2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

One-Shot View Planning and Online Optimization-based Replanning for Unknown Object Reconstruction

José J. Patiño, Zachary Kingston, Victor Romero-Cano, Yu-Kun Lai, and Juan David Hernández

***Abstract*— Robotic inspection tasks often require constructing high-quality 3D models of objects from a minimal number of views. Traditional next-best view planning (NBVP) approaches incrementally select view poses but fail to account for global optimality of the inspection trajectory, thus leading to inefficient inspection paths. Recent one-shot view planning (OSVP) methods address this challenge by predicting informative view poses from an initial observation. While subsequent improvements on the pioneering OSVP approach attempt to improve prediction accuracy, they can still fail when faced with out of distribution (OoD) examples. With recent advances in generative modeling, OSVP methods can infer a plausible object shape from one observation and then derive the corresponding solution set of view poses. However, because the predicted shape may deviate from the true geometry, these methods can still generate infeasible views. To overcome these limitations, we propose a novel OSVP framework that leverages RGB-D data to generate geometric priors and incorporates online video-based reconstruction. Our method formulates viewpoint selection and path optimization, so that both the calculated poses and the connecting trajectories satisfy visibility constraints, maintain smoothness, and can be locally replanned to compensate for discrepancies between predicted and real object geometries. We validate our OSVP approach through simulation benchmarks against state-of-the-art OSVP techniques and demonstrate its effectiveness on a real Franka Emika manipulator.**

I. Introduction

In different applications, robots are equipped with cameras that allow them to inspect objects and create their 3D models [1]. These 3D models play an important role in diverse domains, such as quality control [2], preservation of cultural heritage [3] and additive manufacturing [4]. In all these applications, a common problem consists of determining the most efficient set of view poses at which the robot must position the camera, so that enough data can be gathered to create a complete reconstruction of a desired quality in minimum time, thus the most informative shots are desired. Such a problem is commonly referred to as view planning (VP) [5].

In the absence of prior information about the object, a VP approach, which is commonly referred to as next-best view planning (NBVP), consists in iteratively determining the next view pose as the object is incrementally inspected [6, 7]. However, the main limitation of NBVP methods is that they only select the immediately subsequent view poses, which does not allow them to consider the complexity of the full continuous trajectory. Furthermore, it is also difficult to incorporate global budget constraints, such as maximum allocated time or energy. These limitations often lead to suboptimal movements to complete the object's inspection.

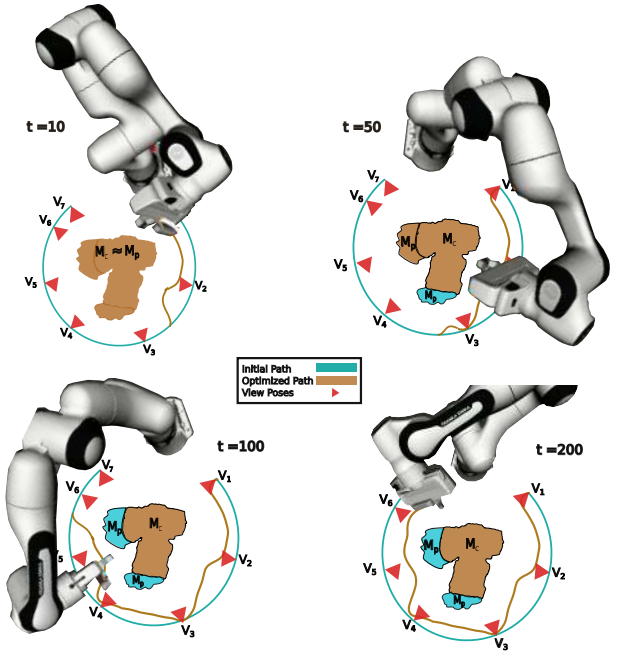


Fig. 1: Our approach produces a constraint-optimized path (shown in brown), starting from an initial path (turquoise). The initial path is guided by the solution set of views (red triangles) obtained by solving the OSVP problem, taking as input a 3D geometric prior (M_p) constructed from an initial view pose, whereas the optimized path incorporates online observations (M_c) that capture the actual geometry of the object.

Recent work has introduced one-shot view planning (OSVP), which is an alternative VP approach that aims to overcome such optimality limitations [8–10]. OSVP uses an initial observation of the object (e.g., an RGB image) to predict a collection of informative view poses. Such predicted view poses are then used to calculate the shortest path to traverse them. One main limitation of OSVP methods is that they rely on prior knowledge of the object, which might not be available when inspecting an unknown object. To work around this requirement, a geometric prior can be obtained from generative models, e.g., an RGB-based 3D diffusion model [9]. The generated geometric prior can then be used to hypothesize about the areas of the object that are not captured in the initial shot, thus allowing the planner to estimate the location of the additional view poses, which are required to inspect the whole object.

However, using a diffusion model to generate geometric

This work has been supported by the School of Computer Science & Informatics at Cardiff University.

J.J. Patiño, V. Romero-Cano, Y. Lai and J.D. Hernández are with the School of Computer Science and Informatics at Cardiff University, UK.

Z. Kingston is with the Computer Science Department at Purdue University, USA.

priors has two main limitations. Since the diffusion model relies on an RGB image, the predicted geometry may differ from the real object's scale [9]. Moreover, because the geometric priors are inferred, the estimated shape can deviate from the real object's shape for irregular objects (e.g., plants) or categories outside the training set. Such discrepancies may yield view poses that violate visibility constraints such as maintaining proper distance from and orienting the camera toward the object's surface. An alternative for overcoming these limitations in OSVP approaches is to endow the robot with online replanning capabilities, so that the robot can compensate for the discrepancies between the estimated and actual geometries of the object while conducting the object's inspection. However, replanning only the view poses when reconstructing an object from discrete images ignores the rich data that the robot receives when moving between these view poses.

