applied sciences

MDPI

Article

A Capacitive Tactile Sensor Digital Twin for Real-Time Synthetic Data Generation and Sim-to-Real Transfer in NVIDIA Isaac Sim

Berith Atemoztli De la Cruz Sánchez * and Jean-Philippe Roberge

Command and Robotics Laboratory, École de Technologie Supérieure, Montreal, QC H3C 1K3, Canada; jean-philippe.roberge@etsmtl.ca

* Correspondence: berith.de-la-cruz-sanchez.1@ens.etsmtl.ca

Featured Application

The proposed framework can be applied to robotic manipulation tasks requiring tactile feedback, including grasping, object recognition, and closed-loop policy learning using synthetic tactile data generated directly within NVIDIA Isaac Sim.

Abstract

With advances in robotic manipulation in recent years, tactile sensing has become increasingly important in scenarios where visual information is unreliable or insufficient. However, the development of learning-based tactile algorithms and policies is limited by the cost and time required to collect large-scale physical datasets. While robotic simulation offers an alternative for synthetic data generation, current simulation platforms, such as NVIDIA Isaac Sim, lack integrated capacitive tactile sensors. This paper presents a finite element method (FEM)-based digital twin of a capacitive tactile sensor and an extension for NVIDIA Isaac Sim that enables the real-time generation of synthetic tactile data directly within the simulation environment. The proposed system extracts nodal deformations from the Isaac Sim PhysX engine and uses a convolutional neural network (CNN) to predict synthetic tactile maps that replicate the response of the physical sensor. We further demonstrate adaptability by retraining the model from an initial sensor to a second capacitive sensor, the Robotiq TSF-85, operating under the same sensing principle. The complete generation pipeline executes in 8.04 ms per frame on the laptop configuration and 6.71 ms on the workstation, enabling real-time operation at 60 Hz on both, and at 120 Hz on the workstation. The similarity of the generated tactile data for the Robotiq TSF-85 is evaluated using complementary similarity metrics, achieving a mean Structural Similarity Index Measure (SSIM) of 0.727 ± 0.16 and a mean Pearson correlation of 0.87 ± 0.16 against real measurements. To demonstrate the utility of the proposed framework, a shape-recognition task (cylinder, sphere, cube) was performed using only synthetic tactile data and evaluated on real-world sensor data, achieving an accuracy of 69.3% with zero real training labels. By enabling integrated tactile simulation and synthetic data generation within Isaac Sim, this work provides a practical tool for tactile perception using capacitive tactile sensors.

Keywords: tactile sensor; digital twin; tactile simulation; sim-to-real transfer; synthetic data generation; convolutional neural network

Check for updates

Academic Editors: Jong-Hwan Kim, Joon-Hyuk Park and Namsuk Cho

Received: 1 July 2026
Revised: 28 July 2026
Accepted: 30 July 2026
Published: 3 August 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Robotic manipulation has experienced significant advances in recent years, driven by AI, collaborative robots (cobots), and the need for automation. This area is experiencing

an unprecedented boom, stimulating development across a wide variety of tasks and applications. However, a common thread among these tasks is the reliance on vision systems. In situations where it is not possible to fully rely on vision, or where vision is partially obstructed, the use of tactile feedback has proven to be highly beneficial to systems and even increase their performance, as shown by Melnik et al. [1]. Although the adoption of tactile sensors has become popular in recent years, this technology has not yet reached a level of maturity comparable to other technologies, such as vision systems; therefore, its industrial-scale implementation remains underdeveloped. In more dynamic and less controlled environments, learning-based methods are often used to handle variability and uncertainty. However, their practical deployment can be hindered by data-related requirements, including the collection and labeling of sufficiently diverse tactile datasets, as well as challenges in generalization across objects, materials, and contact conditions.

Collecting data from real-world physical systems is expensive, time-consuming, and challenging to scale. This challenge has led to the development of simulation environments that enable controlled and reproducible generation of large volumes of tactile information. Synthetic generation of tactile data is therefore becoming important for developing contact-rich scenarios and supporting their transfer to real-world applications. Robotic simulation platforms capable of simulating tactile sensors have become increasingly important. Notably, NVIDIA Isaac Sim (NVIDIA, Santa Clara, CA, USA) [2] is widely used for robotic simulation, yet it lacks a native pipeline to simulate tactile sensors based on direct deformation-to-signal transduction, for instance, capacitive sensors where contact-induced deformation modulates electrode spacing and thus capacitance, as opposed to vision-based tactile sensors that infer deformation from camera images.

To address these limitations, this work introduces a new Isaac Sim extension, natively integrated into the simulation environment, that provides a finite element method (FEM)-based digital twin of a capacitive tactile sensor for synthetic tactile data generation. The extension identifies the sensor's primitive within the scene, extracts node-level surface deformations from the PhysX engine, and synthesizes tactile maps that closely match the output of the physical sensor in real time. The generated data are recorded synchronously with the simulation and saved for subsequent post-processing. Building on our prior elastic and hyperelastic modelling efforts in Abaqus and Isaac Gym presented in [3,4], we re-engineered and consolidated the full pipeline into a single, Isaac Sim-native implementation. This transition results in a unified tactile-sensing simulation framework, tightly integrated with the scene graph and physics engine, and extended with end-to-end capabilities for sensor identification, deformation extraction, and real-time tactile map synthesis. To assess the generality of the approach, we further adapt the digital twin, through retraining, from the initial sensor to a second capacitive sensor of the same operating principle—the Robotiq TSF-85—showing how the pipeline can be retrained for a new capacitive sensor. Figure 1 shows a grasping example with the tactile maps generated in simulation and from the physical sensor.

The main contributions of this work are summarized as follows:

- An FEM-based digital twin of a capacitive tactile sensor, integrated into NVIDIA Isaac Sim, that synthesizes tactile maps from simulated contact deformation.
- An extension for Isaac Sim that performs automatic sensor identification, node-level deformation extraction, and real-time tactile map generation, with a measured per-step cost compatible with real-time simulation.
- A demonstration of the pipeline's adaptability across sensors, retraining the digital twin from an initial capacitive sensor to the Robotiq TSF-85.
- Validation of synthetic-to-real fidelity using complementary similarity metrics, together with a sim-to-real shape-recognition case study trained exclusively on synthetic data.

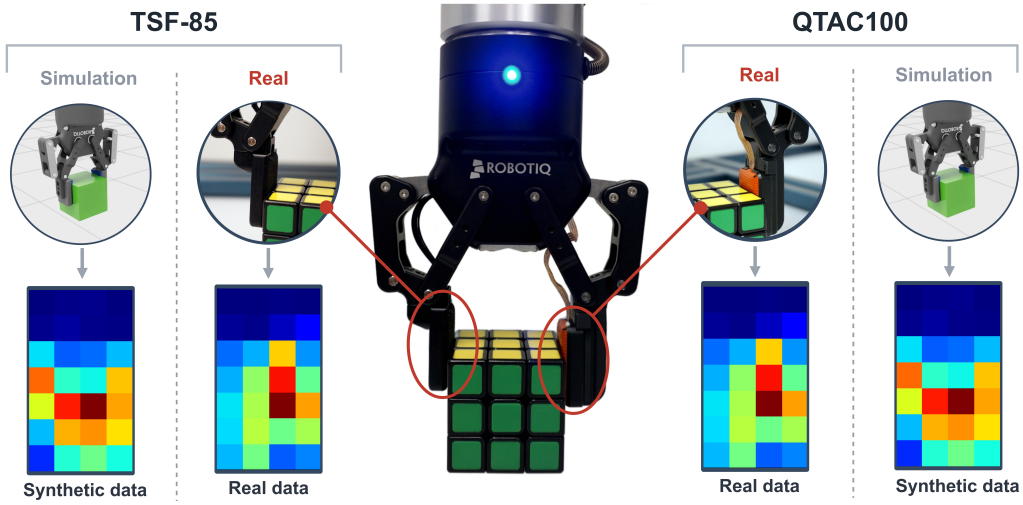


Figure 1. Example grasping results for the two capacitive tactile sensors in a real-world scenario. The Robotiq TSF-85 sensor is shown on the left, and the QTAC100 sensor is shown on the right. For each sensor, the simulated gripper-mounted asset is shown alongside the corresponding synthetic and real tactile maps produced by the extension.

2. Literature Review

In recent years, tactile perception has emerged as a key enabler for robotic systems performing advanced manipulation tasks [5]. This is because tactile sensors, unlike vision systems, can directly measure interaction cues during contact, including contact forces, local deformation, incipient slip, and surface texture. For example, Hu et al. [6] report a framework for the dexterous manipulation of long, thin, cylindrical objects using reinforcement learning (RL) for a robotic hand with three piezoresistive tactile sensors. For this type of task, tactile feedback helps the system maintain control of the object and react to disturbances, even in the absence of vision, since the camera can be occluded by the hand itself. Similarly, Gu et al. [7] demonstrate the limitations of purely visual perception systems using TactileALOHA. This bimanual system uses GelSight tactile sensors in conjunction with vision to perform tasks involving textures, tasks that a vision-only system would fail to complete.

