THE UNIVERSITY *of* ADELAIDE

Foundation Models in Robotics: Bridging AI and Physical Interaction

Feras Dayoub

Australian Institute for Machine Learning

adelaide.edu.au/aiml

Overview:

- Introduction to Foundation Models.
- Types and Applications.
- Grounding in Robotics.
- Action Generation with Foundation Models.
- Benefits, Challenges, and Future Directions.

THE UNIVERSITY *of* ADELAIDE

Definitions and Intro

Definition of Foundation Models (FMs)

A Foundation Model is any ML model **trained on broad data at scale**, typically using **self-supervised learning** methods. These models are designed to be **adaptable** to a wide range of **downstream tasks** through **fine-tuning or prompting**.

They serve as a **'foundation'** upon which more specialized applications can be built.

THE UNIVERSITY *of* ADELAIDE

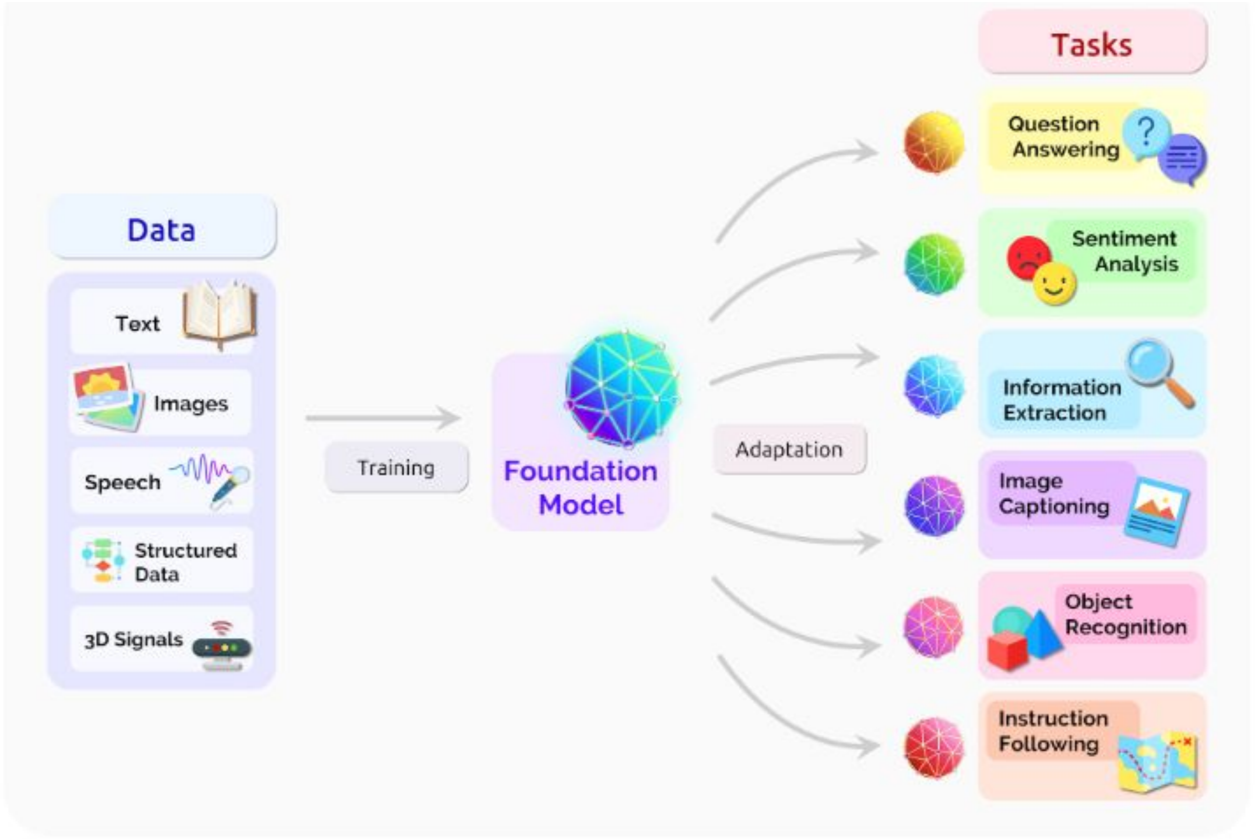


Fig. 2. A foundation model can centralize the information from all the data from various modalities. This one model can then be adapted to a wide range of downstream tasks.

* Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein et al. "On the opportunities and risks of foundation models." *arXiv preprint arXiv:2108.07258* (2021).

Brief history and evolution

- The concept has its roots in **transfer learning and pre-training**, which gained popularity in the early **2010s**.
- The term 'Foundation Model' itself was coined more recently, in **a 2021 paper from Stanford***.
  - "any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks"
- The real explosion in Foundation Models came with the advent of large language models like **GPT-3 in 2020**.
- Since then, we've seen rapid developments in **vision models**, **multimodal models**, and now, models **specifically designed for robotics applications**.

* Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein et al. "On the opportunities and risks of foundation models." *arXiv preprint arXiv:2108.07258* (2021).

Key milestones

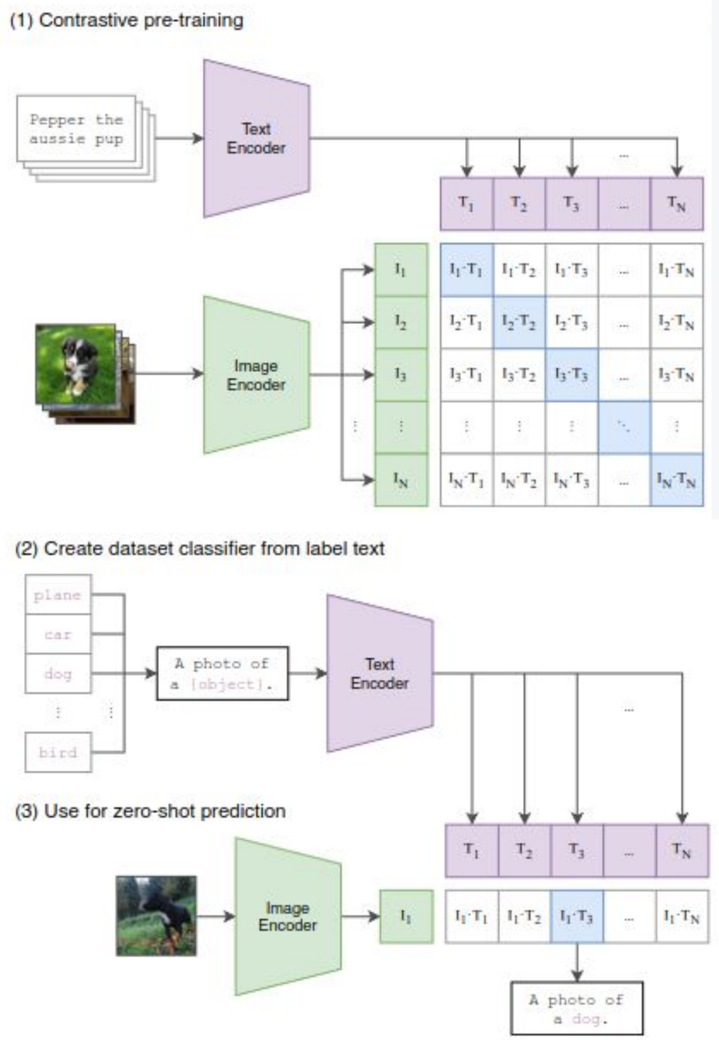
2018: BERT, which revolutionized natural language processing

2020: GPT-3, showing the power of scale in language models

2021: CLIP, demonstrating strong vision-language connections

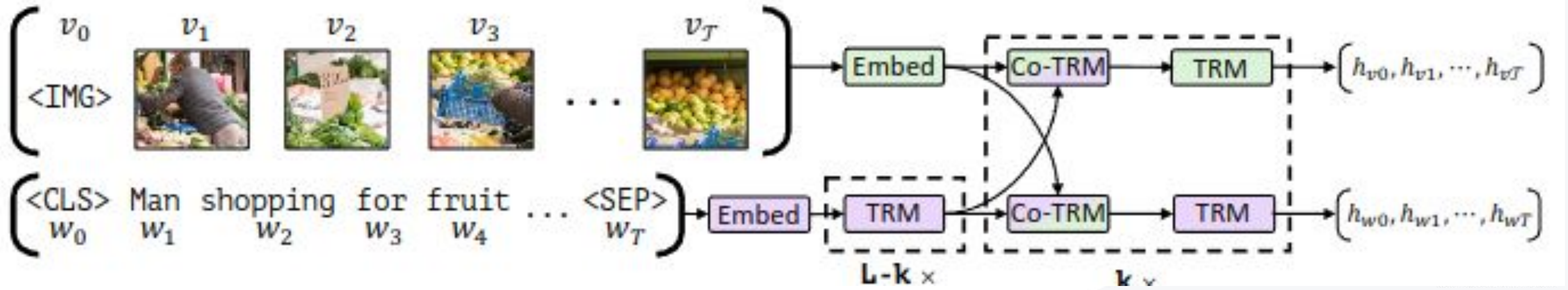
2022: DALL-E 2 and SAM (Segment Anything Model)

2023: Robotics-specific Foundation Models like RT-1 and RT-2 and more.