Therefore, we propose a new OSVP approach that aims to overcome the aforementioned limitations. Our method uses predicted geometric priors from RGB-D observations, which not only allows us to calculate a set of view poses that are globally optimal, but also to overcome the rescaling requirements reported in [9]. Our proposed method also conducts online video-based 3D reconstruction of the object, generating paths that connect the calculated view poses v_i and v_{i+1}, which meet the previously mentioned visibility constraints, i.e., a desired distance and orientation of the camera with respect to the object's surface. These paths are calculated with an optimization-based technique that can also locally compensate for discrepancies between the estimated and real geometries of the object, while also avoiding collisions and maintaining a smooth and continuous path. We extensively validate our approach in simulation against other state-of-the-art OSVP approaches. We demonstrate that our method leads to better and more efficient reconstruction. We also test our method with the real-world Franka manipulator [11] that is equipped with an Intel Realsense RGB-D camera.

II. Related Work

A. VP for 3D Reconstruction of Unknown Objects

As mentioned earlier, although NBVP methods enable inspection and reconstruction of objects that are initially unknown, they cannot guarantee that the final sequence of views is globally optimal. Alternatively, OSVP approaches can help overcome such a limitation by using prior knowledge. Two examples of approaches that embed such prior knowledge are: 1) a neural network (NN) model, which, given a first shot (image) of the object, aims to predict all the necessary view poses to reconstruct the object [10]; 2) once the first shot of the object is taken, an NN estimates the object's 3D model, which is then used by an optimization-based method to establish the minimal set of views that cover the object's surface for its reconstruction [9]. In either case, OSVP methods seek to estimate a sequence of view poses that achieve minimal movements to complete the object's reconstruction.

In [8], an OSVP method employs supervised learning to train an NN, which maps an initial point cloud input directly to a solution set of view poses. Their training method solves a set covering problem, thus ensuring the selected view poses provide complete coverage of the ground-truth 3D models. Given that this approach predicts the solution set of view poses by only using one observation as input, the network does not have access to additional sensory input that would reveal unobserved or occluded surfaces of the object. Consequently, the view poses are chosen based solely on the areas of the object that are visible in the initial observation, and therefore the prediction lacks the object's surface context. In this paper, we will be referring to the object's surface context as the initial local shape and visibility information around a surface region, which guides the selection of the most informative views.

Some works have added more context to the set covering optimization problem. In Pan et al. [9], a geometric prior is estimated from a single-view point cloud by using a mesh completion network. However, if the completion network fails to predict the object's shape accurately, the views might not satisfy the constraints mentioned earlier for the reconstruction of real objects. Another way to add more context is to execute one next-best view (NBV) step before predicting the solution set of view poses with an enhanced OSVP strategy, which includes an NN with multi-view activation capabilities [10]. Although this additional NBV step provides increased surface information, the posterior views calculated by the multi-view activated NN are still obtained from partial observations: the first view and the NBV. The robot then exclusively traverses the predicted views, and is unable to refine the views' positioning as new observations of the object's actual geometry become available. One alternative for overcoming this limitation in OSVP approaches would be replanning, so that the VP system can refine the initially estimated path as the object is incrementally reconstructed. Our proposed approach introduces such a strategy, which will be discussed in Section IV.

B. Planning the Camera Motion Between View Poses

VP approaches are commonly focused on generating the view poses to cover the object, but they do not calculate the robot's motion between generated view poses. The latter problem consists in finding feasible paths that connect start and goal configurations within the robot's configuration space (C-Space) [12].

Regarding NBVP, a common approach has been to use bidirectional sampling-based tree planners to calculate the motion between two view poses [13, 14]. It is worth noting that by planning in the robot's C-Space, the calculated motion is feasible for a given robot. However, the sampling-based methods that have been used neither attempt to provide optimal solutions, nor do they ensure any visibility constraints between view poses. Furthermore, such connecting paths are suboptimal due to the nature of those sampling-based planners [15].

With OSVP methods, on the other hand, a common approach to calculate the global shortest path that traverses the view poses has been to solve a Hamiltonian path problem [16]. By doing this, a straight-line path is generated in Cartesian space between successive view poses under visibility constraints [10]. The resulting trajectory is discretized, and then each discrete pose is passed to the robot's inverse kinematics (IK) solver to obtain a corresponding robot configuration. A limitation of these approaches is that they do not support replanning the path to compensate for the differences between the prior 3D model and the model that is being incrementally built. To do so, any changes in the connecting paths will need to account for different visibility constraints such as keeping a required distance to and desired orientation towards the object's surface. Furthermore, considering that the path is obtained from solving inverse kinematics of a series of discrete poses, the path is not guaranteed to be continuous or smooth. In contrast, our method not only computes a Hamiltonian path over the resulting set of view poses, but also dynamically compensates for visibility constraint errors while considering the continuity and smoothness of the path.

C. Motion Planning Under Constraints

An overview of constrained motion planning is provided in Kingston et al. [17], which categorizes techniques for integrating task-specific requirements into sampling-based frameworks while outlining key challenges. Within this context, Berenson et al. [18] proposed planning directly on constraint manifolds, embedding conditions such as end-effector poses into the search space to guarantee feasibility, though at the expense of costly projection operations. Despite these advances, existing approaches generally lack mechanisms to dynamically adapt initial paths to evolving constraints and remain underexplored in vision-guided tasks.

More recently, [19] extended constrained planning to perception-driven tasks, using vector field inequalities to maintain visibility of dynamic objects in cluttered scenes. Their formulation highlights how constraints can incorporate sensing requirements alongside kinematic feasibility; however, their work idealizes the object as a point in space, which might underestimate collision risks or ignore the actual shape of the object during the constraint optimization.

