PRESENTATION OF VIVIDEX: LEARNING VISION-BASED DEXTEROUS MANIPULATION FROM HUMAN VIDEOS

Victorin Turnel & Thomas Winninger
Master MVA, ENS Paris-Saclay
Gif-sur-Yvette, France
`victorin.turnel@ens-paris-saclay.fr, thomas.winninger@telecom-sudparis.eu`

ABSTRACT

This document presents a comprehensive study of the ViViDex framework, a novel approach designed to improve dexterous manipulation in multi-fingered robots hands using visual signals. We begin by providing a detailed summary of the original article. Next, we identify specific limitations within the existing method and propose a new module to address them. Finally, we show some experiments carried out and offer a critical feedback on the framework.

1 GENERAL PRESENTATION

1.1 CHALLENGES AND GOALS

Dexterous manipulation with multi-fingered robot hands remains a significant challenge in robotics, particularly when manipulating a variety of objects in diverse poses. A major limitation of prior works is their reliance on privileged information, such as ground-truth object states or CAD models. Obtaining such data is often unrealistic in real-world scenarios due to sensor limitations.

ViViDex is proposed as a solution to these limitations, taking inspiration from the natural dexterity of humans. The core objective is to translate visual signals into the high-dimensional control commands necessary to replicate human-like object manipulation. This framework specifically leverages the high availability of human videos capturing hand-object interactions to facilitate policy training.

To address these challenges, ViViDex consists of three main modules:

- Reference trajectory extraction: This module aims to extract human hand poses and object trajectories from video demonstrations.
- Trajectory-guided state-based policy learning: This step involves training a state-based policy using Reinforcement Learning (RL). It utilizes trajectory-guided rewards to generate trajectories that are both visually natural and physically plausible.
- Unified vision-based policy learning: The goal of this final step is to train a visual policy without utilizing any privileged information, by rolling out successful episodes from the optimized state-based policies.

1.2 METHOD

1.2.1 REFERENCE TRAJECTORY EXTRACTION

The goal of this module is to extract hand and object poses from video demonstrations. To achieve this, the method relies on MANO to represent the poses and shapes of the hand, resulting in 3D locations of joints $\psi_h \in \mathbb{R}^{21\times 3}$. To minimize the geometric differences between the human and robot hands, the motion retargeting is formulated as an optimization problem over a video of length T. Using the NLopt solver (Steven, 2019), the objective function is defined as:

$$\min_{q_r^t} \sum_{t=1}^{T} \left\| \hat{x}_{rj}^t(q_r^t) - \psi_{hj}^t \right\|_2^2 + \alpha \left\| q_r^t - q_r^{t-1} \right\| \tag{1}$$

with q_r^t represents the robot joint rotation angles, ψ_{hj}^t is the human hand tip and middle phalanx position and $\hat{x}_{rj}^t$ the robot counterparts solved via forward kinematics. The first term aims to minimize the L^2 difference between the 3D locations of the robot joints and the human hand. The second term acts as a regularization to prevent sudden changes in the robot pose over time.

1.2.2 STATE-BASED POLICY

The previously generated trajectories are visually plausible but are not necessarily physically plausible. To address this challenge, a state-based policy is trained using RL to recover physically plausible trajectories, using the reference trajectories as a reward function. The architecture consists of actor and critic MLPs taking as input both robot and object states to predict the robot control commands.

The RL training process is divided into two stages. The first step, called the “pre-grasp” stage, focuses on learning a coarse motion to quickly approach the object without establishing physical contact. To achieve this, the following reward function is defined:

$$R_p = \sum_{t=1}^{T_p} 10 \cdot \exp\Big(-10 \cdot \left\| x_{rt}^t(q_r^t) - \hat{x}_{rt}^t \right\|_2^2\Big) \tag{2}$$

where T_p denotes the length of the pre-grasp steps, $\hat{x}_{rt}^t$ is the robot finger tip position in the reference trajectory at timestep t, and x_{rt}^t is the current robot finger tip position.

The manipulation stage begins when the robot successfully reaches its pre-grasp configuration. In this stage, the objective is to refine the trajectory with finer and more constrained motions to bring the object to the desired configuration. This step is guided by the following reward function:

$$R_m = \sum_{t=T_p+1}^{T_r} \lambda_1 R_m^h + \lambda_2 R_m^o + \lambda_3 \mathbf{1}_{\text{cont}} + \lambda_4 \mathbf{1}_{\text{lift}} \tag{3}$$

where T_r is the length of the reference trajectory. R_m^h constrains the hand motion similarly to (2). The term $R_m^o = \exp\Big(-\alpha_1 \Big(\left\| x_o^t - \hat{x}_o^t \right\|_2^2 + \alpha_2 \Phi\big(\theta_o^t, \hat{\theta}_o^t\big)\Big)\Big)$ geometrically constrains the object motion, with x_o^t and $\hat{x}_o^t$ representing the current and reference object positions at timestep t, and $\Phi(.)$ representing the angular distance between the current object orientation and the reference $\hat{\theta}_o^t$. Finally, $\mathbf{1}_{\text{cont}}$ and $\mathbf{1}_{\text{lift}}$ compute the number of fingertips in contact with the object and the bonus points for lifting the object off the table, respectively.

