A. Dynamic Scene Reconstruction

As mentioned in the submission, we collected a RGBD-pet dataset containing videos of a cat and a dog, captured by an iPad with RGBD sensor. We use the RGB stream for reconstruction. To evaluate the dynamic scene reconstruction accuracy, although one would want to use the complete scene geometry as ground-truth, it is difficult to obtain for in-the-wild dynamic scenes. Instead, we render the depth and evaluate against the depth from LiDAR sensors as a proxy.

Depth Metrics. Following Eigen et al. [2], we compute the root mean squared error (RMSE) for both rendered depth and disparity (inverse depth) maps. To find the unknown global scale factor, we align the median value of the rendered depth with the ground truth similar to Luo et al. [6]:

\[
s_i = \text{median}_x \left\{ D_i^{\text{pred}}(x) / D_i^{\text{ground-truth}}(x) \right\}.
\]

Table 1: Comparison of scene reconstruction on RGBD-pet. We report root-mean-square-error (RMSE, ↓) on rendered depth and disparity (inverse depth) maps, averaged over all frames. DPT-omnidata [1, 8] trains transformer-based depth predictors on a mix of multiple depth datasets. BANMo∗ [11] applies differentiable rendering to reconstruct deformable objects, and we follow Neu-Man [4] to fit the object scale to a ground plane. PPR out-performs DPT-omnidata on the cat sequence, and out-performs BANMo∗ on both sequences.

<table>
<thead>
<tr>
<th>Method</th>
<th>cat</th>
<th>dog</th>
<th>cat</th>
<th>dog</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>depth</td>
<td>disparity</td>
<td>depth</td>
<td>disparity</td>
</tr>
<tr>
<td>DPT-omnidata</td>
<td>0.620</td>
<td>0.201</td>
<td>0.165</td>
<td>0.027</td>
</tr>
<tr>
<td>BANMo∗</td>
<td>0.181</td>
<td>0.149</td>
<td>0.232</td>
<td>0.061</td>
</tr>
<tr>
<td>PPR</td>
<td>0.179</td>
<td>0.139</td>
<td>0.216</td>
<td>0.041</td>
</tr>
</tbody>
</table>

Results. The results are shown in Tab. 1. We first interpret the results of DPT-omnidata. Leveraging depth priors learned from large-scale training data, DPT-omnidata performs very well for the dog sequence. However, it fails to produce accurate depth estimates for the cat sequence, possibly due to the uncommon top-down view angle of the video. PPR produces much better results on the cat sequence because it relies on multiview constraints that is more robust than depth priors. BANMo with ground fitting computes a rough relative scale between the object and the scene. As a result, the object still appears floating in many frames, producing less accurate depth estimations. In contrast, PPR couples differentiable physics optimization with differentiable rendering to jointly solve for the object scale and its global movements, which successfully reduces errors on the dynamic scene reconstruction task.

B. Additional Implementation Details

Regularization Terms. During differentiable rendering optimization, we apply shape and motion regularization terms as follows. We use 3D cycle loss to encourage the forward and backward warping fields \(\mathcal{W}\) to be consistent with each other [5, 11]. We additionally apply an eikonal loss [3, 12] to both scene and object fields, which enforces the reconstructed signed distances to represent a surface:

\[
\mathcal{L}_{\text{eikonal}} = (\| \nabla x \text{MLP}_{\text{SDF}}(X) \| - 1)^2,
\]

where we force the first order gradient of predicted SDF to have unit norm. Eikonal regularization helps produce
well-defined mesh when running marching cubes on the implicitly-defined surface.

**Rag Doll Optimization.** To optimize the object fields, we start with a general rest shape (a unit sphere) and a known skeleton topology of the rag doll model. During optimization, both the shape and rag doll model (joint locations and generalized mass of each link) are specialized to fit the input videos. Please see Fig. 2 for the visualization of rest shapes and rag doll models.

**Contact Plane Fitting.** We assume the potential contact bones of a skeleton (the “feet”) are known, and the contact locations are visible. The algorithm is as follows:

- Input: Scene points \( p \in \mathbb{R}^{N \times 3} \), scene-to-camera transforms \( G_{a \rightarrow c} \in \mathbb{R}^{4 \times 4} \) over \( T \) frames, camera intrinsics \( K \in \mathbb{R}^{3 \times 3} \), and object “feet” trajectories in the camera space \( J \in \mathbb{R}^{T \times B \times 3} \).
- Output: Contact plane parameters \( A = (n, d) \).
- Parameters: Number of plane hypotheses \( K = 5 \), threshold \( T_1 = 0.01 \).

**Step 1: Fit Multiple Planes**

For \( k \) in \( 1 : K \):

- Fit a plane \( A_k \) to \( p \) using RANSAC with threshold \( T_1 \).
- Set inlier points of \( A_k \) as \( P_k \), and remove those from \( p \).

**Step 2: Find the Plane in Contact**

- **Project** scene points to images: \( p = KG_{a \rightarrow c}p \in \mathbb{R}^{T \times N \times 2} \).
- **Project** “feet” points to images: \( q = KJ \in \mathbb{R}^{T \times B \times 2} \).

For \( k \) in \( 1 : K \):

- **Score** \( A_k \) by “feet”-to-\( P_k \) distance over frames and “feet”:
  \[ d = \sum_{t=1}^{T} \sum_{j=1}^{B} \min(||p_{kj} - q_{tj}||) \]  
- **Return** \( A_k \) with the lowest total “feet”-to-\( P_k \) distance.

Under those assumptions, the contact plane does not have to occupy the majority of the background, and cameras do not have to point forward. Our algorithm works for the videos we tested on (included in the supplementary page), but breaks: (1) when the contact points are hard to define (e.g., cat lying sideways), or (2) when the object makes contact with multiple planes in a video.

**Gradient Clipping.** We find that differentiable physics introduces unstable gradients to the optimization, causing a high final reconstruction loss. Therefore, we clip outlier gradients to an empirical value \( c = 0.1 \):

\[
\nabla_\phi L_{DP} = \begin{cases} 
\frac{\nabla_\phi L_{DP}}{c} 
& \text{if } \|\nabla_\phi L_{DP}\| \leq c \\
\frac{\|\nabla_\phi L_{DP}\|}{c} \nabla_\phi L_{DP} 
& \text{if } \|\nabla_\phi L_{DP}\| > c
\end{cases}
\]  

(3)

where \( L_{DP} \) is the differentiable physics loss in Eq. (11) and \( \phi \) is the physics parameters.

**C. Additional Results**

**Comparison with animal body models.** Creating accurate body models for animals is difficult due to lack of 3D data containing diverse animal shape, appearance, and pose. In the following, we show a visual comparison with BARC [10], a state-of-the-art dog body model in Fig. 3. The video comparison can be found on the supplement website.

![Comparison with BARC](image)

**Roll-out Performance.** In Fig. 4, we show qualitative results of simulating the physical system (rag doll model) for various time windows. Within the time window \( T \) in training, the simulation is almost always stable. When simulating a time window greater than \( T \), the controller might fail to track the motion.

We posit that it is because the error in the states of the rag doll model accumulates over time [9]. The PD controller is not able to generalize to never-before scenarios. One potential direction to improve this is to ask the controller to reason about future time horizons (instead of the direct next step) [7].

![Simulation over long time window](image)
References


