Break and Make: Interactive Structural Understanding Using LEGO Bricks Supplementary Material

Aaron Walsman,[1] Muru Zhang[1], Klemen Kotar[2], Karthik Desingh[1], Ali Farhadi[1], and Dieter Fox[1,3]

[1] University of Washington awalsman@cs.washington.edu
[2] Allen Institute for Artificial Intelligence
[3] NVIDIA

A Qualatative Examples

Figure 1 shows ten randomly chosen target assemblies and assemblies predicted by the Stubnet-B model on each dataset. Very few examples are completed precisely correct. In the random 2-brick, 4-brick and the OMR 2-brick examples, the agent is able to build an assembly using the correct bricks but fails to connect them correctly. The model largely fails to build anything resembling the target for the random 8-brick and OMR 4-brick and 8-brick scenes. This clearly demonstrates the difficulty of this problem even in simple settings, and shows how the problem becomes more difficult as the number of bricks per scene increases, and the total number of bricks used in the dataset increases. See the supplemental video for an example of a successful episode on a 2 brick model generated with the Stubnet-B model.

B Dataset

B.1 Simulator Details and Performance

The **LTRON** simulator uses native Python along with the Splendor-Render rendering package that provides a python interface for OpenGL rendering. The speed of the simulator is dependent on the size and complexity of the scene, but for the scenes used in the experiments here, the simulator runs at over 100fps on an Nvidia 2080ti graphics card, and can be parallelized. The simulator also supports headless EGL rendering for use on clusters that do not have graphical sessions. **LTRON** is not as heavily optimized as some other 3D environments such as Habitat 2.0 [3], but the simulator speed was fast enough not to be a bottleneck during experiments.

B.2 Statistics

We source brick shapes from the LDRAW[1] database, which contains over ten thousand official bricks. However, many bricks in the official parts list are not