Types of Foundation Models

1. **Large Language Models (LLMs) (e.g., GPT-3, BERT)**
2. **VM / VLM Models (e.g., CLIP)**
3. **Multimodal Models (e.g., ViLBERT, Whisper)**



v_0 v_1 v_2 v_3 v_T
<IMG>
<CLS> Man shopping for fruit ... <SEP>
w_0 w_1 w_2 w_3 w_4 w_T
Embed
Co-TRM
TRM
$(h_{v0}, h_{v1}, \cdots, h_{vT})$
$(h_{w0}, h_{w1}, \cdots, h_{wT})$
L-k ×
k ×

Large Language Models (LLMs)

Architecture:

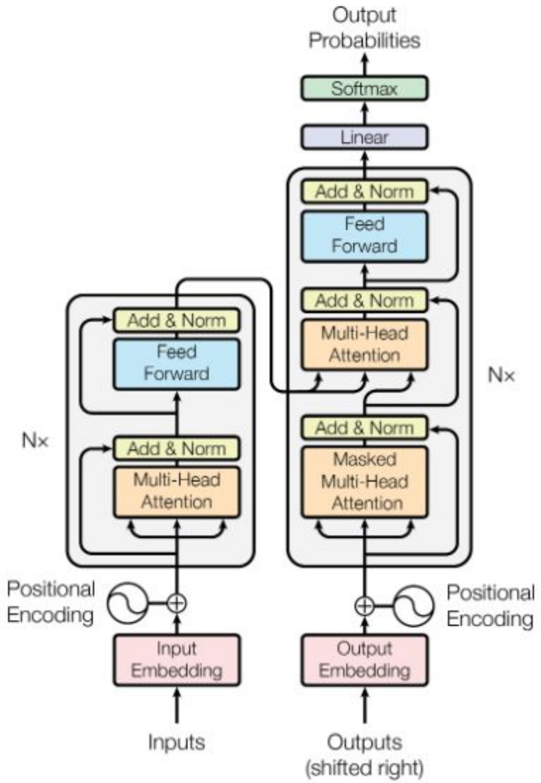
- Typically based on transformer architecture
- Massive scale: Billions to trillions of parameters

Training

- Self-supervised learning on vast text corpora
- Objective: Next token prediction

Key Capabilities

- Few-shot and zero-shot learning
- Task-agnostic performance across various NLP tasks
- Emergent abilities at scale (e.g., reasoning, code generation)



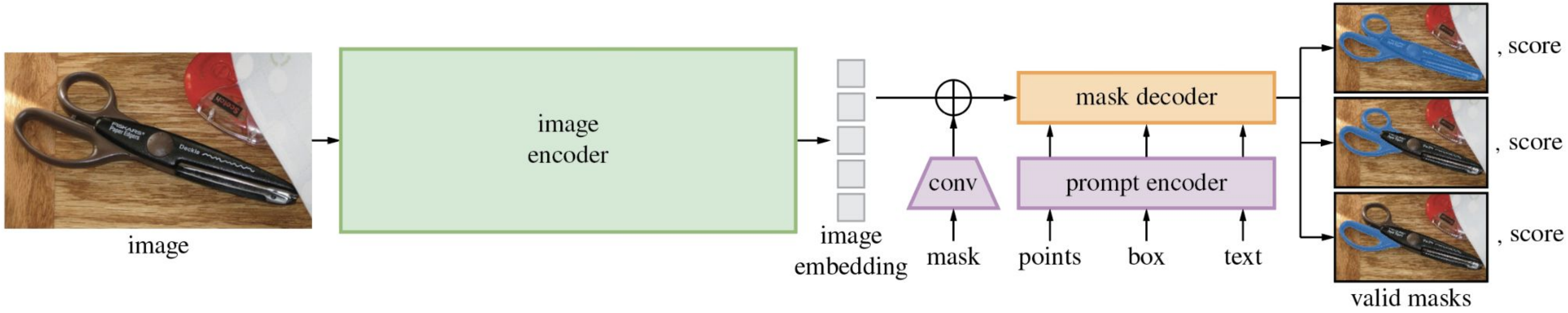
In robotics, LLMs are increasingly used for task planning, instruction understanding, and generating reward functions for reinforcement learning.

THE UNIVERSITY of ADELAIDE

Vision and Vision-Language Models

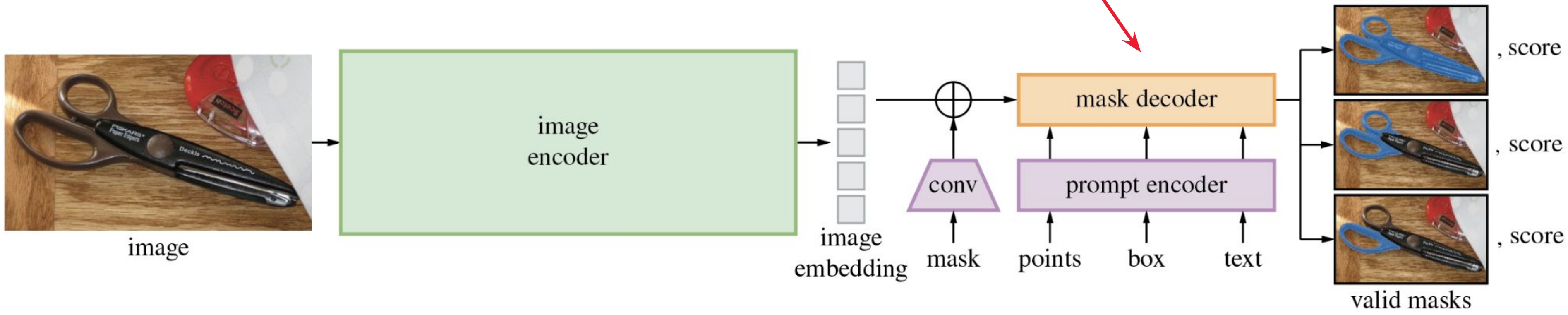
VLM are Foundation Models trained on large datasets of images and text.

- **CLIP:** Jointly trained on image and text data, allowing it to understand the relationship between visual concepts and language.
- **SAM:** Specializes in image segmentation, capable of identifying and outlining objects in images.



SAM (Segment Anything Model)

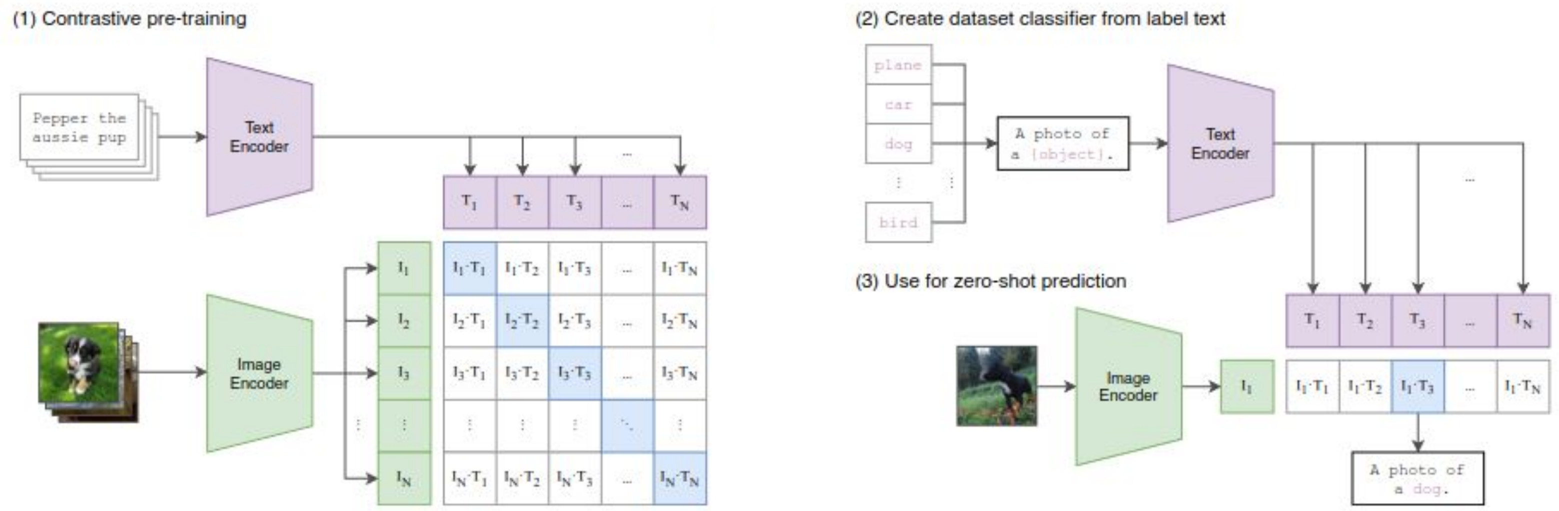
Attention mechanisms within the decoder allow the model to focus on relevant parts of the image based on the prompt. This selective focus ensures that the generated masks are precise and contextually accurate.



