Structured differentiable models of 3D scenes via generative scene graphs

Ben Zinberg
bzinberg@mit.edu

Marco Cusumano-Towner
marcoct@mit.edu

Vikash K. Mansinghka
vkm@mit.edu

1 Introduction

Generative approaches to 3D scene perception and physical reasoning are increasingly common [1, 2, 3, 4, 5, 6, 7]. There is a widespread need for scene models that can incorporate planar and pointwise contacts between physical objects, and ensure that objects do not inter-penetrate [8]. Generic priors on 6DOF poses — and bottom-up neural networks that de-render images into scenes [6] — typically violate these constraints.

This abstract introduces a family of generative models for 3D scenes that respect pairwise physical contact constraints between objects. The technical innovation is to represent scenes in terms of hybrid symbolic-numerical scene graphs over objects, where the edges correspond to parameterized contact relationships (see Figure 1). Given this representation, 3D poses can be generated via a graph traversal. We show how to define prior distributions on scene graphs, and how 3D scene understanding can be phrased as posterior inference over the scene graph. This abstract also shows preliminary evidence that it is possible to infer scene graph parameters from empirical data, "de-rendering" images into structured 3D scene representations. This is implemented using the Gen probabilistic programming language. Finally, this abstract briefly discusses representation and inference challenges that need to be addressed to scale up the approach to more complex, real-world scenes.

2 Scene graphs

In this section we define a scene graph that can explicitly model pairwise physical contact relationships between objects. In this framework, a scene graph is a directed dependency tree over objects that have parametric 3D shapes. Object types come from an extensible library. Edges in the graph have parameters that describe how the pose for an object is generated as a function of the pose and shape of the object it depends on. Crucially, edge parametrizations can represent physical contact constraints that would not be represented by a collection of independent 6DOF poses for each object in the scene (see Figure 3). We give examples of scene graphs involving objects of two types: floor planes and cubes. We also show (see Figure 1) the effect of different graph structures on the continuous parameterization by representing the same physical scene using two different scene graphs.

Let $\mathcal{S}$ be a finite set of possible *shape types*. For each $s \in \mathcal{S}$ let Ψ_s denote the set of possible (continuous) shape parameters for shape type s. For each pair of shape types $s, t \in \mathcal{S}$ let $\mathcal{C}_{s,t}$ denote a finite set of possible *contact types* for these two shape types. For each possible contact type c let Φ_c denote the set of (continuous) contact parameters for contact type c. Throughout, we use the convention that V denotes a set of vertices, and (V, E) denotes a directed tree over vertices with edges $E \subset V \times V$. Let $\pi_E(i)$ for $i \in V$ denote the *parent* of vertex i, with value $\perp$ if vertex i is the root.

Workshop on Perception as Generative Reasoning, NeurIPS 2019, Vancouver, Canada.

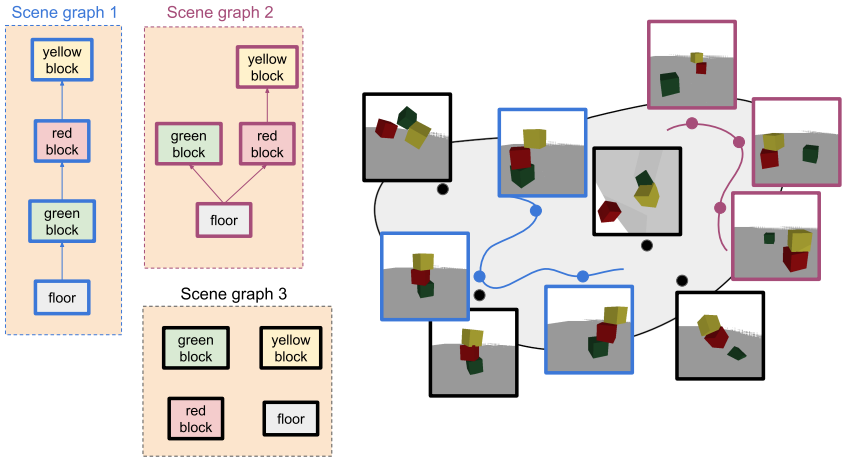

Figure 1: Several random scenes conditioned on each of three contact graphs. For a depth-1 scene graph (black), any time two objects are touching, those objects are also interpenetrating (i.e., violating a physical constraint) with probability 1. For a scene graph representing a single stack of boxes on the floor (blue), all the scenes are physically plausible (no interpenetration), and the set of such scenes is a 15-dimensional submanifold (shown in blue) of the 24-dimensional space of free poses (shown in gray). In magenta, we show a different scene graph, which represents two stacks of boxes sitting on the floor. Note that the scene graph does not guarantee non-interpenetration between the red box and the green box; it only guarantees non-interpenetration between objects that have an edge between them in the scene graph.

