Knolling Bot: Teaching Robots the Human Notion of Tidiness

Yuhang Hu[1], **Judah Goldfeder**[1], **Zhizhuo Zhang**[1], **Xinyue Zhu**[1], **Ruibo Liu**[1], **Philippe M. Wyder**[1], **Jiong Lin**[1], **Hod Lipson**[1]

[1]**Columbia University**

https://youtu.be/QcDtNfAYQXk

Abstract

For robots to truly collaborate and assist humans, they must understand not only logic and instructions, but also the subtle emotions, aesthetics, and feelings that define our humanity. Human art and aesthetics are among the most elusive concepts—often difficult even for people to articulate—and without grasping these fundamentals, robots will be unable to help in many spheres of daily life. Consider the long-promised robotic butler: automating domestic chores demands more than motion planning; it requires an internal model of cleanliness and tidiness—a challenge largely unexplored by AI. To bridge this gap, we propose an approach that equips domestic robots to perform simple tidying tasks via knolling, the practice of arranging scattered items into neat, space-efficient layouts. Unlike the uniformity of industrial settings, household environments feature diverse objects and highly subjective notions of tidiness. Drawing inspiration from NLP, we treat knolling as a sequential prediction problem and employ a transformer-based model to forecast each object's placement. Our method learns a generalizable concept of tidiness, generates diverse solutions adaptable to varying object sets, and incorporates human preferences for personalized arrangements. This work represents a step forward in building robots that internalize human aesthetic sense and can genuinely co-create in our living spaces. Code is available at github.com/H-Y-H-Y-H/knolling_bot

1 Introduction

Designing a robot for household tasks has always presented unique challenges [46, 23, 49, 3]. Unlike industrial settings characterized by uniformity and limited object variety, household environments are filled with diverse objects. Recent progress in object rearrangement has shown that robots can organize objects in partially arranged scenes, cluttered tabletops, and constrained environments[34, 11, 30]. While their work has provided effective approaches to solving task-specific problems, it is crucial to develop methods that can learn a generalized representation of tidiness. Our motivation is to enable robots to understand and apply tidiness concepts that improve with more data, unlike traditional rule-based methods that become increasingly complex with more scenarios.

In common daily environments, including both local environments like a desk, and larger environments like a house, the objects encountered vary constantly. In such dynamic settings, providing specific goals for each object is impractical and limits the robot's generalizability. Instead, robots should be capable of organizing the environment without relying on specific target positions or constant human supervision. This requires the robot to understand and apply a broad concept of

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Creative AI Track.

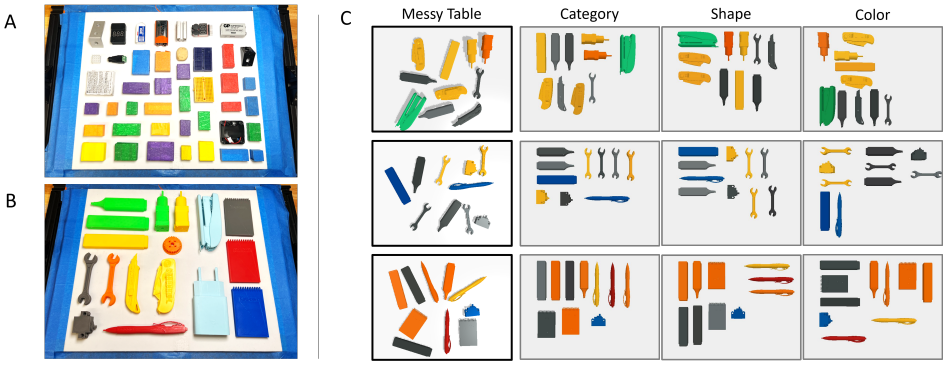

Figure 1: **Examples from the knolling task.** A) A batch of small items, including daily necessities. B) A batch of big items fabricated by a 3D printer. C) Diverse knolling preferences demonstrated in experiments. The model adapts its tidying behavior based on different preferences: object category (group by function), size (large objects first), and color (group by hue).

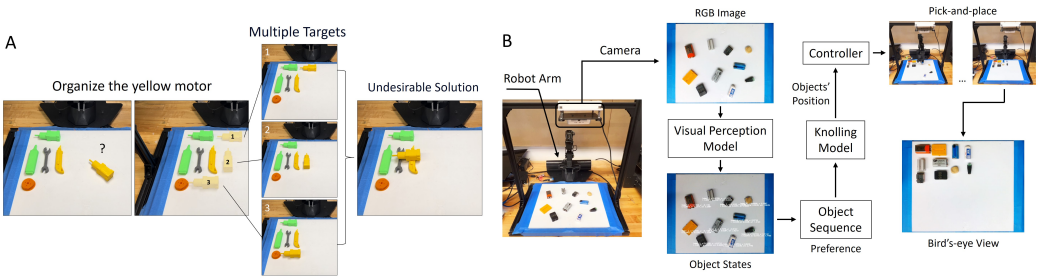

Figure 2: **A) Challenges in rearrangement tasks with multiple solutions.** From left to right: the initial state of the work area with an unplaced yellow motor. Three proposed placement options (1, 2, 3) for the yellow motor, considering factors such as object category, color similarity, and spatial efficiency. Placement by a regression-based model optimizing for minimal loss across three solutions, leading to undesirable results: placing the motor on a utility knife. **B) Knolling pipeline.** The left side of the figure displays a cluttered desktop with various objects such as batteries, erasers, electronic components, and other daily necessities in the lab. Our robot initiates the knolling process after detecting and identifying these objects through a camera. The right side of the figure depicts the outcome of this process, presenting a tidy, well-organized desktop. This transformation exemplifies the robot's ability to apply the knolling model, execute a tidying task, and create a pleasing and space-efficient arrangement.

organization that extends beyond the specifics of any single task. The necessary adaptability is similar to human-like cognition, where decisions are made not only based on spatial arrangement but also considering the aesthetic and functional aspects of tidiness.

"Knolling" is one such concept intrinsic to humans, referring to the intuitive ability to organize items in a manner that is both aesthetically pleasing and space-efficient (Fig. 1). Our solution lies in decoupling the cognitive model, which encapsulates the representation of knolling, from the other modules, such as the visual perception system and motor controllers[15, 31]. This division enhances the modularity and interpretability of the entire system compared to only training a single policy mapping from observation to action.

Given the subjectivity of tidiness and the multiple optimal solutions that could satisfy different individuals' preferences, knolling is an abstract problem without any definitive standards. Consider the scenario depicted in Fig.2A, where a robotic arm performs a task to integrate a yellow motor into a partially organized environment. The decision on where to place the motor can be depended on various factors, including object category, color, and the need for space efficiency. Training a model solely on regression to predict target positions based on object states might lead to local optimal outcomes. For instance, in Fig.2A a model optimized to minimize the loss across different placement preferences could converge to an average of the three potential target positions, resulting in the motor being placed in a position that is overlapped with the utility knife. This is just an example of a

one-step rearrangement of a partially organized scenario. A wrong-placed object may cause the next step of rearrangement to become worse because the errors have accumulated. Moreover, quantifying tidiness with a singular metric is challenging, as no simple equation can encapsulate the diverse aspects of what constitutes a “tidy" space.

In this work, we propose a self-supervised learning framework for modeling tidiness through demonstrations, akin to training a Large Language Model (LLM) with human conversational datasets[6, 47, 1, 33]. We draw parallels between knolling objects and language, where objects act as individual “words" that can be combined in various ways to form “sentences” with identical meanings. Besides, the transformer architecture can handle varying input and output sizes with autoregression, which fits the knolling task as the number of objects in the environment is arbitrary[42, 12]. Leveraging the Gaussian Mixture Model (GMM), we address the multi-label prediction challenges inherent in the knolling task, where multiple valid placements may exist for the same object depending on contextual preferences (e.g., spatial efficiency or category)[37].

