Simulation-Ready Cluttered Scene Estimation via Physics-aware Joint Shape and Pose Optimization

Wei-Cheng Huang[1], Jiaheng Han[1], Xiaohan Ye[2], Zherong Pan[3], and Kris Hauser[1]

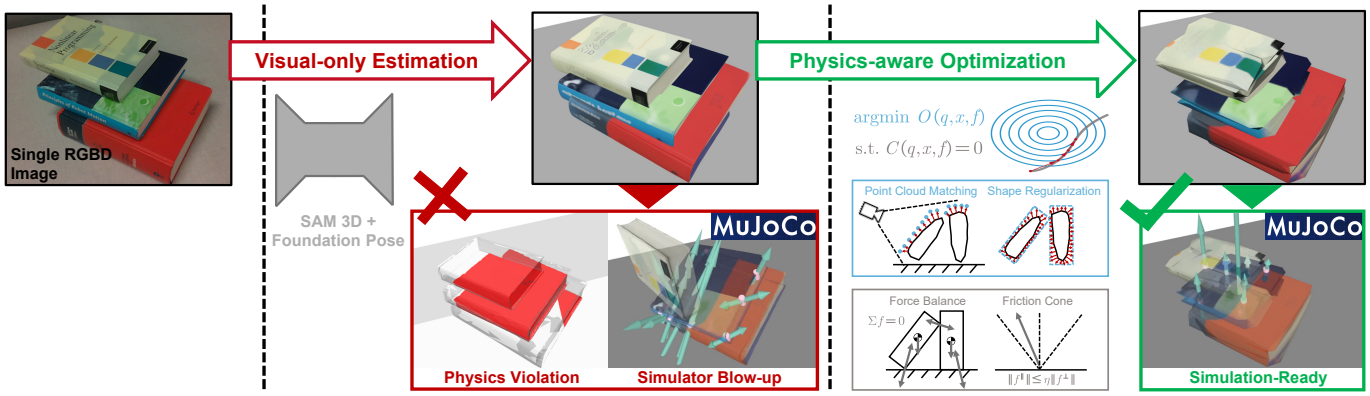

Fig. 1: Given a single RGBD image observation of a cluttered scene, we use SAM3D and FoundationPose to derive an initial estimation of object shapes and poses. But these estimates can violate physical constraints and are not simulation ready (red). Our method jointly adjusts shape and pose parameters to enforce physics constraints while minimizing a perceptual loss, leading to simulation ready results (green).

***Abstract*—Estimating simulation-ready scenes from real-world observations is crucial for downstream planning and policy learning tasks. Regretfully, existing methods struggle in cluttered environments, often exhibiting prohibitive computational cost, poor robustness, and restricted generality when scaling to multiple interacting objects. We propose a unified optimization-based formulation for real-to-sim scene estimation that jointly recovers the shapes and poses of multiple rigid objects under physical constraints. Our method is built on two key technical innovations. First, we leverage the recently introduced shape-differentiable contact model, whose global differentiability permits joint optimization over object geometry and pose while modeling inter-object contacts. Second, we exploit the structured sparsity of the augmented Lagrangian Hessian to derive an efficient linear system solver whose computational cost scales favorably with scene complexity. Building on this formulation, we develop an end-to-end real-to-sim scene estimation pipeline that integrates learning-based object initialization, physics-constrained joint shape-pose optimization, and differentiable texture refinement. Experiments on cluttered scenes with up to 5 objects and 22 convex hulls demonstrate that our approach robustly reconstructs physically valid, simulation-ready object shapes and poses. Accompanying media can be found at the paper webpage.**

I. Introduction

Scene estimation is a fundamental problem in robotics and embodied AI, particularly for real-to-sim transfer. An ideal scene estimator should reconstruct a simulation-ready environment from sparse observations such as images. Beyond perceptual fidelity, the estimated object shapes, poses, and physical properties must be physically consistent and directly usable within a physics simulator, which is critical for downstream tasks such as motion planning, model predictive control, and policy learning. This problem is rather challenging in cluttered scenes, where multiple objects interact through contact, and where accurate physical reasoning is essential for tasks such as robotic manipulation. Over the years, a wide range of scene estimation paradigms have been developed, including Bayesian inference [12], deep learning [38], and numerical optimization [44]. Among these, optimization-based scene estimator [44] offers a distinctive advantage for real-to-sim applications: they allow explicit incorporation of physical laws and constraints into the estimation process. By enforcing non-penetration, contact consistency, and equilibrium conditions, physics-based constraints can substantially regularize the solution space and reduce ambiguities.

Despite these advantages, a major challenge for optimization-based state estimators lies in the formulation of physics constraints, which introduces a large number of auxiliary variables, including normal and frictional contact forces as well as Lagrange multipliers. Most prior approaches jointly optimize all variables within a single large-scale nonlinear programming (NLP) formulation using off-the-shelf solvers [13, 34]. This monolithic strategy leads to computationally expensive problems that scale poorly to cluttered scenes with many interacting objects. To mitigate

[1] Siebel School of Computing and Data Science, University of Illinois at Urbana-Champaign. [2] Department of Computer Science, The University of Hong Kong. [3] Meta Reality Labs.

This project was partially supported by NSF Grant #IIS-1911087 and #2409661. All experiments, data collection, and processing activities were conducted by the University of Illinois Urbana-Champaign. Meta was involved solely in an advisory role.

this complexity, practical methods such as [44] rely on heuristic contact-selection oracles, which are inherently brittle and may fail when contacts are missed. More fundamentally, to keep computational costs tractable, existing approaches assume known object geometries and restrict optimization to object poses. In contrast, scene estimation from sparse observations inherently requires jointly inferring both object shapes and poses, dramatically increasing the dimensionality of the decision space. The resulting proliferation of shape parameters renders existing optimization-based techniques computationally prohibitive and, in practice, intractable.

We attribute these limitations to a common root cause: existing optimization-based state estimators are not structure-aware [44, 45]. By formulating all variables within a monolithic NLP, they fail to exploit the structure of physics-constrained optimization. We therefore propose a structure-aware framework for real-to-sim scene estimation in cluttered environments—the first practical algorithm for numerical optimization in the joint shape-pose space. Our approach builds on the separating-plane-based shape-differentiable contact model (SDRS) [41], which eliminates normal contact forces as explicit variables by expressing them as functions of object pose. We adapt SDRS to quasistatic configuration optimization, reducing problem dimensionality, and show that the resulting augmented Lagrangian Hessian has a highly structured sparsity pattern that enables efficient solvers via Woodbury and Schur complement reductions. The proposed formulation is globally differentiable with respect to both shape and pose, enabling joint optimization under arbitrary contact conditions with shapes represented as unions of convex hulls. To improve robustness, we consider all potential contact pairs without heuristic contact selection [44], while maintaining tractable computational cost through our reduced representation. A globally supported contact activation function [42] further mitigates vanishing gradients and constraint-qualification issues.

Building upon our joint optimization framework, we develop an end-to-end real-to-sim scene estimation pipeline that operates directly on a single RGBD image observation as illustrated in Figure 1. We evaluate our method on a diverse set of cluttered benchmarks containing up to 5 objects and 22 convex hulls, where our method can robustly produce physically valid, simulation-ready reconstructions.

II. Related Work

In this section, we review prior work on state estimation and scene understanding, focusing on their relevance to real-to-sim transfer, cluttered environments, and simulation-ready reconstruction.

a) Scene Estimation: Early scene estimation methods typically assume known object geometries and focus on recovering object poses from partial observations. Classical approaches [11, 30] formulate rigid-body registration as geometric alignment, with later extensions to non-rigid [3] and articulated [9] models. While geometrically well founded, these methods are brittle under occlusion and missing data, which are ubiquitous in cluttered real-world scenes. Learning-based approaches, such as PoseCNN [38] and FoundationPose [36], improve robustness by leveraging learned priors, but remain purely perception-driven. Lacking explicit physics constraints, they are ill-suited for simulation-ready scene estimation. More recent works [32, 25, 8, 40, 37] incorporate physical reasoning through sampling-based optimization or physics-violation loss terms, but remain limited in scope: they assume a small set of fixed hypothesized object shapes [32], optimize the shape of only a single object [8], require dense observation such as RGB video [25, 37], or model collision constraints without enforcing full physical consistency [40].

b) Physics-aware Numerical Optimization: To improve physical validity, prior work has integrated physics constraints into numerical optimization. Methods such as PhysPose [23] and Verefine [6] encourage physically plausible configurations via penalties or post hoc simulation checks, but do not provide full physics-consistent optimization. A more principled approach [44] enforces non-penetration and force equilibrium constraints directly, but relies on heuristic contact-selection oracles that degrade robustness in cluttered scenes. Extensions to deformable objects [15] further expand the scope of physics-aware estimation. Nevertheless, most existing methods assume known object geometry and focus exclusively on pose estimation, making them unsuitable for real-to-sim transfer where object shapes must also be recovered. We are aware of one prior work [8] incorporating both shape and pose reasoning but limited to a single object.

c) Differentiable Simulation & Rendering: Differentiable simulators and renderers [17, 24] enable gradient-based optimization of physically grounded scenes and have been applied to large-scale state estimation [39, 22, 46]. However, most differentiable simulators target dynamic trajectories rather than quasistatic, force-balanced configurations typical of cluttered scenes. Moreover, non-smooth contact and visibility events violate the smoothness assumptions of NLP solvers. We build upon the globally differentiable SDRS contact model [41] and reformulate it for quasistatic equilibrium, enabling smooth and scalable joint optimization of object shape and pose. This design directly supports simulation-ready reconstruction of cluttered real-world scenes for real-to-sim transfer.

III. Methods

In this section, we present our complete pipeline for simulation-ready cluttered scene reconstruction. We then focus on the details of our optimization-based scene estimator for a set of rigid bodies in the joint pose- and shape-space.

A. Problem Statement

Following [44], we express the problem as an equality-constrained NLP of the following form:

$$\underset{q,x}{\operatorname{argmin}} \; O(q,x) \quad \text{s.t. } C(q,x) = 0, \tag{1}$$

and we propose a structured variant of Augmented Lagrangian Method (ALM) to find locally optimal solutions. We will omit function parameters below when confusion is unlikely. Here, q and x denote the vectors describing the poses and

shapes of all objects, respectively, and the physics constraints are modeled as the globally differentiable equality constraint $C(q,x)=0$. The objective $O(q,x)$ is assumed to be an arbitrary globally differentiable function, allowing flexibility for various state-estimation tasks. For instance, O could represent a differentiably rendered image-space loss [21] or a point cloud registration loss [2].