Definition 2.1. *Given shape types $\mathcal{S}$, contact types $\mathcal{C}$, shape parametrization Ψ and contact parametrization Φ, a* scene graph *$\mathcal{G}$ is a tuple $\mathcal{G} = (V, E, s, c, \psi, \phi)$ where:*

- *(V, E) is a directed tree where vertices $i \in V$ represent objects and edges $(i, j) \in E$ represent directed contact relationships between object i and object j.*
- *$s_i \in \mathcal{S}$ for each object $i \in V$ is the* shape type *for the object.*
- *$\psi_i \in \Psi_{s_i}$ for $i \in V$ are the* shape parameters *of object i*
- *$c_i \in \mathcal{C}_{s_{\pi(i)}, s_i}$ is the* contact type *for the contact between object $\pi(i)$ and object i.*
- *$\phi_i \in \Phi_{s_{\pi(i)}, s_i}$ for $i \in V$ are the* contact parameters *for the contact between $\pi(i)$ and i.*

Recursively computing the 6DOF poses of objects Let $\mathbf{z}_i$ denote the 6DOF pose of vertex $i \in V$. For notational and implementational convenience, we treat the root node $\mathbf{z}_{root}$ specially: it has no spatial extent, and its 6DOF pose defines the base coordinate frame with respect to which the pose of all objects are defined. Given a scene graph $\mathcal{G}$ and a 6DOF pose $\mathbf{z}_{root}$ for the root node, we recursively compute the 6DOF pose of every other node in the graph using the following recursion over the scene graph tree, where i ranges over all nodes in V except the root node:

$$\mathbf{z}_i = f_{c_i}(\phi_i, \psi_i, \mathbf{z}_{\pi(i)}, \psi_{\pi(i)}). \tag{1}$$

The functions f_c for $c \in \mathcal{C}$ implement the geometry of contact parameterization, namely, how to compute the pose of the child object ($\mathbf{z}_i$) given the contact parameters (ϕ_i), the pose of the parent object ($\mathbf{z}_{\pi(i)}$), and the shape parameters of the parent and child objects ($\psi_{\pi(i)}$ and ψ_i respectively). Note that form of f_{c_i} depends on the contact type c_i. In Section A.2 we will use the 6DOF poses of all objects to compose a depth image of the scene.

Example The framework for scene graphs above gives a great deal of freedom in the choice of shape types $\mathcal{S}$, contact types $\mathcal{C}$, and how contacts are used to compute 6DOF poses. We now describe an example. Suppose our set of shapes types includes cubes and planes, so that $(\texttt{cube}, \texttt{plane} \in \mathcal{S})$. We parametrize the shape of a cube using its side length, so that $\Psi_{\texttt{cube}} := (0, \infty)$. Planes have no shape parameters ($\Psi_{\texttt{plane}}$ is a singleton set). Suppose we want to model the possibility that two

cubes are in face-to-face contact, and that a cube and a plane are in face-to-face contact. Note that various other contact types (e.g. point–edge and edge–edge) between cubes and planes are also possible, but beyond the scope of this paper. Let `ccf2f` be a token that stands for 'cube-cube-face-to-face' and let `cpf2f` be a token that stands for 'cube-plane-face-to-face'. Then we define the set of contact types for two cubes to be all choices of one face from each cube:

$$\mathcal{C}_{\texttt{cube},\texttt{cube}} := \{(\texttt{ccf2f}, a, b) : a, b \in \{1, \ldots, 6\}\} \tag{2}$$

and the set of contact types for a plane and a cube consists of all choices of (i) a face of the cube (there are six of these), and (ii) a face of the plane (there are two of these):

$$\mathcal{C}_{\texttt{cube},\texttt{plane}} = \mathcal{C}_{\texttt{plane},\texttt{cube}} := \{(\texttt{cpf2f}, a, b) : a \in \{1, \ldots, 6\}, b \in \{0, 1\}\}. \tag{3}$$

For face-to-face contacts between cubes we assume that the exterior sides of the two faces are in contact, and flush with one another. We parametrize these contacts (of type $(\texttt{ccf2f}, a, b)$) by three degrees of freedom (two translation and one rotation), so that $\Phi_{(\texttt{ccf2f},a,b)} := [0, \pi/2] \times [0, 1] \times [0, 1]$ for all $a, b \in \{1, \ldots, 6\}$, as shown in Figure 2.[1] For face-to-face contacts between cubes and planes, the translation parameters are unbounded, so that $\Phi_{(\texttt{cpf2f},a,b)} := [0, \pi/2] \times \mathbb{R}^2$ for all $(a, b) \in \{1, \ldots, 6\} \times \{0, 1\}$. Suppose that we also want to allow objects in the scene have independent 6DOF poses from one another (e.g. so they are not required to by in physical contact). We can represent this within our scene graph by introducing an additional shape type $\texttt{frame} \in \mathcal{S}$, that has no shape parameters. Then, for each $s \in \mathcal{S}$ we introduce a contact type $\mathcal{C}_{s,\texttt{frame}} = \mathcal{C}_{\texttt{frame},s} := \{(\texttt{6dof}, \texttt{frame}, s)\}$ with contact parameters $\Phi_{(\texttt{6dof},\texttt{frame},s)}$ that range over all possible relative 6DOF poses of the child object with respect to the pose of the parent 'frame' object.

