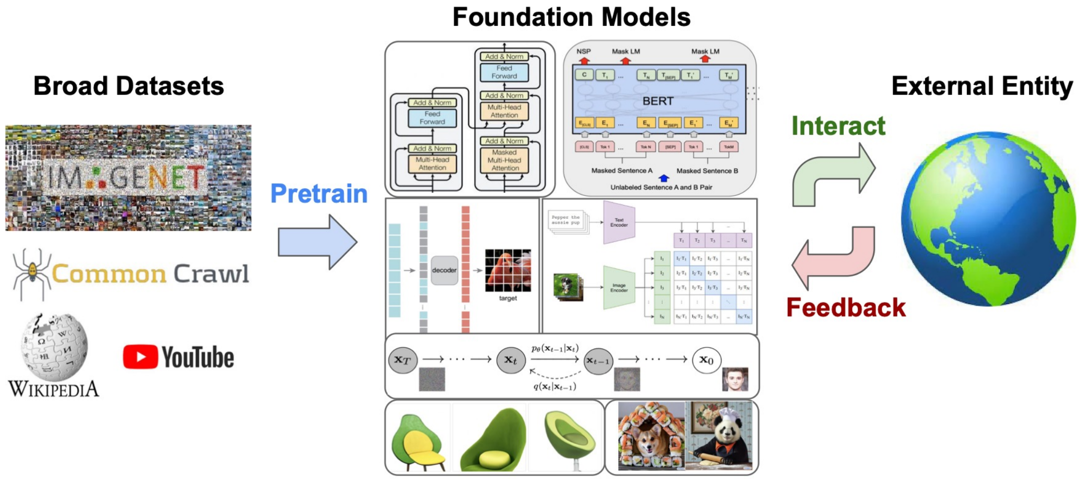
Microsoft

FOUNDATION MODELS FOR AUTONOMOUS SYSTEMS

Sai Vemprala
Senior Researcher
Microsoft Autonomous Systems and Robotics Research

VISION

- Pretraining and foundation models have seen immense success in language and computer vision.
- What are foundation models?
  "A neural network trained on mountains of raw data, generally with unsupervised learning — that can be adapted to accomplish a broad range of tasks." [1]

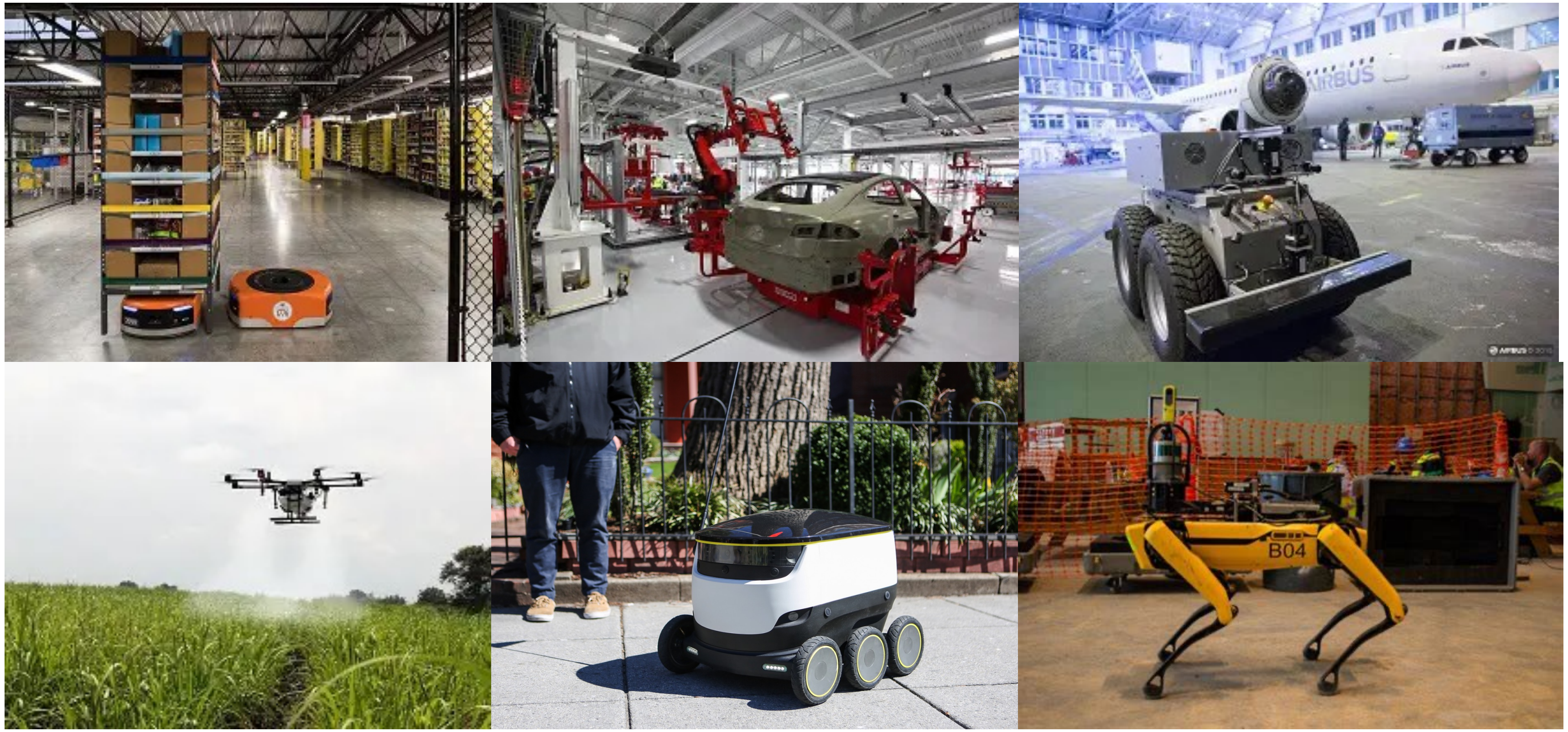


[1] Bommasani, Rishi, et al. "On the opportunities and risks of foundation models." arXiv preprint arXiv:2108.07258 (2021).

[2] Figure from Yang, Sherry, et al. "Foundation Models for Decision Making: Problems, Methods, and Opportunities." *arXiv preprint arXiv:2303.04129* (2023).

VISION

- Robotics – multiple form factors, long tail scenarios, out of distribution data.
- Scalability issues due to handcrafted, task-specific modules.



WHAT DO WE NEED?

How can we build robotics foundation models?

- Human/expert demonstrations
- Simulated/synthetic data
- Pretrained models

What does a robotics foundation model need to ‘understand’?

- Geometry and semantics
- Dynamics
- State-observation-action mapping

What does a robotics foundation model need to be able to ‘do’?

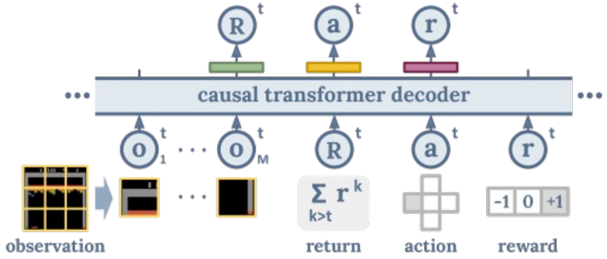
- Recognition / Mapping
- Planning / Reasoning / Safety
- Human interaction (e.g. Language)

RELEVANT WORK

- Perception-Action Representation Learning
  - PACT: Perception-Action Causal Transformer for Autoregressive Robotics Pre-Training
  - ConBaT: Control Barrier Transformer for Safe Policy Learning
- Language + Robotics
  - LATTE: LAnguage Trajectory TransformEr
  - ChatGPT for Robotics (microsoft.com)

REPRESENTATION LEARNING FOR ROBOTICS

- GPT learns notions of grammar and semantics from large corpuses of text.
- If a robot can fully understand state-action transitions, this leads to a high-quality mental model of the robot-world interactions.
- Idea first explored by Decision Transformer [3], but with a notion of task reward.

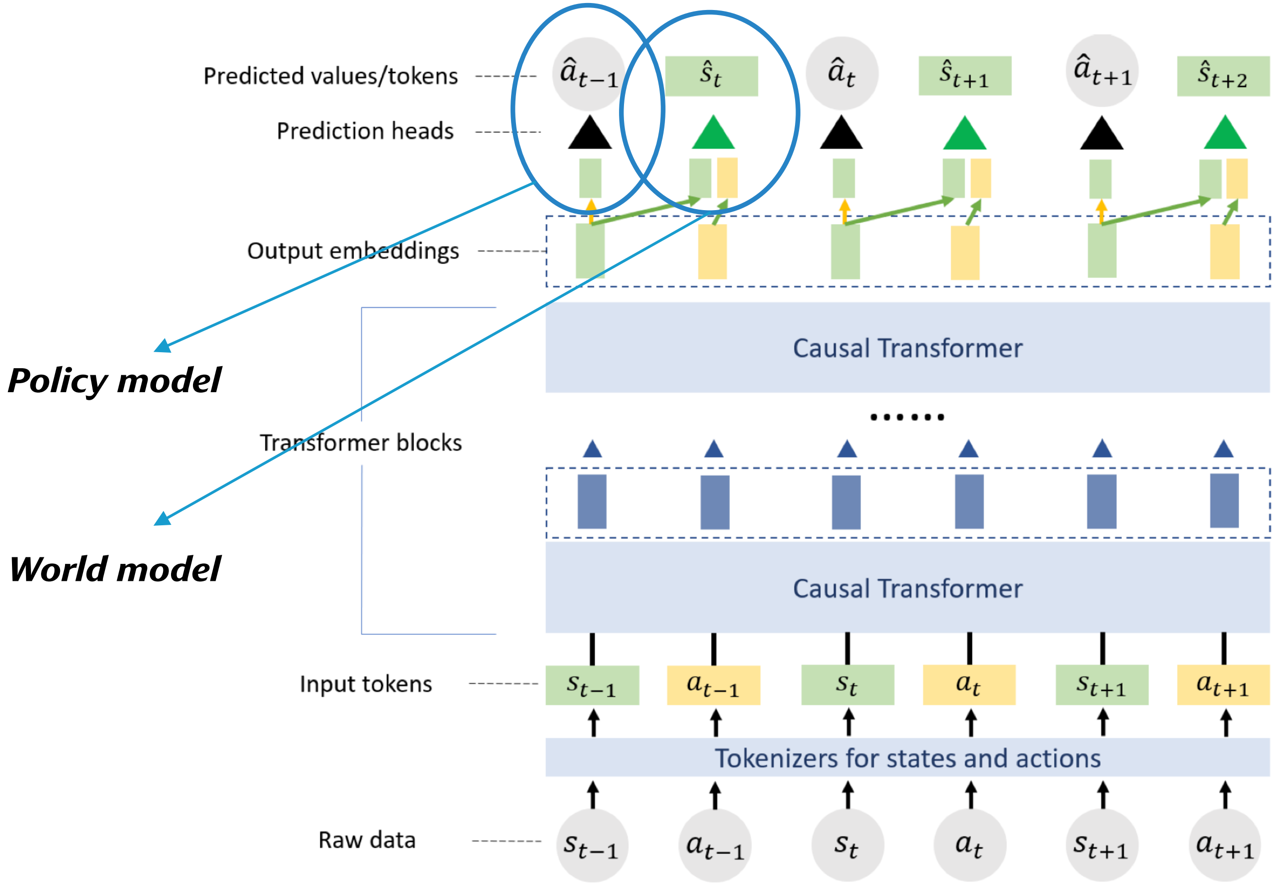


- PACT: Autoregressive Transformer architecture that trains on task-agnostic sequences of states (observations) and actions.

