
IKEA Furniture Assembly Environment for Long-Horizon Complex Manipulation Tasks

Youngwoon Lee, Edward S. Hu, Zhengyu Yang, Alex Yin, and Joseph J. Lim

Department of Computer Science

University of Southern California

{ lee504, hues, yang765, alexyin, limjj }@usc.edu

<https://clvrai.com/furniture>

Abstract

The *IKEA Furniture Assembly Environment* is one of the first benchmarks for testing and accelerating the automation of complex manipulation tasks. The environment is designed to advance reinforcement learning from simple toy tasks to complex tasks requiring both long-term planning and sophisticated low-level control. Our environment supports over 80 different furniture models, Sawyer and Baxter robot simulation, and domain randomization. The IKEA Furniture Assembly Environment is a testbed for methods aiming to solve complex manipulation tasks. The environment is publicly available at <https://clvrai.com/furniture>.

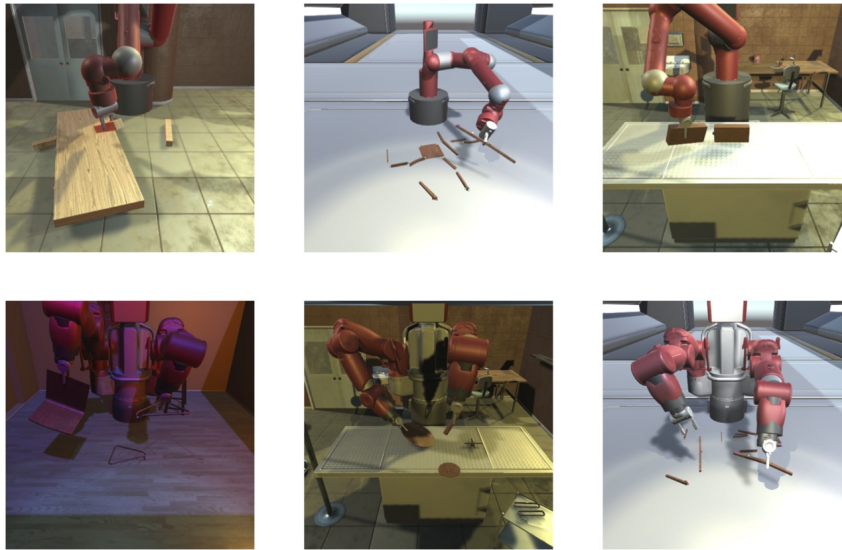


Figure 1: The *IKEA Furniture Assembly Environment* is a furniture assembly simulator. It contains diverse sets of furniture models and robots, and supports various background, lighting, and textures.

1 Introduction

The ability to plan and manipulate physical objects is a necessity to use tools, build structures, and ultimately interact with the world in a meaningful way. Thus, solving long-horizon manipulation tasks has been an active challenge in robot learning. One main bottleneck is the lack of a "standardized"

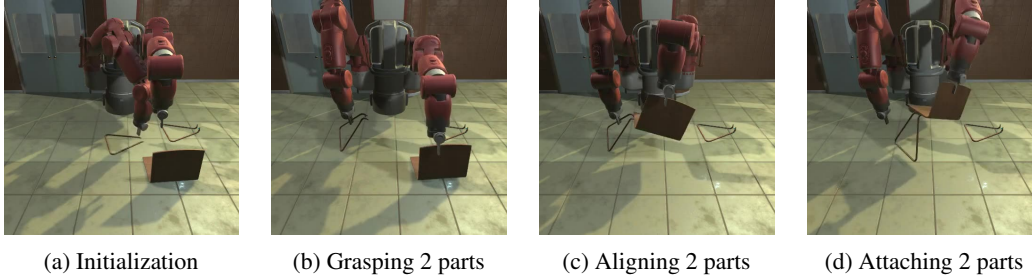


Figure 2: The proposed IKEA Furniture Assembly Environment simulates robotic furniture assembly. The robot decides which parts to attach (a), grasps the desired parts (b), aligns the grasped parts (c), and then attaches the grasped parts (d).

simulation environment for long-term physical tasks. Such a simulator for robot learning needs to have the following properties: long-horizon hierarchical tasks for planning, high-quality rendering for sim-to-real transfer, variable agents and dynamic domains.