CLIP

(Contrastive Language-Image Pre-training)

Unified Embedding Space: Both the image and text encoders map their inputs into a shared embedding space where similar images and texts are close to each other

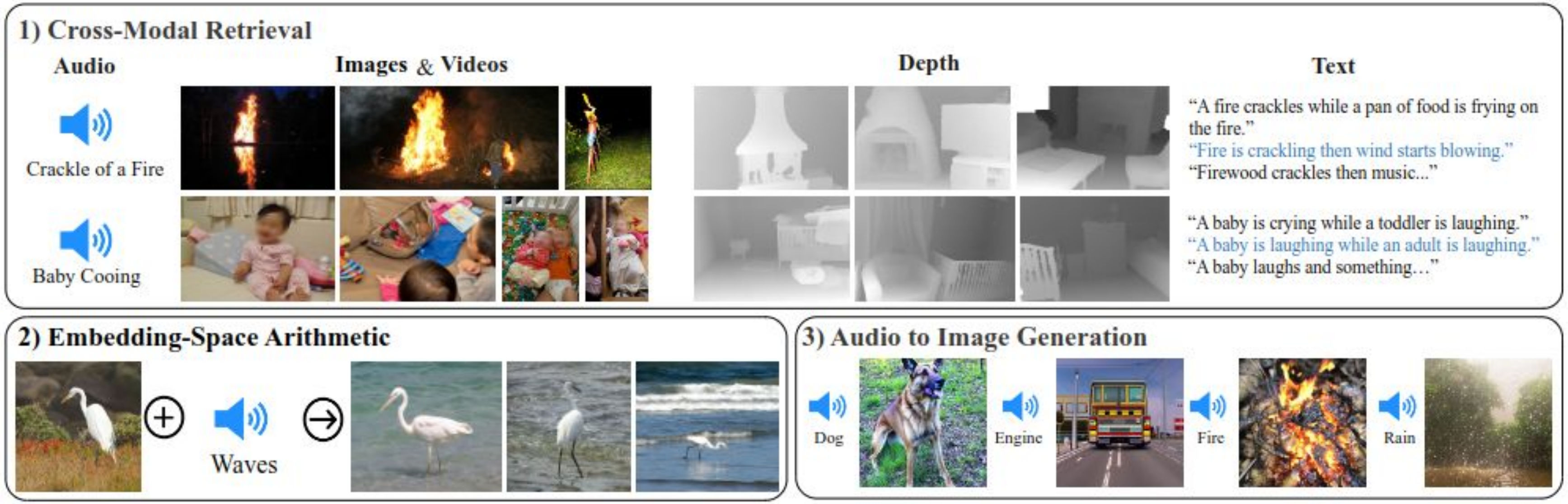


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ... I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

THE UNIVERSITY of ADELAIDE

IMAGEBIND: One Embedding Space To Bind Them All

Rohit Girdhar* Alaaeldin El-Nouby* Zhuang Liu Mannat Singh
Kalyan Vasudev Alwala Armand Joulin Ishan Misra*
FAIR, Meta AI
https://facebookresearch.github.io/ImageBind



"A fire crackles while a pan of food is frying on the fire."
"Fire is crackling then wind starts blowing."
"Firewood crackles then music..."

"A baby is crying while a toddler is laughing."
"A baby is laughing while an adult is laughing."
"A baby laughs and something..."

Multimodal Models

Input Modalities:

- Combinations of text, vision, audio, and sometimes other sensor data

Training:

- Self-supervised learning on diverse, large-scale multimodal datasets
- Objectives: Cross-modal alignment, joint representation learning

Key Capabilities:

- **Unified understanding across modalities**
- **Cross-modal generation and translation**
- **Zero-shot, few-shot learning for multimodal tasks**

THE UNIVERSITY of ADELAIDE

FM and robotics

THE UNIVERSITY *of* ADELAIDE

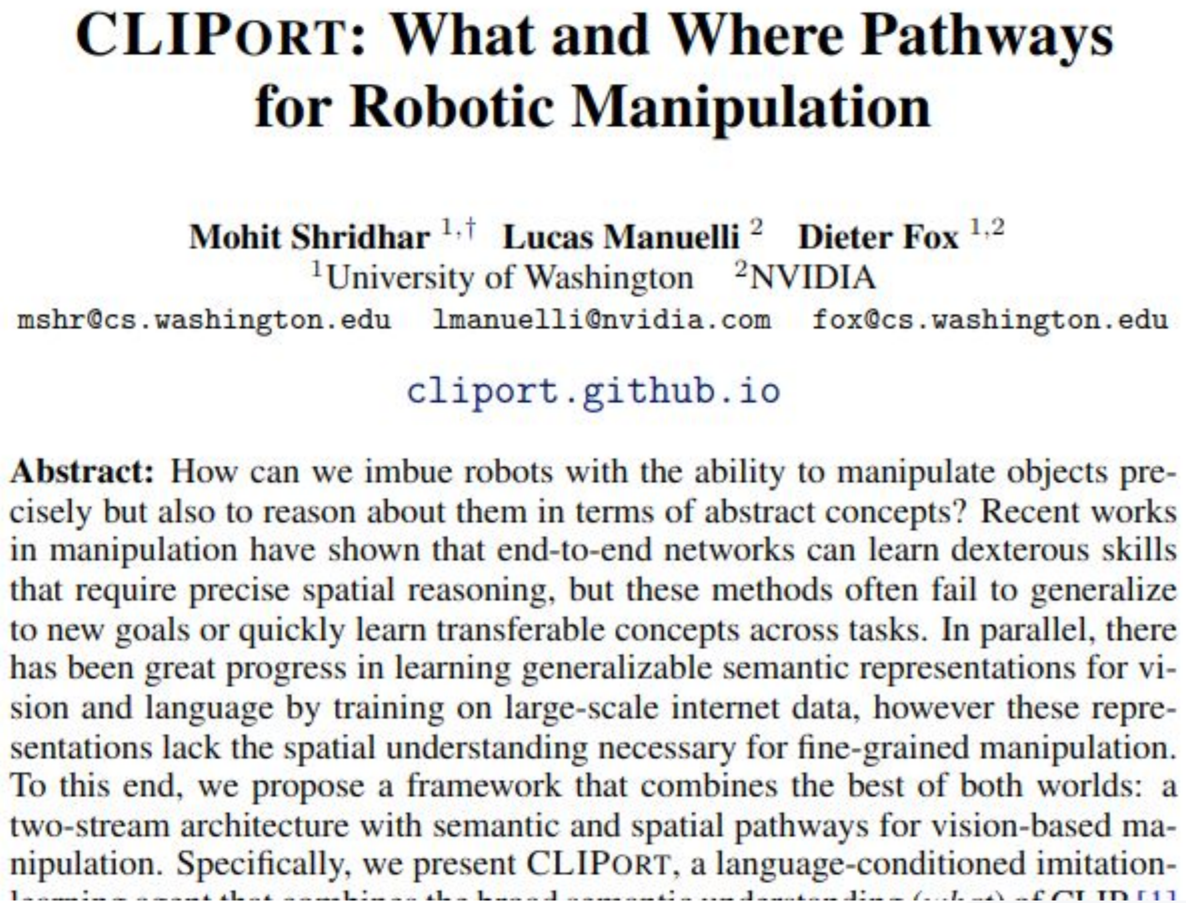
Key advantages of FMs for robotics

- Offer the potential for more **flexible, adaptable, and intelligent** robotic systems that can operate effectively **in complex, real-world environments**.
    - Self-supervised learning at scale
    - Adaptability to downstream tasks
    - Generalization capabilities

THE UNIVERSITY *of* ADELAIDE

Integrating FM in Robotic Systems

Large foundational models can enhance the capabilities of robotic systems by **combining semantic understanding** with **spatial reasoning**, leading to more intelligent and adaptable robots capable of performing complex tasks.