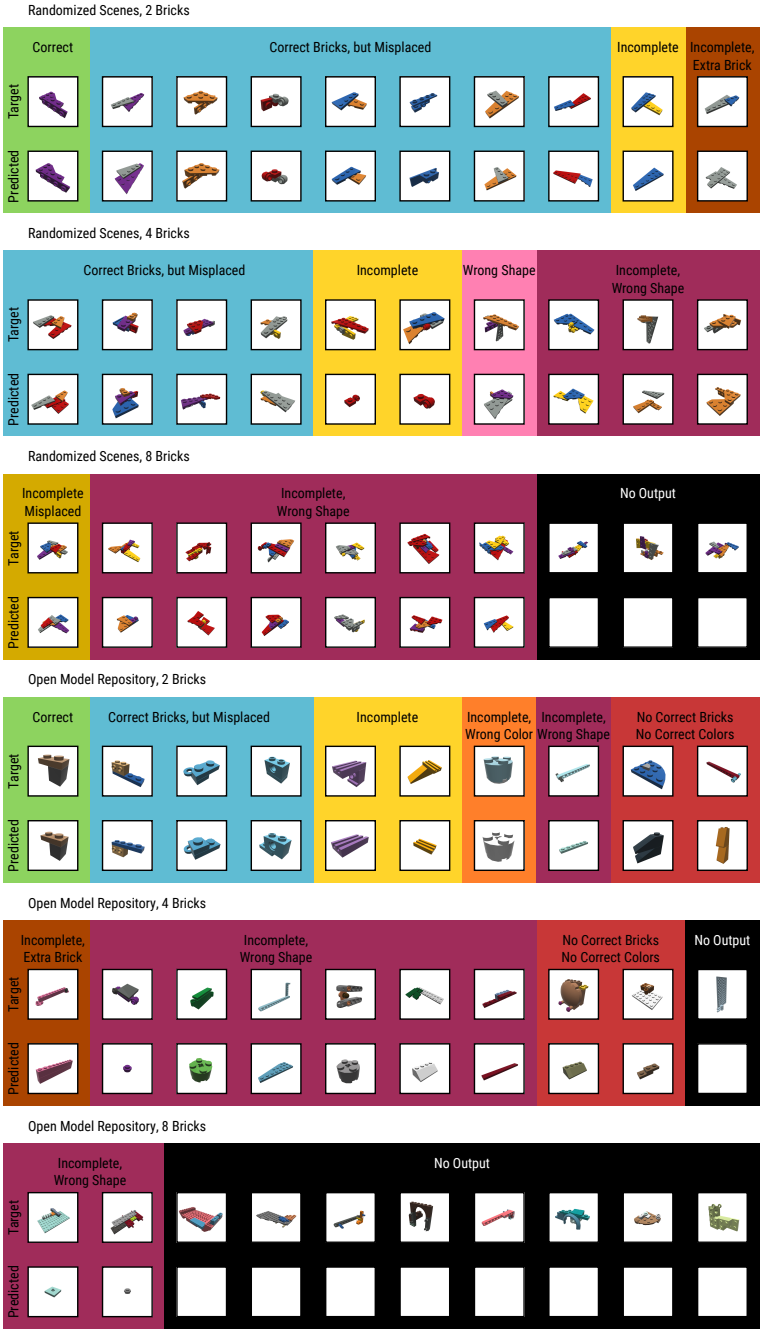

Fig. 1. Qualatative examples.

used in any of the models in the OMR dataset. Thus while **LTRON** supports over ten thousand individual bricks, only around four thousand are used in the files that we slice to produce our training data.

As one might expect, the distribution of part usage has a long tail of increasingly rare brick shapes. To illustrate this, we plot the log of the individual brick usage counts against the log of the rank of these counts. These diagrams are frequently used in language applications to demonstrate long-tail distributional statistics. Figure 2 illustrates the usage distribution.

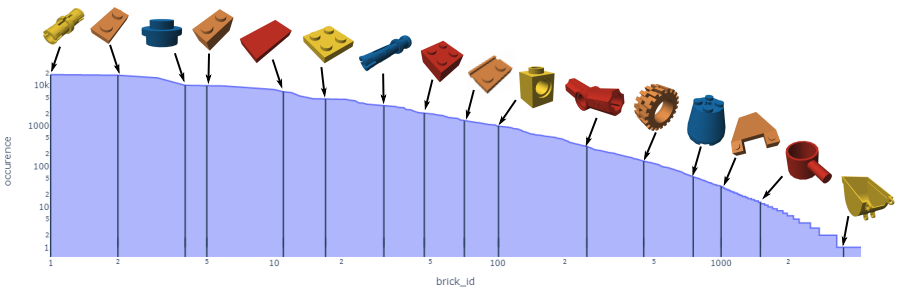

Fig. 2. Distribution of brick frequency in the OMR data with examples of various common and rare bricks. The x-axis is the log-rank of each brick shape sorted by frequency with the most common brick on the left and the least common on the right. The y-axis is the log-frequency of each class. The most common brick is a simple connector pin which is used over 18,000 times throughout the dataset. There are around one hundred classes with one thousand or more examples, and over five hundred classes with one hundred or more examples. There is also a long tail of rare bricks, with more than half of all classes occurring less than ten times.

In order to measure how common various combinations are we also compute statistics of brick *bigrams* which consist of two bricks that are attached to each other with the same relative transform between them. In total we have 222,679 bigrams where the most common one occurs 1,471 times in the data. The bigrams also have a long tail of increasingly rare combinations as seen in Figure 3.

B.3 Data Preprocessing

When working with data from the Open Model Repository, we have taken several steps to make the data more manageable for the learning agent.

First we have blacklisted several bricks that are very large that would not fit into the default screen viewport.

Second, many of the brick shapes in the LDRAW repository represent small changes and variations that have been made to different bricks over the years. Where possible, we have collapsed multiple brick shapes that represent only minor variations into a single class by replacing variations with a single canonical version of a part. This further reduced the number of shapes in our dataset from

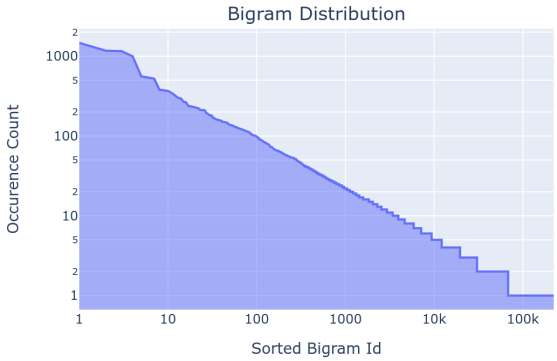

Fig. 3. The distribution of brick bigrams–two brick types and an associated transform–in the data included in **LTRON**.

over four thousand to 1790. Figure 4 shows examples of bricks bricks that have been collapsed into a single category.

