Contact Imitation Interface: Transferring Dexterous Skills by Showing How Contact Unfolds

Supplementary Material

Anonymous Authors

SUPPLEMENTARY CONTENTS

I. DEMONSTRATION INTERFACE

This section expands the capture hardware (D1), visual tracking (D2), contact-surface retargeting (D3), and context-camera mounting used for observation conversion (D4) in the main paper.

A. Contact-Grounded Capture Hardware

This section describes the hardware design and fabrication of the demonstration interface in D1 of the main paper. The design combines matched fingertip modules, wider-view stereo cameras, and dedicated tracking views while keeping the custom mechanical components straightforward to reproduce.

1) Hardware Design Details: **Matched fingertip modules.** Each human or robot fingertip module uses one HBVCAM-2307-FPC82 V11 USB camera. The camera measures approximately $8.6 \times 8.6 \times 5.4\,\text{mm}$ and has a 120° field of view. It is mounted at the fingertip root and oriented tangentially toward the local contact region. The human and robot variants preserve the same nominal contact-surface geometry and camera placement, with hand-specific 3D-printed mounting structures. A silicone sleeve provides compliant contact without obstructing the camera view. We apply one layer of friction tape on the human contact surface and two on the robot surface; the additional robot-side thickness supports contact compression, as discussed in Sec. II-D.

Context and tracking cameras. Two Orbbec Gemini 305 stereo cameras are mounted at the wrist and near the thumb–index web space. Together they provide the four context-image streams used for reconstruction and viewpoint conversion. One DCXG500 RGB camera tracks the middle-finger fiducial, and an external DCXG500 camera tracks the dorsal fiducial on the back of the hand. The fingertip targets, dorsal target, and tracking-camera mounts are rigidly attached so that their fixed transformations can be calibrated once and reused across recordings.

Fabrication and assembly. We first 3D print the fingertip bodies, camera holders, hand-specific adapters, tracking-camera mounts, and fiducial supports. We then install one contact camera and one silicone sleeve on each fingertip module, attach the AprilCube targets, and mount the modules to the human interface or robot hand. Finally, we install the two Gemini 305 cameras and the auxiliary tracking cameras, calibrate camera intrinsics and the fixed transformations in Fig. S1, and verify all nine streams at the 10 Hz recording rate. The printed geometry fixes the nominal camera–contact-surface placement, which is shared between human demonstration and robot execution.

2) Bill of Materials: Table S1 lists the principal components for one three-finger demonstration interface in USD; the contact-camera price follows the FingerEye BOM. Printed parts, silicone sleeves, friction tape, and printed fiducials have a combined estimated material cost of $7.00. Costs exclude labor, printer depreciation, cables, and the host computer.

Component	Qty.	Cost (USD)
HBVCAM-2307-FPC82 V11 contact camera	3	$28.25 each
Orbbec Gemini 305 stereo camera	2	$259.18 each
DCXG500 RGB tracking camera (middle-finger and external views)	2	$84.42 each
Fabrication materials (printed parts, silicone sleeves, friction tape, and fiducials)	1 set	$7.00 (est.)
Estimated hardware total		$778.95

TABLE S1: Bill of materials for one three-finger demonstration interface. The total covers three contact cameras, two Gemini 305 cameras, two auxiliary tracking cameras, and the combined fabrication-material estimate.

B. Visual Contact-Surface Tracking

This section details the visual estimator and calibrated transformation tree used to extract contact-surface trajectories in D2 of the main paper.

1) Fiducial Targets and Tracking Frames: We use AprilTag markers [1] on AprilCube targets [2] and detect their dense grid keypoints with DeepTag [3]. Each fingertip target is rigidly attached to its contact module. A dorsal fiducial

