Multi-Object Manipulation via Object-Centric Neural Scattering Functions

Stephen Tian[1*] Yancheng Cai[1*] Hong-Xing Yu[1] Sergey Zakharov [2]
Katherine Liu[2] Adrien Gaidon[2] Yunzhu Li [1] Jiajun Wu[1]
[1]Stanford University [2] Toyota Research Institute

Abstract

Learned visual dynamics models have proven effective for robotic manipulation tasks. Yet, it remains unclear how best to represent scenes involving multi-object interactions. Current methods decompose a scene into discrete objects, but they struggle with precise modeling and manipulation amid challenging lighting conditions as they only encode appearance tied with specific illuminations. In this work, we propose using object-centric neural scattering functions (OSFs) as object representations in a model-predictive control framework. OSFs model per-object light transport, enabling compositional scene re-rendering under object rearrangement and varying lighting conditions. By combining this approach with inverse parameter estimation and graph-based neural dynamics models, we demonstrate improved model-predictive control performance and generalization in compositional multi-object environments, even in previously unseen scenarios and harsh lighting conditions.

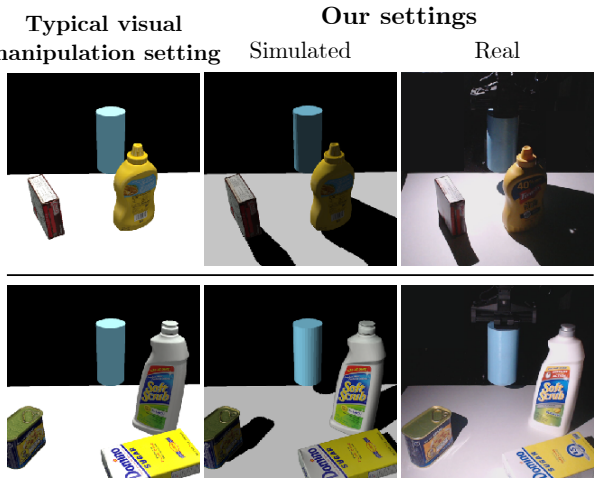


Figure 1. While typically studied visual manipulation settings are carefully controlled environments, we consider scenarios with varying and even harsh lighting, in addition to novel object configurations, that are more similar to real-world scenarios.

1. Introduction

Predictive models are the core components of many robotic systems for solving inverse problems such as planning and control. Physics-based models built on first principles have shown impressive performance in domains such as drone navigation [3] and robot locomotion [28]. However, such methods usually rely on complete *a priori* knowledge of the environment, limiting their use in complicated manipulation problems where full-state estimation is complex and often impossible. Therefore, a growing number of approaches alternatively propose to learn dynamics models directly from raw visual observations [2, 12, 14, 17, 21, 48].

Although using raw sensor measurements as inputs to predictive models is an attractive paradigm as they are readily available, visual data can be challenging to work with directly due to its high dimensionality. Prior methods proposed to learn dynamics models over latent vectors, demonstrating promising results in a range of robotics tasks [17, 18, 44, 52]. However, with multi-object interactions, the underlying physical world is 3D and compositional. Encoding everything into a single latent vector fails to consider the relational structure within the environment, limiting its generalization outside the training distribution.

Another promising strategy is to build more structured visual representations of the environment, including the use of particles [31, 33, 36], keypoints [32, 37, 38], and object meshes [22]. Among the structured representations, Driess *et al*. [10] leveraged compositional neural implicit representations in combination with graph neural networks (GNNs) for the dynamic modeling of multi-object interactions. The inductive bias introduced by GNNs captures the environment's underlying structure, enabling generalization to scenarios containing more objects than during training, and the neural implicit representations allow precise estimation and modeling of object geometry and interactions. However, Driess *et al*. [10] only considered objects of uniform color

*indicates equal contribution. Yancheng is affiliated with Fudan University; this work was done while he was a summer intern at Stanford.

in well-lit scenarios. It is unclear how the method works for objects with more complicated geometries and textures. The lack of explicit modeling of light transport also limits its use in scenarios of varying lighting conditions, especially those vastly different from the training distributions.

In this paper, we propose to combine object-centric neural scattering functions (OSFs) [57] and graph neural networks for the dynamics modeling and manipulation of multi-object scenes. OSFs explicitly model light transport and learn to approximate the cumulative radiance transfer, which allows relighting and inverse estimation of scenes involving multiple objects and the change of lights, such as those shown in Figure 1. Combined with gradient-free evolutionary algorithms like covariance matrix adaption (CMA), the learned neural implicit scattering functions support inverse parameter estimation, including object poses and light directions, from visual observations. Based on the estimated scene parameters, a graph-based neural dynamics model considers the interactions between objects and predicts the evolution of the underlying system. The predictive model can then be used within a model-predictive control (MPC) framework for downstream manipulation tasks.

Experiments demonstrate that our method performs more accurate reconstruction in harsh lighting conditions compared to prior methods, producing higher-fidelity long horizon prediction compared to video prediction models. When combined with inverse parameter estimation, our entire control pipeline improves on simulated object manipulation tasks in settings with varying lighting and previously unseen object configurations, compared to performing MPC directly in image space.

We make three contributions. First, the use of neural scattering functions supports inverse parameter estimation in scenarios with challenging and previously unseen lighting conditions. Second, our method models the compositionality of the underlying scene and can make long-term future predictions about the system's evolution to support downstream planning tasks. Third, we conduct and show successful manipulation of simulated multi-object scenes involving extreme lighting directions.

2. Related Work

Implicit object models for robotic control. Neural radiance fields (NeRFs) have emerged as powerful implicit models of scene [39] and object [25, 46, 54, 56] appearance. In robotics, NeRFs have been applied to tackle manipulation problems such as grasp and rearrangement planning [8, 23, 27, 42], determining constraints and collisions [1, 16], learning object descriptors for manipulating object poses [45], and system identification and trajectory optimization [6]. They have also been used as decoders in latent representation learning for model-free [11] and model-based [30] reinforcement learning. While these prior methods have explored the use of NeRFs for robotic manipulation, they train implicit models that are either restricted to baked-in lighting settings determined at training time or model the entire scene together; therefore, these methods cannot handle compositionality. In this work, we aim to leverage object-centric, *relightable* neural scattering functions in a model-based planning framework.

