DOI: https://doi.org/10.15276/hait.08.2025.27
UDC 004.89

Estimation of task-specific manipulability scores for a robotic manipulator in vacuuming scenarios

Andrii Y. Medvid[1)]
ORCID: https://orcid.org/0009-0001-4128-4973; andrii.y.medvid@lpnu.ua
Vitaliy S. Yakovyna[1), 2)]
ORCID: https://orcid.org/0000-0003-0133-8591; vitaliy.s.yakovyna@lpnu.ua. Scopus Author ID: 6602569305
[1)] Lviv Polytechnic National University, 12, Bandera Str. Lviv, 79013, Ukraine
[2)]University of Warmia and Mazury in Olsztyn, 2, Oczapowskiego Str. Olsztyn, 10-719, Poland

ABSTRACT

Reliable vacuum-cleaning trajectories require selecting joint configurations from which a robotic arm can continue motion in task-relevant directions without encountering self-collision, joint-limit violations, or singularities. Classical manipulability measures describe dexterity using the Jacobian but do not account for the geometric constraints and directional motion patterns typical in floor-cleaning tasks. This work presents a data-driven method for estimating a task-specific manipulability score that reflects how easily a seven degrees-of-freedom robotic arm equipped with a floor-contact vacuum tool can continue motion in six primitive directions (forward, backward, left, right, clockwise rotation, counterclockwise rotation). A dataset of one hundred and seventy-six thousand and twenty valid joint configurations was generated by sweeping a four-dimensional grid of feasible end-effector poses, computing up to three inverse-kinematics solutions per pose, and simulating incremental movements with collision checking. Each configuration was assigned a directional score based on inverse-kinematics reachability, collision outcomes, and distance-dependent penalties.

A fully connected neural network was trained to regress six scores from the seven joint angles. The model achieved a denormalized Mean Absolute Error of approximately two point zero for translational directions and one point fifty-five to one point sixty for rotational directions (approximately eight percent of the full score range), while enabling extremely fast inference–around ninety-five thousand eight hundred and eighty evaluations per second on a consumer GPU.

By shifting the computationally expensive stage of collision checking and inverse-kinematics sampling to offline preprocessing, the method provides a lightweight surrogate for online motion planning. The learned score can help planners avoid unfavorable configurations and maintain consistent vacuuming trajectories. Limitations include the absence of environment-aware terms, stochastic inverse-kinematics sampling, and hand-tuned scoring parameters. Future work will focus on integrating obstacle information, improving label generation, and embedding the score into trajectory optimization frameworks for more robust real-world operation.

Keywords: Manipulability; robotic arm; vacuuming robot; neural networks; collision avoidance; trajectory planning; collision prediction

***For citation*: Medvid A. Y., Yakovyna V. S. "Estimation of task-specific manipulability scores for a robotic manipulator in vacuuming scenarios". *Herald of Advanced Information Technology*. 2025; Vol.8 No.4: 434–446. DOI: https://doi.org/10.15276/hait.08.2025.27**

INTRODUCTION

Service robotics is a rapidly growing field, with autonomous cleaning robots becoming increasingly ubiquitous in commercial and industrial environments. While mobile bases handle large-area navigation, the integration of robotic manipulators (robotic arms) allows for more complex cleaning tasks, such as vacuuming floors with vacuum installed at the wrist of the robotic arm.

Many modern robotic arms have more than 6 joints and more than 6 degrees-of-freedom (DOF) respectively. For a redundant manipulator, such as the 7-DOF xArm7, there are theoretically infinite joint configurations (inverse kinematics solutions) for a given end-effector pose [1]. This redundancy is advantageous as it allows the robot to avoid obstacles and self-collisions while maintaining the tool pose [2]. However, selecting the optimal configuration is computationally expensive. In vacuuming scenarios, the robot often needs to follow linear trajectories in Cartesian space on the floor. A poorly chosen initial configuration can lead to situations where the arm "locks up" (reaches a singularity) [1], collides with the mobile base (self-collision), or requires excessive joint velocities to continue the movement.

Classical manipulability indices, such as the Yoshikawa measure [3], describe the robot's ability to move in arbitrary directions but do not account for specific task constraints or collisions. Conversely, standard collision-checking libraries (e.g., Bullet [4]) provide collision safety information but lack the gradient information needed to guide the planner toward "more comfortable" states.

In this study, the data-driven approach proposed to estimate a task-specific manipulability score. The neural network was trained to map the robot's joint

state to a set of six scores representing the ease of movement in the cleaning plane: movement in positive x, negative x, positive y, negative y, positive yaw, negative yaw directions. This approach shifts the heavy computations burden of collision checking of bunches of neighboring vacuum states to an offline training phase, allowing for real-time query speeds suitable for online path planning.

RELATED WORK

Classical approaches to redundancy resolution and manipulator dexterity have been extensively studied in robotics. The Yoshikawa manipulability index [3] is a well-known measure that quantifies a robot arm's ability to move its end-effector in any direction, essentially measuring the volume of the velocity ellipsoid (and thus the distance to kinematic singularities). While useful, this index assumes unconstrained, isotropic motion and does not account for obstacles or task-specific directions. To address some of these limitations, Vahrenkamp et al. [5] proposed an extended manipulability analysis for humanoid robots that incorporate additional constraints such as joint limits and self-collision avoidance (through a "self-distance" metric) into the manipulability measure. This approach stores a representation of the robot's workspace capabilities and favors configurations that are both reachable and allow greater freedom of movement. Similarly, Ghosal [1] analyzed methods for redundancy resolution in robots (and drew parallels with the human arm), demonstrating how an extra degree of freedom can be exploited to optimize performance criteria – for example, using a redundant joint to maintain an isotropic end-effector velocity ellipse or to avoid singular configurations. These mathematically rigorous methods rely on the manipulator's Jacobian and predefined metrics; however, they do not inherently consider the full complexity of a robot's geometry in cluttered environments or the specific motion constraints of a given task.

