Google Scanned Objects:
A High-Quality Dataset of 3D Scanned Household Items

Laura Downs¹, Anthony Francis¹, Nate Koenig³, Brandon Kinman¹, Ryan Hickman¹, Krista Reymann¹, Thomas B. McHugh², and Vincent Vanhoucke¹

Abstract—Interactive 3D simulations have enabled breakthroughs in robotics and computer vision, but simulating the broad diversity of environments needed for deep learning requires large corpora of photo-realistic 3D object models. To address this need, we present Google Scanned Objects, an open-source collection of over one thousand 3D-scanned household items released under a Creative Commons license; these models are preprocessed for use in Ignition Gazebo and the Bullet simulation platforms, but are easily adaptable to other simulators. We describe our object scanning and curation pipeline, then provide statistics about the contents of the dataset and its usage. We hope that the diversity, quality, and flexibility of Google Scanned Objects will lead to advances in interactive simulation, synthetic perception, and robotic learning.

Index Terms—Data Sets for Robot Learning, Data Sets for Robotic Vision, Simulation and Animation

I. INTRODUCTION

Deep learning has enabled many recent advances in computer vision and robotics, but training deep models requires diverse inputs in order to generalize to new scenarios [1]. Computer vision has used web scraping to gather datasets with millions of items, including ImageNet [2], Open Images [3], Youtube-8M [4], and COCO [5]; however, labeling these datasets is labor-intensive, labeling errors can distort the perception of progress [6], and this strategy does not readily generalize to 3D or real-world robotic data. Unlike images, the web does not contain a large population of high-quality 3D scenes, real-world data collection is challenging as robots are expensive and dangerous, and human labelers cannot extract 3D geometric properties from images.

Simulation of robots and environments, using tools such as Gazebo [7], Bullet [8], MuJoCo [9], and Unity [10], can mitigate many of these limitations, as simulated environments can be varied safely, and semantic labels can be easily derived from the simulation state. However, simulation is always an approximation to reality: handcrafted models built from polygons and primitives correspond poorly to real objects. Even if a scene is built directly from a 3D scan of a real environment, the discrete objects in that scan will act like fixed background scenery and will not respond to inputs the way that real-world objects would. A key problem, then, is providing a library of high-quality models of 3D objects which can be incorporated into physical and visual simulations to provide the required variety for deep learning.

To address this issue, we present the Google Scanned Objects (GSO) dataset¹, a curated collection of over 1000 3D scanned common household items for use in the Ignition Gazebo [7] and Bullet [8] simulators, as well as other tools that can read the SDF model format. In this letter, we describe our pipeline for object collection and curation, scalable, high-quality 3D scanning, scan quality assurance and publishing. In addition, we present breakdowns of the statistics of the objects in the dataset and the usage of the dataset in published research.

Our contributions include (a) the Google Scanned Objects dataset, (b) the design of our 3D scanning pipeline, (c) the design of our 3D scan curation and publication process, and (d) a review of the impact of this dataset on research.

II. RELATED WORK

Many simulators are available for robotics applications [17], and many learning systems have used simulation to train models to deploy on robots. While early work used static environments in simulators similar to Bullet [8], more recent work has injected 3D objects into the environment to enable training interactive navigation, such as the Interactive Gibson benchmark [18] which uses GSO.

¹Dataset available at https://goo.gle/scanned-objects
3D objects for simulation have been traditionally created through manual 3D modeling [19], object scanning [20], conversion of CAD models [21], or combinations of these techniques [22]. For example, [23] used several techniques to create furniture models for a virtual environment building walkthrough. New neural rendering techniques, such as neural radiance fields [24], have also shown promising results for 3D object scanning.

3D objects have been collected into datasets for many purposes: [16] describes 33 such datasets, though not all remain extant (for example, the Amazon Picking Challenge dataset [25] is no longer readily available). These datasets serve needs such as shape and object retrieval [26], object recognition and computer vision [15], grasping and manipulation [16], and even research into prosthetics and rehabilitation [27]. Regardless of its original purpose, a dataset can be useful in a simulation system if it is currently available and contains objects in watertight meshes, ideally with high visual fidelity. Datasets that provide only scenes [28], objects in scenes [29], [30], [31], [32], [33] or object point clouds [34] are less comparable to our dataset.