Pose estimation from images. Many prior works investigate the problem of estimating rigid object poses from RGB images, including deep-learning approaches based on correspondences [24, 35, 40, 41, 47, 58] as well as direct regression [9, 13, 29, 53, 59]. While achieving impressive results, these methods require colored 3D object meshes to train on and are sensitive to changing lighting conditions. Yen-Chen *et al*. [55] and Jang *et al*. [25] study the problem of estimating the camera pose of a given image using neural fields. In this work, we propose estimating both object poses **and** lighting conditions from RGB images by "inverting" implicit OSF models.

Visual planning with learned dynamics models. Learned dynamics models are a powerful tool for performing planning. When the agent receives observations in 2D images, one class of prior works directly models dynamics in image space [14]. Another approach is to learn a keypoint [38] or latent state representation via reconstruction by a convolutional neural network [15, 19] or volumetric rendering [30]. Alternatively, scenes can be represented explicitly as discrete objects with learned decoding or rendering modules [26, 49]. While our approach also explicitly models a scene as a set of discrete objects, we perform inverse pose estimation to acquire the 6D pose of each object in the scene. Combined with learned OSFs, we can compose objects to render the scene.

3. Problem Definition

Given a set of N known objects with pre-trained OSFs models, a goal image depicting the desired configuration of the objects I_{goal}, and RGB camera observations of the scene from V different viewpoints at each timestep t, denoted $I_t^{1:V}$, the objective is to execute a sequence of actions $a_{0:T}$ such that the goal object configuration is achieved at the end of T steps. Each action in the action sequence is defined as a 3D change in position for a cylindrical pusher object that is present in all scenes, but could be generalized to represent many types of robot end-effectors. We assume access to the camera parameters for each observation and the goal image, but *not* the lighting parameters.

While we assume access to pre-trained OSFs models for each object, we note that multi-view images of each object with paired light pose information are sufficient supervision for training OSFs. We do not require access to object

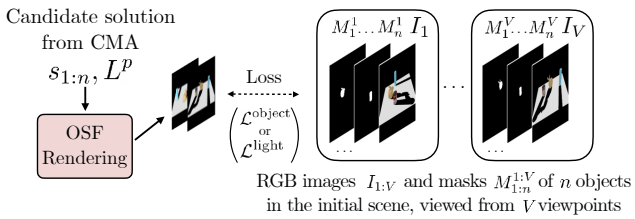


Figure 2. Pose estimation for the light and objects. Given multi-view images of the initial static scene and the binary masks of each object, we use CMA [20] to estimate the initial 6D pose (position and quaternion) of each object and the 3D position of the light source. Unlike previous methods, we explicitly model the lighting to better recover poses in novel lighting directions.

meshes. In this work, we constrain ourselves to handle only rigid objects for simplicity.

4. Methods

To solve tabletop visual object manipulation tasks, we want our model to have two properties:

1. It should have a compact, low-dimensional representation of the scene, which allows learned dynamics models to perform more stable long-horizon prediction.
2. It should be capable of handling scenarios with different object configurations and previously unseen, unknown lighting.

We achieve the first property by representing scenes using the 6D pose of each object. Then, we achieve the second by implicitly modeling each object using object-centric neural scattering functions (OSFs). OSFs model the light transport for a particular object – specifically, given the spatial location and incoming and outgoing light direction, they predict the object's radiance transfer from the light source to the viewer. This enables relighting as well as composition during the rendering process.

Our method consists of three main steps. The first is an offline optimization phase, where we train a dynamics model using simulated data that takes as input 6D object poses of each object in the scene and the pusher's action, and outputs future 6D poses for each object. Then, at inference time, we use inverse parameter estimation with OSFs models to estimate the scene's initial object poses and the light position. Finally, we use the learned dynamics model to perform model-predictive control, updating object states at each step using the same inverse parameter estimation strategy. We introduce each component of our method in the following sections and present a summary in Algorithm 1.

4.1. Neural Implicit Scattering Functions

Since we use object poses to represent the scene, we need to estimate them from visual observations. To this end, we *invert* the rendering process of the scene. Since we would like our estimation approach to generalize to any configuration of objects, we adopt an object-centric, composable implicit model. Specifically, we use the object-centric neural scattering functions (OSFs) [57] to represent each object.

OSFs are relightable, compositional implicit neural rendering models based on neural radiance fields (NeRFs) [39]. To achieve relighting, OSFs learn to approximate the cumulative radiance transfer from a distant light in addition to learning the spatial volume density as NeRFs do:

$$f_\theta : (\mathbf{x}, \boldsymbol{\omega}_{\text{light}}, \boldsymbol{\omega}_{\text{out}}) \rightarrow (\rho, \sigma), \tag{1}$$

where $\mathbf{x}$ denotes the 3D spatial location, σ denotes the spatial volumetric density, $\boldsymbol{\omega}_{\text{out}}$ denotes the outgoing radiation direction, $\boldsymbol{\omega}_{\text{light}}$ denotes the distant light direction and f_θ denotes a learnable deep neural network. $\rho(\mathbf{x}, \boldsymbol{\omega}_{\text{light}}, \boldsymbol{\omega}_{\text{out}})$ denotes the cumulative radiance transfer function. The scattered outgoing radiance L_{out} can then be formulated as

$$L_{\text{out}}(\mathbf{x}, \boldsymbol{\omega}_{\text{out}}) = \int_{S^2} \rho(\mathbf{x}, \boldsymbol{\omega}_{\text{light}}, \boldsymbol{\omega}_{\text{out}}) L_{\text{light}}(\boldsymbol{\omega}_{\text{light}}) d\boldsymbol{\omega}_{\text{light}}, \tag{2}$$

where S^2 denotes the unit sphere for integrating solid angles, and L_{light} denotes the radiance of the distant light.