To address these limitations, a growing research direction investigates learning-based manipulability modeling, where robots acquire manipulability profiles directly from data. These approaches aim to capture how experienced users or optimized controllers position the robot to maximize flexibility, strength, or task-space mobility.

A key early contribution in this domain is the work of Rozo et al. [6], who introduced the concept of manipulability transfer. Their method enables robots to learn and reproduce manipulability ellipsoids from expert demonstrations, framing manipulability as a geometric object lying on the manifold of symmetric positive definite (SPD) matrices. Using a tensor-based Gaussian mixture model that respects the curved structure of this manifold, they demonstrated how robots can adapt their posture to match a desired manipulability ellipsoid learned from human examples. This work established the foundation for treating manipulability not merely as a scalar index but as a structured, learnable descriptor of dexterity.

Building on this idea, Jaquier et al. proposed a series of geometry-aware approaches that further develop the learning and use of manipulability ellipsoids. In their control scheme for tracking manipulability profiles [7], the robot is tasked with following a time-varying desired ellipsoid either as its primary goal or as a secondary objective during motion execution. Their formulation explicitly incorporates the SPD manifold geometry, ensuring that both tracking errors and control updates remain consistent with the mathematical structure of manipulability ellipsoids. Experiments with redundant arms and real robots showed that this geometry-aware method outperforms prior Euclidean manipulations of manipulability data, which often distort ellipsoid shape and orientation.

This line of work culminated in a comprehensive framework by Jaquier et al. [8], which unifies manipulability learning, tracking, and transfer. Their method couples a tensor-based learning model with a geometry-aware controller, enabling robots to learn manipulability ellipsoids from demonstrations and subsequently reproduce or adapt them across a wide variety of platforms, including redundant manipulators, humanoids, and dual-arm systems. The ability to generalize manipulability profiles across embodiments and tasks highlights the potential of learned manipulability for versatile robot motion generation.

Beyond manipulability ellipsoids alone, Abu-Dakka et al. [9] extended the idea of learning geometry-based robot skills using datapoints represented as SPD matrices, including stiffness, inertia, and manipulability. Their probabilistic framework incorporates Kernelized Movement Primitives (KMPs) to learn and adapt SPD-valued trajectories, enabling robots to generalize manipulation skills encoded by geometric quantities to new situations. This approach demonstrates how manipulability can serve as one component within a broader set of learned geometric constraints, further emphasizing the importance of manifold-aware learning techniques.

While the previous works focus primarily on tracking or reproducing manipulability profiles, complementary research investigates neural network–based controllers that can modulate robot motion according to learned internal models. He and Yu [10] explored fuzzy neural network control for a flexible single-link manipulator, demonstrating how NN-based controllers can adjust motion to achieve desired dynamic properties (such as vibration suppression). Although not directly centered on manipulability, such neural controllers illustrate the broader applicability of learning-based approaches for shaping the robot's dynamic and kinematic characteristics – capabilities that can be integrated with manipulability-aware learning.

Overall, learning-based manipulability methods provide a promising alternative to classical Jacobian-based criteria. They allow robots to acquire dexterity profiles from demonstrations, adapt manipulability to task constraints, and incorporate the true geometric structure of manipulability ellipsoids. However, these approaches predominantly operate at the level of full ellipsoids in velocity/force space and focus on representing dexterity in an abstract geometric form, rather than directly predicting task-specific directional feasibility or operational comfort as required in vacuuming scenarios. Consequently, while these works offer powerful theoretical tools, they do not provide a direct mechanism for estimating how suitable a particular configuration is for a constrained floor-cleaning motion. The method proposed in this paper – estimating a custom directional manipulability score from joint angles – aims to fill this gap by combining the strengths of learning-based approaches with task-oriented geometric evaluation.

Beyond manipulability considerations, effective robot performance in constrained environments also depends on the ability to generate safe and efficient motions. Motion planning and collision avoidance is another critical aspect of robotic manipulation. A variety of traditional path planning algorithms exist to handle obstacles in high-dimensional configuration spaces. Probabilistic Roadmaps (PRM) [11] and Rapidly-Exploring Random Trees (RRT) are prominent examples of sampling-based planners: they construct a graph or tree of collision-free configurations via random sampling and then search for a feasible path through these structures. While these methods are probabilistically complete (they will find a path if one exists), their solutions are not necessarily optimal. Karaman and Frazzoli (2011) [11] addressed this by introducing asymptotically optimal variants, PRM* and RRT*, which ensure the path cost converges to the optimum as the number of samples grows.

In contrast, optimization-based planners such as CHOMP [12] and the Elastic Band approach [13] take a different route by deforming an initial trajectory to improve its safety and smoothness. CHOMP uses gradient-based optimization on a trajectory cost function (including obstacle avoidance via distance fields), while the Elastic Band method – implemented in modern variants like the Timed Elastic Band (TEB) [13] – connects path planning and control by treating the path as a “band” that can stretch or compress to avoid obstacles. These approaches can produce collision-free, smoother paths and even incorporate some kinematic constraints, but they tend to be computationally intensive for high-DOF (redundant) manipulators. Computing precise distance gradients or iterating over many samples in a 7-DOF arm’s configuration space is slow, which can be problematic for real-time applications.

