GROUNDING ARTIFICIAL GENERAL INTELLIGENCE WITH ROBOTICS: THE PETITCAT PROJECT

Howard Schneider
Sheppard Clinic North, Vaughan, Canada
hschneidermd@alum.mit.edu

Olivier L. Georgeon
UCLy, Lyon, France
ogeorgeon@univ-catholyon.fr

BICA*AI 2024
2024 Annual International Conference on Brain-Inspired Cognitive Architectures for Artificial Intelligence

OVERVIEW OF THE PRESENTATION

1. There is a need for grounding and embodiment of BICA ideas and systems
2. The PetitCat Open Source robotic grounding for AI/BICA Projects

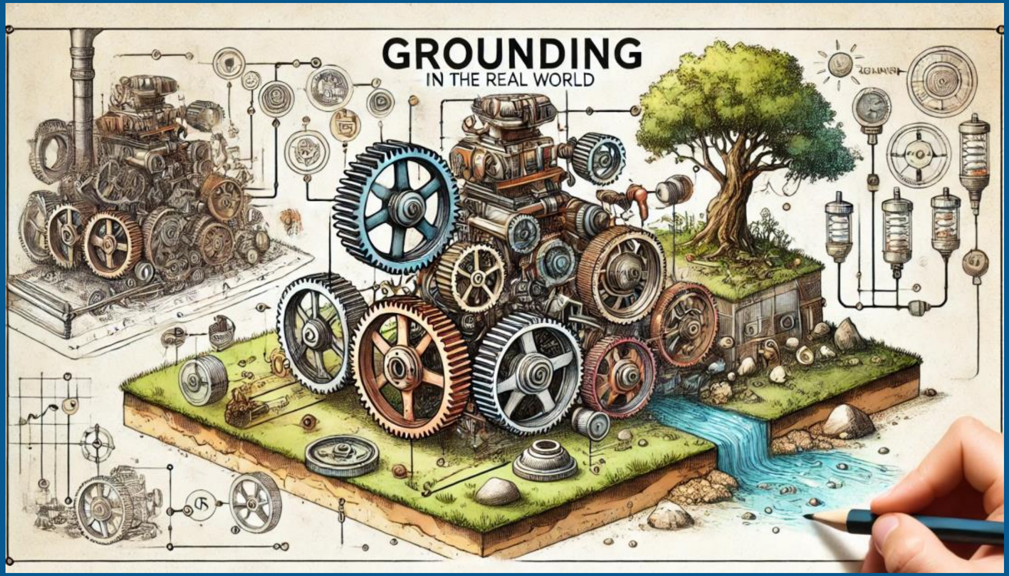
Open Hardware

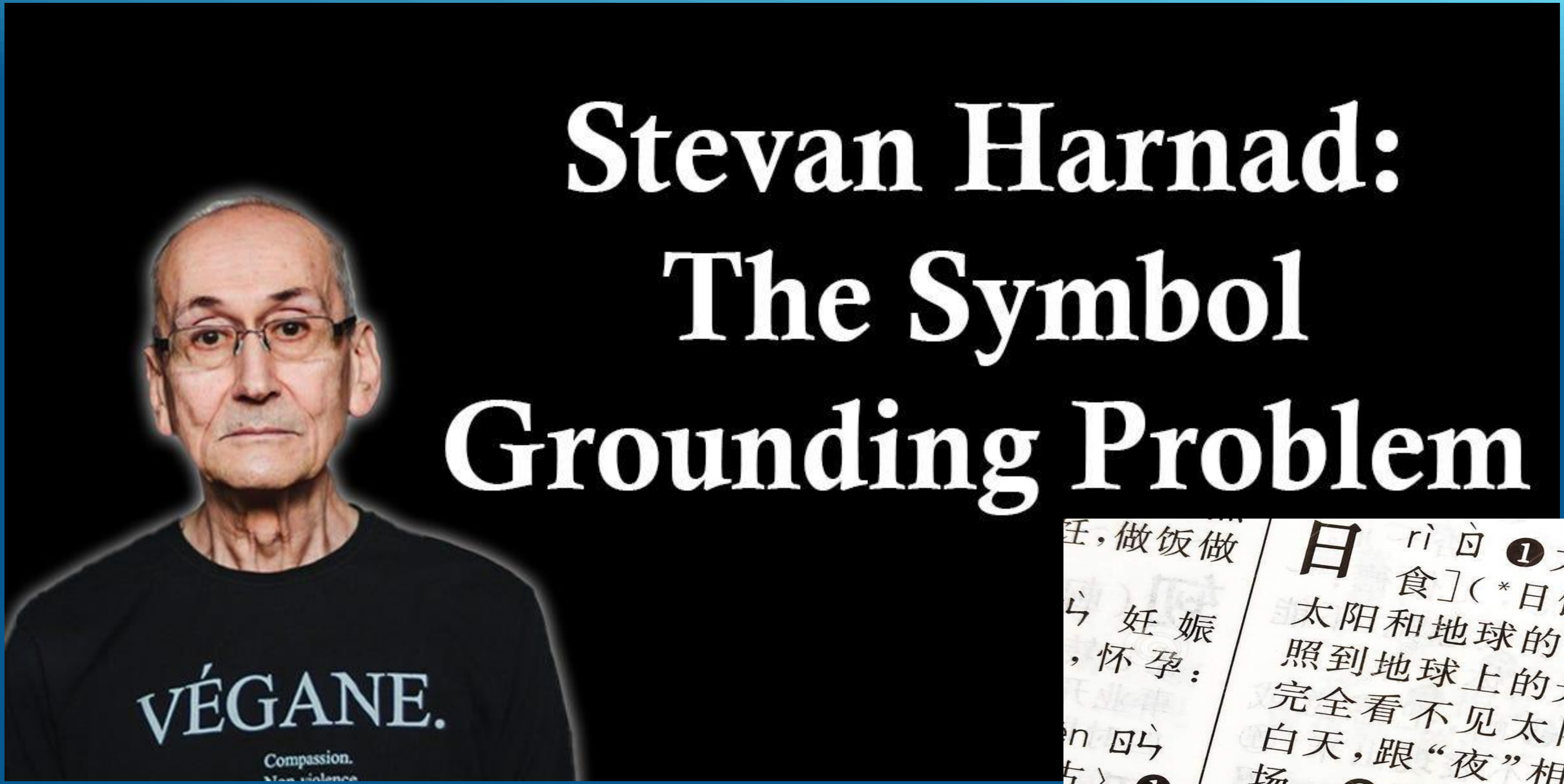
Open Software

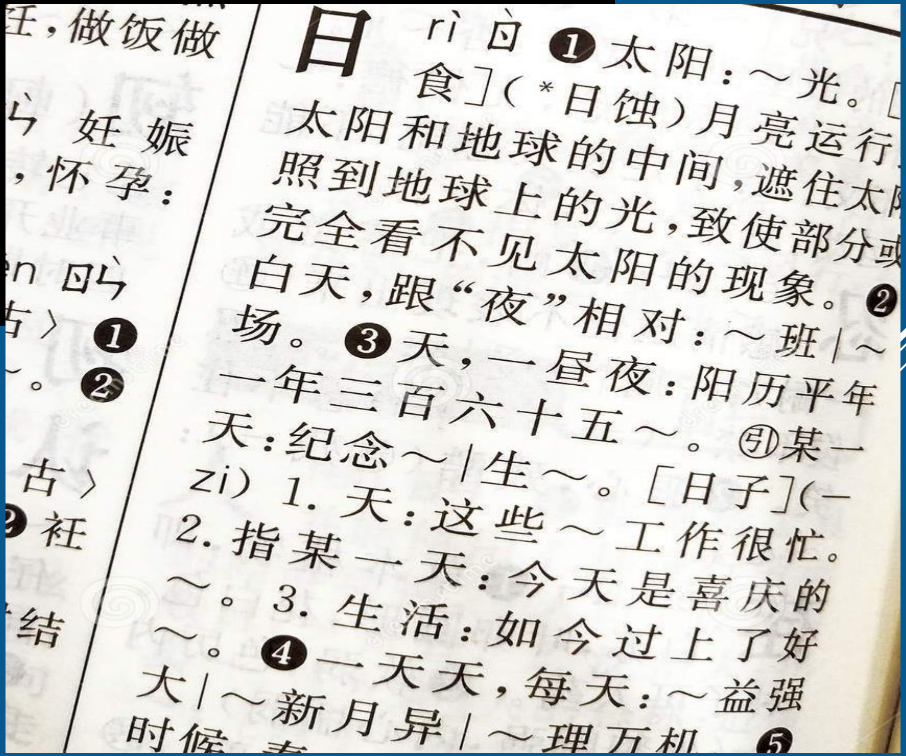
3. Experimental use of the PetitCat project

- Note: This paper was originally a ‘Short Technical Communication’ for the BICA*AI Track of AGI2024 Proceedings, but it is enhanced for the new BICA*AI 2024 Proceedings

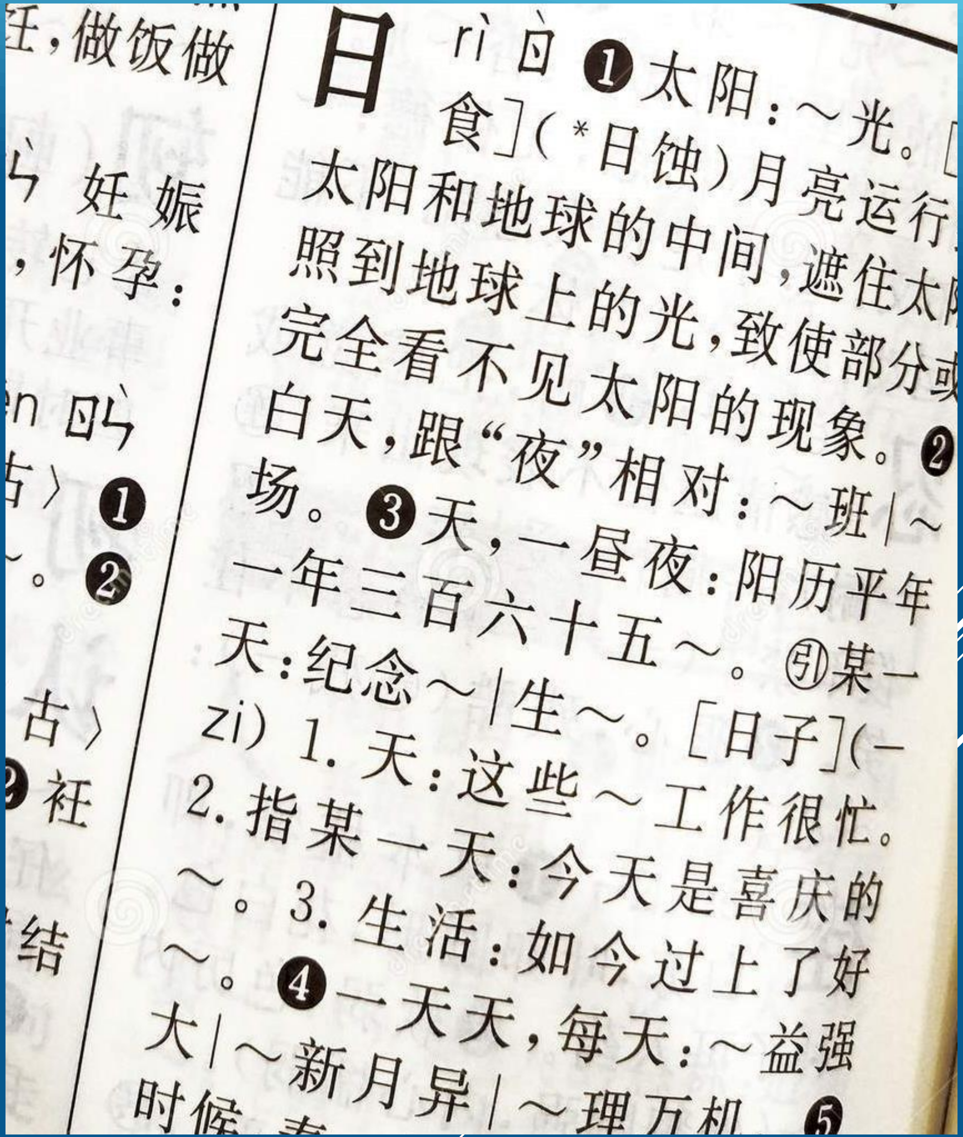
1. THERE IS A NEED FOR GROUNDING AND EMBODIMENT OF BICA IDEAS AND SYSTEMS






日 rì ㄖˋ ❶太阳：～光。[～
食]（*日蚀）月亮运行
太阳和地球的中间，遮住太
照到地球上的光，致使部分
完全看不见太阳的现象。❷
白天，跟"夜"相对：～班|～
场。❸天，一昼夜：阳历平年
一年三百六十五～。㉿某一
天：纪念～|生～。[日子]（—
zi）1. 天：这些～工作很忙。
2. 指某一天：今天是喜庆的
～。3. 生活：如今过上了好
～。❹一天天，每天：～益强
大|～新月异|～理万机。❺
时候

