Supplementary Information

Training Tactile Sensors to Learn Force Sensing from Each Other

Zhuo Chen[1*], Ni Ou[1], Xuyang Zhang[1], Zhiyuan Wu[1], Yongqiang Zhao[1], Yupeng Wang[1], Nathan Lepora[2], Lorenzo Jamone[3], Jiankang Deng[4*], Shan Luo[1*]

[1]King's College London, London, United Kingdom.
[2]University of Bristol, Bristol, United Kingdom.
[3]University College London, London, United Kingdom.
[4]Imperial College London, London, United Kingdom.

*Corresponding authors. Email: shan.luo@kcl.ac.uk,
zhuo.7.chen@kcl.ac.uk , j.deng16@imperial.ac.uk

This PDF file includes:

Supplementary Text 1 to 9

Supplementary Figures 1 to 6

Supplementary Table 1 to 2

Supplementary Caption 1 to 8 for Supplementary Video 1 to 8

Supplementary Video 1 to 8

Text 1. Marker-to-marker translation model

The M2M model consists of two main components: a marker encoder-decoder and an image-conditioned diffusion model. As shown in Fig. **2**C, the marker encoder transforms deformed images $I_t^{S_i}$ from sensor i and the reference image $I_0^{T_j}$ from sensor j into latent vectors $z_t^{S_i}$ and $z_0^{T_j}$ respectively, while the marker decoder converts the output latent vector $z_t^{G_i}$ from the diffusion model to the generated deformed images $I_t^{G_i}$. The image-conditioned diffusion model fuses latent vector $z_t^{S_i}$ with the conditional input $z_0^{T_j}$ through cross-attention mechanisms[58] and denoises the fused feature map to produce latent vectors $z_t^{G_i}$. This end-to-end architecture enables direct translation of marker-based images from $I_t^{S_i}$ to $I_t^{G_i}$ with the image style of $I_t^{T_j}$ while preserving the deformation from $I_t^{S_i}$.The training objective combines two primary components to train the model in a pixel-to-pixel manner[59], i.e. an adversarial loss $\mathcal{L}_{gan}$ [47], and a reconstruction loss $\mathcal{L}_{rec}$ incorporating L2 and LPIPS [60] loss.

Adversarial Loss The adversarial loss aims to align the distribution of generated tactile images $p(I^G)$ with the target images $p(I^T)$. The discriminator D_T learns to differentiate between generated images I^G and real target images I^T. The adversarial loss is formulated as:

$$\mathcal{L}_{gan} = \mathbb{E}_{I^T \sim p(I^T)}[\log D_T(I^T)] + \mathbb{E}_{I^S \sim p(I^S)}[\log(1 - D_T(G(I^S, I_0^T))] \tag{1}$$

where G minimizes this objective while D_T maximizes it: $\min_G \max_{D_T} \mathcal{L}_{gan}$.

Reconstruction Loss The reconstruction loss $\mathcal{L}_{rec}$ ensures both pixel-level and perceptual-level similarity between generated images I^{G_i} and target images I^{T_j} through L2 and LPIPS metrics, capturing subtle marker displacement during translation:

$$\mathcal{L}_{rec} = \sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_{\mathrm{L2}} \mathbb{E}_{I^{S_i} \sim p(I^{S_i})} \left\| I^{T_j}, G(I^{S_i}, I_0^{T_j}) \right\|_2 + \lambda_{\mathrm{Lpips}} \mathbb{E}_{I^{S_i} \sim p(I^{S_i})} \left\| I^{T_j}, G(I^{S_i}, I_0^{T_j}) \right\|_{PIPS} \tag{2}$$

Where λ_{L2} is the weight for L2 loss, λ_{Lpips} is the weight for LPIPS loss.

Overall Objective The complete learning objective for the generative model combines the above losses with weights λ_{gan} and λ_{rec}:

$$\arg\min \lambda_{gan}\mathcal{L}_{gan} + \lambda_{rec}\mathcal{L}_{rec} \tag{3}$$

Marker encoder-decoder. As shown in Supplementary Figure 1, we adapt the variational autoencoder (VAE) architecture from SD-Turbo[50]. The VAE processes marker images with a size of 256×256 and employs an encoder-decoder structure: an encoder that compresses marker patterns into a latent space, and a decoder that reconstructs marker patterns from these latent representations. To optimize the model's performance while maintaining parameter efficiency, we implement Low-Rank Adaptation (LoRA)[53] for efficient fine-tuning. The LoRA is with rank-4 adaptation on key network components, including convolutional layers and attention modules. The training objective combines reconstruction loss (L1 and L2) with a KL divergence loss to balance accurate pattern reconstruction with latent space regularization. This architecture enables effective compression of marker patterns into a structured latent space while preserving essential geometric and spatial relationships between different marker types.

Image-conditioned diffusion model. The conditional diffusion model is based on the UNet[51] architecture from SD-Turbo (Supplementary Figure 1) combined with a DDPM Scheduler[52]. We implement a one-step diffusion process[59] for efficient marker pattern translation. The UNet model is also augmented with LoRA adaptation (rank-8) applied to key network components, including attention layers, convolutional layers, and projection layers. We split the reconstruction loss $\mathcal{L}_{rec}$ into $\mathcal{L}_{\text{Lpips}}$ and $\mathcal{L}_{\text{L2}}$. The model was optimized using a multi-component loss function:

$$\mathcal{L} = \lambda_{\text{gan}}\mathcal{L}_{\text{gan}} + \lambda_{\text{Lpips}}\mathcal{L}_{\text{Lpips}} + \lambda_{\text{L2}}\mathcal{L}_{\text{L2}} \tag{4}$$

where $\lambda_{\text{gan}} = 0.5$, $\lambda_{\text{Lpips}} = 5.0$, and $\lambda_{\text{L2}} = 1.0$ to balance the contributions of adversarial, LPIPS, and L2 loss respectively. We employed a CLIP-based vision-aided discriminator [61] with multilevel sigmoid loss for the adversarial component, and a VGG-based LPIPS network [60] for perceptual loss computation.

