Combining Diffusion Models and Force-Aware Optimization for Robust In-Hand Tool-Use

Jinjia Guo
jinjaguo@umich.edu
Department of Robotics
University of Michigan
Supervised by Prof.Dmitry Berenson

***Abstract*—We propose an end-to-end perception-planning-control framework to solve the high-dimensional mixed contact dynamics and real-time replanning problems faced by multi-fingered dexterous hands in hand-operated tasks such as turning screwdrivers. The system first completes the 6-DoF posture estimation of the screwdriver through feature-RANSAC combined with two-stage ICP based on the point cloud acquired by the depth camera; then uses the diffusion model to quickly generate candidate trajectories in the workspace, and designs two lightweight contact correction strategies: 1. SDF-constrained contact projection, which projects the fingertips back to the object surface with signed distance field iteration after each micro-rotation step; 2. LP-DS force closure relaxation correction, which solves the linear programming to maximize the grasping margin online, and maps the optimal contact force to the fingertip attraction position through the dynamic system. Both avoid strong constraints on the contact mode from time to time, thereby significantly reducing the computational load.**

The random initial posture experiment in Isaac Sim shows that compared with the pure diffusion baseline, the SDF projection scheme improves the final rotation angle by 18.6% on average, and the single-step calculation delay is 0.17 s; the LP-DS scheme has the best trajectory smoothness and contact stability while maintaining a control frequency of 0.06 s. The results verify the effectiveness and real-time potential of "diffusion initialization + minimization constraint correction" in visual-driven hand operation, laying the foundation for subsequent fusion environment contact and real hardware deployment.

I. Introduction

Dexterous in-hand manipulation—such as rotating a screwdriver within a robotic hand—remains a long-standing challenge in robotics due to the hybrid and high-dimensional nature of contact dynamics between the hand and object. Successful manipulation requires the robot to reposition the object while maintaining persistent and stable contact between fingertips and the object surface. Traditional approaches often model this task as a constrained trajectory optimization problem [1], which, while theoretically grounded, tend to be computationally intensive and thus unsuitable for real-time, high-frequency replanning.

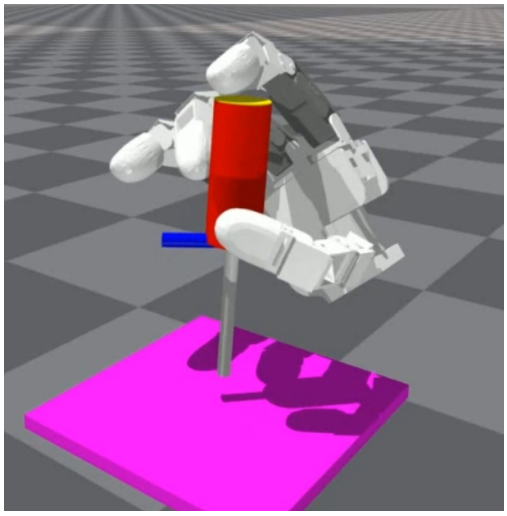

Fig. 1: Our in-hand manipulation task: A multi-fingered robotic hand rotates a screwdriver while maintaining stable contact. This operation requires precise pose estimation, robust contact control, and adaptive planning.

In this work, we aim to enable reactive in-hand manipulation by designing a planning pipeline that operates directly on 3D point cloud observations. Specifically, we estimate the pose of the screwdriver using iterative closest point (ICP) alignment between the observed point cloud and a pre-scanned object model. This pose is then used as input to the downstream modules of trajectory generation and contact correction. By leveraging only raw perceptual input, this framework facilitates future deployment on physical hardware.

We focus on an in-hand manipulation task where a multi-fingered robotic hand rotates a screwdriver by maintaining continuous contact throughout the motion. Figure 1 illustrates our setup, where the robotic hand performs a controlled rotation of a screwdriver, highlighting the challenges in contact maintenance and trajectory planning.

Recent studies have shown promise in combining learning-based generative models with model-based

planning techniques. Diffusion models [2], in particular, offer a solution for generating diverse candidate trajectories. However, these methods often suffer from contact instability and high variance in outcomes, as the success of manipulation is tightly coupled to the quality of the sampled trajectories. While some prior work strictly enforces constraints through trajectory optimization [3] [4], such constraints lead to heavy computational burdens and are impractical for online applications.

Our key insight is that strict constraint enforcement at every time step may not be necessary for accomplishing robust in-hand screwdriver turning. Instead, we explore two lightweight but effective strategies to improve the robustness of diffusion-based planning:

- **Distance-based Diffusion Sampling + Contact Projection:** After each micro-rotation, we project the fingertips back onto the object surface by enforcing a signed distance field (SDF) constraint—i.e., SDF ≈ 0. This projection maintains contact consistency without explicitly modeling contact dynamics, and significantly accelerates trajectory refinement.
- **Force-aware Diffusion Sampling + Linear Programming (LP) Correction:** Diffusion-generated trajectories are corrected using a fast LP-based formulation that maximizes a grasp quality margin subject to relaxed constraints. Unlike CSVTO, we do not strictly require force closure, but instead enforce approximate stability, enabling efficient online correction under relaxed conditions.