To accelerate the rendering of OSFs, we use a variant called KiloOSFs [57]. KiloOSFs extend the idea of KiloNeRFs [43], which represent a static scene as thousands of small independent MLPs, to the object-centric setting. For each pixel depicted by a ray $\mathbf{r}(t) = \mathbf{o} - t\boldsymbol{\omega}_{\text{out}}$, we use compositional volumetric rendering for the scene:

$$L(\mathbf{o}, \boldsymbol{\omega}_{\text{out}}) = \int_{t_n}^{t_f} T(t) \sigma^{\text{s}}(\mathbf{r}(t)) L_{\text{out}}^{\text{s}}(\mathbf{r}(t), \boldsymbol{\omega}_{\text{out}}) dt, \tag{3}$$

where $T(t)$ denotes the accumulated transmittance along the ray from the near plane t_n to the far plane t_f, σ^{s} denotes the sum of density from all objects in the scene, and $L_{\text{out}}^{\text{s}}$ denotes the sum of radiance from all objects. For inference speed, we render shadows with shadow mapping [51].

4.2. Inverse Parameter Estimation

To increase the stability of long-term dynamics predictions, we use coordinate space pose vectors (rather than latent vectors) to represent objects compactly. Following Driess et al. [10], we assume access to RGB images of the initial scene $I_i \in \mathbb{R}^{H \times W \times 3}$, where $i = 1, ..., V$ represents the index into V camera views, as well as binary masks $M_j^i \in \{0, 1\}^{H \times W}$ of each object j in view i. We also assume that we already have KiloOSF models $O_j, j = 1, ..., n$

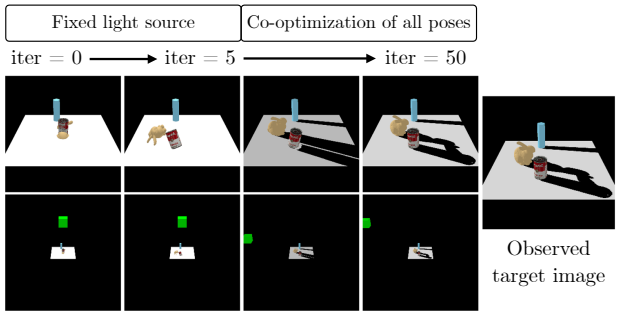


Figure 3. Demonstration of inverse parameter estimation. The top and bottom rows show the **same** tabletop setting from close and far views, respectively. The green cube represents the light position, which always points at the plane's center. In this figure, for the first five iterations, we fix the light pose directly overhead and optimize all object poses. When object poses are approximately optimized, we estimate all object poses and the light pose together.

for each object in the scene. We refer to the KiloOSF models for the compositional rendering of *multiple* objects (including shadows) as $O^{all}_{1:n}$. For the rest of this section, we use the notation $O_j(v, s, L)$ to indicate rendering object j with pose s from viewpoint v with directional light pose L.

We use covariance matrix adaptation (CMA) [20], represented by Ψ, to optimize the 6D poses of each of n objects $s_{1:n} \in \mathbb{R}^{n\times 7}$ and light position $L^p \in \mathbb{R}^3$:

$$s_{1:n}, L^p = \Psi\left(I_{1:V}, M^{1:V}_{1:n}, O_{1:n}\right), \tag{4}$$

as shown in Figure 2. For readability, we write Equation 4 as a function of OSFs $O^{all}_{1:n}$; in practice, the OSFs are used to define the loss functions as described below.

For object pose optimization when given object masks, we use the mean-squared error (MSE) of V multi-view KiloOSF-rendered images of the object and the observed multi-view images $I_{1:V}$, masked by binary masks for each object $M^{1:V}_j$. We initialize the light direction L_{init} as the unit vector in the z-direction. Specifically,

$$\mathcal{L}^{\text{object}}_j = \sum_{v\in V} ||O_j(v, s_j, L_{init}) - (I_v \circ M^v_j)||^2_2. \tag{5}$$

For light pose optimization, we compositionally render the entire scene, including shadows, with KiloOSF $O^{all}_{1:n}$ and compute the MSE with the observed images:

$$\mathcal{L}^{\text{light}} = \sum_{v\in V} ||(O^{all}_{1:n}(v, s_{1:n}, L^p), -I_v||^2_2. \tag{6}$$

Figure 3 shows a visualization of the process.

4.3. Action-Conditioned Dynamics Model

During dynamics prediction, we are given object poses and an action taken by an agent in the scene, and aim to infer subsequent states. Concretely, we train a graph neural network (GNN) dynamics model to make predictions of the future object 6D poses:

$$s^{t+1}_{1:n} = f_{GNN}(s^t_{1:n}, A^t, a^t) \in \mathbb{R}^{n\times 7}, \tag{7}$$

where $A^t \in \{0,1\}^{n,n}$ is the adjacency matrix, $a^t \in \mathbb{R}^3$ is the input action, and $s^t_{1:n} \in \mathbb{R}^{n\times 7}$ are poses of all objects at time t. To model multi-object interactions, we follow Li *et al*. [34] and perform multiple inter-object propagation steps during the prediction for a single future time step.

We use each node in a graph to represent a single object, with its pose s^t as its input node feature. Following [10], we dynamically create graph edges to improve the stability of long-term predictions. We only construct edges between nodes if the objects represented by those nodes can potentially collide during the current timestep. We represent edge information using an adjacency matrix A^t. Because we are working in coordinate space, we use an approximated distance between two objects to determine potential collisions by leveraging the geometric information present in KiloOSF models. We first construct approximate point clouds by evaluating the KiloOSF's density on a grid of points and thresholding points above a certain density. Then, we use the longest axis of the tightest bounding box containing the entire pointcloud as the threshold distance κ for edge creation. Note that the creation of an edge does not mean that the model *must* predict a collision, but rather that it *can*.

$$A^t_{ij} = \begin{cases} 1 & ||s^t_i - s^t_j||^2_2 < \kappa, i \neq j \\ 0 & \text{else} \end{cases} \tag{8}$$

We use a shared edge encoder $\mathbb{E}^{\text{edge}}$ and propagator $\mathbb{P}^{\text{edge}}$ regardless of which objects are present in the scene. This greatly enhances the generalization ability of GNN to handle unseen scenarios while using more training data. The detailed structure of the GNN is shown in the Appendix.

