3DP3: 3D Scene Perception via Probabilistic Programming

Anonymous Author(s)
Affiliation
Address
email

Abstract

Humans learn to parse scenes more robustly than deep learning vision systems, generalizing across large variations in viewpoint, occlusion, lighting, and clutter. In this work, we present an probabilistic programming architecture for parsing scenes using object models learned from 5 or fewer views. Our architecture called 3DP3 uses voxelized models of object shape, a generative 3D scene graph prior that compositionally represents scenes using shapes and contacts between them, and a likelihood model based on real-time graphics. Our inference algorithm accurately parses scenes, using fast bottom-up pose proposals and novel involutive MCMC updates on scene graph structure. We show this approach allows for rapid object learning, and scene parsing that leverages physical constraints. Our quantitative results demonstrate better generalization to scenes with novel viewpoints, contact, and occlusions than is achieved by deep learning baselines, and accurate parsing of real scenes using neural bottom-up proposals.

1 Introduction

A striking feature of human visual intelligence is our ability to learn representations of novel objects from just a few views and robustly parse scenes containing those objects, generalizing across large variations in viewpoint, occlusion, lighting, and clutter. Human scene parsing accurately accounts for physical knowledge to parse scenes that contain complex arrangements of objects. How might we develop computational vision systems that can do the same?

This paper presents a generative probabilistic programming-based architecture for 3D scene perception, called 3DP3. 3DP3 models objects, scenes, and images via generative probabilistic programs. For example, 3D objects are modeled via variationally-trained voxel generators. Scenes are modeled compositionally, via generative 3D scene graphs, that recursively decompose 6DOF poses in terms of object contact relationships. Depth images are modeled via rendering, and intractable depth image likelihoods arising from shape uncertainty are replaced via pseudo-marginal approximations. Inference in 3DP3 is implemented in the Gen [40] probabilistic programming system. Gen makes it practical to integrate top-down model-based Monte Carlo updates over scene graph structure, camera pose, and object shape with bottom-up proposals from deep learning and geometric computer vision.

We demonstrate that our architecture can: (1) rapidly learn object representations from few depth images and then parse scenes containing those objects, (2) generalize to scenes containing contacts, occlusion, and novel viewpoints better than deep learning baselines, and (3) leverage neural detectors as bottom-up proposals in our inference algorithm in order to accurately parse real scenes.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

2 Related Work

Bayesian inverse graphics. A long line of work has explored the idea of computer vision as the Bayesian inverse problem to computer graphics [1, 2, 3, 4]. Analysis-by-synthesis approaches leverage a forward model, often in the form of a generative model, and do inference over the latent variables given observed images. This has been applied to tasks like character recognition, CAPTCHAs, lane detection, object detection, and human pose estimation, among others [5, 6, 7, 8, 9, 10]. Our work is the first, to our knowledge, to apply the analysis-by-synthesis approach to 3D multi-object scenes using structured priors over 3D scenes that are aware of shape and contacts.

Probabilistic programs for vision. Our approach is also related to Bayesian inverse graphics approaches based on probabilistic programming [5, 11], that use inference in generative programs to invert generative models based on components of graphics pipelines. The key contributions of our work compared to these prior works are (i) learnable object models (unlike manually-specified models [5, 11]), (ii) a generic scene graph prior for modeling contact relationships that improves pose estimation accuracy, (iii) a novel involutive MCMC [12] kernel for inferring scene graph structure, and (iv) a pseudomarginal approach for handling uncertainty about object shape during scene parsing.

6D object pose estimation. There are numerous prior works proposing approaches to 6D object pose estimation. The earliest work on detecting objects from depth data focused on template matching of CAD models [13, 14] and voting schemes, most popular of which were Point-Pair features [15, 16, 17]. Recent work has focused on deep learning approaches [18, 19] the most successful of which take depth image inputs [20, 21]. Our method is also related to prior work that models scene structure and contact relationships [22, 23, 24] and probabilistic methods for pose estimation [25, 26, 27, 28]. In our approach, we take a top-down approach to 6D object pose estimation. Our goal is not to propose a replacement for bottom-up methods, but instead to build a hybrid approach that draws on key strengths of the bottom-up and top-down approaches.

Object modeling. Structure-from-motion and bundle adjustment infer object models, often in the form of a mesh, from many viewpoints of an object [29, 30, 31, 32]. More recently, neural radiance fields (NeRFs) have gained popularity, as they learn photorealistic models of objects and can synthesize images from new perspectives [33, 34]. The above methods often require many viewpoints; in contrast, our approach can be applied to 5 of fewer images. Our approach also learns probabilistic representations of shape, capturing uncertainty, unlike most other approaches. Our voxel shape approach can model complex objects that lack a concise parametric representation and allow us to successfully parses scenes with real objects.

Scene graphs. We use a scene graph representation that is closely related to hierarchical scene representations used in computer graphics [35] for rendering images of scenes, and in robotics [36]. Unlike in graphics, we address the *inverse problem* of inferring scene graphs from observed images of scenes. Prior work [37] from robotics on probabilistic inference over the kinematic tree structure of a single articulated object from a time series is most closely related to our approach, but solves a different problem, either requires markers to be placed on scene elements or only works for rectangular object parts, is not designed to handle severely occluded or out-of-view scene elements, and does not use asymptotically exact Bayesian inference. The term 'scene graph' has also been used to refer to various other hierarchical representations of scenes that are less related to ours [38, 23, 39].

3 Parsing point clouds into scene graphs via probabilistic inference

This section describes our latent scene representation based on scene graphs, a generative model based on this representation, and an algorithm that jointly infers scene graph structure and pose parameters from a depth image. We implemented the generative model as a probabilistic program in the Gen probabilistic programming system [40], and implemented our inference algorithm using components from Gen's flexible inference library. Before describing the generative model and algorithm, we describe two representations that are central to our approach.