Some manufacturers provide 3D models of their products that are usable in simulation; these can sometimes describe articulated 3D geometry, mechanical properties, and other features difficult to capture via scanning. Manufacturers such as Steelcase [35] and Siemens [36] provide CAD shape models, while IKEA [37] provides models which often include rich textures. The IKEA dataset has been used for both shape analysis [38] and as a foundation for further dataset creation, such as the IKEA ASM furniture assembly dataset [39]. ModelBank also provides 1200 high-quality licensable models [40].

3D object datasets suitable for simulation research include those composed of 3D CAD models [41], [26], [42], [43], [44], [45], [46], [47], [48], [49], [15], [16], [50], [51] though many combine types [47]. Datasets most comparable to GSO include freely available, watertight, textured models with high visual fidelity usable in simulation. Scanned datasets include YCB [16] with 77 objects in 5 categories, BigBird [15] with 125 objects, and KIT [14] with 145 objects; however these datasets generally do not include object bases. CAD datasets include PhotoShape [11] with 11,000 CAD chairs, 3D-FUTURE [12] with 16K CAD chairs, and most notably the recently released Amazon Berkeley Objects dataset with 8K CAD objects in almost 100 categories [13]. Table I.1 quantitatively compares these datasets on year of release as well as number of categories, objects, and reference images.

Some 3D datasets have proved useful in supporting grasping tasks. ShapeNet collects 51,300 3D CAD models into 55 categories within the WordNet taxonomy [44]. ShapeNet is designed for vision and does not contain mass or friction, but GraspGAN [52] trained a grasping system using photorealistic renders based on both ShapeNet and procedural models. For GraspGAN, procedural models worked better than ShapeNet’s 3D CAD models (though ShapeNet has a stated goal of adding 3D scanned content). The YCB (Yale-CMU-Berkeley) Dataset [16] is 77 real-world graspable objects and parallel model database of textured meshes to

| Dataset | Year | Purpose | Cat. | Obj. | Img. |
|---------|------|---------|------|------|------|
| PhotoShape [11] | 2020 | 3D Graphics | 1 | 11,000 | 40,927 |
| 3D-Future [12] | 2020 | Shape Retr. | 1 | 16,563 | 20,240 |
| ABO [13] | 2021 | Shape Retr. | 98 | 147,702 | 398,212 |
| KIT [14] | 2012 | Object Reco. | N/A | 145 | 196,188 |
| BigBird [15] | 2014 | Object Reco. | N/A | 125 | 75,000 |
| YCB [16] | 2015 | Object Reco. | 5 | 77 | 92,400 |
| GSO (ours) | 2021 | Simulation | 17 | 1,030 | N/A |

features difficult to capture via scanning. Manufacturers such as Steelcase [35] and Siemens [36] provide CAD shape models, while IKEA [37] provides models which often include rich textures. The IKEA dataset has been used for both shape analysis [38] and as a foundation for further dataset creation, such as the IKEA ASM furniture assembly dataset [39]. ModelBank also provides 1200 high-quality licensable models [40].

3D object datasets suitable for simulation research include those composed of 3D CAD models [41], [26], [42], [43], [44], [45], [11], [12], [13] and 3D scanned models [14], [46], [47], [48], [49], [15], [16], [50], [51] though many combine types [47]. Datasets most comparable to GSO include freely available, watertight, textured models with high visual fidelity usable in simulation. Scanned datasets include YCB [16] with 77 objects in 5 categories, BigBird [15] with 125 objects, and KIT [14] with 145 objects; however these datasets generally do not include object bases. CAD datasets include PhotoShape [11] with 11,000 CAD chairs, 3D-FUTURE [12] with 16K CAD chairs, and most notably the recently released Amazon Berkeley Objects dataset with 8K CAD objects in almost 100 categories [13]. Table I.1 quantitatively compares these datasets on year of release as well as number of categories, objects, and reference images.