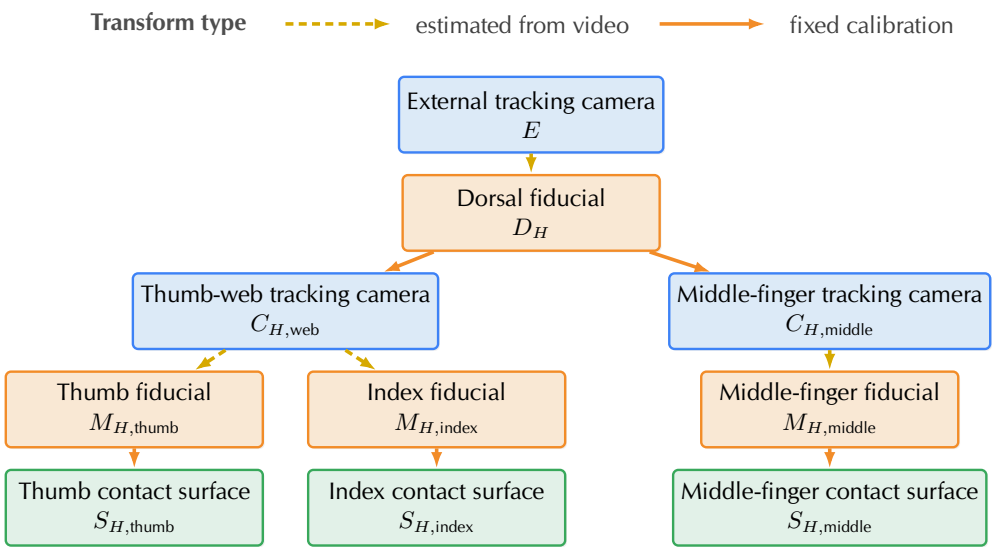

Fig. S1. **Transformation tree for visual contact-surface tracking.** Each edge specifies the child frame's pose in its parent frame. Dashed yellow edges are estimated from videos; solid orange edges are fixed mounting calibrations.

on the back of the hand provides a moving reference that connects the hand-mounted tracking cameras to the external tracking-camera frame.

Let ${}^{A}\mathbf{T}_{t}^{B} \in \mathrm{SE}(3)$ map coordinates from frame B to frame A at time t. The subscript H denotes the human demonstration interface. We use E for the fixed external tracking-camera frame, D_H for the moving dorsal fiducial frame, $M_{H,i}$ for fingertip i's target frame, and $S_{H,i}$ for its contact-surface frame. The index $j(i)$ identifies the hand-mounted tracking camera that observes fingertip i, with frame $C_{H,j(i)}$. The thumb and index targets share a thumb-web tracking view, while the middle finger uses a separate tracking view:

$$j(\text{thumb}) = j(\text{index}) = \text{web},$$
$$j(\text{middle}) = \text{middle}.$$

These tracking views estimate target poses; the contact cameras described in D1 provide the policy's contact observations.

2) Full Contact-Surface Transformation Tree: Figure S1 shows the complete tree for the three-finger capture configuration. Each contact-surface pose in the external camera frame is obtained by composing two tracked poses and two fixed mounting transforms:

$${}^{E}\mathbf{T}_{t}^{S_{H,i}} = {}^{E}\mathbf{T}_{t}^{D_H}\, {}^{D_H}\mathbf{T}^{C_{H,j(i)}}\, {}^{C_{H,j(i)}}\mathbf{T}_{t}^{M_{H,i}}\, {}^{M_{H,i}}\mathbf{T}^{S_{H,i}}. \tag{S1}$$

The external view estimates ${}^{E}\mathbf{T}_{t}^{D_H}$, and the hand-mounted view estimates ${}^{C_{H,j(i)}}\mathbf{T}_{t}^{M_{H,i}}$. The calibrated transforms ${}^{D_H}\mathbf{T}^{C_{H,j(i)}}$ and ${}^{M_{H,i}}\mathbf{T}^{S_{H,i}}$ describe the camera mounts and the contact module's geometry, respectively, and remain fixed during recording. Calibrated camera intrinsics and target keypoint coordinates specified in physical units make the recovered translations metric.

The four local corners of each contact surface are known from its geometry. For corner ℓ, let $\bar{\mathbf{v}}_{i,\ell} = (v_x, v_y, v_z, 1)^{\top}$ be its homogeneous coordinates in $S_{H,i}$. The demonstration target used for contact-surface retargeting in D3 is

$$\mathbf{p}_{i,\ell,t}^{H} = \left[{}^{E}\mathbf{T}_{t}^{S_{H,i}}\, \bar{\mathbf{v}}_{i,\ell}\right]_{1:3}, \tag{S2}$$

where $[\cdot]_{1:3}$ retains the three Cartesian coordinates.

3) Joint Multi-Tag Pose Estimation: DeepTag provides dense image keypoints on each detected tag. We map their canonical grid locations to 3D points using the tag's known dimensions, decoded orientation, and placement on the rigid target. We aggregate these 3D–2D correspondences across visible tags and solve for one target pose with multi-tag Perspective-n-Point (PnP) [4].

For a target frame M observed by camera C, let $\mathbf{x}_k$ be a known 3D keypoint in M and $\mathbf{u}_k$ its detected image location. The pose refinement minimizes the joint reprojection error over accepted keypoints $\mathcal{I}$:

$${}^{C}\mathbf{T}_{t}^{M} = \operatorname*{argmin}_{\mathbf{T}\in\mathrm{SE}(3)} \sum_{k\in\mathcal{I}} \left\| \pi_{\mathbf{K},\mathbf{d}}\left([\mathbf{T}\bar{\mathbf{x}}_k]_{1:3}\right) - \mathbf{u}_k \right\|_2^2, \tag{S3}$$

where $\bar{\mathbf{x}}_k = (\mathbf{x}_k^{\top}, 1)^{\top}$, and $\pi_{\mathbf{K},\mathbf{d}}$ projects a camera-frame point using calibrated intrinsics $\mathbf{K}$ and distortion parameters $\mathbf{d}$.