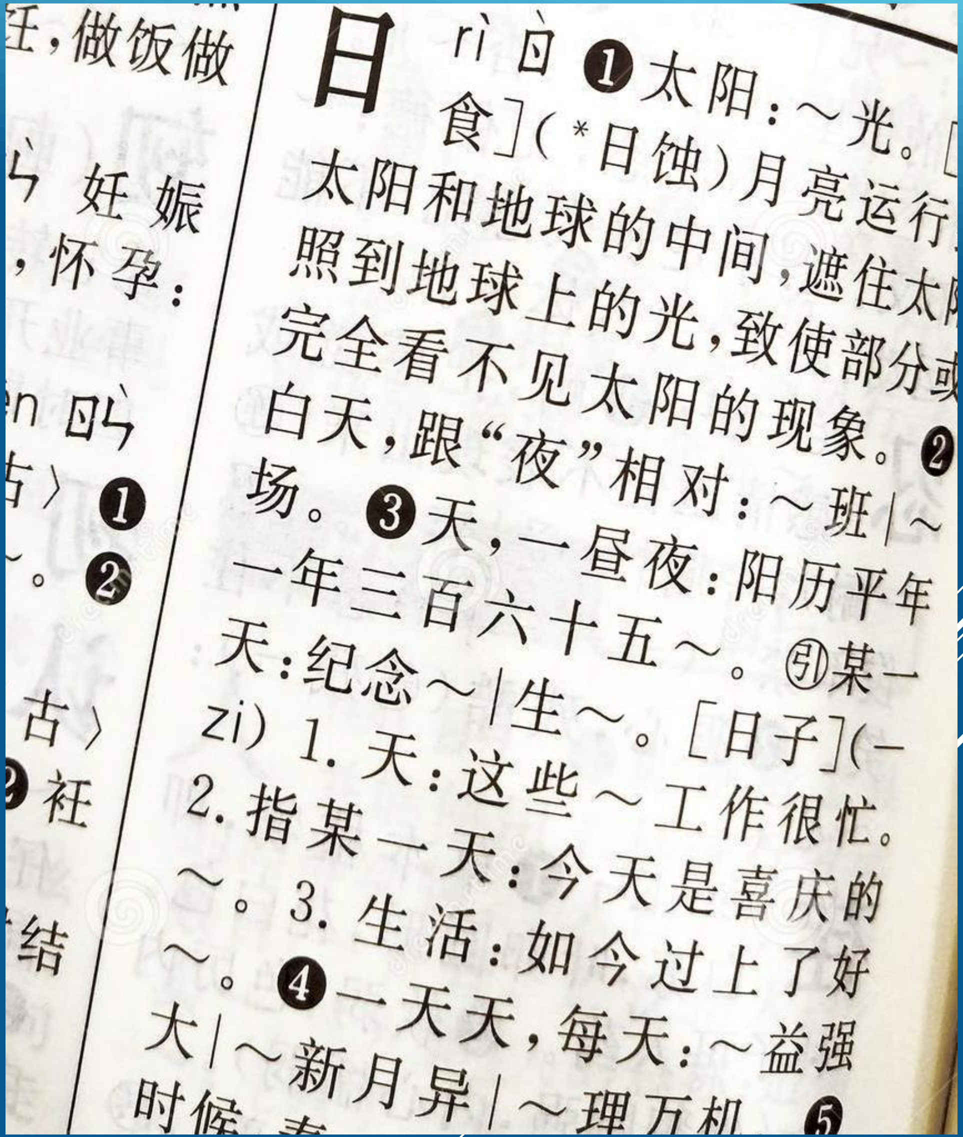
- try to learn Chinese language from Chinese-Chinese dictionary


日 rì ㄖˋ ❶太阳：～光。[日
食]（*日蚀）月亮运行
太阳和地球的中间，遮住太阳
照到地球上的光，致使部分或
完全看不见太阳的现象。❷
白天，跟"夜"相对：～班|～
场。❸天，一昼夜：阳历平年
一年三百六十五～。㊀某一
天：纪念～|生～。[日子]（—
zi）1. 天：这些～工作很忙。
2. 指某一天：今天是喜庆的
～。3. 生活：如今过上了好
～。❹一天天，每天：～益强
大|～新月异|～理万机。❺

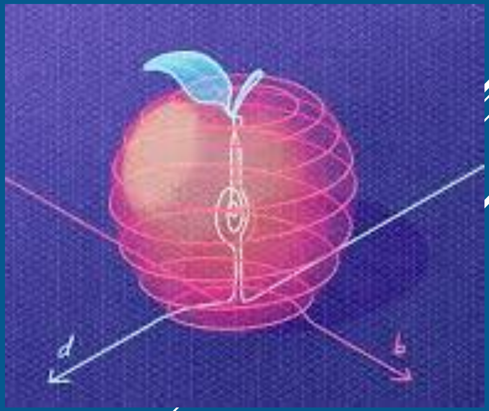
Google translate:
strip: "symbol grounding problem"
Dictionary page: Sun: ~light. Eclipse (solar eclipse) The moon moves between the sun and Earth, blocking the light that hits the Earth….

Go from one string of symbols without any meaning attached to the symbols to another string of symbols

符号接地问题


日 rì ㄖˋ ❶太阳：～光。
食](*日蚀)月亮运行
太阳和地球的中间，遮住太
照到地球上的光，致使部分或
完全看不见太阳的现象。❷
白天，跟“夜”相对：～班|～
场。❸天，一昼夜：阳历平年
一年三百六十五～。㊀某一
天：纪念～|生～。[日子](—
zi) 1. 天：这些～工作很忙。
2. 指某一天：今天是喜庆的
～。3. 生活：如今过上了好
～。❹一天天，每天：～益强
大|～新月异|～理万机。❺

- How can the abstract symbols of a computing system actually understand the external world?



Symbol Grounding Problem

Computer: deals with the **symbols as "shapes"** rather than their "**meaning**"

→How to make these **arbitrary symbols meaningful** without an external interpreter?

e.g., human providing context

Harnad:

- ground symbols with their **real-world** sensations and interactions, objects and actions they are linked to
- **provides meaning to symbols**

Putting philosophy aside, what are the **practical advantages** to grounding an AI or AGI project?

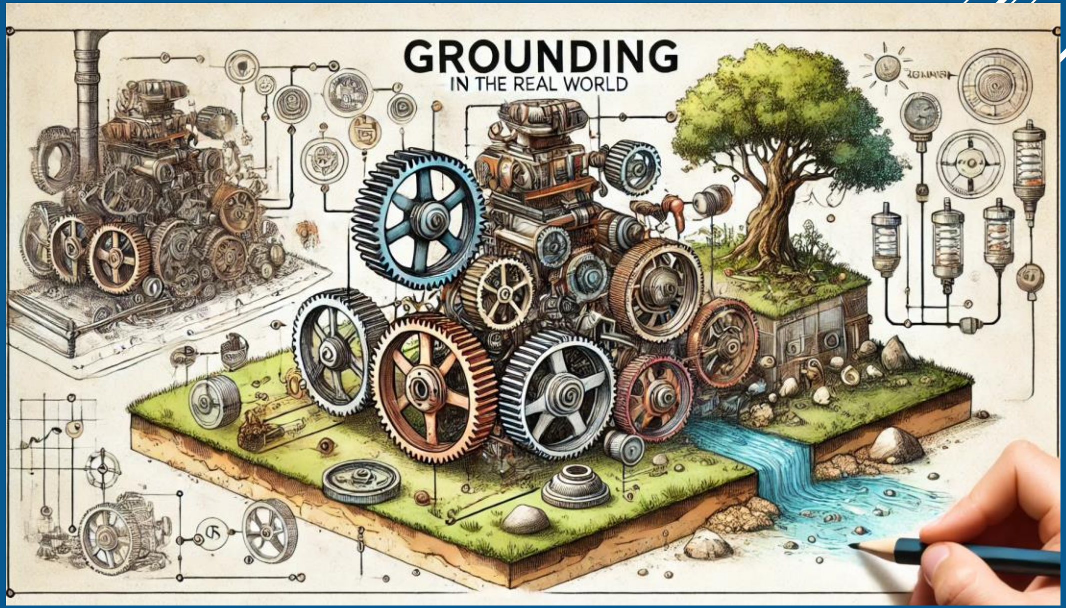


Image by DALL-E, 2024

- GROUNDING ADVANTAGES: AI/AGI can **understand** information with better context → **more accurate decisions**

Partial grounding only. Not full, complex, nuanced understanding for ChatGPT.
[vs GLAM via RL, Pavlick symbols and grounding LLM, etc.]