Our research contributes by equipping robots with a conceptual understanding of tidiness, demonstrating an integrated pipeline for knolling in real-world tasks. As shown in Fig.2B , itcomprises three stages: a knolling model, a visual perception model, and a robot arm controller. The knolling model, based on a transformer architecture, predicts the target positions of objects. The visual perception model, based on our customized YOLO v8, detects objects from an RGB image [35]. Finally, the robot arm controller, leveraging the outputs from the previous stages, guides the robot arm to execute the knolling tasks.

The key contributions of this work are as follows:

1. We propose a novel self-supervised learning framework for modeling the representation of tidiness. This allows the model to learn generalizable tidiness patterns directly from demonstrations, improving its performance as more data becomes available.
2. We leverage transformer architectures and Gaussian Mixture Models to address the multi-label prediction challenges inherent in knolling tasks, and can process unseen objects and preferences, highlighting the potential for deploying such systems in dynamic household environments.
3. We demonstrate an integrated pipeline combining the learned knolling model with visual perception and robotic control modules. This allows for deploying knolling capabilities on a real robotic system to organize cluttered environments with varying object quantities and types.
4. We contribute to the research community by not only open-sourcing our dataset of tidy object arrangements but also establishing a benchmark with comprehensive evaluation metrics for the knolling task. This benchmark will enable comparative studies and further exploration of arrangement tasks with arbitrary object numbers and shapes.

2 RELATED WORK

The domain of robot learning and manipulation tasks has seen significant advancements in recent years. Various manipulation tasks, such as wiping and polishing, stacking, peg-in-hole, and pick-and-place tasks, have been explored[4, 51, 26, 24]. In the realm of stacking tasks, Lee et al. used offline reinforcement learning (RL) to improve upon existing policies for robotic stacking of objects with complex geometry, while Furrer et al. proposed an algorithm for suggesting stable poses for stacking, validated through a real-world autonomous stacking workflow[25, 10]. For pick-and-place tasks, Gualtieri et al. proposed a deep reinforcement learning approach, and Zeng et al. introduced the Transporter Network for vision-based manipulation tasks[39, 48].

In the domain of machine learning models, transformers have revolutionized many fields with their attention-based architecture[32, 21, 29, 19, 44, 28]. The success of transformer models in NLP has prompted exploration in their application to robotic tasks[22, 8, 36]. Jangir et al. proposed the use of transformers with a cross-view attention mechanism for effective fusion of visual information from two cameras for RL policies[16]. Zhu et al. introduced VIOLA, which used a transformer-based policy to improve the robustness of deep imitation learning algorithms[50]. Shridhar et al. proposed PerAct, which used the preceiver transformer to encode language goals and RGB-D voxel observations[40]. Jain et al. proposed the Transformer Task Planner, which could be pre-trained on multiple preferences, and Dasari et al. explored one-shot visual imitation learning using the transformer architecture[14].

In the object rearrangement field, most recent work has focused on rearranging objects based on explicit instructions[13, 7]. Some other works use a more general method to train a representation model or reward function to supervise the object rearrangement policy without explicit human supervision[45, 43]. With the rapid rise of LLM as powerful all-purpose models, many works have leveraged them to processs[27, 5, 2]. Instead of giving natural language instructions to the robot, some other works train a robot a common sense of how to place the objects through demonstrations[14, 18, 38]. Combining both approaches, Housekeep[17] provides a method that leverages LLM to train a common sense of tidiness for robots. While these works have contributed significantly to robot manipulation, we aim to provide a common sense of tidiness for robots through demonstrations. We focus on knolling tasks that rearrange messy objects on a desktop into a neat layout in the real world. Our approach allows for the handling of varying object types, sizes, and quantities, thus enabling the robot to generate an aesthetically pleasing and space-efficient arrangement of items, similar to human performance.

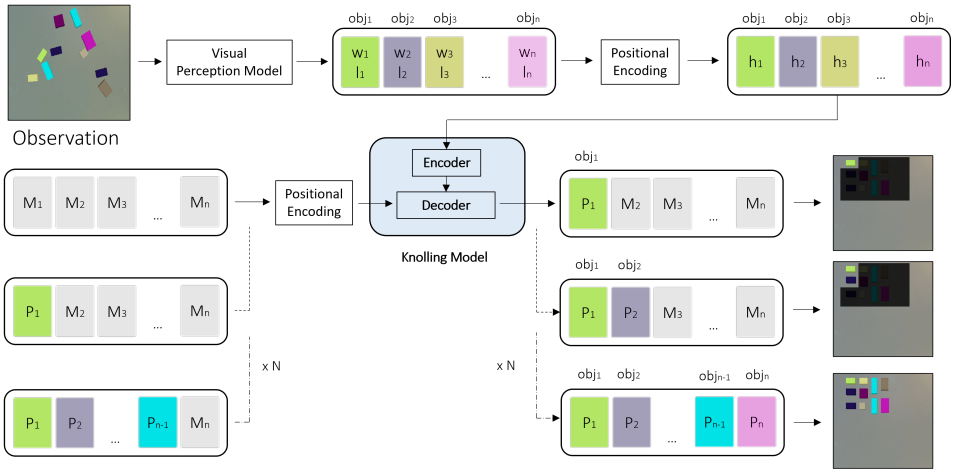

Figure 3: Knolling Model Learning Framework: The pipeline begins with the visual perception model, which processes an input image to identify objects and extract their state representations—including width (w), length (l), position and orientation, presented as a list. However, only w and l are used as input for the knolling. The knolling model takes high-dimensional object states (h) derived from positional encoding as input. During training, a masked learning approach is employed, where part of the object data (M) is masked, and the model learns to predict the next object's position (P). The model predicts N target positions after N iterative processes.

3 METHOD

3.1 Data Representation and Generation

We represent objects solely based on their dimensional parameters (width and length), as these are the fundamental properties governing spatial organization. semantic attributes like color and category are excluded from the model input due to the subjective variability and potential bias they introduce in the context of tidiness. Unlike quantifiable features (e.g., position, size), attributes like color and category are often interpreted differently based on context and individual perception. Focusing on spatial and geometric attributes is more objectively measurable. This approach helps the model generalize tidiness based on universally applicable metrics rather than subjective semantic interpretations that might vary across different settings. To generate a diverse dataset of tidy arrangements, we design an optimization strategy that iteratively adjusts object placements to minimize the occupied area on the table. This strategy can control the results by adjusting placements based on object attributes such as color, dimension, or category, yielding distinct organizational patterns and aesthetic preferences. By iterating through multiple configurations, this stochastic process produces 2.4 million demonstrations that span a broad spectrum of tidiness concepts and visual styles, providing a rich dataset for training. For objects with irregular shapes (Fig. 1B), we train a visual perception system to perform segmentation and compute the minimum bounding box, enabling our framework to accommodate unconventional geometries.

3.2 Training The Knolling Model

Our self-supervised learning framework for training knolling models follows a curriculum-based approach, progressing from simple to complex tasks as shown in Fig. 3. We designed a two-stage training process to first acquaint the model with the basics of object arrangement and then refine its proficiency in executing complete knolling tasks from scratch. By training a representation of tidiness through self-supervised learning, our model's performance improves as more data becomes available, unlike traditional rule-based methods that become increasingly complex with more scenarios. For further details on the two training phases, pre-training with self-supervised Learning, amd finetuning, please see Appendix L. Our knolling pipeline comprises three modules: a knolling model, a visual perception model, and the robot arm controller for executing the pick-and-place task. For further details on the pipeline and training objectives, see Appendix K.

4 Experiments

To evaluate the performance and practical applicability of our proposed knolling framework, we conducted experiments in both simulated and real-world environments. Additionally, we performed quantitative analyses and ablation studies to validate the effectiveness of our approach and elucidate the contributions of different components.

