Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Predicting a Simulation-Derived Spatial Proximity Risk Indicator from Layout Images of a Confined Robot Workspace

**Wookyung Jin**

wk.jin@kmslab.com

KMSLab Co., Ltd

**Heejung Bae**

KMSLab Co., Ltd

**Chun-Il Kim**

University of Alberta

**Research Article**

**Keywords:**

**Posted Date:** November 25th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11396321/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** No competing interests reported.

Categories (2), (8)

# Predicting a Simulation-Derived Spatial Proximity Risk Indicator from Layout Images of a Confined Robot Workspace

Wookyung Jin [ORCID 0009-0000-5010-5095][1*], Heejung Bae [ORCID 0009-0009-3537-5711][1*] and Chun-Il Kim [ORCID 0000-0003-1606-2196][2]

[1*]KMSLab Co., Ltd., 8th Floor, DMC Tower, 189 Seongam-ro, Mapo-gu, Seoul, Republic of Korea.

[2]Department of Mechanical Engineering, University of Alberta, 10-211 Donadeo Innovation Centre for Engineering, 9211 116 St, Edmonton, AB, T6G 2H5, Canada.

*Corresponding author(s). E-mail(s): wk.jin@kmslab.com; hjbae@kmslab.com;
Contributing authors: cikim@ualberta.ca;

**Abstract**

In confined robot workspaces, limited clearance between the robot and fixed structures can create crushing hazards. Assessing these hazards during layout design requires simulating each candidate layout, which becomes costly when many candidates must be compared. This paper presents an image-based surrogate that predicts a simulation-derived spatial proximity risk indicator from top-down layout images to prioritise candidates for detailed simulation and safety verification. This proxy indicator counts the grid points in clearance regions referenced to the 120 mm minimum gap for the human arm in ISO 13854 that are invaded by approximate robot geometry. Parameterised PyBullet simulations of a Franka Panda pick-and-place task provided 24,783 task-complete, collision-free samples for developing ResNet-18 and ViT-Tiny, which were evaluated on held-out layouts and separately generated within-range and extrapolated conditions. Errors increased at parameter values not used during development, but at unused within-range values ResNet-18 achieved a mean absolute error of 15.19, lower

than the 20.53 of a simulator-based grid-interpolation reference. Selecting approximately 20% of candidates with the highest ResNet-18 predictions captured 64.6% of those in the highest quartile of actual values, compared with 20.4% for random selection. The evaluation also showed that both models compressed the prediction range, predominantly underpredicting high values, and that errors were largest under extrapolation. The surrogate can therefore support preliminary screening, provided that out-of-range inputs are flagged and low predictions are not taken as evidence of adequate clearance.

**Keywords:** Robot safety, Confined workspace, Surrogate model, Layout screening, ISO 13854, Out-of-distribution generalisation

**MSC Classification:** 68T40 , 68T07

# 1 Introduction

## 1.1 Research background and necessity

The adoption of industrial robots continues to expand, with the International Federation of Robotics (IFR) reporting that 542,076 units were installed worldwide in 2024, a 0.1% increase from 2023, and that the global operational stock grew by 8.9% to 4,663,773 units. South Korea accounted for 8.4% of the global operational stock and recorded the world's highest robot density, with 1,220 industrial robots per 10,000 persons employed in manufacturing [1]. Against this background of widespread industrial robot adoption, safety should be a central consideration in the design and assessment of robot workspaces.

Occupational accidents involving industrial robots have also been documented. Layne [2] analysed U.S. Census of Fatal Occupational Injuries (CFOI) data and identified 41 occupational fatalities involving robots between 1992 and 2017, with 32 cases (78%) showing evidence that the victim came into direct contact with a robot while it was operating under its own power. A recent analysis of U.S. Occupational Safety and Health Administration (OSHA) Severe Injury Reports found that 54 of the 77 robot-related accidents reported from January 2015 to April 2022 involved fixed industrial

robots. In that analysis, finger amputations and fractures of the head and torso were the principal types of injury [3]. In South Korea, an analysis of 369 industrial-robot accidents involving operator injuries found that 191 cases (52%) were classified as jammed, caught in, or crushed and 134 cases (36%) as collision or impact. The former category included cases in which a body part was trapped between a fixed structure and a robot arm [4]. Taken together, these findings highlight the need to consider contact, entrapment, and crushing hazards in industrial robot applications.

ISO 10218-2:2025 specifies safety requirements for industrial robot applications and robot cells, while ISO 13854:2017 provides minimum gaps intended to prevent the crushing of body parts [5, 6]. As robot-cell layouts vary according to the available floor space, existing equipment locations, and required access or material-flow paths, applying these standards requires a layout-specific assessment. To support the preliminary comparison of numerous candidate layouts, a rapid, approximate method is needed to estimate spatial proximity risk and prioritise candidates for detailed simulation and safety verification.

## 1.2 Research gap

Prior research on robot safety assessment has examined empirical collision-force mapping and real-time collision-risk monitoring or control [7–10]. At the workspace-design stage, by contrast, robot-workstation design alternatives have been evaluated through computer-based simulation [11], and robotic-cell layouts have been optimised [12], but both approaches require each candidate to be evaluated individually. For robot operation time, a machine-learning surrogate has reduced this cost in the simultaneous optimisation of robotic-cell layout and motion planning [13]. However, to the best of our knowledge, no surrogate has been developed to estimate a clearance-based spatial proximity indicator for candidate layouts. The reliability of such a surrogate beyond

its development domain, which determines whether its predictions can be trusted for screening, has also not been characterised.

Guo et al. [14] represented geometry by signed-distance fields sampled on regular grids for convolutional neural network (CNN)-based flow prediction, and Berggold et al. [15] predicted total evacuation time and temporal density maps from floorplans using data generated by a parametric simulation pipeline. Tarabishy et al. [16] encoded floor plans as binary images and trained convolutional surrogates on data synthesised from a parametric model to replace computationally intensive spatial and visual connectivity simulations in layout design. These studies provide methodological precedents for learning simulation-derived outcomes from spatial representations. On this basis, layout images were adopted here because they represent workspace structures and task positions jointly in a common two-dimensional coordinate system and keep the same input format when the number, type, or shape of structures changes, whereas a parameter vector would have to be redefined. The present evaluation is restricted to simple parametric layouts so that conditions within and beyond the parameter ranges used for model development can be specified precisely.

Convolutional and transformer image-model architectures provide a methodological starting point for this task [17, 18]. However, predictive accuracy within the model-development domain alone is insufficient to establish their suitability for preliminary screening. Evaluation must therefore examine the extent to which predictions underestimate the indicator and preserve candidate rankings when layout and task conditions extend beyond that domain, with particular attention to candidate conditions with high simulation-derived indicator values.

## 1.3 Research objective and contributions

The objective of this study is to develop and evaluate an image-based surrogate that estimates a simulation-derived spatial proximity-risk proxy from workspace layout

images, in order to prioritise candidate layouts for detailed simulation and safety verification. Parameterised PyBullet simulations of a Franka Panda pick-and-place task generate the layout and task data, and ResNet-18, a CNN, and ViT-Tiny, a Vision Transformer (ViT), are used as representative convolutional and transformer architectures.

The contributions of this study are threefold:

1. A quantitative spatial proximity-risk proxy that counts the grid points invaded by approximate robot geometry in simulation within clearance regions referenced to the 120 mm minimum gap for the human arm in ISO 13854 (`invaded_grid_count`).
2. An image-based surrogate that estimates this proxy from layout images, evaluated on held-out layouts, previously unused within-range values (C0), and extrapolated conditions (C1–C4), including comparison against a simulator-based grid-interpolation reference.
3. A characterisation of the conditions under which the surrogate should not be trusted, namely prediction-range compression, underprediction of high indicator values, and direction-dependent extrapolation errors, together with the resulting rules for its use in preliminary screening.

The scope is limited to task-complete, collision-free conditions involving a single robot, a predefined motion sequence, and layouts that remain static during each trial. Within this scope, the approach is intended to prioritise candidate conditions for detailed simulation and safety verification rather than provide a final safety assessment.

# 2 Related work

## 2.1 Robot safety and simulation-based risk assessment in confined workspaces

ISO 10218-2:2025 [5] specifies safety requirements for industrial robot applications and robot cells, including requirements relevant to collaborative robot applications.

In particular, ISO 13854:2017 [6] specifies minimum safety gaps to prevent the crushing of body parts and provides different thresholds for each body part, as shown in Table 1. Although the standards provide safety requirements and minimum-gap criteria, comparing spatial proximity risk across candidate layouts requires a separate assessment methodology.

**Table 1**: Minimum safety gaps for body parts according to ISO 13854 [6]

| Body Part | Minimum Safety Gap $d$ (mm) |
|---|---|
| Finger | $\geq 25$ |
| Toes | $\geq 50$ |
| Fist / Hand / Wrist | $\geq 100$ |
| Foot | $\geq 120$ |
| Arm | $\geq 120$ |
| Leg | $\geq 180$ |
| Head | $\geq 300$ |
| Body | $\geq 500$ |

Robot safety research has addressed data-driven collision-force estimation [7] and occupancy-based dynamic risk mapping [8], alongside separation-aware control [9] and energy-aware control addressing collision and clamping [19]. Spatial approaches have also quantified separation-failure severity through bounding-region overlap and penetration [20] and represented whole-trajectory robot occupancy through swept volumes for collision checking [21]. Digital-twin systems have further integrated robot-motion and worker-location data for dynamic collision-zone management [10]. At the workspace-design stage, simulation has been used to evaluate robot-workstation design alternatives [11], and genetic-algorithm optimisation has been applied to robotic-cell workstation layouts [12]. A machine-learning surrogate of robot operation time has also been used to accelerate the simultaneous optimisation of robotic-cell layout and motion planning [13].