GROUNDING ADVANTAGE:

AI/AGI can **learn and adapt** from real-world experiences

- GROUNDING ADVANTAGES : AI/AGI can understand **real-world implications** of different actions → **more accurate decision/action**

GROUNDING ADVANTAGES:

AI/AGI can understand information with better context → **safer action, better alignment with human values possible**

GROUNDING ADVANTAGES : AI/AGI can anchor concepts in real-world experiences → can uses these reference points when encounter a new situation →**better generalizations**

GROUNDING → BETTER GENERALIZATIONS VIA:

- Reference points when encounter new situation
- Multimodal integration
- Reduce overfitting since exposure to diverse real-world data
- Better abstract the essential features from a situation

→ GROUNDING IS ADVANTAGEOUS

→THERE IS A **NEED FOR A ROBOTIC PLATFORM** FOR AN AI/ AGI/ BICA PROJECT

2. THE PETITCAT OPEN SOURCE ROBOTIC GROUNDING FOR AI/BICA PROJECTS

-Open Hardware

-Open Software

PETITCAT PROJECT

- Free, open-source software, GitHub
- Olivier Georgeon started, contributions by Schneider and others
- Interfaces an AI/ AGI/ BICA Python software project to a real-world embodiment
- **Allows you to ground your Python AI project**

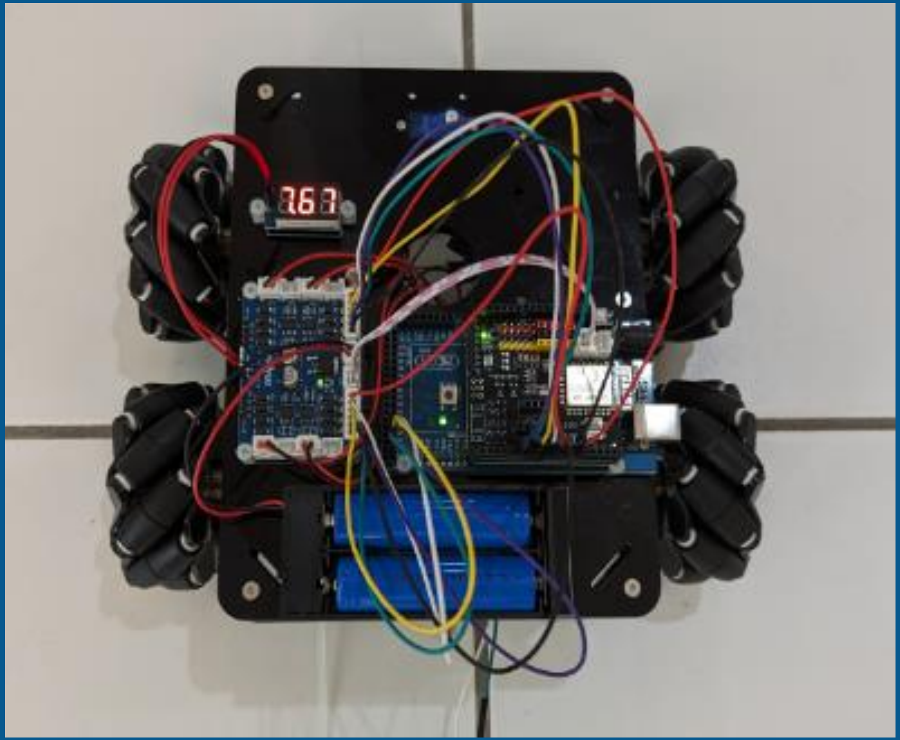
ROBOTS ARE EXPENSIVE (2024)

e.g., Boston Dynamics Spot
US$75,000 (2024)