Although numerous studies have demonstrated the potential of tactile sensing technology across disciplines, its integration into real-world applications remains limited due to two primary challenges. First, there is a scarcity of large-scale datasets [8], which are essential for robust training of learning algorithms. Second, the hyperelastic materials used in sensor fabrication exhibit nonlinear behaviours, making their mathematical modelling and subsequent simulation complex.

Despite these obstacles, simulation methodologies have advanced to provide various approaches for modelling tactile sensors and generating synthetic data. For example, Hu et al. [6] model their sensor without accounting for material nonlinearity. In contrast, Zhang et al. with TacFlex [9] and Du et al. with TacIPC [10] utilize FEM to characterize the deformable components of tactile sensors. The adoption of FEM has increased because it enables accurate modelling of nonlinear contact mechanics and the propagation of internal stresses in hyperelastic materials [11].

However, implementing these models in robotic simulation environments is constrained by a division between two main categories of simulators: conventional robotic simulators and those equipped with advanced physics engines for finite element analysis. The first category, exemplified by Gazebo [12], provides extensive robotic assets and integration capabilities but is limited to rigid-body approximations, which preclude accurate modelling of deformable contacts. In contrast, high-fidelity physics engines such as Abaqus

and ANSYS can simulate soft object behaviour with precision but lack essential tools for robotic integration.

Although tools like MuJoCo, PyBullet, and NVIDIA Isaac Sim have begun to converge within these two groups of simulators, they still fall short of a faithful representation of soft bodies, as their modelling capabilities are limited to elastic models. Within this group, NVIDIA Isaac Sim has gained traction as a robotics simulation environment due to its RTX-based rendering and GPU-accelerated PhysX simulation. It also provides a ROS and ROS 2 bridge for integration with common robotics software stacks, and supports FEM-based deformable-body dynamics through PhysX. These capabilities have led to the growth of its ecosystem, with the expansion of robot models, sensors, and modules, consolidating its use in Digital Twin and Sim-to-Real applications. For example, Huang et al. [13] demonstrate in SyncTwin that a simulated robot is synchronized in real time with a real one to predict the stability of robotic grippers using only vision.

While solutions such as SyncTwin demonstrate the potential of Isaac Sim, its perceptual capabilities remain confined to the visual modality. The integration of tactile sensors within this ecosystem remains limited in the current literature. To date, the most comprehensive frameworks are TacEx [14], which focuses on simulating the GelSight Mini sensor, and UniVTAC [15], a framework designed to generate synthetic data for manipulation tasks.

Furthermore, the recent transition of the Isaac Gym ecosystem, a platform that gained traction for its efficiency in reinforcement learning, toward integration with NVIDIA's Omniverse has helped solidify Isaac Sim as one of the core platforms for robotic simulation. This rapid consolidation of robotics simulation toolchains around Isaac Sim places the tactile-sensing models proposed by Duret et al. [16], Liu et al. [17], Huang et al. [18], Akinola et al. [19], and De la Cruz Sánchez et al. [3] in front of a sustainability trilemma: fully re-engineer the models to comply with the Isaac Sim API and data-generation pipeline, port them to alternative simulation ecosystems, or accept progressive technical obsolescence as interfaces, dependencies, and tooling evolve. In practice, this transition is not merely a software refactor, but often requires revisiting sensor abstractions, contact modelling assumptions, and synthetic data interfaces to preserve fidelity and enable reproducible sim-to-real workflows.

At the same time, most existing simulation-driven tactile sensing proposals remain centred on vision-based tactile sensors, where deformation is inferred from rendered images, leaving comparatively limited support for direct transduction modalities such as capacitive tactile sensing. This is a notable gap, as capacitive tactile sensors are commonly adopted in robotics due to their compact and lightweight form factor.

A limited body of work has considered capacitive sensing in simulation. CapSense [20] proposes a real-time electrostatic FEM framework for capacitive proximity sensing, generating raw capacitive signals and validating the simulated response against physical sensors in human–robot interaction scenarios. Silva-Plata et al. [21] combine a mutual-capacitance sensor with an online FEM simulation implemented in SOFA, where the capacitive transduction is not simulated directly; instead, the mechanical model supports the interpretation of acquired sensor measurements and the localization of human contact on a deformable soft-robot surface. More recently, TactSpace [22] explores capacitive sim-to-real transfer by aligning real capacitance measurements with simulated penetration-depth data from Isaac Lab and stress representations from Abaqus within a shared latent space, achieving zero-shot transfer of task networks trained purely on simulated data. This enables downstream models trained on simulated representations to transfer directly to real sensor measurements, while avoiding reproducing the capacitive signal in simulation. These works reflect growing interest in capacitive sensing within simulation, but they approach it through electrostatic modelling, model-based signal interpretation, or representation alignment.

The direct generation of capacitive tactile data from simulated deformation, performed natively and in real time within the simulator, remains comparatively underexplored.

In this work, we address this gap by introducing a capacitive tactile sensor digital twin and a native Isaac Sim extension that generates synthetic tactile data from direct contact deformation. Building on the model originally proposed in Isaac Gym by De la Cruz Sánchez et al. [3], we re-engineer the full pipeline into an Isaac Sim-native implementation and demonstrate its adaptability by retraining it for a second capacitive sensor. The following sections detail the sensor modelling, the extension architecture, and the experimental validation.

3. Materials and Methods

A digital twin of the sensor was implemented within NVIDIA Isaac Sim to enable synthetic tactile data generation. The simulated environment replicates the CoRo laboratory workstation and includes a UR5e manipulator (Universal Robots, Odense, Denmark) and a Robotiq 2F-85 gripper (Robotiq, Lévis, QC, Canada). Within the USD-based scene graph in Isaac Sim, each object or entity is represented as a scene primitive, which corresponds to a node in the hierarchical scene structure. In this work, the sensor, robot, and gripper are independent primitives.

As shown in Figure 2, the robotics pipeline leverages our Isaac Sim extension as one of its core simulation components, enabling synchronous tactile data generation along the simulation timeline. The process begins with a one-time initialization stage, during which the extension establishes access to the sensor primitives and registers the deformable mesh corresponding to the sensor's dielectric layer within the simulated scene. During runtime execution, the extension continuously extracts FEM vertices from the deforming mesh at each physics step. These deformation values are processed to generate a deformation map, which serves as input to a convolutional neural network (CNN). The CNN outputs a 28-dimensional synthetic tactile vector that replicates the capacitive response of the physical 28-taxel array, which can be used for external processing or directly within the simulation environment.

This extension offers dual functionality. First, it enables the generation of synthetic datasets for offline training. Second, because it operates natively within the simulator, the resulting tactile output can be integrated directly into the robot's observation space as a live, high-frequency feedback signal during simulation. The pipeline described in this section was first developed and validated on an initial capacitive sensor. To evaluate its adaptability, we subsequently applied it to a second capacitive sensor with the same operating principle, the Robotiq TSF-85, whose characteristics are detailed in Section 3.1.

3.1. Capacitive Tactile Sensor

This study uses two capacitive tactile sensors that share the same sensing principle: a compressible dielectric is positioned between two electrodes and deforms under applied pressure, changing the local capacitance and thereby encoding the contact-induced deformation as a 28-element vector of taxel readings expressed in tactile counts (TCs). The first sensor, the QTAC100 [23], was used for the initial development and validation of the pipeline. Its hardware design was previously described in [24]; it is currently available commercially at [23]. It operates within a 0–50 N range, measures 28 × 44 mm and has an active tactile area of 22 × 37 mm, and incorporates an inertial measurement unit (IMU) and a dedicated taxel for dynamic sensing. Its internal structure comprises four primary layers: a printed circuit board (PCB) with 28 taxels for static sensing, a polyurethane dielectric layer, a conductive fabric layer acting as the second capacitor plate, and a neoprene encapsulation layer. The second sensor is the Robotiq TSF-85 [25], a commercial tactile fingertip released

in 2026 for the Robotiq 2F-85 gripper. It provides static pressure sensing through 28 taxels arranged in a 7 × 4 grid, dynamic sensing at 1000 Hz, and proprioceptive feedback through an internal three-axis IMU, in a 28.4 × 44 mm footprint. Although the sensor is rated for forces up to 225 N, all experiments and evaluations in this work were conducted within a 0–50 N range, matching the operating range of the first sensor. Both sensors transduce contact through capacitive deformation and produce a 28-taxel output, allowing the same modelling and inference pipeline to be reused for the TSF-85 after retraining, without altering the network's output structure.