Against this background, the present study develops an image-based surrogate to estimate a simulation-derived spatial proximity-risk proxy and evaluates how it generalises to previously unseen parameterised layout and task conditions.

## 2.2 CNN and Vision Transformer for image-based prediction

Predicting a spatial proximity-risk indicator from a workspace layout image involves learning spatial relationships among the robot base, fixed structures, and pick/place positions encoded in the image. CNNs and ViTs construct visual representations through different architectural mechanisms. Residual CNNs combine local convolutional operations with shortcut connections that facilitate the optimisation of deep networks [17]. In contrast, ViTs represent an image as a sequence of patch embeddings and use self-attention to integrate information across image regions [18].

Previous studies have examined these architectural differences primarily in image-classification settings. In experiments with ConViT, d'Ascoli et al. [22] found that introducing soft convolutional inductive biases improved sample efficiency relative to DeiT on subsampled ImageNet classification. Raghu et al. [23] reported that ViTs aggregate global information from relatively early layers, whereas CNNs progressively construct global representations through local operations. Because the two architectures integrate spatial information differently, ResNet-18 and ViT-Tiny are used here as representatives of the two families rather than as a contribution in themselves, and their comparison is reported as part of the evaluation of the surrogate.

## 2.3 Generalisation under interpolation and extrapolation

Generalisation is relevant to the present study because image-based predictors for preliminary workspace screening may encounter layout and task conditions that were not represented during model development. Prediction beyond the support of the training distribution is commonly treated as extrapolation [24] and remains an important

challenge in scientific regression applications [25]. Although out-of-distribution (OOD) generalisation has been studied extensively for classification, regression with continuous targets has received comparatively less attention [26]. Within this broader context, the present study examines generalisation across the numerical layout and task parameters encoded in workspace images using separately generated interpolation and extrapolation conditions.

# 3 Materials and methods

## 3.1 Simulation environment

To analyse spatial proximity risks that may arise during industrial robot pick-and-place operations, a confined workspace containing a seven-degree-of-freedom Franka Emika Panda robot manipulator [27] was simulated using the PyBullet physics simulation engine [28]. Hereafter, this robot is referred to as the Franka Panda.

The task environment consisted of the robot, safety fences, a pillar, pick and place regions, trays, and a target object and was designed to simulate entrapment or crushing hazards that may occur when sufficient clearance is not secured between the robot motion path and fixed structures in a confined workspace. Figure 1 shows the simulation environment, including the safety fences, a pillar, and worktables. In particular, the safety fences and pillar were defined as representative fixed structures around the robot, enabling analysis of how changes in their positions affect intrusion into the safety-clearance regions during robot motion.

Simulation settings included a fixed robot base and a cubic target object with a side length of 40 mm. Each trial was defined as one complete pick-and-place task in which the robot grasped the target object in the pick region and moved it to the place region.

During each trial, robot joint states, end-effector pose, axis-aligned bounding boxes (AABBs) of the fixed base and all links, contact information, and joint torques were

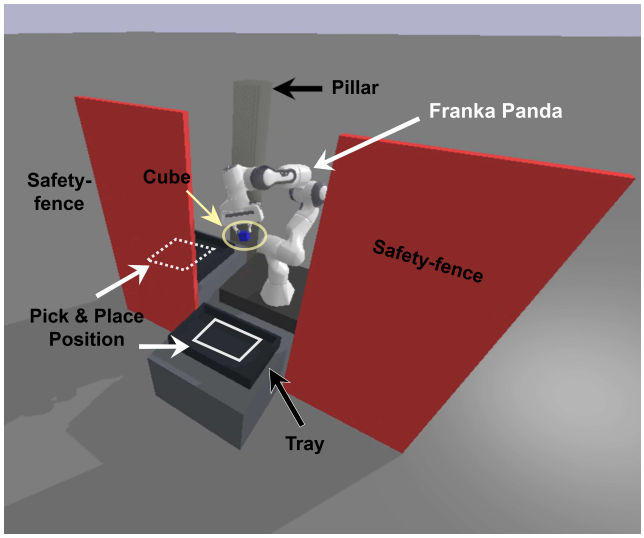



**Fig. 1** PyBullet-based confined-workspace simulation environment for the Franka Panda pick-and-place task

recorded. The recorded information was subsequently used to calculate the spatial proximity-risk proxy and determine collision and clean-success outcomes.

Claude (Anthropic) was used to assist with revision of the study code. The authors reviewed the revised code and verified its outputs.

## 3.2 Layout and trial parameterisation

The simulation conditions were organised into two parameter levels consisting of layout and trial parameters. Layout parameters specified the positions of the safety fences and pillar, whereas trial parameters reflected changes in the pick and place positions within the same layout.

Table 2 summarises the specific parameter values. Layout combinations totalled $7\times3\times3 = 63$, and trial combinations per layout totalled $5\times5\times5\times5 = 625$. Multiplying these values yielded $63 \times 625 = 39{,}375$ theoretical trial conditions in total.

Layout parameters comprised the safety-fence position `fence_x` along the $x$-axis and the pillar coordinates `pillar_x` and `pillar_y`. The trial level used

**Table 2**: Layout and trial parameter settings for simulation data generation

| Category | Parameter | Value Set (m) | Count |
|---|---|---|---|
| Layout | `fence_x` | 0.350, 0.375, 0.400, 0.425, 0.450, 0.475, 0.500 | 7 |
| | `pillar_x` | −0.45, −0.35, −0.25 | 3 |
| | `pillar_y` | −0.55, −0.45, −0.35 | 3 |
| Trial | `pick_x_offset` | −0.10, −0.05, 0.00, 0.05, 0.10 | 5 |
| | `pick_y_offset` | −0.10, −0.05, 0.00, 0.05, 0.10 | 5 |
| | `place_x_offset` | −0.10, −0.05, 0.00, 0.05, 0.10 | 5 |
| | `place_y_offset` | −0.10, −0.05, 0.00, 0.05, 0.10 | 5 |

`pick_x_offset`, `pick_y_offset`, `place_x_offset`, and `place_y_offset`, with the $(x, y)$ offsets for pick and place assigned the same value set. This design enabled changes in structural layout and task position to be systematically separated and combined.

### 3.3 Spatial proximity-risk proxy based on grid invasion of ISO 13854-referenced clearance regions

The 120 mm minimum gap specified in ISO 13854 to prevent crushing of the human arm was adopted as the reference clearance distance $d_{\text{safe}}$. This intermediate value in Table 1 was chosen because substantially larger gaps would be expected to yield high invasion counts in almost every condition, whereas the smallest gaps, such as 25 mm for fingers, would be finer than the 50 mm grid interval used in this study. A safety-clearance region was generated around each structure by expanding box-shaped structures by this distance along each axis and the cylindrical pillar by this distance from its surface. Rather than defining a workspace-wide grid, clearance-region grid points were sampled at regular intervals over the safety-clearance regions around the safety fences, pillar, trays, and surrounding workspace structures defined in each layout.

A grid sampling interval $s_{\text{grid}}$ of 50 mm was used as the discretisation resolution. This interval was an analysis setting rather than an ISO 13854 requirement, and the

absolute value of `invaded_grid_count` may vary with the grid interval. Clearance-region grid points outside the robot's effective operating range were excluded because they had little relevance to robot motion during the actual task. Figure 2 illustrates these 120 mm safety-clearance regions as yellow translucent volumes.

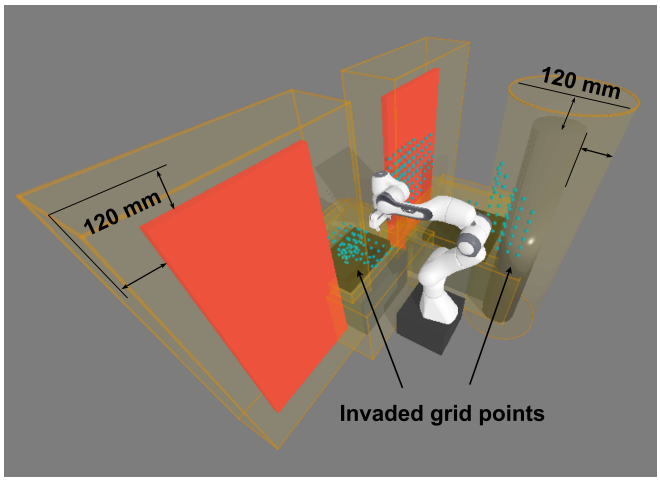



**Fig. 2** Illustration of the 120 mm safety-clearance regions (yellow translucent volumes) and clearance-region grid points invaded at least once during a trial (cyan spheres)

Robot geometry was approximated using axis-aligned bounding boxes (AABBs) recorded for the fixed base and all links at every simulation step of each trial [29]. After the simulation, each clearance-region grid point $g_i$ was classified as invaded if it lay within any recorded AABB at any simulation step. The cyan spheres in Fig. 2 visualise the grid points classified as invaded at least once over the complete trial, rather than at a single simulation step. The number of unique clearance-region grid points invaded in trial $k$ was defined as `invaded_grid_count`, as shown in Equation 1.

$$\text{invaded_grid_count}_k = \sum_{i=1}^{N_k} I_k(g_i) \tag{1}$$

Here, $N_k$ is the total number of clearance-region grid points evaluated in trial $k$, and $I_k(g_i)$ equals 1 for an invaded point and 0 otherwise. Because clearance-region

grid points were generated separately for each structure and were not merged, invaded grid samples in overlapping regions contribute separately to each structure-specific count. At a fixed grid interval, `invaded_grid_count` is proportional to a discretised estimate of the spatial volume represented by grid points classified as invaded by the recorded robot AABBs at one or more simulation steps. The per-structure estimates are summed, so overlapping regions are counted more than once rather than deduplicated into the volume of their geometric union. Each structure-specific grid point is counted only once during a trial, regardless of how long, how often, or how deeply it is entered. This count served as the spatial proximity-risk proxy and the prediction target for the layout-image-based regression models. Hereafter, values of this proxy are referred to as indicator values.