For observations spanning multiple target faces, the implementation initializes the pose with SQPnP inside RANSAC, rejects inconsistent keypoints and tags, and refines the pose with Levenberg–Marquardt using the retained correspondences. When only one face is visible, we use planar pose estimation with reprojection and face-orientation checks. Joint estimation lets visible tags contribute redundant constraints when other tags are occluded, provided enough valid keypoints remain. The same estimator supplies both tracked transforms in Eq. (S1).

C. Contact-Surface Retargeting

The retargeting implementation optimizes the robot root pose and active finger joints over each recorded sequence. Camera alignment initializes the root pose; it is not an additional objective in the free-wrist variant. Inactive joints

remain fixed. The objective combines contact-corner error with first-order temporal regularization.

1) Coordinates and Objective: The numerical optimization expresses both human targets and robot predictions in the moving dorsal fiducial frame D_H, which is rigidly attached to the back of the hand. The estimated transform ${}^{E}\mathbf{T}_{t}^{D_H}$ maps this moving frame into the fixed external tracking-camera frame E; applying the same transform to both targets and predictions leaves their instantaneous Euclidean contact error unchanged. Let $\widetilde{\mathbf{X}}_t = (\widetilde{\mathbf{R}}_t, \widetilde{\mathbf{x}}_t)$ be the robot root pose in this frame, and let $\mathcal{V}$ contain all valid contact-corner tuples (t, i, ℓ). All retained active corners are weighted equally; inactive or invalid corners are omitted. The contact term is

$$\mathcal{L}_{\text{contact}} = \frac{1}{|\mathcal{V}|} \sum_{(t,i,\ell)\in\mathcal{V}} \left\| \widetilde{\mathbf{p}}^{R}_{i,\ell,t} - \widetilde{\mathbf{p}}^{H}_{i,\ell,t} \right\|_2^2 . \tag{S4}$$

Positions are expressed in meters. For consecutive retained frames, define $\delta\mathbf{q}_t = \mathbf{q}_t - \mathbf{q}_{t-1}$, $\delta\mathbf{x}_t = \widetilde{\mathbf{x}}_t - \widetilde{\mathbf{x}}_{t-1}$, and $\delta\boldsymbol{\omega}_t = \log(\widetilde{\mathbf{R}}_{t-1}^{\top}\widetilde{\mathbf{R}}_t)$, where the rotation logarithm returns a three-dimensional rotation vector. With temporal edge weights e_t and $Z_e = \max(\sum_t e_t, 1)$, the temporal term is

$$\mathcal{R} = \frac{1}{Z_e} \sum_t e_t \Big[\lambda_q \|\delta\mathbf{q}_t\|_2^2 + \lambda_x \|\delta\mathbf{x}_t\|_2^2 + \lambda_R \|\delta\boldsymbol{\omega}_t\|_2^2 \Big]. \tag{S5}$$

Unit conversions and per-dimension normalizations are absorbed into the temporal coefficients. We minimize $\mathcal{L}_{\text{contact}} + \mathcal{R}$ subject to joint limits. Temporal differences of the root pose are measured in the moving dorsal fiducial frame D_H; the final robot trajectory is transformed to the fixed external tracking-camera frame E.

2) Optimizer Configuration: Table S2 summarizes the task-independent configuration. The solver differentiates through robot forward kinematics, while contact-module geometry remains fixed. Separate recordings have no cross-recording temporal edges and are normalized independently.

TABLE S2: Task-independent retargeting optimizer configuration.

Setting	Value
Solver schedule	4,000 projected Adam steps, then up to 1,000 L-BFGS-B steps
Optimization variables	Scaled to $[-1, 1]$
Floating-root bound	± 0.20 m per axis from initialization
Temporal-gap handling	Edge weight $1/g$ for a g-frame interval; no edge across more than 16 missing frames

D. Context-Camera Mount Optimization

For each robot hand, we fit one fixed mounting pose per context camera across the collected demonstrations, holding the retargeted wrist and finger motion fixed. Let $\mathbf{B}_t$ be the robot-palm pose and $\mathbf{H}_{j,t}$ the human pose of camera j, expressed in a common reference frame after handedness conversion. The mount $\mathbf{M}_j \in \mathrm{SE}(3)$ maps the left camera's optical frame into the robot palm frame:

$$\mathbf{M}_j^{*} = \operatorname*{argmin}_{\mathbf{M}_j} \sum_t w_t \, d_j(\mathbf{B}_t\mathbf{M}_j, \mathbf{H}_{j,t}). \tag{S6}$$

Here d_j penalizes displacement between the predicted and demonstrated camera centers and disagreement between their orientations. The weights w_t give equal total weight to each task, divided equally among its recordings and then their valid frames. The cameras are fitted independently; the LEAP wrist camera additionally constrains its optical $+X$ axis to lie in the palm's XY plane. D4 uses these fitted mounts for viewpoint reprojection; physical mounting calibration is performed separately. Contact-camera placement remains fixed by the shared fingertip modules.