Imagine that you are building IKEA furniture. It is not trivial to figure out how to assemble pieces into the final configuration given only a glimpse of fully constructed furniture. Specifically, it is not apparent from pieces on the floor which parts to choose for attachment and in what order. Hence, we need to dissect the final configuration and deduce the sequence of tasks necessary to build the furniture by comparing the current and final configuration. Moreover, connecting two parts requires complicated manipulation skills, such as accurate alignment of two attaching points and sophisticated force control to firmly attach them. Therefore, furniture assembly is a comprehensive task requiring reliable perception, high-level planning, and sophisticated control, making it a suitable benchmark for robot learning algorithms.

To this end, we introduce the *IKEA Furniture Assembly Environment* as a new benchmark for complex autonomous manipulation skills. The IKEA Furniture Assembly Environment is a visually realistic environment that simulates the task of furniture assembly as a step toward autonomous manipulation. The furniture assembly task involves not only high-level 3D scene understanding and step-by-step planning but also sophisticated low-level control. Figure 1 shows examples of our rendered environments.

A variety of research problems could be investigated with this new environment, namely perception, planning, and control. For perception, the environment could be used to solve 3D object detection, pose estimation, instance segmentation, scene graph generation, and shape estimation problems. For robotic control, the environment is suitable for testing multi-agent reinforcement learning, hierarchical reinforcement learning, model-based reinforcement learning, imitation learning, and sim-to-real algorithms for long-term complex manipulation tasks.

The IKEA Furniture Assembly Environment simulates over 80 furniture models. The environment supports multiple agents such as Cursor, Baxter and Sawyer robots. To test generalization, the environment supports domain randomization in the furniture, physics, lighting, textures, and more factors of variation found in the real world. We also outline in the conclusion promising future directions for complex manipulation tasks.

2 IKEA Furniture Assembly Environment

To advance reinforcement learning from simple, videogame-esque tasks to complex and realistic tasks, our environment features long-horizon, hierarchical tasks, realistic rendering, domain randomization, and accurate robotic simulation. The furniture assembly task is composed of (1) selecting two compatible parts, (2) grasping these two parts, (3) aligning the attachable spots (namely connectors) of the parts, and (4) connecting the parts as seen in Figure 2. The furniture assembly task can be accomplished by repeating this process until all parts are assembled. We first discuss various research topics that can utilize our environment, and then present the design and implementation of the environment.

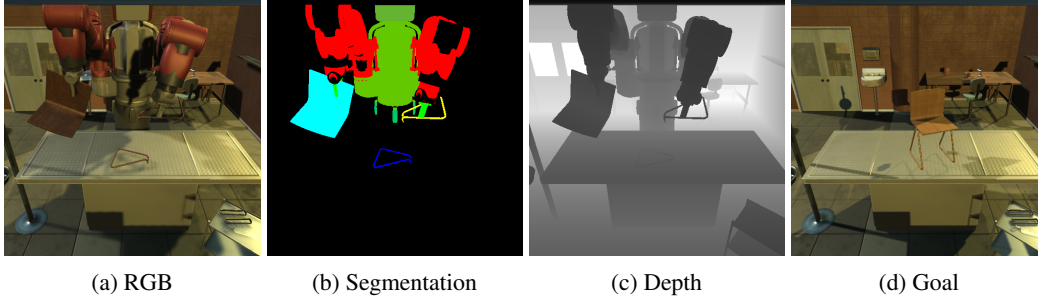


Figure 3: An RGB image, part segmentation, depth map, goal image, and agent state (joint positions and velocities) are available observations. The agent can interact with the environment by either generating torque for each joint or choosing a motion primitive to attach the two parts by aligning the connectors.

2.1 Target Research Topics

The IKEA Furniture Assembly environment simulates complex long-term robotic manipulation tasks with 80+ furniture models and different robots. Moreover, the environment features realistic rendering, configurable scene (e.g., lighting, texture, color), and diverse ground truth labels, such as object pose, shape, instance segmentation mask, depth map, and scene graph. Hence, the environment can be used not only for learning planning and control but also for learning perception as following:

- **Computer Vision:** The environment can generate an abundant amount of data with diverse labels, such as object pose, shape, instance segmentation mask, depth map, part configurations, and scene graph. This synthetic data can be used to tackle many computer vision problems, such as object pose estimation, semantic segmentation, scene graph understanding, shape estimation.
- **Control:** The furniture assembly task requires sophisticated manipulation skills. We support impedance control, inverse kinematics for task space control, as well as keyboard control for humans. Thus, the proposed environment can be used as a challenging benchmark for reinforcement learning and imitation learning methods. Bi-manual or multi-agent manipulation of furniture is another interesting direction.
- **Planning:** The furniture assembly consists of multiple steps of assembling two furniture parts and thus has a long horizon. To tackle this long-horizon task, an agent should learn to plan using approaches from model-based planning and hierarchical reinforcement learning.
- **Others:** The visual and interaction data can be collected to acquire domain knowledge, such as intuitive physics, for other applications. Language-guided robot learning [1, 2] is also a promising research direction to solve complex tasks.

2.2 Environment Development

To cover several challenges including 3D alignment of various shapes of objects and long-horizon robotic manipulation, we developed a novel 3D environment that supports assembling IKEA furniture models using MuJoCo [3] as a physics simulator and Unity3D game engine as a renderer. MuJoCo provides fast and accurate physics simulation, while Unity3d has superior texture and lighting configuration. To exploit the strengths of both frameworks, we use MuJoCo as the underlying physics engine and Unity3d as the renderer. To enable the robotic simulation of Sawyer and Baxter arms in MuJoCo, we use the Robosuite framework [4]. We also use DoorGym [5] to integrate Unity rendering engine with MuJoCo.

2.3 Assembly Simulation

The simulation environment follows the OpenAI Gym [6] protocol where an environment takes an action as input and takes a step. Robotic arms can move around the environment and interact with furniture parts. In addition to actions for robotic arm moves, our environment has a *connect* action. During each step, the environment checks all pairs of connectors. Currently, we only support

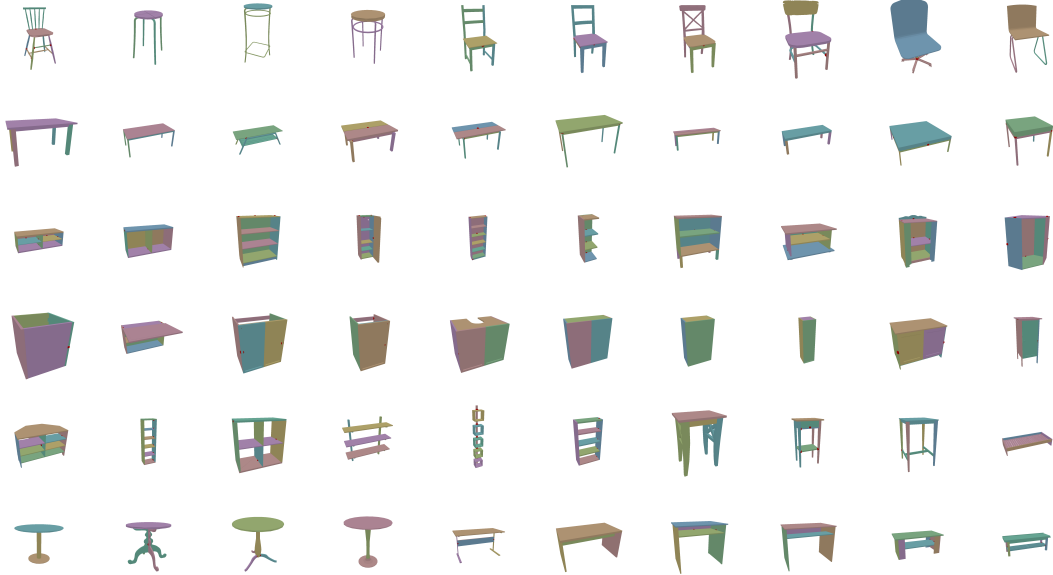


Figure 4: The IKEA furniture assembly environment includes diverse set of furniture models. Each furniture is modeled following the IKEA’s user’s manual and different parts are colored differently.

one-to-one mapping between connectors and plan to implement many-to-many mappings for identical parts. If a pair of connectors have matching IDs and the connectors are within a positional and angular distance threshold, the pair of connectors are flagged as attachable. Specifically, we check the euclidean distance between the connector coordinates (Equation (1)), the cosine distance between the connector up vectors (Equation (2)), and the cosine distance between the connector forward vectors (Equation (3)). If the *connect* action is activated, then the attachable parts are connected using the MuJoCo weld mechanism. The thresholds for distance and angle are configurable. For full details, refer to the `_is_aligned` function in the `furniture.py` file.