Our work not only plans motions under the constraints for vision-guided tasks, but also represents the object as a surface for a reconstruction application optimized by view planning.

III. Problem Definition

Previous work has defined OSVP as a conventional set covering problem [9]. To solve such a problem, it is necessary to have a geometric prior or 3D model of the object. Let M_p be a volumetric object representation, which is predicted from a single RGB-D image and corresponds to the 3D model of the object. From M_p, we can also obtain P_{surf}, which is the set of surface points p_i, such that $p_i \in P_{surf}$. v corresponds to a view pose within the view space $\mathcal{V} \subset SE(3)$ and P_v is the set of p_i that can be observed from v. The OSVP problem seeks to determine the minimal subset of view poses $\mathcal{V}^* \subset \mathcal{V}$, such that $P_{\mathcal{V}^*} = P_{surf}$, where $\mathcal{V}^* = \{v_1, v_2, \dots, v_k\}$, k is the number of view poses, and $P_{\mathcal{V}^*}$ contains the surface points observed from all the view poses contained in $\mathcal{V}^*$. Let $\Gamma : \{\gamma_1, \gamma_2, \dots, \gamma_{k-1}\}$ be the set of continuous Cartesian paths, where γ_i connects the view poses v_i and v_{i+1}, so that $\gamma_i : [0,1] \to SE(3)$, $\gamma_i(0) = v_i$ and $\gamma_i(1) = v_{i+1}$.

Unlike previous works in OSVP in which the 3D reconstruction of the object is done with discrete images that are captured from the view poses, in this work, we propose to use a video-based reconstruction, which also makes use of the frames that are captured while moving the camera in between view poses. Let M_c be the volumetric object representation, which is incrementally built with the continuous sequence of frames that are captured while traversing all γ_i.

Given that M_c might differ from M_p, we now consider the problem of adjusting γ_i such that the end-effector's camera both (i) maintains a desired distance from the real object's surface and (ii) aligns its camera axis with the surface normal, thus compensating for the differences between the object's predicted and actual geometries.

Let the pose of a robotic manipulator's eye-in-hand camera be represented by:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{p} \\ \mathbf{0} & 1 \end{bmatrix} \in SE(3), \tag{1}$$

where $\mathbf{p} \in \mathbb{R}^3$ is the end-effector position and $\mathbf{R} \in SO(3)$ is a rotation matrix that represents its orientation. The camera axis, $\mathbf{z}_c$, is defined as

$$\mathbf{z}_c = \mathbf{R}\,\mathbf{e}_z, \qquad \mathbf{e}_z = [0,0,1]^T. \tag{2}$$

Let $\phi(\mathbf{x})$ be a scalar function, which assigns to each point $\mathbf{x}$ in the Euclidean 3D space ($\mathbb{R}^3$) the shortest distance to the surface of an object, with the sign indicating whether the point lies inside (negative) or outside (positive) the object. For a Cartesian path point $\mathbf{p}$, we define:

a) Distance error: It measures the deviation of the camera distance from the desired offset $d^\star$ to the surface:

$$e_{\text{dist}} \;=\; \phi(\mathbf{p}) \;-\; d^\star, \tag{3}$$

b) Orientation error: Let $\mathbf{n}$ be the object's surface normal at camera position $\mathbf{p}$, and let $\nabla\phi$ be the gradient of ϕ, then:

$$\mathbf{n} = \frac{\nabla\phi(\mathbf{p})}{\|\nabla\phi(\mathbf{p})\|}. \tag{4}$$

Then, the orientation error is defined as:

$$e_{\text{orient}} = 1 - \mathbf{z}_c \cdot (-\mathbf{n}), \tag{5}$$

which measures the misalignment between the camera axis $\mathbf{z}_c$ and the surface normal $\mathbf{n}$.

The main objective is to compute a refined path, that jointly minimizes position and orientation errors:

$$\min_{\{\mathbf{q}_i\}} \sum_{i=1}^{N} \Big(\lambda_p\, e_{\text{dist}}^2 + \lambda_o\, e_{\text{orient}}^2\Big), \tag{6}$$

Here, λ_p and λ_o are scalar weights balancing the position and orientation objectives.

IV. OSVP via Video-based 3D Reconstruction and Optimization-based Replanning

In this section, we present our framework (see Figure 2), which aims to solve the OSVP problem that was discussed in Section III. We first describe the framework's execution pipeline, and then we discuss in detail each of its main functional blocks.

A. Execution Pipeline

Our framework starts by capturing an initial RGB-D image of the object, which is used by the *Geometric Prior Estimator* to predict the object's volumetric model M_p. Then, the *Global View Planner* computes the set of views $\mathcal{V}^*$ that cover M_p, and then calculates the paths $\gamma_i \in \Gamma$ that traverse such computed views. The *Optimization-based Planner* locally reshapes the paths γ_i according to the visibility constraints that are defined in Equation 6. Such constraints are continuously evaluated with respect to ϕ, which is incrementally built from the frames that are captured during the inspection.

B. Geometric Prior Estimator

To estimate M_p, our framework uses a continual learning system for real-time signed distance field reconstruction called iSDF [20], which is capable of predicting an object's shape from a single RGB-D observation. To achieve this, iSDF continuously trains a randomly initialized network to regress signed distances at sampled 3D coordinates from depth images in a camera stream, leveraging a batch-based self-supervision strategy for real-time adaptation. Finally, the generated signed distance field is processed by the marching cubes algorithm [21], which extracts a polygonal mesh that is voxelized and corresponds to M_p.

