Agentic Language-Grounded Adaptive Robotic Assembly

Nicholas Cote
Autodesk Research
nick.cote@autodesk.com

Jaimyn Drake
University of California, Berkeley
jaimyndrake@berkeley.edu

Sachin Chitta
Autodesk Research
sachin.chitta@autodesk.com

Abstract

High-mix, low-volume manufacturing requires systems that can adapt to changing parts and processes. We present an agentic, language-grounded approach for robotic assembly that employs a team of Large Language Models and Vision-Language-Action Models to adapt to parts of similar kind and perform corrective actions during assembly. Given a high-precision wheel-on-axle insertion task, we demonstrate that our agentic approach outperforms a single-model and generalizes to out-of-distribution parts and grasps without changing the nominal assembly process.

1. Introduction

As manufacturing shifts towards high-mix, low-volume production, the need for robotic systems to generalize and adapt has grown. Traditional systems, while fast and precise, still require task-specific programming and perception, making it hard to adapt to new or even slightly varied designs or processes during deployment.

Large Language Models (LLMs) like ChatGPT and LLaMA, known for their ability to generalize across diverse tasks could help us address these challenges [1, 10, 12, 18]. Moreover, Vision-Language-Action Models (VLA), which ground visuomotor control in language, enable robots to interpret and react appropriately to visual inputs.

We wonder: *Can a team of these models learn what a manufacturing process looks like and enable robots to adapt when it varies on the factory floor*?

In this work, we present an agentic, language-grounded approach for robotic assembly that employs a team of LLMs and VLAs to correct offsets during an high-precision *wheel-on-axle* insertion task, shown in Figure 1. In it, the wheel is picked-up, transported to an initial *aligned* pose with the axle, inserted onto the axle, then released. Our experiments show our approach outperforms a single-model in generalizing across in-distribution (ID) and out-of-distribution (OOD) parts and grasps without changing the nominal assembly process.

Figure 1. The gripper in Vertical configuration, approaches the Shark wheel at the start of a wheel-on-axle insertion experiment.

Our contributions include:

1. *An agentic architecture for robotic assembly* with task-specific language agents for distance estimation, direction estimation, and programming.
2. *Industrial application of LLM/VLA* for precise, adaptive control across varying parts and grasps.
3. *Experiments showing this approach outperforms single-models* in terms of adaptivity and task completion.

2. Related Work

Massive, diverse, and domain-specific datasets like Open-X Embodiment and DROID have laid the groundwork for robotics foundation models [8, 13]. Projects like LERF and VoxPoser show that grounding perception in language could enable open-ended, adaptive control [6, 7]. Multi-modal VLAs like OpenVLA and RT-2 show that such groundings could enable visuomotor control of robots, while models like Gemini generalize these capabilities to even more complex and nuanced manipulation tasks [2, 9, 16]. Meanwhile, simple prompting techniques have shown to improve these outcomes: breaking problems down into intermediate steps [20]; leveraging more diverse, rather than larger, datasets to zero-shot new tasks [3]; refining processes through iterative,

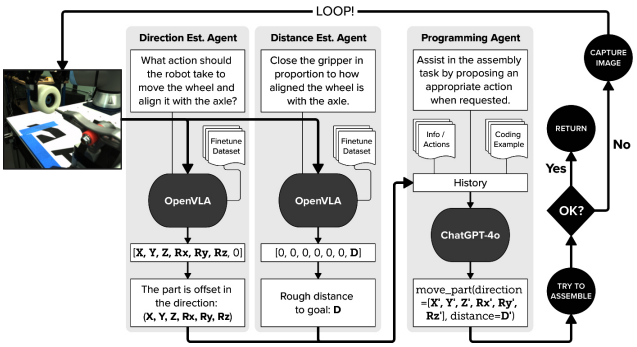

Figure 2. Simplified control-flow diagram showing main loop, agents, prompts, and assembly logic.

verbal feedback [4, 11, 14]; and using code-like specifications for actions and objects [15]. Agentic systems, where multiple models collaborate to solve a problem, also show promise for complex, nuanced tasks [5, 19, 21].