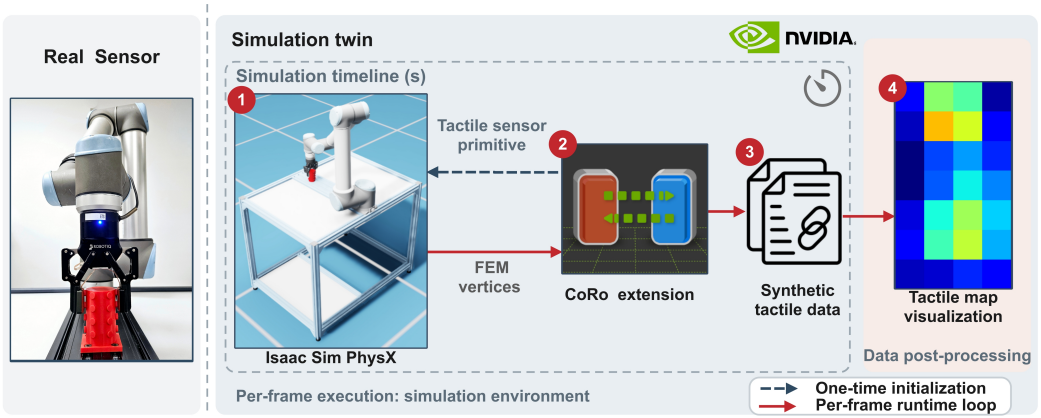


Figure 2. Overview of the proposed pipeline for synthetic tactile map generation within Isaac Sim. The physical sensor (**left**) is replicated by a simulation twin (**right**). Numbers follow the per-frame execution order: PhysX advances the FEM state of the deformable sensor pad (1), the CoRo extension converts the extracted top-layer deformation into a 28-taxel tactile vector via a CNN (2, 3), and the result is visualized or logged (4).

3.2. Sensor Implementation in Isaac Sim

The simulation is implemented in NVIDIA Isaac Sim using the PhysX SDK 107.3.3 (omni.physx extension 107.3.26) physics engine, which provides rigid-body dynamics, contact modelling, collision handling, and deformable object simulation. Soft bodies are simulated using a volumetric FEM with tetrahedral elements, assuming linear elastic behaviour. The multilayer sensor structure (dielectric layer, conductive fabric, and neoprene cover) is modelled as a single equivalent deformable layer. This simplification was required because current PhysX-based deformable simulations do not accurately reproduce the transmission of deformation between superimposed soft-body layers, leading to numerical instabilities. The accuracy of this single-layer approximation was previously characterized in [3], where deformation maps from the simplified elastic model were compared against a full multi-layer hyperelastic FEM model and real sensor measurements. This comparison showed that the simplified model reproduces the deformation patterns of the hyperelastic reference for most contact geometries, with differences appearing mainly when the contact surface has fine or highly detailed features.

The hyperelastic materials used in the physical sensor (neoprene and polyurethane) are approximated as linear elastic materials by linearizing experimentally obtained stress–strain curves. This resulted in a Young's modulus of 1.289 MPa and a Poisson's ratio of 0.1729, values consistent with prior implementations of the sensor model in Isaac Gym [3].

The FEM mesh consists of 432 elements arranged in a 18 × 12 × 2 grid over the active deformable area of the sensor (22 × 37 mm). The bottom layer is fixed to the rigid sensor base, while the top layer interacts with the environment. Consequently, the resulting deformation map has a resolution of 18 × 12. Although a higher-resolution mesh (57 × 33) was previously used in Abaqus 2022 and Isaac Gym Preview 4, the 18 × 12 configuration was selected for Isaac Sim to ensure numerical convergence and solver stability. Given the

small physical dimensions of the sensor deformable area, this mesh density represents a practical trade-off between spatial tactile resolution and simulation robustness.

The same FEM mesh and material parameters are used for both sensors. The Robotiq TSF-85's active tactile sensing area is (22 × 37 mm), matching that of the original sensor for which this pipeline was developed. Combined with the fact that both sensors share the same sensing principle, the same number of taxels (28), and the same operating range in this work, this allows a single deformable model to represent both devices, with sensor-specific adaptation performed entirely at the CNN level, detailed in Section 3.5.

3.3. Learning-Based Generation of Tactile Maps

To approximate the complex underlying function that relates the simulator's physical deformations to the real sensor's capacitive response, a CNN was trained within the proposed pipeline. The mapping from a simulated deformation field to a real tactile sensor readout is highly nonlinear and multivariate; therefore, simple discretization or interpolation of FEM deformation maps is insufficient to reproduce realistic tactile signals. The objective of the learning pipeline is to train a CNN to take as input a 18 × 12 × 1 simulated deformation map and predict a 28-dimensional vector of synthetic tactile data that mirrors the real sensor's output. A real dataset was collected using interchangeable indenters mounted on a dynamometer, with the sensor fixed to a rigid base. Indentation depth was measured with a dial indicator, and force was applied over a range of 0–50 N. The experimental setup, indenters, and conditions for dataset creation were consistent with those previously reported in [4]. For each experiment, the tactile sensor output was recorded as tactile counts (TC) across its 28 taxels. The corresponding simulation in Isaac Sim produces deformation maps, expressed in millimetres, that represent nodal displacements of the soft top layer. Each deformation map was paired with its corresponding real taxel measurement, resulting in a dataset of 13,000 simulation–real pairs.

Given the spatial nature of the deformation maps, they were treated as single-channel images. Since the input deformation map has a spatial grid structure that preserves the spatial layout of the sensor's deformable surface, a convolutional architecture was a natural choice, as it is well-suited to exploiting local spatial structure. The architecture was optimized using the Keras Tuner Hyperband algorithm through a systematic two-stage process based on validation loss. The initial search identified the optimal base topological structure, and a second search refined layer hyperparameters such as the number of filters and units. The resulting network follows a backbone–head structure. The backbone comprises five convolutional layers, all with 2 × 2 kernels and 112, 448, 448, 512, and 512 filters, respectively. The first three convolutional layers are each followed by a 2 × 2 max-pooling layer with stride 2, while the last two are applied consecutively without intermediate pooling. A global average pooling layer then reduces each feature map to a single value, producing a 512-dimensional feature vector. This vector is passed to the head, which consists of a fully connected layer of 512 units with ReLU activation, followed by batch normalization and dropout, a second fully connected layer of 384 units with ReLU activation, and a final linear output layer of 28 units corresponding to the 28 sensor taxels. The backbone–head separation was adopted deliberately to support transfer learning: when adapting the model to a new sensor, the convolutional backbone can be frozen and only the head retrained.

To train the network, the dataset was randomly shuffled and split into training and validation sets at an 80/20 ratio. A custom loss function was used that sums the absolute differences between predicted and actual sensor outputs, quantifying the total error across all output dimensions:

$$\mathcal{L}(y_{\text{true}}, y_{\text{pred}}) = \sum_{i=1}^{N} \left| y_{\text{true},i} - y_{\text{pred},i} \right| \quad (1)$$

where $N = 28$ represents the total number of taxels, y_{true} is the real physical tactile data, and y_{pred} is the CNN-generated synthetic tactile data. The evaluation methodology and performance of this CNN on the initial sensor are reported in [3].

3.4. Isaac Sim Extension Architecture

To build a digital twin of the tactile sensor, the simulator must reproduce the sensor's input–output behaviour, that is, generate synthetic tactile signals consistent with those of the physical device under comparable contact conditions. To this end, we implemented a custom NVIDIA Isaac Sim extension that extracts deformation fields from the simulated sensor and produces synthetic tactile signals synchronously with the simulation timeline, as illustrated in Figure 3.

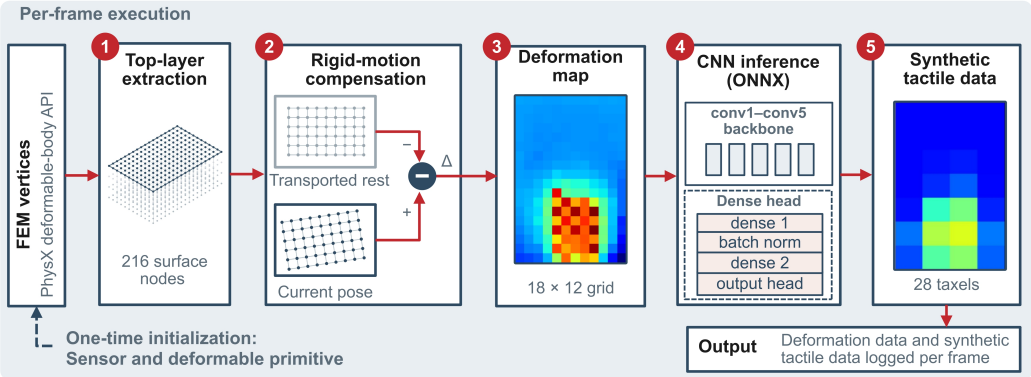


Figure 3. Internal architecture of the CoRo extension (stage 2 of Figure 2). For each frame, the top-layer nodes of the FEM mesh are extracted (1) and referenced against the rest geometry transported to the current sensor pose, isolating local deformation from rigid motion (2). The 216 nodal displacements map one-to-one onto an 18 × 12 grid (3), which a CNN converts into a 28-taxel synthetic tactile vector (4, 5).