C. Global View Planner

To calculate the set of views $\mathcal{V}^*$, our framework first establishes the views that are needed to cover and fully inspect M_p. This is done by formulating the task as a set covering optimization problem and solving it using the Gurobi Optimizer [22]. Then, to determine the order in which those views must be traversed, our framework uses the Held-Karp algorithm [23] to solve the shortest Hamiltonian path problem [24], thus establishing the globally distance-optimal sequence to visit each view pose exactly once without returning to the first one.

To calculate the paths γ_i that connect two consecutive view poses v_i and v_{i+1}, we apply spherical linear interpolation (SLERP) for the translational component and axis projection for the rotational component, as follows:

Translation: Let $\hat{\mathbf{a}}, \hat{\mathbf{b}} \in \mathbb{R}^3$ be unit vectors from the M_p center $\mathbf{c}$ toward consecutive view poses, v_i and v_{i+1}, respectively. The interpolated direction at $t \in [0, 1]$ is given by:

$$\mathbf{d}(t) = \frac{\sin\big((1-t)\theta\big)}{\sin(\theta)}\,\hat{\mathbf{a}} + \frac{\sin\big(t\theta\big)}{\sin(\theta)}\,\hat{\mathbf{b}},$$

with $\theta = \arccos(\hat{\mathbf{a}} \cdot \hat{\mathbf{b}})$. The Cartesian translational vector is $\mathbf{p}(t) = r\,\mathbf{d}(t) + \mathbf{c}$, where r represents the distance to the object center.

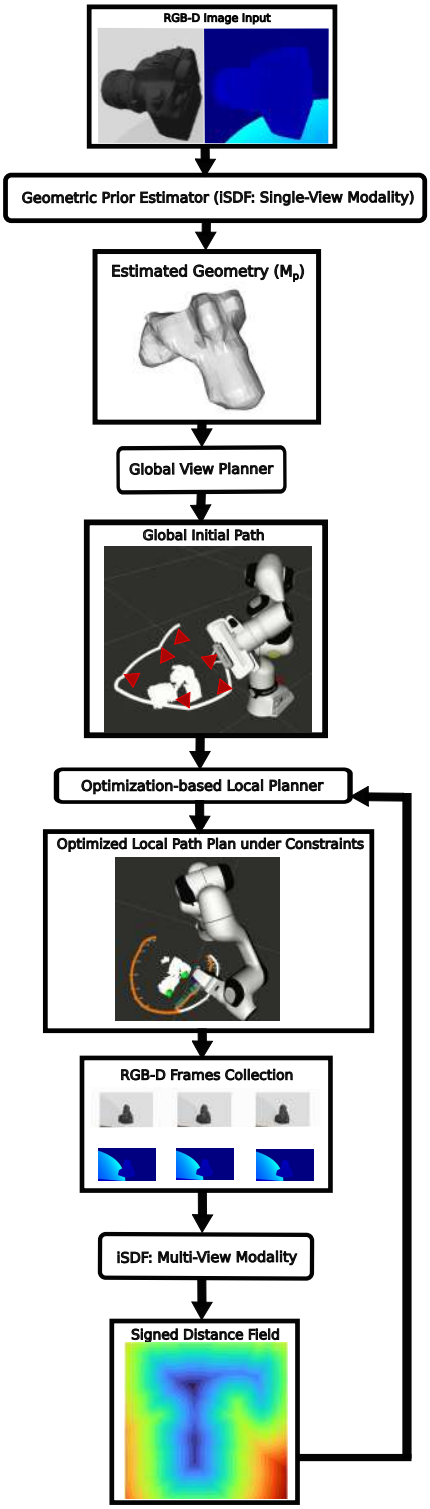


Fig. 2: Proposed OSVP framework to inspect and reconstruct an unknown object using an eye-in-hand fixed-base manipulator. Our framework first uses an initial RGB-D image to estimate a geometric prior of the object, which is used to calculate the inspection path. While traversing such a path, our framework incrementally builds a video-based 3D reconstruction of the object, which allows to identify discrepancies with respect to the estimated prior, thus allowing the robot to adjust the inspection path on the fly.

Rotation: The camera z-axis is aligned with the viewing direction $\hat{\mathbf{z}}_c(t) = (\mathbf{c}-\mathbf{p}(t))/\|\mathbf{c}-\mathbf{p}(t)\|$, which is a unit vector pointing from a trajectory point to the center. A smooth y-axis is obtained by linear interpolation of the input frames and projection orthogonal to $\hat{\mathbf{z}}_c(t)$, after which the x-axis is computed by cross product. The resulting orthonormal basis defines $\mathbf{R}(t)$, from which the interpolated quaternion is extracted.

Using these interpolation methods results in a consistent and dense sequence of poses. After this, the RelaxedIK approach in [25] is used to get the joint space initial path, $\mathbf{q}(t)$.

D. Optimization-based Path Planner under Visibility Constraints

We solve our optimization problem in the joint space to have more control over the path feasibility with respect to the manipulator's kinematic constraints, continuity and smoothness. Taking into consideration the initial path, our main goal is to adapt it according to the constraints mentioned in Section III, which are dictated by the particularities of the object's surface. It is worth mentioning that the function $\phi(\cdot)$ mentioned in Section III is represented by the multi-view modality of iSDF [20]. $\phi(\cdot)$ is built incrementally with online frame acquisition and is used in the optimization approach to satisfy the visibility constraints and avoid collisions.