Pretraining for the marker encoder-decoder. The marker encoder-decoder is first trained on the simulation dataset for marker feature extraction. All raw marker images are 640×480 pixels with packed bits file in .npy format. We employ an 80-20 train-test split. All images are preprocessed to a uniform size of 256×256 pixels and normalized to [0,1] range. The model is trained using AdamW optimizer with a learning rate of 1×10^{-4}, betas=(0.9, 0.999), and weight decay of 1×10^{-2}. We employ mixed-precision training (FP16) with a batch size of 4. The loss function combined a reconstruction loss (L1 + L2) and KL divergence with weights of 1.0 and 1×10^{-6} respectively. Training proceeded for 100,000 steps. The training process for the marker encoder-decoder is demonstrated in Supplementary Figure 3.

Pretraining for M2M model with simulation data. We load the pretrained marker encoder-decoder for the M2M model. For the encoder for the image condition, we freeze the weights to ensure the extracted features are fixed. The training process utilizes all of the 132 combinations from the simulation dataset with 80-20 train-test split. Each training sample in one batch consists of a triplet: a deformed marker image $I_t^{S_i}$ from sensor *i*, its corresponding paired marker image $I_t^{T_j}$ from sensor *j*, and a reference marker image $I_0^{T_j}$ from sensor *j*. The model is trained using AdamW optimizer with an initial learning rate of 5×10^{-6} with 500 warm up steps, betas=(0.9, 0.999), epsilon=1×10^{-8}, and weight decay of 1×10^{-2}. Training proceeded with a batch size of 4. The training process is shown in Supplementary Figure 4.

Training for M2M model with real-world data. For the homogeneous translation, we first split the homogeneous location-paired image data into two groups with seen indenters and unseen indenters. We finetune the simulation pretrained model using the seen group with an 80-20 train-test split with the same hyperparameters as above for the simulation data. The training process for the homogeneous translation is shown in Supplementary Figure 5.

The training for the material effect data uses the same process and hyperparameters but involves loading the model trained with homogeneous data as the pretrained model.

The training for the heterogeneous data loads the model weights trained with homogeneous data. The hyperparameters are the same as the homogeneous training except we change the batch size to 16 for speeding up training. The training process for the heterogeneous translation is shown in Supplementary Figure 6.

Notably, as manual annotation of markers is costly, we employ the original efficient-SAM model for marker extraction without fine-tuning, resulting in a few low-quality marker images in our dataset. Since marker image quality directly impacts both generated marker images and force prediction accuracy, using a dedicated marker segmentation model could further improve performance.

Inference Process. For model inference, we utilize the mean vector, without variance, of the latent distribution from the marker encoder to ensure deterministic outputs. For datasets in homogeneous translation, material effect and heterogeneous translation, each one is preprocessed using consistent image transformations, including resizing and normalization. The model processes images in batches of 8, generating images with a size of 256×256 that are subsequently upscaled to the target resolution (640×480) using Lanczos interpolation. The upscaled outputs are then thresholded to binary marker images. The results are saved as compressed binary Numpy arrays.

Text 2. Spatiotemporal force prediction model

Model architecture. The model consists of four main components demonstrated in Supplementary Figure 2: a marker feature encoder backbone, a spatiotemporal module with convolutional GRU (ConvGRU)[54], a post-processing network with ResNet Unit, and a regression head with multilayer perceptron (MLP). The input to our model is a sequence of tactile images with shape S×N×3×256×256, where S is the sequence length, N is the batch size, and each image has 3 channels with 256×256 spatial resolution. The marker feature encoder processes these images through three convolutional blocks, each incorporating instance normalization and dropout. The first block reduces spatial dimensions to 128×128 while increasing channels to 64, the second block further reduces to 64×64 with 96 channels, and the third block outputs features at 32×32 resolution with 128 channels. These spatial features are then processed by a ConvGRU module that maintains the 32×32 spatial resolution while capturing temporal dependencies across the sequence. With a hidden state dimension of 128 channels, the ConvGRU tracks temporal patterns while preserving spatial information. The temporal features undergo spatial dimension reduction through two residual blocks (stride 2), expanding the channel dimension from 128 to 256, then to 512, while reducing spatial dimensions to 16×16 and 8×8 respectively. An adaptive average pooling layer collapses the remaining spatial dimensions to 1×1, producing a 512-dimensional feature vector per timestep. The regression head maps these features to three-axis force predictions using a fully connected layer followed by sigmoid activation. This architecture effectively combines spatial and temporal processing to capture both the detailed marker deformations in individual frames and their evolution over time, enabling accurate prediction of three-axis force from tactile image sequences.

The network is optimized using a mean absolute error (MAE) loss function:

$$\mathcal{L}_{\text{MAE}} = \frac{1}{N}\sum_{i=1}^{N} || \hat{F}_i - F_i ||_1 \tag{5}$$

where $\hat{F}_i$ and F_i denote the predicted and ground-truth forces respectively

Model Training. The image data undergoes preprocessing including resizing to 256×256 pixels and normalization using ImageNet statistics (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]). Force measurements are normalized using pre-computed global minimum and maximum values to ensure consistent scaling across different samples. Our dataloader implements dynamic sequence sampling, where for each batch, we randomly sample sequence lengths between the first frame and the maximum available length with at least two frames, enabling the model to learn from varying temporal contexts. For model initialization, we employ normalization for convolutional layers and constant initialization for normalization layers. The training process follows a two-stage approach: first, we pre-train the model on a single randomly selected sensor with a learning rate of 0.1 for 40 epochs, then fine-tune on the complete dataset with a learning rate of 1×10^{-3} for another 40 epochs. We use SGD optimization with momentum (0.9) and weight decay 5×10^{-4}, along with a learning rate scheduler. During training, we utilize a custom collate function that handles varying sequence lengths through dynamic padding, where shorter sequences are padded to match the batch's sampled sequence length by repeating the last frame. The model is trained with a batch size of 4 using L1 loss between predicted and ground truth forces, exclusively on the seen group data, with early stopping based on validation performance.