We train the GNN dynamics model using a mean squared-error loss between the predicted and ground truth object poses, each represented by a position $p \in \mathbb{R}^3$ and unit quaternion $q \in \mathbb{R}^4$, summed across all objects in the scene. Dynamics models trained on single-step prediction are prone to compounding errors during longer open-loop model rollouts. Thus, we train the model on predictions of up to 3 future timesteps.

4.4. Visual Model-Predictive Control

Given a goal image and initial visual observations of an environment, the objective of visual model-predictive control is to optimize a robot action sequence to reach the goal.

After estimating the light position and each object's initial pose, we perform sampling-based planning using our learned dynamics model. At timestep t, we first sample K

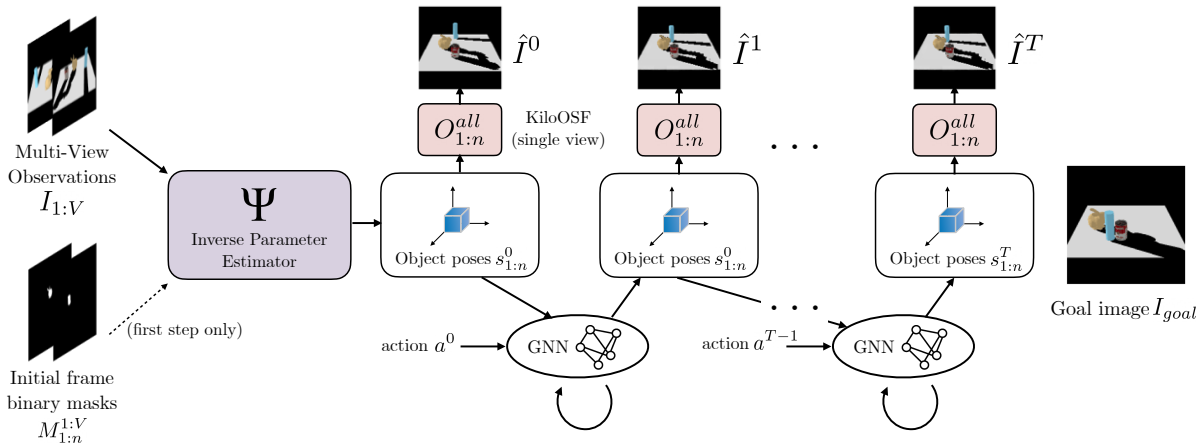


Figure 4. Combined framework for perception (inverse parameter estimation) and prediction (dynamics model). The figure shows the r-th replanning step in model-predictive control. In the first step, we use the multi-view images $I_{1:V}$ and the object's binary masks $M^{1:V}_{1:n}$ for inverse parameter estimation. Afterward, only multi-view images are used. We only render the predicted images $\hat{I}^{t:t+H}$ using KiloOSF with the *goal image view*, and then compare the sequence of images $\hat{I}^{t:t+H}$ with the goal image I_{goal} via an MSE loss.

action sequences of length H. By rolling out open-loop predictions from the GNN dynamics model, we obtain the predicted future poses of each object $\hat{s}^{t:t+H}_{1:n}$ for each sample. We then provide these objects and light poses to KiloOSF to render RGB images $\hat{I}^{t:t+H}$. We use the squared ℓ_2 error between rendered images and the goal image, that is, $\sum_{\tau=t}^{t+H} ||\hat{I}^{\tau} - I_{goal}||_2^2$ as the cost function to score each action sequence sample as in visual foresight [12]. We then use MPPI [50] to update the action sampling distribution based on the action samples and scores. The first step of the action distribution's mean is executed in the environment.

Then, the object pose estimates $s^{t+1}_{1:n}$ are updated again using inverse parameter estimation using new observations from the environment $I_{1:V}$. To reduce computational cost, we decrease the search space for inverse parameter estimation at this step by reducing the initial standard deviation of CMA compared to the initial step. We also alleviate the assumption of object masks past the initial frame, and compute the loss for the CMA optimizer as the mean-squared error (MSE) between rendered and observed RGB frames.

The process is repeated to replan every r time steps. Algorithm 1 outlines the entire planning procedure for $r = 1$, and Figure 4 visualizes a single (re-)planning step.

Algorithm 1 Visual MPC with OSFs and GNN dynamics

Input: Pre-trained KiloOSF models $O_{1:n}(O^{all}_{1:n})$, initial environment image observation $I_{1:V}$, initial object masks $M^{1:V}_{1:n}$, goal image I_{goal} and viewpoint v_{goal}, GNN dynamics model f_{GNN}, planning horizon H

for $t = 0...T$ **do**
  if $t = 0$ **then**
    // Perform initial object & light pose estimation
    $s^0_{1:n}, L^p = \Psi\left(I_{1:V}, M^{1:V}_{1:n}, O_{1:n}\right)$
  else
    // Refine object pose estimates without masks
    $s^t_{1:n} = \Psi\left(I_{1:V}, O^{all}_{1:n}\right)$
  end if
  Sample K random action sequences $a^{t:t+H}_{1:K}$.
  for action sample $a^{t:t+H}$ in $a^{t:t+H}_{1:K}$ **do**
    // Iteratively predict H future states with the action-conditioned GNN dynamics model.
    $\hat{s}^{t:t+H}_{1:n} = f_{GNN}(s^t_{1:n}, a^{t:t+H})$.
    // Use KiloOSF to render image predictions from the goal image viewpoint.
    $\hat{I}^{t:t+H} = O^{all}_{1:n}(v_{goal}, \hat{s}^{t:t+H}_{1:n}, L^p)$.
    // Compute loss for each sampled action sequence.
    $L = \sum_{\tau=t}^{t+H} ||\hat{I}^{\tau} - I_{goal}||_2^2$.
  end for
  Update action sampling distribution via MPPI [50].
  Execute the first step of the mean of the updated action sampling distribution.
  Receive a new environment observation $I_{1:V}$.
end for

5. Experiments

In this section, we empirically evaluate the performance of our method and its components. Specifically, we seek to answer the following questions:

1. How well can KiloOSF reconstruct scenes with composable objects in harsh lighting conditions?
2. Does the combination of KiloOSF and the GNN dynamics model enable long-horizon visual dynamics prediction?
3. How does our proposed method compare to existing methods for performing robotic manipulation using visual model-predictive control?