[3] Chen, Lili, et al. "Decision transformer: Reinforcement learning via sequence modeling." *Advances in neural information processing systems* 34 (2021): 15084-15097.

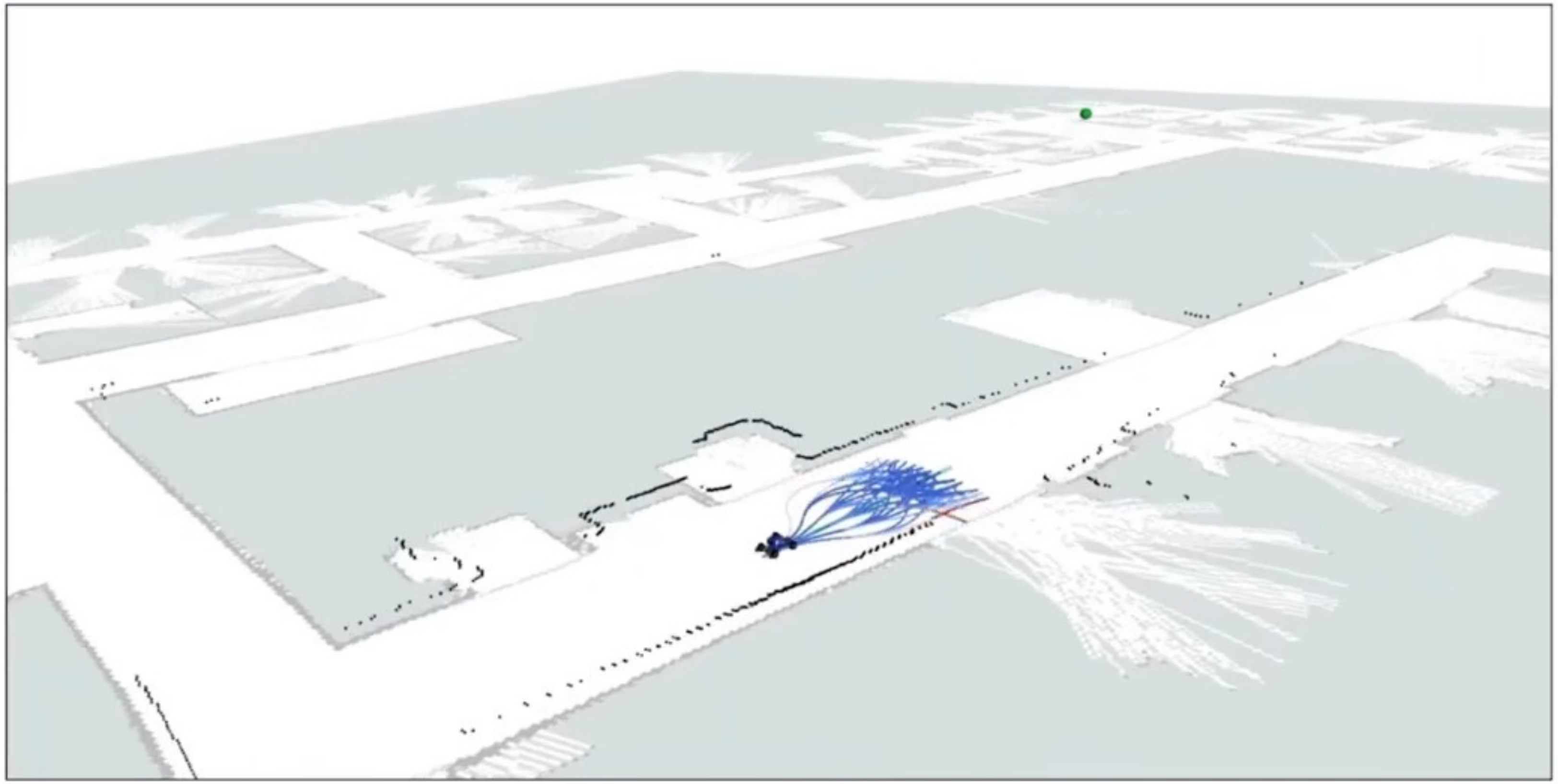
PACT: PERCEPTION-ACTION CAUSAL TRANSFORMER

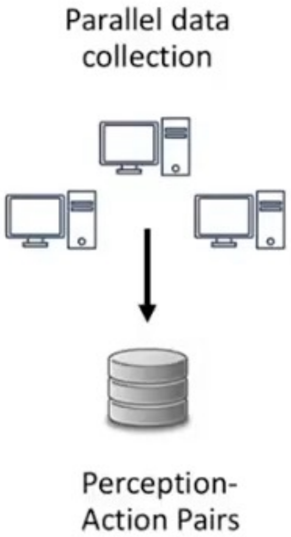
- Tokenize raw observations and actions.
- N transformer blocks operate on this sequence.
- Self supervised pretraining with two objectives
  - Action prediction
  - State embedding prediction