Our formulation builds on the recently proposed convex-hull-based contact model [41]—a provably second-order differentiable contact model, enabling globally twice-differentiable operations under arbitrary changes in convex hull geometry. As illustrated in Figure 3, we represent a scene with N rigid bodies, each modeled as a union of M convex hulls, with each hull having V vertices. Specifically, we denote $x_{ijk} \in \mathbb{R}^3$ as the kth vertex of the jth convex hull belonging to the ith rigid body, expressed in the body frame. Concatenating these vertices, we obtain $x \in \mathbb{R}^{N\times M\times V\times 3}$, which describes the shapes of all rigid bodies. To represent the poses of these bodies, we define the local-to-global transformation of the ith rigid body using a rotation matrix $R_i(\theta_i) = \exp[\theta_i]_\times$ and a translation vector $t_i \in \mathbb{R}^3$, where the rotation is parameterized via the Rodrigues formula with $\theta_i \in \mathbb{R}^3$ and $[\bullet]_\times$ being the cross-product matrix. Concatenating all θ_i and t_i yields the vector q, which encodes the poses of all rigid bodies. It is known that the mapping $R_i(\theta_i)$ is smooth [14]. The local-to-global transformation is defined as $X_{ijk} = R_i x_{ijk} + t_i$ and we omit the function parameters when the context is clear.

B. Scene Estimation Pipeline

Our joint optimizer uses a local gradient-based method that is prone to getting trapped in local minima without a high-quality initialization. To obtain a high-quality initial guess, we first extract the point cloud and then leverage the learning-based model SAM3D [10] to guess initial object shapes from the point cloud. SAM3D allows us to extract the mesh and segment the point cloud for each mesh. The result is denoted as N points clouds $\mathcal{P}_{1,\dots,N}$ and N corresponding meshes $\mathcal{M}_{1,\dots,N}$. While SAM3D also predicts object poses, we observe that these estimates are often inaccurate. Therefore, we refine the object poses using the FoundationPose model [36], producing an initial guess for the pose vector q. Next, for each mesh $\mathcal{M}_i$, we apply convex decomposition [35] to generate the vertices of convex hulls, yielding the initial guess for the vector x. These initial estimates are then adjusted to ensure penetration-free between different bodies and fed into our joint optimization to enforce the physical constraints (see our appendix for details). Although we describe our method assuming each rigid body is represented by M convex hulls, each with V vertices, our implementation is fully general and can accommodate an arbitrary number of convex hulls and vertices as determined by the convex decomposition [35].

With the segmented point clouds $\mathcal{P}_i$, the extracted mesh $\mathcal{M}_i$, and the convex hulls, we can define differentiable objective function O via the similar technique as the trimmed ICP [11], where we select ICP terms to ensure monotonic objective value reduction over iterations, thus guaranteeing convergence. Specifically, for each convex hull vertex X_{ijk} in world space, we already know that it belongs to the ith rigid body, so we search for the closest point on $\mathcal{M}_i$ as a continuous manifold to X_{ijk}, denoted as:

$$p(X_{ijk}) = \operatorname*{argmin}_{x\in\mathcal{M}_i} \|x - X_{ijk}\|. \tag{2}$$

We then fix $p(X_{ijk})$ and introduce a term $\|X_{ijk}(q,x) - p(X_{ijk})\|^2$ to regularize the convex hull shapes. Conversely, for each point p_{il} belonging to the ith point cloud, we penalize its distance to the surface formed by union of convex hulls representing the ith rigid body, denoted as $\partial \cup_j \mathrm{CH}(X_{ij\bullet})$. We find the closest point to p_{il} on the union of convex hulls as a continuous manifold, denoted as:

$$X(p_{il}) = \operatorname*{argmin}_{X\in\partial\cup_j \mathrm{CH}(X_{ij\bullet})} \|X - p_{il}\|. \tag{3}$$

In practice, we use Manifold3d Library [19] to compute union of convex hulls and extract the surface into a triangle mesh, which then allows us to compute $X(p_{il})$ as a point-to-mesh distance problem. The computed $X(p_{il})$ must lie on the surface of a convex hull. Without a loss of generality, we can assume its the $j(p_{il})$'s convex hull, i.e., $X(p_{il}) = \sum_k X_{ij(p_{il})k} w_k$, with w_k being the convex combination weights. We then follow the idea of ICP and fix the convex combination weights w_k so that $X(p_{il})$ is a function of x and q only. We can then introduce the term $\|X(p_{il}) - p_{il}\|^2$ to further regularize the convex hull shapes. Finally, we notice that the point-cloud can only regulate the object shape in visible areas. To further regulate object shapes in invisible areas, we utilize the SAM3D-recovered mesh $\mathcal{M}_i$. For each vertex $p_{il} \in \mathcal{M}_i$, we use a similar technique to compute the closest point $X(p_{il})$ and introduce the term $\|X(p_{il}) - p_{il}\|^2$. In summary, our objective function is formulated as a sum of three types of terms weight by coefficients $w_{1,2,3}$:

$$O(q,x) = \begin{cases} w_1 \sum_{ijk} \|X_{ijk} - p(X_{ijk})\|^2 + & \text{Type I} \\ w_2 \sum_{p_{il}\in\mathcal{P}_i} \|X(p_{il}) - p_{il}\|^2 + & \text{Type II} \\ w_3 \sum_{p_{il}\in\mathcal{M}_i} \|X(p_{il}) - p_{il}\|^2 & \text{Type III.} \end{cases} \tag{4}$$

Our formulation follows the idea of Hausdorff distance that penalizes symmetric distances. Our type I term regularize the distance between convex hull's vertex X_{ijk} and $\mathcal{M}_i$. The remaining type II (resp. type III) term regularize the distance between the point cloud (resp. mesh) and the union of convex hull surface $\partial \cup_j \mathrm{CH}(X_{ij\bullet})$. We assume that the point cloud $\mathcal{P}_i$ is the direct observation and serves a strong guidance for correct object shapes, which must be emphasized using a large weight w_2. On the other hand, the mesh $\mathcal{M}_i$ might be hallucinated by SAM3D [10], which is not necessarily matched exactly, but used as a shape prior via a small weight w_3.

A critical drawback of the above procedure is the violation of NLP convergence guarantee. Note that our type I term $\|X_{ijk} - p(X_{ijk})\|^2$ adopts a standard treatment of ICP, where we can fix $p(X_{ijk})$ to solve ALM optimization, and then update $p(X_{ijk})$ via Equation 2. This procedure is guaranteed

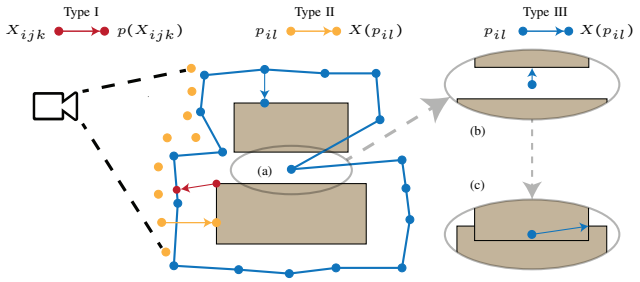

Fig. 2: We illustrate the three types of objectives in Equation 4, regularizing the distance between convex hull vertex X_{ijk} (red), the SAM3D-identified mesh vertex $p_{il} \in \mathcal{M}_i$ (blue), and the point cloud $p_{il} \in \mathcal{P}_i$ (yellow). Further, we highlight a case (a) where objective value can increase. Suppose our rigid body (light brown) consists of two disjoint convex hulls (b), the closest point to the blue vertex is the bottom surface of the top hull. After an update to hull vertices (c), the two hulls merge and the closest point is moved to the right boundary.

to converge [11] because each term monotonically decreases over iterations. However, our type II and type III terms $\|X(p_{il}) - p_{il}\|^2$ are not guaranteed to decrease over iterations because the update in Equation 3 can increase function values. This is essentially because the reference mesh $\mathcal{M}_i$ has fixed geometry but our union of convex hull can undergo shape changes. To rigorously ensure convergence, we introduce a heuristic technique inspired by [11], where we selectively delete a subset of type II and type III terms that can increase function values. Specifically, we sort all type II and type III terms by the amount of function value increment after an update of $X(p_{il})$ according to Equation 3:

$$\Delta_{il} = \|X(p_{il}) - p_{il}\|^2 - \|X(p_{il})^{\text{prev}} - p_{il}\|^2, \tag{5}$$

where $X(p_{il})^{\text{prev}} = \sum_k X_{ij(p_{il})k} w_k^{\text{prev}}$ is the point using convex combination weights from the previous iteration. We repeatedly delete the type II or type III term yielding the highest function value increase until the objective function O is non-increasing. The entire procedure of closest point update and selective term deletion is performed after each ALM subproblem solve. The complete pipeline is outlined our appendix and illustrated in Figure 2.

As an optional final step of our method, we can generate the color texture for each object by differentiable rasterization. Specifically, after the ALM optimization, we fix the object shape and pose. We then use Manifold3d Library [19] to convert the union of convex hulls into a triangle mesh. Next, we use xatlas [43] to generate the UV coordinates for each mesh. Finally, we use differentiable renderer [18] to minimize the difference between the SAM3D-predicted and the mesh-rendered image, with the texture map being the decision variables.

C. Optimization in Joint Shape-Pose Space

It is well-known [31] that the key requirement for finding a locally optimal and feasible solution of Problem 1 is that C satisfies the Linear Independent Constraint Qualification (LICQ), i.e., $\lambda_{\min}(\nabla C \nabla C^T)$ is bounded away from zero for any q and x. In practice, we use a custom version of ALM that iteratively minimizes the following augmented Lagrangian function, where we use Levenberg-Marquardt (LM) algorithm as the sub-problem:

$$\underset{q,x}{\text{argmin}} \; O(q,x) + \lambda^T C(q,x) + \frac{\rho}{2}\|C(q,x)\|^2, \tag{6}$$

and we iteratively increase the Lagrangian multiplier λ and the penalty coefficient ρ according to [26].

Our formulation in Problem 1 differs from prior work in several key ways. First, unlike complementarity constraint-based formulations [44, 27], we eliminate auxiliary variables for normal contact forces, substantially reducing problem dimensionality and subproblem cost. Second, the formulation involves only equality constraints, so each subproblem reduces to solving a linear system rather than a general quadratic program, yielding significant speedups. We first analyze this frictionless case, then introduce friction as additional decision variables and present structured linear solvers to handle them efficiently.

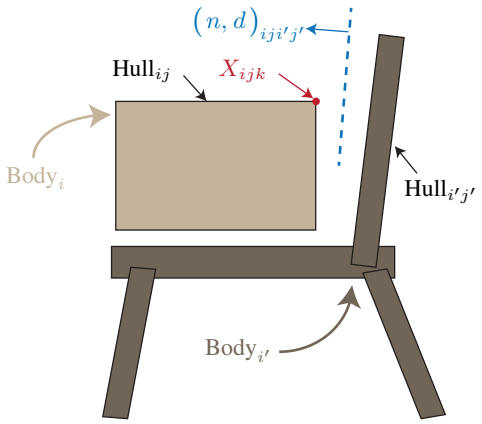

Fig. 3: Suppose we would like to model a box (light brown) put on a chair (dark brown). The box and the chair are the ith and i'th rigid bodies respectively, where the box is modeled as a single convex hull and the chair is modeled as the union of 4 convex hulls. Each convex hull is a polytope spanned by a set of vertices X_{ijk} (red). Between the ijth convex hull on the box and the $i'j'$th convex hull on the chair modeling the back support, we introduce a separating plane $(n,d)_{iji'j'}$ (blue) as a proxy for the contact model.