To address the computational burden of collision checking in high-dimensional spaces, researchers have explored parallel and learning-based techniques. Pan and Manocha [14] leveraged GPU-based parallel collision detection to speed up the evaluation of many potential configurations simultaneously, achieving near real-time performance in motion planning scenarios. Beyond brute-force parallelization, recent efforts focus on learning proxy models that approximate the collision boundary. Das et al. introduced the Fastron algorithm [15], [16], an online learning model (based on a kernel perceptron) that actively samples configurations to label as collision or collision-free, updating its model incrementally. Fastron demonstrated significantly faster collision queries (several times faster than traditional geometric checking) by conservatively approximating the robot’s configuration-space obstacles and refining the model with new data points as needed. Building on this idea, Zhi et al. developed DiffCo [17], an auto-differentiable collision detector that not only classifies configurations by collision status (with the ability to handle multiple object collision classes) but also provides analytical gradients. This means a trajectory optimizer can get feedback on how to adjust the robot’s joints to avoid collisions, enabling smoother and more efficient avoidance maneuvers. Meanwhile, other state-of-the-art approaches have leveraged neural implicit representations of geometry for fast collision evaluation. Ortiz et al. [18] presented a real-time neural signed distance

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

field (SDF) model of the robot's environment, which allows instantaneous queries of distances to the nearest surfaces for any given robot pose. Joho et al. [19] went a step further by learning a neural implicit swept volume model for the manipulator – effectively capturing the volume the robot occupies as it moves – to enable ultra-fast collision checking for entire motion segments. These learning-based methods represent the cutting edge in balancing geometric fidelity and computational speed for motion planning.

Our previous work also explored learning-based collision avoidance in the context of redundant arms. In Medvid and Yakovyna [20], we employed Kolmogorov–Arnold Networks (KAN) to predict self-collisions for a 7-DOF robot. KAN is a neural network architecture derived from a theoretical decomposition of multivariate functions, which we used to map joint configurations to a binary outcome indicating whether any self-collision would occur. This approach proved effective in quickly flagging unsafe (self-colliding) arm postures, illustrating the feasibility of using learned models for internal collision checking. However, a binary collision classification provides limited insight for planning – it tells the planner which states are forbidden, but not how suitable a given free state is. It offers no measure of how close the robot is to a singularity or collision, nor how easily the arm can continue moving if it stays in that state. In other words, the KAN-based predictor could improve safety by avoiding collisions, but it could not guide the robot toward better configurations among the many that are collision-free.

Task-specific planning strategies for redundant manipulators have also been developed, recognizing that not all trajectories are equally desirable for a given job. Zhang and Wang [21] proposed a redundancy-based motion planning method that integrates task constraints directly into the planning process. In their framework, a 7-DOF manipulator's extra degree of freedom is used to satisfy secondary objectives like maintaining a particular end-effector orientation or pose constraint throughout the motion. For example, in a home-care scenario of transporting a bowl or spoon without spilling, the planner can ensure the end-effector remains level while still reaching the goal position. By considering such task constraints (including orientation alignment and workspace limitations), their method improved the efficiency and success rate of planning in complex tasks, as it guides the search toward configurations that inherently respect the task requirements. Wu et al. [22] tackled a related challenge for dual-arm robots operating in open environments. They developed a real-time collision avoidance scheme for two 7-DOF arms working together, which coordinates the arms' motions by exploiting redundancy in each arm to avoid both self-collision and mutual collisions. This approach allows a dual-manipulator system to react on-the-fly and reconfigure its joints to bypass obstacles or avoid the arms interfering with each other, all while fulfilling the task (such as handling an object collaboratively). Both of these works highlight the importance of using redundancy not just for avoiding collisions, but also for maintaining task-specific criteria during motion execution.

Another line of research has applied reinforcement learning (RL) to complex manipulation tasks with redundant robots [23], [24]. Deep reinforcement learning can, in principle, learn control policies that map sensor or state inputs to joint actions, optimizing directly for a task-specific reward function. For instance, Cui et al. [23] introduced a task-adaptive deep RL framework for a dual-arm manipulator, where the system learned to adjust its strategy (including coordination between arms and even impedance parameters) to accomplish coordinated tasks like pick-and-place with heavy or delicate objects. Such an RL-based approach can handle continuous control and discover non-intuitive solutions by trial and error. Lillicrap et al. [24], in a foundational work, demonstrated that deep neural networks combined with an actor-critic architecture (specifically the Deep Deterministic Policy Gradient algorithm) can achieve continuous control on high-dimensional systems, effectively learning to operate a simulated manipulator arm directly from reward feedback. While these RL approaches show promise in leveraging redundancy and mastering complex behaviors, they come with certain drawbacks for safety-critical or real-world deployment. Policies learned end-to-end are often difficult to interpret, and it is challenging to guarantee that the robot will always operate within safe limits, especially when faced with novel situations outside the training distribution. In contrast, classical planners (potentially augmented with learned models or heuristics) can enforce constraints and safety checks explicitly, making their behavior more predictable. Thus, a middle-ground approach is to use learning to assist planning – for example, by providing a learned metric or value function – rather than to replace the planner entirely.

A line of research closely related to our work focuses on learning feasibility or heuristic functions to accelerate task-and-motion planning (TAMP).