The optimization technique is based on a set of energy functionals, which encode both smoothness of the path and task-specific constraints inspired by Kass et al. [26]. Each functional contributes a gradient that guides the trajectory refinement. Formally, the total energy is written as

$$E(\mathbf{q}(t)) \;=\; \sum_{k\in\mathcal{K}} \lambda_k \, E_k(\mathbf{q}(t)), \tag{7}$$

where λ_k are weighting terms and E_k are the energy functionals that encode the objective terms to be optimized. In the following subsections each objective is explained.

a) Tension and smoothness terms: To promote trajectories that are both continuous and smooth, we combine tension and curvature into a single functional that penalizes high velocities and accelerations:

$$E_{\text{ten+curv}}(\mathbf{q}(t)) = \int_0^T \left(\lambda_{\text{ten}} \, \|\dot{\mathbf{q}}(t)\|^2 + \lambda_{\text{curv}} \, \|\ddot{\mathbf{q}}(t)\|^2\right) dt, \tag{8}$$

where λ_{ten} and λ_{curv} balance the contribution of path length minimization and curvature regularization.

b) Distance-to-surface term: For the distance error, defined in Equation 3, the corresponding energy is established as:

$$E_{\text{dist}}(\mathbf{q}(t)) = \int_0^T \left(e_{\text{dist}}(t)\right)^2 dt. \tag{9}$$

It is worth mentioning that since the optimization is formulated in the joint space, we can also present Equation 3 as $e_{\text{dist}} = \phi(\mathbf{p}(\mathbf{q}(t))) - d^\star$, so that the gradient of this objective can be differentiated with respect to the joint variables.

c) Orientation term: The orientation error, as defined in Equation 5, has the following energy functional:

$$E_{\text{orient}}(\mathbf{q}(t)) = \int_0^T \left(e_{\text{orient}}(t)\right)^2 dt. \tag{10}$$

Similar to Equation 3, the orientation error can be presented as a function in terms of the manipulator joints: $e_{\text{orient}} = 1 - \mathbf{z}_c(\mathbf{q}(t)) \cdot (-\mathbf{n}(\mathbf{q}(t)))$.

d) Collision avoidance terms: Similar to Ratliff et al. [27], we incorporate an SDF-based collision avoidance cost to prevent the manipulator from intersecting the environment. For each configuration, a set of points is sampled along the robot links, and their distances to the nearest object's surface are queried from the SDF. Each point contributes an exponential penalty of the form $\exp(-\phi(\mathbf{p})/\tau)$, where $\phi(\mathbf{p})$ is the signed distance at point $\mathbf{p}$ and τ controls the decay rate. This formulation ensures that points far from obstacles contribute negligibly to the cost, while points near or inside obstacles generate large gradients pushing them away from the surface.

e) Joint limit avoidance terms: Following [25], we incorporate a smooth swamp-shaped loss function that penalizes configurations approaching the joint boundaries. This function remains nearly flat when the joint is well inside its valid range, ensuring no unnecessary restriction, but increases rapidly as the joint approaches its lower or upper bound, and grows steeply beyond them. In this way, the optimizer naturally avoids solutions that lie near or outside of the valid joint range, while maintaining the differentiability required for efficient optimization.

1) Trajectory Optimizer: The optimizer seeks to minimize the total functional $E(\mathbf{q}(t))$. Using a gradient descent scheme in the function space, the trajectory is updated as

$$\frac{\partial \mathbf{q}(t)}{\partial j} = -\eta \, \frac{\delta E(\mathbf{q}(t))}{\delta \mathbf{q}(t)}, \tag{11}$$

where j denotes the iteration index and η the learning rate, which is 0.001.

When the distance constraint is already satisfied, i.e. $|e_{\text{dist}}(t)| < \epsilon_{\text{pos}}$, orientation is further optimized in the nullspace of the Jacobian to avoid perturbing the end-effector position:

$$\Delta\mathbf{q}(t)^{\text{orient}} = N(\mathbf{q}(t)) \, \frac{\delta E_{\text{orient}}(\mathbf{q}(t))}{\delta \mathbf{q}(t)}, \tag{12}$$

where $N(\mathbf{q}(t))$ is the nullspace projector. This ensures that orientation refinement does not compromise positional accuracy.

The final trajectory $\mathbf{q}^*(t)$ is therefore smooth, consistent with object surface constraints, and well aligned for reconstruction.

2) Parametric normalization ("groove"): To combine heterogeneous objectives into a single cost, we apply a parametric normalization function inspired by the *RelaxedIK* formulation [25]. The normalization maps a raw scalar error χ into a bounded value that (i) places a narrow, strong attraction (a "groove") around the desired goal, (ii) exhibits a gradual

fall-off away from the groove so gradients remain informative at moderate offsets, and (iii) provides a controlled, consistent gradient direction toward the goal.

Concretely, let χ denote a scalar error (e.g., $\chi_{\text{dist}} = \phi(p(q)) - d^*$ for distance or $\chi_{\text{orient}} = e_{\text{orient}}$ for orientation). We define the parametric normalization function $G(\chi)$ as the sum of a Gaussian-shaped groove and a polynomial tail:

$$G(\chi; s, c, r, p) \;=\; A\,\exp\Big(-\frac{(\chi - s)^2}{2c^2}\Big) \;+\; r\,(\chi - s)^p, \quad (13)$$

where s is the groove center (typically the target value, e.g., $s = 0$), $c > 0$ controls the Gaussian width (narrow c gives a tight groove), r and $p \geq 1$ shape the polynomial tail (gentle global attraction), $A \in \{+1, -1\}$ can flip the sign if needed (e.g., to produce negative/positive groove).

a) Gradient (chain rule) for implementation: For numerical updates, $\delta E/\delta \mathbf{q}(t)$ is computed using the chain rule:

$$\frac{\partial}{\partial \mathbf{q}(t)} G\big(\chi(\mathbf{q}(t)\big) = G'\big(\chi(\mathbf{q}(t))\big)\;\frac{\partial \chi(\mathbf{q}(t))}{\partial \mathbf{q}(t)}.$$