To enhance the generalization capability of the state-based policy, reference trajectory augmentation is introduced during RL training. This consists of diversifying the initial object positions, rotations, and target positions to increase the variety of scenarios. The transformation applied to the object is propagated to the trajectory using interpolation between the original object trajectory and the target pose.

1.2.3 VISION-BASED POLICY

The previous policy was trained using privileged information, such as the object state or the target position. However, in a real-world scenario, this information is not trivial to obtain reliably through sensors. To overcome this limitation, a vision-based policy is implemented to predict robot commands using only the robot's proprioceptive state and 3D scene point clouds as inputs.

To achieve this, the first step consists in creating training data for the policy. The optimized state-based policy is used to generate diverse, physically plausible trajectories. For each trajectory and timestep, the robot command is associated with the 3D point cloud expressed in the world coordinate system, denoted as $\mathrm{PC}_w \in \mathbb{R}^{N\times 3}$, obtained from a depth camera where N is the number of points.

These raw point clouds PC_w are not used directly. Instead, they are transformed into the target coordinate system and various robot joint coordinate systems (specifically the palm and fingertips). These transformed point clouds are combined into a single representation $\mathrm{PC} \in \mathbb{R}^{N\times 3(j+3)}$, where j is the number of fingertips. This unified representation is fed into PointNet (Qi et al., 2017) to extract visual features that enable the prediction of control commands.

Finally, two different strategies are compared to train the vision-based policy. On the one hand, Behavior Cloning (BC) directly takes the transformed point clouds and robot states to predict the command. On the other hand, a 3D Diffusion Policy uses the extracted visual features from PointNet (Qi et al., 2017) as a global condition for a denoising model. In both cases, the models are trained using an L_2 loss.

1.2.4 Data, experiments & results

Data : The experimental evaluation relies primarily on the DexYCB dataset, which contains human demonstration videos of hand-object interactions. The study focuses on five specific objects: a mustard bottle, a tomato soup can, a sugar box, a large clamp, and a mug. To assess the generalization capabilities of the model, ten additional "unseen" objects are introduced during the testing phase. Unlike previous methods that typically require around 100 videos per object, this framework utilizes only one to three videos per object for training. The experiments are conducted in two simulation environments: MuJoCo with an Adroit hand for benchmarking against state-of-the-art methods and SAPIEN with an Allegro hand attached to a UR5 arm to better mimic the real-world hardware configuration. Finally, the approach is validated on a physical robot setup using an Allegro hand and a RealSense depth camera.

Baselines : To validate the performance of ViViDex, several baselines are established. The state-based policy is compared against DexMV. Since the original DexMV training data was not public, a variant was re-trained using 20 DexYCB videos for a fair comparison. Internal ablation studies were also conducted to evaluate the reward functions: "R1" excludes hand rewards during pre-grasp, and "R2" excludes them during manipulation. For the vision-based policy, the study compares Behavior Cloning against the proposed 3D Diffusion Policy.

Metric : The primary metric for evaluation is the Success Rate. For the relocation task, this is measured using a standard threshold of 10cm ($\mathrm{SR}_{\{10\}}$) and a stricter threshold of 3cm ($\mathrm{SR}_{\{3\}}$). Additionally, trajectory quality is assessed using the average object position error (E_o) and fingertip position error (E_h) relative to the reference trajectory.

Results : The results demonstrate that the state-based policy significantly outperforms prior works. While DexMV struggles with complex or thin objects like the sugar box, ViViDex achieves a 100% success rate on the relocate task using only a single training video per object. Ablation studies confirm that incorporating the hand trajectory reward in both the pre-grasp and manipulation stages is crucial; without it, policies tend to produce unstable grasps or unnatural motions. On more complex tasks such as "Pour" and "Place Inside," the method achieves success rates of 97% and 68% respectively, compared to 27% and 31% for the best previous baselines.