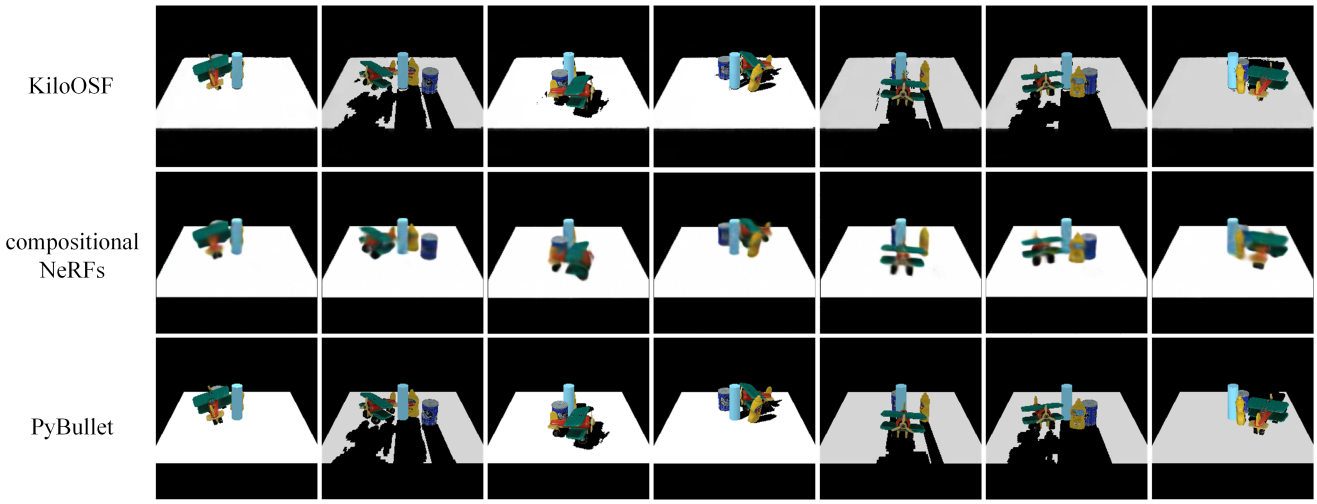


Figure 5. Qualitative comparison of the reconstruction results of KiloOSF compared to compositional NeRFs [10]. The first row shows images rendered by our KiloOSF, and the second row shows the visual reconstruction of compositional NeRFs. The third row shows the ground truth image rendered directly with PyBullet. Our method can reasonably render the color change of the object due to the change of lighting pose, such as the plane becoming darker when the light is tilted. At the same time, our method can render accurate shadows.

4. How well can our method generalize to performing control in settings unseen at training time?
5. Can our lighting and pose estimation methods be applied to real-world settings?

5.1. Experimental Setting

We focus on settings with particularly harsh lighting conditions to determine whether our method can still perform manipulation when lighting causes significant appearance changes in the scene. We use the PyBullet simulator [7] to perform experiments. We consider a tabletop manipulation setting, shown in Figure 3. This consists of a cylindrical pusher, representing the end-effector of a robot, one or more objects to be manipulated, and four camera viewpoints. Compared to previous methods, our object representations can handle objects with finer textures. Therefore, we use a subset of 20 YCB objects [4,5] in our experiments.

To test the generalization ability of our model, we train our dynamics model in scenes with only three objects, along with the pusher. However, during testing, in addition to the three-object setting, we also test on scenes with two and four objects, as described in Section 5.5.

5.2. Visual Reconstruction

Multi-object dynamic interaction models, such as graph neural networks, are effective for making predictions in coordinate or latent space rather than in image space. However, inferring these low-dimensional states from images, for example through inverse parameter estimation, requires faithful image reconstruction from low-dimensional states.

Prior work [10] uses compositional neural radiance fields (compositional NeRFs) to support the rendering of multi-object composite scenes. For this experiment, we compare KiloOSF reconstructions to those of compositional NeRFs.

We collect 1000 trajectories of simulated object data to train the compositional NeRF models and our dynamics model. In each trajectory, three YCB objects are randomly selected to be placed into the scene. The cylinder is then moved randomly for 50 steps with actions drawn from a uniform distribution $[\mathcal{U}(-0.3, 0.3), \mathcal{U}(-0.3, 0.3), 0]$, representing the speed (m/s) at which the cylinder moves. Each action is applied for roughly 0.2 seconds.

In Figure 5, we show qualitative comparisons between the reconstructions generated by our KiloOSF and by compositional NeRFs. Although compositional NeRFs can render multi-object scenes, they can suffer, for example, when the scene's lighting changes significantly from that experienced during training. Additionally, compositional NeRFs cannot fit high-frequency information such as fine textures and complex geometry as well as our KiloOSF model.

5.3. Visual Prediction

Next, we evaluate the visual prediction performance of our combined GNN dynamics and KiloOSF rendering modules. We compare to a state-of-the-art stochastic variational video prediction model, FitVid [2], that takes RGB images as input and makes predictions directly in pixel space. For fairness, we modify the FitVid model to make predictions at 128×128 resolution instead of the default 64×64. We then train FitVid on the same dataset of 1000 trajectories that we use to train our dynamics model, randomizing the lighting present in each trajectory to improve its performance.