These two strategies are evaluated independently to study their trade-offs. While contact projection provides computational speed-up, it may result in discontinuous motions. In contrast, LP correction yields more stable and smoother behaviors while still supporting high-frequency trajectory replanning. Together, they highlight the potential of diffusion-initialized, minimally constrained contact correction as a practical and generalizable method for dexterous in-hand tasks.

The remainder of this paper is organized as follows. Section II introduces our approach, including ICP-based pose alignment, contact projection, and the diffusion + LP planning pipeline. Section III details the experimental setup and evaluation metrics. Section IV presents and compares the performance of different strategies. Finally, Section V summarizes our findings and discusses limitations and future work directions.

II. Approach

To enable efficient and robust in-hand manipulation based on visual input, our method integrates 3D perception, generative trajectory sampling, and contact correction into a unified planning pipeline. The entire system operates on 3D point cloud observations and produces manipulation trajectories suitable for real-time execution during tasks such as screwdriver turning.

A. The ICP-based pose alignment from 3D point clouds:

Given an observed partial point cloud of the object, we first estimate the full object pose using an iterative closest point (ICP) alignment algorithm with a known SDF model. This estimated pose serves as the basis for trajectory generation.

1) Reconstructing 3D point clouds from depth images: We mount a depth camera in the Isaac sim [5] and render RGB-D images from a fixed view. Let $D(u, v)$ denote the depth value at pixel coordinates (u, v), with image width W and height H. Using the camera's intrinsic parameters—focal lengths f_u, f_v (The camera's intrinsic parameters can be obtained directly from the Isaac API) and principal point (c_u, c_v)—each pixel with a valid depth value can be projected into 3D space in the camera coordinate frame:

$$Z = D(u, v), \tag{1}$$

$$X = \frac{-(u - c_u)}{W} \cdot Z \cdot \frac{2}{f_u}, \tag{2}$$

$$Y = \frac{(v - c_v)}{H} \cdot Z \cdot \frac{2}{f_v} \tag{3}$$

The raw depth image is used to reconstruct the full point cloud of the scene, as shown in Fig. 2.

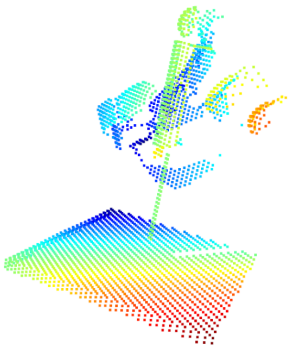

Fig. 2: Raw 3D point cloud reconstructed from depth image before segmentation.

The resulting homogeneous coordinates $(X, Y, Z, 1)^\top$ are then transformed into world coordinates using the inverse of the camera's extrinsic matrix T_{cam}^{-1}:

$$\mathbf{p}_{\text{world}} = T_{\text{cam}}^{-1} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

To isolate the screwdriver from the full scene point cloud, we use the segmentation output from Isaac Sim. Let $S(u, v)$ denote the segmentation class at each pixel. We apply a binary mask:

$$M(u,v) = \begin{cases} 1 & \text{if } S(u,v) = \text{ID}_{\text{scr}} \text{ and } Z \in (-\bar{d}, 0) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

To isolate the screwdriver, a segmentation mask is applied based on asset handle IDs provided by the Isaac sim, as illustrated in Fig. 3.

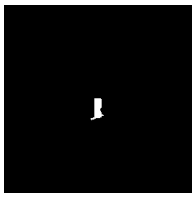

Fig. 3: Segmentation mask used to isolate the screwdriver from the full scene.

Only points with $M(u, v) = 1$ are retained for downstream processing. The resulting 3D point cloud of the screwdriver is then used in an ICP registration step to align with a pre-scanned model, thereby recovering the object pose $\hat{T}_{\text{screwdriver}}$ with respect to the Isaac sim world frame.

After applying the mask, we extract a clean point cloud corresponding to the screwdriver, shown in Fig. 4, which is used for ICP registration.

Fig. 4: Filtered 3D point cloud of the screwdriver extracted using the segmentation mask.

2) Pose Registration via Feature-Based Matching and ICP: To estimate the 6-DoF pose of the screwdriver in the world frame, we perform point cloud registration between the observed point cloud $\mathcal{P}_{\text{obs}}$ and a pre-sampled canonical point cloud $\mathcal{P}_{\text{model}}$ obtained from the object mesh using signed distance field (SDF) sampling. The registration process follows a two-stage procedure: feature-based global alignment followed by local refinement using Iterative Closest Point (ICP).

a) Prior pose estimation from simulation.: To obtain a rough initial estimate of the object pose, we utilize the ground-truth transforms available in Isaac Sim. Specifically, we retrieve the 6-DoF pose of the screwdriver in the camera frame, denoted as $T_{\text{cam}}^{\text{screw}}$, and the pose of the camera in the world frame, $T_{\text{world}}^{\text{cam}}$. The global pose of the screwdriver in the world frame can then be computed by matrix multiplication:

$$T_{\text{world}}^{\text{screw}} = T_{\text{world}}^{\text{cam}} \cdot T_{\text{cam}}^{\text{screw}} \tag{6}$$

This transform $T_{\text{world}}^{\text{screw}}$ is used as our prior estimate T_{prior} in the registration process. Since the poses are extracted from the simulator directly, they provide a reasonable initial guess that significantly narrows the search space for downstream optimization.

b) Global alignment with FPFH and RANSAC.: We begin by downsampling both $\mathcal{P}_{\text{obs}}$ and $\mathcal{P}_{\text{model}}$ using voxel grid filtering, followed by normal estimation for each point using a fixed-radius neighborhood. Then, we compute Fast Point Feature Histograms (FPFH) [6] for all points.