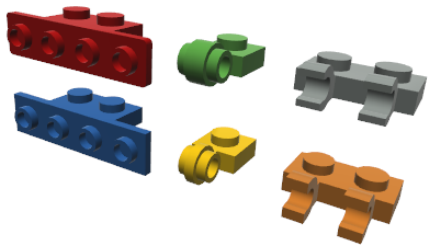

Fig. 4. Examples of brick variants in the LDRAW repository. Note the red brick has slightly rounded corner compared to the blue brick. The thick section between the two connection points is slightly thicker in the green brick than the yellow. The shape of the clips is slightly thinner in the grey brick than the orange. As a preprocessing step we replaced each of these categories with a single canonical version when using these bricks for **Break and Make**.

Third, for each model in the Open Model Repository, we have computed all connected components in the model and split out each as a separate file. These connected components represent groups of bricks that are all attached to each other using the connection points that **LTRON** currently supports. This is important as many official LEGO sets contain multiple detached components, for example a single racing set may contain two separate cars.

Finally, once we have these connected components, we slice them into smaller components in order to generate many examples of small scenes for training.

When slicing we aim to keep the models as small and compact as possible, so we use a simple greedy algorithm that selects the first brick in the scene, then looks at other bricks it is connected to and selects the one that minimizes the largest axis of the bounding box of the new model, and repeats this process until the desired number of bricks has been selected. These bricks are then broken out as their own new model file and removed from the scene. Then this process is repeated until nothing remains of the original model. This process ensures that the resulting sliced models have similar distributional statistics as the original data, not only in terms of individual brick usage, but also local neighborhood structure.

C Symmetry

Many LEGO bricks exhibit rotational symmetry about a one of the three primary X, Y, Z axes. We have generated a table describing these symmetries by automatically analyzing the shape of each brick in order to take symmetry into account when scoring the final models. In order to compute this table, we render a depth map of each brick from six canonical directions (+/- X,Y,Z). We then rotate each brick by 90, 180 and 270 degrees about each of the X, Y and Z axes and re-render these depth maps. We mark any transformation that produces approximately identical depth maps as a symmetry.

Fig. 5. Several examples of models from the Open Model Repository. Each model in the bottom row has over eight hundred bricks, each of which can be individually manipulated within LTRON.

D Ablations and Analysis

Given the relatively low performance of the models presented here on the break and make task, we also conducted several experiments designed to discover which part of this problem is most difficult.

D.1 Ablations

We first modified the environment to provide the correct pick actions (selection of brick shape and color) automatically when necessary. If numbers on this experiment improved dramatically, this would indicate that the models were having difficulty remembering the brick shapes and colors that were observed during the **Break** phase. For the sake of space, we ran this experiment on the randomly constructed 2-brick assemblies using the StudNet-B model. As shown in table 1 this yields a very small performance gain. This result, taken together with the fact that the $F1_b$ score strongly outperforms the $F1_a$ score in almost all experiments indicates that remembering the brick shape and color, and learning when to insert them is not a primary source of error.

We also extracted frames from the random 2-brick assemblies and trained a single-frame FCOS[4] detection model with additional heads to predict 3D position and orientation. This is designed to determine if estimating the 3D pose of the bricks in a single frame is a possible point of failure. To evaluate this model, we use an AP score where we consider an estimated brick to be a true-positive match with a ground truth brick if their shapes and colors match, and their poses are within 30 degrees and 8mm of each other. In this setting a ResNet50 backbone scores 0.97 AP, a ResNet18 backbone scores 0.96 AP and a transformer backbone scores 0.81 AP. These results indicate that estimating the identities and poses of the bricks in a single frame is also not a major challenge.

Taken together, these two results provide indirect evidence for the hypothesis that the most challenging part of this problem is the need for spatially-precise long term memory, and a large interactive action space.

Ground Truth Insertion

Random 2 Brick	$F1_b$	$F1_e$	$F1_a$	AED
StudNet-B Original	0.87	0.77	0.57	1.30
StudNet-B GT-Insert	0.88	0.84	0.57	1.12

Table 1. Test results when providing ground truth brick insertion operations (bottom) compared to original performance (top).

D.2 Fine Tuning

We also conducted an experiment to test whether a network trained on random construction assemblies could be fine-tuned to the OMR assemblies. Due to the

mismatched number of brick shapes and colors, we trained new color and shape heads for the OMR data. As shown in table 2 performance actually gets slightly worse, except for edit distance, which improves slightly.