Model Inference. During inference, our model processes tactile image sequences to predict three-axis force. The inference pipeline utilizes a modified data loading scheme where, unlike training, we process the complete sequence length without random sampling. The dataloader maintains the same image preprocessing pipeline (resizing to 256×256 and normalization with ImageNet statistics). For both source and target domain evaluation, we load full sequences with a batch size of 1 to ensure consistent temporal processing across all samples. The predictions undergo denormalization using globally tracked minimum and maximum force values the same as in training to restore the actual force scale. We evaluate the model's performance using multiple metrics: Mean Absolute Error (MAE) for individual force components $(F_{\mathrm{x}}, F_{\mathrm{y}}, F_{\mathrm{z}})$, MAE for total force magnitude F_{t}, and R^2 values to assess prediction accuracy over the whole force range. Notably, while our unsupervised method has shown impressive performance, a gap remains compared to supervised learning approaches. Enhancing accuracy may involve compensating for additional material properties such as Poisson's ratio, roughness, and viscosity. Alternatively, few-shot finetuning using force labels from simple gauges, weighted objects, or calibrated tactile sensors could help close this gap.

Text 3. Trajectory for marker deformation simulation

The trajectory shown Extended Data Fig. 1B covers a grid of contact locations with horizontal steps Δx and Δy of 4 mm and vertical increments Δz of 0.3 mm, reaching a maximum indentation depth $z_{\max}$ of 1.5 mm. This approach yields 45 target contact locations (5 steps in depth $\times 9$ grid) per indenter, resulting in 810 unique deformed meshes in total. For each movement to target location, the indenter is initialized at a position where its bottom surface is parallel to and 10 mm above the elastomer surface. To ensure we are obtaining smooth mesh, we set the world step time to 1×10^{-4} s and the contact speed to -10 mm/s.

Text 4. Fabrication of soft skins

The fabrication process is demonstrated in Extended Data Fig. 2A. First, we mix XPA-565 silicone base (B) with activator (A) using different ratios to control the softness. For homogeneous translation and heterogeneous translation, we use a ratio of 15:1. In material compensation, we employ seven different ratios ranging from 6:1 to 18:1, where higher ratios produce softer elastomers. We pour the mixture into a mold for 4 mm thickness for 24-hour natural curing to obtain transparent silicone elastomer. Next, we print designed markers (see Fig. **3**A) on sticker paper using an inkjet printer and transfer them onto the cured elastomer. We then prepare a coating mixture by combining aluminum powder and silver bullet powder with solvent in a 1:1:2.5 ratio, then mix this with silicone elastomer (15:1 ratio) to pour onto the elastomers with markers. The pigment mixture ensures opaqueness while maintaining negligible increase in the elastomers' thickness. After another 24 hours of curing, we cut the elastomer to 20 mm × 20 mm dimensions for testing. Notably, increasing the XPA-565 ratio extends the required curing time.

Text 5. Parameters for data collection in real world

For homogeneous and material compensation tests, we implement the following parameters to the parameters defined in Extended Data Fig. 2B: horizontal moving distances $\Delta x = 3$ mm, $\Delta y = 4$ mm, depth step $\Delta z = 0.3$ mm with maximum depth $z_{\max} = 1.2$ mm, moving angle $\theta = 30^{\circ}$, and shear distance $\Delta r = 1$ mm. This configuration yields $5 \times 4 \times 12 = 240$ target points with varying moving directions and locations. The heterogeneous tranlsation employs a moving angle $\theta = 45^{\circ}$ with depth parameters of $\Delta z = 0.25$ mm and $z_{\max} = 1$ mm for GelSight and uSkin. The parameters for TacPalm are configured with $\Delta x = \Delta y = 6.5$ mm, $\Delta z = 1.125$ mm, $z_{\max} = 4.5$ mm, $\theta = 30^{\circ}$ and $\Delta r = 1.5$ mm. This configuration yields $5 \times 4 \times 8 = 160$ target points. This configuration enables image collection at 0.25 mm intervals for GelSight and uSkin to pair with TacPalm collected at 1.125 mm intervals, ensuring comparable force ranges collected from TacPalm.

Text 6. Parameters for marker conversion for uSkin

Through grid search for the parameters shown in Extended Data Fig. 8A, we determine the optimal visualization parameters: $D_{\min} = 300$, $D_{\max} = 6000$, $\Delta X_{\max} = \Delta Y_{\max} = 0.6$, $S_D = 0.2$, $S_x = S_y = 0.002$. These parameters provide an optimal balance between sensitivity to subtle deformations and clear visualization of larger forces while preventing marker overlap or grid distortion.

Text 7. Relationship of force and indentation depth

According to contact mechanics, when a flat rigid indenter applies force F on an elastic specimen's surface[62], the relationship between force F and penetration depth d_{z} is given by:

$$F = \alpha E d_{\mathrm{z}} \tag{6}$$

where α is a geometric constant specific to the indenter, and E represents the elastic modulus of the specimen. Based on Equation (6), we can compare the elastomers' softness among different sensors by measuring the relationship between applied force F and indentation depth d_{z} using a flat rigid indenter.