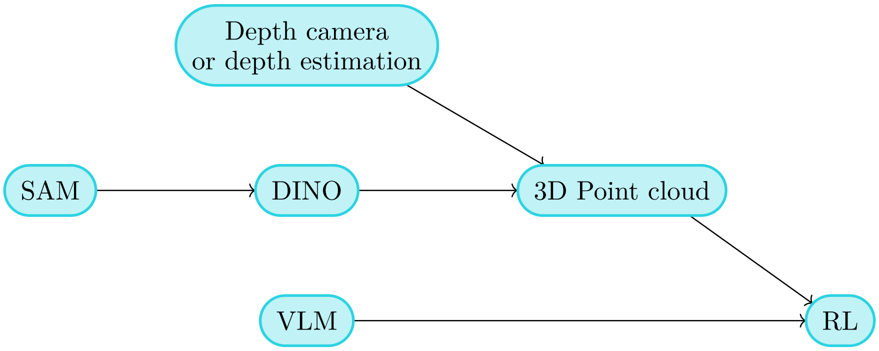
Regarding the vision-based policy, the experiments highlight the importance of visual representation. Increasing the point cloud density and transforming points into the hand-centric coordinate system improved the success rate on seen objects from 81% to 95%. Furthermore, the 3D Diffusion Policy demonstrated superior robustness compared to Behavior Cloning, achieving a 99% success rate on seen objects and significantly improving generalization to unseen objects (50% vs 41%). These findings were confirmed by real-world experiments where the diffusion-based unified policy successfully manipulated both seen and unseen objects.

2 Limitations & Improvements

2.1 Perception module

The original ViViDex pipeline relies on 3D scene point-cloud features and a learned pose estimator, which can be noisy, especially in when using a single video. To strengthen this pipeline, we propose to add a perception module to enrich the visual observations used during the trajectory extraction and the RL phase.

We explored using: (1) objects segmentation and identification with SAM v3 (Carion et al., 2025), which can be used to extract only the information relevant to the hand and the object, removing everything else; (2) dense per-pixel feature embeddings with DINO v3 (Siméoni et al., 2025), which can provide useful information about the physics of the objects, and may help generalisation; (3) a monocular depth estimation (Birkl et al., 2023; Ranftl et al., 2022), to improve the position precision in 3D - though this part might be useless if the depth camera used in the original work (Chen et al., 2025) is powerful enough.



The objective of the perception module is thus to produce temporally consistent object localization, appearance descriptors, and 3D geometric cues, to improve state estimation, and the visual policy.

2.1.1 Video segmentation

The Segment Anything Model (SAM) (Carion et al., 2025) provides robust instance-level masks for the hand, the target objective, while making it easy to remove background elements and noise that could reduce the quality of the 3D point cloud.

Furtheremore, it generalize well without labels, while keeping a temporal stability, especially when paired with feature-based matching. This can be extremely useful to help the robot adapt to new objects in an unsupervised manner.

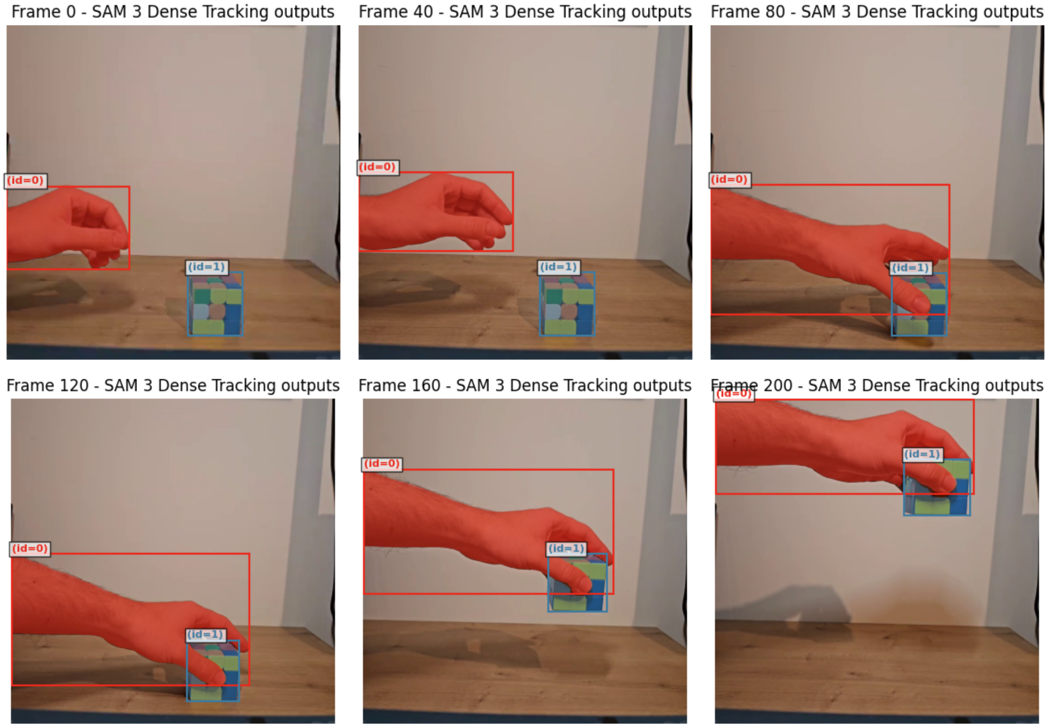

Frame 0 - SAM 3 Dense Tracking outputs
Frame 40 - SAM 3 Dense Tracking outputs
Frame 80 - SAM 3 Dense Tracking outputs
Frame 120 - SAM 3 Dense Tracking outputs
Frame 160 - SAM 3 Dense Tracking outputs
Frame 200 - SAM 3 Dense Tracking outputs
(id=0)
(id=1)