Wells et al. [25] train a classifier that predicts whether a candidate task plan is likely to admit a feasible motion instantiation and use this prediction to bias TAMP search. Kim et al. [26] introduce a score-space representation and learn to guide TAMP using data-driven constraints that prune the search space. More recently, Yang et al. [27] propose PIGINet, a Transformer-based model that predicts plan skeleton feasibility and significantly reduces planning time in long-horizon rearrangement tasks. In all these approaches, learning is used to assess global plan feasibility or cost at the level of symbolic or action sequences.

In parallel, several works leverage neural networks to approximate geometric properties relevant for collision checking and contact handling. Zesch et al. [28] propose neural collision detection and neural collision fields that learn continuous fields encoding collision information for deformable or triangle-based geometry, enabling fast collision queries without explicit spatial data structures. Pulikottil et al. [29] introduce a robotic ease of disassembly metric (Re-DiM) tailored to human–robot cooperative disassembly of products such as vacuum cleaners, highlighting the importance of task-specific performance scores in robotics. Our work is complementary: we learn a task-specific manipulability score at the level of individual joint configurations of a 7-DOF arm in floor-vacuuming scenarios, aggregating collision outcomes and joint-limit information into a scalar measure that can be used as a low-level heuristic inside different motion-planning pipelines.

PROBLEM STATEMENT

Reliable vacuum-cleaning trajectories require selecting joint configurations from which the robotic arm can continue motion in task-relevant directions without encountering self-collision, joint-limit violations, or kinematic singularities. Classical manipulability measures and collision-checking techniques do not directly indicate how "comfortable" a particular configuration is for continuing linear or rotational movement along the floor.

The aim of this research is to develop a fast method for estimating a task-specific manipulability score that quantifies how easily the robot can continue motion from a given joint configuration in six vacuuming-relevant directions (+x, −x, +y, −y, +yaw, −yaw).

To achieve this aim, the following research objectives were formulated:

1) to design a scoring function that evaluates motion feasibility based on inverse kinematics reachability, collision outcomes, and distance-dependent penalties;

2) to generate a large dataset of feasible arm configurations labelled with the proposed directional scores;

3) to train a neural-network regression model capable of estimating these scores from the robot's joint angles;

4) to evaluate the estimation accuracy and computational efficiency of the model for potential integration into motion-planning pipelines.

MATERIALS AND METHODS

Robotic Platform and Vacuuming Task Setup

The experiments were conducted using the xArm7 robotic manipulator (UFactory) mounted on a commercial cleaning robot developed by Somatic Holdings LTD.

The robotic arm operates a custom vacuum-cleaning end-effector designed to maintain contact with the floor during the cleaning motion. Joint limits of the manipulator could be seen at Table 1.

Table 1. **Joint limits of xArm7 arm**

Joint Number	Min Angle	Max Angle
0	-360°	360°
1	-118°	120°
2	-360°	360°
3	-11°	225°
4	-360°	360°
5	-97°	180°
6	-360°	360°

***Source*: compiled by the [30]**

To ensure that the vacuum head remains on the floor, only arm configurations corresponding to feasible end-effector poses at floor height were included in the dataset. An example of a robot state with a vacuum head on the floor could be seen on Fig. 1.

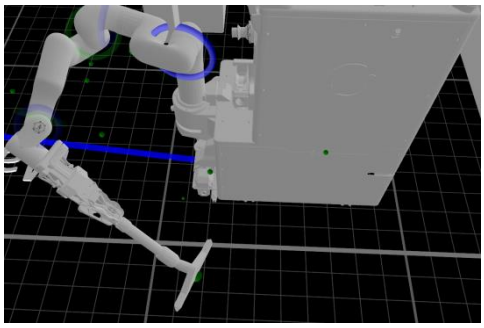

Fig. 1. **Robot with vacuum tool installed**
***Source*: screenshot from Somatic Holdings LTD software**

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

Dataset Generation Procedure

A custom C++ data generation pipeline was developed to compute a manipulability score for a bunch of robot configurations.

The pipeline consisted of:

1. A discretized 4D grid of feasible end-effector poses (x, y, z, Θ_{yaw}).
2. Inverse kinematics (IK) search for valid arm configurations.
3. Collision checking (Bullet physics).
4. Motion simulation in six task-relevant directions: +x, -x, +y, -y, +yaw, -yaw.

Each grid cell represented a possible vacuum head position on the floor. The grid resolution and index ranges are listed in Table 2.

Table 2. **End effector grid parameters**

Axis	Step	Index Range	Physical Range
x	0.04 m	-14…40	-0.56…1.60 m
y	0.04 m	-30…30	-1.20…1.20 m
z	0.015 m	-1…2	-0.015…0.03 m
yaw	20°	-8…9	-160°…180°

***Source*: compiled by the authors**

The end effector in this case means the origin with zero point at the center of the bottom of the vacuum head; Z axis directed upwards perpendicularly to the floor; X axis directed parallel to the floor as a projection of the vacuum pipe on the floor; and the Y axis directed left along vacuum head. At Fig. 2 the end effector origin shown with red cylinder representing X axis, green cylinder representing Y axis and blue cylinder representing Z axis.