$$Pos = d_{L2}((x, y, z)_A, (x, y, z)_B) < \epsilon_{\text{distance}} \quad (1)$$

$$Up = d_{\cos}(Up_A, Up_B) > \epsilon_{\text{up}} \quad (2)$$

$$Forward = d_{\cos}(Forward_A, Forward_B) > \epsilon_{\text{forward}} \quad (3)$$

$$Attachable = \begin{cases} 1 & \text{if } Pos \wedge Up \wedge Forward \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Tasks and Reward Functions Currently, the environment provides a sparse reward for successfully connecting two furniture parts. Users can easily implement their own reward functions and use the rewards for training an agent. For example, the `BAXTER-BLOCK-PICK` task implemented in the `furniture_baxter_block.py` file computes the reward function composed of 10 different components including the distance between the gripper and the target object, angle of the gripper, height of the target object, and gripper’s state.

2.4 Furniture Models

Our environment provides a simulation of the furniture assembly of over 80 different furniture models as shown in Figure 4. We define the furniture as a combination of parts and connectors.

Parts The furniture parts are modeled by following the IKEA’s official user’s manuals with a minor simplification in small details such as carving and screws. The furniture models are created using 3D

modeling tool Rhino¹ and each furniture part is converted to a separate 3D mesh file in a format of STL. Given an STL 3D mesh file, MuJoCo can represent the mesh as a physical object. Each furniture part is represented as one or more meshes. Concave furniture parts need to be represented with multiple meshes, as MuJoCo only supports convex meshes for collision detection. To enable collision of a concave mesh, we split a concave mesh into multiple convex meshes with STL editing software or use MuJoCo’s primitive meshes (e.g., box, cylinder, sphere) only for the collision detection.

Connectors A pair of connectors define the connection information between two furniture parts. The connectors are located on the parts and serve as areas of attachment. On a given part, we parameterize connectors with their ID, size, position, and orientation to the part. All furniture parts and connectors are defined through the MuJoCo XML file. Refer to the XML documentation in the codebase for information and examples on the schema.

Assembly Information To handle part connection, we define in the XML the connector constraints (e.g. Part A can connect to Part B in a certain pose) with the MuJoCo *weld* equality constraint, which allows for welding together two given parts. Once all parts and their connector constraints are specified, the furniture model is complete and can be loaded into the MuJoCo simulator for simulation.

2.5 Agents

The environment supports a variety of agents for interacting with furniture. Currently, three agents are available: Cursor, Sawyer, and Baxter, as illustrated in Figure 5. Their *action spaces* and *observation spaces* are fully configurable to fit a variety of problem settings. The observation space can consist of agent state (e.g., joint positions and velocities), environmental state (e.g., coordinates and rotation of furniture parts), and a camera observation. Aside from RGB images, the environment also supports object segmentation masks and depth camera observations as seen in Figure 3. In general, the action space consists of movement, selection, and attachment primitives.

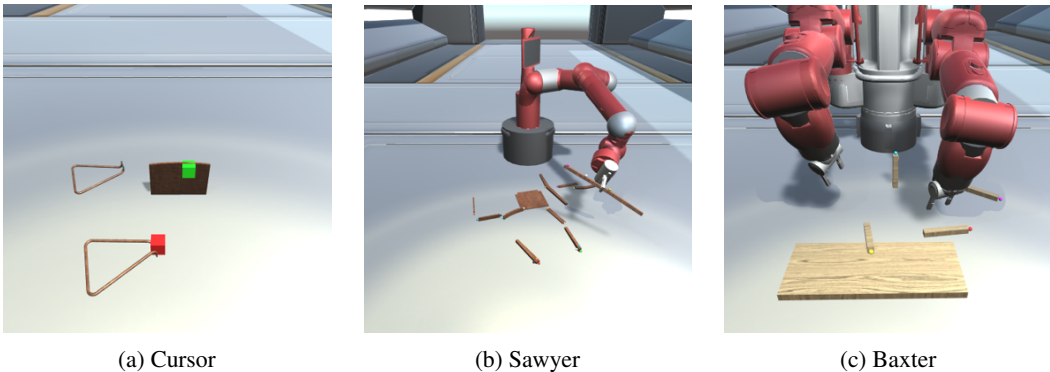


Figure 5: The Cursor, Sawyer, and Baxter agents are supported. We plan to add Fetch, UR, Jaco, and more robots in the next update.