Two useful cases in our implementation are explained below. For the distance term, the constraint is defined as $\chi_{\text{dist}}(\mathbf{q}(t)) = \phi(p(\mathbf{q}(t))) - d^*$, where $\phi(p)$ denotes the signed distance field (SDF) and $p(\mathbf{q}(t))$ is the Cartesian end-effector position. The gradient of the SDF, $\nabla_p \phi(p)$, combined with the translational Jacobian $J_p(\mathbf{q}(t)) = \partial \mathbf{p}(t)/\partial \mathbf{q}(t)$, yields:

$$\nabla_q \chi_{\text{dist}}(\mathbf{q}(t)) = J_p(\mathbf{q}(t))^\top \nabla_p \phi(\mathbf{p}(\mathbf{q}(t)))$$

Accordingly, the joint-space gradient of the normalized distance term becomes:

$$\nabla_q \big(G(\chi_{\text{dist}})\big) = G'(\chi_{\text{dist}})\; J_p(\mathbf{q}(t))^\top \nabla_p \phi(\mathbf{p}(\mathbf{q}(t)))$$

For the orientation term, the constraint is expressed as $\chi_{\text{orient}}(\mathbf{q}(t)) = 1 - z_c(\mathbf{q}(t)) \cdot \big(-\mathbf{n}(\mathbf{p}(\mathbf{q}(t)))\big)$. Letting $J_z(\mathbf{q}(t)) = \partial z_c/\partial \mathbf{q}(t)$ denote the Jacobian of the camera axis, the gradient with respect to the joints is

$$\nabla_q \chi_{\text{orient}}(\mathbf{q}(t)) = J_z(\mathbf{q}(t))^\top n,$$

and the corresponding joint-space contribution is

$$\nabla_q \big(G(\chi_{\text{orient}})\big) = G'(\chi_{\text{orient}})\; J_z(\mathbf{q}(t))^\top n.$$

3) Fixed-window optimization strategy: To balance efficiency and stability, we employ a fixed-window optimization scheme rather than updating the entire trajectory at once. At each optimization step, only a local segment of the trajectory $\{q_i, \ldots, q_{i+w}\}$ of length w is considered for gradient-based updates. In our experiments, the size of the sliding window is 20. This sliding window ensures that corrections remain smooth and localized, preventing oscillations or large deformations that might arise from global updates. The window is shifted forward along the trajectory as the optimization progresses, allowing the entire path to be refined iteratively while maintaining continuity between neighboring segments. This strategy also reduces computational complexity, as Jacobian evaluations and gradient computations are restricted to a subset of waypoints, which is particularly beneficial for real-time or large-scale reconstruction scenarios.

Fig. 3: 3D models of the objects used in our simulation-based experiments.

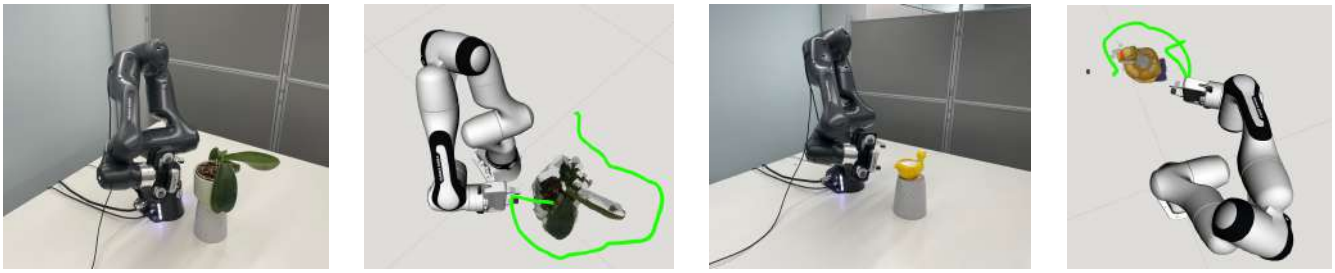
V. Experiments and Evaluation

To evaluate our framework, we first benchmarked it in simulation with other state-of-the-art approaches, and then conducted real-world experiments to validate its deployment. For the benchmark, we used ten different objects that were inspected and reconstructed, each starting from random initial views, using the fixed-base manipulator Franka Emika equipped with a depth camera. The experiments were conducted in the Gazebo Ignition simulation environment [28] and implemented using the Robot Operating System 2 (ROS2) middleware on Ubuntu 22.04. For each object, 66 experiments were carried out on a PC equipped with an NVIDIA GeForce GTX 1660 GPU and an Intel Core i7-10750H CPU.

Our benchmark evaluates the capability of our framework to generate paths that connect successive view poses while being as short as possible, smooth and collision-free, and while satisfying our predefined visibility constraints. For our experiments, the weights $\{\lambda_{\text{ten}}, \lambda_{\text{curv}}, \lambda_{\text{dist}}, \lambda_{\text{orient}}\}$ were set to $\{3.0, 1.5, 7.5, 8.5\}$, respectively. Additionally, to demonstrate that our framework is able to locally adapt to differences between generated and actual geometries, our choice of objects to be reconstructed features both common and uncommon shapes, as well as irregular geometries as shown in Fig. 3. We compare our approach with the baselines DM-OSVP [9] and MA-SCVP [10]. The reader is referred to the following for further experimental details: `https://www.youtube.com/watch?v=Si_atZL73xI&t=1s`.

A. Evaluation Metrics

The following metrics are used to assess our approach. The path length of γ_i is defined as $L = \sum_{i=2}^{N} \|\mathbf{p}_i - \mathbf{p}_{i-1}\|$, where $\mathbf{p}_i \in \mathbb{R}^3$ denotes the translational part of every pose along γ_i. It measures the efficiency of the path, as lower values indicate the robot travels a shorter distance while executing a reconstruction. The mean distance error (MDE) is given by $\text{MDE} = \frac{1}{N}\sum_{i=1}^{N} e_i^{\text{pos}}$ and measures the average position error over all valid poses, while the mean orientation error (MOE) is defined as $\text{MOE} = \frac{1}{N}\sum_{i=1}^{N} e_i^{\text{orient}}$ and captures the average orientation error. Both the MDE and the MOE measure the satisfaction of the visibility constraints presented in Equations 3 and 5, respectively, along the path. The point density per surface area (PDSA) metric measures the number of camera rays that intersect the object surface $N_{\text{rays}\cap\text{surface}}$ per unit surface area A_{surface}, reflecting how much the camera focuses on acquiring object surface information instead of irrelevant background data.