PETITCAT PROJECT USES OPEN-SOURCE HARDWARE → LOW-COST ROBOTS POSSIBLE

open source hardware

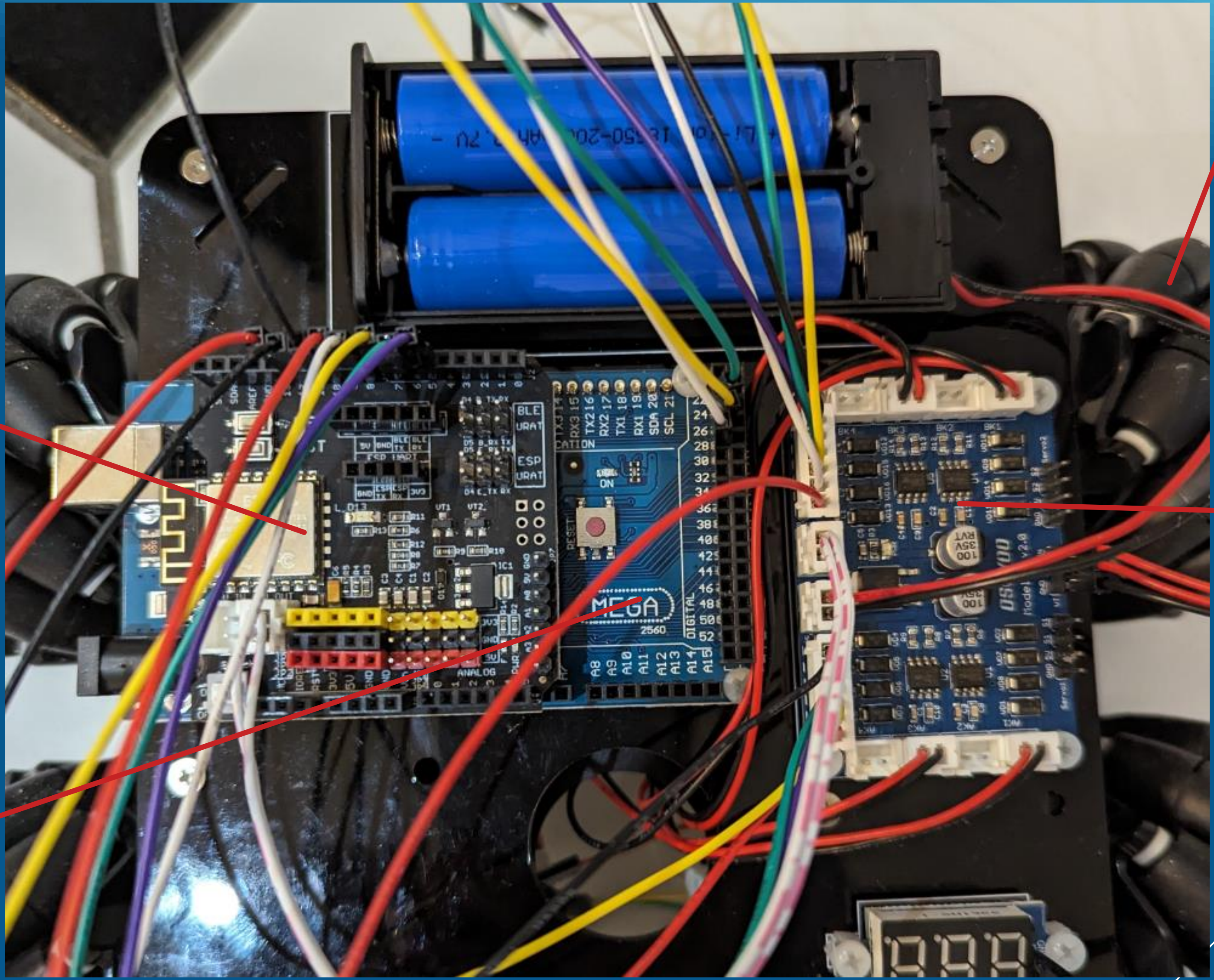
PETITCAT PROJECT USES OPEN-SOURCE HARDWARE -- ARDUINO/OSOYOO PLATFORM



Total costs
~ US$ 150

Wi-Fi Board

Arduino Microcontroller Board



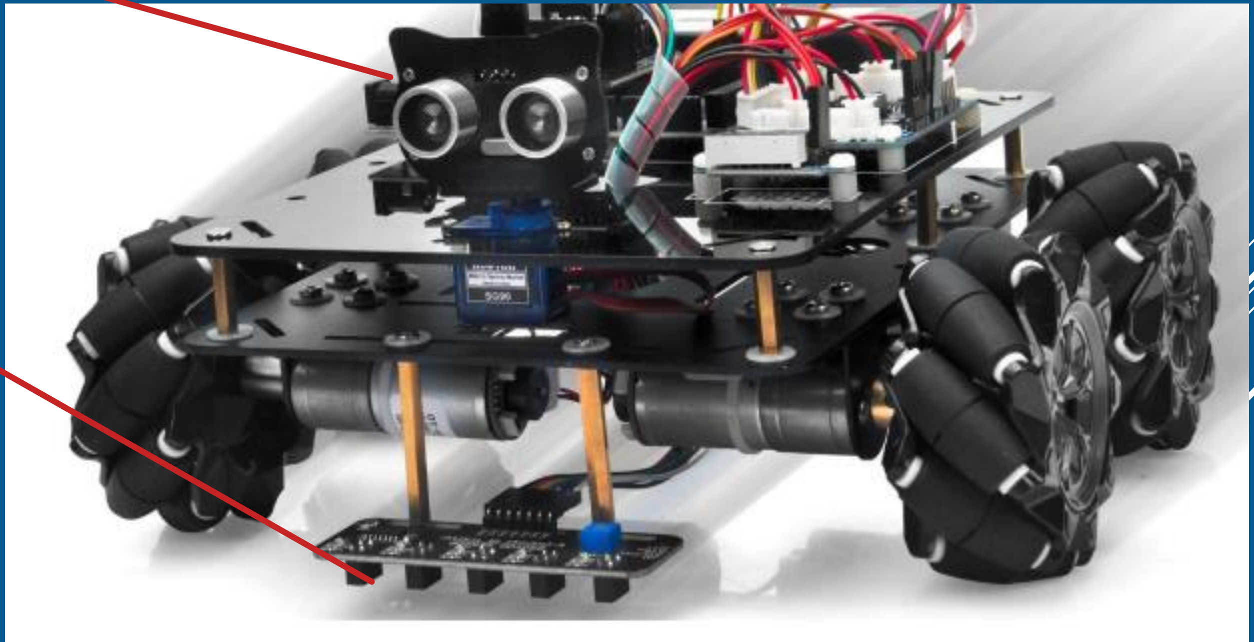
Mecanum Wheels, individual motors

Motor Driver Board

Ultrasonic sensors (moved by servomechanism)

Infrared sensors

EASY TO CUSTOMIZE WITH OTHER SENSORS & ACTUATORS



e.g., 9-axis inertial measurement unit, color sensor, emotion LED