Pretrain/Fine Tune				
OMR 2 Brick	$F1_b$	$F1_e$	$F1_a$	AED
StudNet-B Original	0.36	0.18	0.29	3.74
StudNet-B Finetune	0.29	0.11	0.25	3.32

Table 2. Test results when using a model pretrained on the Random-2 dataset and fine-tuning on OMR-2 (bottom) compared to training from scratch on OMR (top).

D.3 Human Baseline

In order to make sure the **Break and Make** task is possible using the interface provided, we also conducted a small user-experiment. Using a rudimentary interface, our first author was able to perfectly reconstruct 8 out 10 randomly sampled scenes from the Random Construction 2 dataset. The two failures contained a single placement mistake each which could not be fixed within the maximum episode length. This would yield F1b: 1.0, F1e: 1.0, F1a: 0.90 and an AED: 0.2, which is far better than any of the baseline models. This shows that the task is feasible, and that there is substantial room for improvement in the approaches here.

E Ethical Research and Intellectual Property

The **LTRON** environment makes substantial use of data collected from the internet. Whenever working with data that comes from external sources, it is important to protect individuals' privacy and respect the intellectual property rights of the original authors.

To our knowledge, **LTRON** does not pose a substantial privacy risk to individuals. The only personally identifiable information contained in **LTRON** is a name or pseudonym of the individual author of a particular file. These authors have chosen to make their names public, and have requested attribution through Creative Commons licensing when using their work.

Intellectual Property is an important consideration for this project as each of the files included in **LTRON** represent a substantial investment of time and effort by the original authors. We have therefore only included files that have clear and unambiguous open licensing terms. Fortunately the LDraw Open Model Repository [1] contains over a thousand high-quality files with generous Creative Commons licensing. See Appendix F for authorship details. **LTRON** is inter operable with other user-generated content that can be found scattered throughout various community forums. In most cases these do not contain explicit licensing terms, so we have not included this data for distribution and have

not used these files in our experiments. In the future, we may augment **LTRON** with more data when and if we are able to secure open licensing agreements with individual authors.

LEGO is an official trademark of the LEGO Group which is not affiliated with this paper or the authors, and has not endorsed or sponsored this paper. To our knowledge, all material in **LTRON** has been generated either by the fan community or this paper's authors, and are therefore not under copyright or other intellectual property protection by the LEGO Group.

F Acknowledgements

The following is a list of contributors to the LDraw project [1], the Open Model Repository and the LDCad metadata [2]. These online resources have been created and are maintained by a large group of volunteers. These contributors are not affiliated with, and have not directly contributed to the authorship of this paper, however they have provided open resources which have been of enormous value to the **LTRON** environment. We include their names here in order to acknowledge their efforts and contribution.

LDraw Part Authors: These authors have contributed parts or parts of parts to the primary LDraw database. This database contains the geometric shape of over 13,000 individual brick types, and is used in **LTRON** as the primary source of renderable geometry. The authors are listed here in the order of the number of authored files:

Philippe Hurbain [Philo], Magnus Forsberg [MagFors], James Jessiman, Steffen [Steffen], Chris Dee [cwdee], Michael Heidemann [mikeheide], J.C. Tchang [tchang], Gerald Lasser [GeraldLasser], Alex Taylor [anathema], Guy Vivan [guyvivan], Tore Eriksson [Tore_Eriksson], Willy Tschager [Holly-Wood], Max Martin Richter [MMR1988], Damien Roux [Darats], Andy Westrate [westrate], Christian Neumann [Wesley], Ulrich R der [UR], Steve Bliss [sbliss], Franklin W. Cain [fwcain], Rolf Osterthun [Rolf], Massimo Maso [Sirio], Santeri Piippo [arezey], Marc Klein [marckl], Niels Karsdorp [nielsk], Evert-Jan Boer [ejboer], Johann Eisner [technicbasics], Donald Sutter [technog], Vincent Messenet [Cheenzo], Joerg Sommerer [Brickaneer], Paul Easter [pneaster], Daniel Goerner [TK-949], Bernd Broich [bbroich], Nils Schmidt [BlackBrick89], Stan Isachenko [angmarec], William Howard [WilliamH], John Van Zwieten [jvan], Howard Lande [HowardLande], Thomas Burger [grapeape], Owen Burgoyne [C3POwen], Orion Pobursky [OrionP], Takeshi Takahashi [RainbowDolphin], Mikkel Bech Jensen [gaia], Arne Hackstein, John Riley [jriley], Tim Gould [timgould], Mark Kennedy [mkennedy], Tony Hafner [hafhead], Kevin Roach [KROACH], Tim Lampmann [L4mpi], Greg Teft [gregteft], Christophe Mitillo [Christophe_Mitillo], Leonardo Zide, Sylvain Sauvage [SLS], Merlijn Wissink [legolijntje], Mark Chittenden [mdublade], Carsten Schmitz [Deckard], Jaco van der Molen [Jaco], Luis E. Fernandez [lfernand], Miklos Hosszu [hmick], Sascha Broich, Lutz Uhlmann [El-Lutzo], Ronald Vallenduuk [Duq], Matt Schild [mschild], Thomas Woelk [t.woelk], Manfred Moolhuysen, Ross Crawford [rosco], Bertrand Lequy [Berth], Imre Papp [ampi], Remco