Given the FPFH descriptors ϕ_{obs} and ϕ_{model}, we apply RANSAC-based correspondence matching to compute a coarse rigid transformation $T_{\text{RANSAC}} \in SE(3)$ that aligns $\mathcal{P}_{\text{obs}}$ to $\mathcal{P}_{\text{model}}$:

$$T_{\text{RANSAC}} = \arg \min_{T \in SE(3)} \sum_{(i,j) \in \mathcal{C}} \|T\mathbf{p}_i^{\text{obs}} - \mathbf{p}_j^{\text{model}}\|^2 \tag{7}$$

where $\mathcal{C}$ is the set of tentative correspondences established via feature similarity, and distance and edge-length constraints are used to reject outliers.

In practice, we combine this transformation with a prior pose estimate T_{prior}, using spherical linear interpolation (Slerp) on quaternions and linear blending on translations:

$$\mathbf{q}_{\text{blend}} = \text{Slerp}(\mathbf{q}_{\text{prior}}, \mathbf{q}_{\text{RANSAC}}, \alpha) \tag{8}$$

$$\mathbf{t}_{\text{blend}} = (1 - \alpha)\mathbf{t}_{\text{prior}} + \alpha\mathbf{t}_{\text{RANSAC}} \tag{9}$$

The final initial guess T_{init} is reconstructed from $\mathbf{q}_{\text{blend}}$ and $\mathbf{t}_{\text{blend}}$, where $\alpha \in [0, 1]$ is a trust factor (typically $\alpha = 0.01$).

c) Local refinement with ICP: Starting from T_{init}, we perform two-stage ICP registration. The goal is to refine the transformation by minimizing pointwise distance between the aligned clouds.

- **Coarse ICP:** Point-to-point ICP with a large correspondence threshold δ_{coarse} is used for initial alignment.

- **Fine ICP:** Point-to-plane ICP with a smaller threshold δ_{fine} refines the pose by minimizing the projection of alignment error onto target normals.

The fine-level optimization solves:

$$T^* = \arg\min_{T \in SE(3)} \sum_i \left(\mathbf{n}_i^\top (T\mathbf{p}_i^{\text{obs}} - \mathbf{p}_i^{\text{model}})\right)^2 \quad (10)$$

where $\mathbf{n}_i$ is the estimated normal at point $\mathbf{p}_i^{\text{model}}$. The final transformation T^* serves as the estimated screwdriver pose in the world frame.

d) Implementation details.: We use a voxel size of $0.005\,\text{m}$, with $k = 30$ nearest neighbors for normal estimation. FPFH feature extraction uses a support radius of $5\times$ the voxel size. The RANSAC step is configured with 50,000 iterations and a success probability of 0.999. ICP refinement runs for 200 and 500 iterations in coarse and fine phases, respectively.

The result of the point cloud registration is shown in Fig. 5. In this visualization, the yellow point cloud represents the observed screwdriver point cloud corrupted with Gaussian noise to simulate sensor uncertainty in the real world. The blue point cloud corresponds to the surface point cloud sampled from the SDF, serving as the registration target.

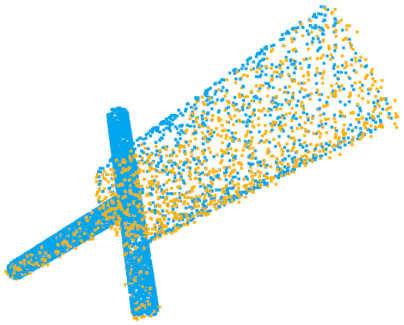

Fig. 5: ICP result with Gaussian noise.

B. Contact Projection via Signed Distance Fields

To maintain reliable contact between the fingers and the object throughout the manipulation, we propose a lightweight projection method based on signed distance fields (SDF). This method is used to post-process each intermediate configuration q of the hand such that all fingertips are in contact with the object's surface. The idea is to iteratively minimize the SDF value at each contact point using gradient descent while penalizing any unintended object motion. This is a low-frequency online algorithm. After each rotation operation, we execute the algorithm

1) Method of Contact Projection: The algorithm is summarized in Algorithm 1.

Algorithm 1 Contact Projection using SDF

Require: Initial configuration q, signed distance field $\text{SDF}(\cdot)$, contact finger indices $\mathcal{F}$, threshold ϵ
Ensure: Projected configuration q^*
1: **while** $\max_{f\in\mathcal{F}} |\text{SDF}(f)| > \epsilon$ **do**
2: Compute $\nabla_q \text{SDF}(f)$ for all $f \in \mathcal{F}$
3: Update $q \leftarrow q - \eta \sum_{f\in\mathcal{F}} \nabla_q \text{SDF}(f)$
4: Apply regularization to penalize object motion
return $q^* = q$

The projection algorithm aims to enforce fingertip contact by minimizing the signed distance between each fingertip and the object surface. Let $\text{SDF}(\mathbf{p})$ denote the signed distance function evaluated at point $\mathbf{p}$, such that $\text{SDF}(\mathbf{p}) = 0$ indicates the point lies exactly on the surface. For each joint configuration q, the fingertip positions $\mathbf{p}_f(q)$ are computed via forward kinematics.