E. Processing Efficiency and Data Quality

Table S3 covers videos to policy-ready features: two contact views and nominal/augmented wrist views, using the augmentation in Section II-A.

TABLE S3: Coin-picking processing on an RTX 4090 D (10 demonstrations, 557 frames). Serial stage wall times include initialization and output writing.

Stage	Time (s)
Visual tracking (image2cube)	57
Contact-surface retargeting (cube2qpos)	43
Viewpoint reprojection and robot overlay	72
Image augmentation and RADIO caching	52
Total	224

Table S4 reports retention among stored task demonstrations, not all collection attempts. Coin rejects one recording shorter than 16 steps; bulb rejects two below 50% valid frames. Invalid frames remain masked during learning.

TABLE S4: Data retention for audited LEAP task datasets. Valid-frame coverage is weighted by recording length, not averaged over recordings.

Task	Retained / stored	Retention	Valid frames
Coin	209 / 210	99.5%	99.3%
Letter	205 / 205	100.0%	95.5%
Bulb	115 / 117	98.3%	79.0%

II. Learning Interface

We first detail the contact-surface planner (L2), then the controller (L3), followed by execution, reuse across tasks, policy-camera preprocessing, and computational cost.

A. Contact-Surface Planner

1) Network Architecture and Dimensions: Figure S2 shows the planner's visual conditioning and flow-prediction paths with their dimensions; Table S5 lists normalization and training settings. Here F is the number of controlled fingers, V the number of RGB views, and H the prediction horizon.

The configured view set defines V: **`CTX`** uses the task's context views, while **`CTX+CC`** additionally includes one contact view per controlled finger. The wrist context view

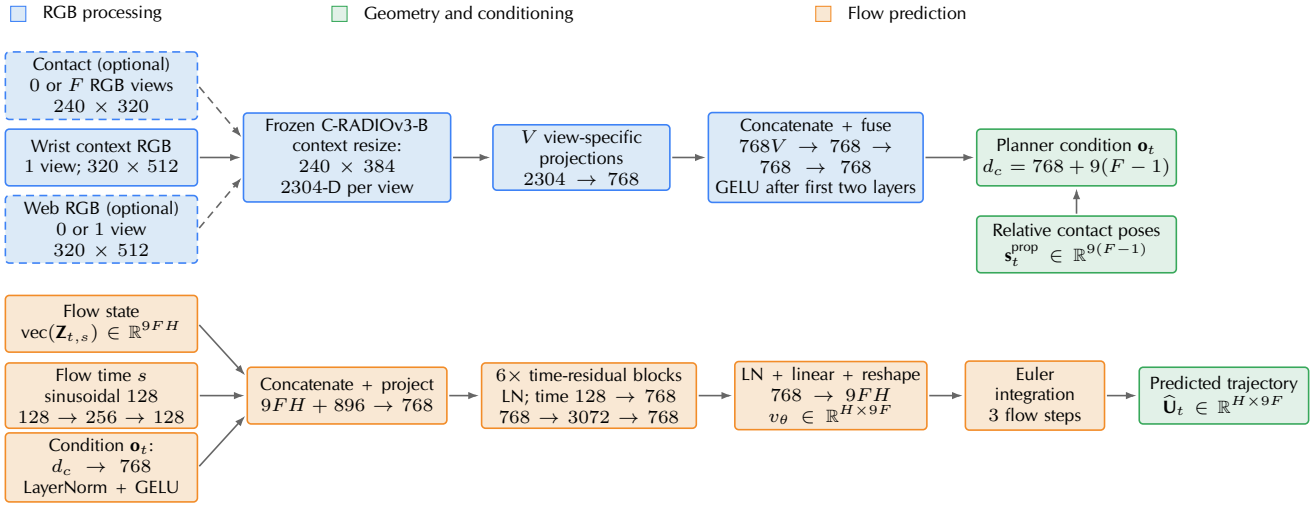

Fig. S2. **Contact-surface planner architecture.** The upper path encodes the V configured RGB views and the current inter-finger geometry into a condition vector. The lower path evaluates the conditional velocity network and integrates it for three Euler steps to predict an H-step contact-surface trajectory. Blue groups RGB processing, green groups geometric inputs, conditioning, and the predicted trajectory, and orange groups flow prediction. Dashed boxes and arrows identify optional camera inputs. Here F is the number of controlled fingers and each contact-surface pose has nine coordinates (three translation and six rotation coordinates).

is used for all reported tasks, whereas the web-space context view is enabled only for bulb installation. Separate policies are trained for each view set. Only the selected views are encoded and concatenated; unused cameras are not inputs to that policy.