Building on these techniques, we create a language-grounded, agentic system for adaptive robotic assembly, making use of OpenVLA for adaptive visuomotor control and ChatGPT-4o for dynamic task control during assembly. In Section 3, we describe the architecture of our approach, language agents, correction algorithm, as well as our approach to dataset generation and fine-tuning. In Section 4, we present experiments wherein our agentic architecture is applied to a wheel-insertion task.

3. Approach

Our system uses three specialized language model *agents* for distance estimation, direction estimation, and programming that work together to adaptively insert a part. After a failed insertion, the system analyzes visual feedback to compute and execute corrective actions proposed by the agents, repeating until the part is successfully inserted (or after a maximum number of attempts is reached), as shown in Figure 2.

3.1. Dataset Generation

Although OpenVLA is trained on the Open X-Embodiment dataset, differences in coordinate systems, viewing perspectives, and robot architectures mean it can't zero-shot without fine-tuning. To gather a fine-tuning dataset, we simulate correcting random offsets, $[X, Y, Z, Rx, Ry, Rz]$, generated using three schemes: 40% involve random translation of Y and Z by $[-20, 20]$ mm; 20% involve random rotation of Rx, Ry, and/or Rz by $[-\frac{\pi}{8}, \frac{\pi}{8}]$ radians; and the rest combine both.

After the wheel is offset, it's gradually moved back to the aligned pose (generated using the ASAP algorithm [17]) through a series of interpolated *waypoints*. For each, we save the episode number, waypoint index, offset, and image

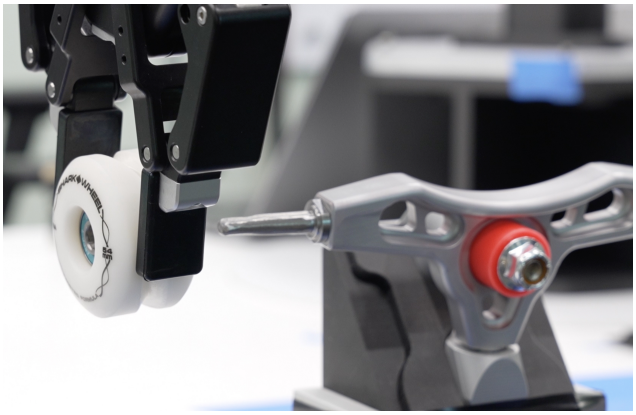

Figure 3. Close-up of fine-tuning dataset generation, showing Small wheel being manipulated by gripper near axle.

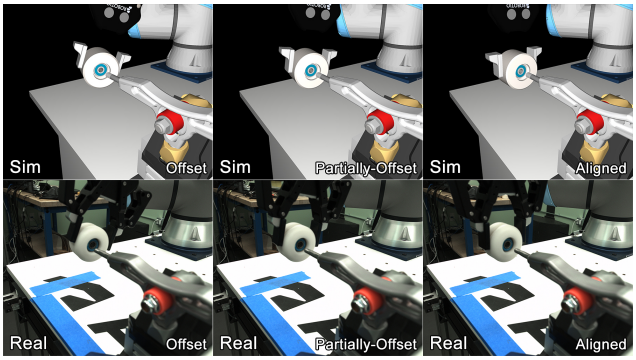

Figure 4. Typical images used for fine-tuning, showing Simulated and Small wheels at Offset, Waypoint, and Aligned poses.

of the assembly scene (see Figure 4). This is done for a *Simulated* wheel and four real-world wheels (*Base*, *Large*, *Shark*, and *Small*), shown in Figure 5.

Building off the OpenVLA release checkpoint, a fine-tuned distance or direction estimation model can be generated using this dataset. At inference time, we provide the model the same prompt used for fine-tuning and an image of the current assembly scene (captured from same viewpoint used for fine-tuning).