CLIPort: What and Where Pathways for Robotic Manipulation

Mohit Shridhar [1,†] **Lucas Manuelli** [2] **Dieter Fox** [1,2]
[1]University of Washington [2]NVIDIA
mshr@cs.washington.edu lmanuelli@nvidia.com fox@cs.washington.edu

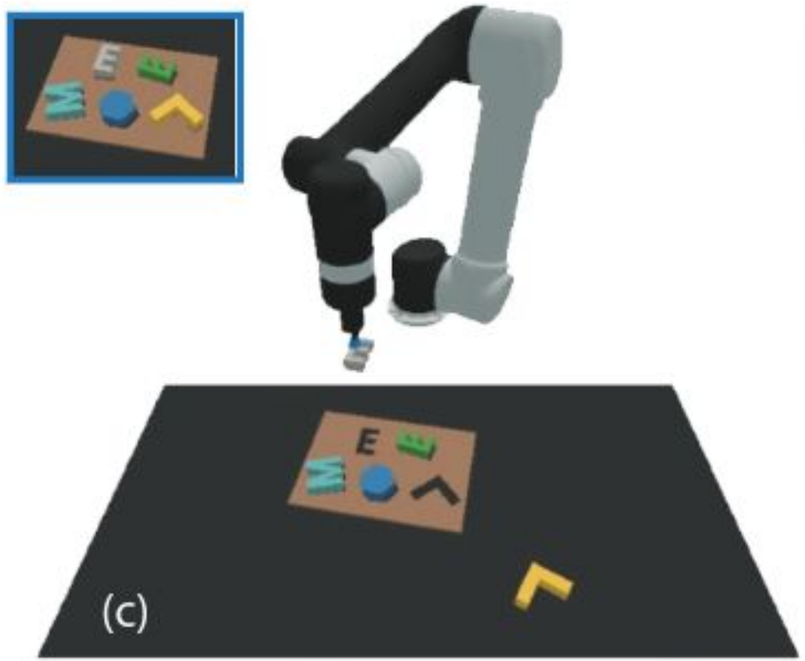
cliport.github.io

Abstract: How can we imbue robots with the ability to manipulate objects precisely but also to reason about them in terms of abstract concepts? Recent works in manipulation have shown that end-to-end networks can learn dexterous skills that require precise spatial reasoning, but these methods often fail to generalize to new goals or quickly learn transferable concepts across tasks. In parallel, there has been great progress in learning generalizable semantic representations for vision and language by training on large-scale internet data, however these representations lack the spatial understanding necessary for fine-grained manipulation. To this end, we propose a framework that combines the best of both worlds: a two-stream architecture with semantic and spatial pathways for vision-based manipulation. Specifically, we present CLIPort, a language-conditioned imitation-

Case study: CLIPort for object manipulation

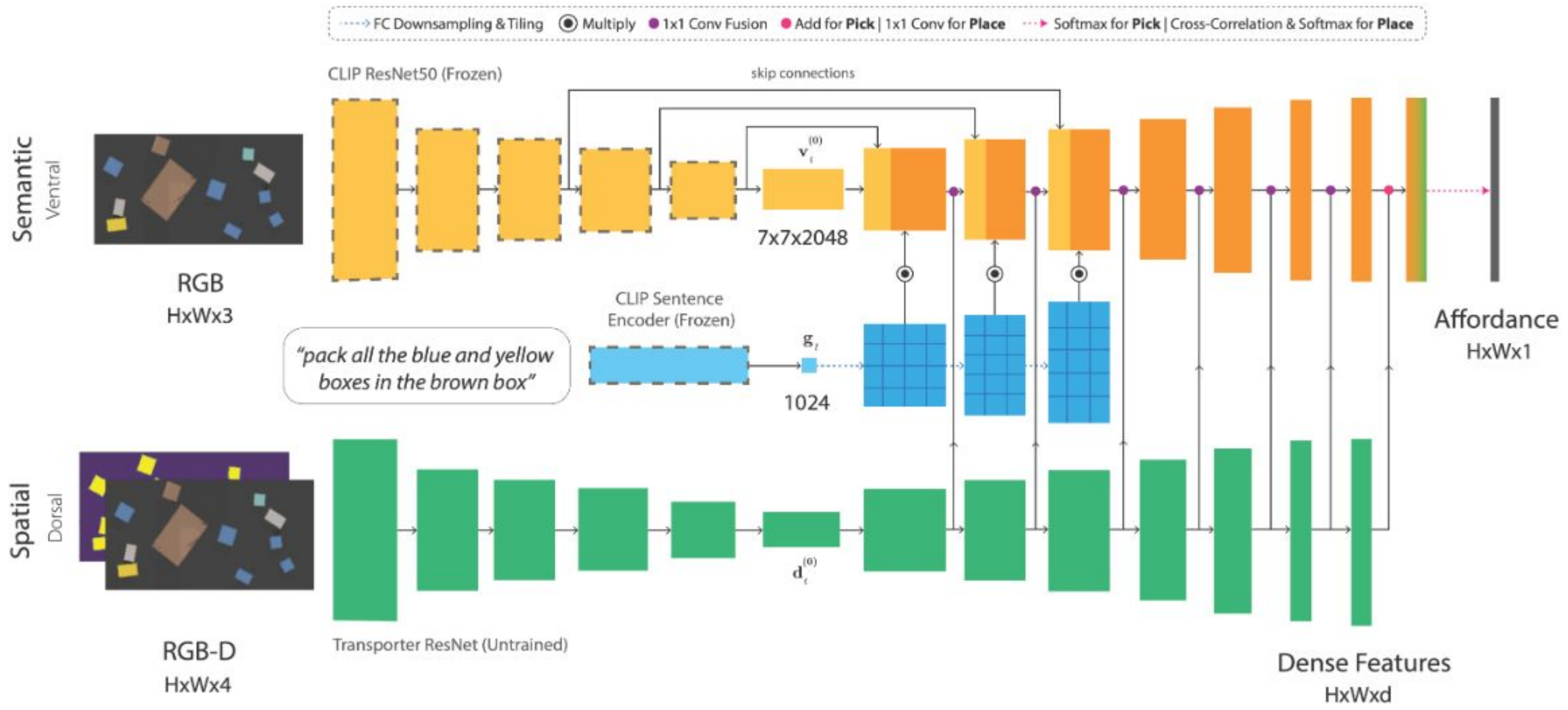
- Combines the **CLIP model** for semantic understanding and **Transporter Networks** for transport-based policies.

- **Uses a two-stream architecture:** one for encoding semantic information (via CLIP) and another for encoding spatial information (via Transporter Networks).



"put the gray letter E in the left letter E shape hole"

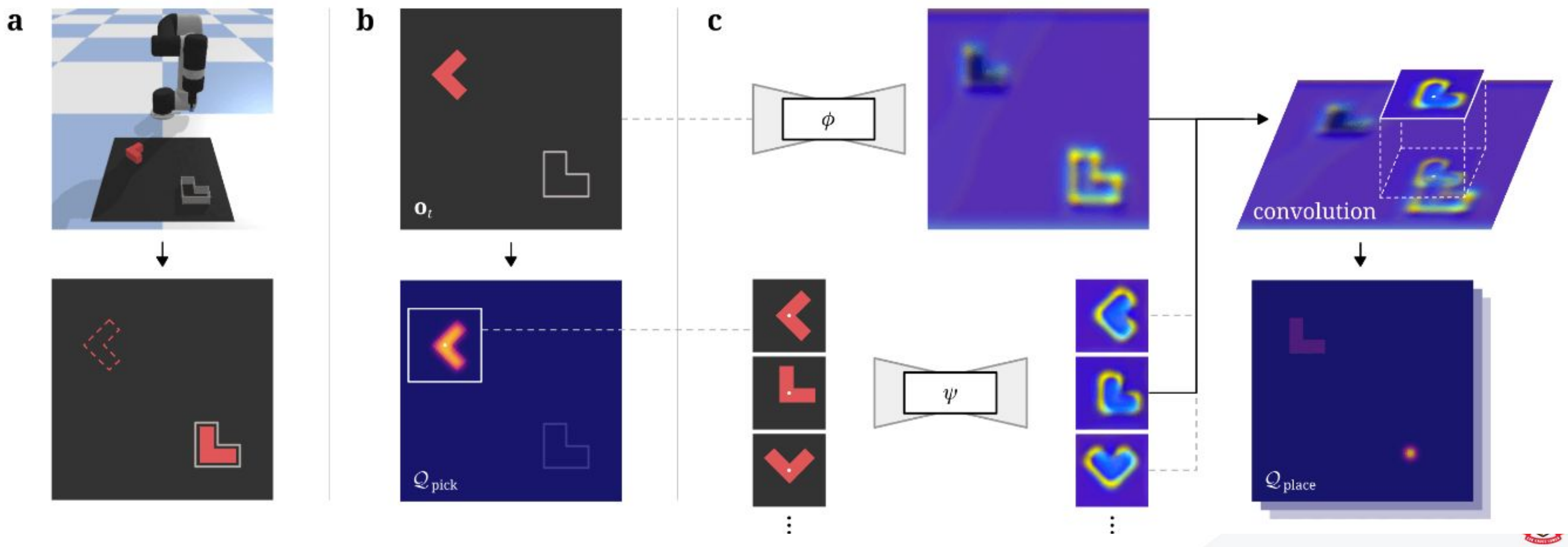
How CLIPort works



FC Downsampling & Tiling
Multiply
1x1 Conv Fusion
Add for **Pick** | 1x1 Conv for **Place**
Softmax for **Pick** | Cross-Correlation & Softmax for **Place**
CLIP ResNet50 (Frozen)
skip connections
Semantic
Ventral
RGB
HxWx3
$v_t^{(0)}$
7x7x2048
CLIP Sentence Encoder (Frozen)
"pack all the blue and yellow boxes in the brown box"
g_t
1024
Affordance
HxWx1
Spatial
Dorsal
RGB-D
HxWx4
$d_t^{(0)}$
Transporter ResNet (Untrained)
Dense Features
HxWxd

Transporter Networks

- It can predict pick-and-place affordances by attending to **local regions for picking** and **matching placements through cross-correlation of visual features**.



Zeng, A., Florence, P., Tompson, J., Welker, S., Chien, J., Attarian, M., ... & Lee, J. (2021, October). Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning* (pp. 726-747). PMLR.