2) Flow Matching and Conditioning: Given the combined observation $\mathbf{o}_t = (\mathbf{z}_t^{\text{vis}}, \mathbf{s}_t^{\text{prop}})$, the conditional planner $\pi_\theta(\mathbf{U}_t \mid \mathbf{o}_t)$ is implemented with a velocity field v_θ that transforms a source sample $\boldsymbol{\epsilon} \sim \mathcal{U}([-1,1]^{H\times 9F})$ into a contact-motion chunk. For flow time $s \in [0,1]$, generation follows

$$\begin{aligned} \frac{d\mathbf{Z}_{t,s}}{ds} &= v_\theta(\mathbf{Z}_{t,s}, s \mid \mathbf{o}_t), \\ \mathbf{Z}_{t,0} &= \boldsymbol{\epsilon}, \qquad \widehat{\mathbf{U}}_t = \mathbf{Z}_{t,1}. \end{aligned} \tag{S7}$$

Here $\mathbf{Z}_{t,s}$ is the evolving chunk, and $\widehat{\mathbf{U}}_t = [\hat{\mathbf{u}}_{t,1}, \ldots, \hat{\mathbf{u}}_{t,H}]$ is the predicted relative contact-surface trajectory. We train v_θ with conditional flow matching [5]. For demonstration chunk $\mathbf{U}_t = [\mathbf{u}_{t,1}, \ldots, \mathbf{u}_{t,H}]$, let $\mathbf{U}_{t,s} = (1-s)\boldsymbol{\epsilon} + s\mathbf{U}_t$. Training minimizes

$$\mathcal{L}_{\text{FM}} = \mathbb{E}\left[\|v_\theta(\mathbf{U}_{t,s}, s \mid \mathbf{o}_t) - (\mathbf{U}_t - \boldsymbol{\epsilon})\|_2^2\right]. \tag{S8}$$

The planner's proprioceptive input consists of non-thumb contact-surface poses relative to the thumb. These poses are computed by forward kinematics from retargeted joints during training and measured robot joints at execution. Only valid future steps contribute to the planner's flow-matching loss; padded horizon entries are masked out. The controller is trained separately, and its parameters remain fixed during planner training.

3) Optimization Settings: LEAP and Wuji share the planner settings in Table S5.

4) Visual Augmentation: Before frozen C-RADIOv3-B encoding, we cache one original and eight augmented images per view. Four blend distinct MIL object or table textures [6] over the full image with original/texture weights of 50/50, 60/40, 60/40, and 70/30, then apply photometric augmentation. Four use photometric augmentation alone. Parameters are sampled independently from Table S6. Figure S3 shows the resulting candidates for a coin-picking demonstration. Context images additionally use camera-extrinsic perturbations and rendered-robot lighting and material variations before image-space augmentation.

TABLE S5: Planner normalization and training settings.

Setting	Value
Contact-surface scaling	Translation ×20; six rotation coordinates unchanged
Direct baseline scaling	Global 2nd–98th percentile for non-rotation coordinates; rotation unchanged
Dropout	0
Batch and budget	256; up to 63,000 optimizer updates
AdamW	LR 10^{-4}; $\beta = (0.95, 0.999)$; $\epsilon = 10^{-8}$; weight decay 10^{-6}
LR and EMA schedules	500 warm-up updates then cosine decay; EMA inverse-gamma 1, power 0.75, maximum 0.9999

Parameter	Range
Temperature τ	$[-0.40, 0.40]$
Tint η	$[-0.12, 0.12]$
Exposure (EV)	$[-1, 1]$
Contrast	$[0.60, 1.40]$
Saturation	$[0.60, 1.40]$
Hue	$[-0.04, 0.04]$
Gamma	$[0.75, 1.35]$
Shading strength a	$[-0.30, 0.30]$
Shading direction φ	$[0, 2\pi]$

TABLE S6: Uniform photometric-augmentation ranges; hue is measured in turns. Temperature and tint set RGB gains to $(e^{\tau-\eta/2}, e^{\eta}, e^{-\tau-\eta/2})$. Shading applies gain $1 + a(x\cos\varphi + y\sin\varphi)/\sqrt{2}$, where $x, y \in [-1, 1]$ are normalized image coordinates.