Text 8. Parameters for data collection in force-depth curve

For heterogeneous translation, we applied maximum depths d_{max} of 1mm for GelSight and uSkin, while extending to 4.5mm for TacPalm due to its extremely soft property. For comparing the softness among heterogeneous sensors, we normalize their indentation depths to the range of 0 to 1 (see Extended Data Fig. 8B). The $F - d_z$ curves are drawn by using the mean and variance values during three indentations.

Text 9. Material compensation process

As shown in Fig. **5**E-i, the pipeline for material compensation is included in training the force prediction model by correcting the force label F^S to F^{SC}. When loading the force-image pair data, the contact depth d_z is used to index force f_z^S and f_z^T from the material priors of the source sensor S and the target sensor T respectively. The compensation ratio r can then be calculated by:

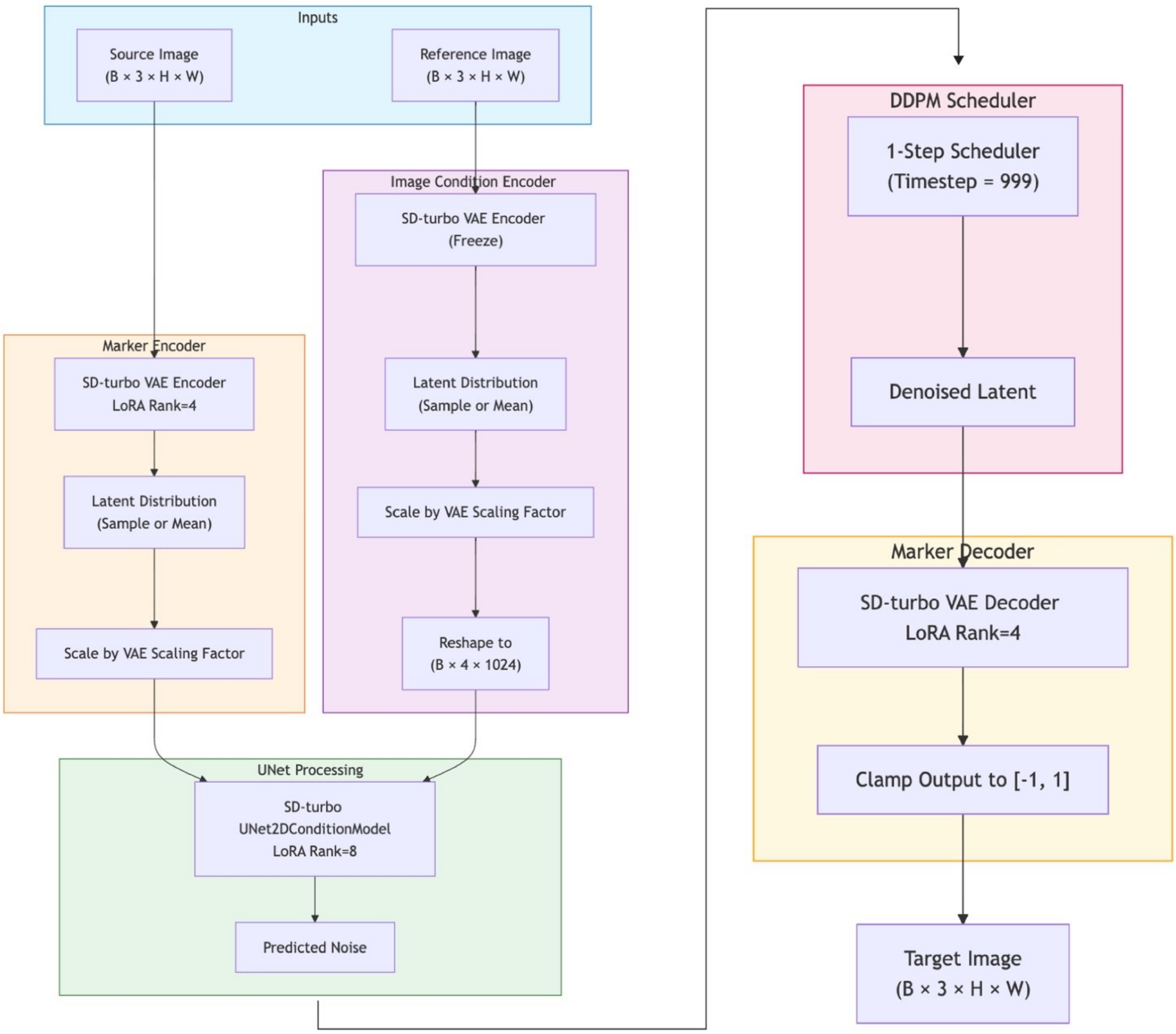
$$r = \frac{f_z^T}{f_z^S} - 1 \tag{7}$$

Then, the force label F^S can be corrected with either the ratio of r_{L} or r_{U} depending on the contact is in loading phase L or unloading phase U. Notably, we introduce two additional hyperparameters: starting depth d_0 ($0 < d_0 < d_{max}$) and correction weight λ ($0 < \lambda < 1$). d_0 limits the compensation where the contact depth d_z exceeds its value. λ controls the compensation magnitude. These parameters used in our paper are obtained via grid search (see Supplementary Table 1 and Supplementary Table 2). Thus, the corrected force label F^{SC} can be derived as:

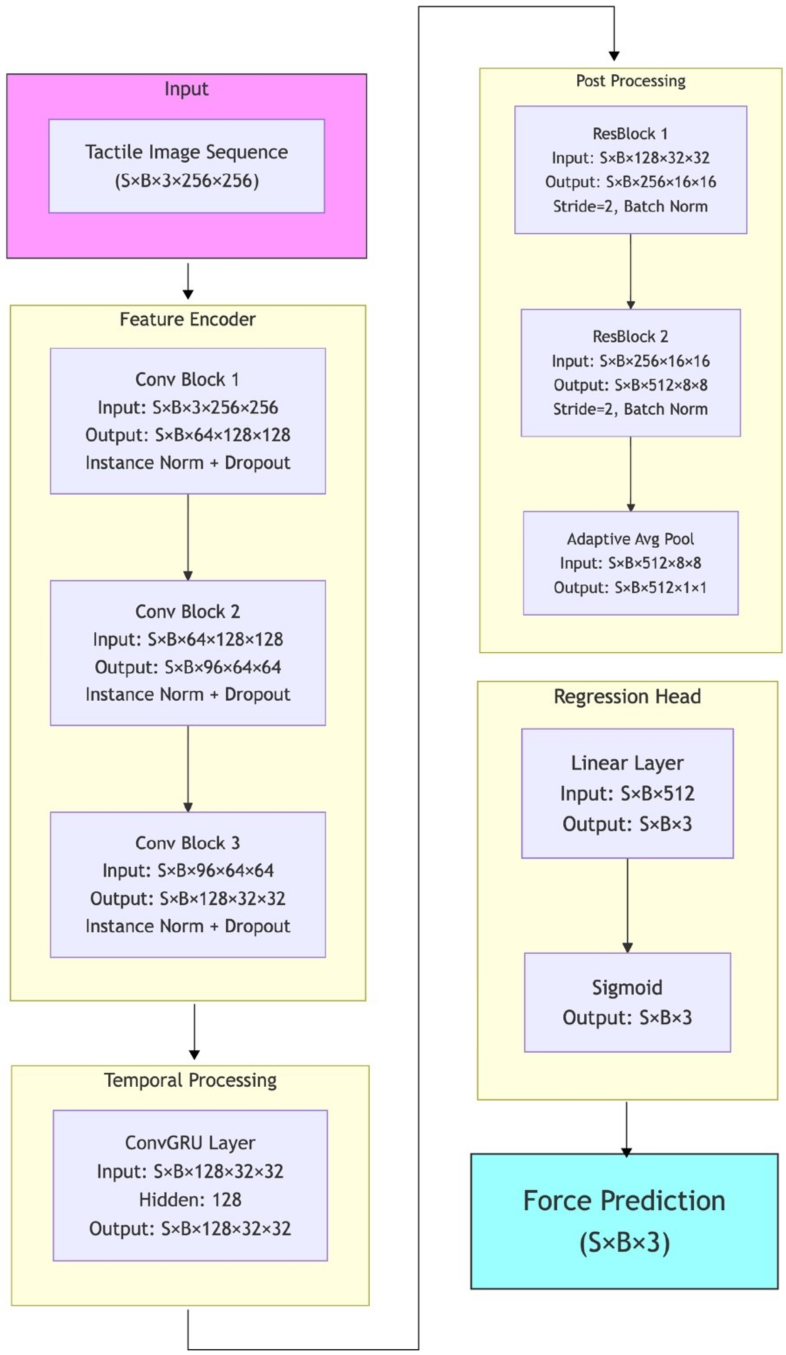
$$F^{SC} = F^{S} \cdot (1 + \lambda r) \tag{8}$$

where r is r_{L} or r_{U} indexed with the contact location d_z:

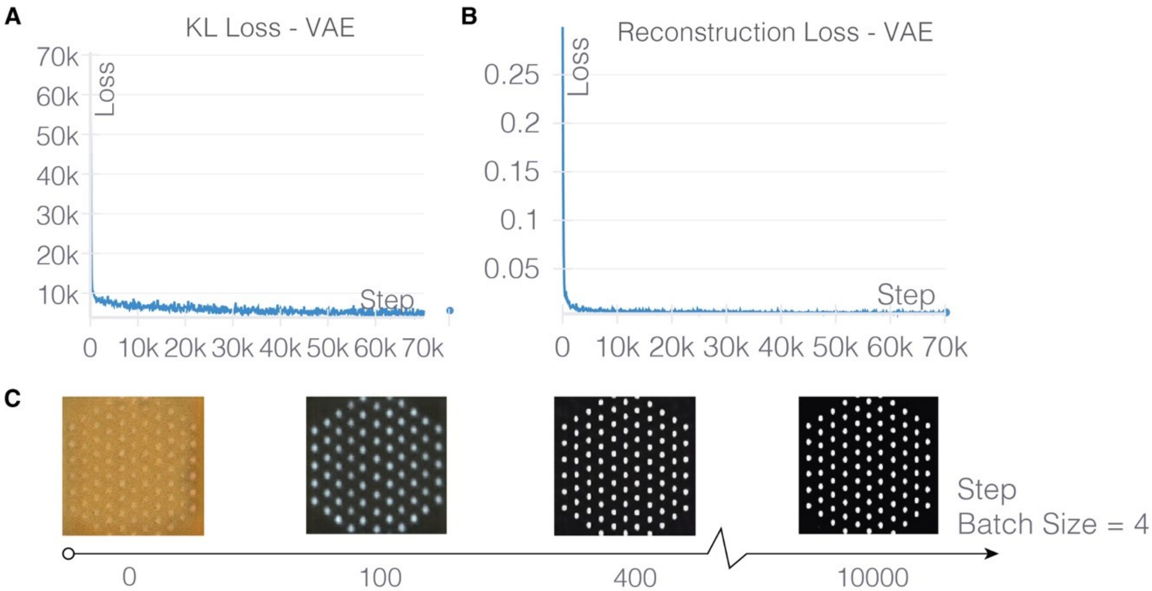
$$r = \begin{cases} r_{\mathrm{L}}, & \text{if } d_z > d_0 \text{ and } d_z \in L \\ 0, & \text{if } d_z \le d_0 \\ r_{\mathrm{U}}, & \text{if } d_z > d_0 \text{ and } d_z \in U \end{cases} \tag{9}$$



Supplementary Figure 1. Maker-to-marker translation model architecture.