Voxel shape models We model objects via a coarse representation of their 3D shape on voxel occupancy grids. These grids are defined with fixed dimensions $h, w, l \in \mathbb{N}$ and grid resolution $s \in \mathbb{R}^+$. An object's *shape* is defined by a binary assignment $\mathbf{O}$ of occupancy states to cells in the 3D grid i.e. $O_{ijk} \in \{0, 1\}$ for all $(i, j, k) \in [h] \times [w] \times [l]$, where 1 indicates the voxel cell is

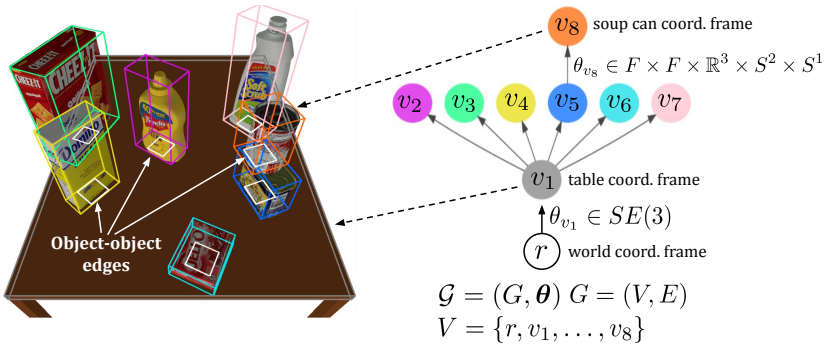


Figure 1: Scene graphs encode a tree of coordinate frames representing entities in a scene and their geometric relationships (e.g. flush contact between faces of two objects). We show a synthetic scene with object-to-object contacts indicated with white squares and the corresponding scene graph.

occupied and 0 indicates it is free. This assignment coarsely captures the 3D volume occupied by the object. We also fit a 3D axis-aligned bounding box to the voxelized shape model and define the faces $F = \{Top, Bottom, Left, Right, Front, Back\}$ of that box to be the object's contact planes. Section 4 describes one way of learning these voxelized shape models from data.

Scene graphs We model the geometric state of a scene as a *scene graph* $\mathcal{G}$ (Figure 1), which is a tuple $\mathcal{G} = (G, \boldsymbol{\theta})$ where $G = (V, E)$ is a directed rooted tree and $\boldsymbol{\theta}$ are parameters of this tree. The vertices $V := \{r, v_1, \ldots, v_N\}$ represent 3D coordinate frames, with r representing the world coordinate frame. Edges $(u, v) \in E$ indicate that reference frame v is parametrized relative to frame u, with parameters θ_v. The 6DoF pose $\mathbf{x}_v \in SE(3)$ of reference frame v relative to reference frame u with pose $\mathbf{x}_u \in SE(3)$ is given by a function $\Delta\mathbf{x}_v$, where $\Delta\mathbf{x}_v(\theta_v) \in SE(3)$ so that $\mathbf{x}_v := \mathbf{x}_u \cdot \Delta\mathbf{x}_v(\theta_v)$. Here, $\cdot$ is the $SE(3)$ group operation, and the world coordinate frame is defined as the identity element ($\mathbf{x}_r := I$). While the vertices of scene graphs can represent arbitrary coordinate frames in a scene (e.g. the coordinate frames of articulated joints, object poses, etc.), in the remainder of this paper we assume that each non-root vertex corresponds to rigid object with a finite set of contact faces, whose poses relative to the object's coordinate frame are known. For vertices v that are children of the root vertex r, $\theta_v \in SE(3)$ defines the absolute 6DoF pose of the corresponding object ($\Delta\mathbf{x}_v(\theta_v) = \theta_v$). For vertices v that are children of a non-root vertex u, the parameters take the form $\theta_v = (f_v, f'_v, a_v, b_v, z_v, \phi_v)$ and represent a contact relationship between the two objects. $f_v \in F$ and $f'_v \in F$ are faces of the parent and child objects, respectively, that are contacting. $(a_v, b_v) \in \mathbb{R}^2$ is the in-plane offset of the origin of face f_v of object v from the origin of face f'_v of object u. $z_v \in \mathbb{R}$ is the perpendicular distance of the origin of face f_v from face f'_v of object u. $\phi_v \in S^2 \times S^1$ represents the deviation the normal vectors of the two faces from anti-parallel (in S^2) and a relative in-plane rotation of the two faces (in S^1). The relative pose $\Delta\mathbf{x}_v(\theta_v)$ of v with respect to u is the composition (in $SE(3)$) of three relative poses: (i) v with respect to its face f_v, (ii) v's face f_v with respect to u's face f'_v, and (iii) u's face f'_v with respect to u.

3.1 Generative model

This section describes a class of geometric and probabilistic generative models for multi-object scenes (Figure 2). We leverage the scene graph representation to encode useful prior knowledge about the regularities in real-world scenes: The poses of objects in the real world are not independent but rather are often coupled (e.g. consider objects stacked on a table). Our models also leverage prior probability distributions $p(\mathbf{O}_c)$ on the voxel shape model $\mathbf{O}_c$ for each of M object classes[1] ($c \in \{1, \ldots, M\}$). Scene graphs can flexibly compose these object shape models to parsimoniously explain scenes with previously unseen combinations and arrangements of previously seen objects.

Prior on scene graphs We now describe a fairly general class of priors on scene graphs (illustrated in Figure 3). First, we sample the number of objects N in the scene from a prior $p(N)$. Possibilities for $p(N)$ include (i) an a-priori known number of objects N_0 (used in the experiments in Section 5),

[1] Section 4 describes one method for obtaining these prior distributions.

(ii) $\text{Binomial}(N_0, p_{\text{present}})$, which is induced by a prior belief that each of N_0 objects is present with probability p_{present} (used in experiments described in the supplement), and (iii) $\text{Poisson}(\lambda)$, which places no a-priori upper bound on the number of objects. Next, we sample the classes $c_i \in \{1, \ldots, M\}$ of all objects from an exchangeable prior $p(c_1, \ldots, c_N|N)$. This includes as a special case distributions where all classes are represented at most once among the objects ($\sum_{i=1}^{N} \mathbf{1}[c_i = c] \leq 1$), which is the case in our experiments. Next, we sample the scene graph structure G from $p(G|N)$. We experiment with two priors $p(G|N)$: (i) a uniform distribution on the set of $(N+1)^{N-1}$ directed trees that are rooted at a vertex r, and (ii) $\delta_{G_0}(G)$, where G_0 is a graph on $N+1$ vertices where $(r, v) \in E$ for all $v \in V \setminus \{r\}$ so that each object vertex has an independent 6DoF pose. For objects whose parent is r (the world coordinate frame), we sample the pose $\theta_v \sim p_{\text{unif}}$, which samples the translation component uniformly from a cuboid scene extent, and the orientation uniformly over $SO(3)$. For objects whose parent is another object u, we sample $(f_v, f'_v) \sim \text{Uniform}(F^2)$, $(a_v, b_v) \sim \text{Uniform}([-50\text{cm}, 50\text{cm}]^2)$, $z_v \sim \text{N}(0, 1\text{cm})$, the S^2 component of ϕ_v from a von-Mises Fisher distribution concentrated ($\kappa = 250$) on anti-parallel face normals, and the S^1 component from $\text{Uniform}(S^1)$. The resulting distribution is denoted $p_{\text{cont}}(\theta_v)$. In summary, the prior is:

$$p(\mathbf{O}_{1:M}, N, \mathbf{c}, G, \theta) = \left(\prod_{i=1}^{M} p(\mathbf{O}_i)\right) \frac{p(N)}{(N+1)^{N-1}} \, p(\mathbf{c}|N) \prod_{\substack{v \in V: \\ (r,v) \in E}} p_{\text{unif}}(\theta_v) \prod_{\substack{(u,v) \in E: \\ u \neq r}} \frac{1}{|F|^2} p_{\text{cont}}(\theta_v) \tag{1}$$

Note that the relative pose function $\Delta\mathbf{x}_{v_j}$ for $(v_i, v_j) \in E$ depends on object models $\mathbf{O}_{c_i}$ and $\mathbf{O}_{c_j}$. This prior induces a prior probability distribution on the 6DoF pose ($\mathbf{x}_v$) of each object relative to the world coordinate frame that favors scenes that can be parsimoniously described via either fewer objects, or fewer objects with independently-sampled 6DoF poses.

Likelihood model based on depth rendering To model point cloud observations $\mathbf{Y} \in \mathbb{R}^{K \times 3}$ with K points, we use a likelihood model based on rendering a synthetic depth image of the scene graph. Specifically, given the object models $\mathbf{O}_c$ for each $c \in \{1, \ldots, M\}$, the object classes $\mathbf{c}$, the scene graph $\mathcal{G}$, and the camera intrinsic and extrinsic parameters relative to the world frame, we (i) compute meshes for each object model, (ii) compute the 6DoF poses ($\mathbf{x}_v$) of objects with respect to the world frame by walking the scene graph $\mathcal{G}$, and (iii) render a depth image $\tilde{\mathbf{I}}$ of the sampled scene via an OpenGL depth buffer. We then deterministically subsample the point cloud obtained by unprojecting the rendered depth image to obtain a point cloud $\tilde{\mathbf{Y}}$ with $\tilde{K}$ points. Finally, we generate an observed point cloud $\mathbf{Y}$ with K points from $\tilde{\mathbf{Y}}$ by drawing each point from a mixture:

$$p(\mathbf{Y}|\tilde{\mathbf{Y}}) := \prod_{i=1}^{K} \left(C + \frac{1-C}{\tilde{K}} \sum_{j=1}^{\tilde{K}} p(\mathbf{Y}_i|\tilde{\mathbf{Y}}_j) \right) \text{ for } p(\mathbf{Y}_i|\tilde{\mathbf{Y}}_j) := \begin{cases} \frac{1}{\frac{4}{3}\pi r^3} & \text{if } ||\mathbf{Y}_i - \tilde{\mathbf{Y}}_j||_2 \leq r \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

for some $0 < C < 1$ and some $r > 0$. The components of this mixture are uniform distributions over the balls of radius r centered at each point in $\tilde{\mathbf{Y}}$ (with weights $(1-C)/\tilde{K}$) and a uniform distribution over all occupiable space (weight C).[2]

3.2 Probabilistic inference algorithm

Given an observed point cloud $\mathbf{Y}$, our aim is to sample approximately from the conditional distribution $p(N, \mathbf{c}, G, \boldsymbol{\theta}|\mathbf{Y})$ over the number of objects (N), the scene graph structure (G) and parameters ($\boldsymbol{\theta}$) and the object classes ($\mathbf{c}$), which amounts to inferring the individual object classes, their poses, and probable contact relationships between them. The remainder of this section describes an inference algorithm for the special case when N and $\mathbf{c}$ are known a-priori. Note that even this restricted case is a challenging inference problem because it involves model selection over a set of $(N+1)^{N-1}$ possible graph structures G, where each G induces a different prior distribution on a $6N$ dimensional space of pose parameters $\boldsymbol{\theta}$. The supplement describes an inference algorithm for an alternative prior distribution where N (and by extension $\mathbf{c}$) are a-priori unknown.

[2] By using a distribution that is uniform over a small, spherical region rather than a Gaussian distribution, we avoid (via k-d trees) computing pairwise distances between all points in $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$, resulting in $\approx 10\times$ speedup.