Figure 3 shows two different scene graphs involving a plane and three cubes. The two scene graphs over objects $V = \{1, 2, 3, 4, 5\}$ represent the same set of 6D poses for all objects, but using different graph structure. In the first graph, each of the cubes are assumed to be in flush contact with another object, giving the following graph topology:

$$1(\texttt{frame}) \to 2(\texttt{plane}) \to 3(\texttt{cube}) \to 4(\texttt{cube}) \to 5(\texttt{cube}).$$

In the second graph, each object has an independent 6DOF pose, and the graph topology is:

$$1(\texttt{frame}) \to 2(\texttt{plane}) \mid 1(\texttt{frame}) \to 3(\texttt{cube}) \mid 1(\texttt{frame}) \to 4(\texttt{cube}) \mid 1(\texttt{frame}) \to 5(\texttt{cube}).$$

Note that these two scene graphs behave differently under the action of inference operations. For example, in the first graph, if we perturb the contact parameters ϕ_3 of cube $3 \in V$ relative to its parent object $2 \in V$, then, the 6DOF poses of descendent objects $4, 5 \in V$ are also affected, whereas in the second graph they are not.

Scenes in which objects are touching or resting on one another are commonplace, yet cannot be generated by generic priors on object pose. A crucial property of scene graphs is that by jointly sampling graph structure and pose given graph structure, one can specify priors that place positive measure on scenes in which objects are flush against one another, yet locally do not interpenetrate. Compared to approaches based on optimization subject to constraints [8], this approach greatly reduces both the number of constraints and the dimension of the state space; any remaining constraints that cannot be handled this way (e.g., global non-interpenetration between objects separated by many links in the scene graph) can be incorporated into inference via additional likelihood terms that penalize violation of those constraints.

Consider Figure 1, which shows three scene graphs that represent qualitatively different scene structures, along with three scenes sampled from the distribution on images induced by each graph structure. These images were obtained by generating random parameters, calculating poses induced by those parameters, and rendering the scenes that result. Scene graphs 1 (blue) and 2 (purple) include cubes resting on one another, but also exhibit broad variability in pose given those qualitative constraints.

The semantics of scene graphs may need to be significantly richer before they will be suitable for accurately describing generic indoor and outdoor scenes in the real world. For example, it might help to have higher-level edge types, such as "object a is on top of object b," which encapsulate a

[1] Here we have assumed that the cubes are rotationally symmetric, and thus assume without loss of generality we can assume the rotation angle lies in the interval $[0, \pi/2]$. A full treatment of optimizing around symmetries is beyond the scope of this paper.

latent choice of geometric structure that is guaranteed to satisfy a desired set of higher-level spatial relationships. It also might help to support qualitative relationships that induce undirected constraints on scenes, such as "object a does not touch object b." It will also be important to integrate scene graphs with models of physical dynamics subject to contact constraints [9] and also models of goal-directed behavior of agents based on planning [10] and inverse planning [11, 12]. Additionally, it is desirable to support arbitrary acyclic scene graphs, not just trees; while possible in principle, this extension carries the challenge of satisfying the (not always solvable) constraints implied by an object's simultaneous relation to multiple parents. In principle, these constraints — such as global non-interpenetration constraints [8] — could be handled by extensions to the model. We hope to pursue these extensions in future work.

3 Inference

We now briefly discuss a prototype implementation of a system that infers scene graph structure from data, written using the Gen probabilistic programming language [13]. (See Appendix A for an explanation of the joint generative model of scenes and images.) We performed inference via both MCMC and gradient-based MAP estimation. In the MCMC case, we (i) initialize via importance resampling, and (ii) perform resimulation Metropolis–Hastings updates [14, 15] to the scene graph parameters. Note that updates to parameters for upstream edges alter the pose not just for the child object of the edge, but for all downstream objects. Figure 4 shows the data and some example inference results. In the gradient case, we fix the scene graph structure and the pose of all but one object in the scene, and use gradient MAP estimation to infer the pose of the remaining object. Figure 5 shows the data and some example inference results.

In the current work the MCMC and gradient algorithms were run separately, but it is straightforward to combine them. In future work we hope to explore hybrid inference strategies that include both MCMC moves (especially for inferring structure) and gradient moves (for inferring continuous parameters); we believe such hybrids will be a key innovation towards inference algorithms that are both general and scalable.