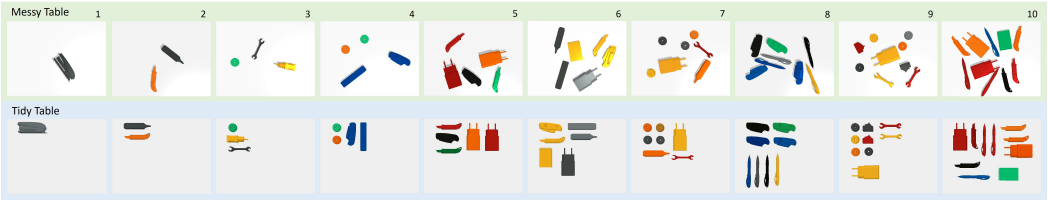

Figure 4: Examples of knolling messy tables with different numbers of objects. The figure shows ten examples of tables before and after the knolling process in the simulation.

4.1 Qualitative Evaluations in the Simulation

Handling Varying Object Quantities. In real-world scenarios, the number of objects on a surface can vary significantly. To assess our model's ability to handle this variability, we conducted experiments with diverse object counts ranging from 2 to 10 items. Our model, leveraging its autoregressive nature, successfully generated tidy arrangements across all object quantities, as shown in Fig. 4. This adaptability to varying input sizes is a crucial aspect of our approach, enabling it to generalize to different scenarios without explicit retraining.

Generating Diverse Solutions Based on Preferences. Tidiness preferences can differ significantly among individuals, leading to multiple valid solutions for the same set of objects. To capture this diversity, our model decouples preferences from the input data (which consists solely of object dimensions) and instead encodes preferences through the order of the input sequence. By altering the input order based on criteria such as object category, color, or size, our model generates distinct tidy arrangements tailored to specific preferences, as illustrated in Fig. 1 C. This flexibility to incorporate user preferences without modifying the model architecture is a key strength of our approach.

4.2 Real-World Knolling Experiments

To demonstrate the practical applicability of our framework, we deployed our trained knolling model on a 5-DoF robotic arm (WidowX 200) equipped with an Intel Realsense D435 camera for visual perception. We randomly placed 6-10 boxes of varying sizes and colors within the robot's workspace, simulating cluttered tabletop scenarios. The integrated pipeline (Fig. 2) first captures an overhead image of the cluttered scene, which is processed by our visual perception module to detect objects, perform segmentation, and extract their dimensions and poses. The knolling model then predicts the target positions for each object based on their dimensions. Finally, the robotic arm controller executes smooth pick-and-place operations, guided by the predicted target positions and current object poses, to realize the tidy arrangements. Fig. 5A showcases real-world knolling tasks performed on different

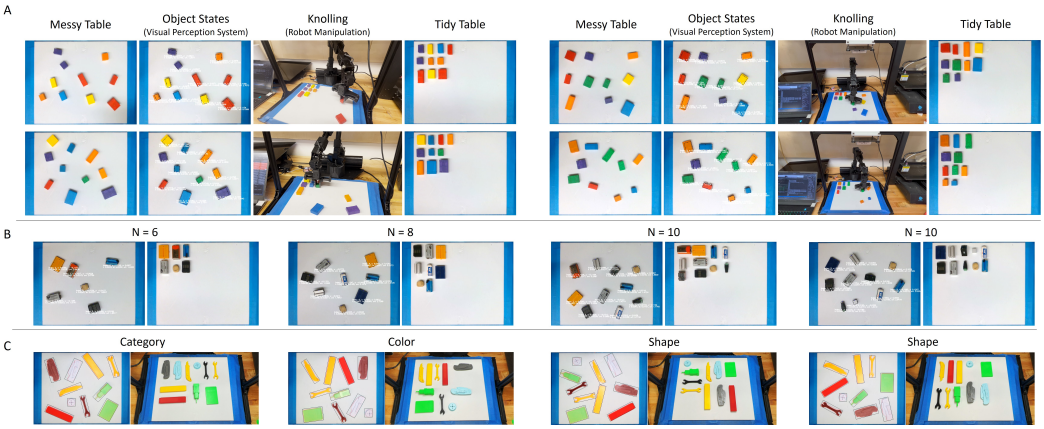

Figure 5: **A)** Box knolling in the real world. In each test, we show four columns. Column 1: The initial state of the objects on a table, as captured by the overhead camera. Column 2: The same scenario as Column 1, with added key points and contour outlines indicating the detected objects. Column 3: Action snapshot of the robot executing the knolling task. Column 4: The final state of the workspace post-knolling, presenting an tidy table. **B)** Real-world Knolling Process with Different Object Numbers. This figure exhibits the practical application of our knolling model in four diverse scenarios. Each column corresponds to a different setup with a distinct number of objects (6, 8, 10, and 10). We show the initial messy state captured by the overhead camera and the organized layout after the knolling task is completed by the robot arm. These comparative visuals underline our robot's proficiency in performing real-world knolling tasks across varied object quantities. **C)** For the same objects on the table, our robot can perform knolling tasks with different solutions based on preferences based on category, color, or shape.

box configurations, demonstrating our system's capability to organize diverse object sets. Fig. 5B further illustrates the system's adaptability to varying object quantities, successfully knolling scenes with 6, 8, and 10 objects. Additionally, Fig. 5C highlights the generation of distinct tidy arrangements for the same set of objects, reflecting different user preferences based on category, color, or shape. We encourage readers to view our supplementary video for a comprehensive visualization of these real-world knolling experiments. For further quantitative evaluation and an ablation study, please see Appendix I. For an analysis of how performance scales with data, see Appendix J.

5 CONCLUSIONS

In this work, we introduced a self-supervised learning framework for enabling robots to understand and replicate the representation of tidiness, or "knolling," from demonstrations of well-organized object arrangements. Our approach leverages transformer architectures and Mixture Density Networks to model the inherent multi-modality of knolling tasks, where identical object configurations can lead to diverse but tidy arrangements. Our model learns a generalizable representation of tidiness that can adapt to varying object quantities and incorporate user preferences by training on a large dataset of knolling demonstrations without explicit human supervision or target positions. We demonstrated the effectiveness of our framework in generating aesthetically pleasing and space-efficient object arrangements. Our quantitative evaluations showed that our transformer-based model consistently outperformed baseline architectures, highlighting the advantages of self-attention and autoregressive mechanisms for handling variable input sizes and multi-label predictions. Moreover, we presented an integrated pipeline that combines our learned knolling model with visual perception and robotic control modules, enabling the deployment of knolling capabilities on a real robotic system. Our real-world experiments showcased the system's ability to organize cluttered environments with diverse object types and quantities, adapting to different user preferences based on criteria such as object category, color, or shape. This work represents an important step forward in imbuing robots with human aesthetics, a critical requirement for effective human-robot interaction, and we hope our work inspires further research in this area.

Acknowledgments: This work was supported in part by the US National Science Foundation AI Institute for Dynamical Systems (dynamicsAI.org) (grant no. 2112085).

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.

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[3] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020.

[4] Lars Berscheid, Pascal Meißner, and Torsten Kröger. Robot learning of shifting objects for grasping in cluttered environments. In *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 612–618. IEEE, 2019.

[5] Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. A persistent spatial semantic representation for high-level natural language instruction execution. In *Conference on Robot Learning*, pages 706–717. PMLR, 2022.

[6] Boyuan Chen, Yuhang Hu, Lianfeng Li, Sara Cummings, and Hod Lipson. Smile like you mean it: Driving animatronic robotic face with learned models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2739–2746. IEEE, 2021.

[7] Sang Hun Cheong, Brian Y Cho, Jinhwi Lee, ChangHwan Kim, and Changjoo Nam. Where to relocate?: Object rearrangement inside cluttered and confined environments for robotic manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7791–7797. IEEE, 2020.

[8] Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *Conference on Robot Learning*, pages 2071–2084. PMLR, 2021.