Some 3D datasets have proved useful in supporting grasping tasks. ShapeNet collects 51,300 3D CAD models into 55 categories within the WordNet taxonomy [44]. ShapeNet is designed for vision and does not contain mass or friction, but GraspGAN [52] trained a grasping system using photorealistic renders based on both ShapeNet and procedural models. For GraspGAN, procedural models worked better than ShapeNet’s 3D CAD models (though ShapeNet has a stated goal of adding 3D scanned content). The YCB (Yale-CMU-Berkeley) Dataset [16] is 77 real-world graspable objects and parallel model database of textured meshes to

| Dataset | Year | Purpose | Cat. | Obj. | Img. |
|---------|------|---------|------|------|------|
| PhotoShape [11] | 2020 | 3D Graphics | 1 | 11,000 | 40,927 |
| 3D-Future [12] | 2020 | Shape Retr. | 1 | 16,563 | 20,240 |
| ABO [13] | 2021 | Shape Retr. | 98 | 147,702 | 398,212 |
| KIT [14] | 2012 | Object Reco. | N/A | 145 | 196,188 |
| BigBird [15] | 2014 | Object Reco. | N/A | 125 | 75,000 |
| YCB [16] | 2015 | Object Reco. | 5 | 77 | 92,400 |
| GSO (ours) | 2021 | Simulation | 17 | 1,030 | N/A |
facilitate sim-to-real domain adaptation. The KIT object models database [14] uses a setup similar to GSO to capture 145 scanned objects with calibrated images and grasp data for commercial manipulators; however, this scanner does not capture object bases.

Several other 3D object datasets have been created for computer vision. Objectron [32] consists of object-centered videos and augmented reality 3D pose information, used to create machine learning pipelines for 3D object bounding box detection. The Berkeley 3-D Object Dataset (B3DO) [30] collects over 800 3D scanned models in over 50 categories, scanned via the Kinect 3D sensor and annotated via the Amazon Mechanical Turk (AMT) [53]. ScanNet is a collection of over 1500 3D scanned room-sized scenes collected via a similar Kinect-AMT pipeline, augmented with dense 3D reconstructions, and PartNet [45] consists of over 25K 3D CAD models broken down into almost 575K constituent parts; both have been used for semantic segmentation. ObjectNet3D [31] consists of 44K 3D shapes in 100 categories, aligned with 90K images annotated with 3D poses, and has been used for object detection. BigBIRD, the (Big) Berkeley Instance Recognition Dataset [15] contains 125 3D meshes, each with 600 calibrated RGB-D images and segmented point clouds to facilitate object recognition, captured with a setup similar to GSO; like KIT, however, these meshes lack views of object bases. 3D-Future [12] uses 16K CAD textured chairs to generate 20K realistic images corresponding to 5K scenes, designed to enable precise comparison of images to models for benchmarking. The Amazon Berkeley Objects [13] dataset contains 8K high-quality textured CAD objects in 98 categories, along with reference images; this dataset, along with GSO, was used to benchmark a differentiable stereopsis shape reconstruction approach [54].

III. CREATION OF THE DATASET

The GSO dataset began as an effort by Google’s Cloud Robotics initiative in 2011 to enable robots to recognize and grasp objects in their environments using high-fidelity 3D models of common household objects. However, 3D models have many uses beyond object recognition and robotic grasping, including scene construction for physical simulations and 3D object visualization for end-user applications. Therefore, Google Research initiated a project to bring 3D experiences to Google at scale, collecting a large number of 3D scans of household objects at a cost less than that of traditional commercial-grade product photography.

This was an end-to-end effort, including object acquisition, novel 3D scanning hardware, efficient 3D scanning software, fast 3D rendering for QA, web and mobile viewers, and human-computer interaction studies to create effective experiences for interacting with 3D objects. Following data collection, we then built a pipeline to make this data available in a variety of formats. In the following sections, we discuss the 3D scanning innovations necessary to collect scans of household items, followed by the pipeline necessary to make them available to simulation systems.