Cursor The *Cursor* agent abstracts away the low-level object grasping problem. Composed of two floating and collision-free cursors in the environment, the agent can move the cursors in the (x,y,z) -axes, and hold parts that are encapsulated by the cubes. Parts that are held by the cursor can be rotated on the (x,y,z) -axes. Movement and rotation can either be applied continuously or in discrete steps depending on the configuration. This agent is suitable for methods that focus on the planning and reasoning portion of the furniture assembly problem by abstracting away the low-level object manipulation problem. The internal state contains the (x,y,z) -coordinates of the cursor.

Sawyer and Baxter The Sawyer and Baxter robots are available for simulation in the environment. Sawyer uses 7 DoF robotic arm, while the Baxter robot has two 7 DoF arms. They can be controlled either through impedance control and task space control (end-effector pose). For task space control,

¹<https://www.rhino3d.com/>

we use the PyBullet library [7] to calculate inverse kinematics for the end-effector control. Unlike the cursor agent, object selection is not abstracted, and objects need to be grasped realistically by the grippers using contact forces. The internal state contains the joint angles and gripping status of the robot. An egocentric or third-person viewpoint can be selected for the camera. RGB, segmentation, and depth modes are available for the camera.

2.6 Domain Randomization

To reduce the reality gap [8], all the furniture models are created following the IKEA’s official user’s manuals with a minor simplification in small details such as carving and screws. For generalization of the learned skills, the environment should provide enough variability in furniture compositions, visual appearances, object shapes, and physical properties. For example, the environment will contain a diverse set of furniture including chair, table, cabinet, bookcase, desk, shelf, and tv unit (see Figure 4). For given furniture, the environment can randomly spawn subsets of the furniture parts, and randomly initialize their positions and orientation them in the scene to increase generalization. In addition to random part selection and placement, the environment can also randomize physics such as gravity and friction to add more variation in the task. The environment will also support the randomization of visual properties like lighting, background colors, texture, and more. Figure 6 illustrates examples of domain randomization.