Once a scene graph has been inferred, it is straightforward to use a physics engine to simulate the trajectories of the physical objects in the scene. We assume that object types have known material properties, such as mass and elasticity, though an extension to unknown masses and elasticities is conceptually and technically straightforward [16]. Compared to neural de-animation [6], which de-renders images into sets of objects with generic poses, our approach infers initial scenes that include face-to-face contacts. In future work, we hope to develop a scene-graph–friendly version of the contact-aware dynamics engine from MuJoCo [9]. This would allow for joint inference over initial conditions and physical object properties, given noisy/low-resolution physics that is still guaranteed to preserve qualitative contacts between objects.

3.1 Scaling up inference

We are a long way from algorithms that rapidly and robustly infer scene graph structure and parameters from data. Some specific approaches that we hope to pursue include (i) birth–death proposals that create and delete objects [17, 18, 19]; (ii) node and edge parameterizations that facilitate "annealing in" of new objects via both implicit [2] and explicit [20] annealing; (iii) moves that make large, global adjustments to the scene graph, that implicitly change the blocking of proposals to edge parameters [21]; and (iv) heuristics for joint inference over scene graph fragments that combine variational learning with computer-vision heuristics [13, 22]. Combinations of automatic differentiation (to infer object pose) with bottom up proposals (to detect objects) and SMC (to infer graph structure) to seem especially promising. Our prototype is implemented in Gen [13], a general purpose probabilistic programming language that makes it possible to apply custom combinations of these kinds of inference techniques to a broad class of open universe scene graph priors. We hope that the right combinations of these kinds of inference techniques could make it possible to someday carry out fully Bayesian de-rendering of realistic physical scenes, producing accurate symbolic descriptions of real-world images and videos.

References

[1] Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

[2] Vikash K Mansinghka, Tejas D Kulkarni, Yura N Perov, and Josh Tenenbaum. Approximate bayesian image interpretation using generative probabilistic graphics programs. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1520–1528. Curran Associates, Inc., 2013.

[3] Tejas D. Kulkarni, Pushmeet Kohli, Joshua B. Tenenbaum, and Vikash K. Mansinghka. Picture: A probabilistic programming language for scene perception. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4390–4399. IEEE Computer Society, 2015.

[4] Varun Jampani, Sebastian Nowozin, Matthew Loper, and Peter V. Gehler. The informed sampler: A discriminative approach to bayesian inference in generative computer vision models. *CoRR*, abs/1402.0859, 2014.

[5] Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. Neural scene de-rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[6] Jiajun Wu, Erika Lu, Pushmeet Kohli, William T Freeman, and Joshua B Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, 2017.

[7] Gregory Izatt and Russ Tedrake. Generative modeling of environments with scene grammars and variational inference. *(Under review)*, 2019.

[8] Lawson L.S. Wong, Leslie P. Kaelbling, and Tomas Lozano-Perez. Collision-free state estimation. In *IEEE Conference on Robotics and Automation (ICRA)*, 2012.

[9] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, Oct 2012.

[10] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Sample-based methods for factored task and motion planning. In *Robotics: Science and Systems*, 2017.

[11] Chris L Baker, Rebecca Saxe, and Joshua B Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.

[12] Marco F Cusumano-Towner, Alexey Radul, David Wingate, and Vikash K Mansinghka. Probabilistic programs for inferring the goals of autonomous agents. *arXiv preprint arXiv:1704.04977*, 2017.

[13] Marco Cusumano-Towner, Feras Saad, Alexander Lew, and Vikash Mansinghka. Gen: A general-purpose probabilistic programming system with programmable inference. 11 2018.

[14] Noah Goodman, Vikash Mansinghka, Daniel Roy, Keith Bonawitz, and Joshua Tenenbaum. Church: A language for generative models. volume 2008, pages 220–229, 01 2008.

[15] Vikash Mansinghka, Daniel Selsam, and Yura Perov. Venture: a higher-order probabilistic programming platform with programmable inference. 03 2014.

[16] Adam Sanborn, Vikash K. Mansinghka, and Thomas L. Griffiths. Reconciling intuitive physics and newtonian mechanics for colliding objects. *Psychological review*, 120 2:411–37, 2013.

[17] PETER J. GREEN. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 12 1995.

[18] Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. *BLOG: Probabilistic Models with Unknown Objects*. MIT Press, January 2007.

[19] Nimar S. Arora, Rodrigo de Salvo Braz, Erik B. Sudderth, and Stuart J. Russell. Gibbs sampling in open-universe stochastic languages. *CoRR*, abs/1203.3464, 2012.

[20] Yi-Ting Yeh, Lingfeng Yang, Matthew Watson, Noah D Goodman, and Pat Hanrahan. Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Transactions on Graphics (TOG)*, 31(4):56, 2012.

[21] Robert G Edwards and Alan D Sokal. Generalization of the fortuin-kasteleyn-swendsen-wang representation and monte carlo algorithm. *Physical review D*, 38(6):2009, 1988.