3.2. Distance Estimation Agent

The *Distance Estimation Agent* outputs the wheel's distance from the aligned pose given an image of the assembly scene. A fine-tuned OpenVLA model outputs $[0, 0, 0, 0, 0, 0, D]$, where D represents the estimated distance – repurposing the model's original gripper-action output. The agent also outputs a *retry flag* to trigger reinsertion, raised after at least 5 cycles when the distance is minimal or constant over multiple cycles.

For fine-tuning, the current offset of each waypoint is converted to a distance using a scaled, normalized version

of the 6D $L1$ distance. This is stored in the output vector, then paired with the corresponding image and prompt to create an input-output pair. Most models in our experiments were fine-tuned on 1000 episodes (11000 entries) for one wheel, except the *All+* model, which was fine-tuned on the Simulated wheel and all four ID wheels (55000 entries)

3.3. Direction Estimation Agent

The *Direction Estimation Agent* outputs the directional offset of the wheel from the aligned pose. The fine-tuned model outputs $[X, Y, Z, Rx, Ry, Rz, 0]$, where the first six components represent the estimated offset. While their magnitudes are usually extreme, their signs (1, -1, or 0) correspond consistently with the correct direction and are used to guide corrective movement.

Like the Distance Estimation Agent, the initial offset of each waypoint is scaled, stored in the output vector, and paired with its image and prompt for fine-tuning. We then train three types of model: (1) real-world data over 1000 episodes per ID wheel: *Base*, *Large*, *Shark*, *Small*; (2) combined real and simulated data over 1000 episodes per ID wheel and the Simulated wheel: *Base+*, *Large+*, *Shark+*, *Small+*; a large, diverse model trained on 5000 episodes across all wheels: *All+*.

3.4. Programming Agent

The *Programming Agent*, built on ChatGPT-4o, acts as an adaptive controller synthesizing outputs from the Distance and Direction Estimation Agents into executable code. Unlike the other agents, it relies on prompt engineering instead of fine-tuning. At initialization, we provide it with the task context, coding references and examples, and behavioral guidelines. It can also control the robot, capture images, and talk to other agents.

Typically, the agent moves the wheel along the estimated direction, scaled by the estimated distance. However, it can adjust this behavior dynamically by reversing the direction, reducing the scale of movements over time, and triggering early retries when no correction appears to be needed. These behaviors aren't canned, and emerge from ChatGPT's ability to reason about its task, context, status, and goals.

4. Experiments

We evaluate accuracy and adaptivity of our approach for the wheel-on-axle insertion task described in Section 1, testing agents on an array of in-distribution (ID), out-of-distribution (OOD), and far-out-of-distribution (FOOD) parts and grasps.

We use two UR10e robots, a Robotiq 2F-140 gripper, and a Zivid 2+ depth camera. A partially-assembled skateboard truck is mounted in a fixture that allows ± 2 mm positional variation. The wheels have a 8 mm hole and the tapered axle allows an insertion tolerance of ± 0.75 mm, as shown in Figure 3.

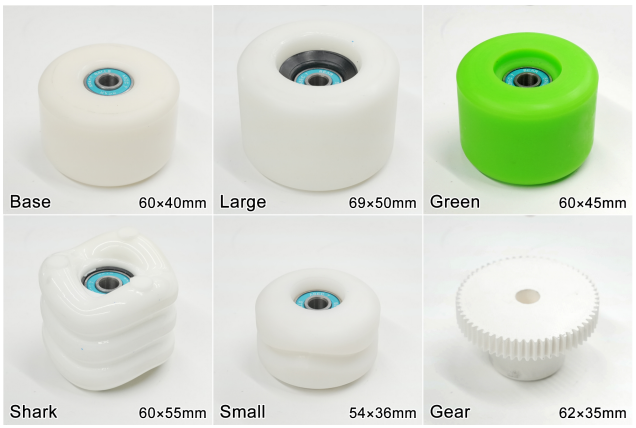

Figure 5. Side-by-side comparison of ID (Base, Large, Shark, Small) and OOD (Green, Gear) wheels used in experiments.