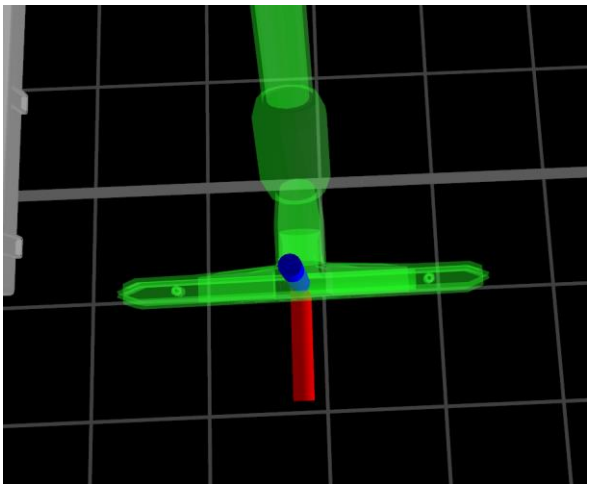

Fig. 2. **Vacuum head end effector origin**
***Source*: screenshot from Somatic Holdings LTD software**

The physical range of the end effector origin was selected based on the physical reachability of the tool for a specific robot model. A few additional steps in z coordinates were added because of not ideally flat floors and some errors of the physical model of the robotic arm. So sometimes the robot operates with vacuum head positioned not at theoretical 0 meters level.

The conversion from the grid indices to end effector position in the robot base calculated with formula (1):

$$EE_{pos} = \{i_x \cdot step_x, i_y \cdot step_y, i_z \cdot step_z, \}, \quad (1)$$

where EE_{pos} is a three-dimensional vector defining position of an end effector in the robot base coordinate system; i_x, i_y, i_z are x, y and z indices in grid; $step_x$, $step_y$, $step_z$, - are x, y and z steps in the grid.

The conversion from the yaw index to end effector rotation in the robot base calculated with formula (2):

$$EE_{yaw} = i_{yaw} \cdot step_{yaw}, \quad (2)$$

where EE_{yaw} is a rotation angle along Z axis; i_{yaw} is a yaw index in the grid; $step_{yaw}$ is a yaw step in the grid.

Inverse Kinematics Sampling

For each grid cell end effector at first the end effector origin is calculated. Then using IK algorithm arm states, with corresponding end effector origins generated. To reduce computational cost, a maximum of 3 IK solutions per grid position was retained. Note that for some end effector origins there might not exist corresponding arm states.

In this case these positions are ignored and the data generation process proceeds.

Manipulability Score Evaluation

In this work, we use a task-specific manipulability score – a scalar value estimating how easily the robot can continue motion from a given joint configuration in several vacuuming-relevant directions. The score is a practical heuristic derived from inverse kinematics feasibility and collision-checking results, intended to help a motion planner prefer configurations that are more likely to allow longer, collision-free movements without hitting joint limits or singularities.

For each arm state (configuration) generated on previous steps (let's call it a base state in this subsection) the ability of the manipulator to move in six directions $(x^+, x^-, y^+, y^-, yaw^+, yaw^-)$ was evaluated.

Each direction was tested using a sequence of incremental displacements:

• linear directions: 10 steps × 2 cm;

• rotational directions: 6 steps × 5°.

For each step:

• a new target pose was calculated considering current step number and direction;

• arm configurations for a new origin calculated with IK;

• if we get more than 8 configurations, then only 8 random configurations kept to speed up data generation;

• each candidate configuration was checked for collisions;

• a partial score was accumulated to get a total score for current base state and direction.

Scoring formula

The final manipulability score for a base state and a direction $d \in \{x^+, x^-, y^+, y^-, yaw^+, yaw^-\}$ calculates with formula (3):

$$S_d = S_{coll} + \sum_{st=1}^{steps_d} score_{D_{st}}, \quad (3)$$

where S_{coll} is a penalty if the original state colliding (-8 used in experiments); $steps_d$ is a number of steps (10 for translation, 6 for rotation) for current direction; st is a current step of displacement from the base state; and $score_{D_{st}}$ is a score for st -th displacement in current direction, calculated by formula 4:

$$score_{D_{st}} = \begin{cases} \frac{-1}{dist\_weight}, if\ |IK\ solutions| = 0 \\ \sum_{sol=1}^{N_{st}} \frac{score_{st,sol}}{dist\_weight}, if\ |IK\ solutions| > 0 \end{cases}, \quad (4)$$

where N_{st} is a number of IK solutions for a current pose (base pose + st steps in current direction), limited by maximum 8 solutions as it described in *Manipulability Score Evaluation* subsection; also, it worth noting that if no IK solutions found then the arm possibly can't reach the position so we have a penalty '-1' in this case; sol is a current IK solution index; $score_{st,sol}$ is a score for current IK solution of a current step in a current direction; it equals 1 if there is no collision for this configuration and equals -0.18 if there is at least one collision; $dist\_weight$ is a multiplier that reduces the score based on distance from starting origin; it's calculated with formula 5:

$$dist\_weight = \frac{1}{1+\alpha \cdot st}, \quad (5)$$

where α is a distance penalty factor (0.2 used in experiments) to make closer states more important, than distant ones.