[9] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Fadri Furrer, Martin Wermelinger, Hironori Yoshida, Fabio Gramazio, Matthias Kohler, Roland Siegwart, and Marco Hutter. Autonomous robotic stone stacking with online next best object target pose planning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2350–2356, 2017. doi: 10.1109/ICRA.2017.7989272.

[11] Kai Gao, Justin Yu, Tanay Sandeep Punjabi, and Jingjin Yu. Effectively rearranging heterogeneous objects on cluttered tabletops. *arXiv preprint arXiv:2306.14240*, 2023.

[12] Anthony Gillioz, Jacky Casas, Elena Mugellini, and Omar Abou Khaled. Overview of the transformer-based models for nlp tasks. In *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, pages 179–183. IEEE, 2020.

[13] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Semantically grounded object matching for robust robotic scene rearrangement. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11138–11144. IEEE, 2022.

[14] Vidhi Jain, Yixin Lin, Eric Undersander, Yonatan Bisk, and Akshara Rai. Transformers are adaptable task planners. In *Conference on Robot Learning*, pages 1011–1037. PMLR, 2023.

[15] Eric Jang, Sudheendra Vijayanarasimhan, Peter Pastor, Julian Ibarz, and Sergey Levine. End-to-end learning of semantic grasping. *arXiv preprint arXiv:1707.01932*, 2017.

[16] Rishabh Jangir, Nicklas Hansen, Sambaran Ghosal, Mohit Jain, and Xiaolong Wang. Look closer: Bridging egocentric and third-person views with transformers for robotic manipulation. *IEEE Robotics and Automation Letters*, 7(2):3046–3053, 2022. doi: 10.1109/LRA.2022.3144512.

[17] Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. Housekeep: Tidying virtual households using commonsense reasoning. In *European Conference on Computer Vision*, pages 355–373. Springer, 2022.

[18] Ivan Kapelyukh and Edward Johns. My house, my rules: Learning tidying preferences with graph neural networks. In *Conference on Robot Learning*, pages 740–749. PMLR, 2022.

[19] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.

[20] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[21] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54 (10s):1–41, 2022.

[22] Heecheol Kim, Yoshiyuki Ohmura, and Yasuo Kuniyoshi. Transformer-based deep imitation learning for dual-arm robot manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8965–8972. IEEE, 2021.

[23] Jaeseok Kim, Anand Kumar Mishra, Raffaele Limosani, Marco Scafuro, Nino Cauli, Jose Santos-Victor, Barbara Mazzolai, and Filippo Cavallo. Control strategies for cleaning robots in domestic applications: A comprehensive review. *International Journal of Advanced Robotic Systems*, 16(4):1729881419857432, 2019.

[24] Aljaž Kramberger, Erfan Shahriari, Andrej Gams, Bojan Nemec, Aleš Ude, and Sami Haddadin. Passivity based iterative learning of admittance-coupled dynamic movement primitives for interaction with changing environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6023–6028, 2018. doi: 10.1109/IROS.2018.8593647.

[25] Alex X Lee, Coline Manon Devin, Yuxiang Zhou, Thomas Lampe, Konstantinos Bousmalis, Jost Tobias Springenberg, Arunkumar Byravan, Abbas Abdolmaleki, Nimrod Gileadi, David Khosid, et al. Beyond pick-and-place: Tackling robotic stacking of diverse shapes. In *5th Annual Conference on Robot Learning*, 2021.

[26] Daniel Leidner, Georg Bartels, Wissam Bejjani, Alin Albu-Schäffer, and Michael Beetz. Cognition-enabled robotic wiping: Representation, planning, execution, and interpretation. *Robotics and Autonomous Systems*, 114:199–216, 2019.

[27] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[28] Weiyu Liu, Tucker Hermans, Sonia Chernova, and Chris Paxton. Structdiffusion: Object-centric diffusion for semantic rearrangement of novel objects. *arXiv preprint arXiv:2211.04604*, 2022.

[29] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6322–6329. IEEE, 2022.

[30] Xibai Lou, Houjian Yu, Ross Worobel, Yang Yang, and Changhyun Choi. Adversarial object rearrangement in constrained environments with heterogeneous graph neural networks. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1008–1015. IEEE, 2023.

[31] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2901–2910, 2019.

[32] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *arxiv*, 2018.

[33] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[34] Kartik Ramachandruni, Max Zuo, and Sonia Chernova. Consor: A context-aware semantic object rearrangement framework for partially arranged scenes. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 82–89. IEEE, 2023.

[35] Dillon Reis, Jordan Kupec, Jacqueline Hong, and Ahmad Daoudi. Real-time flying object detection with yolov8. *arXiv preprint arXiv:2305.09972*, 2023.

[36] Hanwen Ren and Ahmed H Qureshi. Neural rearrangement planning for object retrieval from confined spaces perceivable by robot's in-hand rgb-d sensor. *arXiv preprint arXiv:2402.06976*, 2024.

[37] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[38] Gabriel Sarch, Zhaoyuan Fang, Adam W Harley, Paul Schydlo, Michael J Tarr, Saurabh Gupta, and Katerina Fragkiadaki. Tidee: Tidying up novel rooms using visuo-semantic commonsense priors. In *European Conference on Computer Vision*, pages 480–496. Springer, 2022.

[39] Gerrit Schoettler, Ashvin Nair, Juan Aparicio Ojea, Sergey Levine, and Eugen Solowjow. Meta-reinforcement learning for robotic industrial insertion tasks. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9728–9735. IEEE, 2020.

[40] Mohit Shridhar, Lucas Manuegrantlli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[41] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 969–977, 2018.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[43] Chen Wang, Danfei Xu, and Li Fei-Fei. Generalizable task planning through representation pretraining. *IEEE Robotics and Automation Letters*, 7(3):8299–8306, 2022.

[44] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19037–19047, 2023.

[45] Mingdong Wu, Fangwei Zhong, Yulong Xia, and Hao Dong. Targf: Learning target gradient field to rearrange objects without explicit goal specification. *Advances in Neural Information Processing Systems*, 35:31986–31999, 2022.

[46] Georgios A Zachiotis, George Andrikopoulos, Randy Gornez, Keisuke Nakamura, and George Nikolakopoulos. A survey on the application trends of home service robotics. In *2018 IEEE international conference on Robotics and Biomimetics (ROBIO)*, pages 1999–2006. IEEE, 2018.

[47] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36 (4):1307–1319, 2020.

[48] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[49] Junpei Zhong, Chaofan Ling, Angelo Cangelosi, Ahmad Lotfi, and Xiaofeng Liu. On the gap between domestic robotic applications and computational intelligence. *Electronics*, 10(7):793, 2021.

[50] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Object-centric imitation learning for vision-based robot manipulation. In *Conference on Robot Learning*, pages 1199–1210. PMLR, 2023.

[51] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

Technical Appendices

A Training Objectives

We employ a combination of loss functions during training:

1. Log-likelihood Loss (L_{ll})

The Log-likelihood loss is a standard approach when working with GMM, which allows the model to express uncertainty and multi-targets in the predictions by assigning probabilities to a range of possible positions for each object. This is particularly useful for handling the diversity and complexity of knolling tasks, where the spatial arrangement of objects can vary significantly. The log-likelihood loss for each object i, considering a Gaussian Mixture Model (GMM) with J components, is given by:

$$L_{\text{ll}} = -\sum_{i=1}^{N} \log \left(\sum_{j=1}^{J} p_{ij} \cdot \mathcal{N}(S_i | \mu_{ij}, \sigma_{ij}^2) \right) \tag{1}$$

where N is the total number of objects, p_{ij} is the weight of the j-th component for the i-th object, $\mathcal{N}(S_i|\mu_{ij}, \sigma_{ij}^2)$ is the probability density of the i-th object's target position S_i under the j-th Gaussian component with mean μ_{ij} and variance σ_{ij}^2.