1) Physics Constraints Without Friction: Let us consider the case where all objects experience only normal contact forces, without friction. Using the representation introduced in the previous section, the physics constraints reduce to minimizing the internal and external potential energy at the resting pose. The potential energy $\Psi(q,x)$ consists of two terms: the gravitational potential Ψ_g and the collision potential Ψ_c, such that $\Psi = \Psi_g + \Psi_c$. The gravitational potential is formulated under the assumption that the mass is concentrated at the convex hull vertices [41], since a uniform mass distribution

would be non-differentiable with respect to x. Under this assumption, we define:

$$\Psi_g(q,x) = -\rho \sum_{i,j,k} \langle X_{ijk}, g \rangle, \tag{7}$$

where ρ and g denote the density and gravitational acceleration, respectively. Clearly, Ψ_g is twice-differentiable with all variables, which allow users to optimize object density and mass distributions. We then define the constraint as $C = \nabla_q \Psi$. If Ψ is globally twice-differentiable, then C is differentiable and satisfies the smoothness requirement.

Our second term is the collision potential. Compared with hard constraints for modeling collision [27, 44], our method follows recent findings that using a primal-only, instead of primal-dual, interior-point formulation [20] provides a more robust approach. This is because interior-point methods maintain satisfaction of collision constraints, avoiding the vanishing gradient issue that can occur under deep penetration. Ye et al. [41] generalized the idea of [20] by defining a potential function between two general convex hulls. Specifically, two convex hulls are intersection-free if and only if there exists a separating plane between them [29]. Accordingly, we introduce the separating plane as an additional set of variables $\left(n,d\right)_{iji'j'} \in \mathbb{R}^4$, which separates the ijth and $i'j'$th convex hulls. Here, $n_{iji'j'}$ is the plane normal and $d_{iji'j'}$ is the plane offset. The collision potential between the two convex hulls is then defined as $\bar{\Psi}_{iji'j'}(q,x,n_{iji'j'},d_{iji'j'})$:

$$\bar{\Psi}_{iji'j'} = \begin{cases} -\log(1-\|n_{iji'j'}\|) \\ \sum_k -\log(\langle n_{iji'j'}, X_{ijk}\rangle + d_{iji'j'}) \\ \sum_{k'} -\log(-\langle n_{iji'j'}, X_{i'j'k'}\rangle - d_{iji'j'}), \end{cases} \tag{8}$$

The first term ensures that the plane normal has a magnitude no greater than 1, while the second and third terms ensure that the two convex objects lie on opposite sides of the separating plane, thereby enforcing collision-free constraints. It has been shown that $\bar{\Psi}_{iji'j'}$ is well-defined and globally twice-differentiable. Although the definition of $\bar{\Psi}_{iji'j'}$ introduces the separating-plane variables $\left(n,d\right)_{iji'j'}$, the function is strictly convex with respect to these variables. Therefore, we can eliminate them implicitly and define:

$$\Psi_{iji'j'}(q,x) = \min_{n_{iji'j'}, d_{iji'j'}} \bar{\Psi}_{iji'j'}(q,x,n_{iji'j'},d_{iji'j'}), \tag{9}$$

By the implicit function theorem, $\Psi_{iji'j'}$ remains globally twice-differentiable. Finally, the full collision potential is defined as $\Psi_c = \mu \sum_{i \neq i'} \sum_{j,j'} \Psi_{iji'j'}$, where μ is the complementarity gap. As shown in [20], as $\mu \to 0$, Ψ_c approximates hard collision-free constraints arbitrarily well. Further, the optimality of $\bar{\Psi}_{iji'j'}$ with respect to the separating plane $\left(n,d\right)_{iji'j'}$ ensures that equal and opposite forces are applied on the two convex hulls, which essentially satisfy the Newton's third law. Finally, the function $\Psi = \infty$ when collision constraints are violated, in which case the solution will be rejected by the ALM subproblem solver to use a smaller search step size, thus ensuring the collision constraints are satisfied at every iteration. To improve efficiency, a clamped log function can be used to make the potential locally supported, so that $\Psi_{iji'j'}$ evaluates to zero when objects are far apart. In practice, thanks to the compact convex-hull representation, computing all pairs of collisions is sufficiently efficient, allowing us to use a globally supported log function. The global support is beneficial because it ensures non-zero gradients even for distant objects, which helps satisfy the LICQ condition. We summarize the well-behaved properties of our physics constraints below, which is formally proved in [41]:

Property III.1. *Under frictionless setting, $C(q,x)$ is globally differentiable with respect to both q and x. When $C(q,x) = 0$, all objects are force and torque balanced while satisfying the Newton's third law.*

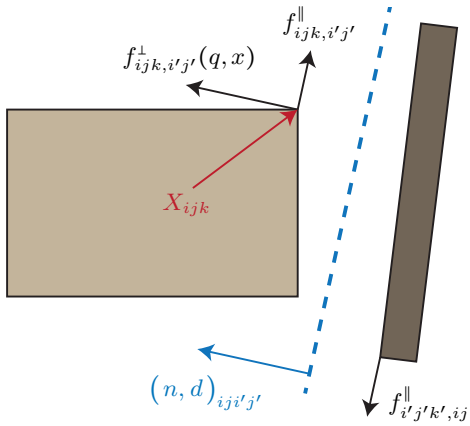

Fig. 4: An illustration of our friction model, between the ijth and $i'j'$th convex hull. On each vertex, e.g. X_{ijk}, the normal force $f^{\perp}_{ijk,i'j'}$ is a function of x and q (Equation 10) and the friction force $f^{\parallel}_{ijk,i'j'}$ is additional decision variables to be optimized. We follow the idea of SDRS contact model and use the separating plane as the proxy for contact modeling and each force $f^{\parallel}_{ijk,i'j'}$ applied on the ijth convex hull is counteracted by $-f^{\parallel}_{ijk,i'j'}$ applied on the separating plane $\left(n,d\right)_{iji'j'}$, and the case is the same for the $i'j'$th convex hull. We then model the separating plane as a physical object with zero mass, so that all the forces applied on it must be balanced.

2) Incorporating Frictional Contacts: Frictional contacts are more involved, since friction forces always dissipate kinetic energy and therefore do not admit a corresponding potential energy formulation. In this section, we first formulate our frictional contact model and then present our structure-aware algorithm for solving the ALM subproblem Problem 6. As illustrated in Figure 4, we consider a pair of convex hulls. For each vertex x_{ijk} (the case for $x_{i'j'k'}$ is symmetric), the normal contact force is given by:

$$f^{\perp}_{ijk,i'j'} = \partial \Psi_{iji'j'} / \partial X_{ijk}. \tag{10}$$

In addition, by the friction cone constraint, the vertex may also experience tangential frictional forces $f^{\parallel}_{ijk,i'j'}$, which satisfy:

$$\left\langle f^{\perp}_{ijk,i'j'}, f^{\parallel}_{ijk,i'j'} \right\rangle = 0 \quad \|f^{\parallel}_{ijk,i'j'}\| \leq \eta \|f^{\perp}_{ijk,i'j'}\|, \tag{11}$$

where η is the friction coefficient of the cone. However, the friction cone constraints alone are not sufficient. We must also ensure that the collection of frictional forces applied on the two convex hulls satisfies both force and torque balance. To achieve this, we adopt the idea of [41], in which the separating plane is treated as a fictitious physical object with zero mass. While the separating plane has no direct physical impact on the system, it serves as a proxy for formulating and enforcing the balance of frictional forces. Concretely, for a frictional force applied on x_{ijk}, an opposing force is applied to the plane, resulting in an in-plane force of $-f^{\parallel}_{ijk,i'j'}$ and an in-plane torque of $-T_{iji'j'}X_{ijk} \times f^{\parallel}_{ijk,i'j'}$, where $T_{iji'j'}$ denotes the projection operator onto the plane normal space and $n_{iji'j'}$ is the (unnormalized) separating plane normal, as defined in the normal collision potential (Equation 8). Finally, since the separating plane has zero mass, it must remain in equilibrium to avoid unbounded accelerations. This yields the following in-plane force and torque balance conditions:

$$\begin{aligned} &\sum_k f^{\parallel}_{ijk,i'j'} + \sum_{k'} f^{\parallel}_{i'j'k',ij} = 0, \\ &T_{iji'j'}\left[\sum_k X_{ijk} \times f^{\parallel}_{ijk,i'j'} + \sum_{k'} X_{i'j'k'} \times f^{\parallel}_{i'j'k',ij}\right] = 0. \end{aligned} \tag{12}$$

Finally, we incorporate the frictional contact forces by defining the augmented potential energy and constraint function $\bar{C}$:

$$\bar{C}(q, x, f^{\parallel}_{iji'j'}) = \nabla_q \left[\Psi(q,x) - \sum_{i \neq i'} \sum_{j,j'} \sum_k \langle X_{ijk}, f^{\parallel}_{ijk,i'j'} \rangle\right]. \tag{13}$$

Again, we summarize the well-behaved properties of our augmented physical constraints below:

Property III.2. *When $\mu > 0$, $\bar{C}(q, x, f^{\parallel}_{iji'j'})$ is globally differentiable with respect to both q and x. When $\bar{C}(q, x, f^{\parallel}_{iji'j'}) = 0$, all objects are force and in-plane torque balanced while satisfying the Newton's third law.*

We emphasize that an important difference from the frictionaless case is that we can only ensure torque balance for the components perpendicular to the plane normal [41]. Unfortunately, the torque component along the plane normal direction is not balance in general. But the violation to torque balance along this component is controlled by the complementarity gap μ. By selecting a small value of μ, the violation is negligible in practice.