Code snipped of base states score calculation could be seen at Fig. 3.

```
for (size_t step = 1; step <= stepsCount; step++) {
    Pose target = basePose.Translate(step * posStep)
                          .Rotate(step * yawStep);
    auto ik = solver.InverseKinematicsAll(target, seed, limit);
    ClampToMaxSolutions(ik, 8);

    if (ik.empty()) {
        score += (-1.0) / (1 + 0.2 * step);
        continue;
    }

    for (auto& sol : ik) {
        bool coll = collisionEngine.IsColliding(sol);
        score += (coll ? -0.18 : 1.0) / (1 + 0.2 * step);
    }
}
```

Fig. 3. **Scoring formula code snippet**

***Source*: compiled by the authors**

Based on scoring formula the minimum possible score for translational directions equals:

$$SX_{min} = SY_{min} = -0.18 \cdot 8 \cdot \sum_{step=1}^{10} \frac{1}{1+0.2 \cdot step} \approx -7.45.$$

The minimum possible score for rotational directions equals:

$$SYaw_{min} = -0.18 \cdot 8 \cdot \sum_{step=1}^{6} \frac{1}{1+0.2 \cdot step} \approx -5.3.$$

The total minimum possible score equals:

$$Total_{min} = 2 \cdot (SX_{min} + SY_{min} + SYaw_{min}) \approx -25.5$$

At the same time maximum possible scores equlas:

$$SX_{max} = SY_{max} = 1.0 \cdot 8 \cdot \sum_{step=1}^{10} \frac{1}{1+0.2 \cdot step} \approx 41.4.$$

$$SYaw_{max} = 1.0 \cdot 8 \cdot \sum_{step=1}^{6} \frac{1}{1+0.2 \cdot step} \approx 29.46.$$

$$Total_{max} = 2 \cdot (SX_{max} + SY_{max} + SYaw_{max}) \approx 224.5.$$

JSON dataset structure

After calculating manipulability scores for each state all the data saved in a json file which has some metainformation and the main data structured as follows: a field "States" has a dictionary value and each entry of this dictionary has a robotic arm state as a key (represented by string consisting of 7 floats separated by whitespace – angles of arm joints) and a manipulability scores as a value (represented by string consisting of 6 floats separated by whitespace - $\{x^+, x^-, y^+, y^-, yaw^+, yaw^-\}$ scores calculated by the presented algorithm). A total of 176,020 valid configurations with corresponding scores were collected.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

Neural Network Model

The mapping $\mathbb{R}^7 \rightarrow \mathbb{R}^6$ was learned using a fully connected Multi-Layer Perceptron (MLP). The architecture of the neural network layers shown at Fig. 4.

```
MLP([
  Linear(7 → 128), ReLU(),
  Linear(128 → 256), ReLU(),
  Linear(256 → 128), ReLU(),
  Linear(128 → 64), ReLU(),
  Linear(64 → 6)
])
```

Fig. 4. **Architecture of NN snippet**
***Source*: compiled by the authors**

Activation function used – ReLU, optimizer – Adam with starting learning rate 0.001 and increasing with cosine annealing to 0.000001 during training process. Loss used for training is mean squared error (MSE), but also mean absolute error (MAE) used for final model evaluation.

Normalization

Mean and standard deviation were computed on the training set. Loss was computed in normalized space; MAE/MSE were reported in denormalized form.

Training procedure

Epochs – 1500, batch size – 128, validation split – 20 %.

TensorBoard was used to log:

- train/val MSE;
- train/val MAE;
- MAE per direction.

RESULTS

The following metrics were computed:

- MSE (denormalized);
- MAE (denormalized);
- per-direction MAE (6 curves train + 6 curves validation).

The results of training can be seen at Figures 5-12.

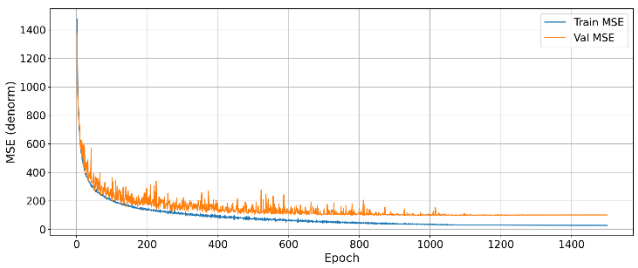

Fig. 5. **Total MSE**
***Source*: compiled by the authors with Matplotlib**

Fig. 5 shows that total mean square error dropped to nearly 27.94 on train data and to 101.22 on a validation data.

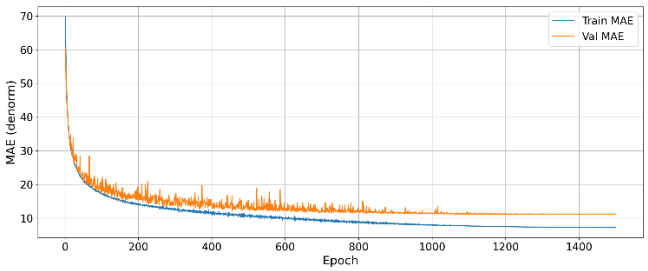

Fig. 6. **Total MAE**
***Source*: compiled by the authors with Matplotlib**

At the same time at Fig. 6 can be seen that the mean absolute error dropped to nearly 7.25 on train data and to 11.15 on a validation data.

The training results divided by different directions shown at Fig.7, Fig 8, Fig 9, Fig 10, Fig 11 and Fig 12.