Figure 2: Segmentation and tracking with SAM v3 (Carion et al., 2025). The model is provided the video, along with textual clues, in this case “Hand and object”. The textual clues can also be used to avoid tracking a specific object.

In the ViViDex pipeline, SAM can be used as a first layer of perception to produce segmentation masks. These segmentation masks can then be feed into DINO, who needs a target to be able to produce features.

2.1.2 Dense per-pixel feature embedding

While SAM can detect, segment, and track objects, it does not provide information about the fine-grained appearance, texture similarity, or task-relevant features, like graspable surfaces. For this reason, we combine SAMv3 with DINOv3 (Siméoni et al., 2025).

DINO provides self-supervised dense embeddings for every pixel or patch, which adds information about texture, material, and local geometry, information relevent for the robotic task. In short, SAM defines *where* the object is, and DINO defines *what* the object is and *how* it should be interacted with.

In Figure 3, the model understood the topology of the object, as it clearly makes the difference between the body of the mugs (in red) and the handles (in light green). It also identified a difference in the physics of the objects. The left image’s mug has a particular bottom (which can be seen in Figure 4), it is not made of the same material, and it’s unusual for mugs, so it has a distinct darker color. The only visible problem is the Pikachu image on the right image’s mug, which might induce errors – though the images are a projection, the vectors lie in high dimension and might carry much more information.

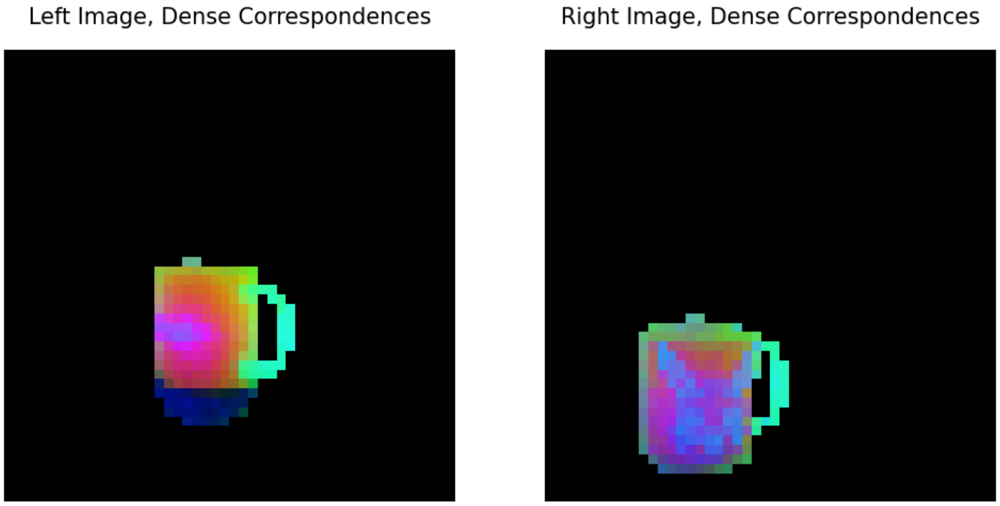


Figure 3: Dense matching with DINOv3 (Siméoni et al., 2025).

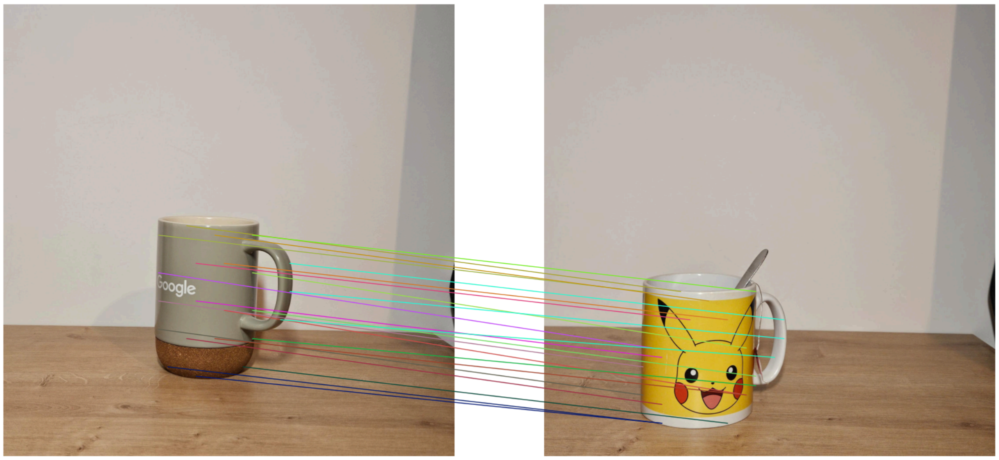


Figure 4: Sparse matching with DINOv3 (Siméoni et al., 2025).

