Foundational Models for Robotics:
Removing the Engineer from the Loop

Rogerio Bonatti
April 27, 2023
Speaker Bio: Rogerio Bonatti

Senior Researcher

PhD in Robotics, School of Computer Science

Research Internships, FAIR and MSR
Research projects:

Multi-modal robot control (vision + language):

Generative models for decision-making:

Autonomous drone cinematography:

Learning emotional camera control:

Visuo-motor representation learning:
Large models are revolutionizing generative AI

Text generation
[ GPT-X, LLaMA ]

Input Prompt: Recite the first law of robotics

Output: A robot may not injure a human being or, through inaction, allow a human being to come to harm.

Text-to-video
[NVIDIA Picasso]

Conversation
[ChatGPT]

ChatGPT
"a corgi playing a flame throwing trumpet"

"Turn it into a still from a western"

Text-to-3D
[Point-e, NVIDIA Picasso]

Image generation
[DALL-E, Stable Diffusion]

Image editing
[InstructPix2Pix, Prompt-to-Prompt]
But generative AI is just starting to revolutionize decision-making:

Human-robot interfaces: [ChatGPT for Robotics]

Human-computer interfaces [video from Adept AI]
Goal: use large models to empower any user with the capabilities of generative AI

- Human-model interaction
- Multi-modal context (text, image, actions)
- Generalize one model for multiple tasks
Goal: use large models to **empower any user** with the capabilities of generative AI

- Close the loop between the human user and the model
- Fusing multi-modal information in I/O
- Generalizing a single model to multiple tasks

**Text**

Goal: use large models to empower any user with the capabilities of generative AI
What is the future of human-computer interfaces?

Remove the engineer from the loop. Bring in the user.

Interfaces today: engineer in the loop

Objective

Deploy, improve

Goal with ChatGPT: user on the loop

Prompt + Objective

Any task, any robot

Inspect the shelves in a lawnmower pattern

Arrange the colored blocks to form the Microsoft logo

Find where I can warm up my lunch

Large model

Deploy, improve

Any task, any robot

Large model

Deploy, improve

Any task, any robot
I want the robot to cook an omelet and serve it to my grandpa

*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.

**User on the loop:** iterate on solution quality and safety

**Execute!**

**Design principles for effective ChatGPT usage:**

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
   - Consider you are a home assistant robot. Your goal is to prepare an omelet 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;

3. Output python code with the sequence of steps that achieves your objective.
Deployment with the user in the loop
When a conversation is needed for complex tasks

ChatGPT for Robotics: Design Principles and Model Abilities

Sai Venkatesh, Rogerio Burtini, Arthur Bucker, Ashish Kapoor
Microsoft Autonomous Systems and Robotics Research
Multi-modal emerging properties
Hallucination over colors, shapes and geometry
Community support: prompt library and simulator

ChatGPT Prompting Guides & Examples

The list below contains links to the different robotics categories and their corresponding prompt examples. We welcome contributions to this repository to add more robotics categories and examples. Please submit prompt examples to the Discussions page, or submit a pull request with your category and examples.

- **Embodied agent**
  - ChatGPT - Habitat, closed loop object navigation 1
  - ChatGPT - Habitat, closed loop object navigation 2
  - ChatGPT - AirSim, object navigation using RGBD

- **Aerial robotics**
  - ChatGPT - Real robot: Tello deployment | Video Link
  - ChatGPT - AirSim turbine Inspection | Video Link
  - ChatGPT - AirSim solar panel Inspection
  - ChatGPT - AirSim obstacle avoidance | Video Link

- **Manipulation**
  - ChatGPT - Real robot: Picking, stacking, and building the MSFT logo | Video Link
  - ChatGPT - Manipulation tasks

- **Spatial-temporal reasoning**
  - ChatGPT - Visual servoing with basketball

Open-sourced AirSim simulator with ChatGPT API:
Goal: use large models to **empower any user** with the capabilities of generative AI

Close the loop between the human user and the model

Fusing multi-modal information in I/O

Generalizing a single model to multiple tasks
Goal: reshape robot trajectories based on user language inputs

- Natural Language
- Mouse-keyboard
- Programming
- Drawing
- Kinesthetic teaching
- Touch-screens

ROBOT, STAY FURTHER AWAY FROM THE GLASSES!!
ROBOT, STAY FURTHER AWAY FROM THE GLASSES!!
Mathematical definitions

• Original trajectory: \( \xi_o : [-1, 1] \to \mathbb{R}^4 \ \{ (x_1, y_1, z_1, v_1), \ldots, (x_N, y_N, z_N, v_N) \} \)

• Objects in the scene: \( \mathcal{O} = \{ O_1, \ldots, O_M \} \)
  - Position \( \hat{P}(O_i) \in \mathbb{R}^3 \)
  - Image label: \( I(O_i) \)

• Language input: \( L_{in} \)

• Learning objective: \( \xi_{mod} = f(\xi_o, L_{in}, \mathcal{O}) \)
System overview
It is **expensive** to collect real-world trajectory correction examples. We use **procedural** data generation.
Key idea: use frozen LLM embeddings to compensate for low synthetic dataset variations

Advantages:

- Richer representations
- Less training data required
- More robust to vocabulary variations
Semantic sentence embedding

Semantic meaning

Textual embedding vector

BERT (distilled)

CLIP (text enc.)