Although the frictional contact forces are straightforward to formulate, incorporating these constraints would significantly increase the dimensions of the decision space. Indeed, we need to optimize a pair of per-vertex frictional forces $f^{\parallel}_{iji'j'}$ between each pair of rigid bodies, leading to the following constrained optimization:

$$\underset{q,x,f^{\parallel}_{iji'j'}}{\text{argmin}} \quad O(q,x) \tag{14}$$

$$\text{s.t.} \begin{cases} \bar{C}(q, x, f^{\parallel}_{iji'j'}) = 0 \\ \left\langle f^{\perp}_{ijk,i'j'}, f^{\parallel}_{ijk,i'j'} \right\rangle = 0 \\ \|f^{\parallel}_{ijk,i'j'}\| \leq \eta \|f^{\perp}_{ijk,i'j'}\| \\ \sum_k f^{\parallel}_{ijk,i'j'} + \sum_{k'} f^{\parallel}_{i'j'k',ij} = 0, \\ T_{iji'j'}\left[\sum_k X_{ijk} \times f^{\parallel}_{ijk,i'j'} + \sum_{k'} X_{i'j'k'} \times f^{\parallel}_{i'j'k',ij}\right] = 0. \end{cases}$$

The above optimized when handled using the ALM algorithm, which in turn requires large-scale linear system solves in the underlying LM algorithm—a major computational bottleneck. To tackle this issue, we notice that the additional complexity due to the frictional forces can be largely eliminated by utilizing the special sparsity pattern in the underlying Gauss-Newton Hessian matrix. First, we notice that the ALM subproblem takes the following form:

$$\underset{q,x,f_{iji'j'}}{\text{argmin}} \quad O + \lambda^T \bar{C} + \frac{\rho}{2}\|\bar{C}\|^2 + \sum_{iji'j'} \Phi(q, x, f_{iji'j'}), \tag{15}$$

where each term $\Phi(q, x, f_{iji'j'})$ includes the constraint (Equation 11 and Equation 12) and augmented Lagrangian terms due to frictions between each pair of convex hulls ij and $i'j'$, with definitions deferred to appendix. Different pairs of convex hulls are coupled only in the term $\bar{C}(q, x, f_{iji'j'})$, so that the Gauss-Newton Hessian matrix takes the following form:

$$H \triangleq \nabla^2 O + \sum_{iji'j'} \nabla^2 \Phi + \nabla \bar{C}^T \nabla \bar{C}. \tag{16}$$

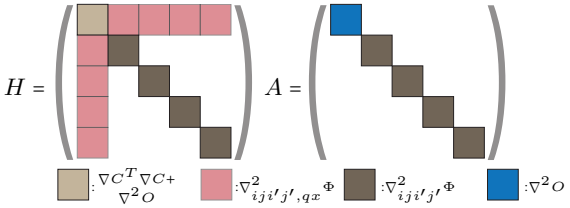

Fig. 5: We illustrate the structure of matrix H (left) and matrix A (right). A is block-diagonal and each block is small. H is factored using the Woodbury matrix identity.

We first notice that $\nabla \bar{C}$ has a rank of at most $|q|$, which is the dimension of the configuration space. Therefore, we could use the Woodbury matrix identity to efficiently solve the linear system via Algorithm 1, where the related matrices are illustrated in Figure 5. However, Algorithm 1 still requires solving the linear system with matrix A on the left hand side, which again involves all the decision variables. Fortunately, since the matrix $\nabla \bar{C}$ has been factored out of the matrix A, frictional forces between different pairs of convex hulls have been decoupled, allowing us to use the Schur complement solver to efficiently solve such linear systems. Specifically, given any righthand side b, we can decompose b as:

$$b = \left({b_{qx}}^T, \ldots^T, {b_{iji'j'}}^T, \ldots^T\right)^T, \tag{17}$$

where the first block corresponds to the first $|q| + |x|$ rows and the follow-up blocks each correspond to the rows for each pair of convex hulls $iji'j'$. Similarly, the Hessian of $\nabla^2\Phi(q, x, f^{\parallel}_{iji'j'})$ has the following decomposition:

$$\nabla^2\Phi = \begin{pmatrix} \nabla^2_{qx}\Phi & \nabla^2_{qx,iji'j'}\Phi \\ \nabla^2_{iji'j',qx}\Phi & \nabla^2_{iji'j'}\Phi \end{pmatrix} \tag{18}$$

With the above notation, we can solve the linear system as outlined in Algorithm 2.

Algorithm 1 Solve-H(b)

Input: $\nabla\bar{C}, \nabla^2\Phi(q, x, f^{\parallel}_{iji'j'}), \nabla^2 O$
Output: $H^{-1}b$
1: $A \leftarrow \nabla^2 O + \sum_{iji'j'} \nabla^2\Phi$
2: $\triangleright$ We use the following Woodbury matrix identity
3: $\triangleright$ $A^{-1} - A^{-1}\nabla\bar{C}^T(I + \nabla\bar{C}A^{-1}\nabla\bar{C}^T)^{-1}\nabla\bar{C}A^{-1}$
4: $b \leftarrow$Solve-A(b)
5: $c \leftarrow \nabla\bar{C}^T(I + \nabla\bar{C}A^{-1}\nabla\bar{C}^T)^{-1}\nabla\bar{C}b$
6: Return b–Solve-A(c)

Algorithm 2 Solve-A(b)

Input: $\nabla^2\Phi(q, x, f^{\parallel}_{iji'j'}), \nabla^2 O$
Output: $A^{-1}b$
1: $A_{qx} \leftarrow \nabla^2 O$
2: **for** Each pair of convex hulls $iji'j'$ **do**
3: $\quad A_{qx} \leftarrow A_{qx} - \nabla^2_{qx,iji'j'}\Phi\nabla^2_{iji'j'}\Phi^{-1}\nabla^2_{iji'j',qx}\Phi$
4: $\quad b_{qx} \leftarrow b_{qx} - \nabla^2_{qx,iji'j'}\Phi^T\nabla^2_{iji'j'}\Phi^{-1}b_{iji'j'}$
5: $b_{qx} \leftarrow A^{-1}_{qx}b_{qx}$
6: **for** Each pair of convex hulls $iji'j'$ **do**
7: $\quad b_{iji'j'} \leftarrow \nabla^2_{iji'j'}\Phi^{-1}(b_{iji'j'} - \nabla^2_{iji'j',qx}\Phi b_{qx})$
8: $\quad$ Return b

IV. Evaluation

We have evaluated our method on a single Intel Core Ultra 9 285K CPU and a single GeForce RTX 5090 GPU with 32GB of memory. All the computations are done on CPU except for differentiable rendering based texture optimization. We use multi-threading to compute evaluate different rows of the Jacobian. We propose to use the Levenberg-Marquardt algorithm, a form of the Quasi-Newton's method to serve as our subproblem solver Problem 6. This is because Problem 6 is essentially solving a least-square problem with $r(q, x)$ being the residual, due to the following rewrite:

$$O + \lambda^T C + \frac{\rho}{2}\|C\|^2 \propto O + \frac{\rho}{2}\|C + \lambda/\rho\|^2 \triangleq \|r(q, x)\|^2, \tag{19}$$

and noting that our objective function O (Equation 4) also takes a least square form (A similar argument applies to the frictional constraint part, which is omitted here for brevity). We terminate the subproblem solver when $\|r\|_\infty \le \epsilon_r$ or $\|r^T\partial r/\partial\big(x, q\big)\|_\infty \le \epsilon_g$. We find that due to the physics constraint C being very stiff, leading to the subproblem solver make very small progress over many iterations. We thus terminate the inner loop when the progress is less than 1% over 20 iterations. Finally, we terminate the outer ALM iteration when $\|C\|_\infty \le \epsilon_C$ or the residual of the KKT condition does not improve by more than 1% over consecutive ALM iterations. Through all our experiments, we choose parameters $\epsilon_r = 10^{-6}, \epsilon_g = 10^{-2}, \epsilon_C = 5 \times 10^{-4}$.

Scenario	Max Kinetic Energy (J) ↓		Max Drift Distance (cm) ↓	
	Ours	SAM3D	Ours	SAM3D
1	$\mathbf{7.19 \times 10^{-4}}$	4.77×10^{0}	**1.62**	59.41
2	$\mathbf{2.24 \times 10^{-3}}$	5.68×10^{0}	**0.83**	87.56
3	$\mathbf{1.12 \times 10^{-2}}$	2.08×10^{0}	**3.10**	31.06
4	$\mathbf{3.32 \times 10^{-3}}$	2.32×10^{0}	**0.73**	172.55
5	$\mathbf{4.36 \times 10^{-3}}$	5.73×10^{0}	**2.28**	51.69

TABLE I: We summarize the simulator stability by the amount of kinetic energy gain (left) during the first 1 second and drift distance (right) over the first 1 minute of simulation time.

Scenario	PSNR↑		
	Ours vs. RGB	SAM3D vs. RGB	Ours vs. SAM3D
1	17.92	**18.11**	20.16
2	**19.99**	18.99	22.32
3	**18.37**	17.34	18.70
4	**21.43**	21.11	23.95
5	20.15	**20.32**	20.17

TABLE II: We profile the PSNR between the images rendered using our estimated shape and pose (Ours), SAM3D estimated initial guess, which is further adjusted by FoundationPose (SAM3D), and the Groudtruth RGBD image (RGB).

a) Simulation-ready Results: All benchmarks are visualized in Figure 6. We evaluate our method on five cluttered tabletop scenes containing up to 5 rigid objects, represented by a total of 22 convex hulls. To assess the simulation-ready quality of our reconstructions, we forward the estimated object shapes and poses into the widely used MuJoCo physics simulator [33]. Under standard physical parameter settings (see Appendix for details), our reconstructions remain in force equilibrium over 1 minute of simulation time. In contrast, the initial shape and pose estimates produced by SAM3D [10] and FoundationPose [36] always contain severe inter-penetrations, causing simulation instability and failure, as reported in Table I. We also try three most recent single-view scene reconstruction works [4, 16, 1], yet none of them are able to produce comparable result (as illustrated in Appendix). We further compare visual fidelity against the initial SAM3D+FoundationPose estimates. As shown in Table II, our results achieve comparable PSNR, indicating that physical consistency is improved without sacrificing visual accuracy.

b) Performance: Benchmark statistics are summarized in Table III. Our algorithm converges within 6-9 ALM iterations, with a representative convergence trajectory shown in Figure 7. A detailed performance breakdown is provided