### 3.4 Collision detection and clean success definition

PyBullet's contact query was used at every simulation step to inspect physical contact between the robot and surrounding structures. Contact was recorded at a given step if at least one contact point was detected between a robot link and a safety fence, pillar, worktable, or tray. The recorded information included the presence and number of contact points, the sum of the contact normal forces, and information on the structure involved. These records were used to determine trial-level collision occurrence and analyse safety-related states during simulation.

Contact between the robot and a structure at any point during the task constituted a trial-level collision. Let $M_k$ denote the total number of simulation steps in trial $k$, and let $c_{k,t}$ denote whether contact occurred at step $t$. The corresponding indicator $C_k$ is given by Equation 2.

$$C_k = \begin{cases} 1, & \sum_{t=1}^{M_k} c_{k,t} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Task completion was assessed by whether the target object was placed sufficiently close to the designated place position within the destination tray. A trial was considered complete when the planar distance between the object's final position and the designated place position was less than 0.15 m, which served as an operational placement tolerance rather than a safety threshold or externally specified criterion.

Clean success was defined as a trial that both completed the task and remained collision-free throughout the trial, as formalised in Equation 3.

$$Q_k = \begin{cases} 1, & \text{if } T_k = 1 \text{ and } C_k = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Here, $T_k$ indicates task completion, $C_k$ indicates trial-level collision occurrence, and $Q_k$ indicates clean success. Based on this criterion, 24,783 clean-success samples were selected from the complete set of 39,375 trials as the analysis dataset for subsequent layout image generation, model training, and evaluation. Trials with task failure or collision were excluded from the regression analysis rather than treated as a higher category of the regression target. This selection focuses the study on layout-dependent spatial proximity among feasible conditions.

## 3.5 Robot motion generation

The pick-and-place motion followed a fixed sequence template in every trial, beginning at the home configuration and proceeding through the approach, grasp, lift, transit, lower, release, and retract stages. Only the absolute coordinates of selected waypoints were parameterised according to layout conditions, such as the safety-fence opening, and trial conditions, such as the pick and place positions. After the retract stage, the robot joint states were reset to the predefined home configuration before the next trial.

Each waypoint specified a target end-effector position and a fixed downward orientation. Inverse kinematics (IK) converted this target pose into values for the seven Franka Panda joints, as expressed in Equation 4.

$$q_i^* = \mathrm{IK}(p_i, R_{\mathrm{down}}) \tag{4}$$

Here, $p_i$ is the target end-effector position of the $i$th waypoint, and $R_{\mathrm{down}}$ is the fixed downward orientation applied to all waypoints. For each inverse-kinematics calculation, the current joint state, rather than the home configuration, was provided as the rest pose. This choice was intended to mitigate abrupt transitions between adjacent waypoint solutions that may arise from the redundant degree of freedom of the Franka Panda and to favour a solution close to the previous joint state. Piecewise fifth-order polynomial trajectories connected the joint waypoints calculated through inverse kinematics, as shown in Equation 5. The fifth-order polynomial form is also associated with minimum-jerk trajectories [30]. The polynomial was used for segment-wise interpolation satisfying the boundary conditions of each waypoint segment rather than for global jerk optimisation.

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{5}$$

The fifth-order polynomial coefficients were determined to satisfy the joint-position, velocity, and acceleration conditions at the start and end of each segment. Joint velocity was zero at the beginning and end of the waypoint sequence, whereas at each intermediate waypoint it equalled the average of the joint rates over the two adjacent segments. Each rate was calculated by dividing the joint-angle change by the corresponding segment duration. Acceleration was zero at both ends of every segment. These conditions maintained continuity of joint position, velocity, and acceleration at the waypoint connections and reduced abrupt joint motion.

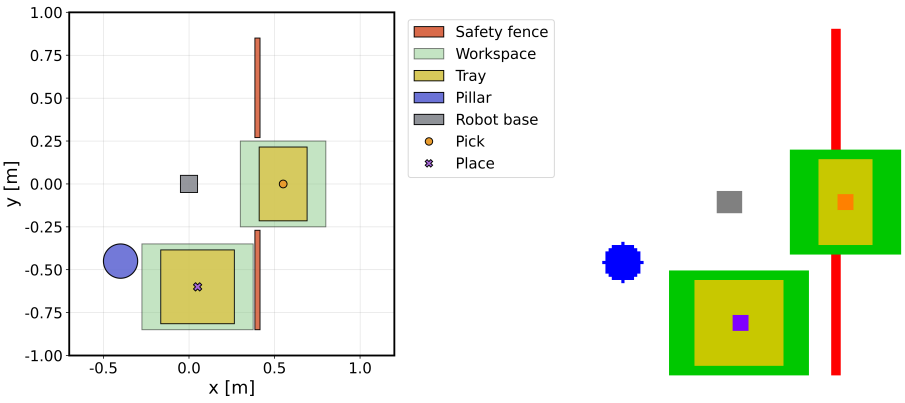
Segment step counts were determined from the joint displacement between the initial and target joint states and a study-defined velocity limit. The official Franka Panda specifications provide the maximum joint velocity [27]. For trajectory validation, a velocity limit was set to 80% of that value as a conservative margin, although this factor does not represent a manufacturer-specified safety limit. Joint velocities along the generated trajectory were then checked against this limit, and any exceedance was addressed by increasing the trajectory step count and regenerating the trajectory to satisfy the limit.

The generated joint target values were sequentially applied using PyBullet joint-position control. A simulation step was executed at intervals of 1/240 s after each control command, and the grasp and release operations were performed using gripper-position control after the predefined waypoints had been reached.

### 3.6 Layout image generation

For each trial, a 128 × 128-pixel red–green–blue (RGB) layout image representing a top-down view of the simulation environment was automatically generated as input to the CNN- and ViT-based regression models. The world-coordinate ranges of the physical workspace, $x \in [-0.7, 1.2]$ and $y \in [-1.0, 1.0]$, were normalised to the image coordinate system and then converted to pixel coordinates. Figure 3(a) presents an enlarged explanatory image with axis ticks and a legend, whereas Figure 3(b) shows the original 128 × 128-pixel image used as model input.

Each layout image combined fixed scene elements with layout- and trial-dependent positions. The Franka Panda base, the pick and place workspaces representing the worktable areas, and the tray regions were fixed. The safety-fence position and pillar coordinates varied by layout, whereas the pick/place target positions varied by trial and remained within their respective tray regions.




**(a)** **(b)**

**Fig. 3** Top-down workspace layout images. (a) Enlarged explanatory view with axis ticks and a legend. (b) Original 128 × 128-pixel RGB image used as input to the regression models

Colour assignments remained fixed across all layouts and trials. Fences appeared in red and the pillar in blue, while workspaces appeared in green and trays in yellow. Grey indicated the robot base, orange the pick position, and purple the place position. Layout images encoded workspace geometry and task positions as planar spatial relationships, while robot motion followed the predefined procedure described in Section 3.5. After generation, the images were stored in the original dataset as 128 × 128-pixel RGB images and subsequently resized and normalised during model training according to each model's input format.

## 3.7 Prediction models and training procedure

ResNet-18 and ViT-Tiny were trained to predict `invaded_grid_count` from the layout images. ResNet-18 was initialised using ImageNet-1K pretrained weights and ViT-Tiny using ImageNet-21K pretrained weights fine-tuned on ImageNet-1K, and their classification heads were replaced with a common regression head comprising a 128-dimensional hidden layer, a rectified linear unit (ReLU), dropout with a

rate of 0.3, and a single output. The model-specific feature dimensions and selected hyperparameters are presented in Table 3.

**Table 3**: Model configurations and selected training hyperparameters

| Model | Feature dimension | Learning rate | Weight decay |
|---|---|---|---|
| ResNet-18 | 512 | $1 \times 10^{-4}$ | $1 \times 10^{-6}$ |
| ViT-Tiny | 192 | $1 \times 10^{-4}$ | $1 \times 10^{-5}$ |

The layout images and prediction target were preprocessed before training. Clean-success samples generated from the discrete parameter settings in Table 2 were divided at the layout level into training, validation, and test sets with target proportions of approximately 80:10:10. To reduce the risk of leakage associated with train–test nonindependence, all trials generated from the same layout were assigned to the same split [31], and the resulting split was held fixed throughout model development.