To bring the fingertips onto the surface, we perform iterative gradient descent over the joint configuration:

$$q \leftarrow q - \eta \sum_{f\in\mathcal{F}} \nabla_q \text{SDF}(\mathbf{p}_f(q)) \quad (11)$$

Here, $\nabla_q \text{SDF}(\mathbf{p}_f(q))$ denotes the gradient of the SDF value at fingertip f with respect to q, computed via the chain rule:

$$\nabla_q \text{SDF}(\mathbf{p}_f(q)) = \frac{\partial \text{SDF}}{\partial \mathbf{p}_f} \cdot \frac{\partial \mathbf{p}_f}{\partial q} \quad (12)$$

Since q also encodes the pose of the manipulated object, naïvely updating q could result in undesired object displacement. To preserve object immobility during projection, we add a regularization term $\mathcal{R}(q)$ penalizing object motion:

$$\mathcal{L}(q) = \sum_{f\in\mathcal{F}} \text{SDF}(\mathbf{p}_f(q))^2 + \lambda \cdot \|\Delta T_{\text{object}}(q)\|^2 \quad (13)$$

where $\Delta T_{\text{object}}(q)$ measures the deviation of the object pose from its original state, and λ is a small positive weight. We stop the optimization once the maximum SDF value across all fingertips drops below a predefined threshold ϵ, indicating contact has been achieved.

This method serves as a fast, differentiable alternative to contact force computation, and is especially suitable for high-frequency replanning where strict force closure is not necessary.

2) Implementation Details: We represent each system state $q \in \mathbb{R}^{16}$ as a concatenation of 12 joint angles for three fingers (index, middle, thumb) and 4 degrees of freedom for the object pose. For each manipulation step, we use an internal SDF-based scene model to compute

both the signed distance value SDF(q) and its gradient ∇_qSDF with respect to each fingertip's configuration.

Because each finger participates to a different degree in the rotation operation, we assign each finger a custom learning rate scaling factor α_f. Specifically, in our implementation, the middle finger is given a significantly larger learning rate (e.g., $\alpha_{\text{middle}} = 0.05$), while index and thumb use smaller values (e.g., $\alpha_{\text{index}} = \alpha_{\text{thumb}} = 0.01$). This ensures efficient convergence without instability.

During each update, the contact-aware SDF loss is computed for each finger individually and their gradients are aggregated with the appropriate scaling:

$$\nabla_q \mathcal{L}_{\text{total}} = \sum_{f \in \mathcal{F}} \alpha_f \cdot \nabla_q \text{SDF}_f(q) \tag{14}$$

A regularization term is added to the objective to penalize unintended object motion, given that q also controls the object state. Gradient updates are clipped via ℓ_2-norm constraint to avoid instability.

Finally, our projection routine stops when the maximum SDF value across all active fingers is below a threshold ($\epsilon = 1.5 \times 10^{-3}$). The projected configuration q^* is then used to replace the diffusion model output before proceeding to the next manipulation step.

C. Force-Aware Correction via Linear Programming

Although contact projection guarantees contact, it does not explicitly consider the stability of contact and the real-time nature of the solution. To address this issue, we introduce a force-aware correction strategy that adjusts the generated trajectory by solving a grasping optimization problem after each step. We adopt the force closure linear programming-based optimization. Our method decouples the problem into two parts:

- A **linear program (LP)** that solves for feasible contact wrenches under grasp constraints, ensuring force balance and maximizing a stability margin;
- A **DS-based attractor construction**, which converts the resulting contact wrenches into fingertip attractor positions, guiding each finger toward its force target via a dynamical system.

Together, these modules refine the diffusion trajectory into one that is both physically feasible and graspable.

1) Linear Programming Formulation: We formulate a grasp quality optimization problem as a linear program (LP) that seeks to maximize the minimum margin η by which the contact wrenches satisfy the friction cone constraints, while maintaining force balance and respecting per-contact normal force limits.

a) Decision Variables.: Let $\lambda \in \mathbb{R}^{3n}$ be the stacked vector of contact wrenches at n contact points, where each $\lambda_i \in \mathbb{R}^3$ represents the force at the i-th contact in the world frame:

$$\lambda = \begin{bmatrix} \lambda_1^\top & \lambda_2^\top & \cdots & \lambda_n^\top \end{bmatrix}^\top$$

We also define a scalar variable $\eta \in \mathbb{R}_+$ representing the grasp quality margin to be maximized.

b) Grasp Matrix.: The grasp matrix $G \in \mathbb{R}^{m \times 3n}$ maps the contact wrenches to the net wrench on the object(m=3 here).

$$G = \begin{bmatrix} \mathbb{I}_{2\times 3} & \cdots & \mathbb{I}_{2\times 3} \\ [\text{skew}(r_1)]_{2,:} & \cdots & [\text{skew}(r_n)]_{2,:} \end{bmatrix} \in \mathbb{R}^{3\times 3n}$$

where r_i is the vector from the object center of mass to the i-th contact point, and skew(r) denotes the skew-symmetric matrix of r.