Grounding

Definition of Grounding

- **G**rounding refers to the process of connecting abstract concepts or symbols (like words or phrases) to concrete, real-world entities or actions.
- **For robots,** this means linking language understanding to physical perceptions and actions in the real world.

Why is Grounding Important?

- LLMs are trained on vast amounts of text data, giving them broad knowledge and language understanding.
- However, this knowledge is abstract and disconnected from the physical world.
- For a robot to use this knowledge effectively, it needs to ground it in its physical environment and capabilities.

Challenges of Grounding in Robotics

- **Symbol Grounding Problem:** This is the challenge of connecting symbolic representations (like words) to their physical referents in the world.
- **Embodiment:** Robots have specific physical capabilities and limitations that may not align perfectly with language descriptions.
- **Context Dependency:** The meaning of language can change based on the physical context, which the robot needs to understand.

Approaches to Grounding in Robotics

- **Perception-based Grounding: Linking language to visual or sensory inputs.**
  - Example: Associating the word "cup" with visual features of cups the robot has seen.
- **Action-based Grounding: Connecting language to the robot's possible actions.**
  - Example: Understanding "pick up" in terms of the robot's grasping capabilities.
- **Context-aware Grounding: Interpreting language based on the current environment and task.**
  - Example: Understanding that "move left" means different things depending on the robot's current orientation.

THE UNIVERSITY of ADELAIDE

Role of Foundation Models in Grounding

- Foundation Models, especially **multimodal** ones, can help bridge the gap between language and physical reality.
- They can provide rich, **pre-trained representations** that link language to visual features or common sense knowledge about the world.
- However, these models still need to be **adapted to the specific embodiment** and capabilities of each robot.

Example: Grounding in Task Planning

Consider a robot instructed to **"bring me a cold drink."**

Grounding this instruction involves:

a) Understanding what constitutes a "cold drink" in the current context.

b) Locating the fridge in the environment.

c) Knowing how to open the fridge based on its specific design.

d) Identifying suitable object inside the fridge.

e) Understanding how to grasp and carry the drink safely.

THE UNIVERSITY *of* ADELAIDE

Ongoing Challenges in Grounding

- **Handling Ambiguity:** Language can be ambiguous, and robots need to resolve this ambiguity based on context.
- **Dealing with Novel Situations:** Robots need to ground language in situations they haven't encountered before.
- **Feedback and Learning:** Developing systems that can learn and improve their grounding over time through interaction and feedback.

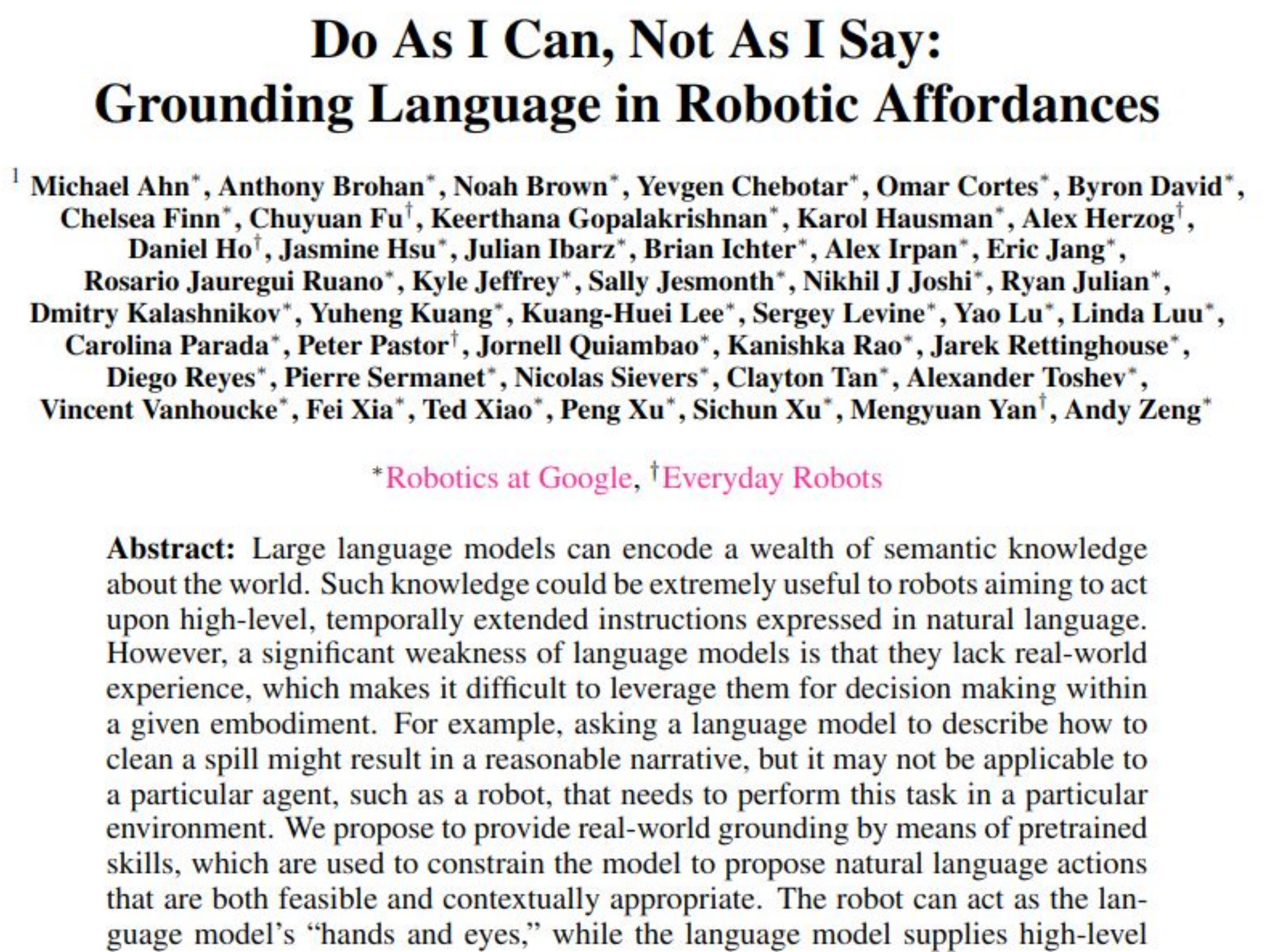
THE UNIVERSITY *of* ADELAIDE

Action Generation with FMs

A. LLMs for task planning: SayCan approach

B. Reward function generation: Language to Rewards (L2R)

C. Action generation: RT-1 and RT-2 models



Do As I Can, Not As I Say: Grounding Language in Robotic Affordances

[1] Michael Ahn*, Anthony Brohan*, Noah Brown*, Yevgen Chebotar*, Omar Cortes*, Byron David*, Chelsea Finn*, Chuyuan Fu†, Keerthana Gopalakrishnan*, Karol Hausman*, Alex Herzog†, Daniel Ho†, Jasmine Hsu*, Julian Ibarz*, Brian Ichter*, Alex Irpan*, Eric Jang*, Rosario Jauregui Ruano*, Kyle Jeffrey*, Sally Jesmonth*, Nikhil J Joshi*, Ryan Julian*, Dmitry Kalashnikov*, Yuheng Kuang*, Kuang-Huei Lee*, Sergey Levine*, Yao Lu*, Linda Luu*, Carolina Parada*, Peter Pastor†, Jornell Quiambao*, Kanishka Rao*, Jarek Rettinghouse*, Diego Reyes*, Pierre Sermanet*, Nicolas Sievers*, Clayton Tan*, Alexander Toshev*, Vincent Vanhoucke*, Fei Xia*, Ted Xiao*, Peng Xu*, Sichun Xu*, Mengyuan Yan†, Andy Zeng*

*Robotics at Google, †Everyday Robots

Abstract: Large language models can encode a wealth of semantic knowledge about the world. Such knowledge could be extremely useful to robots aiming to act upon high-level, temporally extended instructions expressed in natural language. However, a significant weakness of language models is that they lack real-world experience, which makes it difficult to leverage them for decision making within a given embodiment. For example, asking a language model to describe how to clean a spill might result in a reasonable narrative, but it may not be applicable to a particular agent, such as a robot, that needs to perform this task in a particular environment. We propose to provide real-world grounding by means of pretrained skills, which are used to constrain the model to propose natural language actions that are both feasible and contextually appropriate. The robot can act as the language model's "hands and eyes," while the language model supplies high-level