Training samples uniformly from the nine candidates

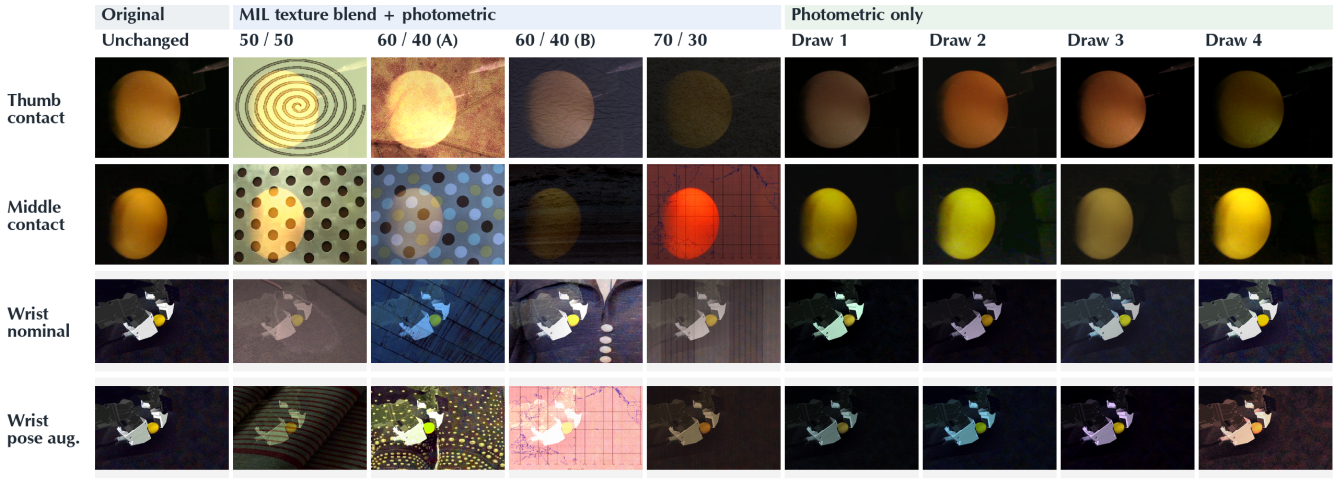

Fig. S3. **Visual augmentation for coin picking.** Rows show the thumb and middle-finger contact views, followed by wrist context views with nominal and perturbed camera extrinsics. The perturbed wrist render also varies robot lighting and materials. Each base image has four whole-image MIL blends with photometric augmentation and four photometric-only variants. Blend columns give original/texture percentages; the two 60/40 candidates use distinct textures.

independently for each view; validation uses originals. The no-visual-augmentation ablation uses only originals while retaining the separate context-rendering selection.

B. Contact-Surface Controller

This section details the controller in L3 of the main paper. Let i index the F controlled fingers, t be the current time, and h be an offset within a trajectory of length H. The robot's current joints are $\mathbf{q}_t$, and $\mathrm{FK}_i(\mathbf{q}_t)$ is contact surface i's pose in the current wrist frame. The predicted $\widehat{\Delta\mathbf{T}}_{i,t,h}$ gives its future pose relative to its current contact-surface frame. Wrist poses are parameterized by translation and the flattened first two rows of the rotation matrix.

1) Geometric Inputs and Command Prediction: The controller C_ϕ in the main paper takes the current joints and the full contact-surface trajectory. It constructs geometric inputs and horizon coordinates internally, then applies the shared network c_ϕ to all predicted steps in parallel. To express the trajectory in the current wrist frame, we compose each relative contact-surface pose with its current forward-kinematics pose. We reconstruct the rotation from its six-dimensional representation and compute each predicted corner as

$$\widehat{\mathbf{p}}_{i,\ell,t,h} = \left[\mathrm{FK}_i(\mathbf{q}_t)\widehat{\Delta\mathbf{T}}_{i,t,h}\bar{\mathbf{v}}_{i,\ell}\right]_{1:3}. \tag{S9}$$

Here $\bar{\mathbf{v}}_{i,\ell}$ is the homogeneous local corner defined in Section I-B. The concatenated corners form $\widehat{\mathbf{P}}_{t,h} \in \mathbb{R}^{12F}$. A deterministic residual MLP predicts wrist and finger commands for each horizon step:

$$(\widehat{\Delta\mathbf{X}}_{t,h}, \widehat{\mathbf{q}}_{t,h}) = c_\phi(\widehat{\mathbf{P}}_{t,h}, \mathbf{q}_t, h/H), \tag{S10}$$

where $\widehat{\Delta\mathbf{X}}_{t,h}$ is the future wrist pose relative to the current wrist, and $\widehat{\mathbf{q}}_{t,h}$ contains absolute joint targets. The controller derives target-frame and thumb-relative-frame features from the corners and combines them with normalized joints and h/H. Each frame feature has 12 entries: three translation coordinates scaled by 10 and all nine rotation-matrix entries, unlike the planner's nine-coordinate pose representation. The output has $9+4F$ entries for the supported four-joint fingers. Joint targets are clipped to their limits, although this does not guarantee that every predicted multi-surface trajectory is kinematically reachable.