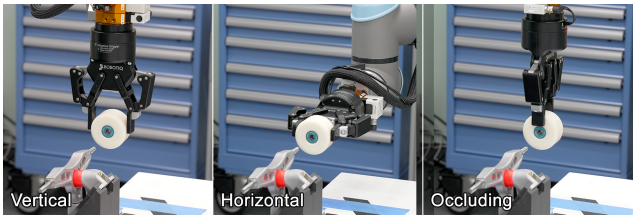

Figure 6. Side-by-side comparison of ID (Vertical) and OOD (Horizontal, Occluding) grasps used in experiments.

4.1. Distance Estimation Experiments

To validate distance estimation models, we conduct 100 tests per wheel – including those not used for fine-tuning. In each, the wheel is moved to a random offset, and the model receives a text prompt and image of the scene to estimate distance.

In Table 1, we see a positive correlation (C) between actual and estimated distances. Models fine-tuned on a wheel show an average of 0.83 for that wheel and 0.64 for wheels they weren't fine-tuned. The model fine-tuned on all wheels, All+, shows an average correlation of 0.92, suggesting that a larger, more diverse dataset improves performance. We also correlate the order of distances in a set without outliers, Cq. Models fine-tuned on a single wheel show an average of 0.86 for that wheel and 0.70 for wheels they weren't fine-tuned. The estimates also appear to cluster in *tiers*, or discrete values rather than a continuous range. For example, the Shark model produced 21 tiers for the Shark wheel but only 12 for the Large wheel, suggesting it learns classifications of distance and uses coarser classifications for OOD parts.

4.2. Direction Estimation Experiments

For direction estimation we use the same testing approach for distance models but apply offsets along all axes: Y and Z in $[-20, 20]$ mm and Rx, Ry, and Rz in $[-\frac{\pi}{8}, \frac{\pi}{8}]$ radians. This insures the model will estimate direction, not just detect offset presence.

In Table 1, the All+ model performed comparably to single-wheel models, but estimates rotational offsets more often. On average, it estimates Ry and Rz offsets 69% and 72% of the time, as opposed to 8% and 11% for the single-wheel models. Both rarely estimate offsets in Rx, the axis of rotation about the axle, likely due to minimal visual cues.

Most models classify direction with more than 80% accuracy, so a more telling metric is how often they estimate a non-zero offset. The high frequency of the All+ model suggests that it's able to detect small rotations, which is useful in an iterative approach. Additionally, smaller + models perform nearly as well as the All+ model for translation and estimate Ry and Rz rotations 25% and 22% of the time, suggesting that simulated data indeed helps compensate for limited physical datasets.

Many of the wheels resemble those used in fine-tuning but are still considered OOD, allowing us to compare their zero-shot performance. Single-wheel models achieved 90% accuracy estimating Y and Z for OOD wheels, compared to 96% for ID wheels. Some models also performed well on specific OOD wheels (e.g. the Shark model on the Small wheel), while others struggled (e.g. the Base model on the Shark wheel).

4.3. Full Assembly Experiments

We also conduct full assembly experiments where the robot removes the wheel from the axle, applies a random Y and Z offset, and attempts reinsertion. Importantly, the agentic system is unaware of the initial offset or which wheel it's assembling.

In Table 2, the All+ model consistently performs well, achieving a perfect success rate for both agentic and single-agent approaches, while the Base+ and Shark+ models perform worse with average success rates of 56% and 81%. The agentic approach also performs well, achieving average success rate of 90%. By comparison, the single-agent approach generally struggles with OOD parts, for which it achieves an averages success rate of only 73%. This is most obvious for the Base+ model, which achieves success rates of 0% two OOD parts and only 10% for the third. On the flip side, the single-agent approach completes the task faster than the agentic approach by an average of 165 s and for ID wheels achieves a 95% success rate with 1.215 mm less error. This suggests the agentic approach is a good choice for adapting to OOD parts, rather than for fast and precise assembly of ID parts.