Learning rate and weight decay were selected separately for each model from the same 12 candidate configurations based on the lowest validation loss. After the selected settings were fixed, each model was independently trained in three final runs with seeds 42, 43, and 44, and the checkpoint with the lowest validation loss from each run was used for evaluation. Detailed preprocessing, hyperparameter-search, and final-training settings are provided in Appendix A.1.

### 3.8 Additional interpolation and extrapolation evaluation

To assess generalisation to parameter values not used during model development, a separate evaluation dataset was generated using the same PyBullet procedure and excluded from training, validation, and hyperparameter selection. The dataset included both intermediate values within the original parameter ranges and values outside those

ranges. Figure 4 compares these conditions with the original discrete data-generation settings and their ranges.

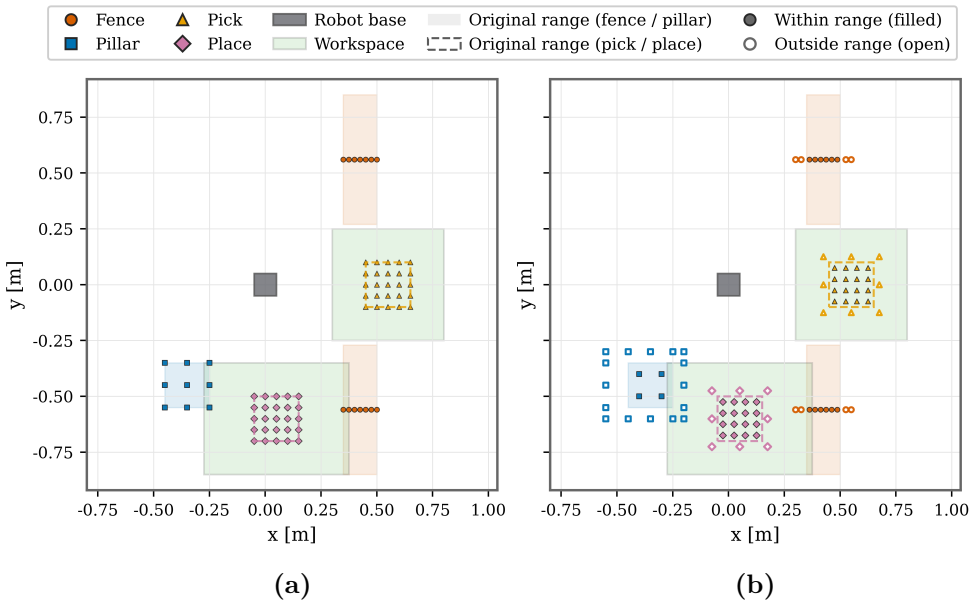



**Fig. 4** Parameter settings for the original data generation and additional evaluation. (a) Original discrete data-generation settings. (b) Separately generated intermediate within-range and extrapolation settings

The fence varied only along the $x$-axis, while the two-dimensional pillar and pick/place offsets varied along both axes using the same coordinate step sizes as the original data-generation grid. In Figure 4(b), the filled markers denote intermediate values that lie within the original parameter ranges but were not included in the discrete settings used to generate the training, validation, or standard test sets. The open markers denote values outside those ranges and are treated as extrapolation conditions. In principle, the pillar coordinates were extended symmetrically along both axes, but the extension range on one side of the $x$-axis was reduced to avoid spatial overlap with the place-offset coordinates. The coordinates and intervals of the intermediate within-range and extrapolation conditions for all four parameter groups are

summarised in Table A2, and the resulting C0–C4 category definitions and sample composition are presented in Table 4.

**Table 4**: Definitions, simulated conditions, and clean-success rates of the C0–C4 additional evaluation categories

| Category | Condition Description | Simulated ($n$) | Clean-success rate (%) | Samples ($N$) |
|---|---|---|---|---|
| C0 | All 4 groups use previously unused intermediate values within the original ranges | 299 | 66.9 | 200 |
| C1 | 1 extrapolated group + 3 intermediate within-range groups | 313 | 63.9 | 200 |
| C2 | 2 extrapolated groups + 2 intermediate within-range groups | 403 | 49.6 | 200 |
| C3 | 3 extrapolated groups + 1 intermediate within-range group | 623 | 32.1 | 200 |
| C4 | All 4 parameter groups in extrapolation condition | 890 | 22.5 | 200 |
| All | | 2,528 | 39.6 | 1,000 |

The additional evaluation dataset was constructed by sequential random sampling. For each C0–C4 category, all admissible parameter combinations were arranged in a random order generated with a fixed seed of 42 and simulated in that order until 200 clean-success samples satisfying the same task-completion and collision-free conditions as the original dataset had been obtained. The final dataset comprised 200 samples per category and 1,000 samples in total, obtained from 2,528 simulated conditions (Table 4), and the same fixed dataset was used to evaluate all three training runs. Layout images were generated for the selected samples using the same procedure as for the original dataset.

The evaluation was restricted to clean-success conditions, as for the model-development dataset, because conditions involving task failure or collision would not be adopted as layout candidates. The clean-success rate decreased as the number of extrapolated groups increased (Table 4), so the composition of each category reflects the feasible part of the design space. In particular, none of the 794 simulated conditions produced a clean-success sample when the fence was placed closer to the robot

than the original range, at `fence_x` = 0.300 or 0.325 m. Fence extrapolation in the evaluation dataset therefore covers only positions farther from the robot.

### 3.9 Evaluation metrics and indicator-value analysis

The evaluation considered overall prediction accuracy, systematic bias, performance at different actual indicator levels, and the ability to prioritise high-indicator samples for further assessment. Overall regression performance was measured using mean absolute error (MAE), root mean squared error (RMSE), $R^2$, and mean error (ME). MAE and RMSE quantified error magnitude, while $R^2$ measured goodness of fit. ME captured the direction of prediction bias and was calculated as

$$\mathrm{ME} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)\,, \tag{6}$$

where $N$ is the number of samples and $y_i$ and $\hat{y}_i$ are the actual and predicted values, respectively. Positive ME indicates overprediction and negative ME indicates underprediction. These metrics were applied to both the standard test set and the additional evaluation dataset using inverse-normalised `invaded_grid_count` values. For C0, each condition lies midway between adjacent levels of the original grid for every parameter. A simulator-based grid-interpolation reference, which involves no learning, was computed as the mean `invaded_grid_count` of the clean-success trials at the surrounding grid corners (up to $2^7 = 128$ corners, with 109 on average), indicating how closely the target can be recovered from the original grid alone. These trials were taken from the complete original grid, irrespective of the training–validation–test split.

The additional evaluation included metrics describing directional error, prediction response, and rank preservation. Underprediction rate (UPR) measured the percentage of samples for which the predicted value was lower than the actual value:

$$\mathrm{UPR} = \frac{1}{N}\sum_{i=1}^{N} I(\hat{y}_i < y_i) \times 100, \tag{7}$$

where $I(\cdot)$ is the indicator function. The prediction-response slope was obtained from the linear regression $\hat{y} = a + by$ and used to characterise how predicted values changed across the actual-value range. Spearman rank correlation measured the agreement between the rankings of actual and predicted values. UPR, prediction-response slope, and Spearman correlation were calculated for the complete additional dataset and separately for each C0–C4 category.

To examine whether prediction errors varied with the actual indicator value, the additional dataset was divided into four analysis intervals based on the distribution of actual `invaded_grid_count`. These intervals were labelled Low, Medium, High, and Extreme. The intervals were defined using the 25th, 75th, and 90th percentiles. Low, Medium, High, and Extreme corresponded approximately to the bottom 25%, the 25–75% range, the 75–90% range, and the top 10%, respectively. These labels describe relative target-value intervals and do not constitute safety classifications. MAE, ME, and UPR were calculated within each interval.

Finally, the screening analysis examined whether predicted values could prioritise samples with high actual `invaded_grid_count`. Samples were sorted in descending order of predicted value, and those in the top 25% of actual values were defined as targets. Screening recall at each top-$K$ fraction was calculated as the proportion of target samples included within that fraction and compared with a random-selection baseline. All metrics and screening results were calculated separately for each training run and summarised using the mean and sample standard deviation across the three runs.

# 4 Performance evaluation and discussion

## 4.1 Layout-dependent characteristics of the generated dataset

Before model performance was evaluated, the generated data were characterised by safety-fence and pillar position using the clean-success rate and mean

`invaded_grid_count`. The former represents the proportion of trials satisfying the clean-success criterion for each layout-parameter condition, whereas the latter was calculated over the clean-success trials and treated as a spatial proximity-risk proxy. Figure 5 shows these measures by safety-fence and pillar position.