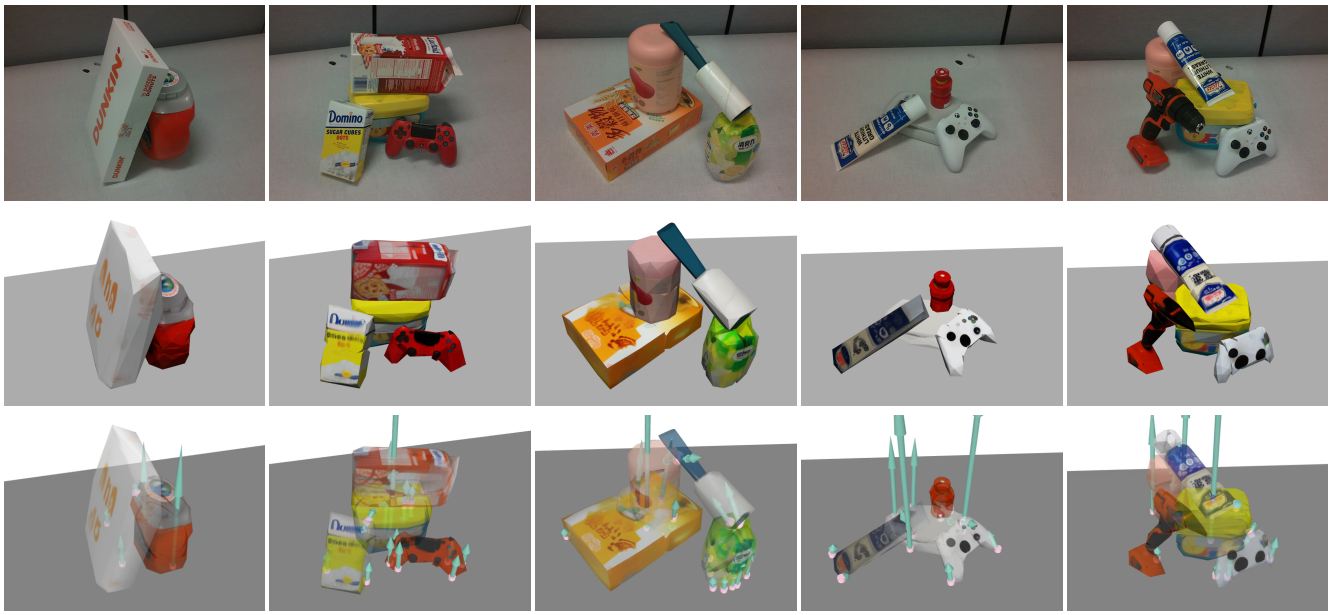

Fig. 6: A visualization of our benchmarking scenarios. Top Row: The input single-view image observation. Middle Row: The estimated simulation-ready rigid bodies models. Bottom Row: Our estimated scenes achieve physical force equilibrium in MuJoCo [33], showing simulated contact forces after settling.

Scenario	#Hull	#Vertex	#ALM	#LM	Wall Time (min)
1	6	299	7	2536	46.1
2	16	795	6	2322	259.7
3	10	487	9	4529	224.2
4	12	597	7	3106	202.15
5	22	1099	7	2640	539.92

TABLE III: We summarize the statistics of our benchmarking scenarios. From left to right: the total number of convex hulls, the total number of vertices, number of ALM outer iterations, number of LM iterations, and the total computational time.

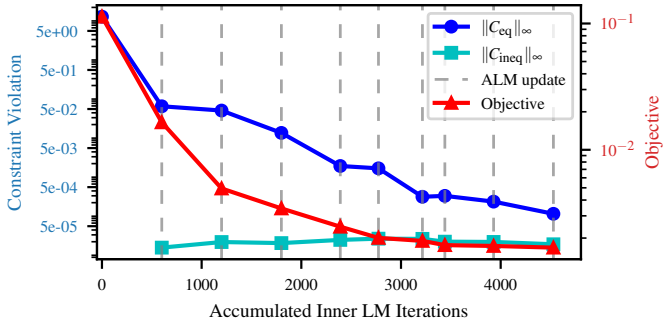

Fig. 7: The convergence history of a typical optimization procedure, which converges within 9 ALM iterations.

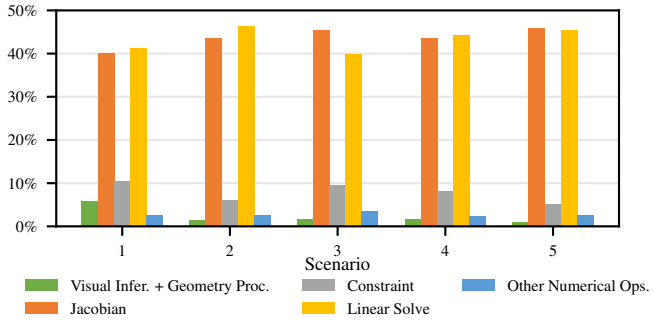

Fig. 8: Our method involves Visual (SAM3D, FoundationPose) inference & geometry process, constraint & Jacobian evaluation, linear solve, and other operations. This figure shows the performance breakdown in percentage.

Scenario	#Params	Woodbury (Alg. 1)	Direct LU	Speedup
1	2751	0.475	0.681	1.43×
2	12045	3.162	19.713	6.23×
3	7425	1.318	7.226	5.48×
4	9075	1.814	8.085	4.45×
5	19932	7.342	63.909	8.71×

TABLE IV: The average cost comparison of solving the structured linear system (in seconds) using our method (Algorithm 1) and direct LU factorization.

in Figure 8. The dominant computational cost arises from repeated evaluations of the physics constraint $\bar{C}$ and its Jacobian, which require solving nested separating-plane optimizations between convex hulls. The second most expensive component is the linear solve. We further evaluate our structured linear solver (Algorithm 1) and compare it against direct LU factorization in Table IV. The results demonstrate an overall speedup of up to 8.7×.

V. Conclusion

We present a real-to-sim scene estimation framework that reconstructs physically consistent, simulation-ready object shapes and poses from sparse observations in cluttered environments. Our core contribution is a joint physics-constrained

optimization formulated directly in the coupled shape-pose space. Building on the novel SDRS contact model, we enforce quasistatic force equilibrium and develop a structure-aware linear solver that enables efficient and stable optimization for large-scale scenes with many interacting objects. Integrated into a pipeline with learning-based initialization and differentiable texture refinement, our method robustly produces physically valid, simulation-ready reconstructions.

Our method open doors to several avenues of future work. The major limitation of our method is the high computational cost due to the extended decision variables parameterizing object shapes. We plan to utilize GPU to further reduce the computational overhead. Further, the SAM3D estimated object shapes can be rather inaccurate in cluttered scenes with severe occlusions. In future works, we plan to enable image-guided end-to-end real-to-sim optimization without relying on full mesh-based initial guess providied by SAM3D.

REFERENCES

[1] Aditya Agarwal, Gaurav Singh, Bipasha Sen, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Scenecomplete: Open-world 3d scene completion in cluttered real world environments for robot manipulation. *IEEE Robotics and Automation Letters*, 11(1):482–489, 2026.

[2] M. Aiger, D. Cohen-Or, and D. Levin. A global registration method for 3d point clouds. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, pages 211–220. Eurographics Association, 2008. doi: 10.2312/SGP/SGP08/211-220.

[3] Brian Amberg, Sami Romdhani, and Thomas Vetter. Optimal step nonrigid icp algorithms for surface registration. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[4] Andreea Ardelean, Mert Özer, and Bernhard Egger. Generalizable 3d scene reconstruction via divide and conquer from a single view. In *International Conference on 3D Vision (3DV)*, 2025.

[5] Marco Attene. A lightweight approach to repairing digitized polygon meshes. *The visual computer*, 26(11): 1393–1406, 2010.

[6] Dominik Bauer, Timothy Patten, and Markus Vincze. Verefine: Integrating object pose verification with physics-guided iterative refinement. *IEEE Robotics and Automation Letters*, 5(3):4289–4296, 2020.

[7] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[8] Bibit Bianchini, Minghan Zhu, Mengti Sun, Bowen Jiang, Camillo J. Taylor, and Michael Posa. Vysics: Object reconstruction under occlusion by fusing vision and contact-rich physics. In *Robotics: Science and Systems (RSS)*, june 2025.

[9] Will Chang and Matthias Zwicker. Automatic registration for articulated shapes. In *Computer Graphics Forum*, volume 27, pages 1459–1468. Wiley Online Library, 2008.

[10] Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, et al. Sam 3d: 3dfy anything in images. *arXiv preprint arXiv:2511.16624*, 2025.

[11] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Krsek. The trimmed iterative closest point algorithm. In *2002 International Conference on Pattern Recognition*, volume 3, pages 545–548. IEEE, 2002.

[12] 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.

[13] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Rev.*, 47:99–131, 2005.

[14] Joachim Hilgert and Karl-Hermann Neeb. *Structure and geometry of Lie groups*. Springer Science & Business Media, 2011.

[15] Jerry Hsu, Nghia Truong, Cem Yuksel, and Kui Wu. A general two-stage initialization for sag-free deformable simulations. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2022)*, 41(4):64:1–64:13, 07 2022. ISSN 0730-0301. doi: 10.1145/3528223.3530165.

[16] Zehuan Huang, Yuan-Chen Guo, Xingqiao An, Yunhan Yang, Yangguang Li, Zi-Xin Zou, Ding Liang, Xihui Liu, Yan-Pei Cao, and Lu Sheng. Midi: Multi-instance diffusion for single image to 3d scene generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 23646–23657, 2025.

[17] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020.

[18] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020.

[19] Emmett Lalish and contributors. Manifold: Geometry library for topological robustness, 2025.

[20] Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. Incremental potential contact: Intersection- and inversion-free large deformation dynamics. *ACM Transactions on Graphics (SIGGRAPH)*, 39(4), 2020. doi: 10.1145/3386569.3392425.

[21] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019.

[22] Pingchuan Ma, Tao Du, Joshua B Tenenbaum, Wojciech Matusik, and Chuang Gan. RISP: Rendering-invariant state predictor with differentiable simulation and rendering for cross-domain parameter estimation. In *International Conference on Learning Representations*, 2021.

[23] Martin Malenickỳ, Martin Cífka, Médéric Fourmy, Louis

Montaut, Justin Carpentier, Josef Sivic, and Vladimir Petrik. Physpose: Refining 6d object poses with physical constraints. *arXiv preprint arXiv:2503.23587*, 2025.

[24] Rhys Newbury, Jack Collins, Kerry He, Jiahe Pan, Ingmar Posner, David Howard, and Akansel Cosgun. A review of differentiable simulators. *IEEE Access*, 2024.

[25] Junfeng Ni, Yixin Chen, Bohan Jing, Nan Jiang, Bin Wang, Bo Dai, Puhao Li, Yixin Zhu, Song-Chun Zhu, and Siyuan Huang. Phyrecon: Physically plausible neural scene reconstruction. *Advances in Neural Information Processing Systems*, 37:25747–25780, 2024.

[26] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 2006.

[27] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

[28] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[29] R. Tyrrell Rockafellar. *Convex analysis*. Princeton Mathematical Series. Princeton University Press, Princeton, N. J., 1970.

[30] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[31] Mikhail V Solodov. Global convergence of an sqp method without boundedness assumptions on any of the iterative sequences. *Mathematical programming*, 118(1): 1–12, 2009.

[32] Changkyu Song and Abdeslam Boularias. Inferring 3d shapes of unknown rigid objects in clutter through inverse physics reasoning. *IEEE robotics and automation letters*, 4(2):201–208, 2018.

[33] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[34] Andreas Wächter and Laurence T. Biegler. On the implementation of a primal-dual interior point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[35] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. *ACM Transactions on Graphics (TOG)*, 41(4):1–18, 2022.

[36] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024.