A. 3D SCANNING PIPELINE

Even when focusing only on household objects, 3D scanning presents distinctive challenges, including efficient physical scanning setups, target lighting, camera reliability, scanner performance, color matching, texture rendering, and dealing with optically uncooperative materials, such as near-white, shiny, or transparent surfaces. Dedicated 3D scanning hardware existed at the time but was labor-intensive and not cost-effective. To scan at scale, we needed something that functioned less like a scientific instrument and more like a microwave in its usability and its reliability.

Therefore, we designed our own dedicated scanning hardware and software (Fig. 2), capable of scanning objects in 10 minutes and producing high-resolution models with 2M triangles. The lighting-controlled physical enclosure (Fig. 2(b)) captures 3D geometry using structured light scanning with two machine vision cameras and a projector, and captures textures with product-friendly lighting with a separate DSLR high-resolution camera. By the end of our project’s first year, we were capturing over 400 scans per week, and over the course of the project, we captured 100K 360° photo swivels and 10K fully-3D scans of unique objects.
Our scanning software pipeline treats 3D scanning as a calibration problem (Fig. 2(a)), using libraries shared with our 360° swivel scanning pipeline. To stitch multiple 2D images into a smooth animated rotation, the 360° swivel pipeline scans 3D reference calibration patterns as if they are ordinary objects (Fig. 3(a)), enabling camera geometry to be calibrated with sub-pixel accuracy (Fig. 5(c)).

The 3D pipeline extends this process using a projector to generate over 200 patterns (Fig. 3(b)). Building on an approach developed by Jens Gühring [55], we use a stereo pair of machine vision cameras to capture gray codes and phase patterns, projected in a range of intensities, to build up HDR (High Dynamic Range) single-color pattern frames, augmented by a centrally mounted DSLR which captures an aligned view of the color textures (Fig. 3(d)).

Gray codes [56], an indexing method robust to bit errors, are used to identify projector pixels in camera images, but are not sufficient to guarantee sub-pixel accuracy needed for texture mapping. Phase patterns (X and Y stripes) identify exact projector lines to sub-pixel accuracy, but are not globally unique without the gray codes for correspondence.

We use camera calibration and image reconstruction to build solid color HDR images of the various patterns. These images, along with the original raw frames, form the basis for parallel gray code indexing and stripe normalization pipelines. A convolution operation enables us to find the true centers of the stripes, producing horizontal and vertical code frames. To speed up the combination of horizontal and vertical frames into joint x, y coordinates, we took advantage of the fact that the original projected frames have a known width and height in pixels. Therefore, any given image can be viewed as having a known integer range in one dimension and a to-be-determined floating point dimension in the other. This permits a fast \( O(n^2 \log(n)) \) active interval sweep line algorithm which requires effectively logarithmic per-pixel work for normal image sizes.

This algorithm produces \( p\text{coord} \) (projector coordinate) frames with sub-pixel scale resolution, each representing a unique position on the projector. Combined with a camera’s location, each pcoord generates a camera ray. Therefore, 2 or more pcoord pixels define a 3D point; the nearest point to the 3D “intersection” has a fast closed-form solution for 2 rays. This generates a depth map, which we use to generate a mesh by looking for valid quad-pixel neighborhoods, outputting two mesh triangles per “quad” found. The result is a “raw mesh” from one camera view (Fig. 1 upper right).

The computer-controlled turntable enables the scanner to repeat the mesh generation process at precise angles around an object, resulting in a set of raw meshes already in close alignment. We use the Ceres solver [57] to implement a fast ICP solver on all pairs in parallel, focusing on not breaking the good initial convergence. This process optimizes 100K to 400K constraints per iteration and runs in less than a minute even on the relatively large scans produced by our scanner. This results in an aligned mesh (Fig. 1 lower left), which the local scanner software then decimates, assigns initial colors, and uploads to the cloud for further processing.

This pipeline enables the scanner to produce nearly complete raw scans directly on the hardware as fast as it can physically scan objects, allowing our remaining pipeline to focus on more intensive issues, such as reconstruction of the inside of concave objects or human quality assurance.

B. Simulation Model Conversion

These original scanned models used protocol buffer meta-data [58], very high resolution visuals, and formats unsuitable for simulation. Some physical properties, such as mass, were captured for many objects, but surface properties, such as friction, were not represented in the metadata.