The extension interfaces with the PhysX deformable-body API to access the FEM state of the sensor during simulation. At scene initialization, the primitive corresponding to the tactile sensor is identified and its deformable mesh is registered. From all the FEM vertices provided by the engine, only the top surface layer (216 nodes) in contact with the environment is processed to reduce computational cost. At each simulation frame, the extension retrieves the current vertex positions of the deformable mesh. By default, the extension loads a predefined list of top-layer node indices. At initialization it verifies this list against the mesh by recomputing the top surface layer; if the stored indices do not match, it extracts the current top layer and reorders the nodes to match the spatial arrangement used to train the CNN, ensuring consistency between the deformation map and the network's expected input.

At each simulation frame, the extension retrieves the current world positions of the top-layer nodes. Because the sensor moves rigidly with the robot and gripper, comparing these positions against a static reference would conflate the sensor's bulk motion with its local deformation. The reference, or rest state, is the undeformed geometry of the top-layer nodes. To isolate deformation from rigid motion, this rest geometry is rigidly transported to the sensor's current pose using the rotation and translation of the sensor case, whose pose tracks the gripper and maintains a fixed relationship to the deformable body. The nodal displacement of node i at frame t is then

$$\mathbf{d}_i(t) = \mathbf{p}_i(t) - \Big(\mathbf{R}(t)\,\mathbf{p}_i^0 + \mathbf{t}(t)\Big), \tag{2}$$

where $\mathbf{p}_i(t)$ is the current world position of node i, $\mathbf{p}_i^0$ is its rest-state position, and $\mathbf{R}(t) \in SO(3)$ and $\mathbf{t}(t)$ are the rotation and translation of the sensor case at frame t. The bracketed term is the undeformed reference transported to the sensor's current pose, so the difference isolates true surface deformation. The signed component of $\mathbf{d}_i(t)$ along the sensing axis is retained for each of the 216 top-layer nodes to form the 18×12 deformation map.

The deformation map is used as input to a pre-trained CNN, which predicts a 28-dimensional vector corresponding to the physical sensor's taxel outputs (Section 3.3). The CNN was trained in TensorFlow; however, running TensorFlow directly inside Isaac Sim causes a CUDA version conflict with the simulator's own runtime. To avoid this, the trained model is exported to ONNX and executed through ONNX Runtime within the extension, providing fast GPU inference without interfering with the Isaac Sim environment. Inference runs once per simulation frame, enabling continuous generation of synthetic tactile signals synchronized with the simulation timeline. During execution, the extension logs two data streams for offline use: first, nodal deformation values (216 elements) and second, predicted tactile outputs (28 elements). The extension also provides real-time visualization of the generated synthetic tactile maps within Isaac Sim, potentially enabling integration into fully simulated tactile perception pipelines or external learning frameworks using the recorded data.

The extension is synchronized with the physics solver so that data acquisition, inference, and logging all occur within the same time step. It connects to the on_physics_step callback and operates at a nominal frequency of 60 Hz, scaling dynamically with the simulator's timestep. After importing the extension and sensor asset into a scene, the user selects the primitive corresponding to the tactile sensor; the extension then automatically registers the associated PhysX deformable body and initializes data acquisition. Output directories and filenames are specified through the extension interface.

During simulation, two synchronized CSV files are generated, one row per frame: nodal deformation from the upper mesh layer, stored as 216-element vectors (18×12 deformation map), and the CNN-predicted synthetic tactile data, stored as 28-element vectors representing the individual taxels. The extension also renders the generated tactile maps in real time within the Isaac Sim user interface and exposes them through the internal Python 3.11 API. This enables future reinforcement learning applications in which synthetic tactile data serve as a feedback signal for closed-loop policy training.

The extension is publicly available and can be added to NVIDIA Isaac Sim by placing the repository in the Isaac Sim extensions directory and enabling it through the Extensions interface. The source code is available in two repositories: the QTAC100 pipeline at https://github.com/Lab-CORO/coro_tactile_isaacsim (accessed on 29 July 2026) and the Robotiq TSF-85 extension at https://github.com/Lab-CORO/TSF-85 (accessed on 29 July 2026). The documentation includes installation instructions and example scenes.

Latency was measured frame by frame across 50 simulation runs on two hardware configurations. The first is the development laptop (NVIDIA RTX A2000 8 GB GPU, Intel Core i9-12900H, 32 GB RAM); the second is a workstation (NVIDIA RTX A4000 16 GB GPU, Intel Core i9-12900, 62 GB RAM). Both run Isaac Sim 5.1.0. Table 1 reports the per-stage latency on both machines, including the PhysX FEM solve. On the laptop, deformation extraction dominates the 2.93 ms inference pipeline at 2.51 ms (≈86%), while CNN inference accounts for 0.37 ms (≈12% of the inference pipeline subtotal), and the combined grid projection and logging consume less than 2% of it. The combined mean per-frame cost was 8.04 ms, corresponding to 48% of the 16.67 ms frame budget at 60 Hz, so the pipeline operates in real time. At 120 Hz the frame budget is 8.33 ms, which the combined

mean approaches closely, leaving little margin on this hardware. On the workstation, the combined mean dropped to 6.71 ms, which fits within the 120 Hz budget. Overall, the extension supports real-time synthetic tactile generation at 60 Hz on both machines, while operation at 120 Hz depends on the available hardware. Peak inference spikes (up to 27.1 ms) were observed in fewer than 0.1% of frames and are attributed to OS scheduling and GPU contention rather than algorithmic cost.

Table 1. Per-stage latency on the two hardware configurations. Values are the mean ± standard deviation of the per-run means across 50 simulation runs (596 frames each, warm-up excluded). Shares are expressed as a percentage of the combined per-frame cost.

	Laptop (RTX A2000)		Workstation (RTX A4000)	
Stage	**Latency (ms)**	**Share (%)**	**Latency (ms)**	**Share (%)**
Deformation extraction	2.51 ± 0.35	31.2	2.35 ± 0.94	35.0
Grid projection	0.011 ± 0.003	0.1	0.007 ± 0.002	0.1
CNN inference (ONNX)	0.366 ± 0.038	4.5	0.574 ± 0.061	8.6
Logging	0.046 ± 0.013	0.6	0.032 ± 0.013	0.5
Inference pipeline total	2.93 ± 0.38	36.4	2.96 ± 0.99	44.2
PhysX FEM solve	5.11 ± 0.20	63.6	3.74 ± 0.19	55.8
Combined per frame	8.04 ± 0.40	100.0	6.71 ± 1.01	100.0

3.5. Model Adaptation for a New Sensor Version

Because the digital twin, deformable model, and input/output dimensions are shared across both sensors (Section 3.1), adapting the pipeline to the Robotiq TSF-85 was performed entirely at the level of the tactile-map CNN, without modifying the FEM model or the network's input ($18 \times 12 \times 1$) and output (28 taxels) structure. The adaptation follows a transfer-learning strategy that reuses the convolutional features learned on the initial sensor and refines only the parts of the network responsible for the higher-level, sensor-specific mapping. Starting from the pretrained model, we explored several freezing strategies, progressively unfreezing layers from the output toward the input to identify the configuration that best transferred to the new sensor. Three freezing depths were considered: retraining only the final dense and output layers; retraining the full head; and retraining the last convolutional block and the full head. The search was staged rather than factorial, with the learning rate adjusted to the number of trainable parameters at each depth: 1×10^{-4} for the shallowest configuration, 5×10^{-5} for the full head, and both 1×10^{-5} and 5×10^{-5} for the deepest configuration. Each configuration was trained on the indenter dataset alone and on the combined dataset. The full comparison is reported in Section 4.1. The chosen configuration kept the first four convolutional layers (backbone_conv1–conv4) frozen, while the last convolutional block (backbone_conv5) and the full head—the two fully connected layers, batch normalization, and output layer—were retrained. Freezing the early layers preserves the generic low-level deformation features, while retraining the last convolutional block and the head allows the network to refine the high-level spatial features and the deformation-to-taxel mapping for the new sensor. The model was retrained using the Adam optimizer with a learning rate of 5×10^{-5}, a batch size of 120, and the same L1-sum loss as the original training (Equation (1)). The input normalization constant from the original model was reused rather than recomputed, so that the frozen layers continue to operate on their original input distribution; this is consistent with both sensors operating over the same 0–50 N range.

The retraining dataset consists of 3129 simulation–real pairs. Of these, 2193 pairs were collected in-house following the same protocol as for the initial sensor: interchangeable

indenters mounted on a dynamometer were pressed at different positions on the sensor surface over a 0–50 N range, and each simulated deformation map was paired with its corresponding real taxel measurement. To assess generalization, the indenter shapes used for retraining differed from those used in the original training. The remaining 936 pairs were provided by Robotiq and collected across 38 different TSF-85 units by pressing a flat surface against the entire sensing area, producing full-press samples in which all taxels are loaded simultaneously; this introduces inter-sensor variability to improve the model's robustness to manufacturing tolerances. In the retraining comparison, configurations were trained either on the indenter pairs alone or on the combined dataset; in both cases, the data was split 80/20 into training and validation subsets, and the deployed configuration was selected based on validation performance. A separate evaluation set of 400 samples, collected independently from the 3129 retraining pairs and with a further distinct set of indenter shapes not present in either the training or retraining datasets, was reserved for the final assessment of the adapted model, reported in Section 4.1. The comparison of retraining strategies reported in that section was computed on this same evaluation set after selection, so that all configurations could be compared on a common data set.