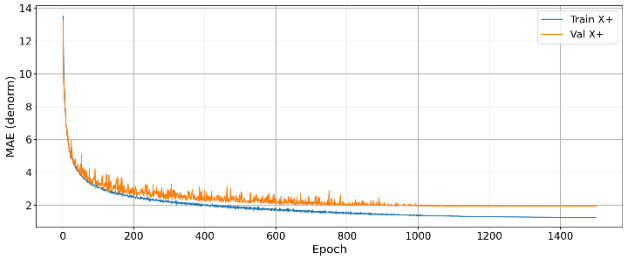

Fig. 7. **X^+ direction MAE**
***Source*: compiled by the authors with Matplotlib**

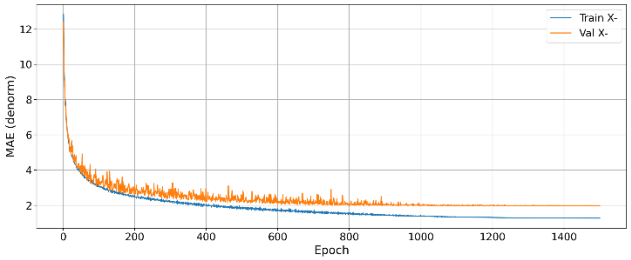

Fig. 8. **X^- direction MAE**
***Source*: compiled by the authors with Matplotlib**

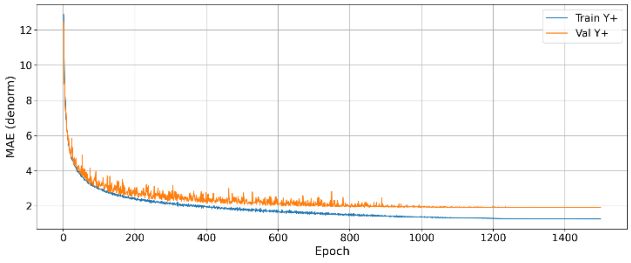

Fig. 9. **Y^+ direction MAE**
***Source*: compiled by the authors with Matplotlib**

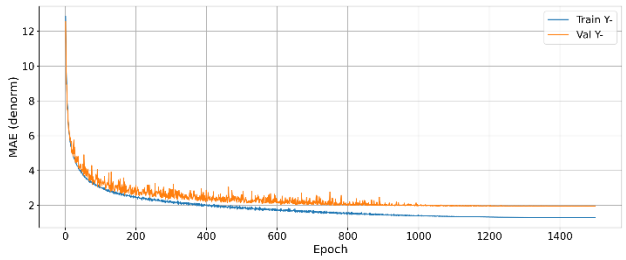

Fig. 10. **Y⁻ direction MAE**
***Source*: compiled by the authors with Matplotlib**

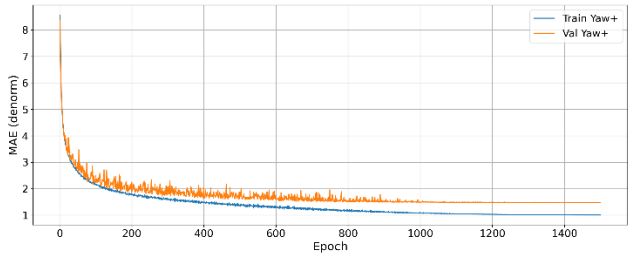

Fig. 11. **Yaw⁺ direction MAE**
***Source*: compiled by the authors with Matplotlib**

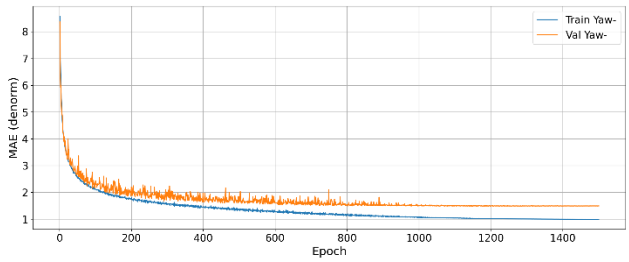

Fig. 12. **Yaw⁻ direction MAE**
***Source*: compiled by the authors with Matplotlib**

The results on Fig.7, Fig 8, Fig 9, Fig 10, Fig. 11 and Fig. 12 shows how mean absolute error on train data drops nearly to 1.33-1.35 for $\{x^+, x^-, y^+, y^-\}$ directions and nearly to 1.0 for $\{yaw^+, yaw^-\}$ directions. At the same time mean absolute error on validation data drops nearly to 2.0 for $\{x^+, x^-, y^+, y^-\}$ directions and to 1.55-1.6 for $\{yaw^+, yaw^-\}$ directions.

The total training time was 2203 seconds for 1500 epochs and 140,816 train cases on a GeForce RTX 3050 Mobile GPU. It means that on average trained neural network calculates scores for approximately 95,880 arm configurations per second.

DISCUSSION

The experimental results demonstrate that the proposed neural model is able to approximate the custom manipulability score with sufficiently low estimation error across all six evaluated motion directions. To interpret the achieved accuracy in context, it is useful to compare the observed MAE values with the theoretical score ranges derived in the “Materials and Methods” section. For translational directions, the score spans approximately from −7.45 to 41.4, while rotational directions cover a range from around −5.3 to 29.46. Considering that the learned model on validation data reaches MAE ≈ 2.0 for translational axes and ≈1.55-1.6 for rotational axes, the error constitutes roughly 8% of the effective value range. This indicates that although the network does not perfectly reproduce the exact numerical scores, it captures the relative ordering of “comfortable” and “uncomfortable” configurations well enough to be applied as a heuristic in planning tasks.

An important advantage of the approach lies in decoupling heavy geometry-based computations from online motion planning. Collision checking, IK sampling, and multi-step evaluation of feasible moves constitute the costliest parts of the scoring procedure. By shifting them to an offline phase and learning a compact parametric regressor over joint angles, the planner is able to evaluate manipulability almost instantaneously during trajectory generation. This can help mitigate common issues in vacuuming scenarios, such as selecting configurations that bring the arm close to self-collision, joint limits, or kinematic singularities.

