

## A Appendix: Behavior Cloning Policy Transfer for Marble Rolling

To further evaluate the effectiveness of our cross-sensor tactile generation approach, we conducted an additional experiment on a marble rolling task. In this setup, a behavior cloning policy is trained to roll a marble—initially placed at a random position—toward the center of the GelSlim sensor image. The policy is trained using supervised learning on GelSlim tactile data and learns to guide the marble using the tactile feedback provided by the sensor.

We then test the transferability of this policy to a different sensor, DIGIT, by using our T2D2 model to generate the corresponding GelSlim tactile signal from the DIGIT input. This allows the original policy, trained exclusively on GelSlim data, to be applied to DIGIT inputs **without any retraining or modification**.

The results show effective transfer: the policy succeeds in **15 out of 20 trials** when using DIGIT inputs translated to GelSlim, compared to **20 out of 20** using real GelSlim signals. This experiment illustrates that our cross-sensor generation framework can support not only perception tasks but also simple sensor-conditioned control, enabling policy reuse across different tactile hardware.

## B Appendix - Implementation Details

### B.1 Diffusion Model for Tactile Cross-Modal Sensing

Our implementation of the diffusion model closely follows Stable Diffusion [32], with the difference that we use a ResNet-50 to generate the GelSlim encoding from GelSlim images for conditioning.

The model is optimized for 30 epochs by Adam [46] optimizer with a base learning rate of  $10^{-5}$ . The learning rate is scaled by  $\text{gpu number} \times \text{batch size}$ . We train the model with batch size of 48 on 4 NVIDIA A40 GPUs.

At inference time, the model conducts 200 steps of the denoising process with a 2.54 guidance scale.

### B.2 VQ-VAE

We use a VQ-VAE architecture similar to the one proposed by Van den Oord et al [47] for the style transfer. Before training VQ-VAE, we processed the sensor images by obtaining the difference between the deformed image at the moment of contact with the object and the undeformed image. In addition, we resize these images from the sensor images' original size to 128x128 and keep their corresponding numbers of channels. The input to our model is a 128x128 subtracted Gelslim RGB image, and the output is the corresponding 128x128 subtracted depth map Soft Bubbles image. The input image  $x$  is passed through a CNN encoder to generate a vector in the latent space  $z$ . This latent vector is then quantized via a collection of discrete vectors known as the *codebook*, such that  $z_e(x)$  is transformed into  $z_q(x)$ . This quantized latent vector is passed through a CNN decoder to generate the final image  $\tilde{x}$ . The encoder parameters, quantization codebook vectors, and decoder parameters are all learned such that mean squared-error in the latent space quantizations and output reconstructions are minimized.

## C Appendix - Depth Adaptation Stage Details

For the adaptation stage, we use the depth map  $D'_S$  and its corresponding binary mask  $M'_S$  from the source sensor tactile image  $I_S$  obtained with our Depth Estimation Model to find the equivalent depth map  $D''_T$  and mask  $M''_T$  as if the target sensor was in contact.

First, we define the set of valid pixel coordinates, using the contact mask  $M'_S$ , as follows:

$$\Omega = \{(u, v) \mid M'_S(u, v) = 1\} \quad (3)$$

For each valid pixel  $(u, v) \in \Omega$ , we back-project the depth value into 3D space using the inverse intrinsic matrix  $K_S^{-1}$  and subsequently transform the resulting 3D point from the source sensor

frame to the target sensor frame using the rigid transformation  $T_{S \rightarrow T}$ . Here,  $T_{S \rightarrow T}$  is defined as the composition of the transformation from the source sensor to the alignment frame,  $T_{S \rightarrow A}$ , and from the alignment frame to the target sensor,  $T_{A \rightarrow T}$ ; that is,  $T_{S \rightarrow T} = T_{A \rightarrow T} \circ T_{S \rightarrow A}$ . The alignment frame is defined as a common reference frame shared across different tactile sensors, representing the same touch event. This standardization allows for consistent interpretation of the contact geometry, regardless of each sensor's unique geometry or pose. This process is expressed as follows.

$$\mathcal{P}_T = \left\{ T_{S \rightarrow T} \left( D'_S(u, v) \cdot K_S^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \right) : (u, v) \in \Omega \right\}. \quad (4)$$

Where,  $\mathcal{P}_T$  represents the resulting point cloud in the target sensor frame. Then we find the target sensor depth map and its mask, defined as:

$$D''_T(u, v) = \begin{cases} Z, & \text{if there exists } p = (X, Y, Z)^\top \in \mathcal{P}_T \text{ such that } \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{Z} K_T p \\ & \text{and } (u, v) \text{ is within the image bounds,} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

$$M''_T(u, v) = \begin{cases} 1, & \text{if there exists } p \in \mathcal{P}_T \text{ such that } p \text{ projects to } (u, v) \\ & \text{and } \text{SDF}(p, \mathcal{M}_{\text{target}}) \leq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

## D Appendix - Sensor Alignment Details

Figure 6 shows each sensor's main coordinate frames: GelSlim camera frames (gR, gL), Soft Bubbles camera frames (BR, BL), and grasp frame (G). For our Touch2Touch dataset, we align both