ROBOT SOFTWARE IS COMPLEX TO LEARN AND TO IMPLEMENT

ROS

e.g., ROS2 Robot Operating System (framework)

nb.:
-ROS1 required Linux but ROS2 also allows Windows and macOS
-many sophisticated modules now, e.g., control multiple robots
-very secure communications now built into ROS2
-supports C/C++ and Python
-learning curve much higher, i.e., ROS2 time >> Arduino time

PETIT CAT SOFTWARE HAS A SHORT LEARNING CURVE

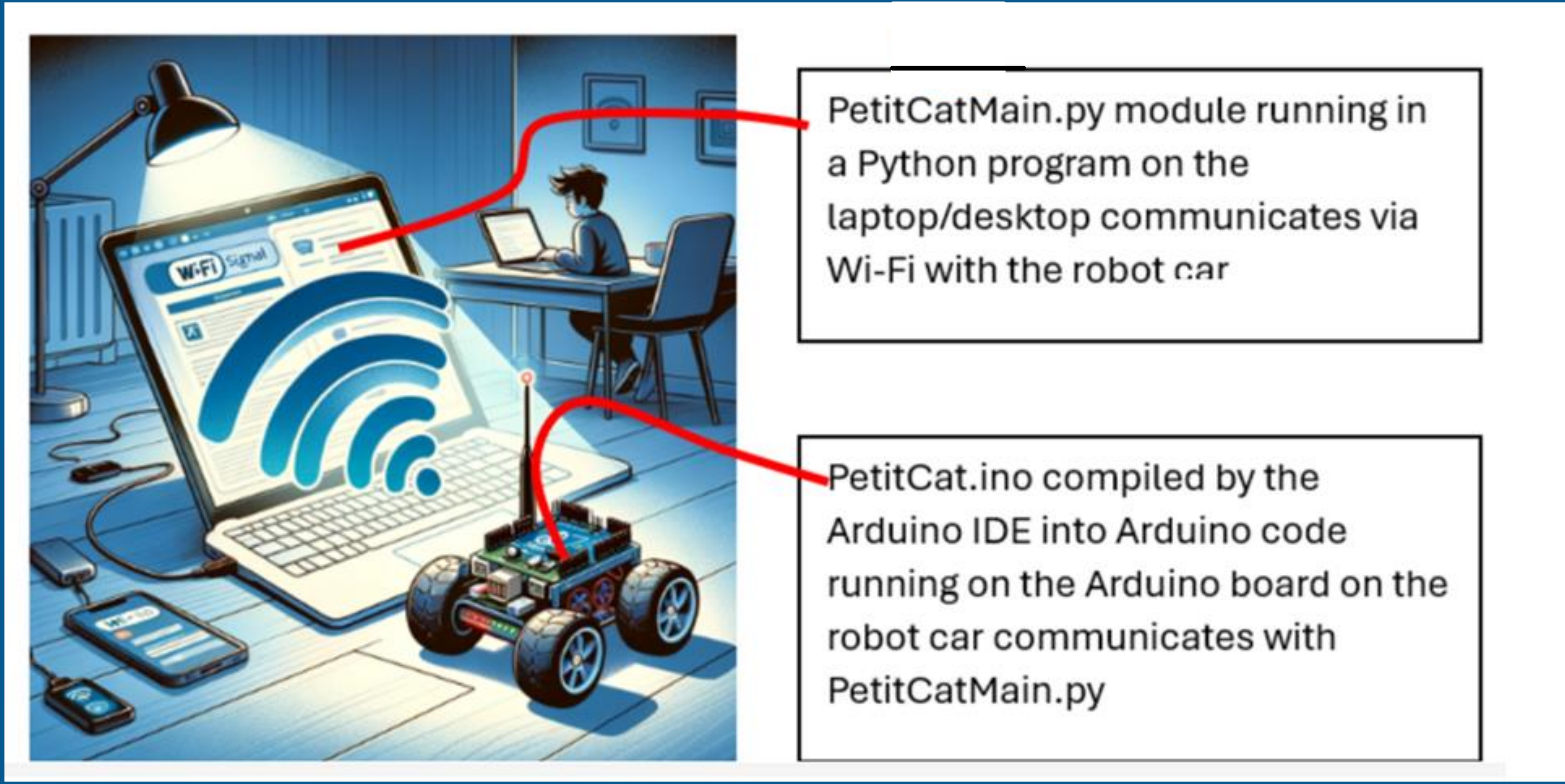
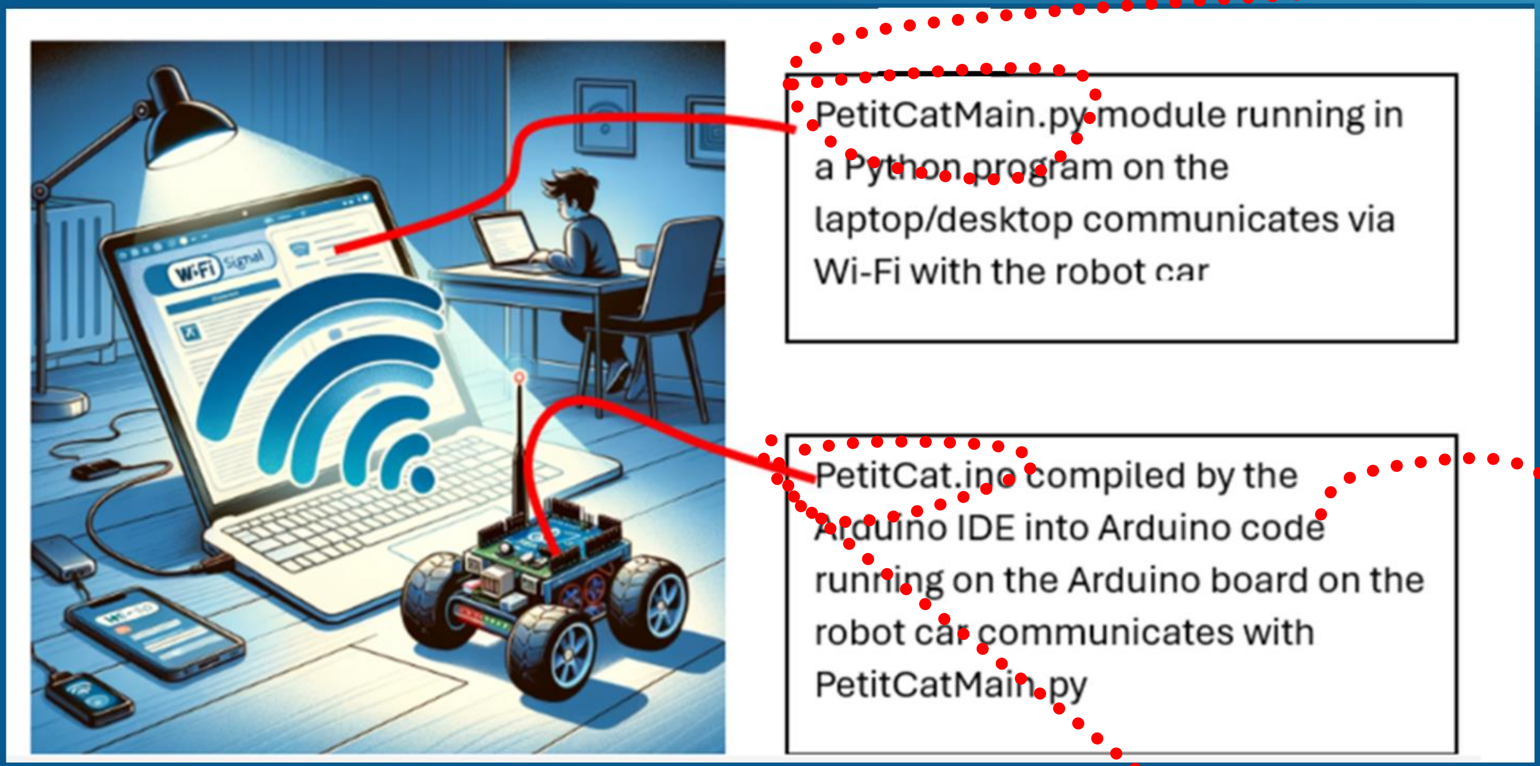