[37] Hongchi Xia, Chih-Hao Lin, Hao-Yu Hsu, Quentin Leboutet, Katelyn Gao, Michael Paulitsch, Benjamin Ummenhofer, and Shenlong Wang. Holoscene: Simulation-ready interactive 3d worlds from a single video. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[38] 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.

[39] Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An End-to-End Differentiable Framework for Contact-Aware Robot Design. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi: 10.15607/RSS.2021.XVII.008.

[40] Kaixin Yao, Longwen Zhang, Xinhao Yan, Yan Zeng, Qixuan Zhang, Lan Xu, Wei Yang, Jiayuan Gu, and Jingyi Yu. Cast: Component-aligned 3d scene reconstruction from an rgb image. *ACM Transactions on Graphics (TOG)*, 44(4):1–19, 2025.

[41] Xiaohan Ye, Xifeng Gao, Kui Wu, Zherong Pan, and Taku Komura. Sdrs: Shape-differentiable robot simulator. *IEEE Transactions on Robotics*, pages 1–20, 2025.

[42] Xiaohan Ye, Kui Wu, Zherong Pan, and Taku Komura. Efficient differentiable contact model with long-range influence. *arXiv preprint arXiv:2509.20917*, 2025.

[43] Jonathan Young. xatlas, 2019.

[44] Mengchao Zhang and Kris Hauser. Semi-infinite programming with complementarity constraints for pose optimization with pervasive contact. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6329–6335, 2021. doi: 10.1109/ICRA48506.2021.9561609.

[45] Mengchao Zhang, Devesh K Jha, Arvind U Raghunathan, and Kris Hauser. Simultaneous trajectory optimization and contact selection for contact-rich manipulation with high-fidelity geometry. *IEEE Transactions on Robotics*, 41:2677–2690, 2025.

[46] Yifan Zhu, Tianyi Xiang, Aaron M Dollar, and Zherong Pan. One-shot real-to-sim via end-to-end differentiable simulation and rendering. *IEEE Robotics and Automation Letters*, 2025.

APPENDIX

In this appendix, we present the complete details of our methods, as well as several additional results. First in Appendix A, we provide the complete detail of our ALM-based optimizer. Then in Appendix B, we present details of our geometric processing algorithms for generating the valid initial guess. Finally, in Appendix C, we show more examples and baseline comparisons.

APPENDIX A
OPTIMIZATION DETAILS

A. Complete ALM formulation

In this section, we expand the frictional ALM subproblem in Equation 15 by explicitly defining the per-contact augmented term $\Phi(q, x, f_{iji'j'})$, and illustrate our complete algorithmic flow to solve the ALM problem. Recall that for each interacting convex-hull pair $\left(ij, i'j'\right)$ and each vertex X_{ijk}, the normal force $f^{\perp}_{ijk,i'j'}$ is a function of shape x and pose q according to Equation 10, and we introduce per-vertex tangential friction force variables $f^{\parallel}_{ijk,i'j'}$. We use the vector $f^{\parallel}_{iji'j'}$ to denotes all force terms $\{f^{\parallel}_{ijk,i'j'}\}_k$ stacked together. For brevity, we denote $z \triangleq \left(q, x, \cdots, f^{\parallel}_{iji'j'}, \cdots\right)$. With such notation, our formulation consists of 4 kinds of frictional contact constraints for our scene estimation problem:

1) Force equilibrium for objects:

$$C_{\text{equi}}(z) \triangleq \bar{C}(z) = \nabla_q \left[\Psi(q, x) - \sum_{i \neq i'} \sum_{j,j'} \sum_{k} \langle X_{ijk}, f^{\parallel}_{ijk,i'j'} \rangle \right] = 0,$$

2) Orthogonality:

$$C^{iji'j'}_{\text{orth}}(z) \triangleq \begin{pmatrix} \vdots \\ \left\langle f^{\perp}_{ijk,i'j'},\, f^{\parallel}_{ijk,i'j'} \right\rangle \\ \vdots \end{pmatrix} = 0,$$

3) Friction cone:

$$C^{iji'j'}_{\text{cone}}(z) \triangleq \begin{pmatrix} \vdots \\ \|f^{\parallel}_{ijk,i'j'}\| - \eta \|f^{\perp}_{ijk,i'j'}\| \\ \vdots \end{pmatrix} \leq 0,$$

4) Tangential force equilibrium for separating plane:

$$C^{iji'j'}_{\text{plane}}(z) \triangleq \begin{pmatrix} \sum_k f^{\parallel}_{ijk,i'j'} + \sum_{k'} f^{\parallel}_{i'j'k',ij} \\ T_{iji'j'}\left[\sum_k X_{ijk} \times f^{\parallel}_{ijk,i'j'} + \sum_{k'} X_{i'j'k'} \times f^{\parallel}_{i'j'k',ij}\right] \end{pmatrix} = 0.$$

Note that we redefine $\bar{C}$ as C_{equi} for more conveniently distinguishing different types of constraints. We then compactly group the constraints as follows:

$$C_{\text{eq}}(z) \triangleq \begin{pmatrix} C_{\text{equi}}(z) \\ \{C^{iji'j'}_{\text{orth}}(z)\}_{ij,i'j'} \\ \{C^{iji'j'}_{\text{plane}}(z)\}_{ij,i'j'} \end{pmatrix}$$

$$C_{\text{ineq}}(z) \triangleq \{C^{iji'j'}_{\text{cone}}(z)\}_{ij,i'j'}.$$

To handle inequality constraint in the Augmented Lagrangian formulation, following [7], we introduce the element-wise clamp operation by defining $[u]_+ \triangleq \max(u, 0)$ and $\hat{C}_{\text{ineq}}(z) \triangleq [C_{\text{ineq}}(z)]_+$. Let λ_{eq} and λ_{ineq} denote Lagrange multipliers for equality and inequality constraint, respectively. We further define the corresponding penalty parameter be $\rho_{\text{eq}} > 0$ and $\rho_{\text{ineq}} > 0$. (We define similarly for constraint groups $\left(\lambda_{\text{equi}}, \rho_{\text{equi}}\right)$, $\left(\lambda^{iji'j'}_{\text{orth}}, \rho^{iji'j'}_{\text{orth}}\right)$ etc.). The complete Augmented Lagrangian subproblem takes the following form:

$$\underset{z}{\text{argmin}} \begin{cases} O(z) + \\ \lambda^T_{\text{eq}} C_{\text{eq}}(z) + \frac{\rho_{\text{eq}}}{2} \|C_{\text{eq}}(z)\|^2 + \\ \lambda^T_{\text{ineq}} \hat{C}_{\text{ineq}}(z) + \frac{\rho_{\text{ineq}}}{2} \|\hat{C}_{\text{ineq}}(z)\|^2. \end{cases} \tag{A.1}$$

For each interacting pair $\left(ij, i'j'\right)$ of convex hulls, define the pairwise equality and inequality constraint:

$$C^{iji'j'}_{\text{eq}}(z) \triangleq \begin{pmatrix} C^{iji'j'}_{\text{orth}}(z) \\ C^{iji'j'}_{\text{plane}}(z) \end{pmatrix} \qquad \hat{C}^{iji'j'}_{\text{ineq}}(z) \triangleq [C^{iji'j'}_{\text{cone}}(z)]_+.$$

Then Problem A.1 can be equivalently written as:

$$\mathcal{L}(z) = O(z) + \lambda^T_{\text{equi}} C_{\text{equi}}(z) + \frac{\rho_{\text{eq}}}{2} \|C_{\text{equi}}(z)\|^2 + \sum_{ij,i'j'} \Phi^{iji'j'}(z),$$

which is exactly Equation 15 in our main paper with the per-pair augmented term:

$$\Phi^{iji'j'}(z) \triangleq \begin{cases} (\lambda^{iji'j'}_{\text{eq}})^T C^{iji'j'}_{\text{eq}}(z) + \frac{\rho_{\text{eq}}}{2} \|C^{iji'j'}_{\text{eq}}(z)\|^2 + \\ (\lambda^{iji'j'}_{\text{ineq}})^T C^{iji'j'}_{\text{ineq}}(z) + \frac{\rho_{\text{ineq}}}{2} \|\hat{C}^{iji'j'}_{\text{ineq}}(z)\|^2. \end{cases} \tag{A.2}$$

Still, since our visual objective takes a least-square form, i.e., we can rewrite $O(z)$ by finding $r_O(z)$ satisfying $\|r_O(z)\|^2 = O(z)$, solving the complete ALM Problem A.1 is equivalent to minimizing $\|r(z)\|^2$ with:

$$r(z) \triangleq \begin{pmatrix} r_O(z) \\ \sqrt{\rho_{\text{eq}}}\left(C_{\text{eq}}(z) + \lambda_{\text{eq}}/\rho_{\text{eq}}\right) \\ \sqrt{\rho_{\text{ineq}}}\left(\hat{C}_{\text{ineq}}(z) + \lambda_{\text{ineq}}/\rho_{\text{ineq}}\right) \end{pmatrix},$$

and we solve the least-square problem using the LM algorithm with our structure aware linear solver (Algorithm 1 in the main paper) that exploit the low rank and pairwise sparsity. The ALM solving process is summarized in Algorithm 3 below.

We use a unified optimization parameter setting across all the scenarios illustrated in the main paper and Appendix C. We set visual objective weights to be $w_1 = 2 \times 10^{-2}, w_2 = 10^{-1}, w_3 = 2 \times 10^{-2}$, the complementarity gap is chosen to be a small value $\mu = 5 \times 10^{-5}$ and the frictional coefficient $\eta = 1$. We terminate the subproblem LM solver when $\|r(z)\|_\infty \leq \epsilon_r$ or $\|r^T \partial r / \partial z\|_\infty \leq \epsilon_g$. We find that, due to the physics constraint being very stiff, leading to the subproblem solver occasionally make very small progress over many iterations. We thus terminate the inner loop when the progress is less than 1% over 20 iterations. Finally, we terminate the outer ALM iteration when $\|C_{\text{eq}}\|_\infty \leq \epsilon_{C\text{eq}}$ and $\|C_{\text{ineq}}\|_\infty \leq \epsilon_{C\text{ineq}}$ or the residual of the KKT condition does not improve by more than 1% over consecutive ALM iterations. Through all our experiments, we choose parameters $\rho^{\text{init}}_{\text{eq}} = 10^{-2}, \rho^{\text{init}}_{\text{ineq}} = 2, \lambda^{\text{init}}_{\text{eq}} = \lambda^{\text{init}}_{\text{ineq}} = 0, \gamma_{\text{eq}} = \gamma_{\text{ineq}} = 0.25, \beta_{\text{eq}} = 8, \beta_{\text{ineq}} = 4, \epsilon_r = 10^{-6}, \epsilon_g = 10^{-2}, \epsilon_{C\text{eq}} = \epsilon_{C\text{ineq}} = 5 \times 10^{-4}$.