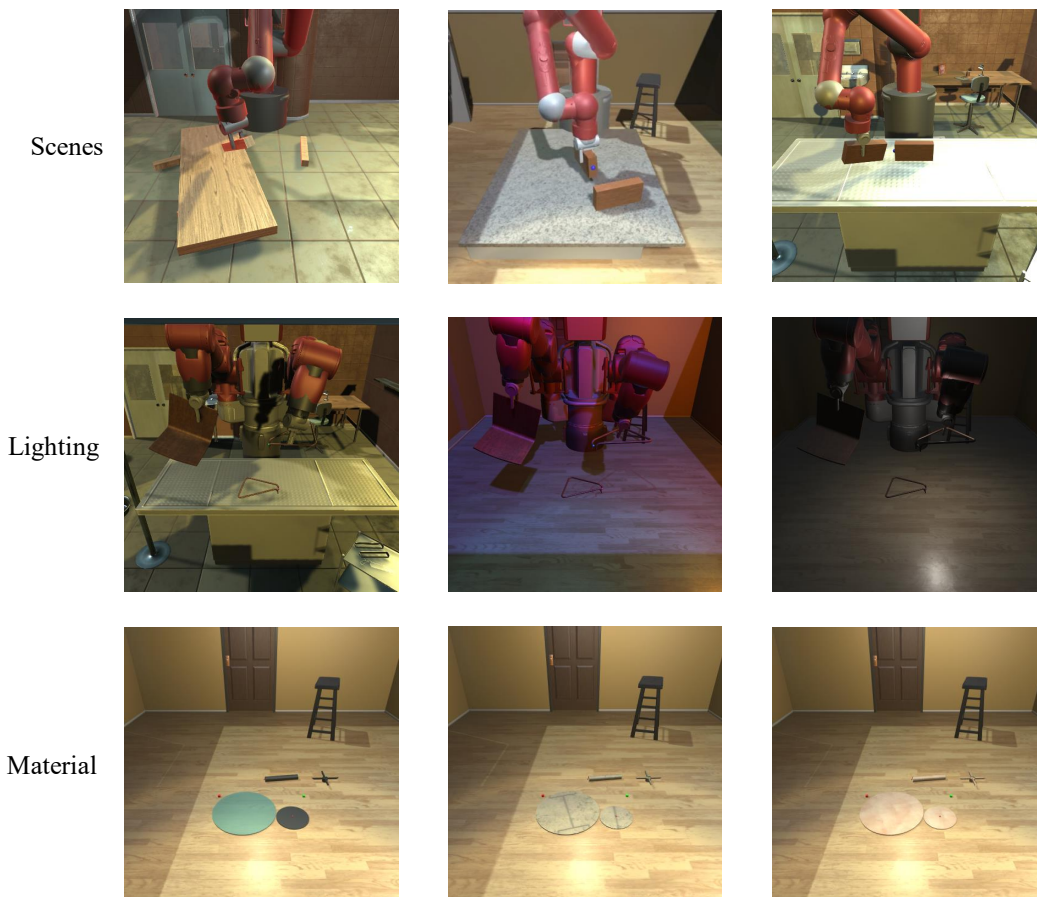


Figure 6: Examples of visual randomization. The first row shows different scenes such as indoors, industrial, and lab settings. The second row shows different lighting configurations like interior, low-visibility, and ambient lighting. The final row shows variations in furniture textures like plastic, marble, and sandstone.

3 Limitations and Future Work

Our environment contains several limitations. First, connection process of two furniture parts are not physically simulated due to the difficulty of physics simulation of screwing. Instead of simulating screws, the environment allows two parts to be connected if two connectors of parts are close and well-aligned as described in Equation (4). Next, identical furniture pieces are considered differently and one connector can only be connected to the predefined furniture part, unlike the real world where the identical pieces can be used interchangeably. In the next release, we plan to allow a furniture part replacing another identical part.

The followings are possible future extensions:

- **3D motion devices:** VR controllers and 3D mouses can be used to teleoperate robots to collect human demonstration. The data can be used for training robot agents.
- **Additional robot support:** more diverse robots, such as Fetch, UR, and higher DoF grippers, can be trained using the proposed environment.
- **Realistic part attachment:** the connection mechanism can be more realistic by implementing peg insertion, screwing, nailing mechanisms.
- **Tool use:** an agent can learn to use a screwdriver or a hammer for assembling furniture.
- **Multi-agent assembly:** multiple robots collaborate to assemble complex furniture.
- **Instructions and demonstrations:** language and visual instructions or demonstrations can be used to guide an agent to solve a complex manipulation task.

4 Related Work

4.1 Simulated Environments for Reinforcement Learning and Robot Learning

Reinforcement learning has made rapid progress with the advent of standardized and simulated environments. Most progress has been made in game environments, such as Atari [9], OpenAI gym [6], VizDoom [10], and StarCraft2 [11]. Recently, many simulated environments have been introduced in diverse applications, such as autonomous driving [12, 13], indoor navigation [14, 15], continuous control [16–18], and recommendation systems [19].

In robotics, most existing environments focus on short-term object manipulation tasks, such as picking and placing [20, 21], in-hand dexterous manipulation [22, 23], door opening [5], and peg inserting [24]. Recent advancements in simulators have made a push towards more complex and realistic tasks. [4] takes a step towards a comprehensive manipulation simulator by offering a variety of manipulation tasks. However, the tasks, consisting of lifting, stacking, pick and place, are still limited to primitive skills.

Composite manipulation tasks such as block stacking [25, 26] and ball serving [18] have also been proposed. However, these tasks have small variations in shapes and physical properties of objects. Metaworld and RL Bench [27, 28] offer environments with diverse but simple manipulation tasks. In contrast, we propose a complex manipulation task with a hierarchical task structure and long horizon, *furniture assembly*, which requires long-term planning and generalizable skills for various shapes, textures, and materials of objects.

4.2 Domain Randomization

The bottleneck of transferring an agent trained in simulation to real-world robots is the reality gap between simulation and the real world [8]. Recently, domain randomization technique [29] has been proposed to reduce the reality gap by training the policy on multiple instances of data differing in textures, backgrounds, lighting and more. Once trained with enough variation, the policy should be able to generalize to the real world [30–32, 22, 5]. To enable transfer learning of the challenging furniture assembly task, the IKEA Furniture Assembly Environment supports diverse and configurable textures, backgrounds, and lighting. Moreover, we provide over 80 furniture models with 3 different robots for further domain randomization.