A. LLMs for Task Planning: SayCan Approach

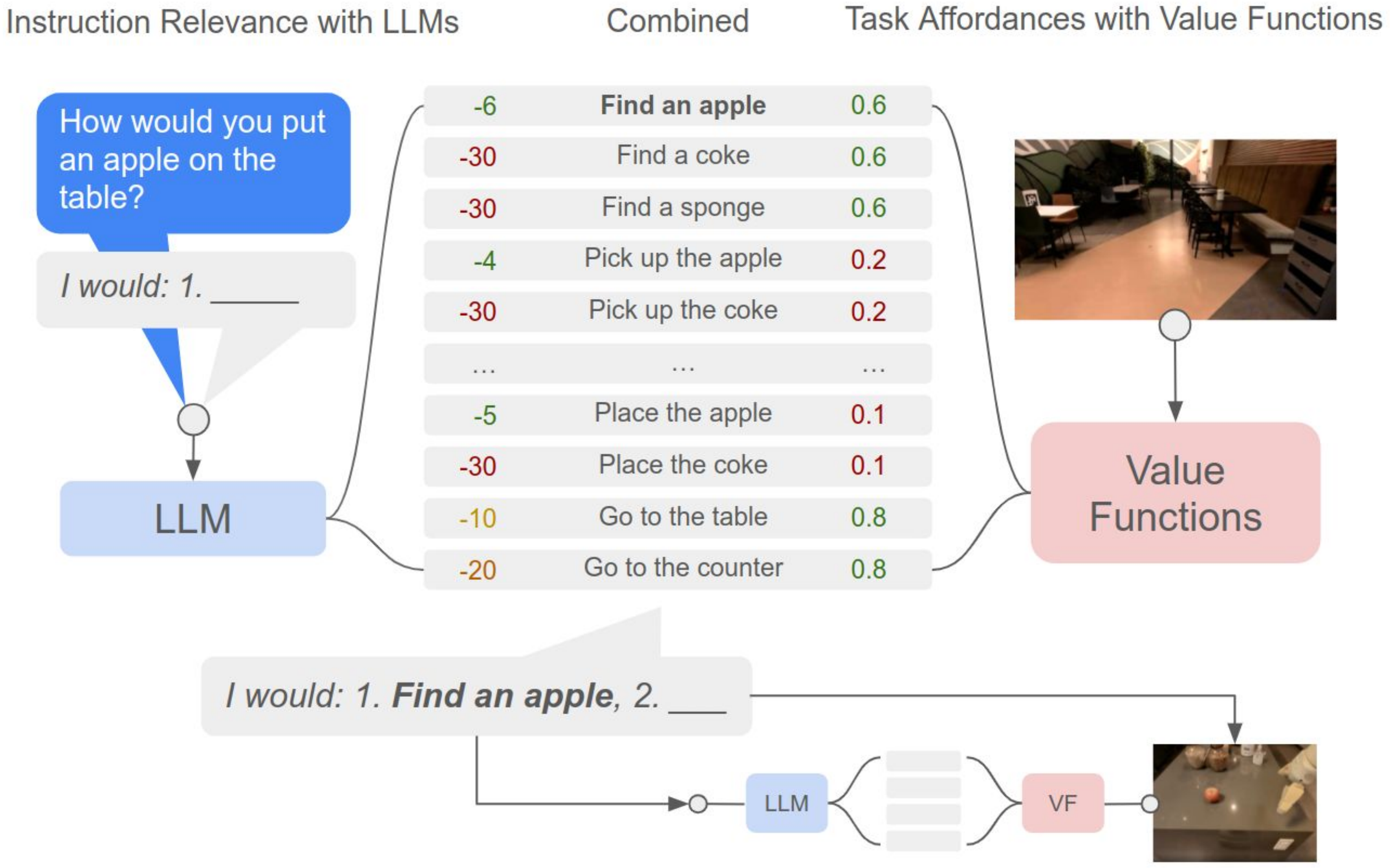
Bridge the gap between abstract language understanding and concrete, contextually appropriate robotic actions:

- **Pre-trained Behaviors:** The system uses pretrained robotic behaviors to provide grounding for the language model.
- **Robot as "Hands and Eyes":** The robot acts as the physical interface for the language model, providing real-world context and execution capabilities.
- **Language Model for High-Level Knowledge:** The language model supplies high-level semantic knowledge about tasks and procedures.
- **Value Functions for Grounding:** Value functions associated with low-level tasks provide the necessary grounding to connect the language model's knowledge to the physical environment.

THE UNIVERSITY *of* ADELAIDE

Key Components of SayCan

1. **Language Model for Task Decomposition:**
   - The language model breaks down high-level instructions (e.g., "I spilled my drink, can you help?") into sub-tasks.
   - It generates a sequence of steps, structured as a dialogue between the user and the robot.
2. **Skill Scoring:**
   - The language model scores the likelihood of each available skill contributing to the completion of the high-level instruction.
   - This helps in selecting contextually appropriate actions.
3. **Affordance Functions:**
   - Each skill has an associated affordance function (e.g., a learned value function).
   - This function quantifies the likelihood of the skill's successful execution in the current state.


Instruction Relevance with LLMs
Combined
Task Affordances with Value Functions
How would you put an apple on the table?
I would: 1. ______
LLM
Value Functions
I would: 1. ***Find an apple****, 2.* ___
LLM
VF

-6	**Find an apple**	0.6
-30	Find a coke	0.6
-30	Find a sponge	0.6
-4	Pick up the apple	0.2
-30	Pick up the coke	0.2
…	…	…
-5	Place the apple	0.1
-30	Place the coke	0.1
-10	Go to the table	0.8
-20	Go to the counter	0.8

Key Components of SayCan

4. **Probability Combination:**
   - combines two probabilities:

     a) The language model's probability of a skill being useful for the instruction.

     b) The value function's probability of successfully executing the skill.

5. **Iterative Skill Selection and Execution:**
   - The system selects the skill with the highest combined probability.
   - The selected skill is executed on the robot.
   - The executed skill is appended to the robot's response in the dialogue.
6. **Continuous Planning:**
   - After each skill execution, the process repeats:
     - The language model is queried again with the updated context.
     - New probabilities are calculated.
     - The next most appropriate skill is selected.
7. **Termination:**
   - The process continues until the language model outputs a termination step (e.g., "Done").

THE UNIVERSITY of ADELAIDE

Demo

https://say-can.github.io/

Challenges and Limitations

- **Computational Intensity:** Running large language models in real-time can be challenging.
- **Generalization:** While LLMs have broad knowledge, they may struggle with very specific or novel scenarios.
- **Safety Considerations:** Ensuring that the generated plans are always safe and appropriate is crucial.