2) Augmented Motion Supervision: We augment wrist and joint trajectories using the endpoint perturbations in Table S7. For horizon step h, the smooth ramp $\alpha_h = s_h^2(3-2s_h)$, with $s_h = h/H$, scales translation and joint offsets and the angle of the left-multiplied rotation perturbation. After clipping joints to their limits, forward kinematics supplies consistent contact-corner targets. Following an action-regression warm-up, we minimize

$$\mathcal{L}_{\mathrm{ctrl}} = \mathbb{E}\left[\frac{1}{4F}\sum_{i,\ell}\|100(\mathbf{p}_{i,\ell}^{\mathrm{FK}}(\widehat{\mathbf{a}}) - \mathbf{p}_{i,\ell})\|_2^2\right] + \lambda_a\mathcal{L}_{\mathrm{act}}, \tag{S11}$$

where $\widehat{\mathbf{a}}$ is the predicted wrist–joint action and $\mathbf{p}_{i,\ell}$ is the target contact-surface corner. The factor 100 converts positions from meters to centimeters before squaring the residual. $\mathcal{L}_{\mathrm{act}}$ is normalized wrist–joint regression error, and $\lambda_a = 0.01$. The geometric term directly penalizes realized contact-surface error, while action regression regularizes the kinematic solution.

3) Architecture and Optimization: The residual MLP in Fig. S4 is shared across all horizon steps and evaluates them in parallel. Table S7 lists its motion-augmentation and base optimization settings.

C. Execution and Reuse

The planner predicts $H = 16$ targets at 10 Hz, spanning 1.6 s; execution uses an eight-target chunk (0.8 s at nominal speed). The 10 Hz rate specifies trajectory sampling, not replanning frequency. The controller remains frozen across tasks that share the robot hand, controlled fingers, joint ordering, contact geometry, and trajectory sampling; a different hand or finger set requires its own controller.

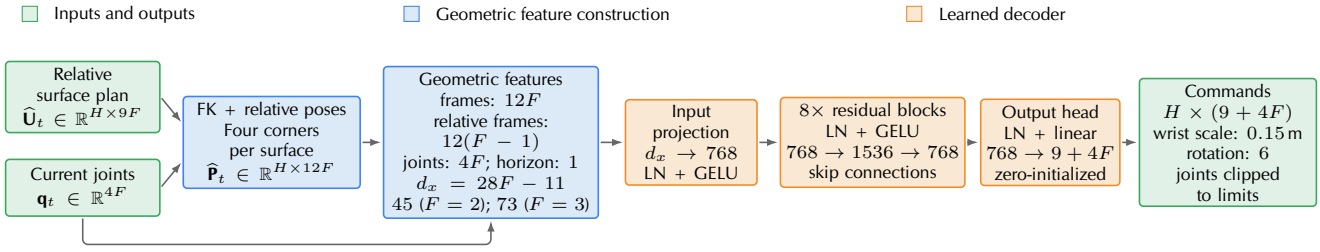

Fig. S4. **Contact-surface controller architecture.** The deterministic geometric front end converts the planner's relative contact-surface trajectory to ordered target corners and constructs absolute and reference-relative frame features. A shared residual MLP then decodes every horizon step in one batched forward pass. Green denotes inputs and outputs, blue denotes deterministic geometric feature construction, and orange denotes the learned decoder. Here F is the number of controlled four-joint fingers, d_x is the per-step feature dimension, and $H = 16$ is the trajectory horizon.

TABLE S7: Controller motion-augmentation and base optimization settings.

Setting	Value
Motion augmentation	
Endpoint translation	Independent uniform offsets in $[-0.02, 0.02]$ m per axis
Endpoint rotation	Uniform angle in $[-0.15, 0.15]$ rad; uniformly sampled axis
Endpoint joint offsets	Gaussian; standard deviation 8% of each joint's range
Application	Probability 0.9 per batch; independent perturbations per trajectory
Base optimization	
Batch sizes	Training 512; validation 128
Warm-up	1,000 action-only updates; LR 10^{-4}
Main objective	Geometric loss $+0.01\mathcal{L}_{\text{act}}$; LR 3×10^{-6}
AdamW	$\beta = (0.9, 0.99)$; weight decay 10^{-6}
Stabilization	Gradient-norm clip 10; EMA decay 0.999
Budget and validation	At most 20,000 updates; validate every 500 updates
Stopping target	0.1 mm worst-task mean corner error

D. Discussion on Next-State Action Targets

Following UMI [7], we use future demonstrated poses as action targets, expressed relative to the current pose. These targets describe motion rather than contact forces, but CIMI can still perform contact-rich manipulation without force sensing. Three complementary mechanisms help explain this behavior.