PACT: TRAINING DATA

Pre-training data collection for MuSHR: millions of perception-action pairs

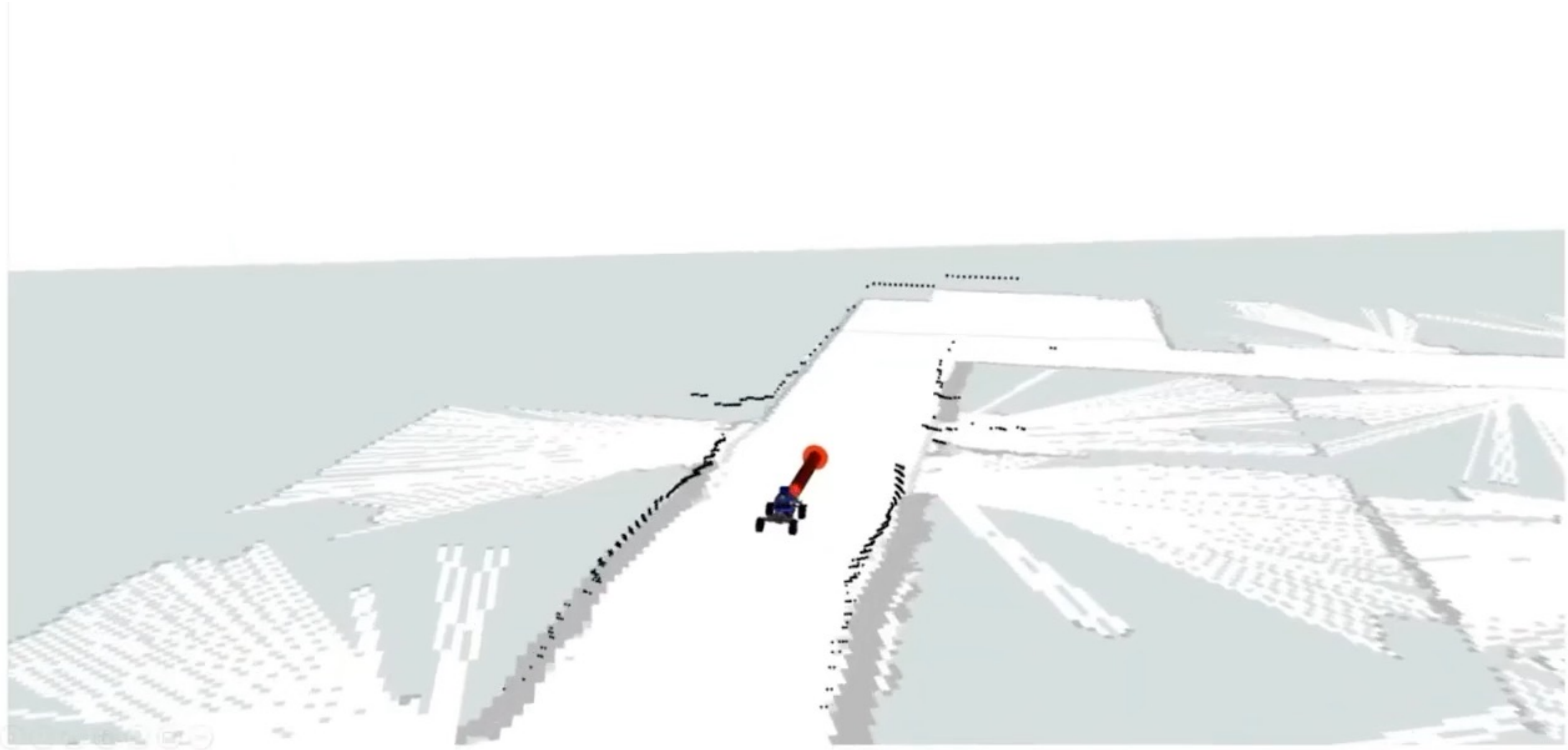




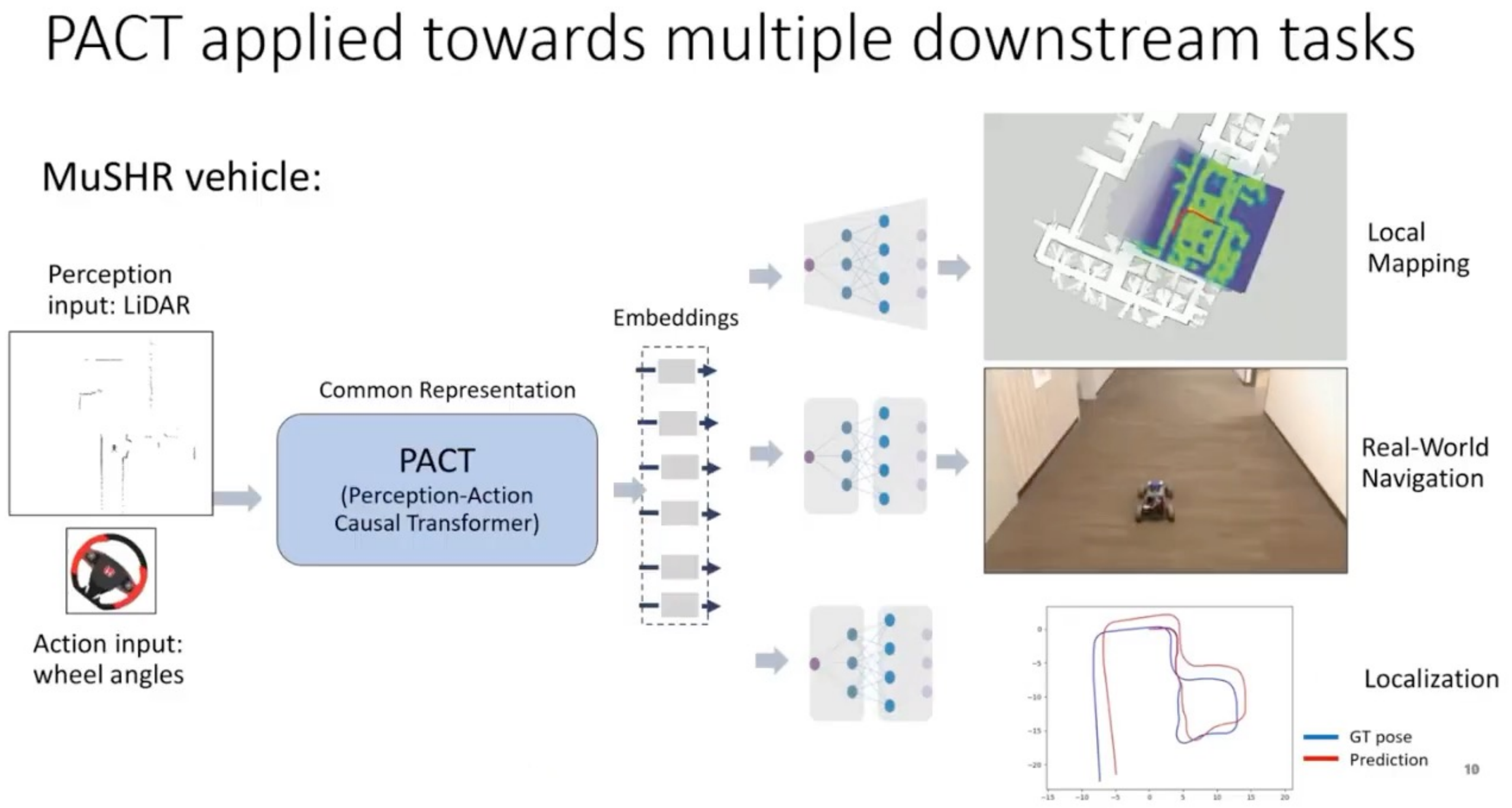
Parallel data collection

Perception-Action Pairs

PACT: RESULTANT POLICIES



PACT: DOWNSTREAM TASKS



PACT applied towards multiple downstream tasks

MuSHR vehicle:

Perception input: LiDAR

Action input: wheel angles

Common Representation

PACT
(Perception-Action Causal Transformer)

Embeddings

Local Mapping

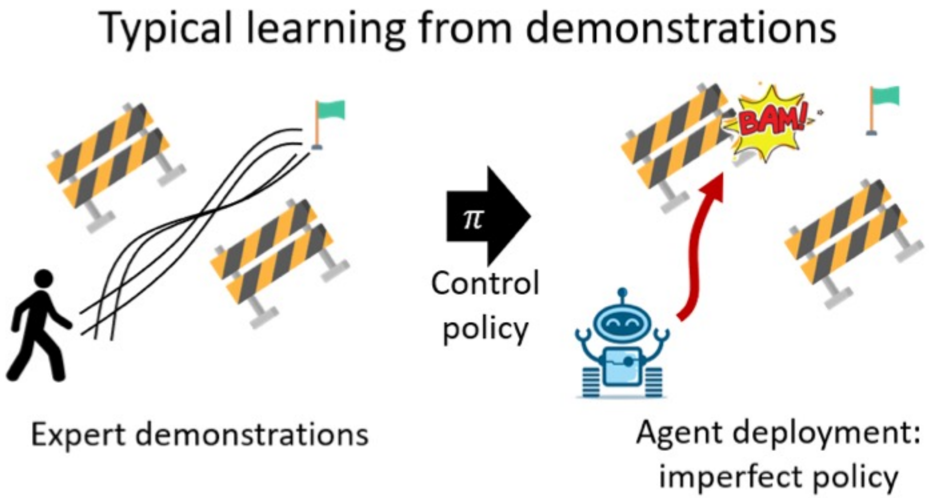
Real-World Navigation

Localization

GT pose

Prediction

TOWARDS NOTIONS OF SAFETY

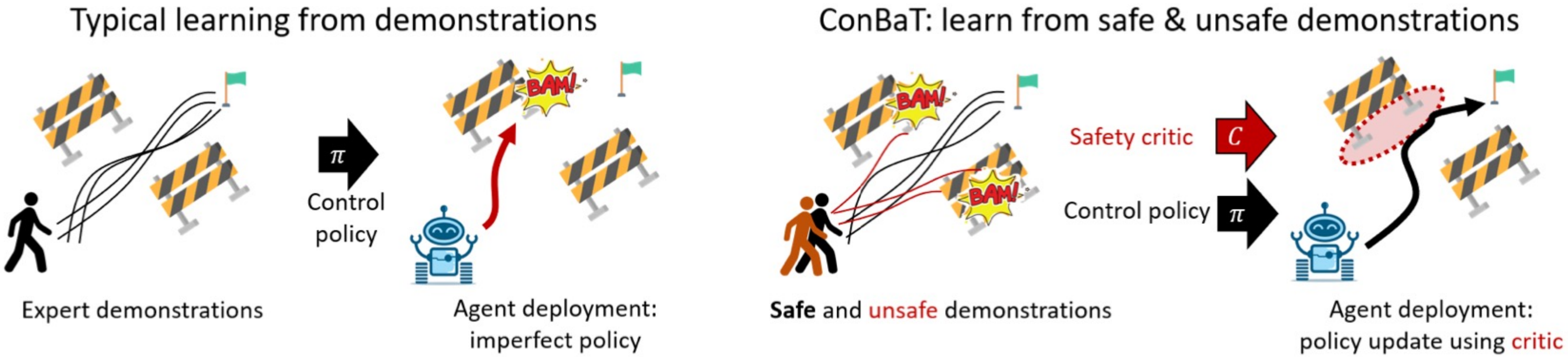

Typical learning from demonstrations
BAM!
π
Control policy
Expert demonstrations
Agent deployment: imperfect policy

How to handle safety constraints?

Classical methods: Hand tuned parameters, expensive optimization problems

Reinforcement learning: Reward shaping efforts

TOWARDS NOTIONS OF SAFETY

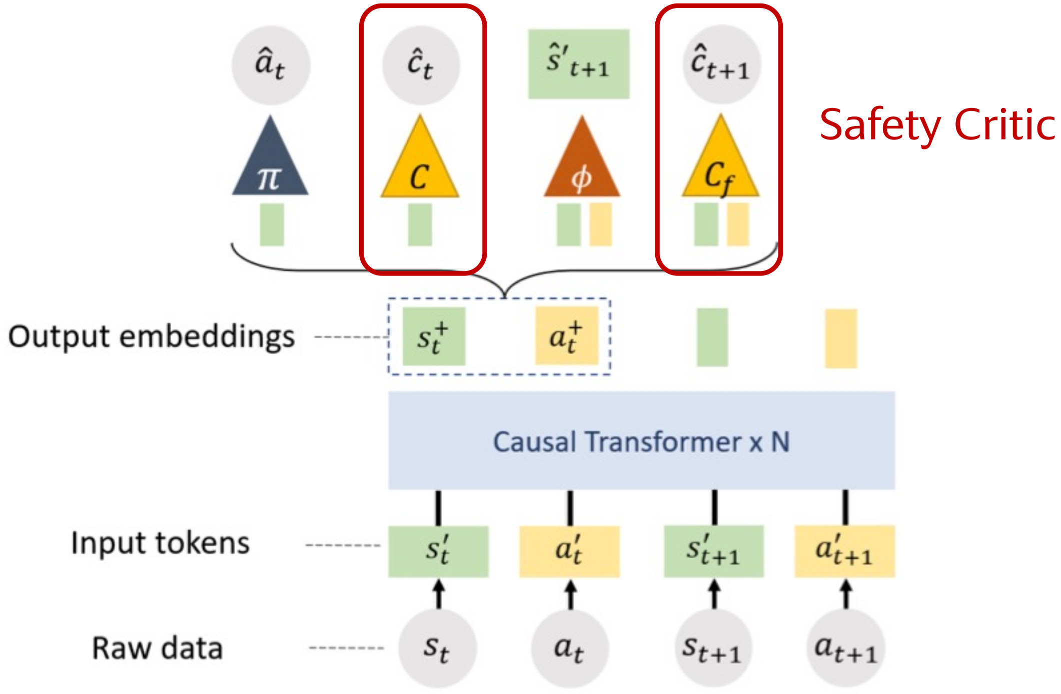

Typical learning from demonstrations

ConBaT: learn from safe & unsafe demonstrations

BAM!
π
Control policy
Expert demonstrations
Agent deployment: imperfect policy
BAM!
BAM!
Safety critic
C
Control policy
π
Safe and unsafe demonstrations
Agent deployment: policy update using critic

CONBAT: CONTROL BARRIER TRANSFORMER

Pretraining phase



Critic classification loss

$$\mathcal{L}_c = \mathop{\mathbb{E}}_{s_t^+ \sim \tilde{\mathcal{S}}_s^+} \left[\sigma_+ \left(\gamma - C(s_t^+)\right)\right] + \mathop{\mathbb{E}}_{s_t^+ \sim \tilde{\mathcal{S}}_u^+} \left[\sigma_+ \left(\gamma + C(s_t^+)\right)\right]$$

Critic smoothness loss

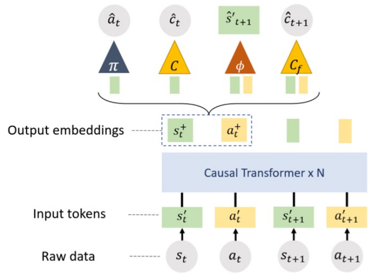
$$\mathcal{L}_s = \mathop{\mathbb{E}}_{s_t^+ \sim \tilde{\mathcal{S}}^+} \left[\sigma_+ \left((1-\alpha)C(s_t^+) - C(s_{t+1}^+)\right)\right]$$