To enable these scanned models to be used in simulation systems, each was passed through a pipeline which performed the following steps:

1) Filter invalid objects. The dataset needed to be cleaned prior to processing. Duplicate items were removed, along with test objects, calibration objects, and a few hand-modeled objects. Models with invalid or missing meshes, textures or metadata were also excluded. The remaining valid models went through a manual QA process prior to inclusion (Fig. 4).

2) Assign object names. Human readable names were constructed automatically from text descriptions of the objects, followed by manual QA to fix more mistakes.

3) Validate object meshes. Simulation requires a closed manifold mesh, a physically realizable volume described by a surface of polygonal facets with an interior, exterior, no self-intersections, no holes, and non-zero thickness everywhere. Furthermore, all geometry must be represented directly in the mesh, since bump maps, offset maps, and normal maps are not generally used in simulations. Almost all objects in our dataset had closed manifold meshes, and the few invalid meshes were excluded.

4) Calculate physical properties. The original pipeline focused on visualization, not simulation, and did not capture properties like object density or friction coefficients. Starting with a clean mesh, we calculated the bounding box and volume. While some objects had recorded mass, for the rest we estimated mass from volume; similarly, we calculated center of mass and moments of inertia assuming objects were solid and homogeneous. When mass was not provided, we estimated a default constant density of 0.1 g/ml, which is a reasonable estimate for a hollow plastic object. We left friction values unspecified, which should use the simulator default values. While these estimates limit the use of these models for tasks which require accurate physical parameters, they are sufficient to enable their use in many simulation platforms.

5) Construct collision volumes. We explored several techniques to generate collision volumes for our models, including VHACD (Volumetric Hierarchical Approximate Convex Decomposition) [59] to generate convex collision volumes for use in simulation tools that require a convex decomposition. However, we
identified different use cases which would require more or less complex collision meshes, and ultimately chose to use the visual mesh directly as the collision volumes.

6) **Reduce model size.** Our scanning pipeline produces meshes and with high visual fidelity but these needed to be simplified to reasonable resolutions for rendering using edge collapse algorithms with quadratic error metrics [60]. The smallest of the simplified meshes provided by the original scanning pipeline down-sampled the original million-vertex meshes to tens of thousands of vertices and were generally suitable for physical simulations. We used Blender [61] to further down-sample meshes and textures for internal projects.

7) **Create SDF models.** These scans did not include linkage or joint information, so a simple model in SDF format was created for each object with one link to one OBJ visual mesh and one collision mesh.

8) **Create thumbnail images.** We mined thumbnail images from the preview images of the original dataset.

9) **Package the model.** For use in simulators or in public interfaces with searchable metadata, each model had to be self-contained with a metadata file and all related assets. Associated file inputs such as visual meshes, collision meshes, and textures were organized by a relative path from the model directory.

The output of this pipeline was a simulation model in an appropriate format with a name, mass, friction, inertia, and collision information, along with searchable metadata in a public interface compatible with Ignition Fuel.

C. Ignition Fuel

To host the models, we chose Ignition Fuel, an open-source hosting environment that implements an evolution of Gazebo’s model hosting strategy. Historically, Gazebo used models hosted on version-controlled repositories such as Git [62]. This requires minimal development and maintenance, but does not scale to large numbers of models. Code repositories are not designed for large binary datasets or discoverability, making storing and finding models cumbersome.

To address these issues, Open Robotics developed Ignition Fuel, which encapsulates each model within its own Git repository, hosted on a server with a scalable filesystem. This approach provides the benefits of model version control while limiting repository size. A front-end web application provides a user-friendly interface to the server with features including account management, model upload and download, 3D visualization of a model, and search. Each model released to Ignition Fuel includes an OBJ visual mesh, MTL material file, PNG diffuse texture map, SDF model description, text protobuf metadata, and thumbnail images.

To host GSO, Open Robotics collaborated with Google to expand Ignition Fuel’s metadata, to scale to larger sets of models, and to improve model discoverability and accessibility through ElasticSearch. Ignition Fuel model metadata includes model name, description, and authors; for GSO, we expanded this to include SKUs, manufacturer part numbers, brand, tool-chain compatibility, copyright and license information, tags, version, annotations and categories. This additional metadata supports improved search and vetting of models, and is captured in a protobuf text file alongside other model assets.