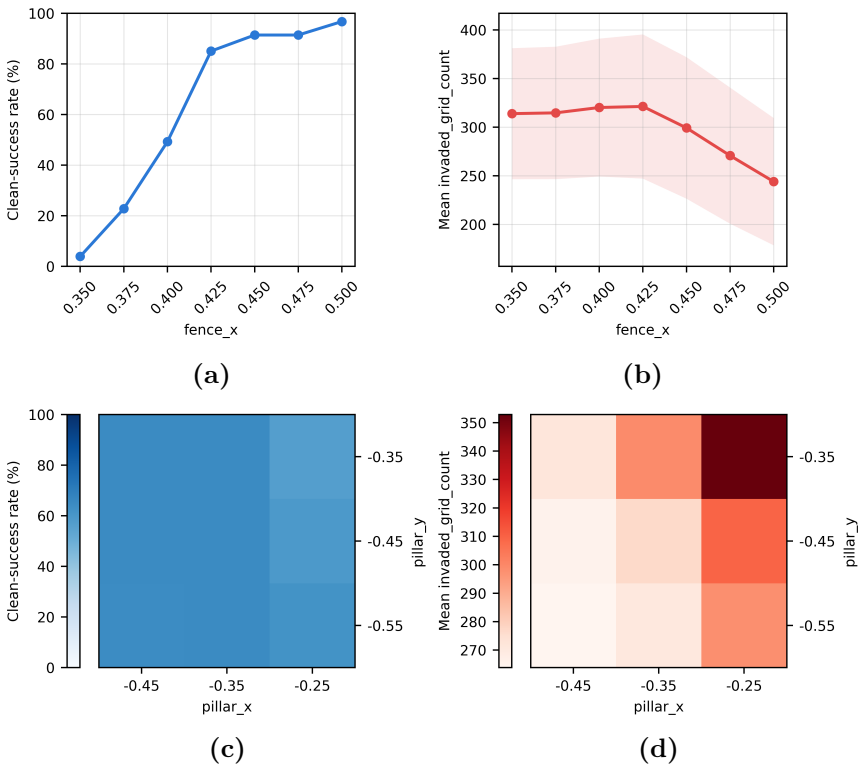



**Fig. 5** Layout-dependent characteristics of the generated dataset. (a) Clean-success rate by safety-fence position. (b) Mean `invaded_grid_count` by safety-fence position. (c) Clean-success rate by pillar position. (d) Mean `invaded_grid_count` by pillar position. The shaded region in (b) indicates ±1 standard deviation across clean-success trials

Moving the safety fence from 0.350 to 0.425 m was accompanied by a marked increase in the clean-success rate, which remained above 90% from 0.450 to 0.500 m. Mean `invaded_grid_count` remained relatively stable from 0.350 to 0.425 m and subsequently decreased as the fence moved farther away, reaching its lowest mean at 0.500 m.

Across the pillar-position grid, the clean-success rate was broadly similar at `pillar_x` values of $-0.45$ and $-0.35$ m but lower at $-0.25$ m, whereas mean `invaded_grid_count` generally increased towards $-0.25$ m and reached its highest value at the $(-0.25, -0.35)$ m position.

These patterns show that the generated data exhibited systematic layout-dependent variation in both clean-success outcomes and the spatial proximity-risk proxy.

## 4.2 Standard test performance

Table 5 presents the results of ResNet-18 and ViT-Tiny on the standard test set described in Section 3.7.

**Table 5**: Predictive performance of ResNet-18 and ViT-Tiny on the standard test set

| Model | MAE | RMSE | ME (Bias) | $R^2$ |
|---|---|---|---|---|
| ResNet-18 | $1.20 \pm 0.16$ | $1.62 \pm 0.19$ | $+0.074 \pm 0.092$ | $0.9995 \pm 0.0001$ |
| ViT-Tiny | $1.85 \pm 0.15$ | $2.49 \pm 0.21$ | $-0.379 \pm 0.406$ | $0.9988 \pm 0.0002$ |

Both models achieved high predictive performance, with mean $R^2$ values above 0.998 and mean MAE values below 2. ResNet-18 had a lower mean MAE than ViT-Tiny (1.20 versus 1.85), as well as a lower mean RMSE and a slightly higher mean $R^2$. The mean ME values indicated limited average bias under the standard test conditions. However, although the layout combinations in the standard test set were not used in training, each individual parameter value had appeared in other training

combinations. Performance at parameter values not used during model development is therefore examined separately in Section 4.3.1.

## 4.3 Performance under interpolation and extrapolation

Results are reported for the C0–C4 categories defined in Section 3.8 and Table 4.

### 4.3.1 Overall and category-wise performance

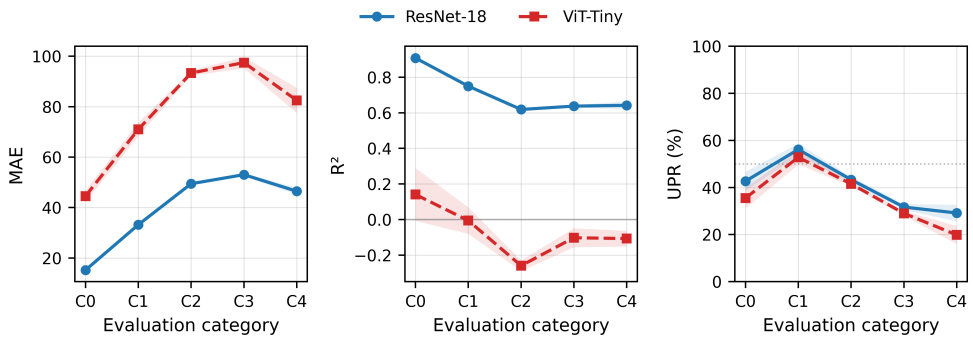
Table 6 summarises the overall performance of the two models across the complete additional evaluation dataset of 1,000 samples.

**Table 6**: Model performance on the complete additional evaluation dataset covering interpolation and extrapolation conditions

| Model | MAE | RMSE | ME | UPR (%) | $R^2$ |
|---|---|---|---|---|---|
| ResNet-18 | $39.46 \pm 0.48$ | $52.56 \pm 0.59$ | $+9.54 \pm 1.00$ | $40.6 \pm 1.0$ | $0.6935 \pm 0.0068$ |
| ViT-Tiny | $77.75 \pm 0.27$ | $97.55 \pm 1.36$ | $+29.60 \pm 1.48$ | $35.7 \pm 0.7$ | $-0.0558 \pm 0.0295$ |

Across the complete additional evaluation dataset, ResNet-18 had lower errors and a higher $R^2$ than ViT-Tiny (MAE 39.46 versus 77.75; $R^2$ 0.6935 versus $-0.0558$). The negative $R^2$ of ViT-Tiny indicates performance below that of predicting the mean value. Both models overpredicted on average, with a positive ME and a UPR below 50%. Because this dataset pools within-range and extrapolated conditions, Figure 6 compares the two models by category.

Errors increased once parameter groups were extrapolated but did not grow in proportion to the number of extrapolated groups (Figure 6). For ResNet-18, mean MAE rose from 15.19 in C0 to 49.44 in C2 and remained at a similar level in C3 and C4, and $R^2$ followed the same pattern. ResNet-18 had a lower MAE and a higher $R^2$ than ViT-Tiny in every category, whereas ViT-Tiny had a positive $R^2$ only in C0. Mean UPR exceeded 50% only in C1 (56.2% and 52.8% for ResNet-18 and ViT-Tiny),




**(a)** **(b)** **(c)**

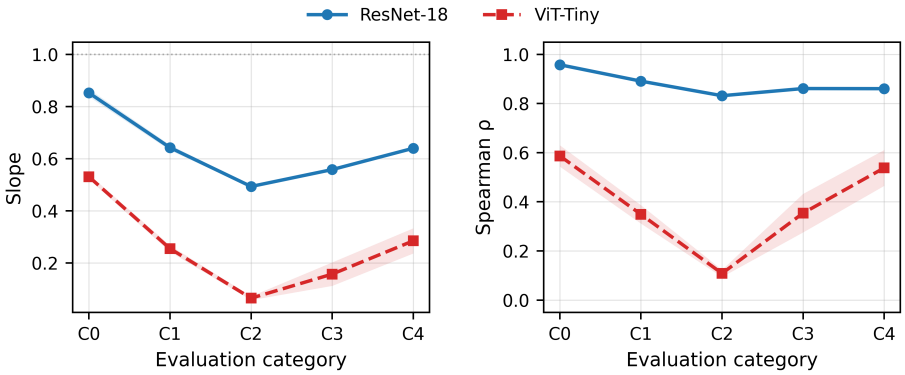
**Fig. 6** Category-wise performance of ResNet-18 and ViT-Tiny across C0–C4. (a) Mean absolute error (MAE). (b) $R^2$. (c) Underprediction rate (UPR). Shaded regions indicate the sample standard deviation across the three training runs

indicating that overprediction predominated in the other categories. Detailed results are provided in Table A3.

Although C0 lies within the original parameter ranges, the MAE of ResNet-18 in C0 was much larger than on the standard test set (15.19 versus 1.20). Unlike in the standard test set, every C0 parameter took a value between the original grid levels. To determine whether this increase arose from the model or from the target itself, the C0 samples were also estimated with the grid-interpolation reference (Section 3.9), which uses only the original-grid simulation results. This reference had an MAE of 20.53 on the same samples, and ResNet-18 was closer to the actual value for 61.5% of them. The C0 error of ResNet-18 is therefore consistent with variation of the target between the original grid levels, which the original grid does not resolve, rather than with a breakdown of the learned relationship. ViT-Tiny did not reach this reference (MAE 44.52). C0 thus provides the more representative estimate of accuracy for intermediate design values within the original ranges.

### 4.3.2 Prediction-range compression and rank preservation

Figure 7 shows prediction-response slopes and Spearman rank correlations across C0–C4. Both models had mean slopes below 1 in every category, indicating prediction-range compression, in which predicted values varied less than actual values.