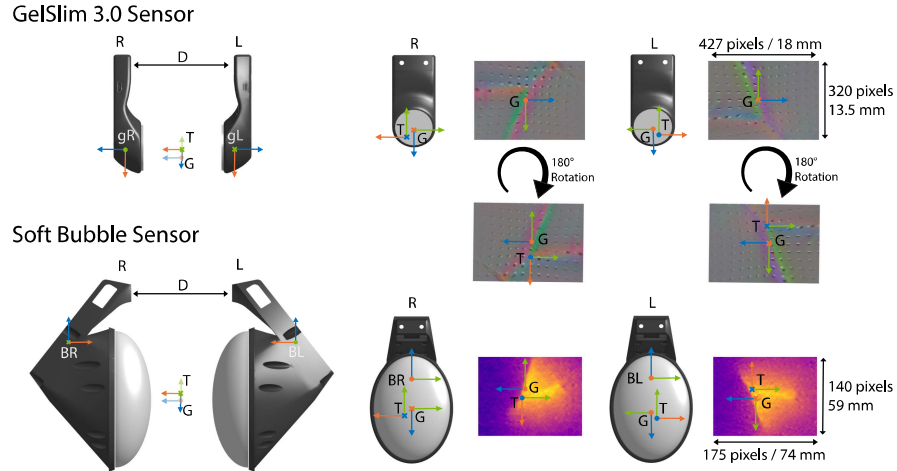


Figure 6: **GelSlim and Soft Bubble Alignment.** This figure shows each sensor's main coordinate frames: GelSlim camera frames (gR, gL), Soft Bubbles camera frames (BR, BL), grasp frames (G), and tool frames (T). D corresponds to the distance we keep between the same type of sensors during data collection. We can see the grasp frames projection in the image plane of each sensor and notice the need to rotate for alignment. For each coordinate frame, the x-axis is shown in red, the y-axis is shown in green and the z-axis is shown in blue.

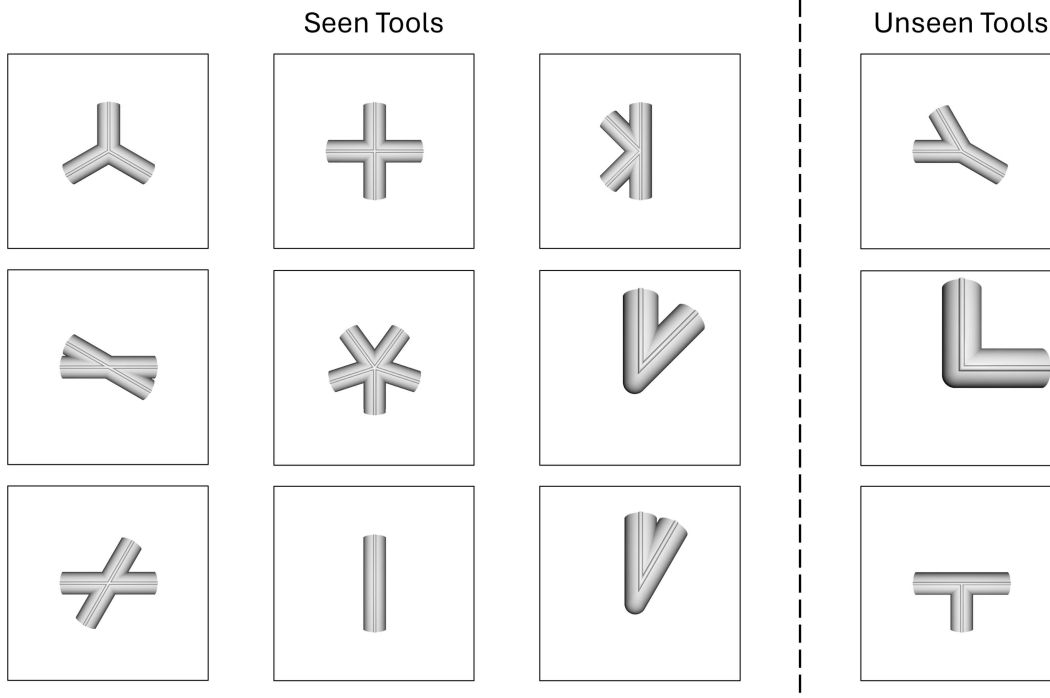


Figure 7: **Dataset Tools.** The left side of the image shows the geometries of the tools that were seen during training of the generative models. The right side of the image shows the geometries of tools that were not seen during training of the generative model.

sensors by locating the grasp frame of each at the same section of the manipulated object to obtain paired tactile signatures. In addition, we keep both sensors at a distance  $D$ , shown in Figure 6. The final step to align the tactile signatures is to rotate one sensor image by  $180^\circ$ . This rotation is necessary for the grasp frames to be aligned. We can see the grasp frames projection in the image plane of each sensor in Figure 6 and notice the need to rotate for alignment. This figure also shows the difference in size between the sensors' images in pixel space and millimeters.

## E Appendix - Data Collection Tools

Figure 7 shows the geometries of the tools that were seen and not during the training of the generative models.