We visualize qualitative prediction results in Figure 6. We see that over longer prediction horizons past around 10

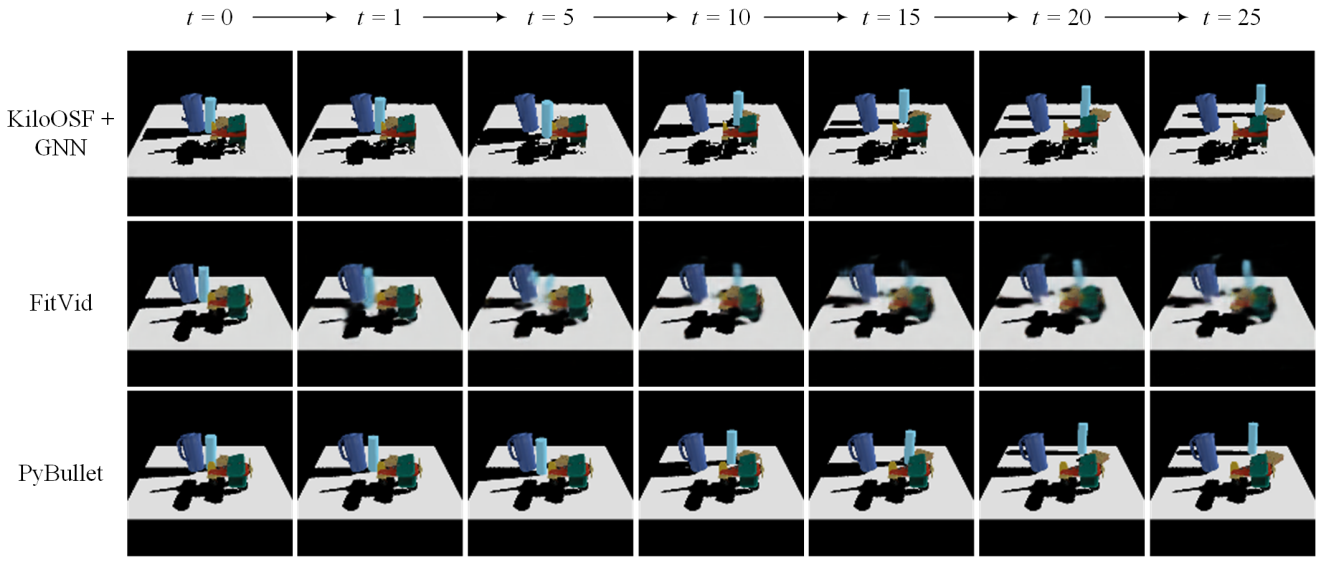


Figure 6. Qualitative comparison of the forward prediction with our method compared to FitVid, a video prediction model. Our method achieves much better prediction results than FitVid, even though FitVid is trained on a dataset that already contains multiple lighting poses and shadows. The third row shows the ground truth images for the trajectory from the PyBullet simulator.

predicted steps, the prediction quality of FitVid quickly deteriorates. This is because small errors in the pixel predictions of the model accumulate rapidly. By separating the dynamics and rendering model, our GNN dynamics model can make long-horizon predictions in low-dimensional coordinate space to be later rendered using KiloOSF, without sacrificing appearance quality.

5.4. Model Predictive Control

In this section, we evaluate the performance of our method as a full model-predictive control pipeline. This includes performing inverse object pose and lighting estimation, planning and scoring action sequences, and updating estimated parameters in a closed-loop fashion.

We evaluate our method as well as visual foresight [14] with FitVid as the visual dynamics model on 20 different testing tasks, each with randomized combinations of three objects and a randomized, unknown light pose sampled in a hemisphere above the ground plane. Each goal image specifies moving one or more objects at a distance of at least 0.075m from their initial location. This is a challenging setting because the randomized lighting conditions can create harsh lighting conditions, such as those shown in Figure 6.

We execute model-predictive control for 30 steps for each method, and report the performance in terms of the final 6D object pose error (Euclidean distance from ground truth goal pose) over all objects. The results are presented in Figure 7. By inferring the lighting parameters in each trial, our model can better predict the underlying scene dynamics, improving planning performance.

5.5. Generalization to Unseen Scenarios

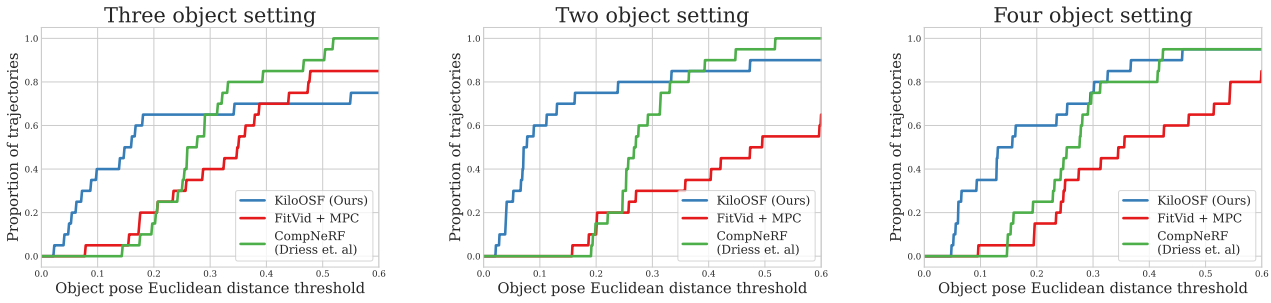
Furthermore, we evaluate how our method generalizes to previously unseen scenarios. We consider testing scenes that contain two or four objects, while during training, there are always three objects present. In Figure 7, we present the control performance of the same model across these settings. Because our method factorizes the scene into individual object representations and uses a graph neural network dynamics model for object interactions, it naturally handles this case. For models trained on visual observations of the entire scene, these settings are out of the training distribution. For computational reasons, we perform evaluation while providing object masks at all steps, but we find that our method achieves similar performance in the three-object setting with and without full mask information.

5.6. Real World Lighting and Pose Estimation

Finally, we demonstrate our lighting and pose estimation pipelines with stimuli from a real scene. We collect images in real-world harsh lighting settings created by positioning a single spotlight and blocking external light from the scene.

For light optimization, we optimize not only the light source position as in the simulated experiments, but also the light distance, "look at" direction, and intensity. In the top of Figure 8, we show qualitative examples of the light poses optimized by our method. We see that our method is able to recover a reasonable estimate of the lighting and creates shadows similar to those in the scene.