(a) (b) (c) (d)

Fig. 4: Real-world experiments using our proposed OSVP approach to inspect different objects using a Franka Research 3 manipulator equipped with an eye-in-hand Intel Realsense D435i camera. (a) An orchid in a pot. (b) The 3D reconstruction of the orchid and the trajectory followed by the camera is shown in green. (c) A ceramic duck. (d) The reconstruction of the ceramic duck and the trajectory followed by the camera is shown in green.

Method	$\overline{L}$ (SD) [m]	$\overline{MDE}$ (SD) [m]	$\overline{MOE}$ (SD)	$\overline{PDSA}$ (SD) $[m^{-2}]$	$\overline{CD}$ (SD) $[m^{-2}]$
Ours	1.62 (1.52)	0.0065 (0.014)	0.016 (0.02)	24371497.69 (8161892.56)	0.0083 (0.0058)
DM-OSVP	1.82 (1.72)	0.061 (0.015)	0.085 (0.05)	15867570.22 (6176503.44)	0.0090 (0.0065)
MA-SCVP	1.78 (1.66)	0.113 (0.071)	0.314 (0.27)	5391066.31 (2835223.216)	0.0085 (0.0031)

TABLE I: Comparison of our method against baseline OSVP approaches in simulation using the Franka Emika manipulator with a depth camera in Gazebo Ignition. Metrics include the mean ($\overline{\cdot}$) and standard deviation (SD) of the path length (L), distance error (MDE), orientation error (MOE), per-surface-ray density ($PDSA$), and Chamfer Distance (CD). Results are averaged over 66 reconstruction experiments per object on a set of ten objects. Our method consistently achieves shorter, better-constrained paths, while providing higher surface coverage and improved reconstruction fidelity.

The PDSA is defined as PDSA $= \frac{N_{\text{rays}\cap\text{surface}}}{A_{\text{surface}}}$. Finally, the Chamfer distance (CD) quantifies the similarity between two point clouds $\mathcal{P}$ and $\mathcal{Q}$. It is computed by averaging the distance from each point in $\mathcal{P}$ to its nearest neighbor in $\mathcal{Q}$, and vice versa, with lower values indicating closer alignment between the sets. CD is expressed as: $\text{CD}(\mathcal{P},\mathcal{Q}) = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}\min_{q\in\mathcal{Q}}\|p-q\|^2 + \frac{1}{|\mathcal{Q}|}\sum_{q\in\mathcal{Q}}\min_{p\in\mathcal{P}}\|q-p\|^2$.

B. Simulation-based Benchmark

Table I presents the performance of our proposed approach compared to other OSVP methods, showing that our method outperforms the baselines across every evaluated metric. As our optimization-based approach keeps the camera at a predefined distance to the surface, the length of the path is reduced compared to approaches that keep a fixed distance with respect to an estimated object center, which basically restricts the camera to lie in the periphery of a hemisphere. Our approach's main goals are to compensate for any differences between the view poses' distances to the surface and the desired distance to the surface, while keeping the camera axis directed normal to that surface, which explains the improvement in the $\overline{MDE}(m)$ and $\overline{MOE}$ metrics. With regard to the PDSA, as our approach compels the camera to point to the object's surface and keep a distance, the camera observes less irrelevant background information, and concentrates more on the object's details, which makes our method stand out from other approaches, in terms of this metric. Finally, $\overline{CD}$ reflects the capability of our approach to minimize regions without surface observations in the reconstructed model, thus making it more similar to the ground truth. Whilst the approach presented in this paper demonstrates a meaningful improvement on $\overline{MDE}(m)$, $\overline{MOE}$ and $\overline{PDSA}$ metrics with respect to the state of the art, the improvements for CD are marginal, as we do not present a method to cover large unseen areas. The standard deviations are also reported; their values reflect the large variability with respect to the mean of every metric, as our experiments vary not only in the initial view and object shape, but also because of the stochastic nature of the geometric prior estimator, which affects the diversity of our results. The average computation time per optimization step is $0.62\,\text{s}$ for a 20-point sliding window ($\approx 31\,\text{ms}$ per point).

Compared to DM-OSVP and MA-SCVP, our method excels especially in terms of path length, a result of keeping a fixed distance with respect to the object's surface in combination with our tension objective. Our method also achieves a substantial reduction in the MOE and MDE metrics compared to the baselines, as it adaptively compensates for these errors online after solving the OSVP problem. As our method optimizes under visibility constraints, it focuses more on the object surface than the baselines as indicated by the PDSA metric. Our method achieves a reconstruction quality, measured by the CD metric, that is comparable to the MA-SCVP approach and the DM-OSVP method.

C. Real-world Experiments

Our real-world experiments were conducted using a Franka Research 3 manipulator, which was equipped with an eye-in-hand Intel Realsense D435i camera. Figures 4a and 4b show an orchid in a pot being inspected by the manipulator, while Figures 4c and 4d illustrate the trajectories generated by our proposed framework (green curves). For the experiment with the orchid, the optimized path length was $1.17m$, which corresponds to a 19.86% reduction with respect to the initial path, which had a length of $1.46m$. The MDE was $1.17\times10^{-4}m$ and the MOE was 2.8×10^{-3}. Figures 4c and 4d show a similar test with a ceramic duck, for which the optimized path length was $0.88m$, which corresponds to a 14.81% reduction with respect to the initial path, which had

a length of $1.03m$. The MDE was $1.02 \times 10^{-4}m$ and the MOE was 5.2×10^{-3}. Both tests demonstrate the capability of our approach to minimize the errors along the trajectory under visibility constraints in a real environment and for both regular (the duck) and irregular (the orchid) objects.