2. MSE Loss (L_{MSE})

The Mean Squared Error (MSE) Loss computes the squared difference between the sampled predicted position ($\hat{S}$) and the target position (S):

$$L_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} (\hat{S}_i - S_i)^2 \tag{2}$$

where $\hat{S}_i$ is the sampled position from the predicted distribution for the i-th object, and S_i is the actual target position.

3. MSE Min Loss ($L_{\text{MSE Min}}$)

This encourages the model not only to predict accurately but also to account for the multiple solutions in the dataset. Considering J components in the Gaussian mixtures for each prediction step, the MSE Min Loss is calculated as follows:

$$L_{\text{MSE Min}} = \frac{1}{N} \sum_{i=1}^{N} \min_{j=1}^{J} \left((\hat{S}_{ij} - S_i)^2 \right) \tag{3}$$

In a multi-target task, relying solely on MSE could lead the model to predict average positions in the presence of ambiguity, which is not desirable. The MSE Min Loss addresses this by allowing the model to choose the prediction (from among multiple Gaussian components) that is closest to the actual target position.

4. Overlapping Loss (L_{overlap})

The Overlapping Loss penalizes predictions where objects overlap or cross boundaries. It acts as a form of regularization but with a very specific purpose: to ensure that the predicted positions do not result in physically impossible arrangements, such as overlapping objects or objects placed outside the designated workspace. This loss component is essential for maintaining the feasibility of the predicted knolling arrangements, ensuring that the model's predictions adhere to spatial constraints inherent in the real-world application of knolling. The loss function can be expressed as:

$$L_{\text{overlap}} = \sum_{i \neq k} \text{Penalty}(\hat{S}_i, \hat{S}_k, w_n, l_n, W, L) \tag{4}$$

The penalty function calculates the overlap and out-of-boundaries areas based on their predicted positions ($\hat{S}_i, \hat{S}_j$), the given dimensions of objects (w_n, l_n) and workspace (W, L).

5. Entropy Loss (L_{entropy})

This loss function encourages diversity in the predictions by measuring the entropy across the probabilities of the J Gaussian components for each prediction. The entropy loss is defined as:

$$L_{\text{entropy}} = -\sum_{i=1}^{N}\sum_{j=1}^{J} p_{ij}\log(p_{ij}) \tag{5}$$

6. Weights of Loss Components

Each loss component was assigned a specific weight to balance their contributions during training, with the aim of optimizing knolling performance. The weights were determined empirically, through iterative experimentation and performance evaluation, to ensure the model adequately minimized both positional errors and object overlap.

B Iterative Optimization for Knolling Dataset Generation

While the rule-based arrangement method presented here provides a systematic approach to generating a knolling dataset, it also has inherent limitations. Rule-based methods excel at articulating specific, human-defined rules of neatness—such as alignment, spacing, and categorization by color or type—that are relatively straightforward to parameterize. However, much of neatness is inherently subjective and nuanced. Although humans can often recognize whether an arrangement is "neat" or "not neat," they may struggle to articulate the underlying principles that define these notions. This subjectivity poses a challenge for rule-based systems, which can only operate within explicitly defined rules and fail to capture the subtleties of more subjective or abstract concepts of neatness.

Defining an exhaustive set of rules to account for all possible tidiness concepts is infeasible, as the number of potential layouts grows exponentially with object variety, user preferences, and spatial constraints. As a result, rule-based or optimization methods alone cannot cover all valid solutions. Instead, we take the opposite approach, learning a generalizable representation of tidiness from demonstrations through self-supervised learning.

While rule-based optimization is valuable for data generation, it is often computationally expensive and inefficient for real-time object arrangement. In contrast, the learned knolling model operates in a single forward pass, making predictions significantly faster than iterative optimization. This efficiency enhances scalability, enabling practical deployment in real-world robotic applications.

The rule-based optimization bootstraps the learning process, providing structured training examples to establish an initial understanding of tidiness. However, it is not a substitute for data-driven method. The model leverages this structured data as a foundation but learns beyond it, capturing nuanced spatial relationships and placement strategies that a purely optimization-based approach could never explicitly define.

B.1 Objects Used in the Knolling Task

we provide the following details regarding our dataset. **Rectangular Objects:** A subset of the dataset consists of randomly generated rectangular objects, with their length and width sampled from a predefined range. These synthetic objects serve as fundamental elements in training the model to generalize tidiness principles. Some of these generated objects were physically printed and used in real-world experiments to validate the learned representations. **Laboratory Items:** In addition to the rectangular objects, we incorporated commonly used laboratory items such as batteries, erasers, and electronic components. These objects were selected to introduce variation in shape and appearance. To facilitate color preference learning, we 3D-printed different versions of these objects in multiple colors. This ensures that the model can account for color-based grouping in tidiness preferences.

B.2 Arrangement Strategy and Policy Setup

To generate a knolling dataset, a meticulous arrangement framework is developed, which integrates parameterized object properties, multiple preferences, and an iterative layout optimization process.

This structured dataset generation ensures that the knolling model captures diverse object configurations, achieving precise arrangements. The framework enables training on a fixed number of objects (e.g., 10 objects), while flexibility in handling arbitrary numbers during deployment is achieved through masking techniques.

The knolling framework begins by setting arrangement policies that define the characteristics and layout behavior of each object. These policies include:

Object Dimensions: Length, width, and height ranges for each object are specified. For example, the length range is set to [0.036, 0.06] m, the width range to [0.016, 0.036] m, and the height range to [0.01, 0.02] m. This range allows for a controlled variation in object sizes within a standard workspace.

Classification Criteria: Objects are grouped based on attributes like type and color. Each classification criterion adds to the combinatorial complexity, as multiple preferences, creating distinct layouts can group objects.

Preferences and Policies: Policies for even distribution, forced alignment, and iteration constraints are set to control spatial relationships. These preferences guide the arrangement structure, affecting how objects are spaced, oriented, and aligned in the final layout.

This policy setup defines a complex framework where each additional preference increases layout complexity. Expanding beyond ten objects or adding more preferences exponentially intensifies the dataset generation's computational requirements.

B.3 Arrangement Iteration

Object data, such as length, width, height, class, and color, are generated randomly, guided by the predefined limits set in the arrangement policies. With initial object attributes and groupings established, an iterative process is applied to optimize object layout. Iterations focus on minimizing the space occupied while meeting arrangement constraints, such as spacing, alignment, and rotational preferences. For each iteration, factors of the object count are calculated to determine potential grid configurations. For example, an object count of 10 allows configurations like a 2x5 grid, which are systematically tested. Objects are arranged iteratively in each grid configuration. During each iteration, the layout is scored based on spatial minimization and adherence to the arrangement preferences. Across iterations, the layout with the minimal space occupation and best policy adherence is selected as the final arrangement. Iteration count determines the thoroughness of this search, with higher values leading to more refined arrangements.

To improve computational efficiency, a threshold-based early-stopping mechanism is integrated. A threshold score for spatial occupation is predefined within the policy. If a layout configuration meets or surpasses this threshold, the iteration process halts early, reducing the total number of iterations. This early stopping criterion is crucial in managing computational resources, especially as the number of objects and arrangement preferences increases.