**(a)** **(b)**

**Fig. 7** Prediction-response characteristics across C0–C4. (a) Mean prediction-response slope. (b) Mean Spearman rank correlation. Shaded regions indicate the sample standard deviation across the three training runs

Some degree of compression is expected for any imperfect regression model trained by minimising squared error, as in this study, so the slopes are used here to quantify its extent in each category and its consequence for high indicator values (Section 4.4.1). For ResNet-18, the mean slope was 0.852 in C0 and 0.493–0.642 in C1–C4. Despite this compression, ResNet-18 retained high rank agreement, with a mean Spearman correlation of 0.958 in C0 and 0.832–0.891 in C1–C4. ViT-Tiny was weaker in both measures, with mean slopes of 0.065–0.531 and Spearman correlations of 0.108–0.587, indicating that its low slopes reflected weak correspondence with the actual values rather than compression alone.

### 4.3.3 Screening efficiency

Screening performance was evaluated using the procedure described in Section 3.9 to assess how effectively predicted values prioritised samples with high actual values. Figure 8 shows screening recall across the evaluated fractions.

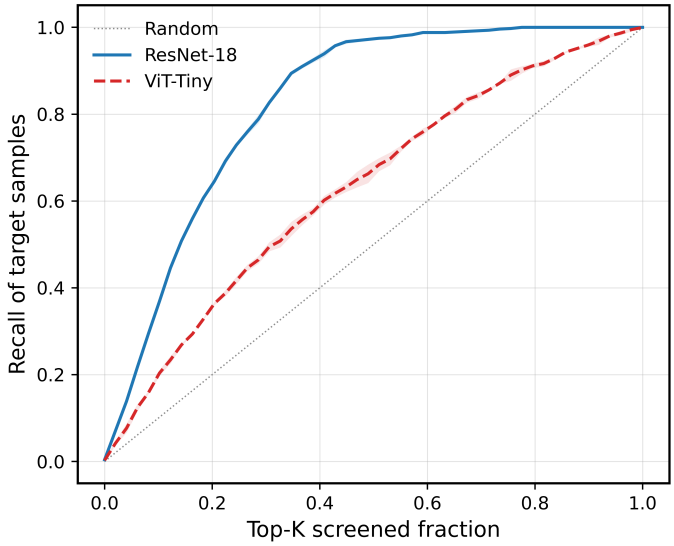


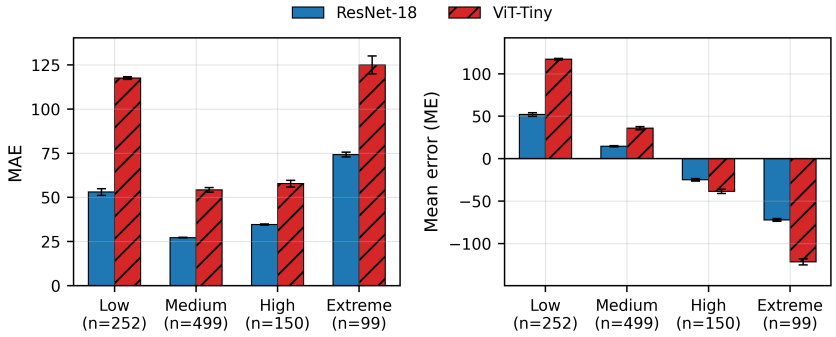

**Fig. 8** Top-$K$ screening curves showing the recall of target samples as candidate conditions were screened in descending order of predicted `invaded_grid_count`. Lines indicate the mean across the three training runs. Shaded regions indicate the sample standard deviation across these runs. The diagonal line indicates random selection

ResNet-18 achieved higher screening recall than ViT-Tiny and random selection at all screening fractions $0 < K < 1$, reflecting its stronger rank agreement (Section 4.3.2). At the evaluated fractions closest to 0.2 and 0.4 ($K \approx 0.20$ and 0.41), the mean recall was 0.646 and 0.939 for ResNet-18, 0.364 and 0.602 for ViT-Tiny, and 0.204 and 0.408 for random selection. Because the targets comprised 25% of the samples, the maximum attainable recall was approximately 0.82 at $K \approx 0.20$ and 1.0 at $K \approx 0.41$.

## 4.4 Indicator-value- and parameter-dependent errors

### 4.4.1 Error and bias across indicator-value intervals

Figure 9 summarises prediction error and bias across the predefined intervals of actual `invaded_grid_count`. The intervals were Low ($\leq$ 213), Medium (214–340), High (341–409), and Extreme ($\geq$ 410).




**(a)** **(b)**

**Fig. 9** Prediction performance across actual `invaded_grid_count` intervals for ResNet-18 and ViT-Tiny. (a) Mean absolute error (MAE). (b) Mean error (ME). Error bars indicate the sample standard deviation across the three training runs

Mean error changed sign across the intervals (Figure 9(b)). Both models overpredicted in the Low and Medium intervals and underpredicted in the High and Extreme intervals, consistent with the prediction-range compression identified in Section 4.3.2. For ResNet-18, ME ranged from +52.04 in the Low interval to −72.25 in the Extreme interval, and UPR increased from 3.6% to 96.0%, with similar trends for ViT-Tiny (Table A4). High actual `invaded_grid_count` values were therefore predominantly underpredicted. Because the Low and Medium intervals contained 751 of the 1,000 samples, these opposite biases resulted in the positive overall ME reported in Section 4.3.1.

For both models, MAE was smallest in the Medium interval and larger towards both ends of the actual-value range, reaching its maximum in the Extreme interval (ResNet-18: 27.18 in the Medium and 74.22 in the Extreme interval; ViT-Tiny: 54.26 and 125.05).

### 4.4.2 Errors beyond the original parameter ranges

To examine where in the layout space the errors reported in Sections 4.3.1 and 4.4.1 arose, Figure 10 shows the mean signed error (predicted − actual) of the additional evaluation samples grouped by fence position and by pillar position.

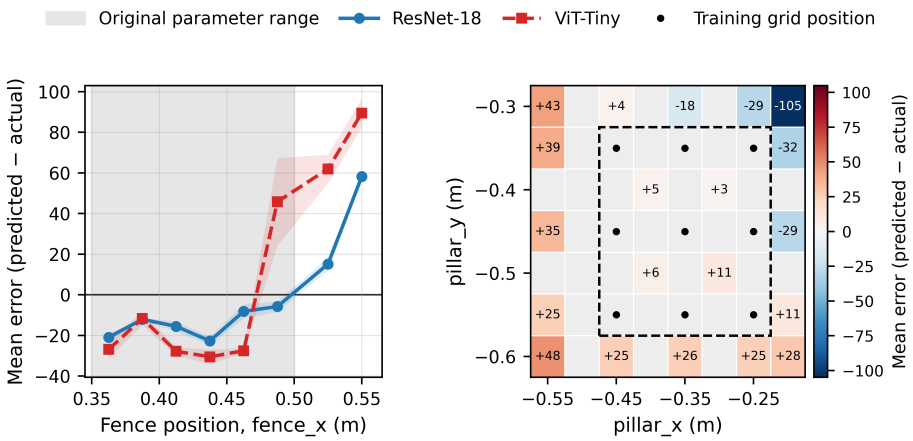



**Fig. 10** Mean signed error (predicted − actual) of the additional evaluation samples by layout position. (a) Fence position for ResNet-18 and ViT-Tiny. (b) Pillar position for ResNet-18. Values are averaged over the evaluation samples sharing each position and over the three training runs, and shaded regions in (a) indicate the sample standard deviation across the three training runs. The grey band in (a) and the dashed box in (b) mark the original parameter ranges, dots in (b) mark the training grid positions, and grey cells contain no evaluation samples. The robot base is located at the origin, towards the upper right in (b)

For the fence, the mean error of ResNet-18 was negative within the original range (−5.8 to −22.7) and became positive beyond it, reaching +58.3 at 0.550 m, because the predictions did not follow the decrease in the mean actual value from 270.8 at

0.4875 m to 229.1 at 0.550 m. ViT-Tiny overpredicted more strongly, already by +45.8 at 0.4875 m. Fence positions closer to the robot than the original range were not represented because no clean-success sample was obtained for them (Section 3.8).

For the pillar, the mean error of ResNet-18 was +3 to +11 within the original range. Outside it, the error was positive (+25 to +48) when the pillar lay away from the robot (`pillar_x` = −0.55 m or `pillar_y` = −0.60 m) and mostly negative when it lay towards the robot (`pillar_x` = −0.20 m or `pillar_y` = −0.30 m). The largest underprediction (−105) occurred at (−0.20, −0.30) m, the evaluated pillar position closest to the robot, where the mean actual value was 431.

Outside the original ranges, the error therefore depended on the direction in which a layout parameter left the range. Predictions exceeded the actual values when the fence or pillar lay farther from the robot, where the indicator was lower, and mostly fell below them when the pillar lay closer to it, where the indicator was higher. Because the samples sharing a position differ in their other parameters, these position-wise means describe where the errors occurred rather than isolating the effect of each parameter.

## 4.5 Practical implications

A practical implementation could use the surrogate to support workspace layout design by comparing candidate layouts and task conditions. Consistent with the clean-success criterion used to construct the dataset (Section 3.4), it is intended for candidates that satisfy task-completion and collision-free conditions. The screening results indicate that ranking by predicted `invaded_grid_count` can concentrate higher-indicator conditions within a smaller assessment subset. In this use, the predicted values determine which candidates are examined or redesigned first, whereas any candidate retained for adoption still undergoes detailed simulation and safety verification. On the computer used to simulate the additional evaluation conditions, generating a layout image and predicting its `invaded_grid_count` with ResNet-18 on

the CPU took approximately 35 ms per candidate, compared with a mean of 6.9 s for one headless simulated trial. Because errors increased outside the original parameter ranges and, for pillar positions closer to the robot, took the form of underprediction, inputs outside these ranges should be flagged on the basis of their parameter values and directed to detailed simulation. Because high actual values were predominantly underpredicted, a low predicted `invaded_grid_count` should not be regarded as evidence of adequate clearance.