VI. Conclusions

This paper has presented a novel OSVP approach that integrates a correctly scaled 3D geometric prior, generated from RGB-D data, into the set covering optimization problem, thereby eliminating the need for post-scaling of the resulting 3D reconstruction. In order to account for differences between the predicted geometric prior and the actual object, we introduced an optimization-based local planner to compensate for the errors in distance to and orientation towards the object's surface. The results show that our approach outperforms other state-of-the-art methods with respect to different metrics that account for both the path traversed for the inspection and the quality of the reconstruction. Another key effect of our approach is that the point density is increased, which means that our approach leads the camera to be more focused on the object rather than the surrounding environment. For future work, we plan to include an additional term in our local planner that permits the path to deviate in order to cover zones of the object that may have been left unexplored.

References

[1] C. Mineo, D. Cerniglia, V. Ricotta, and B. Reitinger, "Autonomous 3d geometry reconstruction through robot-manipulated optical sensors," *The International Journal of Advanced Manufacturing Technology*, vol. 116, pp. 1–17, 09 2021.

[2] M. Bitzidou, D. Chrysostomou, and A. Gasteratos, "Multi-camera 3D object reconstruction for industrial automation," in *Advances in Production Management Systems. Competitive Manufacturing for Innovative Products and Services*, C. Emmanouilidis, M. Taisch, and D. Kiritsis, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 526–533.

[3] G. Marchello, R. Giovanelli, E. Fontana, F. Cannella, and A. Traviglia, "Cultural heritage digital preservation through ai-driven robotics," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLVIII-M-2-2023, pp. 995–1000, 2023.

[4] A. Malik, H. Lhachemi, J. Ploennigs, A. Ba, and R. Shorten, "An application of 3D model reconstruction and augmented reality for real-time monitoring of additive manufacturing," *Procedia CIRP*, vol. 81, pp. 346–351, 2019, 52nd CIRP Conference on Manufacturing Systems (CMS), Ljubljana, Slovenia, June 12-14, 2019.

[5] R. Zeng, Y. Wen, W. Zhao, and Y.-J. Liu, "View planning in robot active vision: A survey of systems, algorithms, and applications," *Computational Visual Media*, vol. 6, no. 3, pp. 225–245, 2020.

[6] C. Connolly, "The determination of next best views," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 432–435.

[7] L. Jin, X. Chen, J. Rückin, and M. Popovic, "Neu-nbv: Next best view planning using uncertainty estimation in image-based neural rendering," 03 2023.

[8] S. Pan, H. Hu, and H. Wei, "SCVP: Learning one-shot view planning via set covering for unknown object reconstruction," *IEEE Robotics and Automation Letters*, vol. 7, pp. 1463–1470, 4 2022.

[9] S. Pan, L. Jin, X. Huang, C. Stachniss, M. Popović, and M. Bennewitz, "Exploiting priors from 3d diffusion models for rgb-based one-shot view planning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 13 341–13 348.

[10] S. Pan, H. Hu, H. Wei, N. Dengler, T. Zaenker, M. Dawood, and M. Bennewitz, "Integrating one-shot view planning with a single next-best view via long-tail multiview sampling," *IEEE Transactions on Robotics*, vol. 41, pp. 394–414, 2025.

[11] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jähne, L. Hausperger, and S. Haddadin, "The franka emika robot: A reference platform for robotics research and education," *IEEE Robotics & Automation Magazine*, vol. 29, no. 2, pp. 46–64, 2022.

[12] Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, vol. C-32, no. 2, pp. 108–120, 1983.

[13] J. I. Vasquez-Gomez, L. E. Sucar, R. Murrieta-Cid, and E. Lopez-Damian, "Volumetric next-best-view planning for 3d object reconstruction with positioning error," *International Journal of Advanced Robotic Systems*, vol. 11, 10 2014.

[14] C. Wu, R. Zeng, J. Pan, C. C. L. Wang, and Y.-J. Liu, "Plant phenotyping by deep-learning-based planner for multi-robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3113–3120, 2019.

[15] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, 2005.

[16] R. Gould, "Advances on the hamiltonian problem - a survey," *Graphs and Combinatorics*, vol. 19, pp. 7–52, 03 2003.

[17] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. Volume 1, 2018, pp. 159–185, 2018. [Online]. Available: https://www.annualreviews.org/content/journals/10.1146/annurev-control-060117-105226

[18] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 625–632.

[19] F. Dursun, B. V. Adorno, S. Watson, and W. Pan, "Maintaining visibility of dynamic objects in cluttered environments using mobile manipulators and vector field inequalities," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 6371–6378.

[20] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam, "isdf: Real-time neural signed distance fields for robot perception," in *Robotics: Science and Systems*, 2022.

[21] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87. New York, NY, USA: Association for Computing Machinery, 1987, p. 163–169. [Online]. Available: https://doi.org/10.1145/37401.37422

[22] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com

[23] M. Held and R. M. Karp, "A dynamic programming approach to sequencing problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 10, no. 1, pp. 196–210, 1962.

[24] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. Springer, 1972, pp. 85–103.

[25] D. Rakita, B. Mutlu, and M. Gleicher, "RelaxedIK: Real-time Synthesis of Accurate and Feasible Robot Arm Motion," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[26] M. Kass, A. P. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, pp. 321–331, 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:12849354

[27] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 489–494, 2009.

[28] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2149–2154 vol.3.