4.4. Assembly with FOOD Parts

A hypothesis in this work is that the language-grounding of these models might enable them to generalize to FOOD parts. We use the All+ model on two unseen parts – a Green wheel and a Gear – as shown in Figure 5.

Table 3 shows a clear distinction between the single-agent and agentic approaches in terms of adaptability and efficiency. For the Gear, both achieve 100% success rate, with the single-agent showing 0.10 mm greater accuracy and completing the task 158 s faster. For the Green wheel, the agentic approach achieves a 67% success rate while the single-agent fails entirely. Despite taking considerably longer and requiring more queries and attempts, the agentic approach keeps trying until it finds a solution. While both models performed well for the White Gear, performance dropped significantly for the Green wheel, suggesting that color affects the models' ability to identify and generalize relevant visual features.

4.5. Assembly with FOOD Grasps

We also hypothesize that the language-grounding of our models enables them to generalize to FOOD grasps. Using the All+ model and Base wheel, we compare the performance of two unseen grasps – Horizontal and Occluding – against the trained Vertical grasp (Figure 6).

The results in Table 4 show a noticeable performance change for the FOOD grasps. The difference in error between the ID and FOOD grasps for the single-agent is 1.14 mm for the Horizontal Grasp and 5.45 mm for the Occluding Grasp, which, for an industrial assembly task, is considerable. While the agentic approach had slightly lower success rate for the Horizontal Grasp, it was twice as likely to succeed for the Occluding Grasp, despite considerably large errors and a longer completion time. This reinforces the idea that the agentic approach offers benefits for tasks where the ability to complete a task outweighs speed and precision.

5. Conclusions

We presented a novel approach to robotic assembly using a team of specialized language-agents for perception and decision-making to complete an assembly task. Applied to a wheel-on-axle insertion task, the system was effective in correcting part offsets and generalizing to OOD parts and grasps. Its iterative design allows it to propose and evaluate multiple options before re-attempting assembly, making it slower but more adaptive than a single-model. This makes it well-suited for high-mix manufacturing and, with broader training data, we hope an approach like this could someday enable robots to make *anything*.

Agentic Language-Grounded Adaptive Robotic Assembly

Supplementary Material

Part	Model	Direction					Distance		
		Y	Z	Rx	Ry	Rz	C	Cq	Tr
Base	All+	97	97	100	100	82	0.91	0.90	34
	Base+	96	95	100	100	89	-	-	-
	Small+	84	94	100	100	100	-	-	-
	Shark+	86	94	0	100	70	-	-	-
	Large+	85	91	100	100	67	-	-	-
	Base	94	93	100	0	75	0.84	0.84	26
	Small	92	92	100	100	100	0.74	0.78	21
	Shark	93	91	0	100	93	0.47	0.62	17
	Large	92	93	100	100	83	0.68	0.71	27
Large	All+	96	95	100	100	75	0.93	0.94	29
	Base+	86	91	100	100	81	-	-	-
	Small+	93	96	80	100	40	-	-	-
	Shark+	93	90	0	100	96	-	-	-
	Large+	96	98	0	97	75	-	-	-
	Base	87	96	86	100	100	0.71	0.81	17
	Small	92	92	100	100	100	0.45	0.46	21
	Shark	91	95	0	100	89	0.55	0.66	12
	Large	98	96	100	100	86	0.86	0.89	22
Shark	All+	95	92	100	100	74	0.93	0.94	29
	Base+	76	78	0	100	80	-	-	-
	Small+	90	91	0	100	70	-	-	-
	Shark+	94	92	0	100	92	-	-	-
	Large+	83	92	100	100	86	-	-	-
	Base	67	84	100	100	90	0.66	0.75	25
	Small	95	86	0	100	0	0.69	0.74	18
	Shark	95	98	0	95	97	0.82	0.86	21
	Large	88	88	90	100	100	0.62	0.64	22
Small	All+	99	99	100	100	78	0.72	0.75	23
	Base+	81	92	100	100	92	-	-	-
	Small+	94	95	75	100	100	-	-	-
	Shark+	96	91	0	100	79	-	-	-
	Large+	88	94	0	100	82	-	-	-
	Base	89	88	100	0	100	0.72	0.75	23
	Small	98	97	100	100	0	0.80	0.85	27
	Shark	97	93	0	100	91	0.71	0.74	16
	Large	82	87	83	75	93	0.71	0.72	25