3.6. Shape-Recognition Validation Setup

This section presents an application case study that demonstrates the added value of generating synthetic tactile signals with the proposed Isaac Sim extension and tactile sensor model in a zero-shot sim-to-real shape-recognition setting. The task consists of recognizing three fundamental geometric categories: cylinders, spheres, and cubes. These primitives were selected as canonical shape classes commonly used in robotic manipulation research and as representative contact geometries. Cuboids are grouped within the cube category, since they induce locally planar contact patterns under the grasp configurations considered. An overview of the procedure is shown in Figure 4.

A classification CNN, independent of the sensor-model CNN described in Section 3.3, was trained exclusively with synthetic tactile maps generated by the proposed sensor model and Isaac Sim extension, and evaluated solely on real tactile data collected from the physical sensors. It takes the generated tactile maps as input and predicts the shape class. The experimental setup follows a parallel-jaw grasping protocol: the robot was positioned at a fixed point in space with the gripper fully open, followed by closure until a stable grasp was achieved. Both simulated and real experiments followed this protocol, using a UR5e robot equipped with a Robotiq 2F-85 gripper fitted with two Robotiq TSF-85 tactile sensors, one per finger. In simulation, closure was controlled via incremental position commands, owing to the absence of direct force feedback and rigid–deformable contact information in Isaac Sim. In the real system, closure was regulated using the gripper's internal force-control interface. Despite the difference in control strategy, tactile maps were recorded only after contact stabilization, ensuring that each sample represented a steady contact state.

To create the training dataset, standard geometric primitives—spheres, cylinders, cubes, and cuboids—were used in various sizes. These objects were modelled with perfectly smooth surfaces and imported into Isaac Sim as rigid bodies. Across all grasp configurations, this process yielded 2910 synthetic tactile samples, evenly distributed among the three shape classes. The testing dataset comprised only real-world data, with 450 tactile samples (150 per class) collected from objects of varying sizes and materials, including rigid and mildly deformable items such as packaging boxes and foam-based objects. Prior to each real grasp acquisition, the sensor bias was removed to ensure a consistent baseline. The real dataset was used exclusively for the final evaluation. A summary of the dataset composition is presented in Table 2.

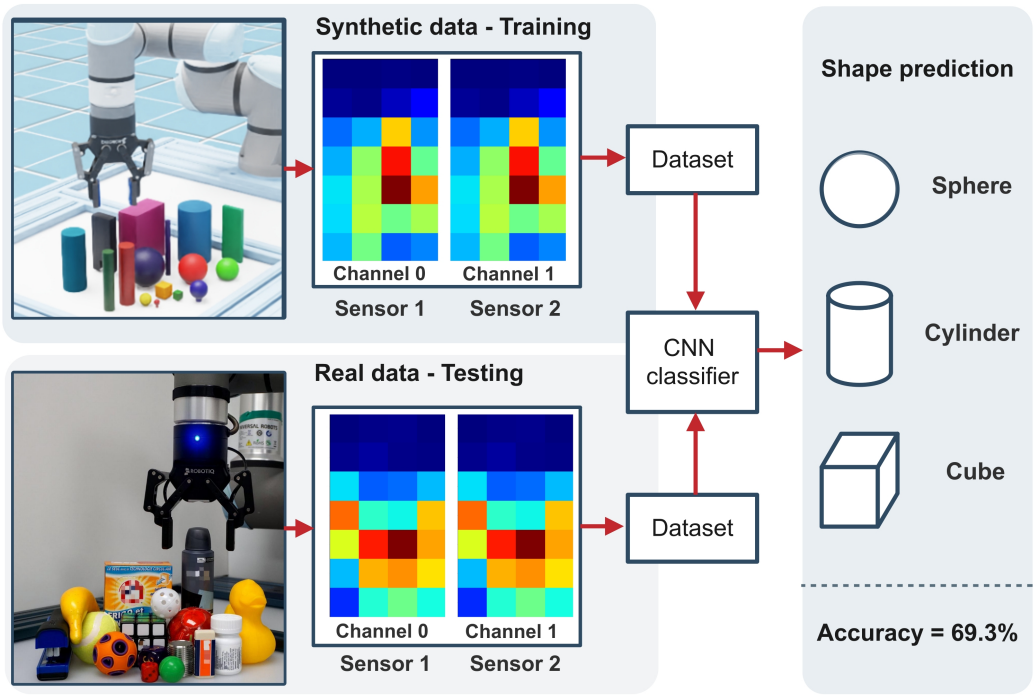


Figure 4. Shape recognition (cylinder, sphere, cube) using a CNN trained exclusively on synthetic tactile data and evaluated on real sensor tactile data.

Table 2. Composition for synthetic and real tactile datasets.

Domain	Class	Objects	Configurations	Samples
Synthetic	Sphere	6	97	970
	Cylinder	6	97	970
	Cube	5	97	970
	Total	17	291	2910
Real	Sphere	6	30	150
	Cylinder	6	30	150
	Cube	6	30	150
	Total	18	90	450

The synthetic and real object sets are comparable in scale but differ in their mechanical and geometric properties. The synthetic objects measure between 10 and 80 mm along the grasp direction, while the real objects range from approximately 15 to 70 mm. All synthetic objects were modelled as rigid bodies, whereas the real set includes five compliant items: a tennis ball, a perforated rubber pet ball, a foam toy, a foam banana toy, and a cardboard baking-soda box, all of which deform under gripper closure. The simulated objects are idealized geometric shapes with smooth surfaces, whereas the real objects are everyday items with additional surface features, including felt coverings, threading, perforations, and segmented faces.

The synthetic and real datasets were organized and stored identically: each sample corresponds to a single stabilized grasp state represented by a 56-element tactile vector given from two sensors, with 28 taxels per sensor. For classification, each sensor's taxel vector was reshaped into a 7×4 map (7 rows $\times$ 4 columns), and the two maps were stacked into a spatial tensor of size $7 \times 4 \times 2$. This representation preserves the spatial topology of the sensor layout and allows the network to exploit local spatial correlations and inter-finger contact patterns. The same spatial encoding was applied to both synthetic and real data to ensure consistency across domains.

The classifier architecture was determined through a two-stage optimization process using Keras Tuner. An initial search identified the optimal topological structure, followed by a second search to fine-tune hyperparameters such as filter density and neuron counts. The final architecture comprises two convolutional layers with filters $(3 \times 3, 32)$, and $(3 \times 3, 64)$. Each convolutional layer is followed by batch normalization and ReLU activation. The resulting feature map is flattened and passed to a fully connected layer with 96 units, activated by ReLU, followed by a softmax layer predicting the three shape classes.

Training used only synthetic tactile maps, with an 80/20 train/validation split within the synthetic dataset. The trained classifier was then evaluated directly on the independent real-world dataset without any fine-tuning on real data.

4. Results

4.1. Synthetic Tactile Map Fidelity

The retrained TSF-85 models were evaluated on an independent test set of 400 real contact samples collected with indenter shapes distinct from those used in both the original training and the retraining phase, and not seen by the model at any stage of training. It is organized into 10 distinct indenter-location classes, with 40 samples per class, over a range of applied forces from approximately 0–50 N. Reconstruction fidelity between predicted and measured tactile maps was quantified using four complementary metrics. The structural similarity index (SSIM) and the two-dimensional Pearson correlation coefficient were used to assess spatial–structural agreement, determining whether the predicted pressure pattern accurately reflects the shape and distribution of the measured pattern, largely independent of absolute magnitude. The center-of-pressure error (CoP) is a spatial metric that quantifies the accuracy of contact localization. In contrast, the root-mean-square error (RMSE) is a magnitude metric that measures the per-taxel difference in absolute pressure values and scales with signal strength. Reporting at least one metric from each family is essential, as a prediction may accurately reproduce the contact pattern while deviating in magnitude, or vice versa.

Table 3 summarizes the comparison of retraining strategies using the test set, through SSIM and RMSE. According to those results, retraining only the final layers consistently produced the lowest scores. The best results were obtained when the full dense head was retrained at a learning rate of 5×10^{-5}, with or without the last convolutional block, suggesting that the pretrained convolutional features transfer well across sensors and that the adaptation is concentrated in the dense head. Repeated runs of identical configurations differed by up to 0.03 SSIM, indicating moderate run-to-run variance; the table reports the best run for each configuration.

Analyzing the table and the retrained models based on the dataset composition, the configurations trained only on the indenter data (2193 pairs) achieved slightly higher scores on the indenter test set. The selected model is the best configuration, trained with the combined dataset of 3129 pairs, indenters and full press (SSIM 0.727). Although the configuration trained only on indenters scored higher (SSIM 0.745), the difference is comparable to the variance between runs, whereas the combined training set exposes the model to inter-sensor variability across 38 TSF-85 units. As one of the goals is for the digital twin to generalize across TSF-85 sensors rather than replicate a single one, this generalization was prioritized over a marginal gain on the single-sensor test set. It was also observed that the contact cases reported below (high, medium, and low similarity) are consistent across all retrained models: the same cases yield the highest, intermediate, and lowest similarity in every configuration, so the fidelity patterns described below are not specific to the selected model.