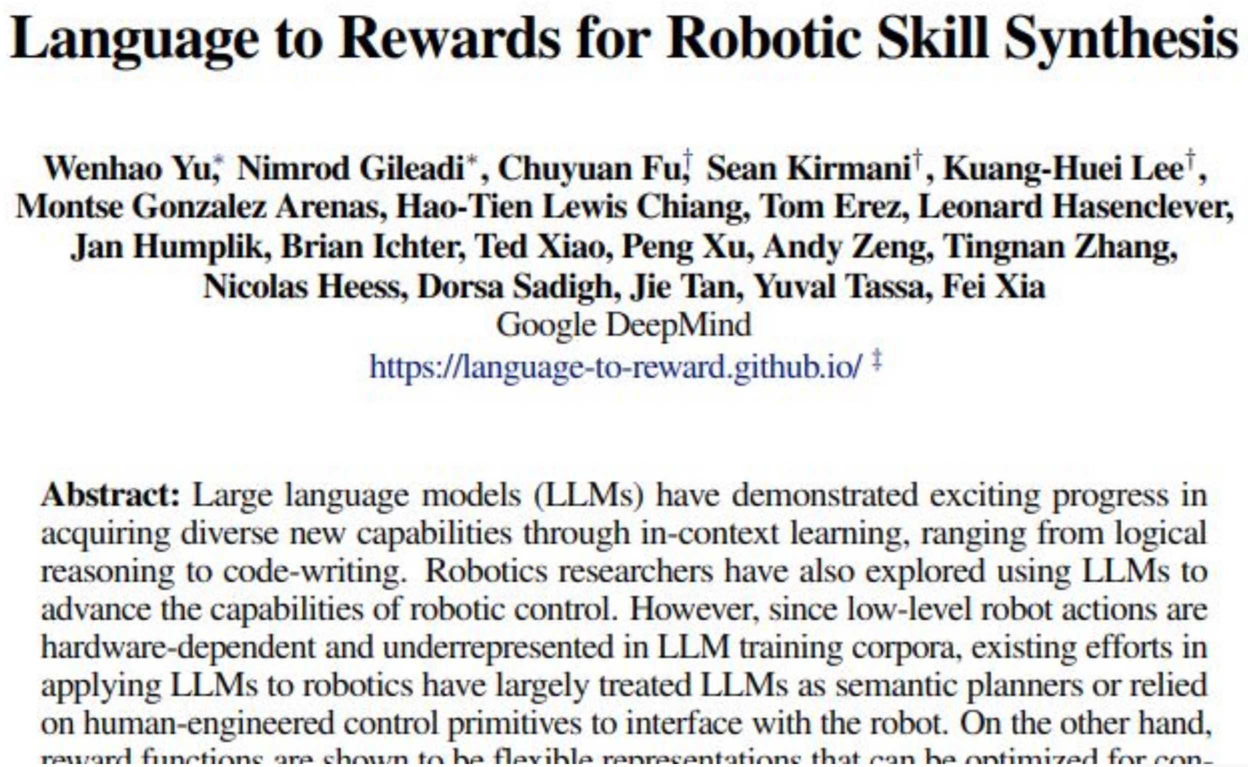
THE UNIVERSITY *of* ADELAIDE

Action Generation with FMs

A. LLMs for task planning: SayCan approach

B. Reward function generation: Language to Rewards (L2R)

C. Action generation: RT-1 and RT-2 models



Language to Rewards for Robotic Skill Synthesis

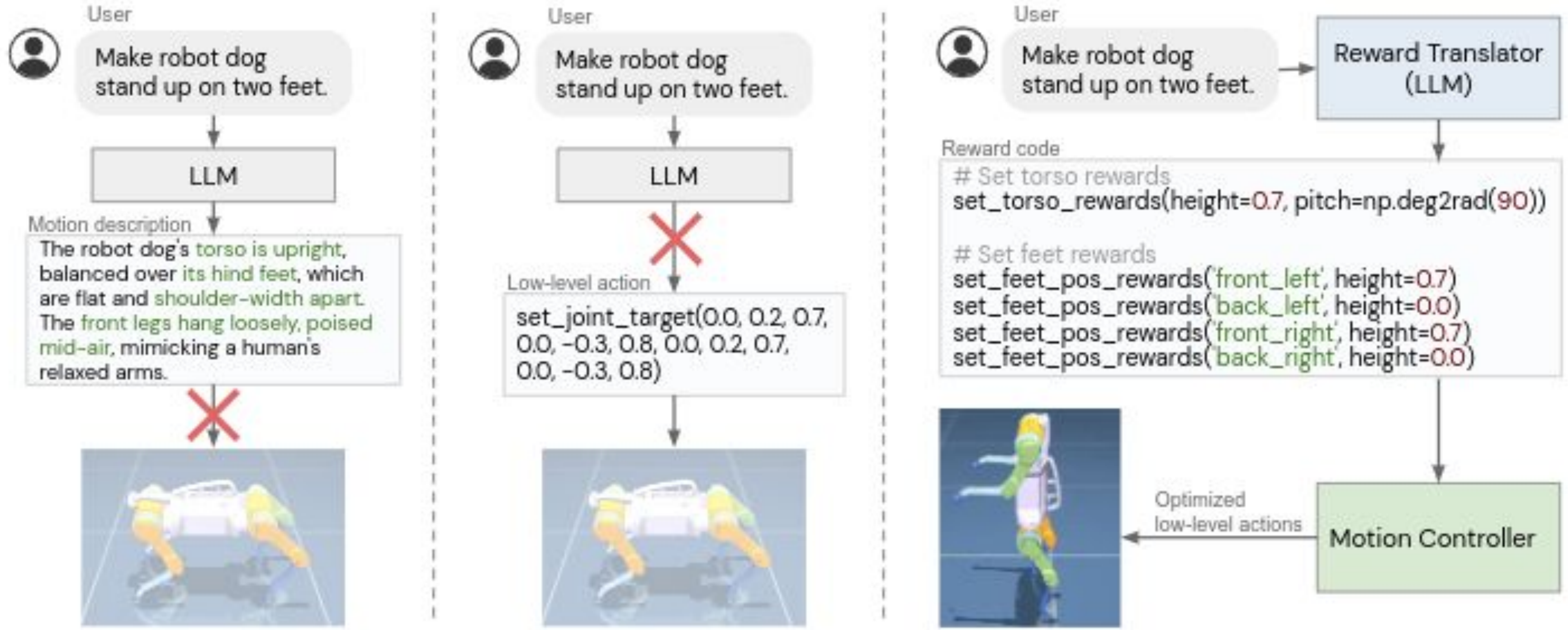
Wenhao Yu[*], Nimrod Gileadi[*], Chuyuan Fu[†], Sean Kirmani[†], Kuang-Huei Lee[†], Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, Fei Xia
Google DeepMind
https://language-to-reward.github.io/ [‡]

Abstract: Large language models (LLMs) have demonstrated exciting progress in acquiring diverse new capabilities through in-context learning, ranging from logical reasoning to code-writing. Robotics researchers have also explored using LLMs to advance the capabilities of robotic control. However, since low-level robot actions are hardware-dependent and underrepresented in LLM training corpora, existing efforts in applying LLMs to robotics have largely treated LLMs as semantic planners or relied on human-engineered control primitives to interface with the robot. On the other hand, reward functions are shown to be flexible representations that can be optimized for con-

C. Reward Function Generation:
Language to Rewards (L2R)

- **Low-level robot actions** are hardware-dependent and underrepresented in LLM training data.
- Existing approaches often use **LLMs only as high-level semantic planners** or rely on **pre-engineered control primitives**, limiting their potential in robotic control.
- L2R is an approach that uses large language models to generate reward functions for robotic tasks based on natural language descriptions.

THE UNIVERSITY *of* ADELAIDE



Make robot dog stand up on two feet.

The robot dog's torso is upright, balanced over its hind feet, which are flat and shoulder-width apart. The front legs hang loosely, poised mid-air, mimicking a human's relaxed arms.

Make robot dog stand up on two feet.

```
set_joint_target(0.0, 0.2, 0.7,
0.0, -0.3, 0.8, 0.0, 0.2, 0.7,
0.0, -0.3, 0.8)
```

Make robot dog stand up on two feet.

```
# Set torso rewards
set_torso_rewards(height=0.7, pitch=np.deg2rad(90))

# Set feet rewards
set_feet_pos_rewards('front_left', height=0.7)
set_feet_pos_rewards('back_left', height=0.0)
set_feet_pos_rewards('front_right', height=0.7)
set_feet_pos_rewards('back_right', height=0.0)
```

Figure 1: LLMs have some internal knowledge about robot motions, but cannot directly translate them into actions (left). Low-level action code can be executed on robots, but LLMs know little about them (mid). We attempt to bridge this gap, by proposing a system (right) consisting of the Reward Translatorthat interprets the user input and transform it into a reward specification. The reward specification is then consumed by a Motion Controller that interactively synthesizes a robot motion which optimizes the given reward.

The Challenge of Reward Specification

- In reinforcement learning, the reward function defines the goal of the task.
- Manually designing reward functions can be challenging, especially for complex tasks.
- There's often a gap between how humans describe tasks and how they need to be specified for robots.

THE UNIVERSITY *of* ADELAIDE

How L2R Works

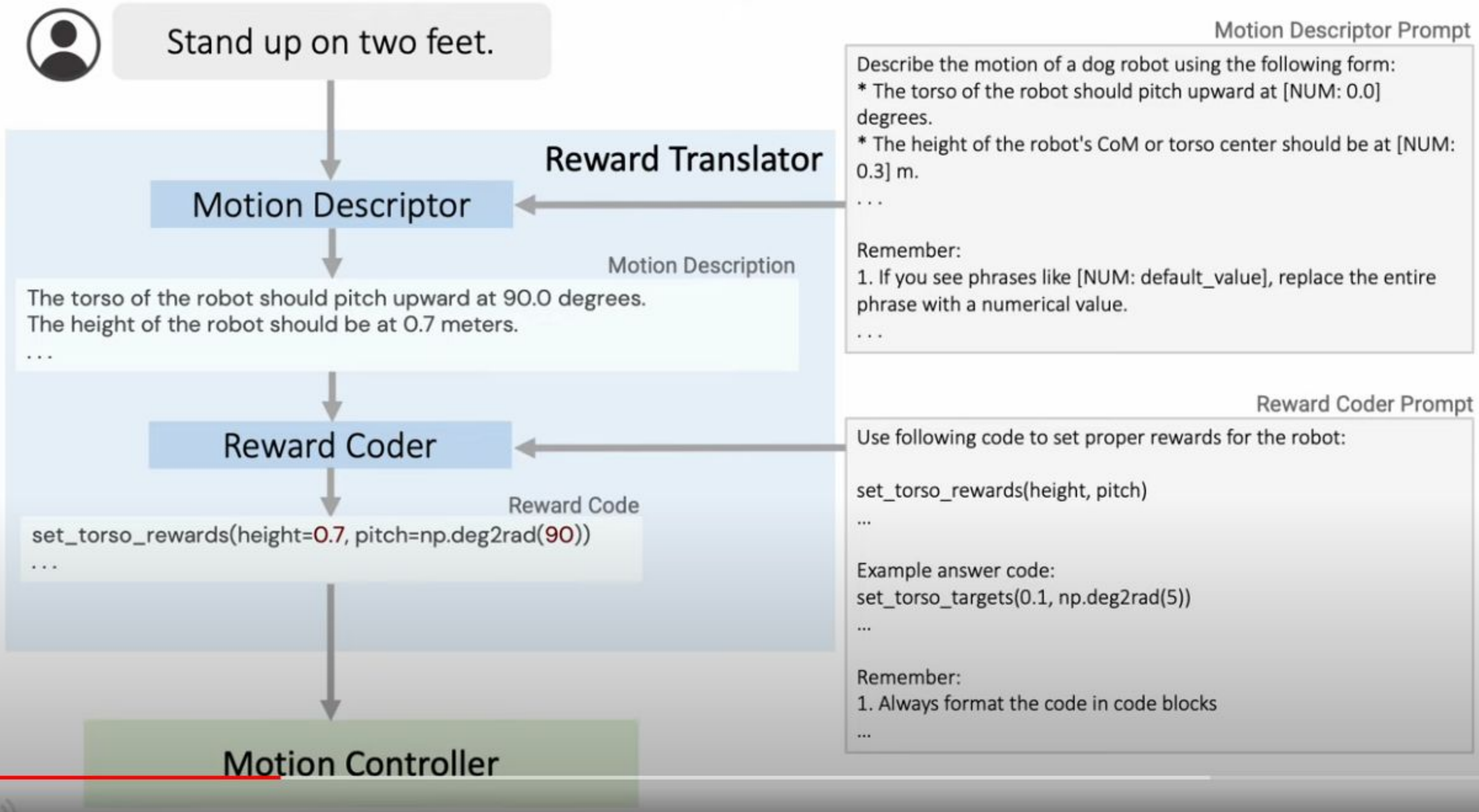
a) **Natural Language Input:** The process starts with a natural language description of the desired task or behavior.

b) **LLM Processing:** An LLM interprets the language description and generates a structured representation of the task.

c) **Reward Translator:** This component converts the LLM's output into a mathematical reward function.