In addition, Ignition Fuel Tools is a command line interface (CLI) which communicates with the Ignition Fuel server through its REST API, presenting a more user-friendly interface than general purpose tools such as curl. For GSO, we expanded the CLI to support bulk upload and downloads via collections - a logical grouping of models such as the GSO dataset.

![Fine surface details.](image1)

![Complex topology.](image2)

Fig. 5. Because our scanner uses calibrated projected patterns rather than raw depth data, the GSO process captures fine surface details (a) and complex topology (b) in scanned models.

IV. PROPERTIES OF THE DATASET

A. Dataset Composition

The GSO dataset contains 1030 scanned objects and associated metadata, totalling 13Gb, licensed under the CC-BY 4.0 License [63]. These models are a subset of the larger collection scanned by Google that pass our curation pipeline. Table III.1 breaks down the model categories in the dataset.

Visual meshes are in Wavefront OBJ format [64] averaging 1.4MB (min: 0.1MB, max: 11.1MB) per model. The accompanying diffuse texture maps are in PNG format, and average 11.2MB (min: 6.5MB, max: 23.5MB) per texture.

| Category       | Models | Vertices: avg. ± σ | min / max         |
|----------------|--------|---------------------|-------------------|
| Action Figures | 15     | 10284 ± 2643        | 5415 / 15621      |
| Bag            | 28     | 9450 ± 3951         | 5511 / 21922      |
| Board Games    | 17     | 2137 ± 2164         | 513 / 9861        |
| Bottles/Cans/Cups | 53  | 4984 ± 1860        | 1035 / 13638      |
| Camera         | 1      | 4785                |                   |
| Car Seat       | 1      | 16416               |                   |
| Consumer Goods | 248    | 3465 ± 3874         | 577 / 42890       |
| Hat            | 2      | 19313 ± 7915        | 11398 / 27229     |
| Headphones     | 4      | 5733 ± 5314         | 1995 / 14819      |
| Keyboard       | 4      | 10682 ± 3709        | 4280 / 13251      |
| Block toys     | 10     | 13477 ± 13072       | 942 / 41036       |
| Media Cases    | 21     | 2557 ± 2033         | 1166 / 10293      |
| Mouse          | 4      | 13386 ± 2888        | 9686 / 16620      |
| Shoe           | 254    | 13091 ± 4931        | 3558 / 32555      |
| Stuffed Toys   | 3      | 31062 ± 12617       | 18237 / 48220     |
| Other toys     | 147    | 7684 ± 4829         | 1419 / 36138      |
| Uncategorized  | 216    | 8216 ± 8669         | 828 / 50561       |
projects generated synthetic data for deep vision systems, while the IBRNet [70] and Neural Ray [71] projects generated training data for deep rendering systems.

The iGibson Challenge 2021 [73] and Hierarchical Mechanical Search (HMS) [72] projects used these objects to populate scenes for robotic navigation training and benchmarking tasks. Across many of these projects, the addition of controllable synthetic data to training pipelines either improved the overall performance of the system or improved the visual realism of the simulated environments.

Notably, three projects - Grasp detection via Implicit Geometry and Affordance (GIGA) [74], Lateral Access X-RAY [75], and Object-Object Affordance Learning [76] - used the GSO dataset for the purpose that inspired it: generating synthetic data to train robotic grasping and manipulation tasks. These pipelines either relied on GSO to create their scenes, or showed improvements by augmenting real data.

An internal version of the dataset with cleaned-up transparency and specular materials was used by Action Image Representation [65], RetinaGAN [66], and RL-CycleGAN [67]. The visual realism of the models, plus randomizing visual and physical properties, facilitated sim-to-real transfer learning for real-world tasks.

Finally, because the dataset is scanned from real objects, KeypointDeformer [77] and Differentiable Stereopsis (DS) [54] were able to use the GSO dataset as a validation set. DS used 50 categories in one evaluation, alongside two other evals using [13], [50].