Image by DALL-E, 2024

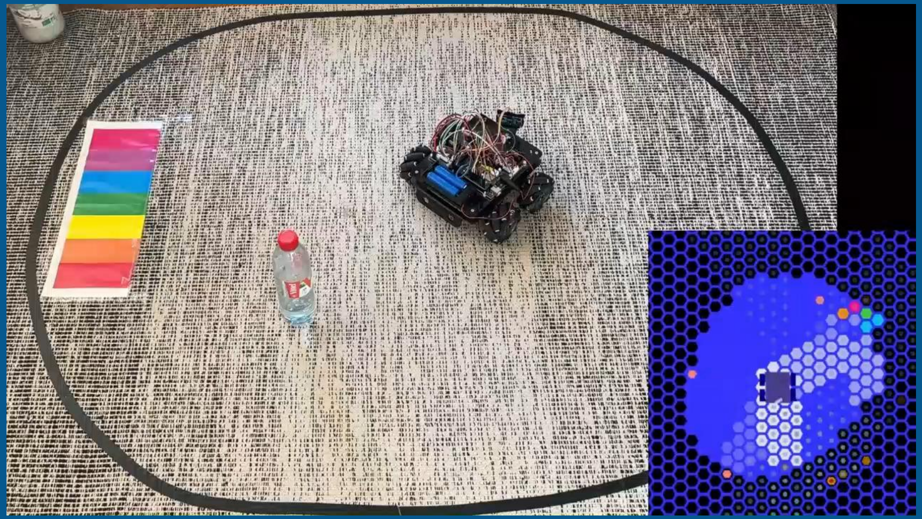


Python code of your AI/ AGI/ BICA project uses PetitCatMain.py to interface with the robotic embodiment

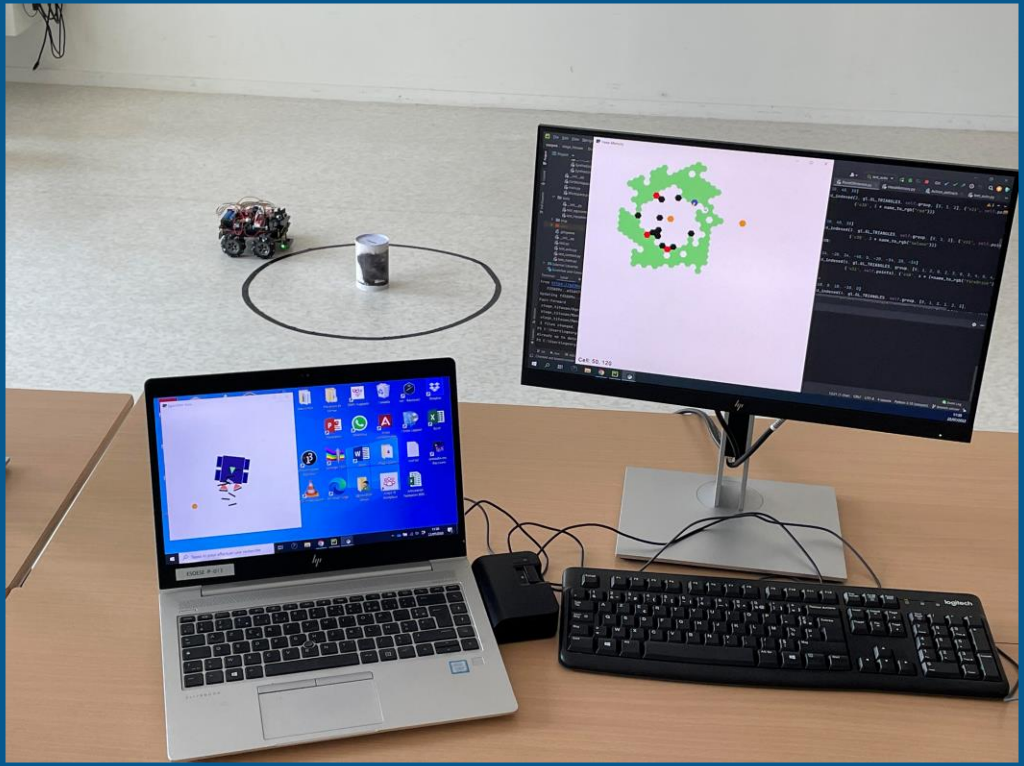
C/C++ code is compiled by Arduino IDE into **machine code** for over a hundred different Arduino and non-Arduino microcontrollers

You can use the existing PetitCat **C/C++ code** to run the embodiment or you can easily modify it

3. EXPERIMENTAL USE OF THE PETITCAT PROJECT



PETITCAT USE EXAMPLE



Implementation of enactive inference (Georgeon et al.)