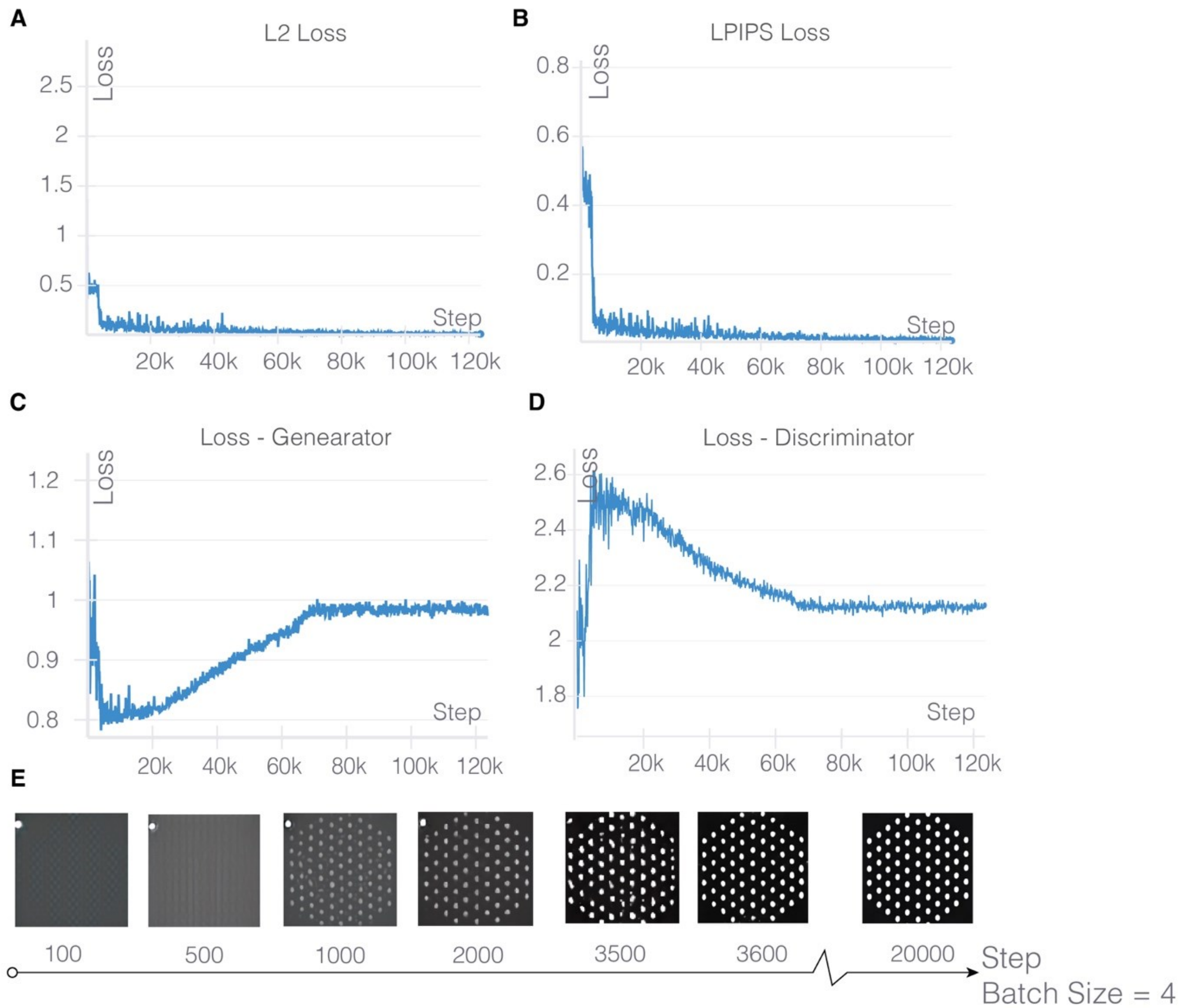


Supplementary Figure 2. Spatiotemporal force prediction model architecture.