Table 3. Evaluation of retraining configurations for the TSF-85 adaptation on an independent test set, ordered by adaptation depth. Ind. = indenter dataset; FP = full-press dataset. The selected configuration is marked with †.

Frozen Layers	Trainable Layers	LR	Dataset	SSIM	RMSE (TC)
Backbone + dense1 + bn	dense2 + output	1×10^{-4}	Ind.	0.717	109.0
Backbone + dense1 + bn	dense2 + output	1×10^{-4}	Ind. + FP	0.716	110.3
Backbone	Full head	5×10^{-5}	Ind.	0.745	107.0
Backbone	Full head	5×10^{-5}	Ind. + FP	0.714	110.0
conv1–4	conv5 + full head	1×10^{-5}	Ind.	0.717	108.6
conv1–4	conv5 + full head	5×10^{-5}	Ind.	0.724	108.2
conv1–4	conv5 + full head	1×10^{-5}	Ind. + FP	0.722	109.8
conv1–4 †	conv5 + full head	5×10^{-5}	Ind. + FP	0.727	109.5

The remainder of this section reports the detailed fidelity results for the deployed model. Across the 400 test samples, it achieved a mean SSIM 0.727 ± 0.16 (median 0.76) and a mean Pearson correlation of 0.87 ± 0.16 (median 0.93). The high median Pearson correlation indicates that, for most samples, the predicted maps preserved the spatial arrangement of the measured contact pattern. The mean RMSE was 109.5 TC, corresponding to approximately 11% of the full-scale taxel range, and the mean center-of-pressure error was 0.41 taxels. These metrics indicate that the model reliably captures contact shape and location in most cases, with remaining errors concentrated in low-force contacts and geometrically challenging curved indenters.

Reconstruction fidelity exhibited a clear dependence on applied contact force. Structural metrics improved with increasing force: mean SSIM increased from 0.51 below 10 N to 0.77 above 30 N, and Pearson correlation rose from 0.74 to 0.90 over the same range, while the center-of-pressure error decreased from 0.50 to 0.36. This trend was statistically significant across the full sample set ($\rho = 0.34$ for SSIM and $\rho = 0.39$ for Pearson, both $p < 10^{-12}$). The observed effect is attributed to the low-force regime, where contact-induced deformation approaches the sensor's noise floor, resulting in a limited structural signal for the network to reconstruct. In contrast to the structural metrics, RMSE did not show a significant dependence on applied force ($\rho = 0.09$, $p = 0.073$), as it captures absolute taxel magnitude rather than spatial pattern agreement.

Performance was consistent across most contact geometries. Eight of the ten classes achieved mean SSIM values between 0.71 and 0.86 and Pearson correlation values between 0.88 and 0.96. The lowest fidelity was observed for a centrally located curved-contact class (mean SSIM 0.51, CoP 1.45), highlighting the increased difficulty in localizing the pressure distribution for curved indenter geometries. Figure 5 illustrates representative predicted and measured tactile maps across three fidelity regimes. In each column, the predicted map is shown above the corresponding real tactile map. The first two columns show high-force contacts with high structural fidelity: a centred cylindrical contact at 42 N (SSIM 0.95) and a cuboid contact covering approximately half of the sensor at 33 N (SSIM 0.93). The next two columns illustrate cases near the median performance: an angled cuboid contact at 20 N (SSIM 0.76) and a cuboid contact covering approximately one quarter of the sensor at 25 N (SSIM 0.76). The final two columns depict the principal failure modes: a structural-localization failure for a centred curved indenter at 14 N (SSIM 0.23, Pearson 0.11) and a low-force case near the sensor noise floor, corresponding to a cuboid contact over the inferior quarter of the sensor at 7 N (SSIM 0.29).

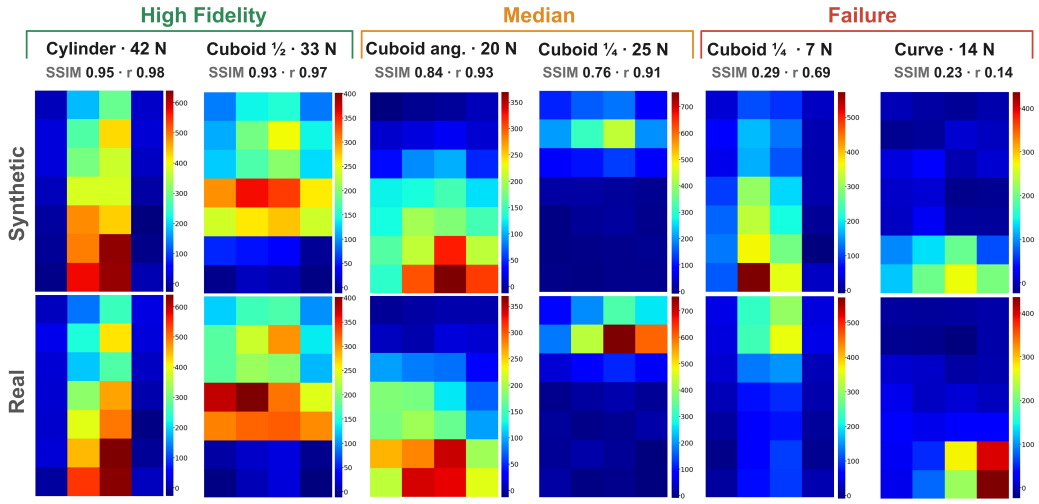


Figure 5. Representative synthetic (**top**) and real (**bottom**) tactile maps across three fidelity regimes: high fidelity (42 N, SSIM 0.95; 33 N, SSIM 0.93), median performance (20 N, SSIM 0.76; 25 N, SSIM 0.76), and failure modes (14 N, SSIM 0.23; 7 N, SSIM 0.29).

4.2. Zero-Shot Sim-to-Real Shape Recognition

The classifier achieved an overall zero-shot sim-to-real accuracy of 69.3%. The classification performance is summarized in the confusion matrix in Table 4. This result shows that synthetic tactile maps generated by the simulation model encode transferable geometric information that is transferred to physical sensor data, even without domain adaptation or fine-tuning on real samples. The confusion matrix reveals structured misclassification patterns rather than random errors. The sphere class attained the highest recognition rate (92%), indicating that symmetric curved contact geometries are consistently represented in both synthetic and real tactile data. Cylinders achieved moderate performance (59.3%), while cubes exhibited the lowest accuracy (56.7%), with most errors arising from confusion with cylinders. This pattern is consistent with the limited spatial resolution of the taxel array, which is discussed in Section 5.

For comparison, several baseline models were trained on the same synthetic dataset and evaluated on the same real test set, using a flattened 56-element vector rather than a spatial map as input. A stratified random baseline achieved 38.0% accuracy, while the random forest, multilayer perceptron, and support vector machine with an RBF kernel achieved 48.7%, 55.1%, and 60.7%, respectively, compared with 69.3% for the convolutional classifier. These baseline models were included as reference points and were not selected through an exhaustive architecture or hyperparameter search. Their performance above the random baseline provides further evidence that the synthetic tactile data contain information that transfers to the physical sensor. Although this comparison does not establish the superiority of the convolutional architecture, its higher accuracy is consistent with the spatial arrangement of the taxels providing useful information for shape recognition.

To examine how transfer performance varied across the real-world test set, the samples were grouped by object size, contact coverage, shape class, and material compliance. Centred partial contacts, in which the contact remained near the center of the sensor without covering the full sensing area, achieved 80.7% accuracy, compared with 67.3% for off-center partial contacts and 56.7% for contacts covering most or all of the sensing area. Performance also varied across shape classes, as reported above. Deformable objects achieved 75.0% accuracy, compared with 67.2% for rigid objects; however, this comparison is confounded by class composition, since 63% of the deformable samples belong to the sphere class, compared with 22% of the rigid samples. The compliance result should therefore not be interpreted as evidence that deformable objects transfer better. Overall, these descriptive

comparisons indicate that the clearest performance differences are associated with object size and with the extent and position of the contact on the sensor, rather than with material compliance alone.

Table 4. Confusion matrix for the shape recognition task.

	Predicted Class				
True Class	**Cube**	**Cylinder**	**Sphere**	**Total**	**Recall (%)**
Cube	85	38	27	150	56.7
Cylinder	47	89	14	150	59.3
Sphere	2	10	138	150	92.0
Total	134	137	179	450	
Precision (%)	63.4	65.0	77.1		69.3 [1]

[1] Overall accuracy.