Perceive world by mismatch of prediction of world and sensory input (action-perception loop)

Reduced computational load; real-time adaptation; more robust with noise; enhanced learning

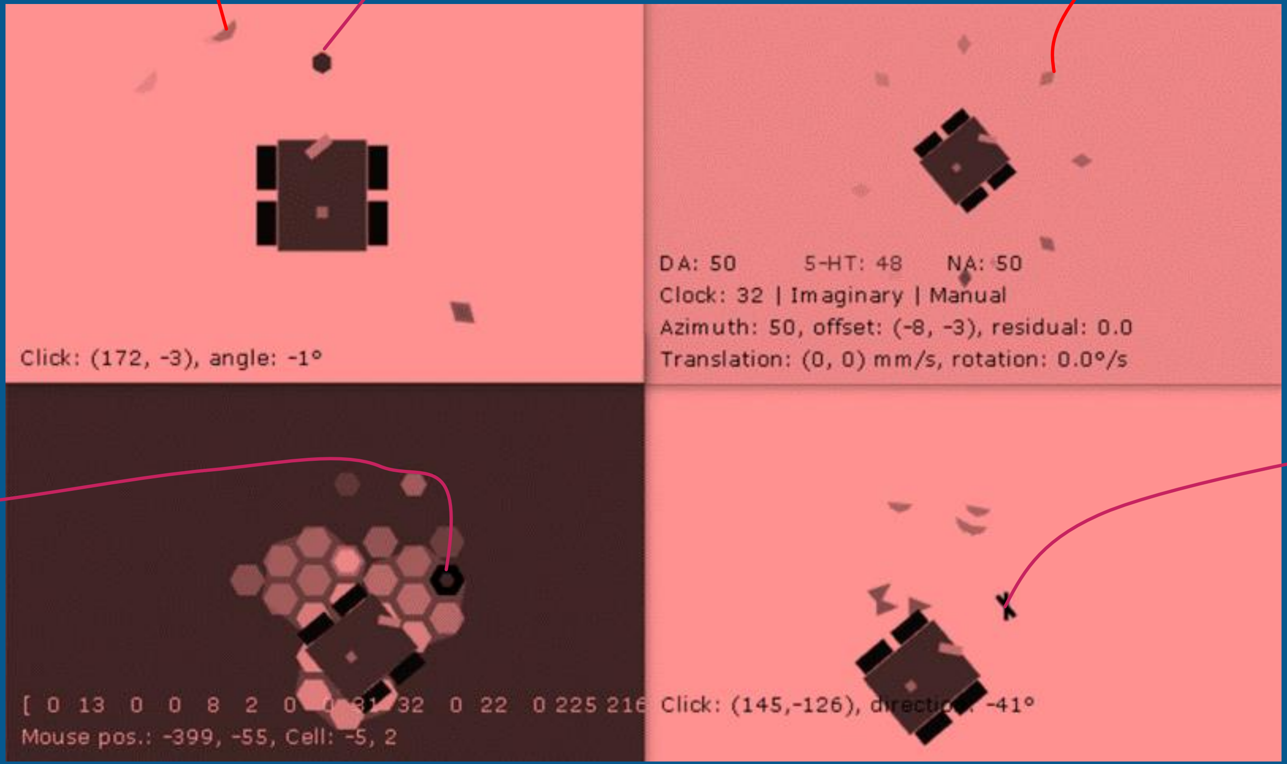
PetitCat Internal Simulator (screenshots)

Egocentric
Spatial Memory
(relative to agent's position)
e.g., hexagon in front of agent

Half-circle - echo

Hexagon – focus of attention

Agent
Spatial Memory
(compass points over time as robot turns around)