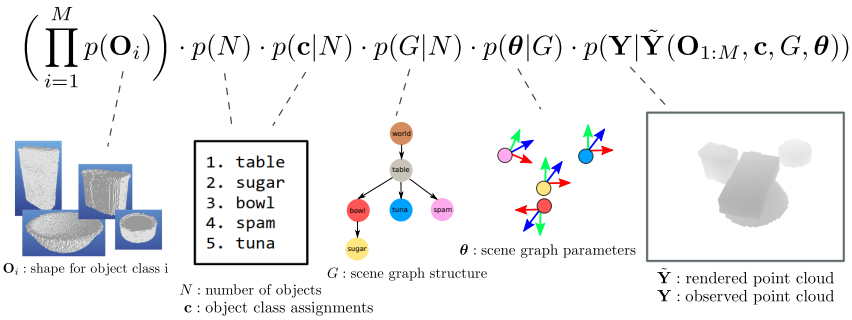


Figure 2: Components of the generative model used for scene parsing (Section 3.1).

```
model SCENE-PARSING-GENERATIVE-MODEL(M)
    parameters: M (= num object types)
    N ~ NUMBER-OF-OBJECTS-PRIOR()
    c ~ OBJECT-CLASS-ASSIGNMENTS-PRIOR(N, M)
    for i ∈ {1, ..., N} do
        O_i ~ VOXEL-SHAPE-PRIOR(c_i)
    end for
    G := (V, E) ~ GRAPH-STRUCTURE-PRIOR(N)
    for v ∈ TOPO-SORT(V, E) where (u, v) ∈ E do
        if u = r then
            θ_v ~ UNIFORM-POSE-PRIOR()
        else
            θ_v ~ CONTACT-PRIOR(G, v, O, c)
        end if
        x_v ← COMPUTE-ABSOLUTE-POSE(x_u, θ_v, O, c)
    end for
    Ỹ ← SUBSAMPLE(DEPTH-RENDER(x, O_{i:i∈c}))
    Y ~ NOISE-MODEL(Ỹ)
end model
```

```
model OBJECT-LEARNING-GENERATIVE-MODEL(M, x_{1:T})
    parameters: M, map; x_{1:T}, camera poses
    O ~ VOXEL-OCCUPANCY-SHAPE-PRIOR()
    for t ∈ [1, T] do
        Ĩ_t ← DEPTH-RENDER(x_t, O, M)
        I_t ~ NOISE-MODEL(Ĩ_t)
    end for
end model

model SLAM-GENERATIVE-MODEL()
    M ~ MAP-PRIOR()
    for t ∈ [1, T] do
        x_t ~ POSE-PRIOR(x_{t-1})
        Ĩ_t ← DEPTH-RENDER(x_t, M)
        I_t ~ NOISE-MODEL(Ĩ_t)
    end for
end model
```

Figure 3: Pseudocode for probabilistic programs defining generative models for scene parsing (left, see Sec. 3), learning object shape (top right, see Sec. 4), and SLAM (bottom right, see supplement).

The algorithm has two stages: First, we obtain initial estimates $\hat{\mathbf{x}}_v \in SE(3)$ of 6DoF object poses $\mathbf{x}_v$ for all object vertices v via maximum-a-posteriori (MAP) inference in a restricted variant of the generative model with graph structure G fixed to G_0 (so there are no edges between object vertices):

$$(\hat{\mathbf{x}}_{v_1}, \ldots, \hat{\mathbf{x}}_{v_N}) \approx \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|N, \mathbf{c}, G_0, \mathbf{Y}) \tag{3}$$

We approximate the MAP inference via Metropolis-Hastings (MH) moves that combine iterative closest point (ICP) and random-walk based proposals (see supplement for details), as well as, optionally, bottom-up neural pose estimators (see Section 5.2). In the second stage, we use the estimated poses to initialize an MCMC algorithm targeting $p(G, \boldsymbol{\theta}|N, \mathbf{c}, \mathbf{Y})$ with state $G \leftarrow G_0$ and $\theta_v \leftarrow \hat{\mathbf{x}}_v$ for each $v \in V \setminus \{r\}$. The Markov chain is a cycle of a structure-changing involutive MCMC [12] kernel that makes small local edits to the graph structure G and also transforms the parameters $\boldsymbol{\theta}$ (to explore the space of graph structures) and a mixture of random-walk MH kernels on the parameters θ_v (to explore the neighborhood of posterior modes, or to correct poor initialization).

Involutive MCMC kernel on scene graph structure and parameters The involutive MCMC kernel takes a graph structure G and proposes a new graph structure G' (Figure 4) by: (i) randomly sampling a node $v \in V \setminus \{r\}$ to 'sever' from the tree, (ii) randomly choosing a node $u \in V \setminus \{v\}$ that is not a descendant of the severed node on which to graft v, (iii) forming a new directed graph G' over vertices V by grafting v to u; it is straightforward to show that the resulting graph G' is also a tree. Note that there is an involution g on the set of all pairs (G, v, u) satifying the above constraints. That is, if $(G', v', u') = g(G, v, u)$ then there exist unique vertices v', u' of G such that $g(G', v', u') = (G, v, u)$ (for example, u' is the parent of v in G). Note that this set of transitions is capable of changing the parent vertex of an object to a different parent object, changing the parent vertex of an object from the root (world frame) to any other object, or changing the parent vertex from another object to the world frame, depending on the random choice of v and u. In general, changing the graph structure also necessitates changing the parameters (θ_v) for the severed node v

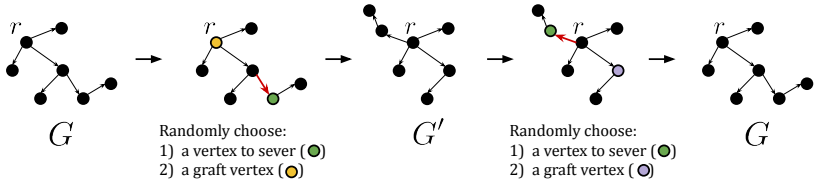


Figure 4: A reversible transition between scene graph structure G and scene graph structure G'.

and possibly other vertices. The full involutive MCMC kernel and a proof that it is invariant for the target distribution $p(G, \boldsymbol{\theta}|N, \mathbf{c}, \mathbf{Y})$ is given in the supplement (C).

Approximately Rao-Blackwellizing object shape via pseudomarginal MCMC The acceptance probability expressions for our involutive MCMC and MH kernels targeting $p(G, \boldsymbol{\theta}|N, \mathbf{c}, \mathbf{Y})$ include factors of the form $p(\mathbf{Y}|N, \mathbf{c}, G, \boldsymbol{\theta})$, which is an intractable sum over the latent object models: $p(\mathbf{Y}|N, \mathbf{c}, G, \boldsymbol{\theta}) = \sum_{\mathbf{O}_{1:M}} p(\mathbf{O}_{1:M}) p(\mathbf{Y}|\mathbf{O}_{1:M}, N, \mathbf{c}, G, \boldsymbol{\theta})$. To overcome this challenge, we employ a pseudo-marginal MCMC approach [41] that uses unbiased estimates of $p(\mathbf{Y}|N, \mathbf{c}, G, \boldsymbol{\theta})$ obtained via likelihood weighting (that is, sampling several times from $p(\mathbf{O}_{1:M})$ and averaging the resulting $p(\mathbf{Y}|\mathbf{O}_{1:M}, N, \mathbf{c}, G, \boldsymbol{\theta})$). The resulting MCMC kernels are invariant with respect to an extended target distribution of which $p(G, \boldsymbol{\theta}|N, \mathbf{c}, Y)$ is a marginal. See supplement B for details.

4 Learning new 3D object models via probabilistic inference

Section 3 described a module that uses probabilistic inference to parse an observed point cloud into a scene graph using prior knowledge about a set of object classes $i = 1, \ldots, M$ in the form of voxel-based object shape priors $p(\mathbf{O}_i)$. This section describes a module for learning these shape priors from depth images $\mathbf{I}_1, \ldots, \mathbf{I}_T$ using inference in another generative model p'. The posterior beliefs $p'(\mathbf{O}|\mathbf{I}_1, \ldots, \mathbf{I}_T)$ from this module become the prior beliefs $p(\mathbf{O})$ used in Section 3.