4.3 Furniture Assembly Datasets

Solving the furniture assembly task requires the solving of multiple subproblems, such as perception, planning, and control. Each of these sub-problems are nontrivial and requires domain knowledge and supervision. For recognizing furniture models and their structures, IKEA 3D model dataset [33] provide labeled data about 3D structures of furniture models and part information. The IKEA furniture assembly video dataset [34], which consists of 480,000 video frames of humans assembling a table from IKEA, can be used as demonstration data for imitation learning. Our proposed environment can be also used to generate synthetic training data for a wide range of 3D perception models with instance segmentation masks, depth image, and furniture part information under diverse furniture models and backgrounds. The expert demonstration videos and trajectories can be collected and used for imitation learning, similar to Roboturk [35].

4.4 Physics Simulation Frameworks

Mujoco [3], Unity ML-Agents [36], PyBullet [7], and Dart [37] are some popular frameworks for physics simulation for reinforcement learning. We chose to use MuJoCo due to fast simulation and its widespread usage in the reinforcement learning community. MuJoCo also supports a Unity plugin for rendering MuJoCo scenes, enabling realistic lighting, scene, and background variations for domain randomization [29].

5 Conclusion

In this paper, we propose the IKEA Furniture Assembly Environment as a novel benchmark for testing complex manipulation tasks. Furniture assembly is a complex manipulation task even for humans, requiring perception capabilities, high-level planning, and sophisticated low-level control. Therefore, it is well suited as a benchmark for control algorithms aiming to solve complex tasks.

We present several directions for future research that are necessary for completely automating furniture assembly.

Representation: Currently, many manipulation methods still require the pose of the object. This is difficult to obtain in the real world. On the other hand, images of the scene are easy to get but obtaining relevant features is difficult due to its high dimensionality. To solve furniture assembly in the real world, extracting object-centric representations from images is a promising direction. These representations may have geometric, relational, and semantic information [38–40]. Unsupervised object discovery with deep neural networks [40, 41] can be also helpful.

Planning: While reinforcement learning has seen success in manipulation tasks, the manipulation tasks are short in horizon and do not require complex planning. Furniture assembly on the other hand, requires long-term planning such as deciding the ordering of parts to assemble. Hierarchical reinforcement learning can segment complex tasks into simpler subtasks, which may be conducive towards furniture assembly [42, 18]. Model-based reinforcement learning could be another direction for tackling the long-horizon problem by giving the model a mechanism to predict the future before taking an action [43].

Dexterous Control: Dexterous manipulation is challenging due to agent and object complexity. Real world objects have irregular shapes, textures, friction, and other physical characteristics. Some dexterous robotic hands have between 24-30 degrees of freedom. Model free methods have seen success in controlling such dexterous hands [44, 23] but gaining full control of robotic hands and grasping arbitrary objects is an open challenge. Multi-agent manipulation of furniture is also another promising direction.

Domain Knowledge: Instruction manuals [45, 1], programs [26, 46], and video demonstrations [21] can be used as domain knowledge to help solving complex tasks. The question of how to integrate additional supervision into an automated furniture assembly process is an interesting future direction.

Acknowledgments

The authors would like to thank Taehoon Kim and many members of the USC CLVR lab for helpful feedback and testing our environment. This project was partially funded by SKT.

References

- [1] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In International Conference on Learning Representations, pages 2661–2670, 2017.
- [2] Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. Gated-attention architectures for task-oriented language grounding. arXiv preprint arXiv:1706.07230, 2017.
- [3] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5026–5033, 2012.
- [4] Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In Conference on Robot Learning, pages 767–782, 2018.
- [5] Yusuke Urakami, Alec Hodgkinson, Casey Carlin, Randall Leu, Luca Rigazio, and Pieter Abbeel. Doorgym: A scalable door opening environment and baseline agent. arXiv preprint arXiv:1908.01887, 2019.
- [6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. arXiv preprint arXiv:1606.01540, 2016.
- [7] Erwin Coumans. Bullet physics simulation. In ACM SIGGRAPH 2015 Courses, SIGGRAPH '15, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3634-5. doi: 10.1145/2776880.2792704. URL <http://doi.acm.org/10.1145/2776880.2792704>.
- [8] Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In European Conference on Artificial Life, pages 704–720. Springer, 1995.
- [9] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research, 47:253–279, jun 2013.
- [10] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In IEEE Conference on Computational Intelligence and Games, pages 341–348, 2016.
- [11] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. arXiv preprint arXiv:1708.04782, 2017.
- [12] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In Field and Service Robotics, 2017. URL <https://arxiv.org/abs/1705.05065>.
- [13] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Conference on Robot Learning, pages 1–16, 2017.
- [14] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474, 2017.
- [15] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8494–8502, 2018.
- [16] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. arXiv preprint arXiv:1801.00690, 2018.