VI. CONCLUSIONS

This letter presented Google Scanned Objects, a diverse collection of high-quality 3D scans of household objects released in Ignition Gazebo format. These objects have already proven useful in ten robotics and simulation projects. We hope that the Google Scanned Objects dataset will be used by more robotics and simulation researchers in the future, and that the example set by this this dataset will inspire other owners of 3D model repositories to make them available for researchers everywhere.

ACKNOWLEDGMENT

The authors thank the Google Scanned Objects team, including Peter Anderson-Sprecher, J.J. Blumenkranz, James Bruce, Ken Conley, Katie Dektar, Charles DuHadway, Anthony Francis, Chaitanya Gharpure, Topraj Gurung, Kristy Headley, Ryan Hickman, John Isidoro, Sumit Jain, Greg Kline, Mach Kobayashi, Kai Kohlhoff, James Kuffner, Thor Lewis, Mike Licitra, Lexi Martin, Julian (Mac) Mason, Rus Maxham, Pascal Muetschard, Kannan Pushapathy, Barbara Petit, Arshan Poursohi, Jared Russell, Matt Seegmiller, John Sheu, Joe Taylor, and Josh Weaver. Special thanks go to James Bruce for the scanning pipeline design and Pascal Muetschard for maintaining the database of object models.

B. Strengths of the Dataset

Our automated pipeline enables quick generation of large numbers of models without manual processing. Because these models are scanned rather than modeled by hand, they are realistic, not idealized, reducing the difficulty of transferring learning from simulation to real [65], [66], [67].

Our scanner’s glass platform enables scanning models from all sides including the base, unlike similar scanners with opaque platforms [15], [14], [34]. Similarly, models extracted from an environment [30], [28] will generally lack occluded regions like the base, which must be stitched.

Because our scanner reconstructs surface shape from projected patterns rather than from depth camera data, the resulting mesh has high fidelity. Smooth surfaces are smooth and silhouette edges are accurate (fig. 3) compared with meshes derived from RGB-D data which can appear mottled and irregular, particularly at silhouettes.

C. Limitations of the Dataset

Our scanner’s capture area cannot accommodate objects larger than a breadbox (roughly 50 cm), so the dataset does not include larger objects, such as the chairs, automobiles, or airplanes, found in some other datasets [44], [40]. Similarly, scan resolution is limited, so very small objects cannot be modeled with reasonable fidelity. Furthermore, generated textures are diffuse: highly specular or transparent objects cannot be represented and can yield poor results.

V. USAGE OF THE DATASET

Hosting GSO on Ignition Fuel has made it possible for people inside and outside the robotics community to find, download and use the dataset in simulation and their research projects. To our knowledge, the GSO dataset has been used in a dozen papers across ten projects spanning computer vision, computer graphics, robot manipulation, robot navigation, and 3D shape processing (Table IV.1).

Most projects used the GSO dataset to provide synthetic training data for learning algorithms. The NVIDIA Scene Imaging Interface (NViSII) [68] and Faster R-CNN [69] projects generated synthetic data for deep vision systems, while the IBRNet [70] and Neural Ray [71] projects generated training data for deep rendering systems.

| Project         | Usage       | Renderer    | Simulator |
|-----------------|-------------|-------------|-----------|
| Vision          | Training    | NViSII      | PyBullet  |
| NViSII [68]     | Training    | NViSII      | PyBullet  |
| Faster R-CNN [69]| Training    | OpenGL      | N/A       |
| Graphics        | Training    | IBRNet      | N/A       |
| IBRNet [70]     | Validation  | DS          | N/A       |
| NeuRay [71]     | Validation  | Trimesh     | N/A       |
| Diff. Stereo. [54]| Training    | iGibson     | iGibson   |
| Navigation      | Benchmark   | iGibson     | iGibson   |
| iGibson [72]    | Training    | iGibson     | iGibson   |
| Manipulation    | Training    | Pyrender    | Trimesh   |
| O2O-Afford [76] | Training    | N/A         | SAPIEN    |
| 3D Shape Processing | Validation | N/A         | N/A       |
| Keypoint Deformer [77] | Validation | N/A         | N/A       |