Using a separate perception module comes at a cost. SAMv3 memory consumption peaked at 10.9Gb (VRAM), while DINOv3′s peaked at 2.2Gb (VRAM), though choosing different, specialized models, might reduce the memory footprint. All experiments ran in seconds in a laptop 4090 RTX GPU (16Go VRAM). The DINOv3 model used was `dinov3_vitl16_pretrain_lvd1689m`.

2.1.3 Additional information with VLMs

Vision–Language Models (VLMs) combine visual and textual processing in a unified framework, enabling models to interpret images while producing semantically rich descriptions of objects, relationships, and context. This multimodal capability allows even compact models – like the 3B Ministral 3 Instruct – to provide scene-level cues that go beyond standard feature extraction.

Recent work has shown that such models can enhance robot perception by adding high-level semantics to tasks like scene understanding, task planning, and active perception. For instance, a survey highlights the benefits of VLMs in robot vision and multimodal fusion (Han et al., 2025), and studies show improved perception through embedding-level integration (Tavassoli et al., 2023), 3D-grounded reasoning for planning (Tang et al., 2025), and active viewpoint selection (Sripada et al., 2024).

We performed a simple experiment on the new Mistral 3 model family. Given the image of the scene, the model was tasked to describe the scene, the object, and provide clues on how to handle it. The prompt was: "Describe the object in a clear and concise way. The information will be used by a robot which will be tasked to grasp the object and move it.".

Ministral 3 3b (0.55s):

The object is a cup with a handle.

Here's a detailed description for robotic manipulation:

- Shape: Cup with a cylindrical body and a rounded base.
- Size:
  - Height: Approximately 12 cm (4.7 inches).
  - Diameter of the cup: Around 6 to 8 cm (2.5 to 3.1 inches).
  - Diameter of the handle: About 2 to 3 cm (0.8 to 1.2 inches) in width.
- Material:
  - The cup appears to have a yellow outer surface with a white inner surface.
  - The handle is silver or metallic grey.
- Handle Position: Single handle on the right side, positioned roughly at the top of the cup's body.
- Features:
  - Tray/Inner Surface: White and smooth, likely for containing liquid.
  - Exterior Design: The cup has a Pikachu face with:
    - Yellow color with black accents around the eyes and mouth.
    - Black triangular ears.
    - Red cheeks and eyes.
    - A silver or metallic spoon resting on the handle.
- Weight Estimation: Based on visual cues, it appears to be lightweight but solid. The cup is likely ceramic or plastic.
- Orientation:
  - For stability when grasping, place the handle in the robot's grasp area and ensure that the base of the cup is flat against a surface.
  - Avoid placing the Pikachu face downwards to prevent damage to the design.

To safely pick up this object:

1. Use the handle for support.
2. Ensure the cup is stable on its flat base before lifting it.
3. The robot should grasp the handle with a firm grip, avoiding the edges of the spoon or the cup's rim.

Answer 1: (Mistral 3 3b Instruct). The model provided interesting information, like the nature of the object (a cup), it's physics (lightweight, solid), and instruction on how to handle it (grasp the handle at the right). However, it cannot be used without fine-tuning or better instruction, as it still provides a lot of useless information (Pikachu), and incorrect information (dimensions should not be measured with a VLM).

Ministral 3 14b Reasoning (14.3s):

The object consists of a mug with a handle and a metallic spoon resting on top of it. Here are the key details for grasping and moving the mug:

Mug:

- Shape: Cylindrical with a handle attached at an angle (approximately 45 degrees).
- Material: Ceramic, indicating fragility; handle is thicker than the body for stability.
- Colors:
    - Exterior: Yellow with black and red design
    - Interior: White

Spoon (resting on top of mug):

- Shape: Long handle with a flat blade; resting horizontally, parallel to the table surface.
- Material: Metallic.
- Color: Silver or metallic.

Surface:

The object is placed on a wooden surface. No other objects are in close proximity that could interfere with grasping. The spoon's position on top of the mug may interfere with secure grasping and should be considered during the picking process."

Answer 2: (Ministral 3 14b Reasoning). Using a better model yielded much better results, without the need of fine-tuning or using a different prompt. It properly identified important information whithout adding to much noise, or guessing sizes.