We start with a uninformed prior distribution $p'(\mathbf{O}) := \prod_{i=1}^{h} \prod_{j=1}^{w} \prod_{k=1}^{l} p_{\text{occ}}^{O_{ijk}} (1 - p_{\text{occ}})^{(1-O_{ijk})}$ on the 3D shape of an object class, for a per-voxel occupancy probability p_{occ} (in our experiments, 0.5). We learn about the object's shape by observing a sequence of depth images $\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_T$ that contain views of the novel object, which is assumed to be static relative to other contents of the scene, which we call the 'map' $\mathbf{M}$. (In our experiments the map contains the novel object, a floor, a ceiling, and four walls of a rectangular room). We posit the following joint distribution over object shape ($\mathbf{O}$) and the observed depth images, conditioned on the map ($\mathbf{M}$) and the poses of the camera relative to the map over time ($\mathbf{x}_1, \ldots, \mathbf{x}_T \in SE(3)$): $p'(\mathbf{O}, \mathbf{I}_{1:T}|\mathbf{M}, \mathbf{x}_{1:T}) := p'(\mathbf{O}) \prod_{t=1}^{T} p'(\mathbf{I}_t|\mathbf{O}, \mathbf{M}, \mathbf{x}_{1:T})$.

The likelihood p' is a depth image likelihood on a latent 3D voxel occupancy grid (see supplement for details). For this model, we can compute $p'(\mathbf{O}|\mathbf{M}, \mathbf{x}_{1:T}, \mathbf{I}_{1:T}) = \prod_{ijk} p'(O_{ijk}|\mathbf{M}, \mathbf{x}_{1:T}, \mathbf{I}_{1:T})$ exactly using ray marching to decide if a voxel cell is occupied, unoccupied, or unobserved (due to being occluded by another occupied cell), and the resulting distribution on $\mathbf{O}$ can be compactly represented as an array of probabilities ($\in [0, 1]^{h \times w \times l}$). However, in real-world scenarios the map $\mathbf{M}$ and the camera poses $\mathbf{x}_{1:T}$ are not known with certainty. To handle this, our algorithm takes as input uncertain beliefs about $\mathbf{M}$ and $\mathbf{x}_{1:T}$ (denoted $p'(\mathbf{M}, \mathbf{x}_{1:T})$) that are produced by a separate probabilistic SLAM (simultaneous localization and mapping) module. These beliefs take the form of a weighted collection of K particles $(\mathbf{M}^{(k)}, \mathbf{x}_{1:T}^{(k)})$: $p'(\mathbf{M}, \mathbf{x}_{1:T}) = \sum_{k=1}^{K} w_k \delta_{\mathbf{M}^{(k)}}(\mathbf{M}) \delta_{\mathbf{x}_{1:T}^{(k)}}(\mathbf{x}_{1:T})$ Various approaches to probabilistic SLAM can be used; we implemented SLAM using sequential Monte Carlo (SMC) in Gen (see supplement for details). From the beliefs $p'(\mathbf{M}, \mathbf{x}_{1:T})$ produced by SLAM, we compute the object shape posterior via:

$$p'(\mathbf{O}|\mathbf{I}_{1:T}) = \iint p'(\mathbf{O}|\mathbf{M}, \mathbf{x}_{1:T}, \mathbf{I}_{1:T}) p'(\mathbf{M}, \mathbf{x}_{1:T}) d\mathbf{M} d\mathbf{x}_{1:T} = \sum_{k=1}^{K} w_k p'(\mathbf{O}|\mathbf{M}^{(k)}, \mathbf{x}_{1:T}^{(k)}, \mathbf{I}_{1:T})$$

Note that while $p'(\mathbf{O}|\mathbf{M}^{(k)}, \mathbf{x}_{1:T}^{(k)}, \mathbf{I}_{1:T})$ for each k and can be compactly represented, the mixture distribution $p'(\mathbf{O}|\mathbf{I}_{1:T})$ lacks the conditional independencies that make this possible. To produce a more compact representation of our beliefs about the object's shape (that is portable to downstream

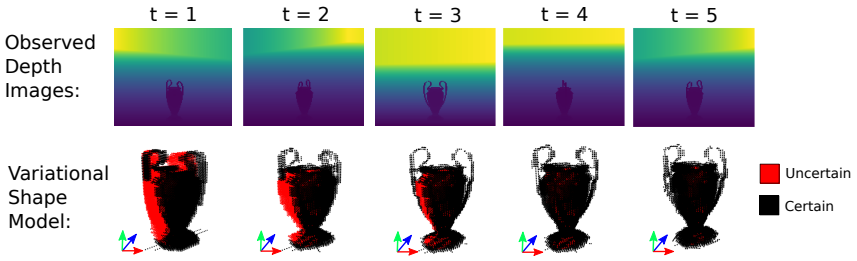


Figure 5: Learning shape models. Top row: Observed depth images of a trophy object viewed from 5 perspectives Bottom row: Visualizing the variational shape distribution $q(\mathbf{O})$ at each timestep. Cells with $q(O_{ijk}) > 0.7$ are black, cells with $q(O_{ijk}) = 0.5$ are red. While initially there is significant uncertainty due to occlusions, in the end we are only uncertain about voxels *inside* the trophy.

probabilistic inference modules), we fit a variational approximation $q_\varphi(\mathbf{O})$ that assumes independence among voxel cells ($q_\varphi(\mathbf{O}) := \prod_{i\in[h]}\prod_{j\in[w]}\prod_{k\in[l]} \varphi_{ijk}^{O_{ijk}} \cdot (1-\varphi_{ijk})^{(1-O_{ijk})}$) to $p'(\mathbf{O}|\mathbf{I}_{1:T})$ using $\varphi^* := \arg\min_\varphi \mathrm{KL}(p'(\mathbf{O}|\mathbf{I}_{1:T})||q_\varphi(\mathbf{O}))$ (see supplement for details). Fig. 5 shows input depth images ($\mathbf{I}_{1:T}$) and resulting samples from the variational voxel shape distribution learned from just $T = 5$ observations. After learning these shape distributions $q_{\boldsymbol{\varphi}}(\mathbf{O}) \approx p'(\mathbf{O}|\mathbf{I}_{1:T})$ for each distinct object class, we use them as the shape priors $p(\mathbf{O}_i)$ within the scene parsing algorithm of Section 3.