## 4.6 Limitations

`invaded_grid_count` is a spatial proximity-risk proxy and does not directly represent accident probability or injury severity. Its absolute value also depends on the 50 mm grid interval and on the number and size of the structures.

The data further restrict the scope of the findings. All data were generated in PyBullet, leaving real-world transfer unvalidated and subject to a potential simulation-to-real gap [32]. The data covered a single fixed-base robot, a predefined pick-and-place motion template, and safety fences and a pillar as the only variable structures.

On the modelling side, only two architectures were evaluated. Because no numerical-parameter baseline was evaluated, the results characterise the feasibility and generalisation behaviour of the selected image-based representation but do not establish its superiority over numerical-input regression models.

The additional evaluation used a fixed sample of conditions. Because the workspace was confined, no task-complete, collision-free condition was obtained with the fence closer to the robot than the original range, so fence extrapolation covered only positions farther from the robot. The categories also differed in which parameter groups were extrapolated, and the category index therefore describes the coverage of extrapolation rather than an ordinal measure of difficulty.

# 5 Conclusion

In confined robot workspaces, assessing crushing hazards between the robot and fixed structures during layout design requires simulating each candidate layout, which becomes costly when many candidates must be compared. An image-based surrogate was developed to estimate a simulation-derived spatial proximity-risk proxy from layout images, using parameterised PyBullet simulations of a Franka Panda pick-and-place task, and ResNet-18 and ViT-Tiny were evaluated as representative architectures.

`invaded_grid_count` provided a quantitative proxy for comparing candidate layouts with the same set of structures, and its values varied systematically with the fence and pillar positions. The high accuracy of the ResNet-18-based surrogate on the standard test set (MAE 1.20) did not carry over to previously unused within-range values (C0), where the MAE was 15.19. This error was nevertheless lower than the 20.53 of the simulator-based grid-interpolation reference, and ResNet-18 had a lower MAE than ViT-Tiny in every interpolation and extrapolation category. Because ResNet-18 largely preserved candidate rankings, selecting approximately 20% of the additional evaluation samples with the highest predictions captured 64.6% of those in the highest quartile of actual indicator values, compared with 20.4% for random selection.

The evaluation also identified the conditions under which the surrogate should not be trusted. Both models compressed the prediction range, so that high actual indicator values were predominantly underpredicted, and errors increased under extrapolation, taking the form of underprediction when the pillar was moved towards the robot. Accordingly, inputs outside the development ranges should be flagged on the basis of their parameter values and directed to detailed simulation, and a low prediction should not be treated as evidence of adequate clearance.

Future work should validate the approach with actual robots, broader robot and task conditions, and layouts containing additional object types and irregular structure shapes, which the image representation can encode without changing the model

input. Denser or adaptive sampling of the layout parameters should be examined in view of the target variation between the original grid levels, together with higher image resolutions that can represent smaller parameter changes. The indicator could further be extended to incorporate actual collision geometry, approach velocity, and contact force. Predictive uncertainty should also be quantified under extrapolation, and combining the model with optimisation methods could support the inverse design of workspace layouts.

Overall, the surrogate may support preliminary screening by prioritising candidate layouts and task conditions for detailed simulation and safety verification, but it should not replace final safety assessment.

## References


[1] Scheuter, M.: Automation is a Cornerstone of Modern Manufacturing (2025). https://ifr.org/post/automation-is-a-cornerstone-of-modern-manufacturing Accessed 2026-08-05

[2] Layne, L.A.: Robot-related fatalities at work in the United States, 1992–2017. American Journal of Industrial Medicine **66**(6), 454–461 (2023) https://doi.org/10.1002/ajim.23470

[3] Sanders, N.E., Şener, E., Chen, K.B.: Robot-related injuries in the workplace: An analysis of OSHA severe injury reports. Applied Ergonomics **121**, 104324 (2024) https://doi.org/10.1016/j.apergo.2024.104324

[4] Lee, K., Shin, J., Lim, J.-Y.: Critical hazard factors in the risk assessments of industrial robots: Causal analysis and case studies. Safety and Health at Work **12**(4), 496–504 (2021) https://doi.org/10.1016/j.shaw.2021.07.010

[5] International Organization for Standardization: ISO 10218-2:2025, Robotics—Safety requirements—Part 2: Industrial robot applications and robot cells. International standard, Geneva, Switzerland (2025). https://www.iso.org/standard/73934.html Accessed 2026-08-06

[6] International Organization for Standardization: ISO 13854:2017, Safety of Machinery—Minimum Gaps to Avoid Crushing of Parts of the Human Body. International standard, Geneva, Switzerland (2017). https://www.iso.org/standard/66459.html Accessed 2026-08-03

[7] Svarny, P., Rozlivek, J., Rustler, L., Hoffmann, M.: 3D Collision-Force-Map for safe human-robot collaboration. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 3829–3835 (2021). https://doi.org/10.1109/ICRA48506.2021.9561845

[8] Anatoliotakis, N., Paraskevopoulos, G., Michalakis, G., Michalellis, I., Zacharaki, E.I., Koustoumpardis, P., Moustakas, K.: Dynamic human–robot collision risk based on octree representation. Machines **11**(8), 793 (2023) https://doi.org/10.3390/machines11080793

[9] Zanchettin, A.M., Ceriani, N.M., Rocco, P., Ding, H., Matthias, B.: Safety in human-robot collaborative manufacturing environments: Metrics and control. IEEE Transactions on Automation Science and Engineering **13**(2), 882–893 (2016) https://doi.org/10.1109/TASE.2015.2412256

[10] Kwon, W., Yang, J., Song, S., Lee, J., Kim, H.: Real-time digital-twin-based cobot-worker collision risk prediction using Unity, ROS, and UWB. IEEE Access **13**, 85967–85978 (2025) https://doi.org/10.1109/ACCESS.2025.3569332

[11] Ore, F., Vemula, B., Hanson, L., Wiktorsson, M., Fagerström, B.: Simulation

methodology for performance and safety evaluation of human–industrial robot collaboration workstation design. International Journal of Intelligent Robotics and Applications **3**(3), 269–282 (2019) https://doi.org/10.1007/s41315-019-00097-0

[12] Sánchez-Sosa, R.-A., Chavero-Navarrete, E.: Robotic cell layout optimization using a genetic algorithm. Applied Sciences **14**(19), 8605 (2024) https://doi.org/10.3390/app14198605

[13] Kawabe, T., Nishi, T., Liu, Z., Fujiwara, T.: Surrogate-assisted motion planning and layout design of robotic cellular manufacturing systems. Engineering Applications of Artificial Intelligence **150**, 110530 (2025) https://doi.org/10.1016/j.engappai.2025.110530

[14] Guo, X., Li, W., Iorio, F.: Convolutional neural networks for steady flow approximation. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 481–490. ACM, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939738

[15] Berggold, P., Nousias, S., Dubey, R.K., Borrmann, A.: Towards predicting pedestrian evacuation time and density from floorplans using a vision transformer. In: Proceedings of the 30th International Workshop on Intelligent Computing in Engineering (EG-ICE 2023), pp. 639–648 (2023)

[16] Tarabishy, S., Psarras, S., Kosicki, M., Tsigkari, M.: Deep learning surrogate models for spatial and visual connectivity. International Journal of Architectural Computing **18**(1), 53–66 (2020) https://doi.org/10.1177/1478077119894483

[17] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern

Recognition (CVPR), pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90

[18] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (ICLR) (2021)

[19] Lachner, J., Allmendinger, F., Hobert, E., Hogan, N., Stramigioli, S.: Energy budgets for coordinate invariant robot control in physical human–robot interaction. The International Journal of Robotics Research **40**(8–9), 968–985 (2021) https://doi.org/10.1177/02783649211011639

[20] Marvel, J.A.: Performance metrics of speed and separation monitoring in shared workspaces. IEEE Transactions on Automation Science and Engineering **10**(2), 405–414 (2013) https://doi.org/10.1109/TASE.2013.2237904

[21] Lee, M.-H., Liu, J.-S.: Reliable and accurate implicit neural representation of multiple swept volumes with application to safe human–robot interaction. SN Computer Science **5**, 313 (2024) https://doi.org/10.1007/s42979-024-02640-8

[22] d'Ascoli, S., Touvron, H., Leavitt, M.L., Morcos, A.S., Biroli, G., Sagun, L.: ConViT: Improving vision transformers with soft convolutional inductive biases. In: Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 2286–2296 (2021)

[23] Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., Dosovitskiy, A.: Do vision transformers see like convolutional neural networks? In: Advances in Neural Information Processing Systems, vol. 34, pp. 12116–12128 (2021)

[24] Xu, K., Zhang, M., Li, J., Du, S.S., Kawarabayashi, K.-i., Jegelka, S.: How

neural networks extrapolate: From feedforward to graph neural networks. In: International Conference on Learning Representations (ICLR) (2021)