As an upper bound, a synthetic-to-synthetic evaluation achieved 90.1% accuracy, indicating that the classification task is well-separable within the simulated domain. The observed performance decrease in the sim-to-real setting illustrates the remaining domain gap that exists between synthetic and real tactile data. This gap likely arises from several factors: the real sensors exhibit trial-to-trial variability even under nominally identical conditions, whereas the simulation produces deterministic deformation patterns for a given grasp; in addition, the linear-elastic approximation of the sensor's hyperelastic materials and unmodelled contact noise contribute to the discrepancy. Overall, these results show that the proposed simulation framework produces tactile data with sufficient structural fidelity to support a zero-shot transfer, while also identifying opportunities for future work to reduce the residual domain gap.

5. Discussion

A central design goal of this work was to keep the tactile-generation pipeline native to NVIDIA Isaac Sim, without requiring an external FEM solver or a cross-simulator data-generation workflow. An external FEM solver would likely improve physical fidelity, but it introduces installation complexity, version coupling, and reproducibility challenges, and it exposes the pipeline to the obsolescence pressures discussed in Section 2. By implementing the sensor model and data generation directly within Isaac Sim, the proposed framework keeps these processes synchronized with the robot and simulation timeline. This provides a self-contained workflow for manipulation experiments.

Compared with existing tactile simulation approaches, the proposed framework occupies a different position on the trade-off between physical fidelity, sensor modality, and integration with robotic workflows. TacEx [14], UniVTAC [15], and TacFlex [9] provide simulation tools for visuotactile sensors, in which an internal camera observes the deformation of a soft gel, so the simulated output is a rendered image or point cloud. The present framework instead targets capacitive sensing, where deformation modulates electrode capacitance directly, so the simulated output is a vector of taxel values. TacIPC [10] achieves higher physical fidelity through a specialized contact formulation but operates as a standalone elastomer simulator, while the present implementation uses a simplified mechanical model that executes synchronously with the robot and scene. Among capacitive approaches, CapSense [20] generates capacitive signals through electrostatic simulation, whereas TactSpace [22] avoids explicit signal reproduction by aligning simulated and real representations. In contrast, our approach learns a calibrated mapping from FEM deformation to the physical sensor output and exposes the resulting tactile signal directly within the simulation loop. Its main practical advantage is therefore a unified real-time

pipeline that provides tactile observations directly to downstream Isaac Sim or Isaac Lab tasks without coupling to external solvers. This integration also introduces limitations: the model uses a single equivalent linear-elastic layer, requires paired simulated and real samples for calibration, and retains the computational cost of FEM, which may limit its use across parallel environments.

One such constraint concerns stable grasping. In our tests, the current PhysX deformable-contact model did not provide sufficiently reliable static-frictional contact to hold a grasped object through contact forces alone. For applications requiring stabilized grasp configurations, a possible workaround is to introduce a temporary virtual joint that fixes the object relative to the gripper during contact. This approach does not reproduce real physical behaviours such as slip or contact-induced object motion, but it could provide steady-state contact configurations from which tactile maps can be extracted. This workaround is proposed as an implementation option and was not used in the shape-recognition experiments reported in this work.

The fidelity assessment in Section 4.1 was based on controlled indenter contacts. Therefore, the corresponding metrics evaluate the model's accuracy under controlled conditions and do not directly measure performance during grasping. In contrast, the shape-recognition tests in Section 4.2 were performed with a parallel gripper. The reported synthetic-to-synthetic accuracy of 90.1% and zero-shot sim-to-real transfer of 69.3% provide indirect evidence that the synthetic data preserve geometric structure that remains transferable across interaction modes.

The observed error pattern appears to be more closely associated with contact coverage and the resolution of the tactile map than with object compliance. Each sensor contains 28 taxels distributed over a 22 $\times$ 37 mm active area, so shape recognition depends on resolving the spatial form of the contact from a relatively coarse tactile map. Spherical contacts generally produce compact, approximately radial pressure patterns that remain distinctive at this resolution, which is consistent with the higher accuracy obtained for this class. In contrast, cylindrical and planar contacts produce elongated or broad pressure regions, whose differences may be less pronounced, consistent with the frequent confusion between cylinders and cubes. This interpretation is also consistent with the grouped results, where accuracy was lower for the smallest objects, where only a few taxels were activated, and for contacts covering most or all of the sensing area, where the contact boundaries and shape-dependent spatial variation were less visible. The highest accuracy was obtained for centred partial contacts, which activated enough taxels to represent the contact while preserving its spatial structure. The higher accuracy observed for deformable objects should not be attributed to compliance alone, since sphere samples represent 63% of the deformable group but only 22% of the rigid group. These observations suggest that the extent and position of the contact are stronger contributors to the remaining classification errors than object compliance.

Several differences between the synthetic and real test conditions may contribute to the remaining domain gap. The simulated objects are idealized geometric shapes with smooth surfaces, whereas the real set consists of everyday objects whose local contact regions only approximate the assigned classes. As a result, a contact pattern that is clearly associated with one class in simulation may be less distinct in a physical grasp. Moreover, the simulation produces repeatable deformation maps under identical conditions, whereas physical measurements vary due to the sensor response. In the current implementation, the simulated objects were modelled as rigid due to software constraints, whereas some real objects deform under gripper closure.

Evaluating the fidelity of synthetic tactile maps is itself non-trivial, as no single metric fully captures the relevant notion of similarity. Magnitude-based metrics such as RMSE

quantify per-taxel value accuracy but are largely insensitive to spatial structure, whereas structural metrics, such as SSIM, reward correct spatial patterns even when absolute taxel values differ. Because these metrics emphasize different, partially complementary aspects of similarity, we adopt a suite of metrics rather than a single score, allowing each to probe a distinct failure mode in the generated maps.

Finally, the choice of reference for validation deserves consideration. Treating the output of a single physical sensor as absolute ground truth can be misleading, because capacitive tactile sensors exhibit intrinsic variability: even nominally identical sensors, under identical contact conditions, produce small but consistent differences in taxel values. Exact taxel-value reproduction is therefore neither achievable nor the appropriate target. Instead, the objective is to reproduce the spatial structure and relative taxel relationships that characterize a contact—the information on which downstream tactile tasks primarily rely. This motivates our emphasis on pattern- and structure-level fidelity in both the choice of metrics and the interpretation of results.

6. Conclusions

In this work, we propose an extension for NVIDIA Isaac Sim that enables the generation of synthetic tactile data using a physical capacitive tactile sensor model. The extension integrates a FEM-based deformable model with a CNN-based inference pipeline to reproduce the complex behaviour of capacitive tactile sensing, allowing for tactile data to be generated directly within the simulation environment. By embedding the sensor model into the scene and physics engine, the proposed framework enables synchronous tactile map generation during simulation, allowing direct interaction with robotic manipulators and other simulated assets.

To illustrate a possible use of the synthetic tactile data, a zero-shot sim-to-real shape recognition task was conducted. A classifier trained entirely on synthetic tactile data and evaluated with new real sensor data shows that the simulated tactile signals capture transferable geometric contact features. Although a performance gap persists between the synthetic and real domains, these results demonstrate the feasibility of the proposed extension for generating structurally consistent tactile data. By making the extension and datasets publicly available, this work aims to support reproducible research and accelerate the integration of tactile sensing within robotic simulation workflows.

In future work, we plan to reduce the sim-to-real gap by addressing the current limitations of the deformable-body solver, which restricts contact to deformable–rigid interactions and prevents more realistic contact behaviour. To this end, we are investigating alternative GPU-based physics engines compatible with Isaac Sim and Isaac Lab that could better model these interactions, while keeping the pipeline within the Isaac ecosystem to preserve its photorealistic rendering capabilities. In parallel, we are working on training reinforcement learning policies entirely on synthetic tactile data for a tactile-driven pushing task, in which a policy learns to reposition the end-effector to maintain contact while displacing an object. This extends the use of the synthetic tactile data from static contact toward dynamic sim-to-real tasks, such as non-prehensile manipulation in cluttered environments without visual feedback.

The procedure followed for the Robotiq TSF-85 could potentially be applied to other capacitive sensors operating under the same principle, by collecting paired simulation–real samples, reusing the shared FEM model when the sensor geometry is compatible or adjusting it otherwise, adapting the CNN by first retraining the head and progressively unfreezing the final convolutional layers if necessary, and validating on an independent test set. The TSF-85 digital twin presented here has been featured by Robotiq in the context

of its integration with NVIDIA Isaac Sim [26], indicating industrial interest in simulator-native capacitive tactile sensing.

Author Contributions: Conceptualization, B.A.D.l.C.S. and J.-P.R.; methodology, B.A.D.l.C.S. and J.-P.R.; software, B.A.D.l.C.S.; validation, B.A.D.l.C.S.; formal analysis, B.A.D.l.C.S.; investigation, B.A.D.l.C.S.; resources, J.-P.R.; data curation, B.A.D.l.C.S.; writing—original draft preparation, B.A.D.l.C.S.; writing—review and editing, B.A.D.l.C.S. and J.-P.R.; visualization, B.A.D.l.C.S.; supervision, J.-P.R.; project administration, J.-P.R.; funding acquisition, J.-P.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC, grant RGPIN 2022-04884) and the Fonds de recherche du Québec—Nature et technologies (FRQNT, grant 327363).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The source code supporting the reported results is publicly available on GitHub: the QTAC100 pipeline at https://github.com/Lab-CORO/coro_tactile_isaacsim (accessed on 29 July 2026) and the Robotiq TSF-85 extension at https://github.com/Lab-CORO/TSF-85 (accessed on 29 July 2026). The synthetic and real tactile datasets generated and analysed in this study are available in the same repositories.