1) Compliant Contact: With two friction-tape layers on the robot and one on the human interface, the extra robot-side thickness can compress against the object at the same nominal module pose. This tape deformation can produce contact loading without targets inside the object. UMI similarly uses passive fingertip compliance [7].

2) Frictional Support: The friction tape helps resist slip under the normal load supplied by robot actuation. Its role is to support contact, not to measure or regulate force.

3) Relative-Motion Feedback: Replanning anchors predicted motion to the current configuration, allowing visual and proprioceptive feedback to guide renewed corrections toward contact. This is consistent with DexUMI's explanation of relative-action robustness [8].

Future work could add post-contact tactile feedback [8] or augment pose demonstrations with force or wrench measurements for force-informed actions [9, 10].

E. Policy Camera Inputs and Preprocessing

Table S8 specifies the task-dependent camera selections. **`CTX`** includes only the listed context views; **`CTX+CC`** adds the listed contact views while retaining the same context inputs. The dataset may store additional images for reconstruction or visualization; these are not automatically supplied to the policy.

For the two-finger LEAP tasks, the human thumb and middle-finger contact modules map to the robot thumb and index finger; the corresponding camera images follow the same mapping. Bulb uses the human and robot thumb, index, and middle fingers. Each selected stereo unit supplies one RGB policy view, using colors from its rectified left image. Both stereo images are used for depth reconstruction; they are not treated as two separate RGB policy inputs.

The deployment implementation rectifies the live stereo pair, reconstructs a colored point cloud, and rasterizes it into the context image grid with nearest-depth visibility. Stereo reconstruction uses an 800×500 grid, and the policy context raster is 512×320. This preprocessing preserves the rasterization used during training. The policy receives the resulting RGB image, not depth or a point cloud. Human-to-robot viewpoint conversion and robot-geometry overlay are performed on demonstration data offline.

Hand	Task	**`CTX`**	Added contact views
LEAP	Cube	Wrist	Thumb, index
LEAP	Coin	Wrist	Thumb, index
LEAP	Letter	Wrist	Thumb, index
LEAP	Bulb	Wrist, web-space	Thumb, index, middle
Wuji	Cube, coin	Wrist	Thumb, middle

TABLE S8: Policy camera inputs. Finger names identify the controlled robot fingers.

F. Training and Inference Efficiency

With cached features, the LEAP letter planner takes 24.3 min for 63,000 updates on an RTX 4090 D, including initialization, validation, and checkpointing. Offline processing is reported separately in Table S3.

Table S9 uses the same GPU model: batch one, two contact views, one wrist view, three flow steps, and 16 future targets. RGB-to-command timing includes encoding and

feature transfer, but excludes capture, stereo reconstruction, rasterization, and robot communication.

TABLE S9: Inference latency (ms). Both policy paths include the controller; the timings are not additive.

Timed path	Median	95th percentile
Prepared RGB to commands	7.47	7.91
Cached features to commands	2.66	2.80
Controller only (16 targets)	0.84	0.98

REFERENCES

[1] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407.

[2] Y. Park and P. Agrawal, "AprilCube: 3D-printable fiducial targets for reliable 6-DoF pose estimation," Software, 2026. [Online]. Available: https://github.com/younghyopark/aprilcube

[3] Z. Zhang, Y. Hu, G. Yu, and J. Dai, "DeepTag: A general framework for fiducial marker design and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 2931–2944, 2023.

[4] Z. Xu, Y. Li, X. Wu, T. Qiu, and L. Shao, "FingerEye: Learning dexterous manipulation with continuous vision-tactile sensing," *arXiv preprint arXiv:2604.20689*, 2026.

[5] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *International Conference on Learning Representations*, 2023.

[6] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Proceedings of the 1st Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 78, 2017, pp. 357–368.

[7] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Robotics: Science and Systems*, 2024.

[8] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song, "DexUMI: Using human hand as the universal manipulation interface for dexterous manipulation," in *Proceedings of the 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 305. PMLR, 2025, pp. 437–459.

[9] C. Chen, Z. Yu, H. Choi, M. Cutkosky, and J. Bohg, "DexForce: Extracting force-informed actions from kinesthetic demonstrations for dexterous manipulation," *IEEE Robotics and Automation Letters*, vol. 10, no. 6, pp. 6416–6423, 2025.

[10] D. Zhang, C. Yuan, C. Wen, H. Zhang, J. Zhao, and Y. Gao, "KineDex: Learning tactile-informed visuomotor policies via kinesthetic teaching for dexterous manipulation," in *Proceedings of the 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 305. PMLR, 2025, pp. 4123–4138.