These examples show promising results, as – apart from memory consumption – using pretrained VLM gives free additional information on the object or the scene.

Both models are quantized (Q4_K_M), and operate with a window size of 4096. The weights can be found on Huggingface: `mistralai/Ministral-3-14B-Reasoning-2512`, and `mistralai/Ministral-3-3B-Instruct-2512`. All experiments were run on a 4090RTX laptop (16G VRAM).

2.2 Unified reward loss for state-based policy

2.2.1 Limitation of the Current Framework

The current method artificially divides the task into two distinct phases: the approach phase (R_p) and the manipulation phase (R_m). These two stages are rigidly separated by a configuration threshold that triggers the switch from one step to the other.

This binary separation renders the overall motion unsmooth by creating two independent and "specialized" movements. For example, the robot can successfully reach the proximity of the object during the first step (R_p is high) but arrive with the wrong wrist orientation, making the manipulation in phase 2 physically impossible. In such a scenario, the robot is forced to execute sharp, unnatural motions during the transition to correct its pose.

2.2.2 Proposed Strategy: Transition Term

To create a unified reward function, we introduce a transition term that facilitates a smooth progression between the two stages. Specifically, this term enables the hand to progressively adopt the optimal pose for the subsequent manipulation as the distance between the hand and the object decreases:

$$R_{\text{unified}(t)} = R_{\text{reach}(t)} + w(d_t) \cdot R_{\text{hand\_shape}(T_p)} + \mathbf{1}_{\text{contact}} \cdot R_{\text{object\_track}(t)} \tag{4}$$

where:

- $R_{\text{reach}(t)}$: Standard distance minimization to the current reference frame t (approaching the object).
- $w(d_t)$: A sigmoid gating factor based on the hand-object distance ($0 \to 1$ as distance decreases).
- $R_{\text{hand\_shape}(T_p)}$: This term minimizes the difference between the robot's current hand pose and the reference hand pose at the specific timestamp of the grasp (T_p). It acts as a fixed goal that does not vary with time t.
- $R_{\text{object\_track}(t)}$: The original object tracking term, strictly gated by physical contact or a temporal threshold ($t > T_p$) to respect causality.

2.3 Improving signal quality

2.3.1 Limitation in the current framework

The standard retargeting optimization described in equation (1) operates directly on the raw estimated poses ψ^t_{hj} extracted from videos. To maintain motion smoothness the method relies solely on a regularization term $\alpha\left\|q^t_r - q^{t-1}_r\right\|$ with a fixed coefficient ($\alpha = 4e^{-30}$).

However, we identified several drawbacks regarding noise management:

- By feeding raw sensor data directly into the solver, the optimization objective treats high-frequently sensor noise (like jitter) as a valid target to track.
- With such a low regularization parameter ($\alpha \approx 0.004$), the penalty is not strong enough to prevent the robot not to track the noise leading to continuous vibrations. Conversely, increasing α to filter this noise would introduce latency during movements.

2.3.2 Proposed strategy : Kalman filtering

To address the previously mentioned limitation, we integrated a linear Kalman filter as a perception pre-processing step. Instead of feeding raw and noisy observations directly into the inverse kinematics solver, we estimate the optimal state of the target with a filter to reduce the noise.

The Kalman filter operates recursively in two phases:

- Prediction : It projects the current state forward using a constant velocity model
- Correction : It refines this prediction using the new noisy measurement from the camera

We define the state vector $\boldsymbol{x}_k$ for each fingertips as a 6 D vector containing position and velocity:

$$\boldsymbol{x}_k = \left[p_x, p_y, p_z, v_x, v_y, v_z\right] \tag{5}$$

Therefore, the system is modeled as a linear discrete-time process:

$$\boldsymbol{x}_k = \boldsymbol{F}\boldsymbol{x}_{k-1} + \boldsymbol{w}_k \tag{6}$$

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{v}_k \tag{7}$$

where $\boldsymbol{F}$ is the state transition matrix, $\boldsymbol{H}$ is the observation matrix, and $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ represent the process and measurement noise covariances respectively.

2.4 Curriculum Learning and Physical Domain Randomization

2.4.1 LIMITATIONS IN THE CURRENT FRAMEWORK

The current ViViDex implementation trains the vision-based policy by aggregating data from five training objects simultaneously, following Protocol 2 Moreover, the “Reference Trajectory Augmentation” section described in the paper is restricted to geometric transformations, such as initial position and rotation randomization. While this strategy effectively enhances the generalization capability of the framework to some extent, it ignores physical dynamics. This limitation likely contributes to the performance drop observed on unseen objects.

Indeed, the current “flat” training strategy has two major drawbacks:

- **Optimization Inconsistency:** The network is forced to learn how to handle complex non-convex geometries (like the handle of a mug) simultaneously with simple geometries. This steep learning curve can lead to sub-optimal convergence or “forgetting” of simple tasks.
- **Sim-to-Real Gap:** The lack of variation in physical parameters (friction, mass, damping) makes the policy brittle when deployed on a real robot, where physical properties differ from the simulator settings.