Table 1. Combined Distance and Direction Estimation Experiment Results. For direction estimation models, we compare % accuracy of estimates per axis. For distance estimation models, we compare correlation C and Cq between estimated and actual distances and number of tiers Tr.

Part	Ag	Model	SR	E	T	Q	A
Base	3	All+	100	1.96	106	8	1
		Base+	100	1.83	227	17	4
		Shark+	70	2.85	456	24	5
	1	All+	100	0.65	72	12	2
		Base+	100	1.81	88	16	2
		Shark+	60	2.44	75	35	1
Large	3	All+	100	1.60	212	16	3
		Base+	90	3.78	303	21	4
		Shark+	100	3.18	287	19	5
	1	All+	60	0.80	172	40	6
		Base+	0	3.31	64	50	1
		Shark+	30	3.03	88	41	2
Shark	3	All+	100	1.44	130	10	2
		Base+	60	11.25	501	34	3
		Shark+	100	1.18	167	12	2
	1	All+	80	1.08	125	27	4
		Base+	10	5.54	64	47	1
		Shark+	90	1.50	84	17	2
Small	3	All+	100	1.42	197	15	3
		Base+	90	2.81	232	18	3
		Shark+	100	1.14	184	14	3
	1	All+	100	1.46	72	15	2
		Base+	0	2.72	64	50	1
		Shark+	100	1.17	59	14	1

Table 2. Full Assembly Experiments Results. We compare success rate $SR\%$, average error Emm, completion time Ts, number queries Q, and number attempts A across approaches and models.

Part	Ag	Model	SR	E	T	Q	A
Gear	3	All+	100	2.82	214	15	3
	1	All+	100	2.72	56	9	1
Green	3	All+	67	2.64	359	25	4
	1	All+	0	4.12	123	50	6

Table 3. Assembly With FOOD Parts Results

Grasp	Ag	Model	SR	E	T	Q	A
Vert	3	All+	100	1.96	106	8	1
	1	All+	100	0.65	72	12	2
Horz	3	All+	80	3.68	372	24	5
	1	All+	100	1.76	76	11	2
Occl	3	All+	40	6.79	600	40	12
	1	All+	20	6.10	945	46	11

Table 4. Assembly With FOOD Grasps Results

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[2] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. 1

[3] Haritheja Etukuru, Norihito Naka, Zijin Hu, Seungjae Lee, Julian Mehu, Aaron Edsinger, Chris Paxton, Soumith Chintala, Lerrel Pinto, and Nur Muhammad Mahi Shafiullah. Robot utility models: General policies for zero-shot deployment in new environments, 2024. 1

[4] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023. 2

[5] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024. 2

[6] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 1

[7] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023. 1

[8] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 1

[9] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 1

[10] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, page 100017, 2023. 1

[11] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[12] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern machines. *arXiv preprint arXiv:2307.04721*, 2023. 1

[13] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 1

[14] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[15] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023. 2

[16] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025. 1

[17] Yunsheng Tian, Karl DD Willis, Bassel Al Omari, Jieliang Luo, Pingchuan Ma, Yichen Li, Farhad Javid, Edward Gu, Joshua Jacob, Shinjiro Sueda, et al. Asap: Automated sequence planning for complex robotic assembly with physi-

cal feasibility. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4380–4386. IEEE, 2024. 2

[18] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[19] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6): 186345, 2024. 2

[20] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1

[21] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023. 2