Algorithm 3 ALM-based Joint Shape and Pose Optimization

Input: Initial guess $z \triangleq (x, q, f^{\parallel}_{iji'j'})$; initial multiplier λ_{eq}, λ_{ineq}; initial penalty $\rho_{\text{eq}}, \rho_{\text{ineq}} \in (0, \infty)$; $\gamma_{\text{eq}}, \gamma_{\text{ineq}} \in (0, 1)$; $\beta_{\text{eq}}, \beta_{\text{ineq}} \in (1, \infty)$
Output: Locally optimal z

$O^{\text{prev}} \leftarrow \infty$
while Not converged **do**
  ▷ ICP-type closest point update
  for Each convex hull vertex X_{ijk} **do**
    Compute and fix $p(X_{ijk})$ (Equation 2)
  for Each $p_{il} \in \mathcal{P}_i$ and $p_{il} \in \mathcal{M}_i$ **do**
    Compute $X(p_{il})$ (Equation 3)
    Compute Δ_{il} by fixing w_k (Equation 5)
  Evaluate $O(z)$ (Equation 4)
  ▷ Heuristic to ensure function value decrease
  while $O(z) > O^{\text{prev}}$ **do**
    Find $X(p_{il})$ with largest Δ_{il}
    Exclude $\|X(p_{il}) - p_{il}\|^2$ from O
  $z^{\text{prev}} \leftarrow z$
  ▷ LM method with linear solver (Algorithm 1)
  Solve Problem A.1 to update z
  ▷ Update multiplier as in [7]
  $\lambda_{\text{eq}} \leftarrow \lambda_{\text{eq}} + \rho_{\text{eq}} C_{\text{eq}}$, $\lambda_{\text{ineq}} \leftarrow \lambda_{\text{ineq}} + \rho_{\text{ineq}} \hat{C}_{\text{ineq}}$
  ▷ Schedule penalty as in [7]
  if $\|C_{\text{eq}}(z)\|_\infty \geq \gamma_{\text{eq}} \|C_{\text{eq}}(z)\|_\infty$ **then**
    $\rho_{\text{eq}} \leftarrow \beta_{\text{eq}} \rho_{\text{eq}}$
  if $\|\hat{C}_{\text{ineq}}(z)\|_\infty \geq \gamma_{\text{ineq}} \|\hat{C}_{\text{ineq}}(z)\|_\infty$ **then**
    $\rho_{\text{ineq}} \leftarrow \beta_{\text{ineq}} \rho_{\text{ineq}}$
  $O^{\text{prev}} \leftarrow O(z)$

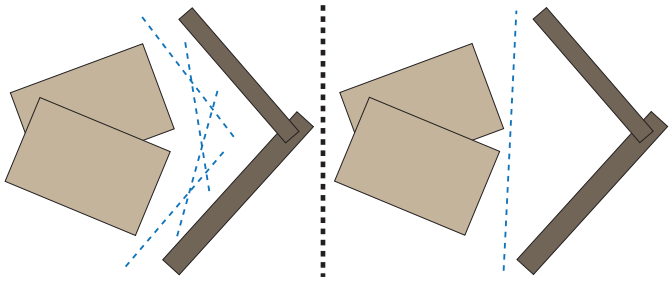

Fig. 9: Two non-convex objects separated using SDRS contact model, each consisting of two convex hulls. Left: Without aggregation, we need 4 separating planes. Right: With aggregation, we only introduce a single separating plane.

B. Separating Plane Aggregation

Our scene estimation problem is inherently ill-posed: we seek to recover complete 3D object shapes from a single RGB-D observation. Addressing this challenge requires introducing a large number of decision variables, which in turn gives rise to numerous feasible local minima. As a result, the optimizer is free to converge to arbitrary solutions, particularly in occluded regions where shape evidence is absent. In practice, this freedom is often undesirable and commonly manifests as noisy or irregular geometry on occluded contact surfaces. To mitigate this issue, we propose separating plane aggregation, a simple yet effective regularization technique that encourages the recovery of smooth object shapes. Although the contact model of [41] conceptually inserts a separating plane between each pair of convex hulls, this assumption is not strictly necessary. In fact, a separating plane can be introduced between arbitrary non-convex shapes, in which case SDRS implicitly treats each shape as its convex hull. Leveraging this observation, we introduce a single separating plane between each pair of interacting objects, rather than multiple planes between all pairs of their constituent convex hulls, as shown in Figure 9. This aggregation effectively regularizes occluded contact regions by encouraging them to align with a common separating plane, resulting in smoother and more physically plausible contact surfaces. We emphasize that, even with the separating plane aggregation, we are still representing each object as multiple convex hulls and the vertices of each convex hull is treated as separate decision variables to be optimized. This is necessary to match arbitrarily non-convex object shapes.

APPENDIX B
VISUAL INFERENCE & GEOMETRY PROCESS

In this section, we explain the details of the visual inference pipeline and geometry processing for initialization of our physics-aware optimization.

A. Visual Inference

Given an RGBD image of the scene, we first use SAM2 [28] to generate the mask for each object in the scene from a user selected bounding box. We then use SAM3D [10] to generate an initial geometry for each object, conditioned on the RGBD image and the mask. The output mesh might not be watertight and has excessively high resolution, which slows down geometric processing. Therefore, we simplify and fix the mesh using PyMeshFix [5] and re-textured the mesh using xatlas [43] and Nvdiffrast [18]. As mentioned in the main content, the poses given by SAM3D are oftentimes inaccurate. To further correct them, we use FoundationPose [36] to register the textured mesh to the RGBD image.

The point clouds derived from the RGBD camera are prone to noisy outliers. We use the SAM3D generated mesh and the registered pose to filter the noisy segmented point cloud used in the optimization. For each object i, we remove any point with segmentation label i if its distance to the mesh of the object i is greater than 0.01m. We include the table in the scene as a static object, i.e. it's pose and shape are not optimized but its contact interactions are accounted in constraint. We fit the upper surface of the table using the observed point cloud. The gravitational direction is assumed to be aligned with the normal of the table upper surface. We further define the body frame of each object using PCA. For each object, we compute its initial translation by the center of its vertices.

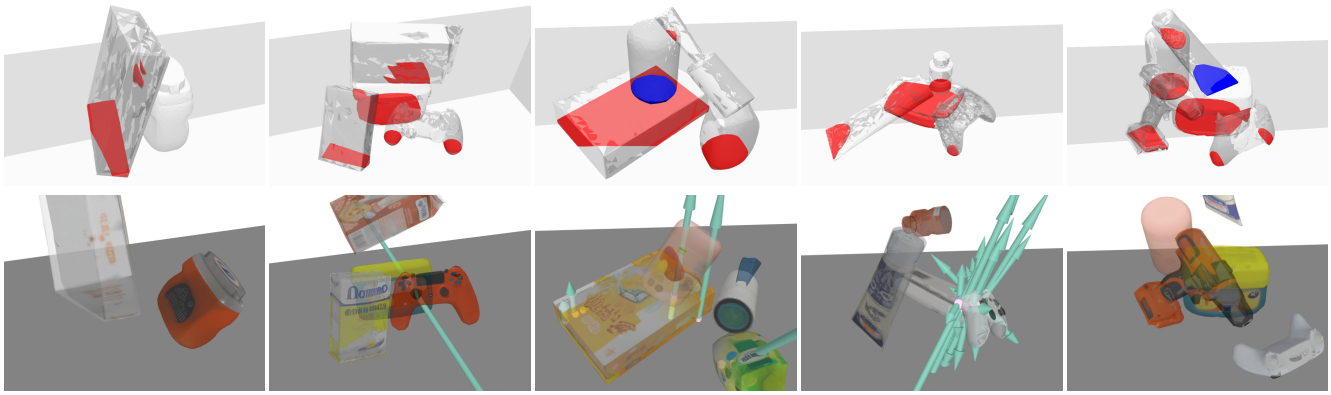

Fig. 10: We illustrate various violations to physical constraints induced by visual inference. Top: penetrations (red) and floating objects (blue) always occur as SAM3D and FoundationPose do not explicitly enforce physics constraints. Bottom: These violations always lead to simulation blow up in MuJoCo.

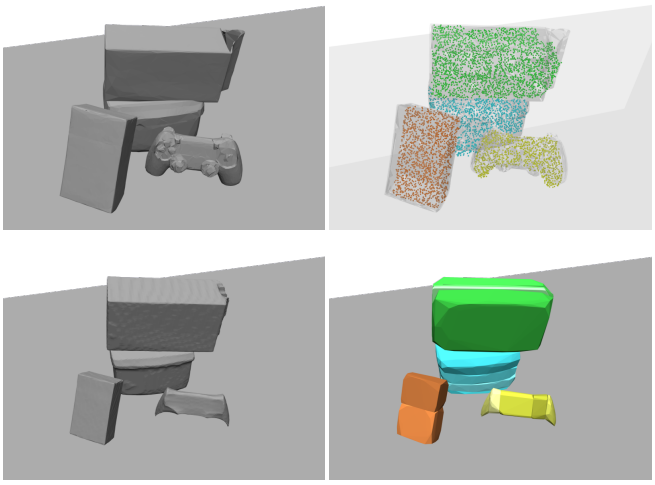

Fig. 11: An illustration of our visual inference and geometry process pipeline. Top Left: Initial estimation of each object mesh; Top Right: Filtered point cloud; Bottom Left: Penetrating free mesh after shrinkage; Bottom Right: Convex decomposition.

The initial orientation is defined by PCA analysis extracting principal axes.

Since mass is ambiguous from only RGBD input, we manually assign the total mass of each object. For each object, we compute the volume of the mesh given by SAM3D and the total mass of the object is set to be the product of density and volume. The density of the objects across all examples is set to be $800\,(\mathrm{kg/m^3})$ (the average density for materials like wood and plastic). Note that the assigned per-vertex mass is just an initial guess and our optimizer is allowed to fine-tune the mass and inertia properties.

B. Penetration-free Initialization

The SDRS contact model we adopt is based on the interior point principle, which requires a strictly feasible, i.e., penetration-free initialization. Unfortunately, this is not guaranteed by our visual inference as shown in Figure 10. Therefore, we adopt the following process to resolve penetrations for initialization of the optimization process. For each colliding mesh pair in the initial estimate, we shrink both objects by extracting the isosurface with the SDF value $-d$, where $d > 0$ is the minimum shrinkage distance for the object pair to become penetration-free. To make this process more robust for thin objects, the SDF is computed in the normalized space, where the object bounding box is normalized to $[0, 1]^3$.

After the penetration is resolved, we use CoACD [35] to decompose the mesh of each object into unions of convex hulls. CoACD provides various options to control the fidelity of the decomposed result. For performance consideration, we set `max-convex-hull`$= 5$, `max-ch-vertex`$= 50$, so that each body is constrained to be decomposed into at most 5 hulls, with each hull having at most 50 vertices. The concavity `threshold`$= 0.05$ which is the recommended default value to capture geometry fidelity, other parameters are set to default values as well. An illustration of the visual inference and geometry process pipeline is shown in Figure 11.