It maps language concepts to specific, measurable aspects of the robot's state and actions.

d) **Reward Function Output:** The result is a formal reward function that can be used in reinforcement learning algorithms.



Stand up on two feet.

Motion Description:

The torso of the robot should pitch upward at 90.0 degrees.
The height of the robot should be at 0.7 meters.
. . .

Reward Code:

```
set_torso_rewards(height=0.7, pitch=np.deg2rad(90))
. . .
```

Motion Descriptor Prompt:

Describe the motion of a dog robot using the following form:
* The torso of the robot should pitch upward at [NUM: 0.0] degrees.
* The height of the robot's CoM or torso center should be at [NUM: 0.3] m.
. . .

Remember:
1. If you see phrases like [NUM: default_value], replace the entire phrase with a numerical value.
. . .

Reward Coder Prompt:

Use following code to set proper rewards for the robot:

set_torso_rewards(height, pitch)
...

Example answer code:
set_torso_targets(0.1, np.deg2rad(5))
...

Remember:
1. Always format the code in code blocks
...

Advantages of L2R

- **Flexibility:** Can handle a wide range of task descriptions.
- **Interpretability:** The language-based approach makes it easier for humans to understand and verify the reward functions.
- **Scalability:** Can leverage improvements in LLMs to handle more complex task specifications.

Challenges and Considerations

- **Ambiguity:** Natural language can be ambiguous, which can lead to incorrect reward functions.
- **Safety:** Ensuring that generated reward functions don't lead to unsafe behaviors is crucial.
- **Generalization:** The system needs to handle a wide variety of task descriptions and environments.

THE UNIVERSITY of ADELAIDE

Action Generation with FMs

A. LLMs for task planning: SayCan approach

B. Reward function generation: Language to Rewards (L2R)

C. Action generation: RT-1 and RT-2 models

C. Action generation: RT-1 and RT-2 models

- RT stands for 'Robotic Transformer'
- A step closer towards end-to-end learning for robotic control.
- Aim to directly map language instructions and visual inputs to robot actions

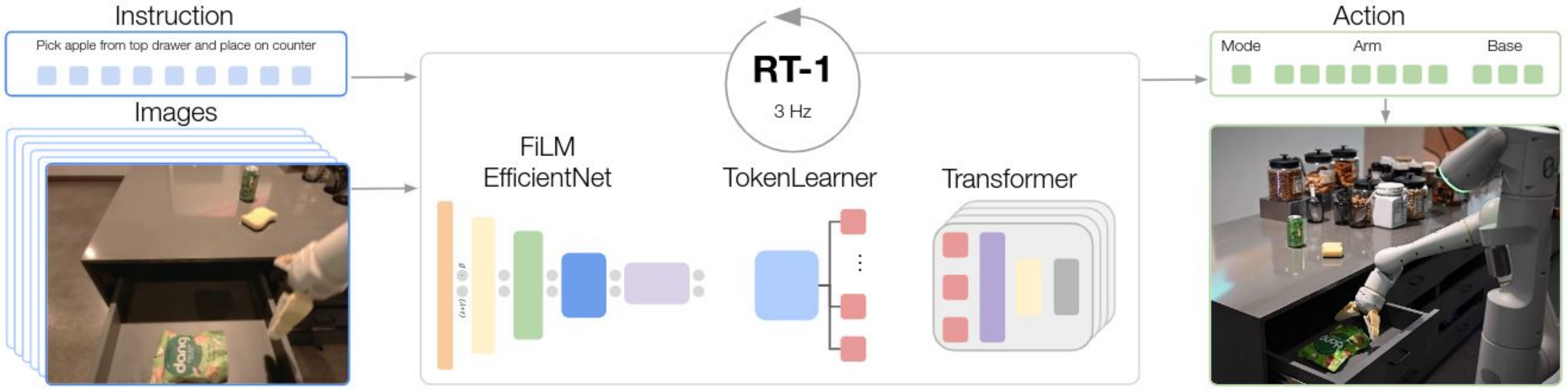
RT-1

- Trained on over 130,000 episodes of robot interaction data
- Data includes a wide variety of objects, tasks, and environments
- Uses imitation learning to mimic expert demonstrations

Input and Output:

- **Inputs:** RGB images, language instructions, and robot state
- **Outputs:** Low-level control commands for the robot

THE UNIVERSITY *of* ADELAIDE



The **actions** consist of **seven dimensions** for the **arm movement (x, y, z, roll, pitch, yaw, opening of the gripper), three dimensions for base movement (x, y, yaw)** and a **discrete dimension to switch between three modes**:

- controlling the arm,
- the base,
- or terminating the episode.

RT-1 performs closed-loop control and commands actions **at 3 Hz** until it either yields a "terminate" action or hits a pre-set time step limit.

THE UNIVERSITY of ADELAIDE

RT-2: Vision-Language-Action Models

Advancements over RT-1:

- Introduces the concept of **Vision-Language-Action (VLA)** models
- Leverages large-scale vision-language models pre-trained on internet data
- Aims for even greater generalization and flexibility

Key Ideas:

- Treats robot actions as another form of language
- Uses a large vision-language backbone (like PaLM-E) fine-tuned for robotics
- Enables more complex reasoning about tasks and environments

RT-2

Architecture:

- Combines visual and language inputs to generate action tokens using transformer-based models like PaLI-X and PaLM-E.

Action Representation:

- Actions are represented as token strings, making them compatible with natural language processing techniques.

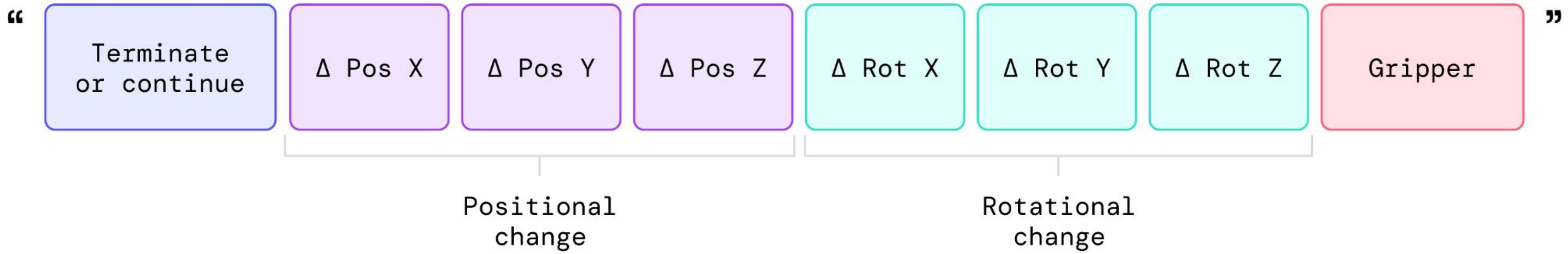
Training:

- Co-fine-tuned on both web-scale data and robotic datasets, enhancing generalization and semantic understanding.

Capabilities: Performs multi-stage reasoning, symbol understanding, and human recognition.

Robot actions as text strings. An example of such a string could be a sequence of robot action token numbers:

"1 128 91 241 5 101 127 217".



Challenges and Limitations

Computational Requirements:

- Especially for RT-2, the large model size poses challenges for real-time operation

Data Requirements:

- Both models require large amounts of diverse robotics data

Safety Concerns:

- Ensuring safe operation in all scenarios remains a challenge

Interpretability:

- Understanding the decision-making process of these complex models can be difficult

THE UNIVERSITY *of* ADELAIDE

Benefits and challenges of using FM in robotics

- Enhanced Understanding of Complex Environments
- More Natural Human-Robot Interaction
- Improved Generalization and Adaptability
- Rich Semantic Understanding
- Flexible Task Specification

- Increased Computational Complexity
- Alignment and Synchronization
- Handling Inconsistencies Between Modalities
- Transfer to Physical Action
- Data Scarcity and Diversity
- Interpretability and Debugging

Promising future directions

- Continual/lifelong learning
- Improved grounding and prompting for robotics
- Enhanced dynamics and world model learning
- Better modeling of uncertainty and safety

THE UNIVERSITY *of* ADELAIDE

Q&A

adelaide.edu.au/aiml

Australian Institute for Machine Learning