[22] Marco F. Cusumano-Towner and Vikash K. Mansinghka. Using probabilistic programs as proposals. *CoRR*, abs/1801.03612, 2018.

A Generative models of scene graphs and depth images

We use our scene graph formalism as the basis for joint generative models of scenes and images, and we cast inference of 3D scene geometry from an image as posterior inference in these generative models. We define prior distributions on scene graphs $\mathcal{G}$, and an observation model $p(\mathcal{I}; \mathcal{G})$ that models an image $\mathcal{I}$ as the result of a noisy rendering of the scene graph $\mathcal{G}$. To reduce the effect of nuisance variables like lighting and texture, we model *depth* images instead of luminance images.

A.1 Prior distributions on scene graphs

To define probability distributions over scene graphs, including prior and posterior distributions, we first require a measure space $\mathcal{M}_{\mathcal{S},\mathcal{C},\Psi,\Phi} = (X, \Sigma, \mu)$ on scene graphs, which depends on $(\mathcal{S}, \mathcal{C}, \Psi, \Phi)$. Note that we can separate each scene graph $\mathcal{G}$ into its discrete component (V, E, s, c) and its continuous component (ψ, ϕ). For each discrete component (V, E, s, c) of the graph, we assume that the continuous component (ψ, ϕ) has fixed dimension and a given base measure (e.g. Lebesgue). We construct the stock measure $\mu_{\mathcal{S},\mathcal{C},\Psi,\Phi}$ from the counting measure on the discrete component with the appropriate base measures for the continuous components. We then represent probability distributions on scene graphs by density functions $p(\mathcal{G})$ with respect to the stock measure $\mu_{\mathcal{S},\mathcal{C},\Psi,\Phi}$.

It is possible to define open-universe probability models [18, 19] over the space of scene graphs, in which the number of objects is unbounded a-priori, but finite for any given scene. For example, we can use a consistent probabilistic context free grammar to define a normalized prior distibution over all possible discrete components (V, E, s, c), including all possible scene graph topologies. Alternatively, we can define prior distributions on scene graphs that that admit an unbounded number of nodes, but with a restricted graph topology.

Example open-universe prior distribution We now give an example of this second approach. Algorithm 1 describes a generative process for producing scene graphs that have two classes of objects: 'parent' objects and 'child' objects. The number of parent objects and the number of child objects are independently sampled from Poisson distributions. Shape types are independently drawn for each parent object from a uniform distribution over the set of possible shapes $\mathcal{S}$. The shape parameters are sampled from a shape-type-dependent prior distribution denoted $p_s(\psi)$. The shape type and shape parameters for child objects are also drawn independently. Each child object is independently and uniformly assigned to a parent object, and given parent shape type $s_\pi(i)$ and child shape type s_i, the contact type c_i and contact parameters ϕ_i are sampled from prior distributions $p_{s_\pi(i),s_i}(c_i)$ and $p_{c_i}(\phi_i)$ respectively.

Let V_k denote the function that returns the set of vertices at depth k, where $k = 0$ is the root. The prior density of this prior distribution on graphs $\mathcal{G}$ relative to the stock measure $\mu_{\mathcal{S},\mathcal{C},\Psi,\Phi}$ is denoted $p(\mathcal{G})$ and is zero if depth of the graph (including the root) is zero or more than three, or if the root

Algorithm 1: An open-universe prior distribution on scene graphs.

$n_1 \sim \text{Poisson}(\lambda_1)$
$n_2 \sim \text{Poisson}(\lambda_2)$
$V \leftarrow \{\texttt{root}\} \cup \{(\texttt{parent}, i) : i \in 1, \ldots, n_1\} \cup \{(\texttt{child}, j) : j \in 1, \ldots, n_2\}$
$s_{\texttt{root}} \leftarrow \texttt{frame}$
$E \leftarrow \{\}$
for $v \in \{(\texttt{parent}, i) : i \in 1, \ldots, n_1\}$ **do**
  $s_v \sim \text{Uniform}(\mathcal{S})$
  $\psi_v \sim p_{s_v}(\cdot)$
  $\pi(v) \leftarrow \texttt{root}$
  $c_v \leftarrow (\texttt{6dof}, \texttt{frame}, s_v)$
  $\phi_v \sim \text{Uniform6DOFPose}()$
end
for $v \in \{(\texttt{child}, j) : j \in 1, \ldots, n_2\}$ **do**
  $s_v \sim \text{Uniform}(\mathcal{S})$
  $\psi_v \sim p_{s_v}(\cdot)$
  $\pi(v) \sim \text{Uniform}(\{(\texttt{parent}, i) : i \in 1, \ldots, n_1\})$
  $c_v \sim p_{s_{\pi(v)}, s_v}(\cdot)$
  $\phi_v \sim p_{c_v}(\cdot)$
end

node v has $s_v \neq \texttt{frame}$. Otherwise, the prior density is:

$$p(\mathcal{G}) := \begin{array}{ll} \text{Pois}(|V_1(\mathcal{G})|; \lambda_1)\cdot & \text{Prior on number of parent objects} \\ \text{Pois}(|V_2(\mathcal{G})|; \lambda_2)\cdot & \text{Prior on number of child objects} \\ \prod_{v \in V} \frac{1}{|\mathcal{S}|} p_{s_v}(\psi_v)\cdot & \text{Prior on shape types and shapes parameters} \\ \prod_{v \in V_2(\mathcal{G})} \frac{1}{|V_1(\mathcal{G})|}\cdot & \text{Uniform prior on parent assignments} \\ \prod_{v \in V_2(\mathcal{G})} p_{s_{\pi(i)}, s_v}(c_v) p_{c_v}(\phi_v)\cdot & \text{Prior on contact types and contact parameters} \\ \prod_{v \in V_1(\mathcal{G})} p_{(\texttt{6dof}, \texttt{frame}, s_v)}(\phi_v) & \text{Prior on 6DOF pose of parent objects} \end{array} \tag{4}$$

A.2 Likelihood model for depth images

We assume we have access to a function g_s for each shape type $s \in \mathcal{S}$ that computes the 3D mesh geometry $\mathbf{y}_i$ of an object i from its pose $\mathbf{z}_i$ and its shape parameters ψ_i. To compute a depth rendering of a scene, we first compute the 6DOF poses $\mathbf{z}_i$ of each object, using Equation (1). Then, we apply the mesh geometry function for each object i:

$$\mathbf{y}_i = g_{s_i}(\mathbf{z}_i, \psi_i). \tag{5}$$

Then, we use standard 3D graphics techniques to compose a depth image (modeling occlusion via the 'depth test'). Suppose the depth image is $M \times N$ pixels. Let $d : \mathcal{G} \to \mathbb{R}^{M \times N}$ denote the function that starts with a scene graph $\mathcal{G}$, recursively computes 6DOF poses via Equation (1), computes mesh geometry via Equation (5), and returns the composed depth image. Note that the 6DOF pose of the root node $\mathbf{z}_{root}$ represents the pose of the scene relative to the camera. We model an observed depth image $\mathcal{I}$ as a blurred and noisy variant of $d(\mathcal{G})$. Let $b : \mathbb{R}^{M \times N} \to \mathbb{R}^{M \times N}$ denote the fuction that applies Gaussian blur to a depth image, producing a new depth image. We then model each pixel as generated from the blurred depth image, with pixel-wise Gaussian noise with variance σ^2:

$$p(\mathcal{I}; \mathcal{G}) := \prod_{m=1}^{M} \prod_{n=1}^{N} \text{Norm}(\mathcal{I}_{m,n}; b(d(\mathcal{G}))_{m,n}, \sigma) \tag{6}$$

A.3 Differentiating the likelihood with respect to continuous scene graph parameters

A key property of scene graphs is they are partially differentiable: given the discrete components (V, E, s, c) of a graph $\mathcal{G} = (V, E, s, c, \psi, \phi)$, it is possible to calculate gradients of the joint probability density over the continuous components (ψ, ϕ). Here, we show how to compute gradients of the depth image with respect to the contact parameters ϕ. A key property of these gradients is that

the gradient with respect to contact parameters for one object influences the poses of all objects that descend from it. Let $\texttt{desc}(i)$ denote the set of descendents of object i in the tree (V, E), and let $\texttt{path}(j, i)$ denote the set of vertices on the path from j to i (not including i). Then:

$$\left[\frac{\partial \mathcal{I}}{\partial \phi_i}\right] = \sum_{j \in \texttt{desc}(i)} \left[\frac{\partial \mathcal{I}}{\partial \mathbf{y}_j}\right] \left[\frac{\partial \mathbf{y}_j}{\partial \mathbf{z}_j}\right] \left[\frac{\partial \mathbf{z}_j}{\partial \phi_i}\right] \tag{7}$$

$$= \left(\sum_{j \in \texttt{desc}(i)} \left[\frac{\partial \mathcal{I}}{\partial \mathbf{y}_j}\right] \left[\frac{\partial \mathbf{y}_j}{\partial \mathbf{z}_j}\right] \prod_{k \in \texttt{path}(j,i)} \left[\frac{\partial \mathbf{z}_k}{\partial \mathbf{z}_{\pi(k)}}\right] \right) \left[\frac{\partial \mathbf{z}_i}{\partial \phi_i}\right] \tag{8}$$

where square brackets indicate Jacobian matrices. To implement these gradients efficiently and incrementally during inference, we can maintain at every vertex $i \in V$ the current Jacobian $[\partial \mathcal{I} / \partial \mathbf{z}_i]$. Whenever the shape type s_j, shape parameters ψ_j, contact type c_j, or contact parameters ψ_j of an object $j \in V$ are modified, we update the Jacobians $[\partial \mathcal{I} / \partial \mathbf{z}_i]$ for only the subset of objects that could influence the value of the Jacobian. Then, the Jacobian with respect to contact parameters for object i is

$$\left[\frac{\partial \mathcal{I}}{\partial \phi_i}\right] = \left[\frac{\partial \mathcal{I}}{\partial \mathbf{z}_i}\right] \left[\frac{\partial \mathbf{z}_i}{\partial \phi_i}\right]. \tag{9}$$