5 Experiments

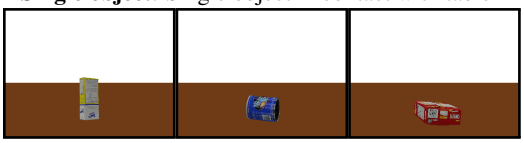
We combine our object learning (Section 4) and scene parsing (Section 3) modules into a method for few-shot 6DoF pose estimation from depth images, and compare this method with two deep learning baselines. Our method, despite being trained on only a few synthetic depth images, generalizes to challenging scenes better than the baselines. Also, when initialized with a deep-learning baseline our method significanly outperforms the baseline on the YCB-Video benchmark.

5.1 Datasets and Baselines

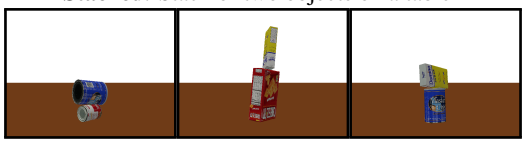
YCB-Video. We evaluate our method on real-world data using the YCB-Video dataset, which contains 91 real videos of different static indoor scenes featuring subsets of 21 objects [42] viewed by a moving camera. Ground-truth 6DoF poses of objects and the camera for each frame available.

YCB-Challenging. To compare the robustness and generalizability of our method to existing methods, we created a synthetic dataset of more challenging scenes containing contact structure, new perspectives of objects, and occlusions. Our synthetic dataset, which features five of the 21 YCB-Video object classes (rendered using the YCB mesh models [42] with simple lighting), consists of 500 RGB-D images in each of the following four types of difficult scenes (the RBG images from three examples of each scene type are shown):

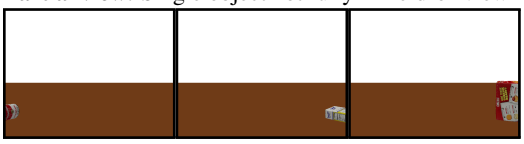
Single object: Single object in contact with table



Stacked: Stack of two objects on a table



Partial view: Single object not fully in field-of-view



Partially Occluded: One object occluded by another

We compare against two deep learning baselines for 6DoF object pose estimation from RGB-D data:

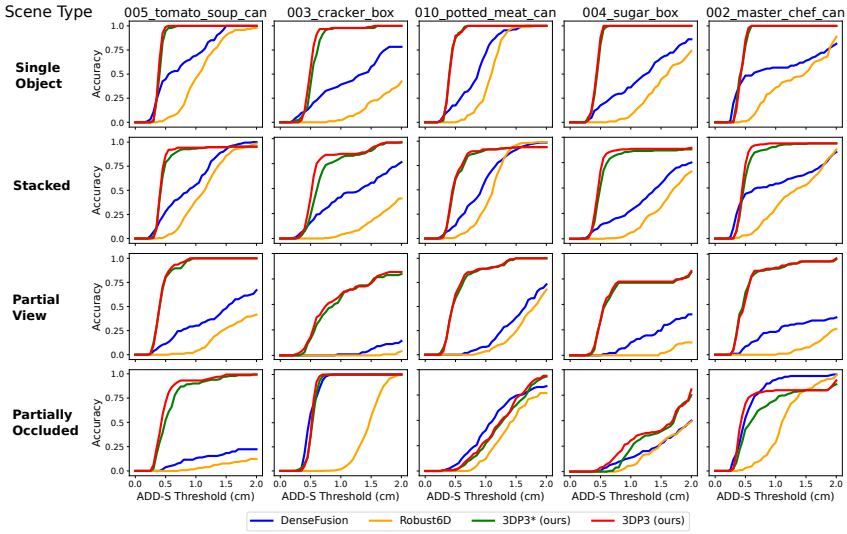


Figure 6: Accuracy of our method and two deep learning baselines (DenseFusion [20] and Robust6D [21]) on the task of 6DoF pose estimation in our synthetic 'YCB-Challenging' dataset. For each of the 4 scene types (rows) and 5 object classes (columns), we measure accuracy for a range of ADD-S thresholds. 3DP3* denotes an ablated version of our method without inference of the scene graph structure and thus object-object contact (i.e. we fix the scene graph to G_0)

		Tomato Soup			Cracker Box			Potted Meat			Sugar Box			Master Chef		
Scene Type	**Method**	ADD-S Q1	ADD-S Q2	ADD-S Q3	ADD-S Q1	ADD-S Q2	ADD-S Q3	ADD-S Q1	ADD-S Q2	ADD-S Q3	ADD-S Q1	ADD-S Q2	ADD-S Q3	ADD-S Q1	ADD-S Q2	ADD-S Q3
Single object	3DP3 (ours)	**0.35**	**0.39**	**0.41**	**0.43**	**0.49**	**0.54**	**0.36**	0.40	0.45	**0.38**	**0.43**	**0.48**	0.37	0.43	**0.48**
	3DP3* (ours)	0.35	0.40	0.43	0.47	0.52	0.62	0.36	**0.39**	**0.44**	0.40	0.45	0.49	0.35	**0.41**	0.49
	DenseFusion	0.35	0.55	1.11	0.65	1.35	1.72	0.58	0.88	1.02	0.67	1.25	1.72	**0.32**	0.61	1.85
	Robust6D	0.84	1.05	1.29	1.65	2.22	2.90	0.97	1.09	1.21	1.25	1.61	2.02	0.83	1.48	1.89
Stacked	3DP3 (ours)	**0.37**	**0.42**	**0.46**	**0.46**	**0.52**	**0.60**	**0.39**	**0.45**	**0.60**	**0.40**	**0.46**	**0.52**	0.40	**0.45**	**0.51**
	3DP3* (ours)	0.38	0.42	0.48	0.50	0.60	0.79	0.40	0.46	0.64	0.43	0.49	0.61	0.41	0.47	0.56
	DenseFusion	0.49	0.87	1.21	0.66	1.33	1.97	0.63	0.92	1.16	0.93	1.42	1.96	**0.37**	0.66	1.83
	Robust6D	0.84	1.15	1.36	1.68	2.21	2.86	0.92	1.11	1.26	1.37	1.72	2.21	0.94	1.38	1.82
Partial view	3DP3 (ours)	0.34	**0.40**	**0.47**	**0.54**	**0.76**	1.56	**0.36**	**0.45**	**0.59**	0.47	**0.55**	**1.80**	**0.36**	0.47	**0.57**
	3DP3* (ours)	**0.33**	0.40	0.47	0.56	0.90	**1.54**	0.37	0.45	0.63	**0.46**	0.59	1.81	0.36	**0.46**	0.58
	DenseFusion	0.79	1.52	2.10	2.33	2.93	3.81	1.26	1.65	2.07	1.52	2.14	2.78	1.05	2.22	2.71
	Robust6D	1.43	2.25	2.93	3.40	4.03	4.77	1.51	1.83	2.13	2.24	2.99	4.30	1.97	2.50	3.27
Partially Occluded	3DP3 (ours)	**0.36**	**0.42**	**0.52**	0.48	0.52	**0.55**	0.91	1.25	1.57	**0.89**	**1.69**	**1.97**	**0.36**	**0.42**	**0.55**
	3DP3* (ours)	0.39	0.49	0.64	0.48	0.53	0.58	0.97	1.29	1.66	1.03	1.72	1.99	0.43	0.58	1.01
	DenseFusion	–	–	–	**0.41**	**0.48**	0.58	**0.81**	**1.11**	**1.53**	1.47	2.01	3.18	0.38	0.48	0.64
	Robust6D	–	–	–	1.30	1.48	1.61	1.15	1.46	1.87	1.48	2.02	3.24	0.94	1.10	1.33