c) Friction Cone Constraints.: We approximate the friction cone at each contact using a polyhedral linearization with k facets. In the local contact frame, the directions spanning the cone are defined as:

$$\texttt{dirs} = \left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} \mu\cos\theta_j \\ \mu\sin\theta_j \\ 1 \end{bmatrix} \text{ for } j = 1, \ldots, k \right\} \tag{15}$$

$$\theta_j = \frac{2\pi j}{k} \tag{16}$$

These directions are assembled as rows of a local matrix $F_i^{\text{local}} \in \mathbb{R}^{(k+1)\times 3}$. We then transform F_i to the world frame via:

$$F_i^{\text{world}} = F_i^{\text{local}} R_i^\top$$

where $R_i \in SO(3)$ is the contact frame rotation matrix.

The full global constraint matrix is constructed as:

$$F = [(F_1^{\text{world}}, F_2^{\text{world}}, \ldots, F_n^{\text{world}})] \in \mathbb{R}^{(k+1)n\times 3n} \tag{17}$$

Then, the friction cone constraint becomes:

$$F\lambda \geq \eta \cdot \mathbf{1}_{(k+1)n} \tag{18}$$

d) Normal Force Constraints.: Let e be a selector vector that extracts the normal component of each λ_i. It is defined as the vertical stack of surface normals in the world frame:

$$e = \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_n \end{bmatrix}, \tag{19}$$

We impose the constraint that the normal force is non-negative and below a per-contact upper bound n_f:

$$0 \leq e^\top \lambda \leq n_f \tag{20}$$

e) Full Linear Program.: Putting all constraints together, the final LP is:

$$
\begin{aligned}
\max_{\lambda,\eta} \quad & \eta \\
\text{s.t.} \quad & G\lambda = 0 && \text{(force and torque balance)} \\
& F\lambda \geq \eta \cdot \mathbf{1} && \text{(friction cone constraints)} \\
& 0 \leq e^\top \lambda \leq n_f && \text{(normal force bounds)} \\
& \eta \geq 0
\end{aligned}
$$

This LP is solved at each time step of the manipulation to evaluate the grasp quality and correct unstable configurations.

2) DS-Based Attractor Construction: Once the contact wrenches λ are computed from the linear program, we convert them into target positions for each fingertip to follow. These attractor positions are computed such that, when followed, the resulting contact forces approximately reproduce the optimized wrench while maintaining grasp stability.

Let p_{com} denote the geometric center of the object (estimated from the point cloud), and let $r_i = p_i - p_{\text{com}}$ be the vector from the object center to the i-th contact point. For each finger i, we construct a local grasp Jacobian $G_i \in \mathbb{R}^{3\times 3}$, which maps contact force to object-frame wrench in a planar setting(see Grasp Matrix).

Let $\lambda_i \in \mathbb{R}^3$ be the optimized contact force at finger i, and let $K_x \in \mathbb{R}^{3\times 3}$ be a diagonal stiffness matrix defined as:

$$
K_x = \text{diag}(k_t, k_t, k_n), \quad k_t = 300,\ k_n = 600 \tag{21}
$$

We then compute the desired displacement $\Delta x_i \in \mathbb{R}^3$ of the fingertip as:

$$
\Delta x_i = K_x^{-1} G_i \lambda_i \tag{22}
$$

Finally, the attractor position $x_i^{(2)}$ for finger i is given by:

$$
x_i^{(2)} = x_i^{(1)} + \Delta x_i \tag{23}
$$

where $x_i^{(1)}$ is the current contact position of the i-th finger. These attractors are then used as the targets for a dynamical system controller, ensuring that the hand moves in a compliant way toward generating the desired contact forces during in-hand manipulation.

Fig. 6 provides a visual illustration of the DS-based attractor construction for three fingers. Each colored trajectory represents a fingertip path toward its optimized attractor:

- red, blue, and green curves correspond to fingers x_1, x_2, and x_3, respectively.
- The gray square markers $x_i^{(1)}$ denote the initial fingertip contact positions.
- The inverted triangle markers $x_i^{(2)}$ represent the final attractor positions computed via the grasp force λ_i and DS dynamics.
- The gray dashed lines show the direct linear displacement $\vec{x_i^{(2)} - x_i^{(1)}}$, used as reference for what purely kinematic correction would look like.
- The central intersection point, shared across the three paths, indicates the object's geometric center p_{com}, which serves as the rotational reference frame for computing each $r_i = p_i - p_{\text{com}}$.