The efficiency of this approach is governed by the size of the set of parameters that could influence the value of the Jacobian $\left[\partial \mathcal{I} / \partial \phi_j\right]$. This set contains (at most) all parameters of ancestors of object j (along the path to the root), as well as the mesh geometries $\mathbf{y}_k$ of all objects (as the Jacobian $\left[\partial \mathcal{I} / \partial \phi_j\right]$ depends on all $\mathbf{y}_k$'s jointly). In future work, we hope to improve this efficiency by incorporating known correspondences between topological proximity in the scene graph and spatial proximity in the rendered image, so that recomputation of image Jacobians is limited to only a small subset of the mesh geometries $\mathbf{y}_k$. This is analogous to to the heuristics employed by game engines to avoid redrawing objects which are known not to have changed appreciably.

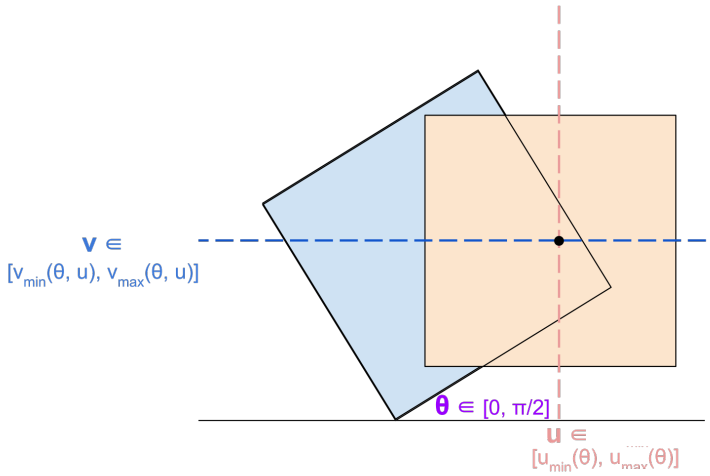

Figure 2: Concrete parameterization of a specific type of surface contact: planar contact between two square surfaces. In this parameterization, θ represents the angle between the two squares, which by symmetry can be assumed to lie in the interval $0 \leq \theta < \pi/2$ (it is also possible to break the symmetries in the cube). Conditioned on θ, there is an interval $[x_{\text{min}}, x_{\text{max}}]$ of x-offsets that maintain the child square (orange) being in contact with the parent square (blue). Conditioned on this x-offset and θ, there is similarly an interval $[y_{\text{min}}, y_{\text{max}}]$ of possible y-offsets. We normalize these x- and y-offsets by an affine transformation to obtain the parameters $u, v \in [0, 1]$. Similarly, given θ, u, v, we can apply the same transformation in reverse to obtain x- and y-offsets. Finally, the known angle θ and the fact that the two squares are in flush contact (i.e. their outward surface normals are exactly opposite) uniquely determine the relative 6D pose, which is an element of $\mathbb{R}^3 \times \text{SO}_3(\mathbb{R})$.