Critic consistency loss

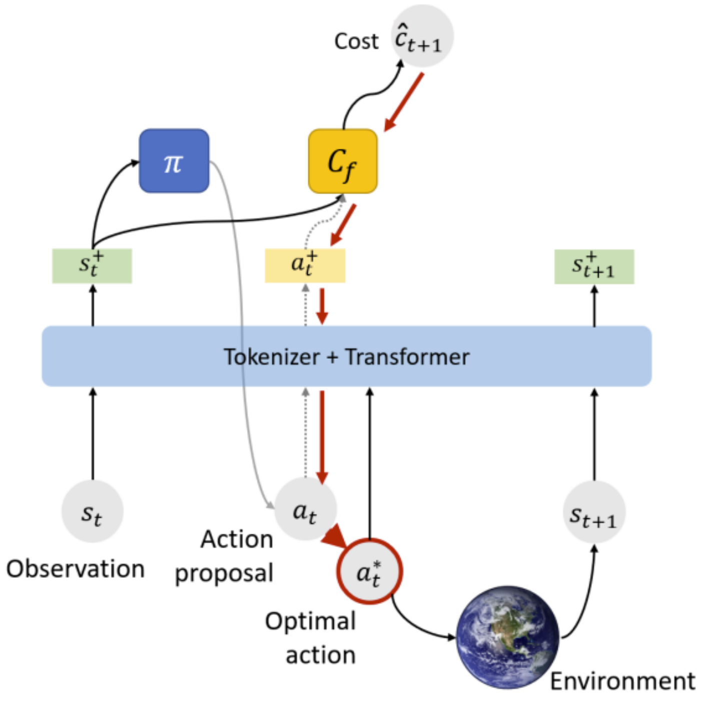
$$\mathcal{L}_f = \mathop{\mathbb{E}}_{s_t^+ \sim \tilde{\mathcal{S}}^+} \left[\left|C_f(s_t^+, a_t^+) - C(s_{t+1}^+)\right|\right]$$

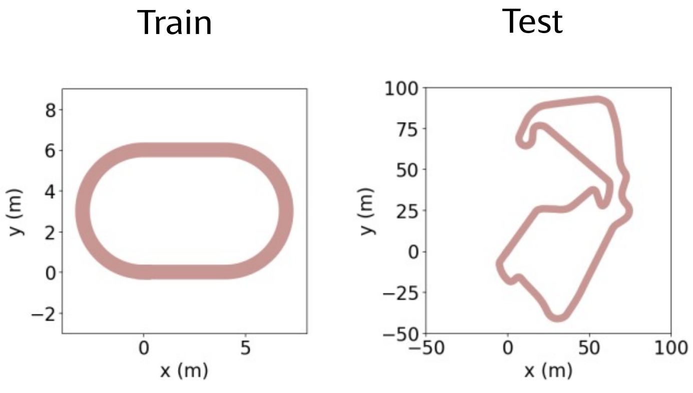
CONBAT: CONTROL BARRIER TRANSFORMER

Pretraining phase

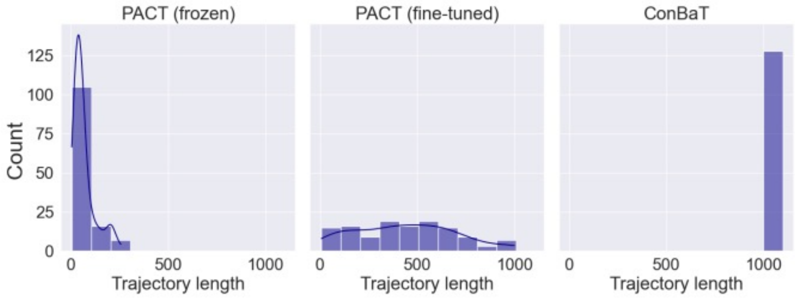


Deployment phase



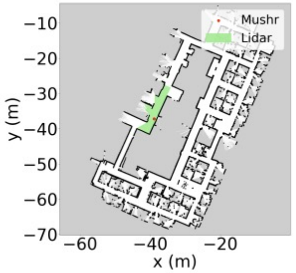


(a) F1/10 (playground) (b) F1/10 (Silverstone)

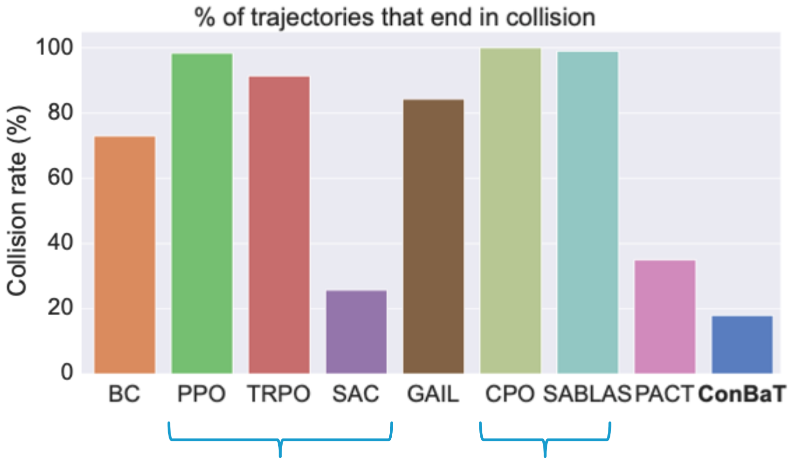


$$\Delta a^* = \underset{\Delta a}{argmin}\, \lambda ||\Delta a|| + \max(-C_f(s_t^+, \hat{a}_t^+ + \Delta a), 0)$$

CONBAT: CONTROL BARRIER TRANSFORMER

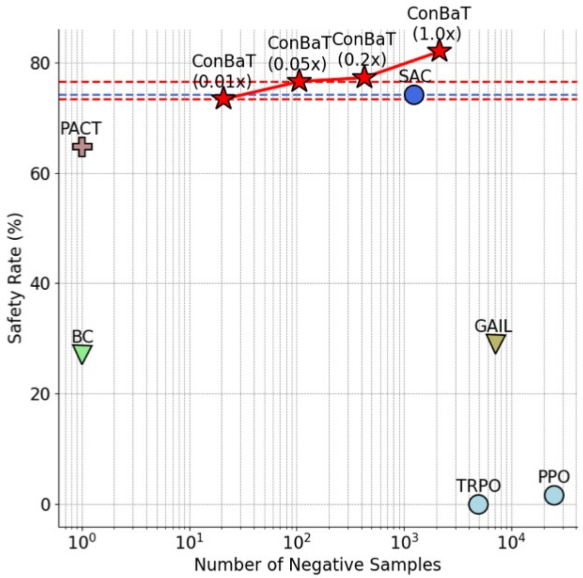


- Office environment with MuSHR car
- LiDAR observation – 720-dim observation space
- Compared with RL, IL, safe learning baselines.



RL

Safe policy learning



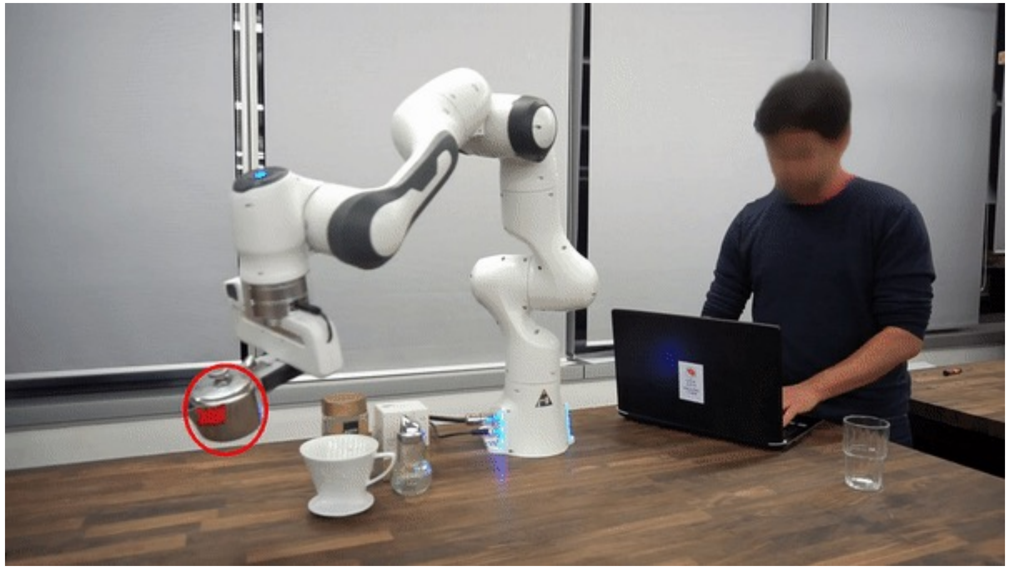
LANGUAGE TO PROGRAM ROBOTS

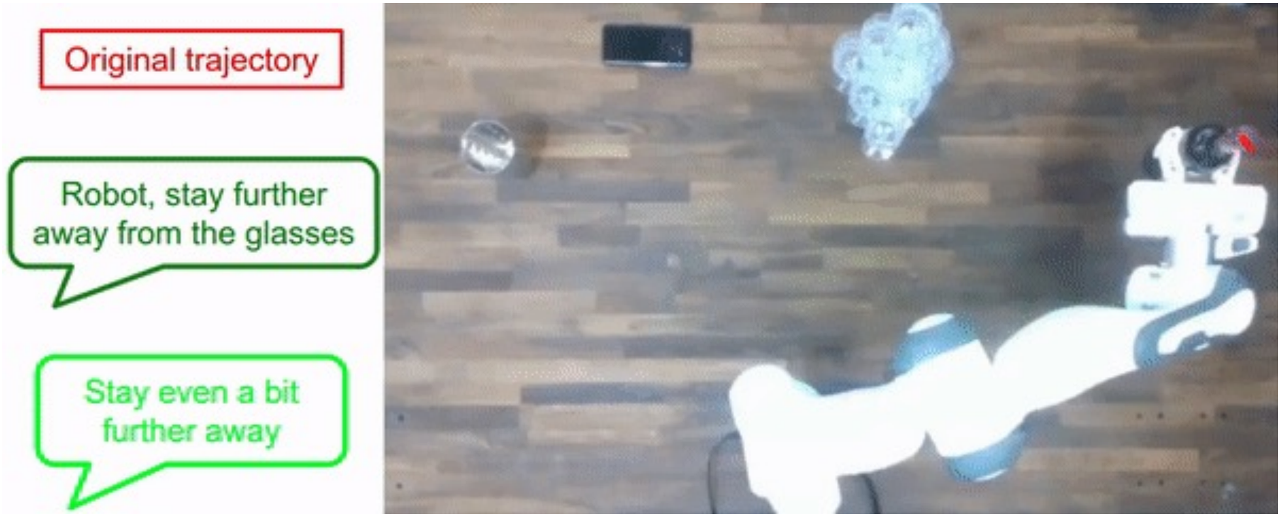
- Task-specific modules need to be re-designed by an expert even if there are minor changes in robot hardware, environment, or operational objectives.
- Language is the most natural interface to indicate changes and constraints.
- Can we leverage information in large pretrained vision-language models?

- Requires combining language, vision, and geometry into a single model.

