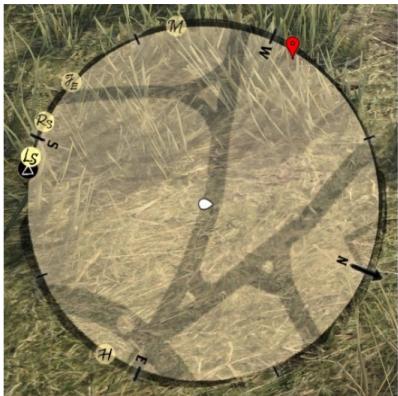
ChatGPT

From the provided image screenshot from the game, I can identify the following keyboard and mouse control prompts:

1. "Toggle View" - The key for this action is "V".
2. "Zoom" - The key for this action is "O".
3. "Flip" - The key for this action is "L".
4. "Back" - The key for this action is "ESC".

(b) GPT-4V struggles to recognize the icons for keys on keyboard and mouse buttons.

Vision Module – Lack of understanding of specifics



The central white arrow is facing towards the **left**.



The central white arrow with the white arrowhead is facing towards the **left**.

(c) GPT-4V cannot understand the correct direction of arrow points, *i.e.*, character, towards in the mini-map.

Figure 9: Example situations of GPT-4V's limitations in understanding visual information from the game.

Why is it so hard?

Table 1: Tasks in the first two missions of RDR2. *Difficulty* refers to how hard it is for our agent to accomplish the corresponding tasks. Figures 10 and 11 showcase snapshots of each task (specific sub-figures marked in parenthesis in the table).

Mission 1: Explore shelter	Description	Difficulty
Follow Dutch (Fig. 10a)	Follow Dutch to the small town, by riding the horse.	Easy
Hitch horse (Fig. 10b)	Dismount at the hitching post, after reaching the town.	Easy
Go to shed (Fig. 10c)	Move to the nearby shed to take cover.	Easy
Choose weapon (Fig. 10d)	Choose the correct weapon to prepare for combat.	Hard
Protect Dutch (Fig. 10e)	Protect Dutch from the enemies through shooting.	Hard
Search house (Fig. 10f)	Follow Dutch to enter the house and search for supplies.	Hard
Eat something (Fig. 10g)	Open Satchel and eat a provision to restore some Health Core.	Medium
Investigate barn (Fig. 10h)	Leave the horse and go to the barn to investigate.	Easy
Defeat O'Driscoll (Fig. 10i)	Fight with the enemy hidden in the barn.	Medium
Pick up equipment (Fig. 10j)	Find and pick up gun and hat lost during barn fight.	Hard
Lead horse (Fig. 10k)	Calm and Lead the horse out of the barn to hitching post.	Medium
Mission 2: Rescue John		
Follow Javier (Fig. 11a)	Follow Javier through the snow mountain to look for John.	Medium
Equip shotgun (Fig. 11b)	Equip correct shotgun to prepare for combat.	Medium
Look for John (Fig. 11c)	Crouch down, enter narrow tunnel, and climb up cliffs.	Hard
Shoot wolves (Fig. 11d)	Shoot wolves that will attack you and companions.	Hard
Protect Javier from wolves (Fig. 11e)	Ride horse and protect your companions from wolves.	Hard

- UI position-based stuff is difficult
- Difficult when there is an open-ended task
- Difficult when real-time interaction is needed