Acknowledgments: This research was enabled in part by support provided by Calcul Quebec (https://www.calculquebec.ca/en/ (accessed on 29 July 2026)) and the Digital Research Alliance of Canada (alliancecan.ca). The authors would like to thank Robotiq for their support of this work, and in particular, Jennifer Kwiatkowski and Vincent Duchaine for facilitating access to the Robotiq TSF-85 tactile sensor.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
CNN	Convolutional Neural Network
CoP	Center-of-Pressure
CSV	Comma-Separated Values
CUDA	Compute Unified Device Architecture
FEM	Finite Element Method
GPU	Graphics Processing Unit
IMU	Inertial Measurement Unit
LR	Learning Rate
ONNX	Open Neural Network Exchange
OS	Operating System
PCB	Printed Circuit Board
RBF	Radial Basis Function
RL	Reinforcement Learning
RMSE	Root Mean Square Error
ROS	Robot Operating System
SSIM	Structural Similarity Index Measure
TC	Tactile Counts
USD	Universal Scene Description

References

1. Melnik, A.; Lach, L.; Plappert, M.; Korthals, T.; Haschke, R.; Ritter, H. Using tactile sensing to improve the sample efficiency and performance of deep deterministic policy gradients for simulated in-hand manipulation tasks. *Front. Robot. AI* **2021**, *8*, 538773. [CrossRef] [PubMed]
2. NVIDIA. *Isaac Sim 5.1.0*, Version 5.1.0; NVIDIA: Santa Clara, CA, USA, 2026. Available online: https://developer.nvidia.com/isaac/sim (accessed on 16 June 2026).
3. De la Cruz Sánchez, B.A.; Roberge, J.P. A hybrid elastic-hyperelastic approach for simulating soft tactile sensors. *Front. Robot. AI* **2025**, *12*, 1639524. [CrossRef] [PubMed]
4. De la Cruz Sánchez, B.A.; Kwiatkowski, J.; Roberge, J.P. Tactile Contact Patterns for Robotic Grasping: A Dataset of Real and Simulated Data. In Proceedings of the 2025 3rd International Conference on Control and Robot Technology (ICCRT), Singapore, 16–18 April 2025; pp. 8–13. [CrossRef]
5. Bai, S.; Song, W.; Chen, J.; Ji, Y.; Zhong, Z.; Yang, J.; Zhao, H.; Zhou, W.; Zhao, W.; Li, Z.; et al. Towards a unified understanding of robot manipulation: A comprehensive survey. *arXiv* **2025**, arXiv:2510.10903.
6. Hu, W.; Huang, B.; Lee, W.W.; Yang, S.; Zheng, Y.; Li, Z. Dexterous in-hand manipulation of slender cylindrical objects through deep reinforcement learning with tactile sensing. *Robot. Auton. Syst.* **2025**, *186*, 104904. [CrossRef]
7. Gu, N.; Kosuge, K.; Hayashibe, M. TactileAloha: Learning Bimanual Manipulation with Tactile Sensing. *IEEE Robot. Autom. Lett.* **2025**, *10*, 8348–8355. [CrossRef]
8. Li, G.; Wang, R.; Xu, P.; Ye, Q.; Chen, J. The Developments and Challenges Toward Dexterous and Embodied Robotic Manipulation: A Survey. *IEEE Robot. Autom. Mag.* **2026**, *33*, 24–38. [CrossRef]
9. Zhang, C.; Cui, S.; Hu, J.; Jiang, T.; Zhang, T.; Wang, R.; Wang, S. TacFlex: Multimode Tactile Imprints Simulation for Visuotactile Sensors with Coating Patterns. *IEEE Trans. Robot.* **2025**, *41*, 3965–3985. [CrossRef]
10. Du, W.; Xu, W.; Ren, J.; Yu, Z.; Lu, C. TacIPC: Intersection- and Inversion-Free FEM-Based Elastomer Simulation for Optical Tactile Sensors. *IEEE Robot. Autom. Lett.* **2024**, *9*, 2559–2566. [CrossRef]
11. Collins, J.; Chand, S.; Vanderkop, A.; Howard, D. A review of physics simulators for robotic applications. *IEEE Access* **2021**, *9*, 51416–51431. [CrossRef]
12. Audonnet, F.P.; Hamilton, A.; Aragon-Camarasa, G. A systematic comparison of simulation software for robotic arm manipulation using ros2. In *Proceedings of the 2022 22nd International Conference on Control, Automation and Systems (ICCAS)*; IEEE: Piscataway, NJ, USA, 2022; pp. 755–762.
13. Huang, R.; Yang, B.; Gui, W.; Morgan, J.; Biyik, E.; Li, J. SyncTwin: Fast Digital Twin Construction and Synchronization for Safe Robotic Grasping. *arXiv* **2026**, arXiv:2601.09920.
14. Nguyen, D.H.; Schneider, T.; Duret, G.; Kshirsagar, A.; Belousov, B.; Peters, J. TacEx: GelSight Tactile Simulation in Isaac Sim–Combining Soft-Body and Visuotactile Simulators. *arXiv* **2024**, arXiv:2411.04776.
15. Chen, B.; Wan, W.; Chen, T.; Guo, X.; Xu, C.; Qi, Y.; Zhang, H.; Wu, L.; Xu, T.; Li, Z.; et al. UniVTAC: A Unified Simulation Platform for Visuo-Tactile Manipulation Data Generation, Learning, and Benchmarking. *arXiv* **2026**, arXiv:2602.10093.
16. Duret, G.; Heller, F.; Mazurak, D.; Schneider, T.; Kshirsagar, A.; Zara, F.; Peters, J.; Chen, L. Real-Time Simulation of Deformable Tactile Sensors and Objects in Robotic Grasping using Graph Neural Networks with Inductive Biases. In Proceedings of the 2026 IEEE 9th International Conference on Soft Robotics (RoboSoft), Kanazawa, Japan, 7–11 April 2026; pp. 73–80. [CrossRef]
17. Liu, Q.; Sun, Z.; Cui, Y.; Li, H.; Li, G.; Shao, L.; Chen, J.; Ye, Q. DexRepNet++: Learning Dexterous Robotic Manipulation with Geometric and Spatial Hand-Object Representations. *IEEE Trans. Robot.* **2026**, *42*, 799–818. [CrossRef]
18. Huang, X.; Xu, Z.; Xiao, C. TwinTac: A Wide-Range, Highly Sensitive Tactile Sensor with Real-To-Sim Digital Twin Sensor Model. In Proceedings of the 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hangzhou, China, 19–25 October 2025; pp. 12286–12292. [CrossRef]
19. Akinola, I.; Xu, J.; Carius, J.; Fox, D.; Narang, Y. TacSL: A Library for Visuotactile Sensor Simulation and Learning. *IEEE Trans. Robot.* **2025**, *41*, 2645–2661. [CrossRef]
20. Schöffmann, C.; Erickson, Z.; Zangl, H. CapSense: A Real-Time Capacitive Sensor Simulation Framework for Physical Human-Robot Interaction. *IEEE Robot. Autom. Lett.* **2022**, *7*, 9929–9936. [CrossRef]
21. Silva-Plata, C.; Rosel, C.; Cangan, B.G.; Alagi, H.; Hein, B.; Katzschmann, R.K.; Fernández, R.; Mojtahedi, Y.; Escaida Navarro, S. Model-Based Capacitive Touch Sensing in Soft Robotics: Achieving Robust Tactile Interactions for Artistic Applications. *IEEE Robot. Autom. Lett.* **2025**, *10*, 4596–4603. [CrossRef]
22. Joarder, A.; Bhardwaj, A.; Zurbrügg, R.; Mittal, M.; Püntener, F.; Bielefeldt, S.; Roman, C.; Patil, V.; Hutter, M. TactSpace: Learning a Physics-enriched Shared Latent Space for Tactile Sim-to-Real Transfer. *arXiv* **2026**, arXiv:2606.18959.
23. H4 Lab. QTAC100 Tactile Sensor. 2026. Available online: https://www.h4-lab.com/store/p/qtac100 (accessed on 30 June 2026).
24. Le, T.H.L.; Maslyczyk, A.; Roberge, J.P.; Duchaine, V. A highly sensitive multimodal capacitive tactile sensor. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 407–412. [CrossRef]

25. Robotiq. Tactile Sensor Fingertips TSF-85. 2026. Available online: https://robotiq.com/tactile-sensor-fingertips (accessed on 14 June 2026).
26. Robotiq. Robotiq Releases TSF-85 Tactile Sensor Digital Twin on NVIDIA Isaac Sim. 2026. Available online: https://blog.robotiq.com/robotiq-releases-tsf-85-digital-twin-on-nvidia-isaac-sim (accessed on 30 June 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.