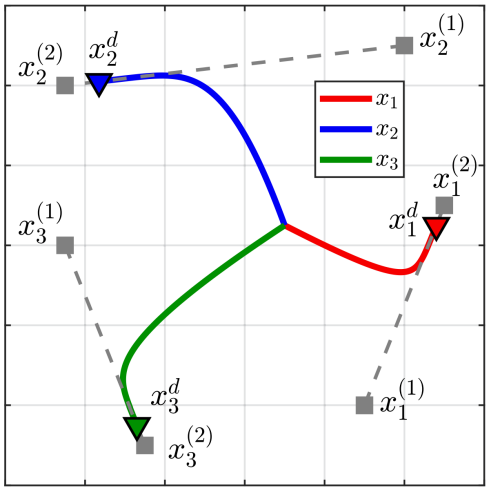

Fig. 6: Illustration of DS-based attractor construction. All fingers start at the same initial position and are guided toward their respective attractor $x_i^{(2)}$ based on LP-optimized contact forces. The common central point represents the object centroid p_{com}, computed from the observed point cloud. The trajectories are computed using grasp-specific stiffness and wrench mappings, ensuring coordinated movement toward stable contact configurations.

Now, we summarize the whole algorithm in Algorithm 2.

Algorithm 2 Force-Aware Trajectory Correction via LP and DS Attractor

Require: Contact positions $\{p_i\}$, contact normals $\{n_i\}$, object point cloud $\mathcal{P}$, friction coefficient μ
Ensure: Target attractor positions $\{x_i^{(2)}\}$ for each finger

Compute object centroid $p_{\text{com}} \leftarrow \text{mean}(\mathcal{P})$
Compute relative contact vectors $r_i \leftarrow p_i - p_{\text{com}}$
Construct grasp matrix G using $\{r_i\}$ and $\{n_i\}$
Build friction cone matrix F and normal selector e in world frame
Define normal force limits n_f, initialize LP variables λ, η
Solve the following LP:

$$\begin{aligned} \max_{\lambda,\eta} \quad & \eta \\ \text{s.t.} \quad & G\lambda = 0 \\ & F\lambda \geq \eta \cdot \mathbf{1} \\ & 0 \leq e^\top \lambda \leq n_f \\ & \eta \geq 0 \end{aligned}$$

for each finger i **do**
    Extract contact force $\lambda_i \in \mathbb{R}^3$
    Compute local grasp map $G_i \in \mathbb{R}^{3\times 3}$
    Compute displacement $\Delta x_i \leftarrow K_x^{-1} G_i \lambda_i$
    Set attractor $x_i^{(2)} \leftarrow p_i + \Delta x_i$
return $\{x_i^{(2)}\}$ as the final corrected trajectory targets

3) Implementation in Closed-Loop Control: At runtime, we deploy the LP+DS attractor pipeline as part of the control loop that adjusts the planned finger configurations based on current contact observations.

III. Results

A. Object Pose Estimation from Point Cloud

We first evaluate the accuracy of object pose estimation based on point cloud alignment using the ICP pipeline described before. The estimated poses are compared against ground truth across all three rotational degrees of freedom: pitch, roll, and yaw.

Fig. 7 presents the trajectory of estimated pitch and roll angles across 12 sequential manipulation steps. The solid yellow lines denote ground truth measurements, while the dashed orange lines correspond to the ICP-based estimates. Both estimates exhibit consistency with the ground truth, indicating reliable reconstruction from raw depth data.

Fig. 8 shows the comparison for yaw angle. While larger deviations appear during certain steps, the overall trend of the estimated orientation remains aligned with the reference trajectory.

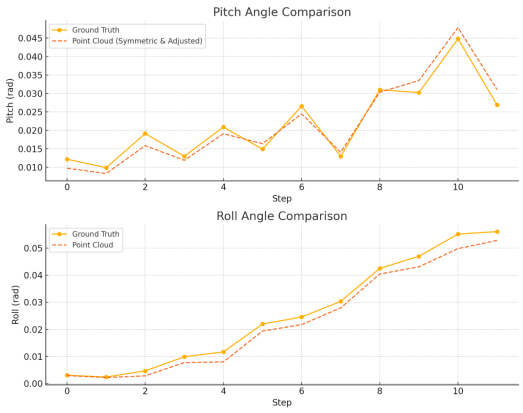

Fig. 7: Comparison of pitch and roll angle estimation from point cloud alignment versus ground truth across 12 manipulation steps.

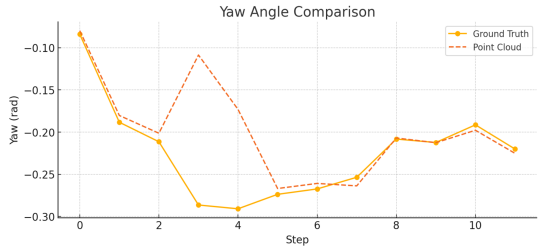

Fig. 8: Comparison of yaw angle estimation from point cloud alignment versus ground truth across 12 manipulation steps.

B. Trajectory Correction: Projection vs. Diffusion

To evaluate the effectiveness of our SDF-based projection method, we compare its performance with a baseline diffusion-only strategy. Both approaches begin by sampling a trajectory from the diffusion model. The baseline directly executes this trajectory step-by-step. In contrast, the projection method applies a contact projection step after each rotation, enforcing fingertip-object contact before continuing to the next action.

We report two results in Fig. 9 and Fig. 10:

- **Yaw Difference Per Trial:** For each trial, we compute the total yaw angle achieved after executing the full trajectory. The figure shows the difference $\Delta_{\text{yaw}} = \text{Compare} - \text{Baseline}$. Positive values indicate that the projection method achieved greater overall rotation.
- **Yaw Difference Per Step:** We also report per-step yaw differences within each trial to analyze local performance. Positive values again indicate that projection outperformed baseline at that step.