For real-world object pose optimization, we initialize pose estimates for the optimization process using a pre-trained PoseCNN [53]. This provides a helpful but imper-



(a) Three object (training) scenario.

(b) Two and four object (unseen) scenarios.

Figure 7. Quantitative results of our method compared to running visual MPC with FitVid [2] and Compositional NeRF [10] in experimental settings with two, three, and four objects respectively. The horizontal axis represents a range of loss thresholds. The vertical axis represents the proportion of experimental trials where models rearrange the objects within the given error threshold. After 30 steps of MPC, we compute the Euclidean distance between the 6D object poses at the final step. Our method outperforms the pixel-space MPC and Compositional NeRF-based methods in all three experimental settings.

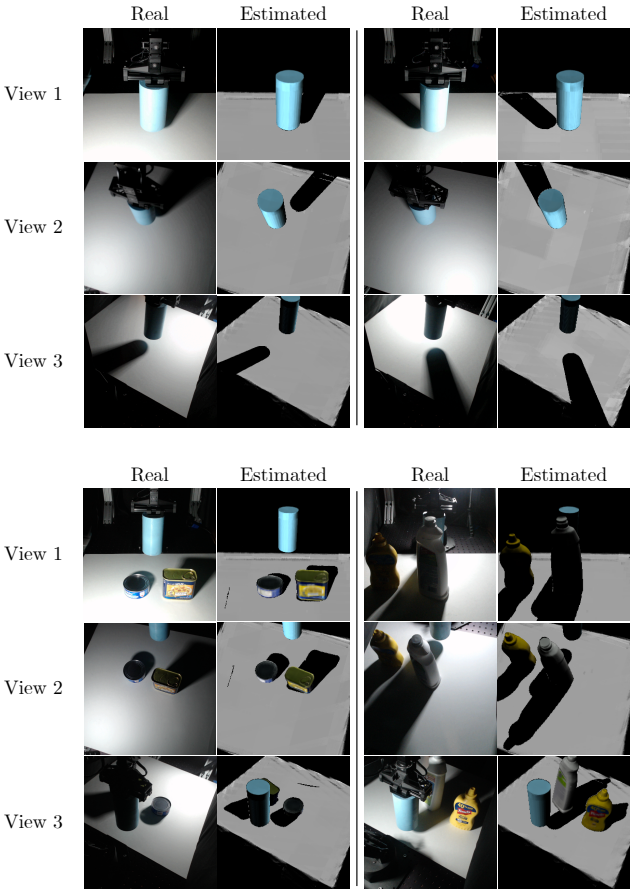


Figure 8. (Top): Lighting estimates produced by inverse parameter estimation from real images. Shadows are well-approximated by our method. (Bottom): Visualized object pose estimates. It is challenging to perfectly optimize the orientation of the bleach bottle due to local minima in appearance-based objectives.

fect initialization that our inverse parameter estimation procedure finetunes. We compute the loss as an intersection-over-union between object masks and masks from the compositional KiloOSF rendering process. In the bottom of Figure 8, we show examples of real observations along with rendered scenes containing objects with poses estimated by our method. The lighting parameters are also determined using our method. Our method is able to reconstruct challenging scenes with reasonable fidelity.

6. Conclusion

We have presented a method that uses object-centric dynamics modeling for robotic manipulation in scenes with unseen object configurations and harsh lighting. We have demonstrated that by leveraging object-centric neural scattering functions, we can invert the rendering procedure to determine object poses and lighting information. This makes our method adaptable to harsh lighting settings and enables us to combine it with a learned neural network dynamics model for use in model-predictive control on long horizon control tasks. We show through experiments that our method achieves better reconstruction in harsh lighting scenarios than previous implicit modeling strategies and improved long horizon prediction and model-predictive control performance compared to video prediction models.

Limitations. Currently, our method requires separate KiloOSF models to be trained for each object to be manipulated. Although advancements in training NeRFs efficiently may accelerate this process, this may be time-consuming for large numbers of real objects. We use a simple lighting model; more complex lighting interactions may help model in-the-wild conditions. Additionally, in this work we only consider rigid object manipulation.

Acknowledgments. This work is in part supported by NSF RI #2211258, ONR MURI N00014-22-1-2740, AFOSR YIP FA9550-23-1-0127, the Stanford Institute for Human-Centered AI (HAI), the Toyota Research Institute (TRI), Amazon, Ford, Google, and Qualcomm. ST is supported by NSF GRFP Grant No. DGE-1656518.

References

[1] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-Only Robot Navigation in a Neural Radiance World. *IEEE Robotics and Automation Letters (RA-L)*, 7(2), 2022. 2

[2] Mohammad Babaeizadeh, Mohammad Taghi Saffar, Suraj Nair, Sergey Levine, Chelsea Finn, and Dumitru Erhan. Fitvid: Overfitting in pixel-level video prediction. *arXiv preprint arXiv:2106.13195*, 2021. 1, 6, 8

[3] Pierre-Jean Bristeau, François Callou, David Vissière, and Nicolas Petit. The navigation and control technology inside the ar. drone micro uav. *IFAC Proceedings Volumes*, 44(1):1477–1484, 2011. 1

[4] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The YCB object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 6

[5] 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. 6

[6] Simon Le Cleac'h, Hong-Xing Yu, Michelle Guo, Taylor A Howell, Ruohan Gao, Jiajun Wu, Zachary Manchester, and Mac Schwager. Differentiable physics simulation of dynamics-augmented neural objects. *arXiv preprint arXiv:2210.09420*, 2022. 2

[7] Erwin Coumans and Yunfei Bai. PyBullet, a python module for physics simulation for games, robotics and machine learning. 2016. 6

[8] Michael Danielczuk, Arsalan Mousavian, Clemens Eppner, and Dieter Fox. Object rearrangement using learned implicit collision functions. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6010–6017, 2021. 2

[9] 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*, 37(5):1328–1342, 2021. 2

[10] Danny Driess, Zhiao Huang, Yunzhu Li, Russ Tedrake, and Marc Toussaint. Learning multi-object dynamics with compositional neural radiance fields, 2022. 1, 3, 4, 6, 8