LATTE: LANGUAGE TRAJECTORY TRANSFORMER

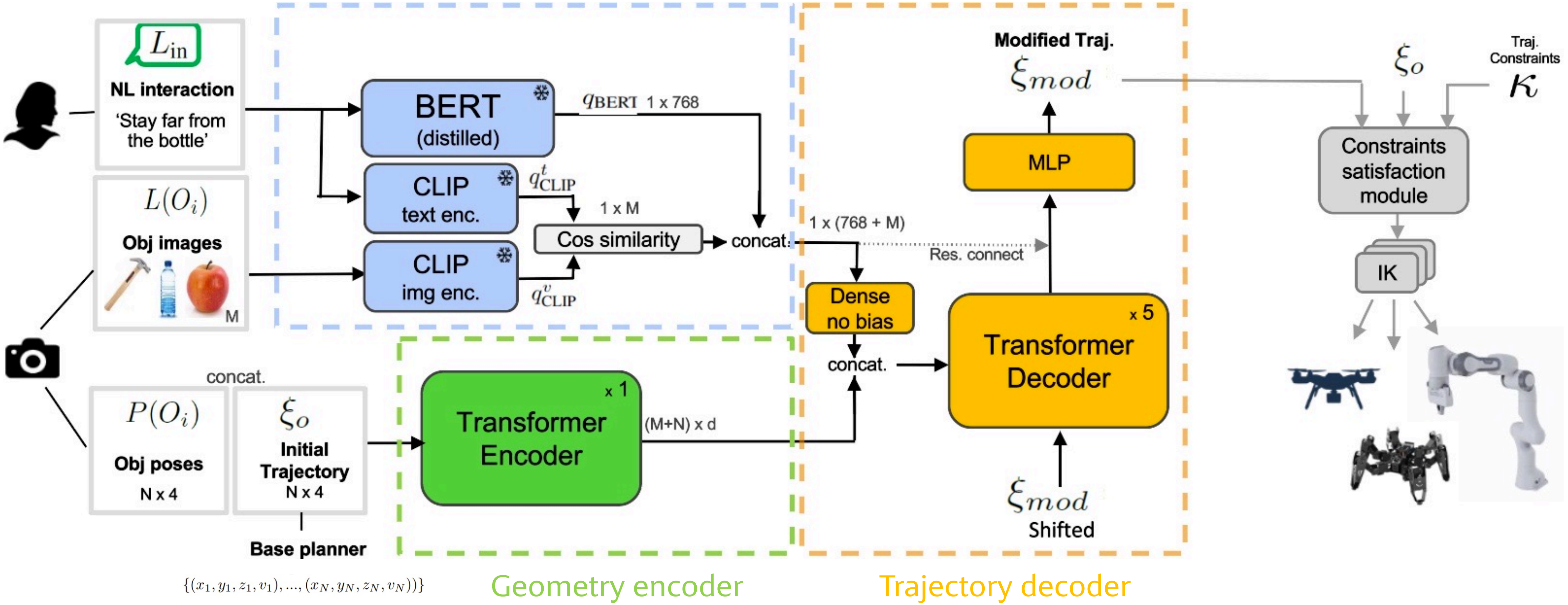
- Flexible interface for human-robot interaction within the context of trajectory reshaping that is agnostic to robotic platforms.




Original trajectory
Robot, stay further away from the glasses
Stay even a bit further away

LATTE: LANGUAGE TRAJECTORY TRANSFORMER

Feature encoder



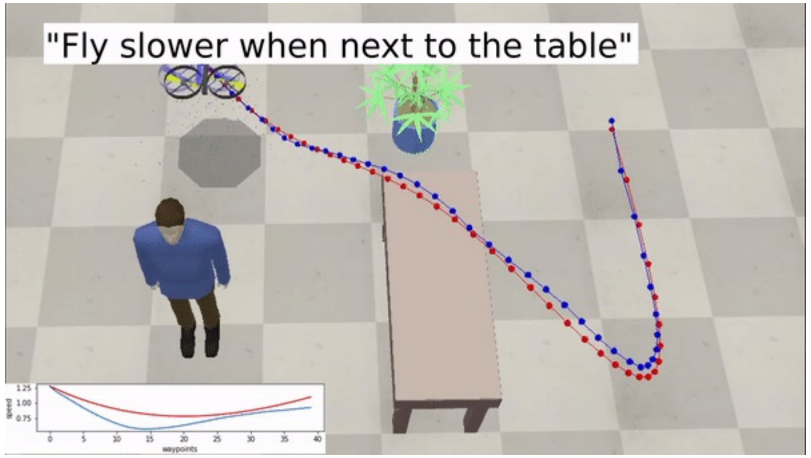
Geometry encoder

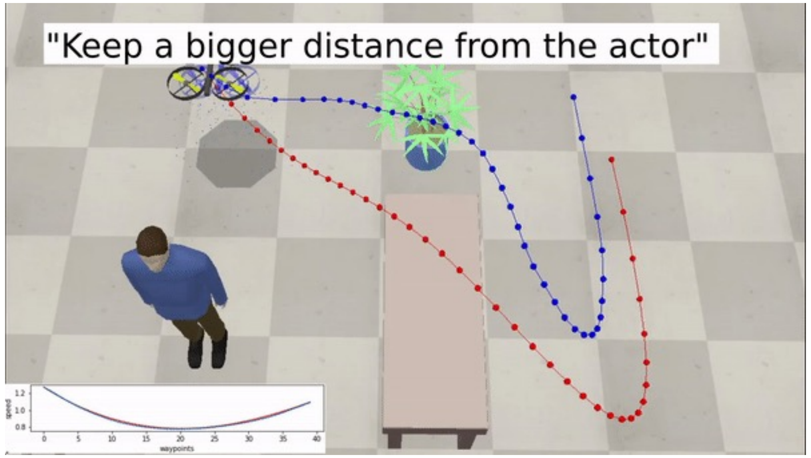
Trajectory decoder

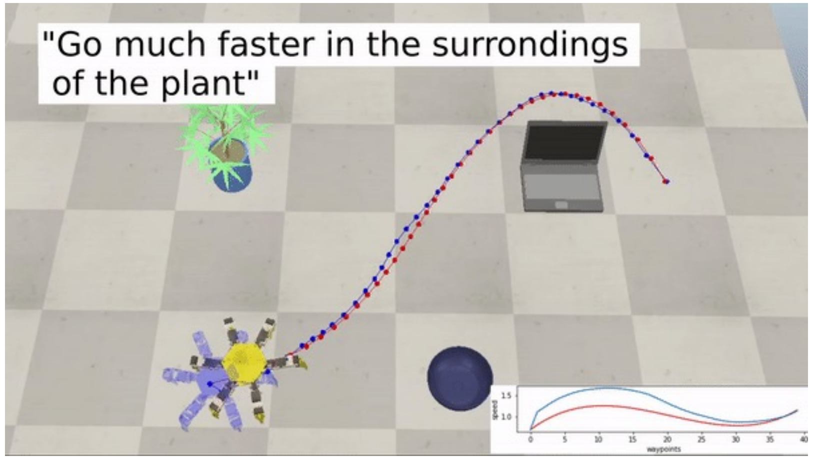
LATTE: TRAINING

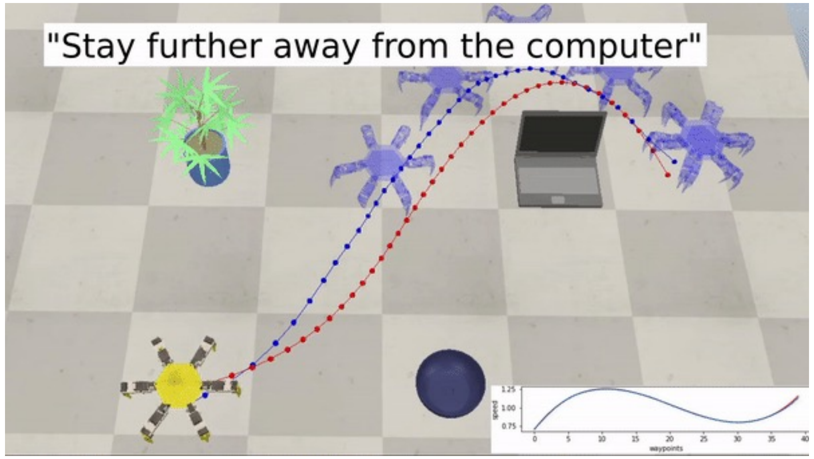
- Procedural synthetic data generation.
  - Object names: Randomly sampled from ImageNet labels
  - Object images: Crawled from Bing Images according to collected label names
  - Language inputs:
    - Translational changes
    - Speed changes
    - Object-relative changes
  - Ground truth trajectories constructed based on a handcrafted force field according to command.

LATTE: LANGUAGE TRAJECTORY TRANSFORMER





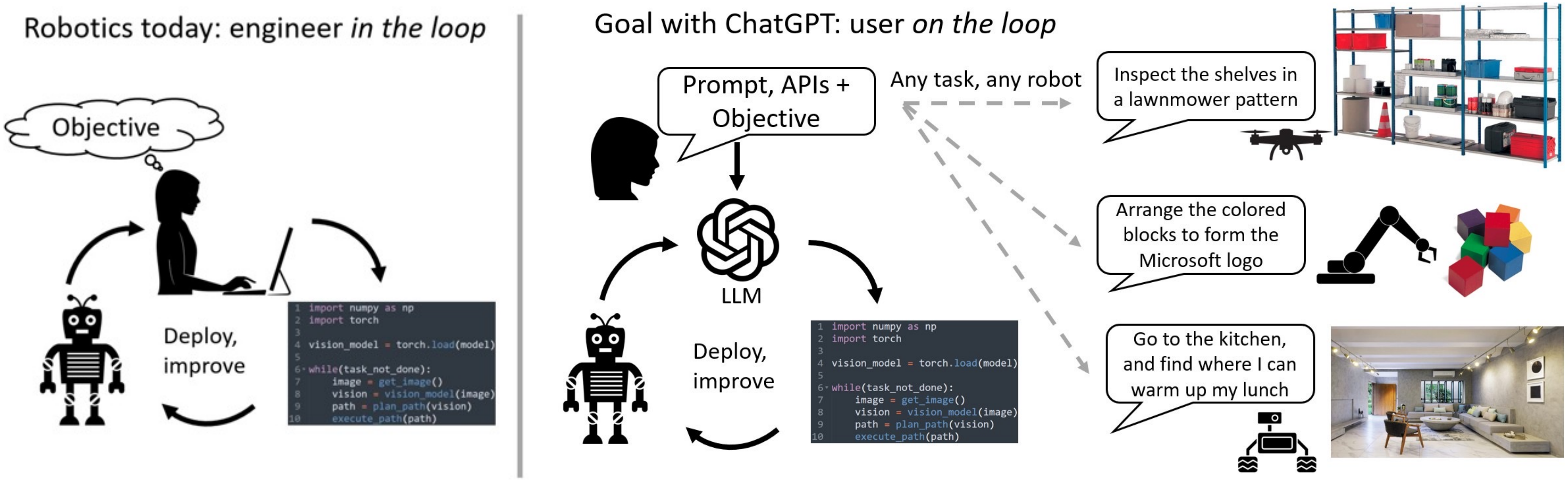




REDUCING THE BARRIER TO ENTRY

- AI in robotics still needs immense expertise and development time.
- Large language models are not just about language anymore – reducing dev cycles in several fields.
- What can LLMs do for robotics?
  - Human robot interface
  - Commonsense reasoning/planning