2.4.2 PROPOSED STRATEGY: CURRICULUM & RANDOMIZATION

We propose a two training strategies to enhance the robustness and generalization of the Vision-based policy.

2.4.2.1 GEOMETRIC CURRICULUM LEARNING WITH SYNTHETIC AUGMENTATION

Instead of shuffling all object data randomly, we organize the training data by geometric complexity. We introduce synthetic primitives to bootstrap the learning process.

- **Stage 1 - Simple Convex Synthetic Objects:** The policy is first trained on generated synthetic objects with simple primitive geometries (cubes, spheres, cylinders). This allows the PointNet encoder to learn basic spatial features and fundamental grasping approaches without the noise of complex textures or shapes.
- **Stage 2 - Simple Convex Real Objects:** The training transitions to real objects from the dataset that possess simple geometries (e.g. Tomato Soup Can, Sugar Box). This bridges the gap between synthetic primitives and realistic object textures/dimensions.
- **Stage 3 - Complex Non-Convex Real Objects:** We progressively introduce objects with complex affordances and self-occlusions (e.g., Mug, Large Clamp). The goal is to fine-tune the representation to handle intricate grasp planning.
- **Stage 4 - Complex Non-Convex Synthetic Objects:** Finally, to maximize generalization to unseen scenarios, the model is trained on procedurally generated complex synthetic objects. This prevents overfitting to the specific dataset distribution.

2.4.2.2 PHYSICAL DOMAIN RANDOMIZATION

To prepare the visual-based policy for real-world deployment, the idea would be to generate the training data (rollouts from the state-based policy) by varying the physical properties of the simulation environment:

- Object mass and inertia.
- Joint damping and actuation noise.
- Surface friction coefficients between the hand and the object.

3 EXPERIMENTS

3.1 IMPROVING SIGNAL QUALITY: KALMAN FILTERING

To quantify the impact of our proposed filtering module on system stability, we conducted a comparative “synthetic” simulation under high-noise conditions.

We utilize a similar physics-based environment (Pinocchio/MeshCat) with the same UR5-Allegro Hand system. To simulate realistic sensor imperfections (from MANO), we injected high-frequency noise into the target trajectory:

- **Trajectory**: A “wide bell” profile designed to test dynamic tracking capabilities.
- **Noise**: A sinusoidal disturbance with a frequency of 50 Hz and an amplitude of 5.5 mm, mimicking sensor jitter.