Appendix C
Additional Results & Baeline Comparisons

In this section, we present our detailed MuJoCo simulator setup and present additional results.

A. MuJoCo Simulation Setting

To set up a simulation-ready scene in MuJoCo [33], we specify the simulator parameters and the states and properties of rigid bodies as follows.

1) Simulator Parameter Setting: For a complete list of simulator related parameters, we refer the readers to the official documentation. Here we explain the critical settings that differ from the default values: the simulator `timstep`$= 10^{-3}$, and the `integrator` is set to "implicit" to enhance simulation stability for the cluttered scene. Moreover, MuJoCo's contact model by default induces gradual slippage phenomenon and

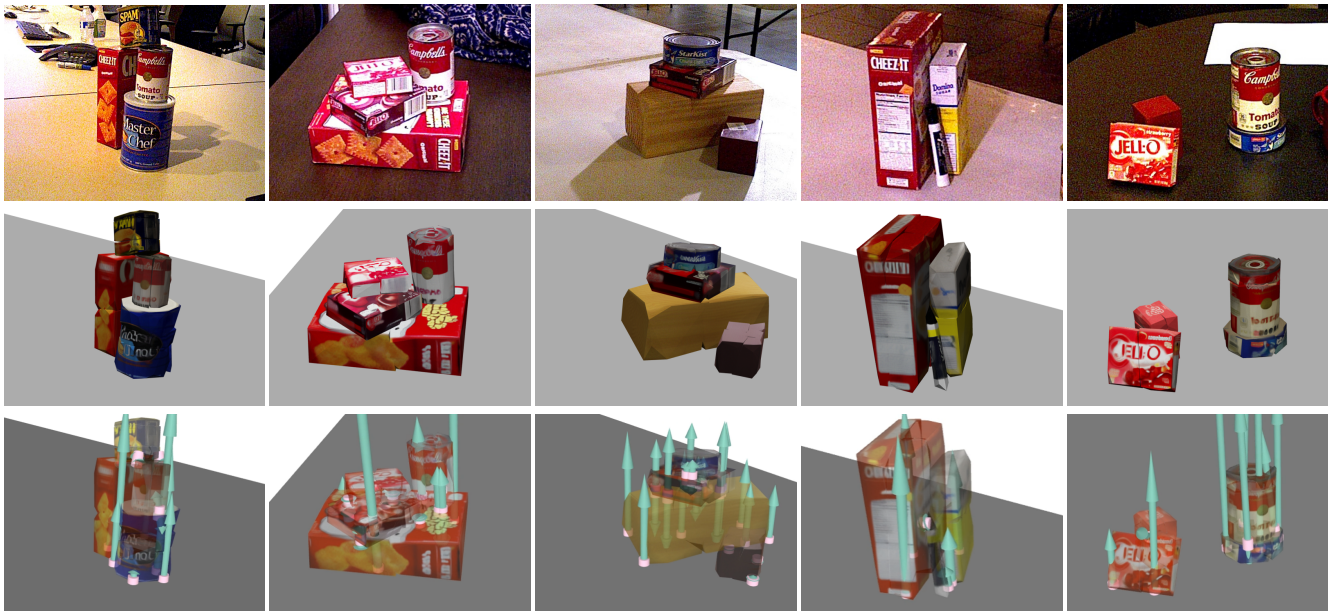

Fig. 12: A visualization of our additional examples on scenes from the YCB-V dataset. Top Row: The input single-view image observation. Middle Row: The estimated simulation-ready rigid bodies models. Bottom Row: Our estimated scenes achieve physical force equilibrium in MuJoCo [33], where we visualized the contact forces.

`noslip_iterations`= 10 is set to prevent such slippage artifact. For the collision detection and contact related settings, we turn on `multi_ccd` to enable the collision detection pipeline to return multiple contact points for a single collision geometry, the `cone` option is set to "elliptic" to match our friction cone formulation and we use `condim`= 4, which provide additional torsional friction around contact normal. This is also recommended by the official documentation to prevent simulation instability and drifting in multi contact scenario. We set the `friction` (coefficient) to be $[1, 0.005]$ to match our optimization parameter setting and only provide minimal torsional friction (default value).

2) Baseline Processing: For comparison, we simulate the scene estimated using various visual inference pipelines, including the SAM3D + FoundationPose pipeline and two baselines below. The results of these scene estimators are using mesh-based object representation. For each of the estimated rigid bodies in the scene, we replace their collision mesh by a high-fidelity convex decomposition using CoACD [35]. This is for fair comparison and recommended by the official documentation of MuJoCo. The visual geometry is the textured geometry returned by the pipelines. The mass and inertia tensor is auto calculated by MuJoCo using it's internal scheme with our assigned density. Internally MuJoCo performs volume based mass calculation as we do and we confirm that the numerical difference is negligible. The pose is also set based on the output of the pipelines. An example of the simulation result is shown in Figure 10 bottom row.

3) Optimized Result Processing: We discuss details for loading results from our optimization result to MuJoCo. For each of the rigid bodies in the scene, it's collision geometry is the unions of convex hulls extracted from the optimization result x, while for visual geometry, we used the triangular mesh merged from x using Manifold3d Library [19] with the texture optimized by differentiable rendering [18]. The inertia tensor is set based on our mass model instead of the default auto-calculation done by MuJoCo, and we set the poses of the objects according to our optimization result q.

B. Additional Examples

In addition to the five self-created cluttered scenes illustrated in our main paper, we provide additional five examples to show the generality of our pipeline. Specifically, we collect five single view images from the YCB-V dataset [38], where the selected scenarios are cluttered with complicated contact relationship. We use the same parameter setting described above. We include the performance statistic of these five examples in Table V and the results are visualized in Figure 12.

Scen.	#Hull	#Vertex	#Params	#ALM	#LM	Wall Time
6	14	698	10590	7	2844	200.11
7	14	695	10545	7	1643	148.48
8	14	694	10485	7	1936	140.58
9	6	299	3678	6	2480	36.13
10	15	746	11310	6	1083	104.48

TABLE V: The statistics of the additional examples. From left to right: the total number of convex hulls, the total number of vertices, number of parameters in linear solve, number of ALM outer iterations, number of LM iterations, and the total computational time in minutes.

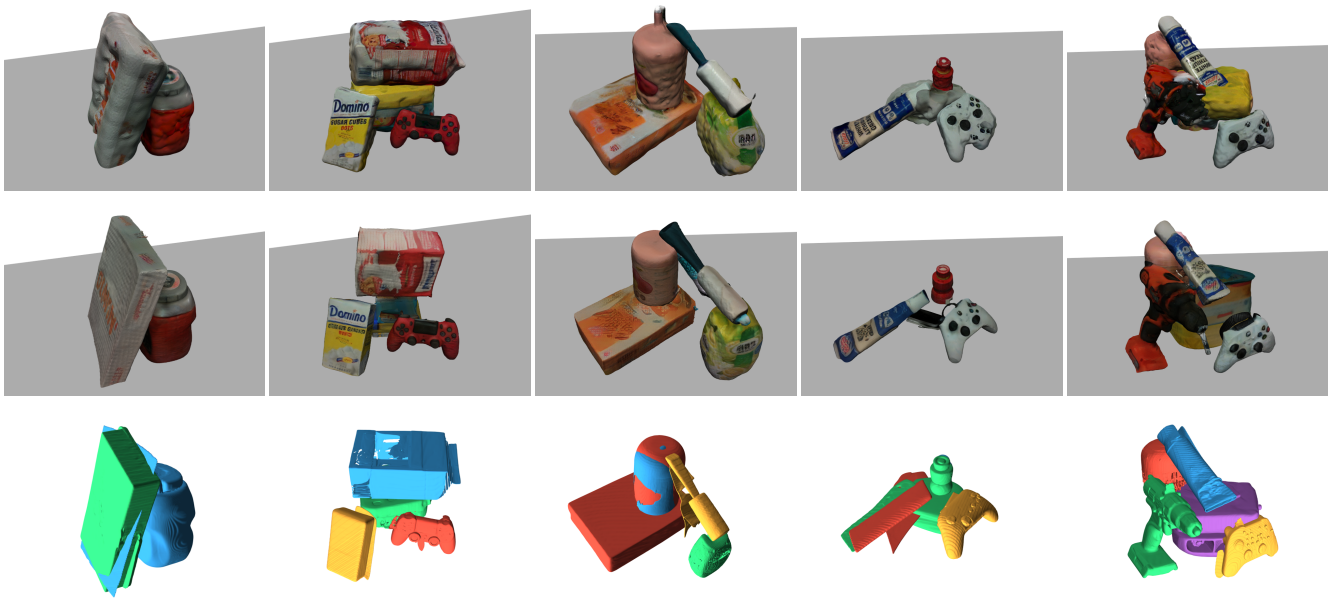

Fig. 13: A visualization of the baseline methods. Top row: Gen3DSR [4], middle row: SceneComplete [1], bottom row: MIDI [16] (with different reconstructed objects visualized in different colors)

C. Baseline Comparisons

Here we include additional comparison with three most recent single-view scene reconstruction methods, including Gen3DSR [4], SceneComplete [1] and MIDI [16]. We directly use the open source implementation of the works with their default model, parameter, and weight settings, while making the following adjustments for our examples. First, we observe that Gen3DSR and SceneComplete use their own automatic segmentation and masking pipeline through entity/background segmentation or vision language model prompt. These pipelines often suffer from segmentation failure such as missing object, mixing multiple objects into a single one, when facing the highly cluttered and occluded scene. For fair comparisons, we enhance their pipeline using the same well-segmented masks that are created and used in our pipeline with user bounding box selection. MIDI has the same user bounding box selection interface as we do, hence we direct use its native interface. In addition, these baselines rarely take into considerations the static environment reconstruction, i.e. table-top surface in our examples, because they are not physically-aware. Gen3DSR can reconstruct environment, yet often falsely reconstructs the table as another bulky object (e.g. a huge bed) in the scene. SceneComplete does not reconstruct the environment, it only focuses on the objects in the scene. Since these two works take RGBD image as input, we can manually assign the table top surface fitted from the point cloud for them as we do for our pipeline. However, MIDI takes only RGB image as input, and it does not provide or estimate camera information. It is also unable to segment or reconstruct the table. Therefore, the reconstructed result is ambiguous in depth and object size, and we are unable to align and set up the static scene for it. Moreover, we are unable to get texture of the scene for MIDI due to computation resource limit. We try our best to align the reconstruction result for these baselines and show them in Figure 13. Regretfully, all of these baselines generate meshes that have severe penetrations, leading to immediate simulator blow up.