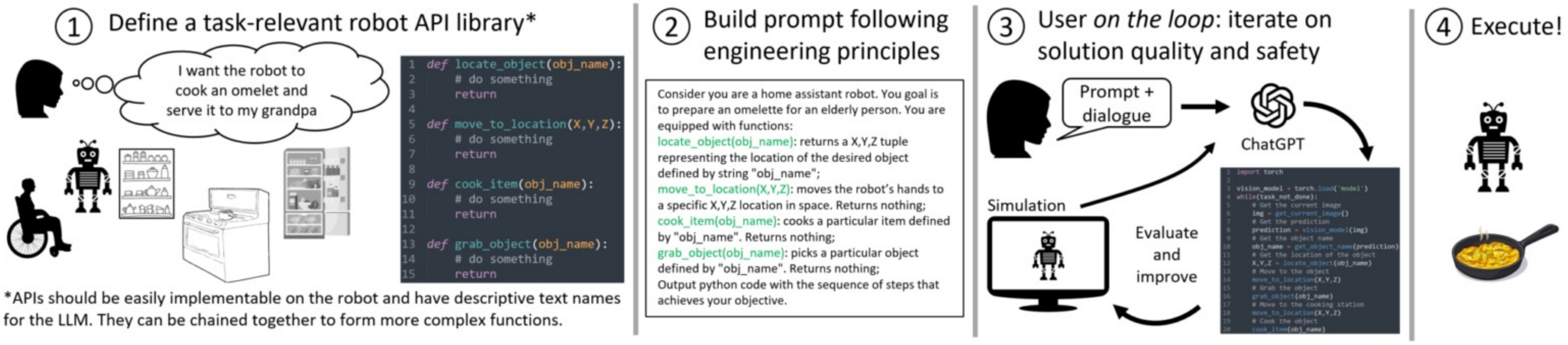
CHATGPT FOR ROBOTICS


Robotics today: engineer *in the loop*
Objective
Deploy, improve
Goal with ChatGPT: user *on the loop*
Prompt, APIs + Objective
LLM
Deploy, improve
Any task, any robot
Inspect the shelves in a lawnmower pattern
Arrange the colored blocks to form the Microsoft logo
Go to the kitchen, and find where I can warm up my lunch

```python
import numpy as np
import torch

vision_model = torch.load(model)

while(task_not_done):
    image = get_image()
    vision = vision_model(image)
    path = plan_path(vision)
    execute_path(path)
```

```python
import numpy as np
import torch

vision_model = torch.load(model)

while(task_not_done):
    image = get_image()
    vision = vision_model(image)
    path = plan_path(vision)
    execute_path(path)
```

CHATGPT FOR ROBOTICS - PIPELINE


① Define a task-relevant robot API library*
I want the robot to cook an omelet and serve it to my grandpa
② Build prompt following engineering principles
③ User *on the loop*: iterate on solution quality and safety
Prompt + dialogue
ChatGPT
Simulation
Evaluate and improve
④ Execute!

```python
def locate_object(obj_name):
    # do something
    return

def move_to_location(X,Y,Z):
    # do something
    return

def cook_item(obj_name):
    # do something
    return

def grab_object(obj_name):
    # do something
    return
```

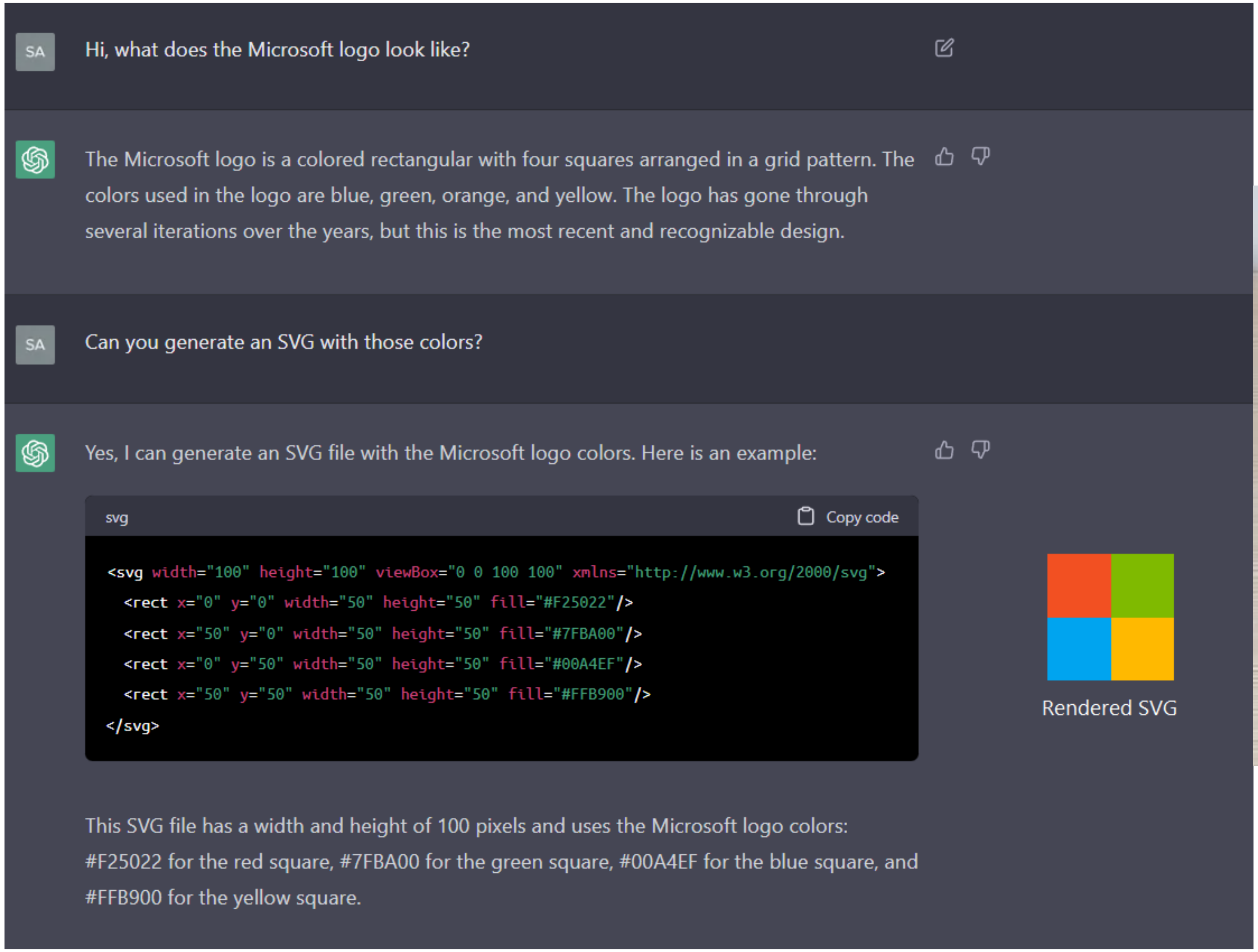
*APIs should be easily implementable on the robot and have descriptive text names for the LLM. They can be chained together to form more complex functions.

Consider you are a home assistant robot. You goal is to prepare an omelette for an elderly person. You are equipped with functions:
locate_object(obj_name): returns a X,Y,Z tuple representing the location of the desired object defined by string "obj_name";
move_to_location(X,Y,Z): moves the robot's hands to a specific X,Y,Z location in space. Returns nothing;
cook_item(obj_name): cooks a particular item defined by "obj_name". Returns nothing;
grab_object(obj_name): picks a particular object defined by "obj_name". Returns nothing;
Output python code with the sequence of steps that achieves your objective.

```python
import torch

vision_model = torch.load('model')
while(task_not_done):
    # Get the current image
    img = get_current_image()
    # Get the prediction
    prediction = vision_model(img)
    # Get the object name
    obj_name = get_object_name(prediction)
    # Get the location of the object
    X,Y,Z = locate_object(obj_name)
    # Move to the object
    move_to_location(X,Y,Z)
    # Grab the object
    grab_object(obj_name)
    # Move to the cooking station
    move_to_location(X,Y,Z)
    # Cook the object
    cook_item(obj_name)
```

Imagine we are working with a household robot. The job of this robot is to make an omelette. The objects available around are:
fridge, bowl, pan, oil, stove
The main functions you can use are:
`locate_object(object_name)`: Returns the XYZ coordinates of an object of interest.
`go_to_location(object_name)`: Moves robot to a location specified by XYZ coordinates. Returns nothing.
`pick_up(object_name)`: Picks up the object of interest. Returns nothing.
`use_item(object_name)`: Takes the name of an object as input. Performs an appropriate action on the object at hand (pour oil, crack eggs). Returns nothing.
Can you make use of these to write code to go to the kitchen and make an omelette? Try to use these low level functions to construct more high level behaviors and then solve the task using those high level primitives.

CHATGPT FOR ROBOTICS



Hi, what does the Microsoft logo look like?

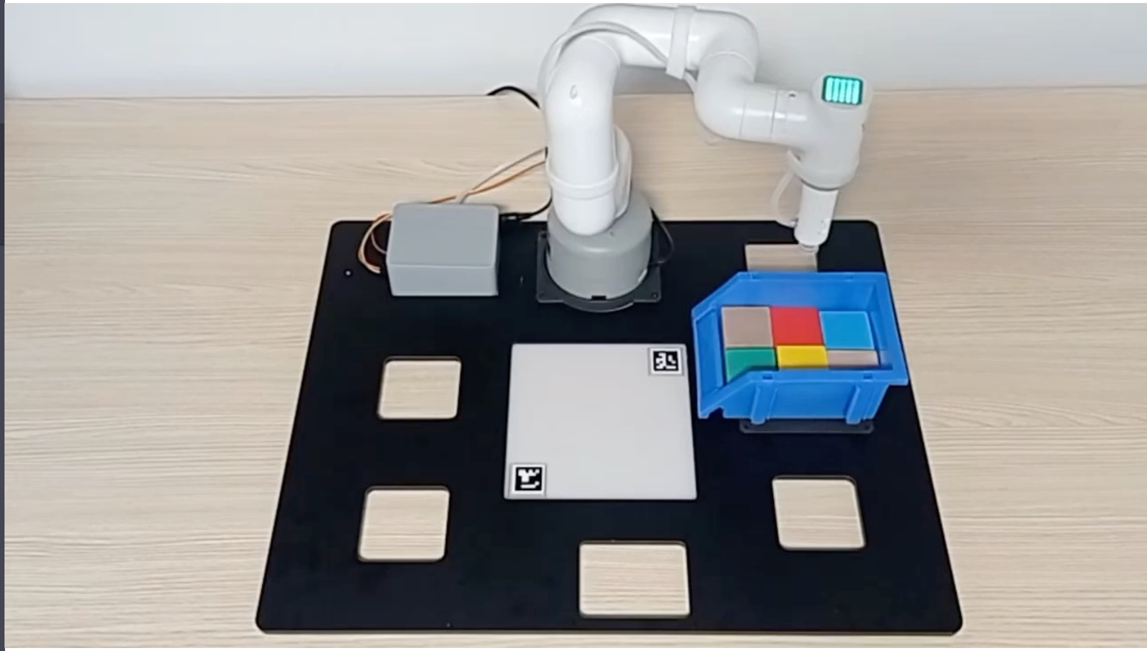
The Microsoft logo is a colored rectangular with four squares arranged in a grid pattern. The colors used in the logo are blue, green, orange, and yellow. The logo has gone through several iterations over the years, but this is the most recent and recognizable design.