C Comparison of Knolling Bot and Related Methods

In this section, we compare our Knolling Bot approach with other notable methods in robotic object rearrangement, specifically StructFormer and LLM-GROP. A visual comparison is provided in Fig. 6. Our method learns generalizable tidiness patterns through a self-supervised learning framework, enabling it to rearrange an arbitrary number of objects into neat and visually organized layouts. This approach allows the robot to autonomously organize objects based on attributes such as color, size, or category, resulting in aesthetically pleasing and space-efficient arrangements. StructFormer employs a transformer-based neural network that interprets structured language commands and partial-view point clouds to rearrange objects into specified configurations. While effective in executing instructions like forming circles or lines, its reliance on explicit commands may limit its adaptability to more complex or unstructured tidiness preferences. Additionally, StructFormer's focus on predefined structures may not fully capture the nuances of human-like organization, and its scalability to larger or more diverse object sets has not been extensively demonstrated. LLM-GROP leverages large language models (LLMs) combined with task and motion planning to achieve semantically valid object rearrangements in tabletop scenarios. It utilizes LLMs to extract commonsense knowledge

about object configurations and integrates this with a task-motion planner to adapt to varying scene geometries. However, as demonstrated in its experiments, LLM-GROP is designed for tasks involving a limited number of objects, typically three to five, and focuses on specific tasks such as setting a dinner table. In contrast, Knolling Bot's self-supervised learning framework enables it to handle a larger and more variable number of objects without explicit instructions, making it more versatile in organizing diverse and unstructured environments. Notably, Knolling Bot achieves a level of neatness and organization that closely aligns with human aesthetic preferences, arranging objects in a manner that is both visually pleasing and functionally efficient. This human-like understanding of tidiness distinguishes Knolling Bot from other methods

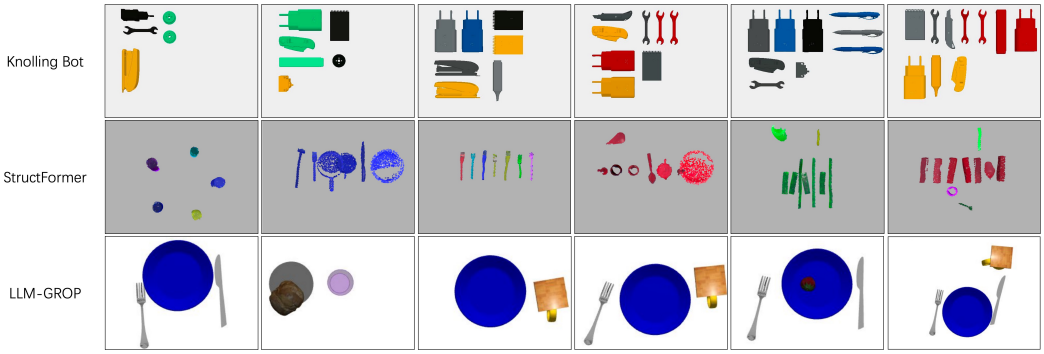

Figure 6: Comparison between our method and other methods. Our model can rearrange an arbitrary number of objects based on color, size, or categories and provide neat layouts.

D Gaussian Mixture Model for Multi-target Learning

To address the multi-target nature of knolling tasks, where multiple valid arrangements are possible for a given set of objects as shown in Fig.2A, we employ Mixture Density Networks (MDNs) to model the potential distribution of object placements. MDNs combine a neural network with a Gaussian Mixture Model (GMM), allowing the network to predict not just a single target position, but a weighted mixture of Gaussian distributions representing the diverse set of plausible positions for each object.

Formally, the MDN component of our model predicts the parameters $(\pi_j, \mu_j, \sigma_j)_{j=1}^{J}$ for a GMM with J components, where π_j are the mixing coefficients, μ_j are the means, and σ_j are the standard deviations of the J Gaussian distributions. For each object i, the predicted probability density function (PDF) over its possible placement positions S_i is given by:

$$p(S_i) = \sum_{j=1}^{J} \pi_{ij} \mathcal{N}(S_i | \mu_{ij}, \sigma_{ij}^2) \tag{6}$$

This formulation allows the model to capture the multiple knolling solutions, where identical object configurations can lead to different but equally valid tidy arrangements. During inference, we can sample from the predicted GMM to obtain diverse knolling predictions, or select the component with the highest mixing coefficient for a single deterministic prediction. Incorporating MDNs into our transformer architecture prevents the model from converging to averaging solutions and captures the inherent uncertainty and variability in tidiness preferences.

E Position Encodering

Our model utilizes two position encoding methods to process the input data effectively. The first method, aligned with the original transformer paper's approach, assigns unique tokens to each position in the list. This gives the data a sense of order, enabling the model to predict the target position of the initial objects and apply autoregression for the remaining objects. The second position encoding method maps the input data into a higher-dimensional space via sinusoidal functions. This method allows the model to handle higher frequency representations that potentially improve performance. The input, originally a 2-dimensional vector representing object length and width, is transformed

into a 21-dimensional feature vector through this encoding method. By varying the wavelength at five different frequencies, the position encoding captures fine-grained patterns in the data, leading to more accurate predictions.

F Transformer Performance across Dataset Sizes

This experiment aimed to evaluate the performance of our knolling model across different scales of data. Four dataset sizes were selected for this study: 125k, 250k, 500k, and 1M. Each variant of the knolling model was trained independently until convergence. Following training, all models were evaluated using a consistent test dataset to maintain uniformity in the assessment.

The test errors obtained for each dataset size are shown in Table 5. The results highlight a consistent decline in error as the dataset size augments. This trend validates the widely accepted belief that transformers excel with the increase in data volume.

G Visual Perception Model Evaluation

G.1 YOLO Segmentation Model

Our visual perception module employs a customized YOLO v8 model, fine-tuned for object detection, segmentation, and attribute prediction. We trained and evaluated two versions of the model: one for the simulated environment (YOLO-sim), and another for real-world deployment, (YOLO-real).

G.1.1 Training Settings

Simulation Model (YOLO-sim):

- Number of categories: 9
- Number of objects per image: 4-11
- Dataset sizes: 3200 train images, 800 test images
- Pre-trained model: yolov8n-seg

Real-World Model (YOLO-real):

- Number of categories: 9
- Number of objects per image: 4-11
- Dataset sizes: 80 train images, 20 test images
- Pre-trained model: YOLO-sim

G.1.2 Evaluation Metrics and Results

We evaluated the models' performance using the following metrics:

- Average Accuracy: Measures the accuracy of category predictions.
- Confusion Matrix: Provides a detailed breakdown of the model's category prediction performance.
- Average Hamming Distance: Assesses the quality of the predicted segmentation masks compared to the ground truth.

The evaluation results for both models are summarized in Tab.1.

Table 1: YOLO Segmentation Model Evaluation Results

Metric	YOLO-sim	YOLO-real
Average Hamming Distance	0.06743	0.05831
Average Accuracy	0.999	0.9857

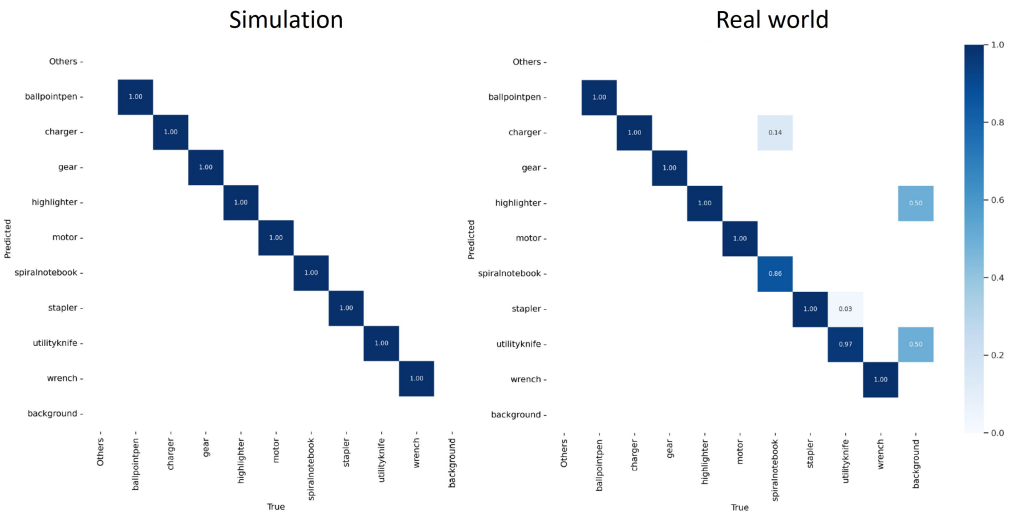

Figure 7: Confusion matrix for the YOLO segmentation model. The matrix shows the model's category prediction performance, with each cell representing the number of instances predicted as a particular category (column) compared to the true category (row). The diagonal cells indicate correct predictions, while off-diagonal cells represent misclassifications.