Both metrics were computed over 120 randomly initialized trials. The projection method achieves consis-

tently higher yaw rotation than the baseline, particularly in early and mid stages. The average control frequency remains feasible for online application, with a mean computation time of $0.166\,\mathrm{s}$ per step. In 120 trials, we have 19 trials where the total rotation angle of the compared method is less than the baseline. The success rate is 84.17%

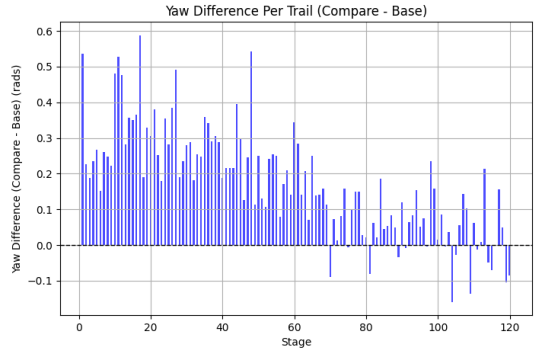

Fig. 9: Yaw angle difference per trial: Projection vs. Diffusion. Positive bars indicate larger rotation using projection.

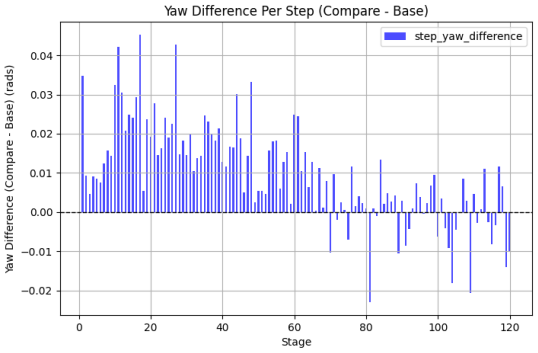

Fig. 10: Yaw angle difference per step: Projection vs. Diffusion. Positive bars indicate larger rotation using projection.

C. Trajectory Correction: LP + DS vs. Diffusion

To further validate the effectiveness of our proposed force-aware correction pipeline, we compare our full method (LP + DS) against a diffusion-only baseline.

Baseline is to sample a full trajectory from the diffusion model and directly execute it, while the compared Method is to sample a trajectory from the diffusion model, but apply LP-based grasp optimization, followed by DS-based attractor correction after executing each motion segment for each trial

The average time per trajectory (12 steps) using LP + DS is $0.0598\,\mathrm{s}$, making it suitable for high-frequency online use. Fig. 11 and Fig. 12 show the results over 30 randomized trials:

- **Mean Final Yaw:** Fig. 11 compares the final yaw rotation achieved by each method across trials. The LP + DS method outperforms the baseline in nearly all cases, with an average improvement of $0.284\,\mathrm{rad}$.
- **Mean Stepwise Yaw Delta:** Fig. 12 compares the average yaw change per step. Again, the LP + DS strategy achieves more consistent and larger rotational increments.

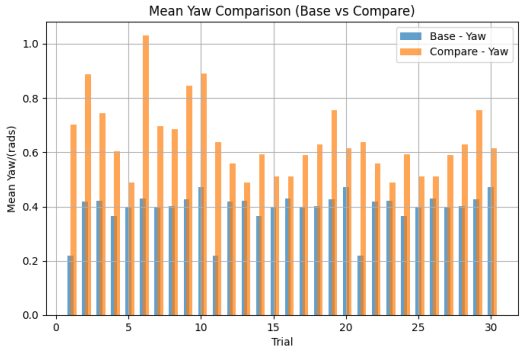

Fig. 11: Mean final yaw achieved per trial. Orange bars indicate LP + DS, blue bars are the diffusion-only baseline.

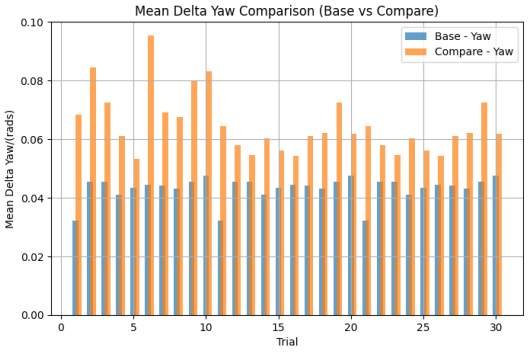

Fig. 12: Mean per-step yaw change across trials. LP + DS consistently achieves greater rotational increments.

IV. Discussion

A. Pose Estimation from ICP and Point Cloud

The pose estimation results shown in Fig. 7 and Fig. 8 demonstrate that our ICP-based pipeline can reasonably approximate the ground truth orientation of the object using only depth images and segmentation

masks. The estimated pitch and roll angles track the true values closely, with consistent trends and small absolute deviations. This suggests that the estimated point cloud centroid and the ICP alignment process are sufficiently accurate in constraining motion in the vertical plane.