Table 1: Robustness of inference. We quantify the ADD-S error at 1st, 2nd, and 3rd quartiles for each scene type and object class in the synthetic dataset of hard scenes. A value of – indicates the method made no prediction for the object's pose. 3DP3* denotes an ablated version of our method without inference of the scene graph structure and thus object-object contact.

DenseFusion. DenseFusion [20] is a deep learning system that extracts dense feature embeddings from RGB-D to estimate pose, and uses an end-to-end pose refinement procedure at inference time. DenseFusion was trained on 16,000 real RGB-D images and 80,000 synthetic RGB-D images from the YCB-Video dataset. Training takes approximately 1 hour per epoch and requires $\geq$ 10 epochs.

Robust6D. Robust6D [21] is an end-to-end system that extracts dense features from RGB-D images to estimate rotation and translation in separate branches, and uses a RANSAC-based voting scheme to produce a final estimate. Robust6D is inspired by DenseFusion and uses the same training data. We include Robust6D because it is reported to improve the robustness of DenseFusion.

5.2 Results

Accuracy and robustness of 6DoF pose estimation on the YCB-Challenging dataset We first evaluate accuracy and robustness of each method on our YCB-Challenging dataset. We train our

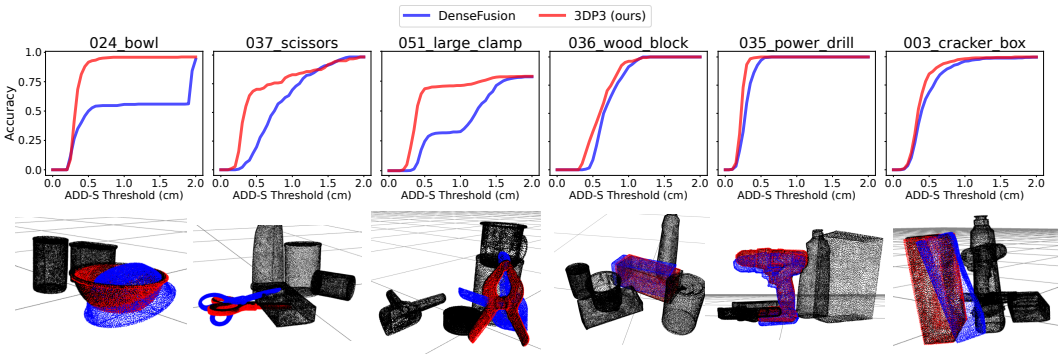


Figure 7: Comparing the accuracy of DenseFusion [20] and our approach (extended to use DenseFusion during the initial pose estimation stage) in 6DoF pose estimation on the YCB-Video test set. We also show example scenes illustrating the types of improvements our method makes over DenseFusion. Black: All scene objects placed at ground truth poses. Blue: Pose estimated by DenseFusion. Red: Pose estimated by our approach.

method on 5 depth images of each object type from different perspectives. In contrast, the baselines are trained on thousands of scenes containing these objects. We quantify the accuracy of pose estimates with ADD-S, the average closest point distance between the 3D object model points at the predicted pose and the ground truth pose [20]. In Fig. 6, for each scene type and object type, we show the fraction of pose estimates with ADD-S below various thresholds. For a majority of the scenes and objects, our approach significantly outperforms the baselines. For a few scenes, at 2cm, the baselines are comparable or slightly outperform our method. But in almost all scenes, for the 0.5cm and 1.0cm threshold our method is superior. In addition, our full method consistently outperforms the ablated version, which doesn't model contact relationships. To evaluate robustness, we inspect the 1st, 2nd, and 3rd quartile errors (Q1, Q2, Q3) for each scene type and object type (show in Table 1). For Q3 errors our method consistently outperforms the baselines and we find the drop in performance from Q1 and Q3 is much smaller for our method. In summary, our method generalizes to these difficult scenes better than the baselines, despite being trained on significantly less data.

Evaluating hybrids of our method with deep learning on the YCB-Video dataset Next we also implemented a hybrid method that leverages the speed of bottom-up inference and the accuracy and robustness of our top-down approach in order to scale to real scenes with many objects and complex contact structure. We use DenseFusion to aid in the initial MAP estimation phase of our method. Fig. 7(top) compares performance of the hybrid approach with DenseFusion alone on the YCB-Video test dataset of 2,949 real RGB-D images of 12 different scenes. We show results for objects where there exists a significant performance difference between the two methods. (See supplement for full results). Fig. 7(bottom) shows example scenes and pose estimates of both methods, illustrating the improvements due to our method. For the bowl, drill, and cracker box, our prior on planar contacts helps correct skewed pose predictions. For the scissor and wood block, since our generative model is occlusion-aware, our method can explain why part of the object is unobserved and infer the correct pose in spite of it. For the clamp object, our method corrects a large error made by DenseFusion.