The results demonstrate that both models achieve high accuracy in category prediction and generate precise segmentation masks, with the real-world model slightly outperforming the simulation model in terms of Average Hamming Distance. The confusion matrices for both models are provided in Fig.7.

These evaluation results validate the effectiveness of our visual perception module in accurately detecting, segmenting, and categorizing objects in both simulated and real-world environments, enabling robust input for the subsequent knolling pipeline.

H Comparison to Baselines

To further validate the effectiveness of our transformer-based knolling model, we conducted additional experiments, exploring the model's performance relative to widely used architectures, including MLP, LSTM, and CNN2d (Table2) . These comparisons go beyond the strict parameter-matching paradigm outlined in the main paper, allowing architectural latitude in the baseline models to showcase their maximum potential performance under flexible conditions.

The motivations for this experiment are twofold. First, it addresses the practical concern that simpler models, such as MLPs or LSTMs, may perform comparably with smaller datasets. By demonstrating the superior performance of the transformer-based model even when baselines are given additional parameters, we underscore its robustness and suitability for complex layout tasks like knolling. Secondly, this comparison contributes to understanding the limitations of conventional architectures in capturing spatial dependencies critical to knolling applications.

Our transformer-based model significantly outperforms these baselines, where we report the error and parameter count for each model and configuration. This comparison illustrates that even with increased parameters, the baseline architectures exhibit higher error rates, reinforcing the advantages of a transformer-based approach for Knolling task.

I Quantitative Evaluation and Ablation Study

To quantitatively assess the performance of our knolling model, we employed the L1 distance metric, which measures the absolute difference between the predicted and ground truth object positions. For deterministic results during evaluation, we selected the predicted Gaussian distribution based on minimum loss instead of the mixture weights. Our evaluation was conducted on a diverse test

Table 2: Comparison to more baselines

MLP	**Error**	**Parameters**	**LSTM**	**Error**	**Parameters**	**CNN2d**	**Error**	**Parameters**
1	3.73E-01	62079	1	3.15E-02	183732	1	3.08E-02	92930
2	3.46E-01	107205	2	3.06E-02	101740	2	3.01E-01	92930
3	3.20E-01	10885	3	3.14E-02	73152	3	3.99E-01	40081
4	3.25E-01	125540	4	2.70E-02	191480	4	3.95E-01	42520
5	1.49E-01	101979	5	2.91E-02	103236	5	4.00E-01	36941
6	2.98E-01	136815	6	2.57E-02	60622	6	3.62E-01	30344
7	1.40E-01	83100	7	3.15E-02	62372	7	3.42E-01	79924
8	2.32E-01	8425	8	2.91E-02	81524	8	4.67E-01	27620
9	2.07E-01	70335	9	3.09E-02	90990	9	4.01E-01	34670
10	3.41E-01	170025	10	2.32E-02	183732	10	3.93E-01	33196
MEAN	2.73E-01		MEAN	2.90E-02		MEAN	3.49E-01	
STD	8.02E-02		STD	2.69E-03		STD	1.14E-01	
MIN	1.40E-01		MIN	2.32E-02		MIN	3.08E-02	

dataset comprising 20,000 samples for each object count ranging from 2 to 10 items, ensuring a comprehensive assessment across various knolling scenarios.

Table 3: Comparison between Our Method (OM) and Baselines (LSTM, MLP)

	N objs.	2	4	6	8	10
OM	MEAN	3.38E-04	2.40E-04	1.72E-04	2.08E-04	3.06E-04
	STD	2.03E-04	2.71E-04	2.94E-04	4.13E-04	5.51E-04
	MIN	8.52E-05	2.60E-06	3.40E-07	3.88E-06	1.36E-05
	MAX	1.74E-03	2.16E-03	3.83E-03	4.95E-03	7.77E-03
LSTM	MEAN	1.72E-02	2.09E-02	2.35E-02	2.62E-02	2.98E-02
	STD	3.78E-03	3.53E-03	4.15E-03	3.88E-03	6.36E-03
	MIN	8.37E-03	1.20E-02	1.43E-02	1.63E-02	1.86E-02
	MAX	2.65E-02	3.10E-02	3.47E-02	4.37E-02	4.99E-02
MLP	MEAN	2.17E-01	1.45E-02	1.90E-01	2.55E-01	3.26E-01
	STD	1.91E-03	4.68E-03	2.42E-03	7.97E-03	8.95E-03
	MIN	2.14E-01	6.98E-03	1.81E-01	2.38E-01	3.07E-01
	MAX	2.23E-01	3.07E-02	2.00E-01	2.83E-01	3.57E-01

Two baseline models were established to enable a performance comparison to Our Method (OM). The first baseline model utilized a Multilayer Perceptron (MLP) architecture, and the second baseline was designed based on the Long Short-Term Memory (LSTM) model. We use LSTM as a baseline due to its effectiveness in sequence prediction tasks. Arranging objects can be framed as predicting a sequence of positions, where each object's placement is influenced by prior placements. LSTMs have been extensively used for various sequence prediction tasks and have shown good performance. Thus, LSTMs serve as a more challenging baseline for benchmarking the performance of our transformer-based model. Moreover, comparing Transformer model with LSTMs helps to highlight the benefits of using self-attention and auto-regression in the transformer for handling variable input and output sizes and multi-label problems. Just as we did with Transformer model, we measured the L1 distance between the actual and predicted positions for each baseline model, with the results detailed in Table 3. As a commitment to a fair evaluation, we have ensured that each model utilizes a similar amount of parameters: Transformer model incorporates 87,458 parameters, the LSTM baseline uses 86,858 parameters, and the MLP baseline operates with 87,788 parameters. Our knolling model consistently outperforms the MLP and LSTM baselines in terms of the L1 error. This superiority of Transformer model is evident in all parameters: mean L1 error, standard deviation, and minimum and maximum error. These results validate the effectiveness of our transformer-based model, which employs self-attention and auto-regression in handling varying input sizes and multi-label problems in knolling tasks.

Furthermore, we conducted an ablation study to evaluate the contributions of the individual loss components employed during training (Table 4). The ablation results demonstrated the importance

Table 4: Ablation study

Model	Test Performace	Log-likelihood Loss	MSE Loss	MSE Min Loss	Overlapping Loss	Entropy Loss
OM	**5.78E-04 ± 6.64E-04**	-6.37E+00 ± 1.25E+00	4.91E-04 ± 6.25E-04	8.76E-03 ± 1.60E-02	9.23E-01 ± 9.55E-03	2.55E-02 ± 1.43E-02
OM w/o LL. Loss	7.24E-04 ± 7.53E-04	1.13E+02 ± 1.61E+02	6.34E-04 ± 7.05E-04	9.07E-03 ± 1.43E-02	9.01E-01 ± 2.25E-08	3.56E-02 ± 8.82E-03
OM w/o Pos. Loss	1.23E-02 ± 2.74E-03	-1.50E+00 ± 4.53E-01	1.05E-02 ± 2.47E-03	1.80E-01 ± 1.25E-01	1.06E+00 ± 4.09E-03	1.71E-01 ± 1.20E-02
OM w/o Pos. Min Loss	3.93E-02 ± 7.48E-03	3.51E-01 ± 5.64E-01	3.90E-02 ± 7.48E-03	2.32E-02 ± 3.37E-02	1.09E+00 ± 4.73E-03	1.14E-01 ± 1.13E-02
OM w/o Over. Loss	5.08E-02 ± 1.22E-02	-6.92E+00 ± 1.55E+00	3.40E-04 ± 4.07E-04	5.05E+00 ± 1.22E+00	9.66E-01 ± 2.44E-02	1.95E-02 ± 1.24E-02
OM w/o Ent. Loss	1.03E-03 ± 1.00E-03	-4.71E+00 ± 1.04E+00	9.07E-04 ± 9.43E-04	1.20E-02 ± 1.97E-02	1.08E+00 ± 5.70E-02	2.54E-02 ± 1.27E-02

of each loss term, with their removal leading to performance degradation, justifying the need for a comprehensive loss function to capture the multifaceted nature of knolling tasks effectively.