- [17] Dibya Ghosh, Avi Singh, Aravind Rajeswaran, Vikash Kumar, and Sergey Levine. Divide and conquer reinforcement learning. In International Conference on Learning Representations, 2018.
- [18] Youngwoon Lee, Shao-Hua Sun, Sriram Somasundaram, Edward S. Hu, and Joseph J. Lim. Composing complex skills by learning transition policies. In International Conference on Learning Representations, 2019. URL <https://openreview.net/forum?id=rygrBhC5tQ>.
- [19] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. arXiv preprint arXiv:1808.00720, 2018.
- [20] Yuke Zhu, Ziyu Wang, Josh Merel, Andrei Rusu, Tom Erez, Serkan Cabi, Saran Tunyasuvunakool, János Kramár, Raia Hadsell, Nando de Freitas, and Nicolas Heess. Reinforcement and imitation learning for diverse visuomotor skills. arXiv preprint arXiv:1802.09564, 2018.
- [21] Youngwoon Lee, Edward S. Hu, Zhengyu Yang, and Joseph J. Lim. To follow or not to follow: Selective imitation learning from observations. In Conference on Robot Learning, 2019.
- [22] Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. arXiv preprint arXiv:1808.00177, 2018.
- [23] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In Robotics: Science and Systems, 2018.
- [24] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. arXiv preprint arXiv:1810.05687, 2018.
- [25] Yan Duan, Marcin Andrychowicz, Bradly Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In Advances in Neural Information Processing Systems, pages 1087–1098, 2017.
- [26] Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In IEEE International Conference on Robotics and Automation, pages 1–8. IEEE, 2018.
- [27] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Conference on Robot Learning (CoRL), 2019.
- [28] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. arXiv preprint arXiv:1909.12271, 2019.
- [29] Fereshteh Sadeghi and Sergey Levine. Cad2rl: Real single-image flight without a single real image. In Robotics: Science and Systems, 2017.
- [30] Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. arXiv preprint arXiv:1610.01283, 2016.
- [31] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 23–30. IEEE, 2017.
- [32] Fereshteh Sadeghi, Alexander Toshev, Eric Jang, and Sergey Levine. Sim2real view invariant visual servoing by recurrent control. arXiv preprint arXiv:1712.07642, 2017.
- [33] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In IEEE Conference on Computer Vision and Pattern Recognition, pages 2992–2999, 2013.

- [34] Tengda Han, Jue Wang, Anoop Cherian, and Stephen Gould. Human action forecasting by learning task grammars. arXiv preprint arXiv:1709.06391, 2017.
- [35] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, Silvio Savarese, and Li Fei-Fei. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. CoRR, abs/1811.02790, 2018. URL <http://arxiv.org/abs/1811.02790>.
- [36] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Matar, and Danny Lange. Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627, 2018.
- [37] Jeongseok Lee, Michael X Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S Srinivasa, Mike Stilman, and C Karen Liu. Dart: Dynamic animation and robotics toolkit. The Journal of Open Source Software, 3(22):500, 2018.
- [38] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 3588–3597, 2018.
- [39] Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Deep reinforcement learning with relational inductive biases. In International Conference on Learning Representations, 2019. URL <https://openreview.net/forum?id=HkxaFoC9KQ>.
- [40] Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, pages 6691–6701, 2017.
- [41] Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. CoRR, abs/1901.11390, 2019. URL <http://arxiv.org/abs/1901.11390>.
- [42] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial intelligence, 112(1-2): 181–211, 1999.
- [43] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. ACM Sigart Bulletin, 2(4):160–163, 1991.
- [44] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In Advances in Neural Information Processing Systems, pages 5048–5058, 2017.
- [45] Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. In International Conference on Machine Learning, pages 166–175, 2017.
- [46] Shao-Hua Sun, Hyeonwoo Noh, Sriram Somasundaram, and Joseph J. Lim. Neural program synthesis from diverse demonstration videos. In International Conference on Machine Learning, 2018.