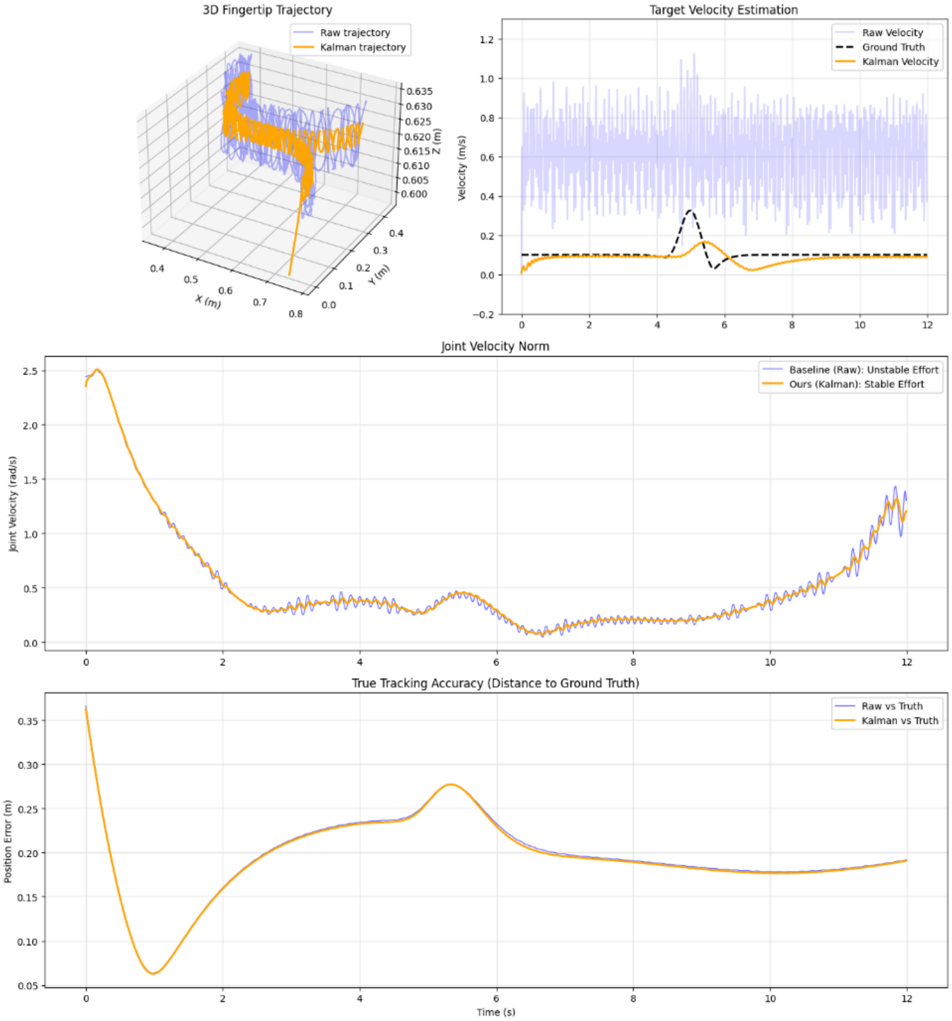


Figure 5: Comparison of control stability: Baseline (Raw input) vs. Kalman Filter.

As illustrated in Figure 5, the comparison reveals a fundamental trade-off between mathematical tracking accuracy and physical realizability:

- **Instability of the Baseline**: The blue curve indicates that the baseline generates significant joint velocity variations. By attempting to track the noise with a low regularization value, the robot achieves a very low positional error relative to the noisy input. However, this accuracy is achieved at the cost of continuous oscillations. While mathematically “accurate” in minimizing the cost function, this behavior is physically unacceptable as it leads to motor overheating.

- **Robustness/Latency trade-off**: The orange curve demonstrates that our Kalman filter approach successfully filters out high-frequency noise, keeping joint velocities smooth during static phases. However, it is important to note that this filtering inherently introduces latency. As seen in the bottom graph, the absolute tracking error of the Kalman approach is not necessarily lower than the Baseline; it may even be marginally higher during fast motions.

Although this approach drastically reduces mechanical effort, its direct implementation might be debated in this specific context. Indeed, the primary goal of the state-based policy is already to transform mathematical trajectories into physically plausible ones. It would, therefore, be interesting to investigate whether this filtering aids the state-based policy by improving the quality of the reference trajectory within the reward function.

4 CONCLUSION

The ViViDex framework is a promising approach to vision-based dexterous manipulation from human videos as it does not require huge amount of labelled data as input. Using an unsupervised perception module may help leveraging unlabelled internet videos to scale and generalize the manipulation skills.

REFERENCES

Birkl, R., Wofk, D., and Müller, M. MiDaS v3.1 – A Model Zoo for Robust Monocular Relative Depth Estimation. *Arxiv Preprint Arxiv:2307.14460*, 2023.

Carion, N., Gustafson, L., Hu, Y.-T., Debnath, S., Hu, R., Suris, D., Ryali, C., Alwala, K. V., Khedr, H., Huang, A., Lei, J., Ma, T., Guo, B., Kalla, A., Marks, M., Greer, J., Wang, M., Sun, P., Rädle, R., … Feichtenhofer, C. *SAM 3: Segment Anything with Concepts*, 2025. https://arxiv.org/abs/2511.16719

Chen, Z., Chen, S., Arlaud, E., Laptev, I., and Schmid, C. *ViViDex: Learning Vision-based Dexterous Manipulation from Human Videos*, 2025. https://arxiv.org/abs/2404.15709

Han, X., Chen, S., Fu, Z., Feng, Z., Fan, L., An, D., Wang, C., Guo, L., Meng, W., Zhang, X., Xu, R., and Xu, S. Multimodal Fusion and Vision-Language Models: A Survey for Robot Vision. *Information Fusion*, *126*, 103652, 2025. https://doi.org/10.1016/j.inffus.2025.103652

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*, 2017. https://arxiv.org/abs/1612.00593

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., and Koltun, V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(3), 2022.

Siméoni, O., Vo, H. V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., … Bojanowski, P. *DINOv3*, 2025. https://arxiv.org/abs/2508.10104

Sripada, V., Carter, S., Guerin, F., and Ghalamzan, A. AP-VLM: Active Perception Enabled by Vision-Language Models. *Arxiv Preprint*, 2024.

Steven, G. J. *The nlopt nonlinear-optimization package*, 2019.

Tang, G., Jia, Q., Huang, Z., Chen, G., Ji, N., and Yao, Z. 3D-Grounded Vision-Language Framework for Robotic Task Planning: Automated Prompt Synthesis and Supervised Reasoning. *Arxiv Preprint*, 2025.

Tavassoli, R., Amani, M., and Akhavian, R. Expanding Frozen Vision-Language Models without Retraining: Towards Improved Robot Perception. *Arxiv Preprint*, 2023.