Can you generate an SVG with those colors?

Yes, I can generate an SVG file with the Microsoft logo colors. Here is an example:

```svg
<svg width="100" height="100" viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="50" height="50" fill="#F25022"/>
  <rect x="50" y="0" width="50" height="50" fill="#7FBA00"/>
  <rect x="0" y="50" width="50" height="50" fill="#00A4EF"/>
  <rect x="50" y="50" width="50" height="50" fill="#FFB900"/>
</svg>
```

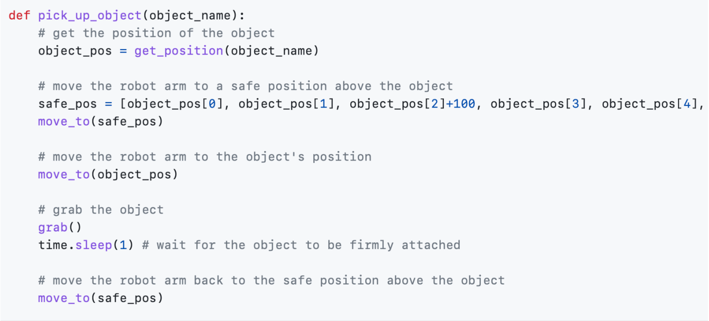
This SVG file has a width and height of 100 pixels and uses the Microsoft logo colors: #F25022 for the red square, #7FBA00 for the green square, #00A4EF for the blue square, and #FFB900 for the yellow square.



INTERESTING FINDINGS

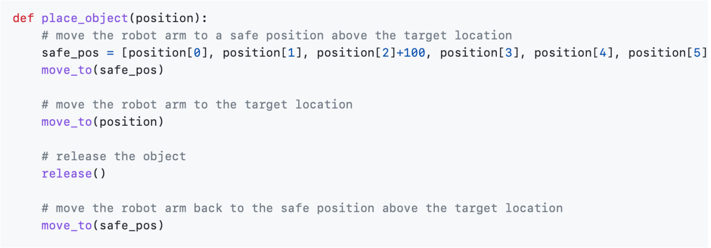
- ChatGPT can adapt to a curriculum – learn primitive skills and chain them for complex tasks.

“I want you to learn the skill of picking up a single object and holding it. For that you need to move a safe distance above the object (100 mm), reach the object, grab it and bring it up.”



```
def pick_up_object(object_name):
    # get the position of the object
    object_pos = get_position(object_name)

    # move the robot arm to a safe position above the object
    safe_pos = [object_pos[0], object_pos[1], object_pos[2]+100, object_pos[3], object_pos[4],
    move_to(safe_pos)

    # move the robot arm to the object's position
    move_to(object_pos)

    # grab the object
    grab()
    time.sleep(1) # wait for the object to be firmly attached

    # move the robot arm back to the safe position above the object
    move_to(safe_pos)
```

Now I want you to learn how to place an object in a position. You first need to first move to a safe position above the target location, bring the object down and then release it.

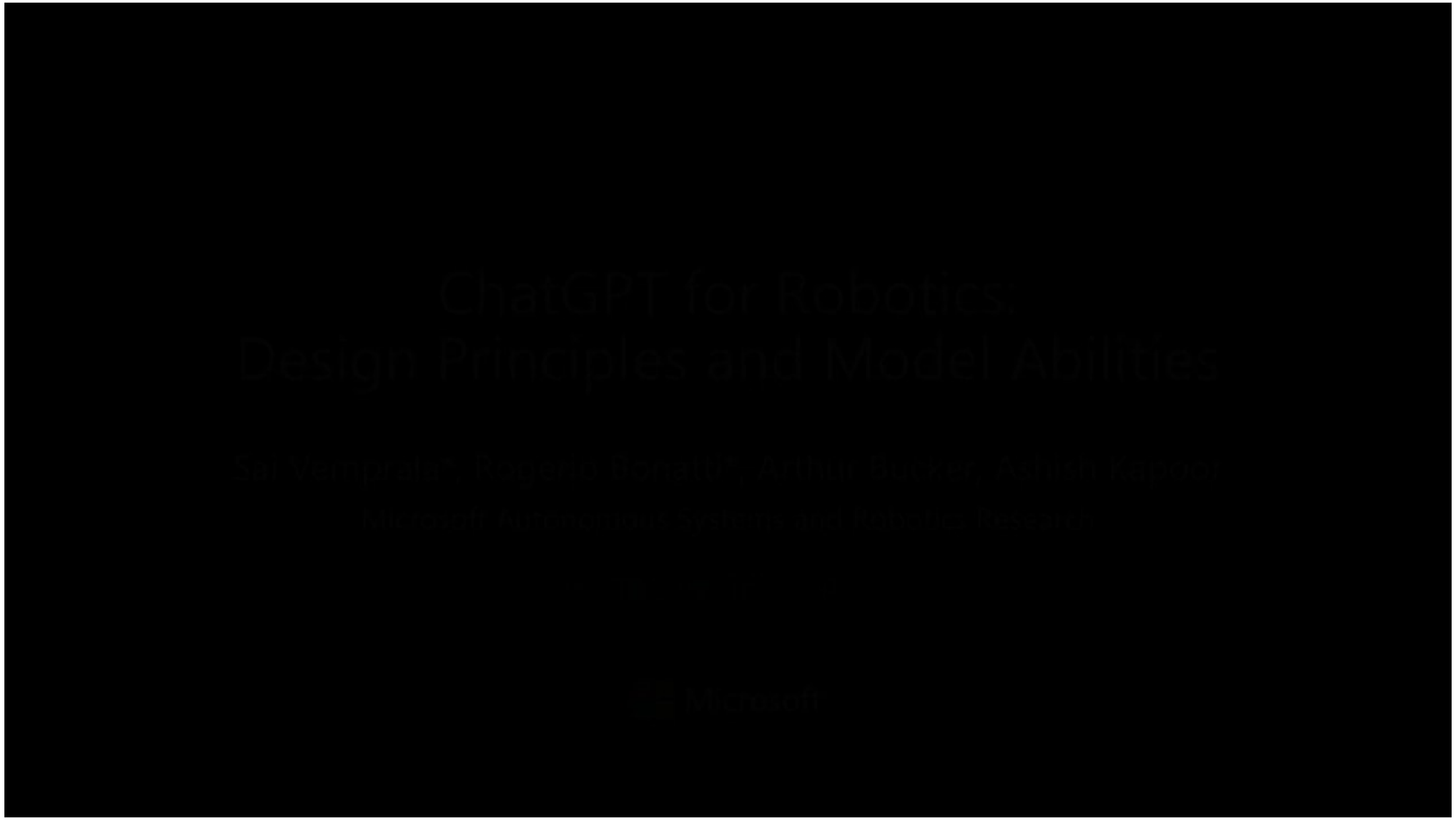


```
def place_object(position):
    # move the robot arm to a safe position above the target location
    safe_pos = [position[0], position[1], position[2]+100, position[3], position[4], position[5]
    move_to(safe_pos)

    # move the robot arm to the target location
    move_to(position)

    # release the object
    release()

    # move the robot arm back to the safe position above the target location
    move_to(safe_pos)
```

Now I want you to learn how to place an object in a position. You first need to first move to a safe position above the target location, bring the object down and then release it.

INTERESTING FINDINGS

- Scales to multiple robots, and tasks.



ChatGPT for Robotics:
Design Principles and Model Abilities

Sai Vemprala*, Rogerio Bonatti*, Arthur Bucker, Ashish Kapoor
Microsoft Autonomous Systems and Robotics Research

INTERESTING FINDINGS

- Scales to multiple form factors, and tasks.

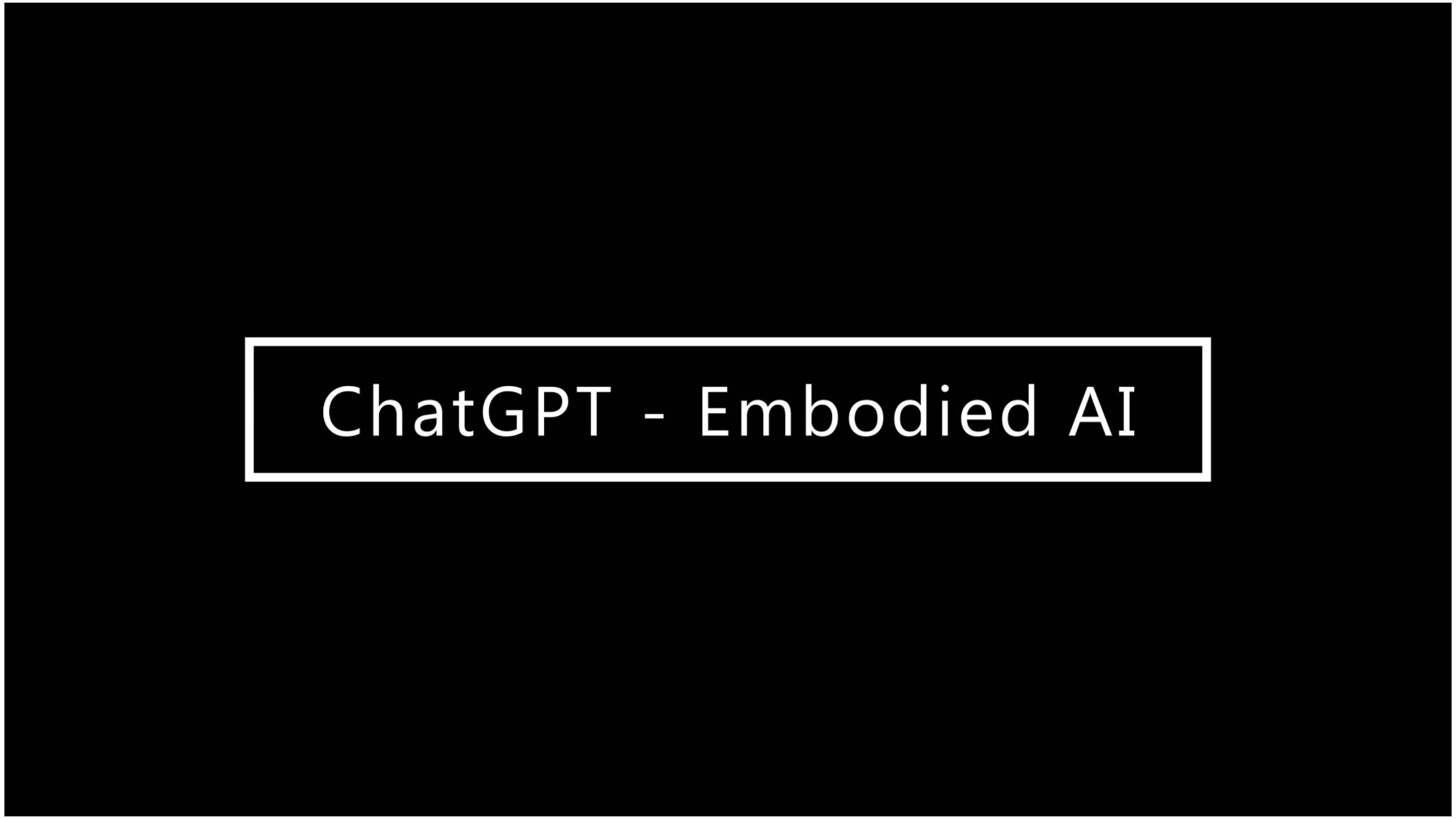
ChatGPT for Robotics:
Design Principles and Model Abilities