6 Discussion

This paper shows that it is possible to use probabilistic programming and top-down model-based inference to make quantifiable improvements over the SOTA in terms of robustness and generalization, for a narrow 3D vision task. It also shows that it is possible to integrate 3D object shape modeling and scene parsing into a modular system that learns new objects from few examples and parse complex scenes containing them. Limitations to be addressed in future work include improving the speed of inference (currently a few seconds for typical synthetic scenes) via hybrids of variational learning, geometric heuristics, and Monte Carlo. It would also be appealing to integrate object learning with scene parsing, to try to build a system that learns, online, to perceive the structure of the 3D world.

References

[1] Alan Yuille and Daniel Kersten. Vision as bayesian inference: analysis by synthesis? *Trends in cognitive sciences*, 10(7):301–308, 2006.

[2] Daniel Kersten, Pascal Mamassian, and Alan Yuille. Object perception as bayesian inference. *Annu. Rev. Psychol.*, 55:271–304, 2004.

[3] Tai Sing Lee and David Mumford. Hierarchical bayesian inference in the visual cortex. *JOSA A*, 20(7):1434–1448, 2003.

[4] David C Knill, Daniel Kersten, and Alan Yuille. Introduction: A bayesian formulation of visual perception. *Perception as Bayesian inference*, 1:1–21, 1996.

[5] Vikash K Mansinghka, Tejas D Kulkarni, Yura N Perov, and Joshua B Tenenbaum. Approximate bayesian image interpretation using generative probabilistic graphics programs. *arXiv preprint arXiv:1307.0060*, 2013.

[6] Dileep George, Wolfgang Lehrach, Ken Kansky, Miguel Lázaro-Gredilla, Christopher Laan, Bhaskara Marthi, Xinghua Lou, Zhaoshi Meng, Yi Liu, Huayan Wang, et al. A generative vision model that trains with high data efficiency and breaks text-based captchas. *Science*, 358(6368), 2017.

[7] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 18(9):884–900, 1996.

[8] Zhuowen Tu and Song-Chun Zhu. Image segmentation by data-driven markov chain monte carlo. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):657–673, 2002.

[9] Lukasz Romaszko, Christopher KI Williams, Pol Moreno, and Pushmeet Kohli. Vision-as-inverse-graphics: Obtaining a rich 3d explanation of a scene from a single image. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 851–859, 2017.

[10] Pol Moreno, Christopher KI Williams, Charlie Nash, and Pushmeet Kohli. Overcoming occlusion with inverse graphics. In *European Conference on Computer Vision*, pages 170–185. Springer, 2016.

[11] Tejas D Kulkarni, Pushmeet Kohli, Joshua B Tenenbaum, and Vikash Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 4390–4399, 2015.

[12] Marco Cusumano-Towner, Alexander K Lew, and Vikash K Mansinghka. Automating involutive mcmc using probabilistic and differentiable programming. *arXiv preprint arXiv:2007.09871*, 2020.

[13] Armen Avetisyan, Angela Dai, and Matthias Nießner. End-to-end cad model retrieval and 9dof alignment in 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2551–2560, 2019.

[14] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2614–2623, 2019.

[15] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 998–1005. Ieee, 2010.

[16] Stefan Hinterstoisser, Vincent Lepetit, Naresh Rajkumar, and Kurt Konolige. Going further with point pair features. In *European conference on computer vision*, pages 834–848. Springer, 2016.

[17] Tolga Birdal and Slobodan Ilic. Point pair features based object detection and pose estimation revisited. In *2015 International Conference on 3D Vision*, pages 527–535. IEEE, 2015.

[18] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018.

[19] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[20] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3343–3352, 2019.

[21] Meng Tian, Liang Pan, Marcelo H Ang Jr, and Gim Hee Lee. Robust 6d object pose estimation by learning rgb-d features. In *International Conference on Robotics and Automation (ICRA)*, 2020.

[22] Siyuan Huang, Siyuan Qi, Yixin Zhu, Yinxue Xiao, Yuanlu Xu, and Song-Chun Zhu. Holistic 3d scene parsing and reconstruction from a single rgb image. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 187–203, 2018.

[23] Yixin Chen, Siyuan Huang, Tao Yuan, Siyuan Qi, Yixin Zhu, and Song-Chun Zhu. Holistic++ scene understanding: Single-view 3d holistic scene parsing and human pose estimation with human-object interaction and physical commonsense. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8648–8657, 2019.

[24] Nilesh Kulkarni, Ishan Misra, Shubham Tulsiani, and Abhinav Gupta. 3d-relnet: Joint object and relational network for 3d prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2212–2221, 2019.

[25] Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. Efficient nonparametric belief propagation for pose estimation and manipulation of articulated objects. *Science Robotics*, 4(30), 2019.

[26] Xiaotong Chen, Rui Chen, Zhiqiang Sui, Zhefan Ye, Yanqi Liu, R Iris Bahar, and Odest Chadwicke Jenkins. Grip: Generative robust inference and perception for semantic robot manipulation in adversarial environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3988–3995. IEEE, 2019.

[27] Jared Glover, Gary Bradski, and Radu Bogdan Rusu. Monte carlo pose estimation with quaternion kernels and the bingham distribution. In *Robotics: science and systems*, volume 7, page 97, 2012.

[28] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking. *IEEE Transactions on Robotics*, 2021.

[29] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[30] Onur Ozyesil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion. *arXiv preprint arXiv:1701.08493*, 2017.

[31] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[32] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. Bundle adjustment in the large. In *European conference on computer vision*, pages 29–42. Springer, 2010.

[33] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pages 405–421. Springer, 2020.

[34] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[35] Dirk Reiners. *OpenSG: A scene graph system for flexible and efficient realtime rendering for virtual and augmented reality applications*. PhD thesis, Reiners, Dirk, 2002.

[36] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[37] Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, 41:477–526, 2011.

[38] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5664–5673, 2019.

[39] Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. *arXiv preprint arXiv:2002.06289*, 2020.

[40] Marco F Cusumano-Towner, Feras A Saad, Alexander K Lew, and Vikash K Mansinghka. Gen: a general-purpose probabilistic programming system with programmable inference. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 221–236, 2019.

[41] Christophe Andrieu, Gareth O Roberts, et al. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

[42] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. *arXiv preprint arXiv:1502.03143*, 2015.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] At this stage of the work, we do not see potential negative impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We evaluate on a large data sets (2000+ scenes with multiple objects), so we expect variations between runs will have little effect on overall accuracy metrics.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] This is a publicly available data set which used in many research papers.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The data consists of images of objects on tables.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The data consists of images of objects on tables.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]