Are we there yet? – Not yet

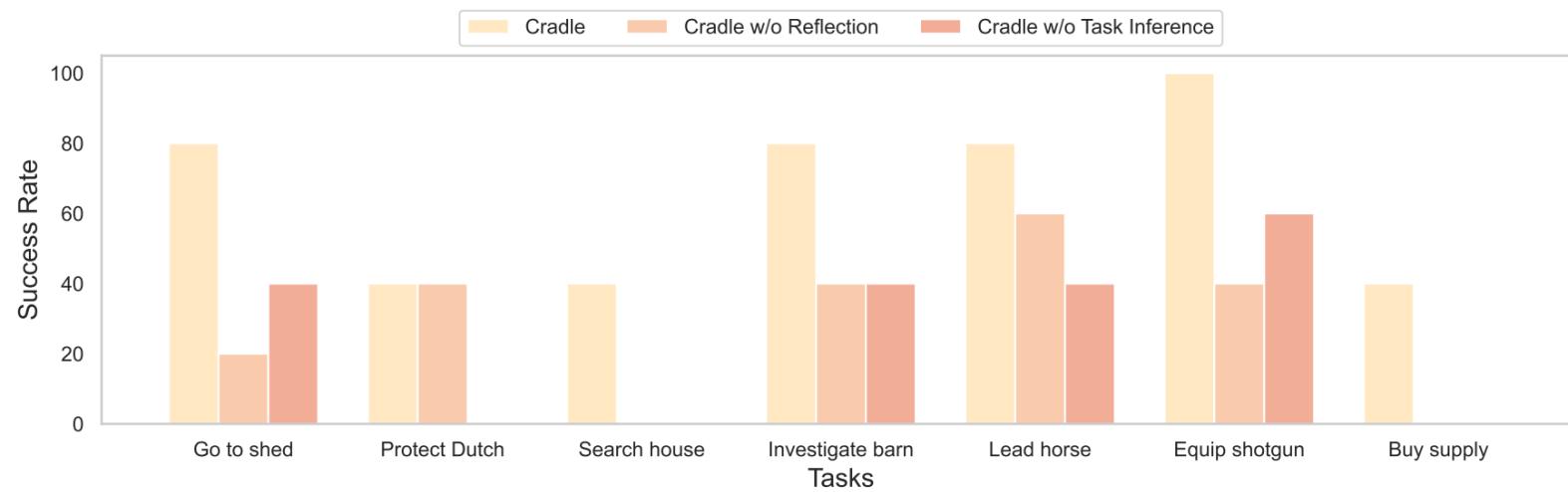


Figure 8: CRADLE performance on seven representative tasks in RDR2. The first six tasks are from the two missions in the main storyline and the last one is one of the open-ended missions. Every task is tested five times with a maximum of ten minutes in-game time. The test will also terminate if the mission fails.

Is RDR 2 hard for decision making?

Well, not quite

RDR may be easy – Instructions are given

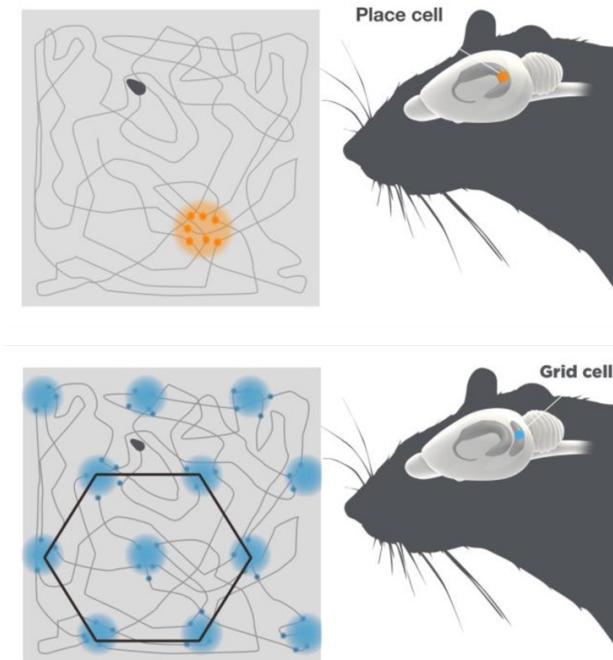


RDR may be easy – Location in-built in minimap

- RDR2 is actually easier as there is an in-built minimap
- Hippocampus in animals store positional information without minimap



RDR2 Compass



Place cells and Grid cells
<https://tkukurin.github.io/storing-episodic-memory>

My thoughts

- LLM is not the best tool to use for long-horizon planning
 - Use rule-based planner like Monte Carlo Tree Search
 - Success checker can just use completion of list of planned actions as a guide rather than LLM-based check
- Vision Module is not advanced enough
 - Vision Transformer loses too much specific object position information
 - Need multiple abstraction spaces, not just one
- Prompts in CRADLE are TOO long – lose context easily
 - Should do task-based prompting to augment required information
- Perhaps AGI is not that general after all
 - Need to imbue with more local context

Questions to Ponder

- What abstraction spaces are needed for video, and for memory?
- Could the problem be better solved with sub-agents that can do various tasks like navigation, shooting, purchasing of shop items etc?
- How can we reduce the latency for decision making?
- How can a system learn the correct actions without the game prompts? Do we need prior knowledge? How much prior knowledge is needed?