CLIP (img enc.)

Similarity vector

Target object information

$L_{in}$
NL interaction

‘Stay far from the bottle’

$L(O_i)$
Object image inputs
LaTTe: Language Trajectory Transformer
User study: comparison between interfaces

Natural Language

Kinesthetic Teaching

Parameter tuning

Drawing
Results: multi-model model is effective, precise and intuitive

<table>
<thead>
<tr>
<th>Interface</th>
<th>Satisfied</th>
<th>Easy to use</th>
<th>Safety</th>
<th>Natural</th>
<th>Predictable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural Language (ours)</td>
<td>90</td>
<td>92</td>
<td>92</td>
<td>98</td>
<td>72</td>
</tr>
<tr>
<td>Kinesthetic Teaching</td>
<td>90</td>
<td>88</td>
<td>88</td>
<td>78</td>
<td>96</td>
</tr>
<tr>
<td>Drawing</td>
<td>88</td>
<td>74</td>
<td>100</td>
<td>80</td>
<td>88</td>
</tr>
<tr>
<td>Parameter tuning</td>
<td>62</td>
<td>58</td>
<td>88</td>
<td>62</td>
<td>48</td>
</tr>
</tbody>
</table>

User ratings collected in the user study

<table>
<thead>
<tr>
<th>Av. Iterations</th>
<th>Success rate (%)</th>
<th>Avr. Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.33</td>
<td>100</td>
<td>81</td>
</tr>
<tr>
<td>1.78</td>
<td>56</td>
<td>139</td>
</tr>
<tr>
<td>1.89</td>
<td>65</td>
<td>120</td>
</tr>
<tr>
<td>4.00</td>
<td>92</td>
<td>284</td>
</tr>
</tbody>
</table>

User study metadata
Goal: use large models to empower any user with the capabilities of generative AI

- Close the loop between the human user and the model
- Fusing multi-modal information in I/O
- Generalizing a single model to multiple tasks
General Pre-Training for Decision-Making

Language models

Input:
Recite the first law of robotics

Output: Sequence of words
A robot may not injure a human being or, through inaction, allow a human to be harmed...

GPT-3

Autonomous systems models

Input:
Images/Depth LiDAR Language: ‘Go fast!’

New sensor inputs
Environment

Output: Sequence of actions

\[
\begin{align*}
\theta_0 &= 10^\circ, V_0 = 0.5m/s \\
\theta_1 &= 15^\circ, V_1 = 0.1m/s \\
\theta_2 &= 18^\circ, V_2 = -0.7m/s
\end{align*}
\]

PACT

Perception-Action Causal Transformer

Actuators
Traditional Autonomous Systems Pipeline

Agent ➔ Observation ➔ Localization ➔ Mapping ➔ Navigation

Large task-specific modules
Foundational models for decision-making: Deployment in multiple downstream tasks
Pre-training network:
Downstream task networks:

Causal Transformer \times N

Localizatio

Mapping

Pretrained Blocks

Causal Transformer \times N

Pretrained Blocks

\begin{align*}
\dot{p}_{t-1} & \rightarrow t \\
\dot{p}_{t} & \rightarrow t+1 \\
\rightarrow & \\
\end{align*}

\begin{align*}
\dot{m}_{t-1} & \rightarrow t+1 \\
\rightarrow & \\
\end{align*}
Pre-training data collection for MuSHR: millions of perception-action pairs
Pre-training data collection for Habitat: millions of perception-action pairs
What does the pre-trained model learn?
PACT applied towards multiple downstream tasks

MuSHR vehicle:
Perception input: LiDAR
Action input: wheel angles

PACT
(Perception-Action Causal Transformer)

Common Representation

GT pose
Prediction
Local Mapping
Real-World Navigation
Localization
Simultaneous navigation, localization and mapping model deployment in simulation

Reference frame for localization (zero) → Estimated vehicle pose (integration over time) → Decoded local map
MuSHR deployment **in the wild:**

Pre-trained model overcomes sim-to-real gap:
- Model dynamics
- Actuator and processing delays
- Sensor noise and imperfect LiDAR returns (e.g. glass surfaces)
Use large models to **empower any user** with the capabilities of generative AI

- Close the loop between the human user and the model
- Fusing multi-modal information in I/O
- Generalizing a single model to multiple tasks
Thanks to collaborators:

- **Microsoft:**
  - Ashish Kapoor
  - Sai Vemprala
  - Shuang Ma
  - Shuhang Chen
  - Felipe Frujeri

- **TUM, Germany:**
  - Arthur Bucker
  - Luis Figueredo
  - Sami Haddadin
Q&A + discussions