The yaw angle, however, exhibits slightly larger deviations, particularly in the early steps of manipulation. This can be attributed to the following factors:

- **Partial observability.** Due to occlusion from the hand and camera viewpoint, the input point cloud often lacks sufficient lateral features(blue stick on the screwdriver), making ICP alignment under yaw rotation less constrained.
- **Symmetry ambiguity.** The screwdriver model used is rotationally symmetric around its main axis, causing ICP to occasionally converge to nearby local minima with small yaw misalignments.

Despite these limitations, the overall trajectory of the estimated yaw angle still follows the ground truth trend. Even with some bias in the early steps but it will recover soon. The accuracy is sufficient to enable our planner to update contact points and recompute force corrections with meaningful geometric grounding.

B. Projection-Based Correction: Effects and Limitations

Our projection-based correction strategy for enforcing contact consistency during in-hand manipulation improves the overall yaw rotation performance compared to the diffusion-only baseline by projecting each fingertip onto the object surface (SDF ≈ 0) after every rotation step. The enforced proximity between the hand and the object reduces contact loss and helps maintain manipulation stability over multiple steps.

However, this method exhibits several notable limitations:

- **Discontinuous contact behavior.** Since the projection step is applied independently at each rotation interval, the resulting contact trajectory may exhibit discontinuities. Fingertips are sometimes projected abruptly from one surface region to another, causing repetitive "re-grasping" behavior on the object. This can introduce undesired velocity spikes or unnatural motion profiles.
- **Object pose perturbation.** Because the projection modifies the full configuration q—which includes both hand and object pose parameters—it may inadvertently shift the object. Although a regularization term is introduced to keep the object approximately fixed, the optimizer may still allow small but noticeable object drift to satisfy fingertip SDF constraints.

These issues highlight the importance of integrating both geometric and force-based reasoning. While the projection method offers an easy way for optimization and performs reasonably well in idealized settings, it performs badly in scenarios requiring smooth, continuous control.

C. Force-Aware Correction: Efficacy Beyond Force Closure

Our LP + DS-based correction pipeline achieves significantly improved yaw rotation performance compared to both the diffusion-only and projection-based baselines, as shown in Section IV. At each time step, the LP solves for a feasible contact wrench configuration, and the DS component maps these forces into per-finger attractor motions. This enables the planner to maintain stable manipulation over time, even without access to full physical dynamics.

It is important to emphasize, however, that our formulation does **not enforce strict force closure**. Specifically, we do not model the screwdriver's contact with the table as a support wrench, nor do we include contact forces from the environment in the LP formulation. This means the object could not remain suspended in air under arbitrary external disturbances. Instead, the LP focuses on maximizing a relaxed grasp quality margin η among the fingertips, subject to friction and normal force limits.

In practice, the optimized grasp margin η typically remains small—on the order of 0.01—which further confirms that the solution lies far from the regime of full force closure. Nevertheless, we observe robust manipulation and consistent task execution. This suggests that, for constrained in-hand manipulation tasks such as screwdriver rotation, a **weaker grasp condition** that prioritizes localized force balance and smooth contact evolution is sufficient.

V. Conclusion

In this work, we present a hybrid planning framework for in-hand screwdriver rotation that couples diffusion-based trajectory synthesis with two lightweight contact-refinement modules: (i) signed-distance-field (SDF) projection for rapid geometric feasibility, and (ii) a force-aware linear-programming + dynamical-system (LP + DS) optimizer that enlarges grasp margins within each replanning cycle. Experiments on 150 randomized initial poses in Isaac Sim substantiate the framework's effectiveness: both refinement strategies markedly enhance rotational accuracy and contact persistence over a diffusion-only baseline, while the LP + DS variant yields the highest performance improvement.

Our experiments show that both contact projection and LP + DS correction improve rotational consistency and contact stability over diffusion-only baselines. In particular, the LP + DS strategy achieves the best performance across multiple metrics while maintaining computational efficiency suitable for online replanning.

However, we emphasize that our approach does not rely on strict force closure. The screwdriver's contact with the environment (e.g., the table) is not modeled as part of the optimization, and the solved grasp margins η are typically small ($\sim$0.01). This highlights a limitation of the current formulation: although effective in practice, it lacks formal guarantees of grasp stability. Furthermore, projection-based methods may lead to discontinuous contact motions and unintended object displacement due to SDF constraints applied directly to the joint space.

Future work will incorporate environmental contact points into the wrench space and better modeling of force propagation through the object. And apply the attractor method in a high-frequency controller.

The results verify the effectiveness and real-time potential of "diffusion initialization + minimization constraint correction" in visual-driven hand operation, laying the foundation for subsequent fusion environment contact and real hardware deployment.

VI. Appendix

Code Repository

The visualization results and the full implementation of our project can be found on the Github Website.

References

[1] T. Power and D. Berenson, "Constrained stein variational trajectory optimization," *IEEE Transactions on Robotics*, 2024.

[2] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[3] A. Kumar, T. Power, F. Yang, S. A. Marinovic, S. Iba, R. S. Zarrin, and D. Berenson, "Diffusion-informed probabilistic contact search for multi-finger manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2410.00841

[4] F. Khadivar and A. Billard, "Adaptive fingers coordination for robust grasp and in-hand manipulation under disturbances and unknown dynamics," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3350–3367, 2023.

[5] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021. [Online]. Available: https://arxiv.org/abs/2108.10470

[6] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 3212–3217.