[11] Danny Driess, Ingmar Schubert, Pete Florence, Yunzhu Li, and Marc Toussaint. Reinforcement learning with neural radiance fields. *arXiv preprint arXiv:2206.01634*, 2022. 2

[12] Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018. 1, 5

[13] Francis Engelmann, Konstantinos Rematas, Bastian Leibe, and Vittorio Ferrari. From points to multi-object 3d reconstruction. In *CVPR*, pages 4588–4597, 2021. 2

[14] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793, 2017. 1, 2, 7

[15] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. 2

[16] Jung-Su Ha, Danny Driess, and Marc Toussaint. Learning neural implicit functions as object representations for robotic manipulation. *arXiv preprint arXiv:2112.04812*, 2021. 2

[17] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. 1

[18] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019. 1

[19] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020. 2

[20] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cmaes). *Evolutionary Computation*, 11(1):1–18, 2003. 3, 4

[21] Ryan Hoque, Daniel Seita, Ashwin Balakrishna, Aditya Ganapathi, Ajay Kumar Tanwani, Nawid Jamali, Katsu Yamane, Soshi Iba, and Ken Goldberg. Visuospatial foresight for multi-step, multi-task fabric manipulation. *arXiv preprint arXiv:2003.09044*, 2020. 1

[22] Zixuan Huang, Xingyu Lin, and David Held. Mesh-based dynamics with occlusion reasoning for cloth manipulation. *arXiv preprint arXiv:2206.02881*, 2022. 1

[23] Jeffrey Ichnowski*, Yahav Avigal*, Justin Kerr, and Ken Goldberg. Dex-NeRF: Using a neural radiance field to grasp transparent objects. In *Conference on Robot Learning (CoRL)*, 2020. 2

[24] Omid Hosseini Jafari, Siva Karthik Mustikovela, Karl Pertsch, Eric Brachmann, and Carsten Rother. iPose: Instance-aware 6D pose estimation of partly occluded objects. In *ACCV*, pages 477–492. Springer, 2018. 2

[25] Wonbong Jang and Lourdes Agapito. CodeNeRF: Disentangled neural radiance fields for object categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12949–12958, 2021. 2

[26] Michael Janner, Sergey Levine, William T. Freeman, Joshua B. Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-oriented prediction and planning. In *International Conference on Learning Representations*, 2019. 2

[27] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-DoF grasp detection via implicit representations. *Robotics: science and systems*, 2021. 2

[28] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous robots*, 40(3):429–455, 2016. 1

[29] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *ECCV*, pages 574–591. Springer, 2020. 2

[30] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3D neural scene representations for visuomotor control. *arXiv preprint arXiv:2107.04004*, 2021. 2

[31] Yunzhu Li, Toru Lin, Kexin Yi, Daniel Bear, Daniel Yamins, Jiajun Wu, Joshua Tenenbaum, and Antonio Torralba. Visual grounding of learned physical models. In *International conference on machine learning*, pages 5927–5936. PMLR, 2020. 1

[32] Yunzhu Li, Antonio Torralba, Anima Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. *Advances in Neural Information Processing Systems*, 33:9180–9192, 2020. 1

[33] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 1

[34] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019. 4

[35] Zhigang Li, Gu Wang, and Xiangyang Ji. CDPN: Coordinates-based disentangled pose network for real-time rgb-based 6-DoF object pose estimation. In *ICCV*, pages 7678–7687, 2019. 2

[36] Xingyu Lin, Yufei Wang, Zixuan Huang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning*, pages 256–266. PMLR, 2022. 1

[37] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. KPAM: Keypoint affordances for category-level robotic manipulation. In *The International Symposium of Robotics Research*, pages 132–157. Springer, 2019. 1

[38] Lucas Manuelli, Yunzhu Li, Pete Florence, and Russ Tedrake. Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning. *arXiv preprint arXiv:2009.05085*, 2020. 1, 2

[39] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2, 3

[40] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6D pose estimation. In *ICCV*, pages 7668–7677, 2019. 2

[41] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *CVPR*, pages 4561–4570, 2019. 2

[42] Ahmed H Qureshi, Arsalan Mousavian, Chris Paxton, Michael C Yip, and Dieter Fox. NeRP: Neural rearrangement planning for unknown objects. *arXiv preprint arXiv:2106.01352*, 2021. 2

[43] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. 3

[44] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. 1

[45] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B. Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: SE(3)-equivariant object representations for manipulation. *arXiv preprint arXiv:2112.05124*, 2021. 2

[46] Cameron Smith, Hong-Xing Yu, Sergey Zakharov, Fredo Durand, Joshua B Tenenbaum, Jiajun Wu, and Vincent Sitzmann. Unsupervised discovery and composition of object light fields. *arXiv preprint arXiv:2205.03923*, 2022. 2

[47] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-Time Seamless Single Shot 6D Object Pose Prediction. In *CVPR*, 2018. 2

[48] Stephen Tian, Chelsea Finn, and Jiajun Wu. A control-centric benchmark for video prediction. In *International Conference on Learning Representations*, 2023. 1

[49] Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, pages 1439–1456, 2020. 2

[50] Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015. 5

[51] Lance Williams. Casting curved shadows on curved surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 270–274, 1978. 3

[52] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World models for physical robot learning. *arXiv preprint arXiv:2206.14176*, 2022. 1

[53] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. 2018. 2, 7

[54] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *International Conference on Computer Vision (ICCV)*, October 2021. 2

[55] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021. 2

[56] Hong-Xing Yu, Leonidas J. Guibas, and Jiajun Wu. Unsupervised discovery of object radiance fields. In *International Conference on Learning Representations*, 2022. 2

[57] Hong-Xing Yu, Michelle Guo, Alireza Fathi, Yen-Yu Chang, Eric Ryan Chan, Ruohan Gao, Thomas Funkhouser, and Jiajun Wu. Learning object-centric neural scattering functions for free-viewpoint relighting and scene composition. *arXiv preprint arXiv:2303.06138*, 2023. 2, 3

[58] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6D pose object detector and refiner. In *ICCV*, 2019. 2

[59] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2