J Transformer Performance across Dataset Sizes

The conducted experiment provides valuable insights into how the performance of our knolling model scales with the size of the training dataset. As shown in Table 5, there is a clear and consistent decline in test error as the dataset size increases from 125,000 to 1 million samples. This reduction in error underscores the model's enhanced ability to generalize from larger amounts of data.

These results align with the well-established scaling laws in transformer architectures, where increasing the quantity of training data leads to improved model performance [20]. The transformer-based knolling model benefits from exposure to a more extensive variety of object arrangements and spatial configurations, allowing it to learn more nuanced representations of tidiness and to better predict optimal object placements.

The implications of this experiment are significant for the future development of robotic tidiness tasks. The observed trend suggests that further increasing the dataset size could continue to enhance the model's performance. Moreover, by incorporating richer data representations, such as three-dimensional spatial information, the model could be extended to handle more complex environments.

This scaling behavior opens the possibility of generalizing our approach beyond tabletop organization to the tidying of entire rooms. With 3D data, the model could learn to navigate and organize objects within a full spatial context, considering factors like object height, room layout, and furniture placement. This would enable robots to perform comprehensive housekeeping tasks, such as arranging furniture, organizing shelves, and decluttering living spaces, ultimately moving closer to autonomous robots that can assist with daily household chores. To support reproducibility and transparency, we note that all training runs were conducted on a single NVIDIA RTX 3090 GPU. Training on 1 million samples required approximately 48 hours.

Table 5: Test Error vs Dataset Size

	Mean	**Std**	**Min**	**Max**
125K	6.46E-04	1.82E-03	1.39E-15	3.17E-02
250K	5.81E-04	1.81E-03	0.00E+00	3.21E-02
500K	5.18E-04	1.79E-03	1.39E-15	3.53E-02
1M	4.29E-04	1.61E-03	0.00E+00	3.49E-02
1M-Fine	3.57E-04	1.54E-03	0.00E+00	3.44E-02

K Knolling Pipeline And Training Objective Details

The visual perception model is a customized YOLO v8 model [35], fine-tuned to predict the information of the objects, including the segmentation, color, and category. We post-process the segmentation to a rectangular box and then extract the object state, including the center position, orientation, and dimensions. Detailed evaluation results and confusion matrices for the visual perception model in both simulated and real-world environments are provided in the supplementary materials. The data required for training our models is generated within the Pybullet simulation environment for pre-training and collected in the real world for fine-tuning. To prepare our Visual Perception Model for real-world deployment and address the simulation-to-reality gap, we apply the visual domain randomization technique to our data collection process [41]. This technique helps in enhancing the model's adaptability and robustness by introducing variations and uncertainties that mimic real-world conditions. We manipulate several factors within the simulation environment, such as brightness and ground textures. Using this approach, we train our model to disregard extraneous noise and focus on essential features.

The robotic arm controller operates in four modes: movement, grasping/releasing, table sweeping, and object separation. It uses the predicted object positions from the knolling model and the current poses from the perception module to execute smooth pick-and-place operations, clearing occupied spaces through sweeping motions when necessary.

This integrated pipeline demonstrates how our self-supervised knolling model can be leveraged for practical robotic organization tasks, enabling autonomous tidying of cluttered environments without explicit human supervision or target specifications.

K.1 Training Objectives

Our training strategy incorporates multiple loss functions to guide the model toward accurate, diverse, and physically feasible object placements. These include log-likelihood loss to capture multimodality, MSE-based losses to encourage precision, and specialized penalties to discourage overlap and encourage solution diversity. The specific formulation of each loss component, including detailed equations and justifications, is provided in the Supplementary Material.

L Training Pipeline Details

Pre-training with Self-supervised Learning. To enable the model to learn underlying structural relationships in object arrangements without explicit supervision, we employ a self-supervised pre-training strategy. In this phase, we mask partial object locations and train the model to predict the next object's position based solely on the existing object states. This approach is analogous to masked language modeling in NLP, where models learn to predict missing words in a sentence [9]. By training the model to predict masked object positions, it learns the spatial dependencies and organizational patterns inherent in tidy arrangements.

In the pre-training phase, the model is exposed to partially organized scenes, where it needs to predict the placement of the next object given the current arrangement. Formally, let $O = o_1, o_2, \ldots, o_N$ be the set of N objects, and $P = p_1, p_2, \ldots, p_{N-1}$ be the given positions of the first $N-1$ objects. The pre-training objective is to predict the position p_N of the remaining object o_N:

$$p_N = f_{\text{pre-train}}(O, P) \tag{7}$$

This pre-training stage allows the model to learn the fundamental principles of spatial organization and object placement without the added complexity of predicting all object positions simultaneously. By masking the position of the next object and training the model to predict it based on the current arrangement, the model learns underlying patterns and relationships in object arrangements without explicit human annotations.

Fine-tuning the Knolling Model. The fine-tuning phase is dedicated to refining the model's ability to execute complete knolling tasks from scratch, starting with cluttered or disorganized scenes. In this stage, the model is tasked with predicting the positions $P = p_1, p_2, \ldots, p_N$ of all N objects given only their dimensions $O = o_1, o_2, \ldots, o_N$:

$$P = f_{\text{fine-tune}}(O), O = \{o_n | o_n = (w_n, l_n), n \in \mathbb{N}^+\}. \tag{8}$$

During fine-tuning, the model leverages the knowledge acquired during pre-training to predict the positions of all objects in an autoregressive manner. At each step, the model predicts the position of the next object based on the previously predicted positions and the objects' dimensions. This process can be formalized as:

$$p_n = f_{\text{fine-tune}}(O, p_1, p_2, \ldots, p_{n-1}), \quad n = 1, 2, \ldots, N \tag{9}$$

By progressively predicting the positions of all objects, the model refines its understanding of complete tidy arrangements, enabling it to handle arbitrary numbers of objects and diverse configurations. The details about dataset are shown in the supplementary materials. The training process leverages a combination of five loss functions to train a model to predict object positions without overlaps while also promoting diversity in predictions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract and introduction, we claim the contributions, including our proposed self-supervised framework, use of transformers with GMMs for multi-target placement, and deployment on real robots.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in the conclusion and within the experimental analysis, noting issues such as the scope of generalization to 3D environments and the difficulty of capturing subjective preferences.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include formal theoretical proofs or theorems; it focuses on an empirical self-supervised learning system.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide a complete description of the architecture, dataset generation, loss functions, and training procedures in Section 4 and supplementary materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will release our code and the full dataset of tidy arrangements with documentation upon publication. The GitHub repo is ready: `https://github.com/H-Y-H-Y-H/knolling_bot`

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe the training and test splits, architecture, loss weights, optimizer, and domain randomization settings in code and technical details in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard deviations in all quantitative tables and discuss variance due to object count and architecture. See Table 3 and 4

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute settings are disclosed in the supplementary material. Training used a single NVIDIA RTX 3090 GPU for 48 hours for 1M samples. The details shown in the experiments section

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research complies with the NeurIPS Code of Ethics, particularly in terms of dataset transparency and open-source release.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss potential applications in household robotics.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our dataset and models do not pose high misuse risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit existing datasets (e.g., YOLOv8) and pretrained weights and use assets compliant with MIT and CC-BY licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce a novel knolling dataset by ourselves and will release full documentation and scripts to support usage.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve human subjects or crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: his research does not involve any human subjects or studies requiring IRB approval.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were not used in the methodology.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.