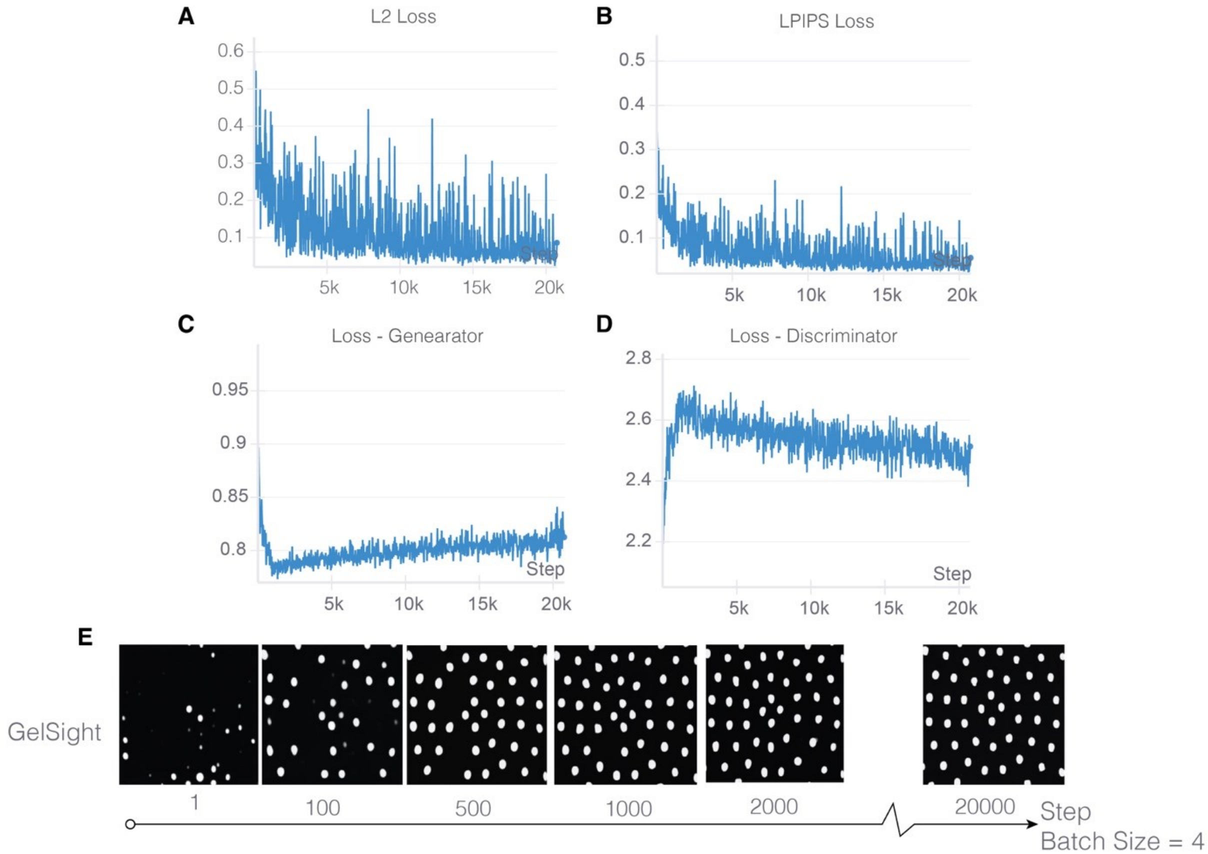


236

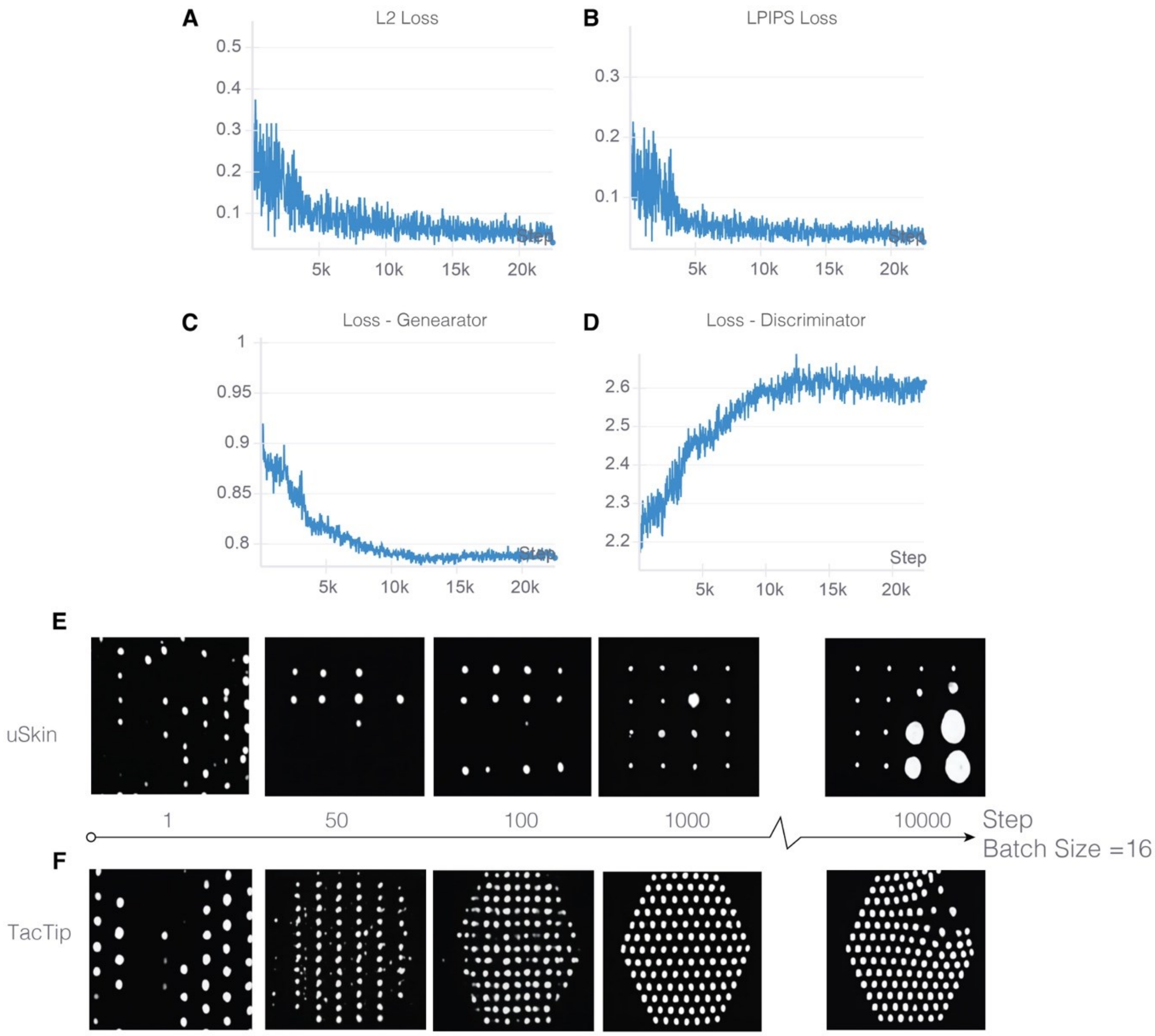
237 **Supplementary Figure 3. Training process for the marker encoder-decoder. (A)** KL Loss. **(B)**
238 Reconstruction Loss. **(C)** The development process of decoded images.