Sai Vemprala*, Rogerio Bonatti*, Arthur Bucker, Ashish Kapoor
Microsoft Autonomous Systems and Robotics Research

Microsoft

INTERESTING FINDINGS

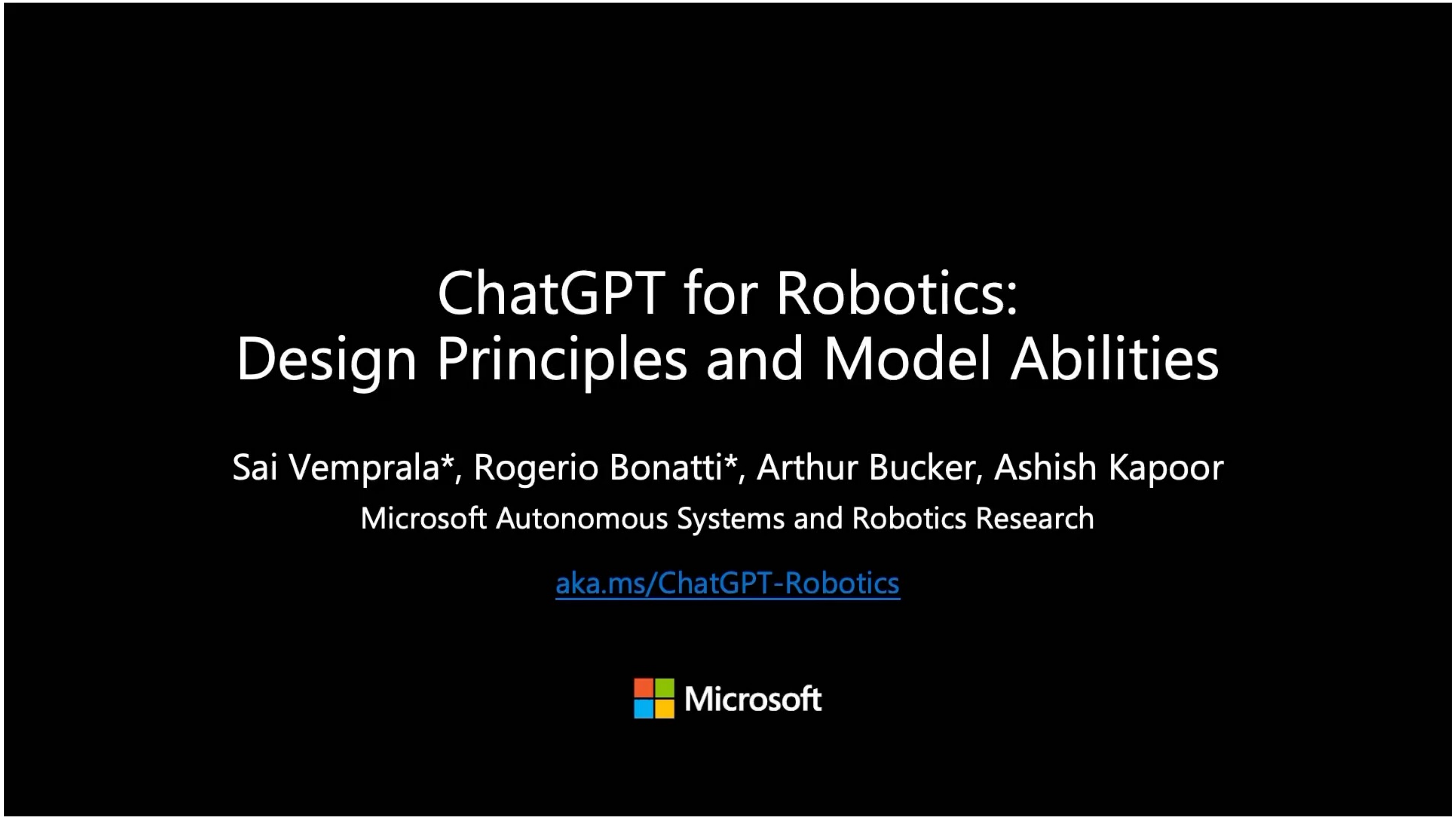
- Writes complex perception-action loops by leveraging “tools” (sensors, ML models)



ChatGPT - Embodied AI

INTERESTING FINDINGS

- Writes complex perception-action loops by leveraging “tools” (sensors, ML models)



ChatGPT for Robotics:
Design Principles and Model Abilities

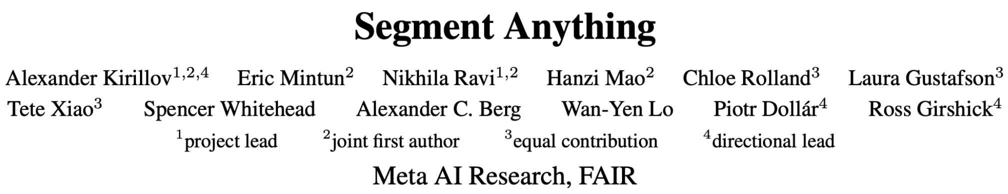
Sai Vemprala*, Rogerio Bonatti*, Arthur Bucker, Ashish Kapoor
Microsoft Autonomous Systems and Robotics Research

aka.ms/ChatGPT-Robotics

Microsoft

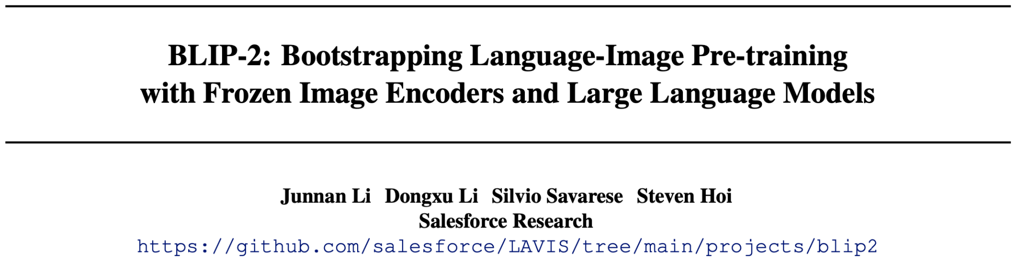
WHAT DOES THE FUTURE HOLD?

- Foundation Models / GPT extensions space is seeing exponential progress.



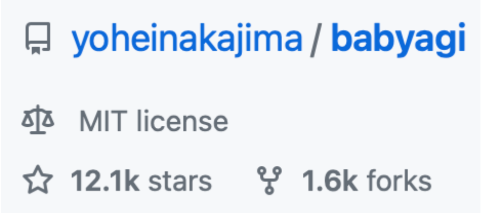
Segment Anything

Alexander Kirillov[1,2,4] Eric Mintun[2] Nikhila Ravi[1,2] Hanzi Mao[2] Chloe Rolland[3] Laura Gustafson[3]
Tete Xiao[3] Spencer Whitehead Alexander C. Berg Wan-Yen Lo Piotr Dollár[4] Ross Girshick[4]
[1]project lead [2]joint first author [3]equal contribution [4]directional lead
Meta AI Research, FAIR



BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models

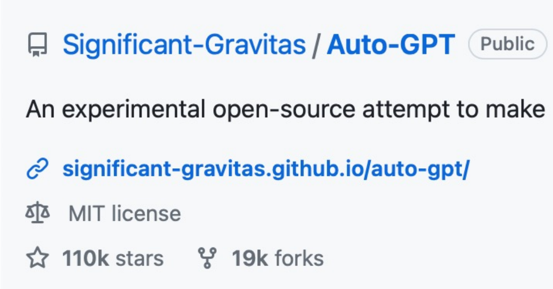
Junnan Li Dongxu Li Silvio Savarese Steven Hoi
Salesforce Research
`https://github.com/salesforce/LAVIS/tree/main/projects/blip2`



yoheinakajima / babyagi

MIT license

☆ 12.1k stars 1.6k forks



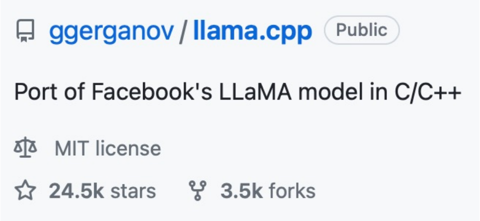
Significant-Gravitas / Auto-GPT Public

An experimental open-source attempt to make

significant-gravitas.github.io/auto-gpt/

MIT license

☆ 110k stars 19k forks



ggerganov / llama.cpp Public

Port of Facebook's LLaMA model in C/C++

MIT license

☆ 24.5k stars 3.5k forks

WHAT DOES THE FUTURE HOLD?

- Foundation Models / GPT extensions space is seeing exponential progress.
- Simulations / generative AI techniques will become key.
- Perception + Reasoning using off the shelf models, translated into control.

microsoft/PromptCraft-Robotics: Community for applying LLMs to robotics and a robot simulator with ChatGPT integration (github.com)

microsoft/PACT: Perception-Action Causal Transformer (github.com)

arthurfenderbucker/LaTTe-Language-Trajectory-TransformEr (github.com)