[25] Na, G.S., Park, C.: Nonlinearity encoding for extrapolation of neural networks. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1284–1294 (2022). https://doi.org/10.1145/3534678.3539326

[26] Eyre, B., Creager, E., Madras, D., Papyan, V., Zemel, R.: Out of the ordinary: Spectrally adapting regression for covariate shift. In: Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 12701–12722. PMLR, Vienna, Austria (2024)

[27] Franka Robotics GmbH: Control Interface Specification and Robot Limits. Franka Control Interface Documentation. https://frankarobotics.github.io/docs/robot_specifications.html Accessed 2026-08-03

[28] Coumans, E., Bai, Y.: PyBullet, a Python module for physics simulation for games, robotics and machine learning. Online software documentation (2016–2021). http://pybullet.org Accessed 2026-08-03

[29] Ericson, C.: Real-Time Collision Detection, 1st edn. CRC Press, Boca Raton, FL, USA (2004)

[30] Flash, T., Hogan, N.: The coordination of arm movements: An experimentally confirmed mathematical model. The Journal of Neuroscience **5**(7), 1688–1703 (1985) https://doi.org/10.1523/JNEUROSCI.05-07-01688.1985

[31] Kapoor, S., Narayanan, A.: Leakage and the reproducibility crisis in machine-learning-based science. Patterns **4**, 100804 (2023) https://doi.org/10.1016/j.patter.2023.100804

[32] Aljalbout, E., Xing, J., Romero, A., Akinola, I., Garrett, C.R., Heiden, E., Gupta, A., Hermans, T., Narang, Y., Fox, D., Scaramuzza, D., Ramos, F.: The reality gap in robotics: Challenges, solutions, and best practices. Annual Review of Control, Robotics, and Autonomous Systems **9**, 403–432 (2026) https://doi.org/10.1146/annurev-control-031924-100130

[33] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015)

## Statements and Declarations

**Author contributions.** Wookyung Jin led the research, developed the software, performed the simulations, data analysis and model validation, prepared the figures, and drafted the manuscript. Heejung Bae and Chun-Il Kim contributed to study design and methodology, supervised the research, managed the project, and reviewed and edited the manuscript.

**Funding.** This research was supported by the Korea Institute for Advancement of Technology (KIAT) grant funded by the Korea Government (MOTIE) (RS-2024-00435502, Human Resource Development Program for Industrial Innovation (Global)).

**Data availability.** The datasets generated and analysed during this study, together with the trained model checkpoints, are available in GitHub release v1.1.0 of the repository https://github.com/wk-Jin/confined-robot-proximity-risk. The same repository provides the simulation, training, evaluation and figure-generation code.

**Ethics approval.** Not applicable.

**Consent to participate.** Not applicable.

**Consent for publication.** Not applicable.

**Competing interests.** The authors declare no relevant financial or non-financial competing interests.

# Appendix A Additional methodological details and evaluation results

## A.1 Preprocessing and training settings

The original 128 × 128-pixel RGB images were resized to 224 × 224 pixels by bilinear interpolation and normalised using the ImageNet channel-wise means $[0.485, 0.456, 0.406]$ and standard deviations $[0.229, 0.224, 0.225]$. The target was z-score normalised using only the training-set statistics, $\mu_{\text{train}} = 287.57$ and $\sigma_{\text{train}} = 78.18$, and predictions were inverse-normalised before evaluation.

The fixed layout-level split was generated using a data-split seed of 42. ReduceLROnPlateau was used for learning-rate scheduling in both training stages.

Final training used Adam [33], mean squared error (MSE), and a batch size of 64.

Table A1 summarises the hyperparameter-search and final-training settings.

**Table A1**: Hyperparameter-search and final-training settings

| Setting | Hyperparameter search | Final training |
|---|---|---|
| Seed(s) | 42 | 42, 43, 44 |
| Learning rate | $1 \times 10^{-5}$, $5 \times 10^{-5}$, $1 \times 10^{-4}$, $5 \times 10^{-4}$ | ResNet-18: $1 \times 10^{-4}$; ViT-Tiny: $1 \times 10^{-4}$ |
| Weight decay | $1 \times 10^{-6}$, $1 \times 10^{-5}$, $1 \times 10^{-4}$ | ResNet-18: $1 \times 10^{-6}$; ViT-Tiny: $1 \times 10^{-5}$ |
| Maximum epochs | 20 | 200 |
| Learning rate scheduling | Factor 0.5, patience 3 | Factor 0.5, patience 5 |
| Early stopping | Patience 5 | Patience 10 |

## A.2 Parameter settings for additional interpolation and extrapolation evaluation

Table A2 lists the coordinate values used to construct the additional evaluation conditions. The intermediate within-range values were located between the discrete parameter settings used for the original data generation and were not included in the training, validation, or standard test sets.

The pillar coordinates, pick offset, and place offset each formed one $(x, y)$ parameter group. A group was classified as extrapolated when either coordinate fell outside the parameter range represented during model development, including cases with only one out-of-range axis.

**Table A2**: Coordinates for the additional evaluation conditions (unit: m)

| Parameter | Step | Intermediate within-range values | Extrapolated-group candidates |
|---|---|---|---|
| Fence $x$ | 0.025 | 0.3625, 0.3875, 0.4125, 0.4375, 0.4625, 0.4875 | 0.300, 0.325; 0.525, 0.550 |
| Pillar $x$ | 0.10 | −0.40, −0.30 | −0.55, −0.45, −0.35, −0.25, −0.20 |
| Pillar $y$ | 0.10 | −0.50, −0.40 | −0.60, −0.55, −0.45, −0.35, −0.30 |
| Pick offset $x$, $y$ | 0.05 | −0.075, −0.025, 0.025, 0.075 | −0.125, 0, 0.125 |
| Place offset $x$, $y$ | 0.05 | −0.075, −0.025, 0.025, 0.075 | −0.125, 0, 0.125 |

**Note:** The final column lists candidate coordinate values used to construct extrapolated groups. For pick/place offsets, candidate pairs were formed as the Cartesian product of the values listed for each axis in the final column, with $(0, 0)$ excluded because it did not satisfy the extrapolation criterion. The inward fence candidates (0.300 and 0.325) produced no clean-success layouts and therefore do not appear in the final additional evaluation dataset.

## A.3 Detailed performance across C0–C4 evaluation categories

Table A3 provides the numerical values for the C0–C4 categories defined in Table 4 and presented graphically in Figure 6.

**Table A3**: Model performance across the C0–C4 evaluation categories

| Category | ResNet-18 | | | ViT-Tiny | | |
|---|---|---|---|---|---|---|
| | MAE | $R^2$ | UPR (%) | MAE | $R^2$ | UPR (%) |
| C0 | $15.19 \pm 0.31$ | $0.9076 \pm 0.0058$ | $42.7 \pm 4.0$ | $44.52 \pm 2.20$ | $0.1408 \pm 0.1479$ | $35.5 \pm 4.4$ |
| C1 | $33.18 \pm 0.45$ | $0.7496 \pm 0.0081$ | $56.2 \pm 2.5$ | $70.97 \pm 2.73$ | $-0.0058 \pm 0.0740$ | $52.8 \pm 3.0$ |
| C2 | $49.44 \pm 0.21$ | $0.6186 \pm 0.0025$ | $43.3 \pm 1.3$ | $93.32 \pm 1.06$ | $-0.2590 \pm 0.0295$ | $41.5 \pm 0.9$ |
| C3 | $53.03 \pm 0.64$ | $0.6374 \pm 0.0071$ | $31.7 \pm 1.3$ | $97.46 \pm 2.21$ | $-0.1022 \pm 0.0529$ | $29.0 \pm 1.0$ |
| C4 | $46.47 \pm 1.35$ | $0.6418 \pm 0.0225$ | $29.2 \pm 3.5$ | $82.49 \pm 4.95$ | $-0.1068 \pm 0.0442$ | $19.8 \pm 3.8$ |

## A.4 Prediction errors and underprediction rates by indicator-value interval

Table A4 provides the numerical values for the indicator-value intervals defined in Section 3.9 and presented graphically in Figure 9, together with UPR for each interval.

**Table A4**: Prediction errors and underprediction rates by indicator-value interval

| Indicator-value interval | ResNet-18 | | | ViT-Tiny | | |
|---|---|---|---|---|---|---|
| | MAE | ME | UPR (%) | MAE | ME | UPR (%) |
| Low ($\leq 213$, $n = 252$) | $53.01 \pm 1.84$ | $+52.04 \pm 2.07$ | $3.6 \pm 0.4$ | $117.59 \pm 0.75$ | $+117.16 \pm 0.89$ | $2.6 \pm 0.6$ |
| Medium (214–340, $n = 499$) | $27.18 \pm 0.16$ | $+14.68 \pm 0.63$ | $37.4 \pm 1.5$ | $54.26 \pm 1.33$ | $+35.88 \pm 1.96$ | $28.9 \pm 1.3$ |
| High (341–409, $n = 150$) | $34.63 \pm 0.35$ | $-24.98 \pm 1.45$ | $76.9 \pm 2.1$ | $57.76 \pm 1.87$ | $-38.52 \pm 2.49$ | $74.0 \pm 2.7$ |
| Extreme ($\geq 410$, $n = 99$) | $74.22 \pm 1.36$ | $-72.25 \pm 1.55$ | $96.0 \pm 0.0$ | $125.05 \pm 5.04$ | $-121.67 \pm 3.56$ | $96.3 \pm 1.5$ |