Supplementary Figure 4. Training Process for M2M model with simulated data. (A) L2 loss. **(B)** LPIPS loss. **(C)** Generator loss. **(D)** Discriminator loss. **(E)** The development process of generated images with simulated data.



Supplementary Figure 5. Training Process for M2M model with homogeneous sensors. (A) L2 loss. **(B)** LPIPS loss. **(C)** Generator loss. **(D)** Discriminator loss. **(E)** The development process of generated images from homogeneous GelSight sensors.



Supplementary Figure 6. Training Process for M2M model with heterogeneous sensors. (A) L2 loss. **(B)** LPIPS loss. **(C)** Generator loss. **(D)** Discriminator loss. **(E)** The development process of generated images from uSkin. **(F)** The development process of generated images from TacPalm.

Supplementary Table 1. Hyperparameters used in material compensation in study of *material softness effect*

Source \ Target	r6	r8	r10	r12	r14	r16	r18
r6		0/0.5	0.4/1	0/0.5	0/0.75	0/0.75	0/0.75
r8	0/1		0.8/1	0/0.75	0/0.75	0/0.75	0/0.75
r10	0/0.5	0/0.75		0.4/0.25	0/0.5	0/0.5	0/0.5
r12	0.8/0.75	0.8/0.25	0.8/0.25		0/0.75	0/0.75	0/0.5
r14	0.8/0.5	0/1	0.8/0.25	0/1		0/1	0/0.75
r16	0.8/0.5	0/1	0/1	0/1	0/1		0/0.5
r18	0.4/0.25	0/1	0/1	0.8/0.5	0.8/0.25	0.4/0.25	

*Demonstrate starting depth d_0 (mm) and correction weights λ as d_0/λ in each cell)

*Grid search in range of [0,1] with a step of 0.4 for d_0 and 0.25 for λ

Supplementary Table 2. Hyperparameters used in material compensation in study of *heterogeneous translation*

Source \ Target	uSkin	GelSight	TacPalm
uSkin		0.5/1	0/0.5
GelSight	0/1		0/0.75
TacPalm	0.75/0.5	0/0.5	

*Demonstrate starting depth d_0 (mm) and correction weights λ as d_0/λ in each cell)

*Grid search in range of [0,1] with a step of 0.25 for d_0 and 0.25 for λ

Supplementary Caption for Video 1. Marker-to-marker translation with simulated data. The examples showcase the marker-to-marker translation results with sequential image translations when *A1*, *C2*, and *D3* are used as the source domains, respectively. The generated images preserve similar deformations to the source domains while adopting the image styles of the target domains.

Supplementary Caption for Video 2. Marker-to-marker translation in homogeneous translation. The examples showcase the marker-to-marker translation results with sequential image translations when *A-I*, *C-I*, *D-I*, *A-II*, and *C-II* are used as the source domains, respectively. The generated images exhibit similar deformations to the source domains while adopting the image styles of the target domains. We observed a few failure cases involving a flickering effect when transferring from *A-II* and *C-II* to *A-I*. This issue is caused by the shift of the elastomer in *A-I* during data collection, leading to continuous changes in the reference marker patterns. These continuous changes result in inconsistency between image conditions and reference images for *A-I*, producing a small number of generated images with noise.

Supplementary Caption for Video 3. Marker-to-marker translation in heterogeneous translation. The examples showcase the marker-to-marker translation results with sequential image translations when uSkin, TacPalm, and GelSight are used as the source domains, respectively. The generated images exhibit similar deformations to the source domains while adopting the image styles of the target domains.

Supplementary Caption for Video 4. Real-time force prediction for homogeneous translation. The examples showcase the force prediction performance before (source-only) and after applying the GenForce model when transferring from *A-I*, *A-II*, *C-I*, and *D-I* to *C-II*. Prior to using the GenForce model, significant force prediction errors are observed across all four combinations. After implementing the GenForce model, the force prediction accuracy is greatly improved, resulting in significantly reduced errors.

Supplementary Caption for Video 5. Material compensation performance. The examples showcase the force prediction performance before and after applying material compensation on the GenForce model when transferring from sensor with hard skin to sensor with soft skin (*r6_r16*), and from sensor with soft skin to sensor with hard skin (*r16_r6*). Noticeable error reduction is observed, particularly in the normal force, after applying material compensation.

Supplementary Caption for Video 6. Real-time force prediction for heterogeneous translation to uSkin. The examples showcase the force prediction performance before (source-only) and after applying the GenForce model when transferring from GelSight and TacPalm to uSkin. Significant force prediction errors are observed in both combinations prior to using the GenForce model. After applying the GenForce model, force prediction accuracy is significantly improved, with greatly reduced errors across the entire tested force range. Notably, lower force errors are observed in the lower force range.

Supplementary Caption for Video 7. Real-time force prediction for heterogeneous translation to TacPalm. The examples showcase the force prediction performance before (source-only) and after applying the GenForce model when transferring from uSkin and GelSight to TacPalm. Significant force prediction errors are observed in both combinations prior to using the GenForce model. After applying the GenForce model, force prediction accuracy is significantly improved, with greatly reduced errors across the entire tested force range. Notably, lower force errors are observed in the lower force range.

Supplementary Caption for Video 8. Real-time force prediction for heterogeneous translation to GelSight. The examples showcase the force prediction performance before (source-only) and after applying the GenForce model when transferring from uSkin and TacPalm to GelSight. Significant force prediction errors are observed in both combinations prior to using the GenForce model. After applying the GenForce model, force prediction accuracy is significantly improved, with greatly reduced errors across the entire tested force range. Notably, lower force errors are observed in the lower force range.