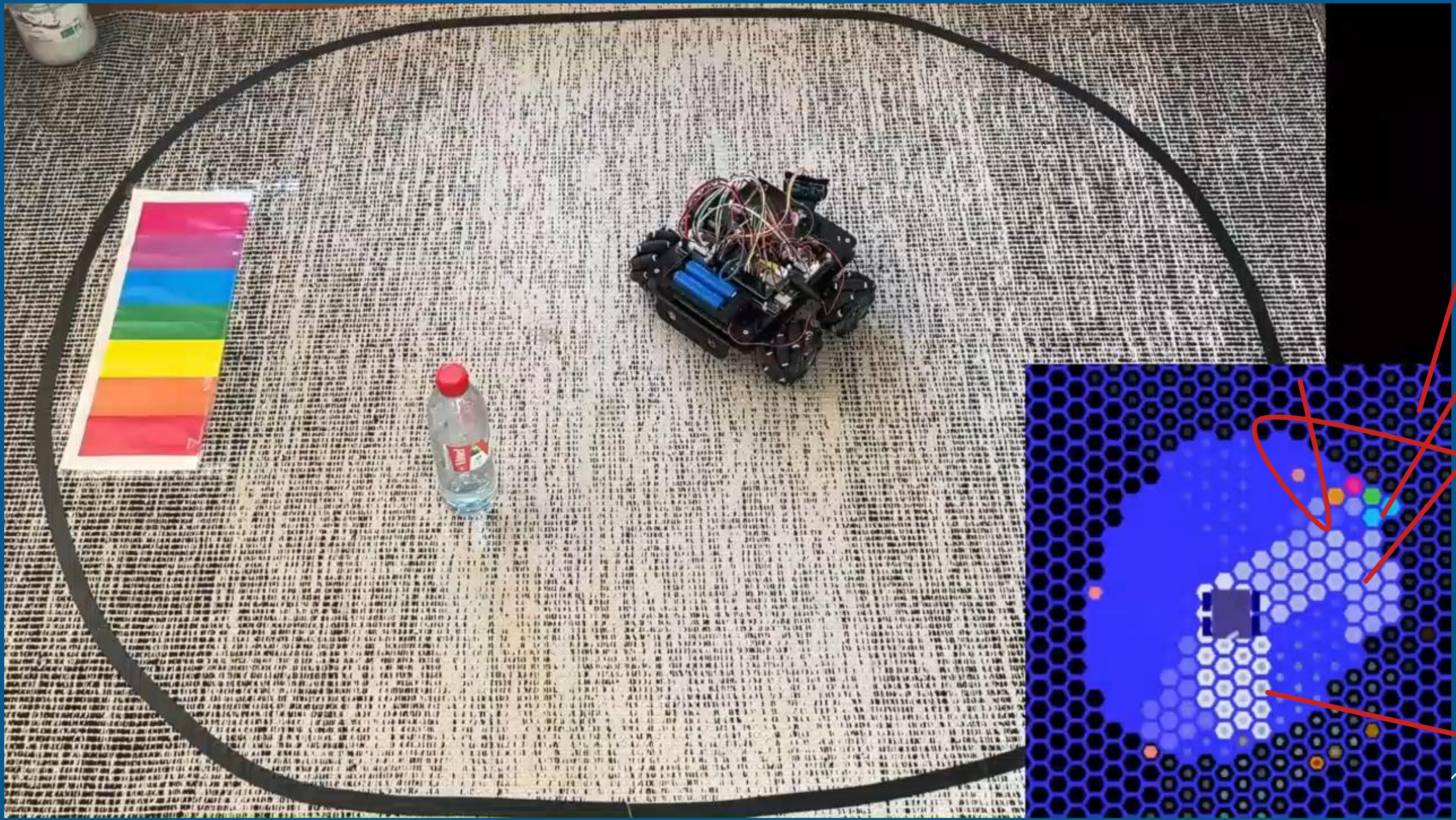
Black mark on the floor

Interactions with the black mark on the floor

Allocentric
Spatial Memory
(bird's-eye view)
i.e., navigate from any perspective

Object-Centric
Spatial Memory
(relations between objects, not agent)
e.g., black mark vs. other echos

VIDEO: ENACTIVE INFERENCE



Hexagons represent grid cells (darker ones not used yet)

Floor colors detected by color sensor

Default floor color

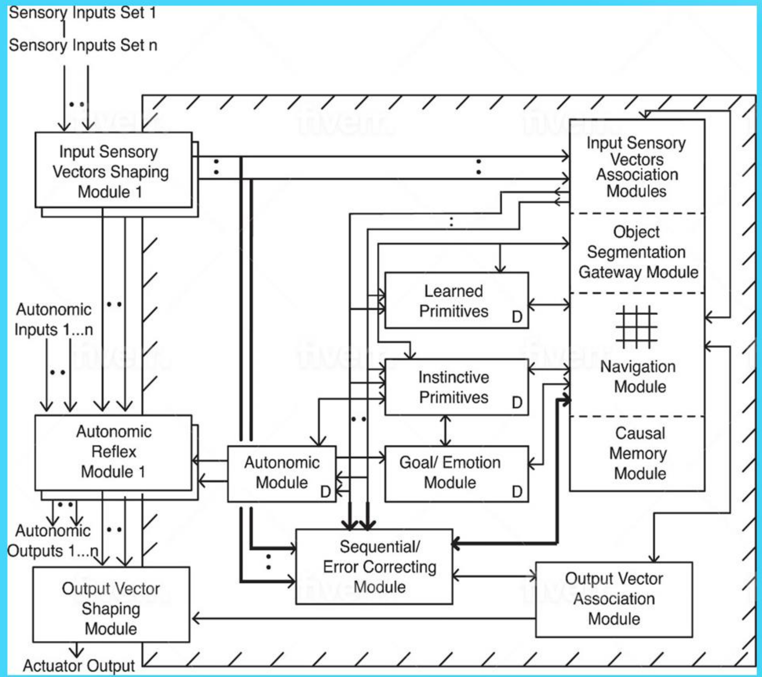
Bottle's echo here as orange dot (video)

White hexagons – place recently (last 10 steps) covered by agent

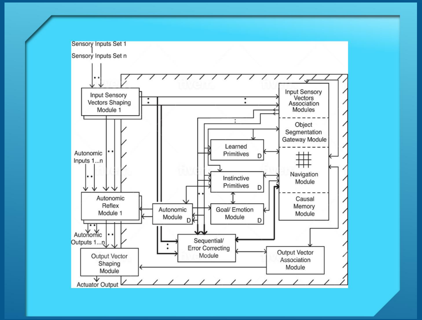
OTHER PROJECTS USING PETIT CAT FOR EMBODIMENT

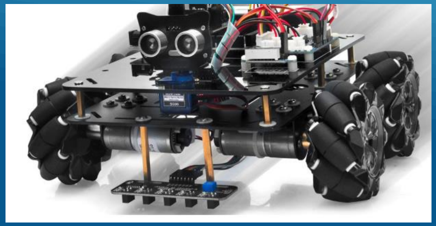
- Gay – bio-inspired spatial navigation

- Schneider – interface to a BICA (Causal Cognitive Architecture)



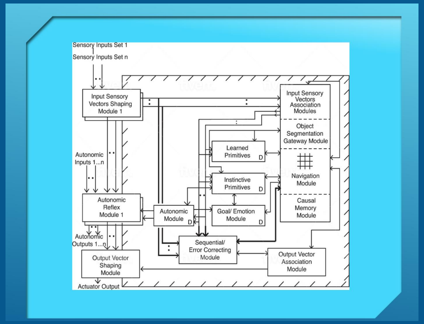
• Schneider – BICA + LLM + PetitCat

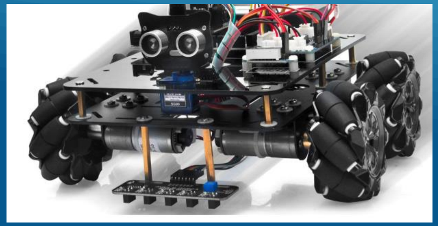




```
The echos values show readings at various angles, with the highest at -60 and 90, which
ecting objects at these angles. The status is set to 0, which could potentially mean tha

Overall, this data suggests that the mobile robot is currently stationary and potentiall
r measurement task. It is actively detecting objects at different angles and has been ru
me.

llm motor code produced:

8 (go forwards)

motor_code to be sent to the robotic car: 8

Motor Command Response: b'{"clock":34,"action":"8","duration1":977,"head_angle":30,"echo
1872,"status":"1"}'
```

- Schneider – BICA + LLM + PetitCat





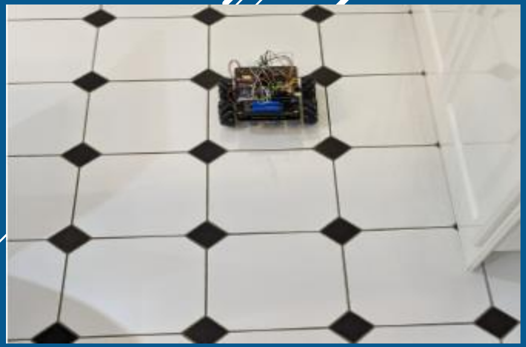
Prompt:

PROMPT2 = ("Based on the previous sensory information please now decide if you want the robotic car to go forwards, backwards, left, right or to even shuffle to the right or shuffle to the left. Make the best decision to avoid hitting any objects detected by ultrasound sensors. The following motor codes are possible: '1' means turn left, '2' which means go backwards, '3' means turn right, '4' means shuffle to the right, '6' means shuffle to the left, '8' means go forwards. Please just output a single digit, nothing else. "

Python Code:

```
for trial in range(TOTAL_TRIALS):
    data = controller.motor_command(SONIC_SCAN).decode('utf-8'))
    prompt1 = "Analyze mobile robot sonic scan: " + data
    print(llm.invoke(input=prompt1))
    llmresponse = llm.invoke(input=PROMPT2)
    controller.motor_command(extract_first_digit(llmresponse))
```

-Actual code ~1000 lines
-Amazing results with modest coding



Free, open-source project, accessible via GitHub

https://github.com/OlivierGeorgeon/osoyoo
https://github.com/UCLy/INIT2