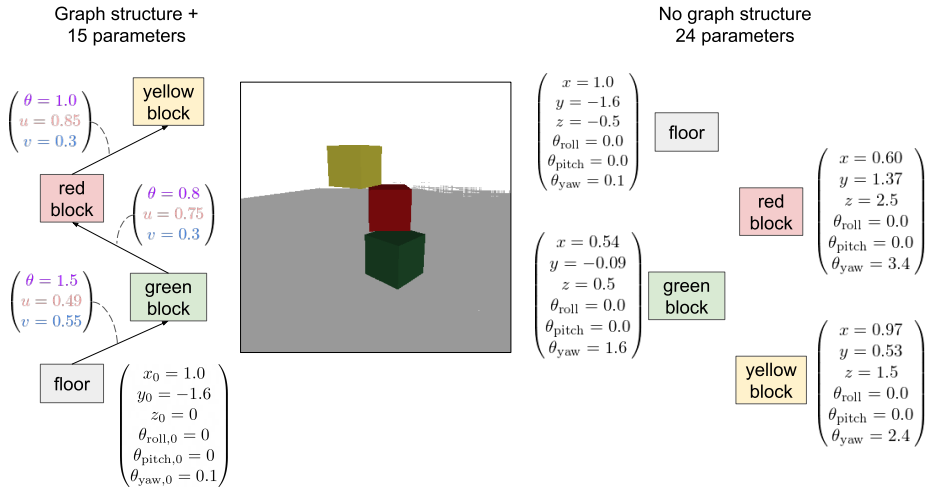

Figure 3: Comparison of the parameter spaces for two scene graphs that describe the same scene (the `frame` root node is ommited in both cases). Right: A scene graph with depth 1. The continuous parameters are 24-dimensional, comprising a full 6D pose for each object. Left: A scene graph with depth 2. The yellow cube is in face-to-face contact with the red cube, the red with the green, and the green with the floor. There are 15 continuous parameters, comprising a 6DOF pose of the floor object and 3 parameters for each contact relation. (Shape parameters of cubes ommited in both cases for simplicity). Figure 2 describes the face-to-face contact parametrization for pairs of cubes.

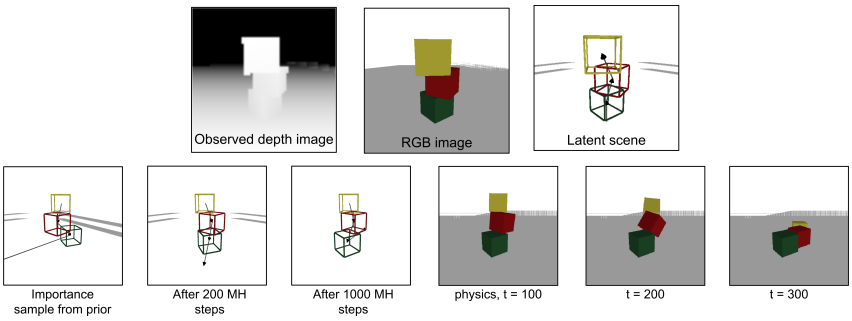

Figure 4: Inference of latent scene parameters conditioned on a specific contact graph structure, and deterministic forward-simulation of physics starting with the inferred scene. The observation is a low-resolution synthetic depth image generated from the prior. The inference uses block Metropolis–Hastings on the contact parameters on the parameters assigned to a single edge of the graph at a time, and the initial state is chosen by importance sampling from the prior (100 particles). Physical simulation then allows us to make physical inferences about the original scene, including qualitative inferences such as whether the tower is stable.

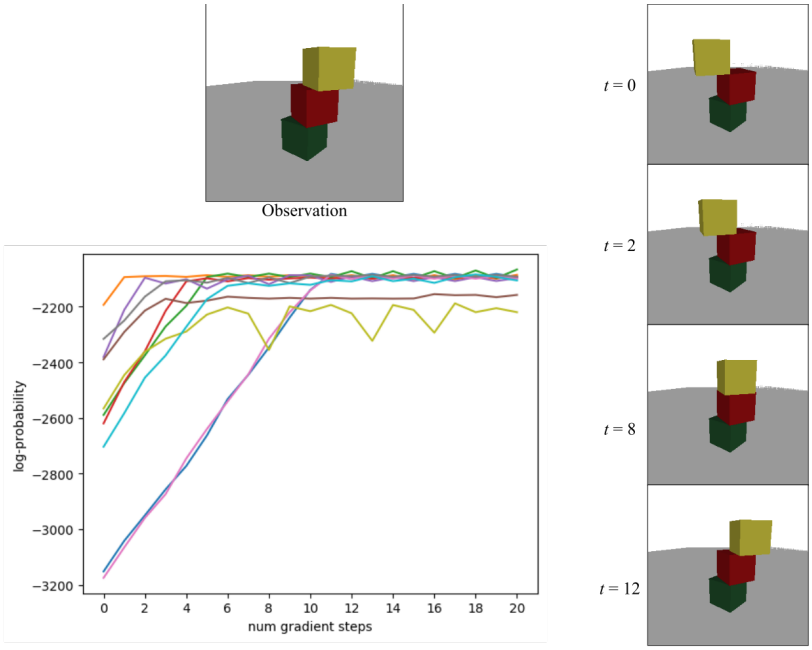

Figure 5: Inference of the continuous contact paremeters of the yellow cube relative to the red cube. Top-left: Observed image (we show RGB here for visual clarity; the input was actually a depth image). Right: A rendering of the inferred parameters after t gradient steps, for $t = 0, 2, 8, 12$, starting from a randomly initial state; we see convergence after 12 steps. Bottom-left: Plot of log-likelihood of the current state versus number of inference steps, for 10 random initializations of gradient ascent. Note that every state in each gradient ascent trajectory satisfies the qualitative constraint that the cubes are stacked in flush contact with each other, which is encoded explicitly in the scene graph $\mathcal{G}$ (which is fixed in this scenario). We differentiate the log-likelihood of the continuous parameters $(\theta, u, v) \in [0, \pi/2] \times [0, 1] \times [0, 1]$ as described in Figure 2 under the model in Equation (6), with a uniform prior, using a fixed step size $\alpha = 0.05$.