Braak [remco1974], George Barnes [glbarnes], Jude Parrill [theJudeAbides], Marc Giraudet [Mad_Marc], Ludo Soete [ludo], El'dar Ismagilov [Eldar], Bjoern Sigve Storesund [Storesund], Tomas Kralicek [RabbiT_CZ], John Troxler [Gargan], Jonathan Wilson [jonwil], Jeff Boen [onyx], Lutz Uhlmann, Peter Watts [Frozen-Pea], Niels Bugge [SirBugge], Marc Schickele [samrotule], Alexandre Bourdais [x-or], Jeff Boen, Ingolf Weisheit [stahlwollschaf], Andrew Ananjev [woozle], Damien Guichard [BrickCaster], Heiko Jelnikar [KlotzKiste], Yann Bouzon [Zaghor], Stephan Meisinger [smr], Lee Gaiteri [LummoxJR], Marek Idec [Maras], Lars C. Hassing [larschassing], Nathan Wright, Remco Braak, Kevin Clague [kclague], Larry Pieniazek [lar], Matthew J. Chiles [mchiles], Guus-Jan Wijnhoven [guus], Victor Di Rienzo [tatubias], Shimpei Ohsumi [Shimpei-Ohsumi], Christian M. Angele [cma_1971], Sven Moritz Hein [smhltec], Ildefonso Zanette [izanette], Ronald Scott Moody [rmoody], Paul Izquierdo Rojas [pir], Joseph H. Cardana, Taylor Bangs [DoomTay], Peter Lind [peterlinddk], Adriano Aicardi, Jonathan P. Brown, Karim Nassar, Dee Earley [DeannaEarley], Brent Jackson [bjackson], Lance Hopenwasser [cavehop], Bram Lambrecht, Ishino Keiichiro, Joachim Probst, Yu Zhang [ishkafel], David Manley [djm], Joshua Delahunty [dulcaoin], Ignacio Fernandez Galvan [Jellby], Heather Patey, Adam Howard [Whist], N. W. Perry [Plastikean], Matthew Morrison [cuddlyogre], Dennis Osborn, Robert Sexton [rhsexton], Sybrand Bonsma [Sybrand], Remco Canten [rempie], R. M. Rodinsky [dublar], Paul Schelleman [schellie], Bob LeVan [kyphurious], Dave Schuler, Jens Bauer [rockford], Jan Folkersma [Stinky], Graham Wilkes [remorse], Damien Duquennoy, Svend Eisenhardt [eisenhardt], Enzo Silvestri [ienzisolves], Bert Van Raemdonck [BEAVeR], Andreas Weissenburg [grubaluk], Amnon Silverstein [Amnon], Frits Blankenzee, Zoltan Keri [kzoltan82], Alex Forencich [aforencich], Daniele Benedettelli [benedettelli], Rafael Skibicki [Rola], Carlos Arbesu [NXTbesu], Philip Peickert [mr51flip], Sam Roberts [sroberts], Gene Welborn [dtaax], Allister Mclaren [amclaren], Joao Almeida [TullariS], Derrick Chiu [LordAdmiral], Niklas Buchmann [NiklasB], Alex Seeley [alex], Paolo Campagnaro [pcampagn], Ryan Dennett, Maciej Kowalik [Madmaks], Ian Reid [Ian_Reid], Jeff Stembel, C.L.Rasmussen [johnny-thunder], Don Heyse, Chris Moseley, Steve Chisnall [StevieC], Edwin Pilobello [gypsy_fly], John Boozer [jediknight219], Ben Lyttle [legotrek], Reinhard "Ben" Beneke [Ben_aus_BS], Jim DeVona [anoved], Jason Mantor [Xanthra47], James Mastros [theorbtwo], Jeffery MacEachern [legonerd], Axel Poque, John Jensen, Douglas Taylor, Jr. [djcool905], Stig-Erik Blomqvist [stigge72], Jeroen Ottens, Bernd Munding, Steve Demlow [demlow], Bert J. Giesen, Reuben Pearse [ReubenPearse], Robert Paciorek [bercik], Richard Baxter [rbaxter], Dan Boger [dan], Duane Hess, Earnest J. Banbury [Banbury], Martin G Cormier, Jack Hawk [jhawk], Arthur Sigg [Pendulum], James Shields [lostcarpark], Richard Finegold, Martyn Boogaarts, Antony Caparica [antonyc], Christopher Bulliner [CMB27], Christopher Pedersen [pedersen], Troy [peloquin], Travis Cobbs [tcobbs].

LDRAW Open Model Repository Authors: These authors have contributed full models to the LDraw Open Model Repository. This repository contains 1,727 high quality models assembled from the bricks in the LDraw parts

library. Again the authors are listed in the order of the number of files contributed:

Robert Paciorek [bercik], Philippe Hurbain [Philo], Marc Giraudet [Mad_Marc], Massimo Maso [Sirio], Damien Roux [Darats], Merlijn Wissink [Legolijntje], Willy Tschager [Holly-Wood], Orion Pobursky [OrionP], Max Martin Richter [MMR1998], Stefan Frenz [smf], Tomas Kralicek [RabbiT_CZ], Victor Di Rienzo [tatubias], Ken Drew [Ken], Johann Eisner [technicbasics], Bert Van Raemdonck [BEAVeR], Evert-Jan Boer [ejboer], Charles Farmer [farmer], Marc Belanger [MonsieurPoulet], Roland Dahl [RolandD], Oh-Seong KWON, Jude Parill [theJudeAbides], Ignacio Fernandez Galvan [jellby], Faramond Florent [Makou], Daniel Goerner [TK-949], Zoltán Kéri [Zoltank82], Takeshi Takahashi [Rainbow-Dolphin], Stan Isachenko [angmarec], Jaco van der Molen [Jaco], Rijk van Voorst [Rijkjavik], Christian Maglekær, Christian Neumann [Wesley], Steffen [Steffen], N. W. Perry [Plastikean], Michael Heidemann [mikeheide], Michal Oravec [Bloodybeast], Magnus Forsberg [MagFors], Lasse Deleuran [Lasse Deleuran], [juraj3579], Allard van Efferen [aefferen], Rafael Skibicki [Rola], Adrien Pennamen [AdrienPennamen], Greg Teft [gregteft], Christophe Mitillo [Christophe_Mitillo], René Frijhoff, Sean Burke [Leftmost], Antony Lodge, Mirjan Lipovcan [LegoZG], Florian Schüller [schuellerf], Tim Singer [tsinger], [EdmanZA], Guido Mauro, Casey Puyleart, Peter Bartfai, Oliver Damman [Nautilus], Steffen Altenburg [SteffenA], Ulrich Röder [UR], Alexey [folkoluck], Kevin Hendirckx [Gebruiker].

LDCAD Metadata Authors: LTRON uses LDraw metadata bundled with the free LDCAD software to find connection points between bricks. *Roland Melkert* is the primary author of LDCAD, but others have contributed to this metadata. They are listed here, again in order of the number of files contributed:

Roland Melkert [roland], Philippe Hurbain [Philo], Milan Vancura [MilanV], Alex Taylor [anathema], Jason McReynolds [Jason McReynolds].

Additional Acknowledgements: We also wish to acknowledge *Toby Nelson*, the author of the open-source *ImportLDraw* plugin for Blender. This was used to convert the LDraw parts library to wavefront OBJ files for use in our rendering system.

References

1. Jessiman, J., et al.: LDraw. http://www.ldraw.org (2022)
2. Melkert, R.: LDCad. http://www.melkert.net/LDCad (2017)
3. Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D.S., Maksymets, O., et al.: Habitat 2.0: Training home assistants to rearrange their habitat. Advances in Neural Information Processing Systems **34**, 251–266 (2021)
4. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: ICCV (2019)