However, several limitations must be acknowledged. First, the dataset is constructed under assumptions of a static environment and does not model interactions with external obstacles, furniture, or varying floor conditions. Thus, the learned score reflects only self-collision and robot–base geometry, not the full operational environment. Second, the quality of the labels depends on stochastic IK sampling: if the solver fails to find a valid configuration, the score may underrepresent true reachability. Third, the scoring formulation includes hand-tuned parameters (step counts, penalty weights, distance decay), which influence the learned function and may not transfer optimally to all task settings.

Finally, the model estimates directional manipulability scores but does not provide gradients with respect to joint angles. While this is not strictly required for sampling-based planners, gradient information could enable direct incorporation of the score into optimization-based motion planning. Future work may therefore explore differentiable approximations of IK feasibility or hybrid methods combining neural estimations with local Jacobian-based metrics to obtain a more complete manipulability descriptor.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

CONCLUSION

This work introduced a data-driven method for estimating task-specific manipulability scores for a 7-DOF robotic arm performing vacuum cleaning using a floor-contact tool. A large dataset of feasible arm configurations was generated and annotated using a multi-step IK- and collision-based scoring procedure. A multilayer perceptron was trained to map joint configurations to six directional manipulability values corresponding to the robot's ability to move in ±x, ±y, and ±yaw directions without encountering collisions or kinematic limitations.

The trained model demonstrates low mean absolute error relative to the full theoretical score range, enabling reliable approximation of local motion feasibility. Since inference is computationally inexpensive, the model is well suited for integration into real-time planning pipelines, where it can help avoid unfavorable arm postures and improve the robustness of vacuuming trajectories.

Nevertheless, the approach remains limited by its reliance on static, environment-free datasets and the inherent imperfections of stochastic IK sampling. Extending the method to incorporate external obstacles, dynamic constraints, and more expressive geometric representations is a promising direction for future research. Integrating the learned manipulability score with trajectory optimization or reinforcement learning frameworks could further enhance the autonomy and reliability of service robots operating in constrained environments.

CODE AVAILABILITY

The source code for the neural network training described in this article is available in the public repository: https://github.com/amedvid/VacuumingManipulabilityPrediction. The source code of training data generation is not publicly available due to NDS, but described in the "Dataset Generation Procedure" subsection.

ACKNOWLEDGEMENTS

The authors would like to express their deep gratitude to Somatic Holdings LTD, whose codebase, the simulation environment and the infrastructure greatly facilitated the development of the method presented in this paper.

DECLARATION ON GENERATIVE AI

During the preparation of this work generative AI tools (ChatGPT 5, 5.1, Gemini 2.5, 3) have been used for code writing, grammar checks, text polishing. The authors reviewed and edited all AI-generated content and took the responsibility for the final content of this publication.

REFERENCES

1. Ghosal, A. "Resolution of redundancy in robots and in a human arm". *Mechanism and Machine Theory*. 2018; 125: 126–136, https://www.scopus.com/pages/publications/85044344469. DOI: https://doi.org/10.1016/j.mechmachtheory.2017.12.008.

2. Scoccia, C., Palmieri, G., Palpacelli, M. C. & Callegari, M. "A collision avoidance strategy for redundant manipulators in dynamically variable environments: On-line perturbations of off-line generated trajectories". *Machines*. 2021; 9 (2): 30, https://www.scopus.com/pages/publications/85100733803. DOI: https://doi.org/10.3390/machines9020030.

3. Patel, S. & Sobh, T. "Manipulator performance measures – A comprehensive literature survey". *Journal of Intelligent & Robotic System*. 2015; 77 (3–4): 547–570, https://www.scopus.com/pages/publications/84924223466. DOI: https://doi.org/10.1007/s10846-014-0024-y.

4. "Bullet collision detection & physics library". *Bullet documentation*. – Available from: https://pybullet.org/Bullet/BulletFull/index.html.

5. Vahrenkamp, N., Asfour, T., Metta, G., Sandini, G. & Dillmann, R. "Manipulability analysis". *Proceedings of the 12th IEEE-RAS International Conference on Humanoid Robots*. 2012. p. 568–573, https://www.scopus.com/pages/publications/84891106700.
DOI: https://doi.org/10.1109/humanoids.2012.6651576.

6. Rozo, L., Jaquier, N., Calinon, S. & Caldwell, D. G. "Learning manipulability ellipsoids for task compatibility in robot manipulation". *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017. p. 3183–3189, https://www.scopus.com/pages/publications/85041965515. DOI: https://doi.org/10.1109/iros.2017.8206150.

7. Jaquier, N., Rozo, L., Caldwell, D. & Calinon, S. “Geometry-aware tracking of manipulability ellipsoids”. *Robotics: Science and Systems XIV*. 2018, https://www.scopus.com/pages/publications/85089870613. DOI: https://doi.org/10.15607/rss.2018.xiv.027.

8. Jaquier, N., Rozo, L., Caldwell, D. G. & Calinon, S. “Geometry-aware manipulability learning, tracking, and transfer”. *The International Journal of Robotics Research.* 2020; 40 (2-3): 624–650, https://www.scopus.com/pages/publications/85089861474. DOI: https://doi.org/10.1177/0278364920946815.

9. Abu-Dakka, F. J., Huang, Y., Silvério, J., & Kyrki, V. “A probabilistic framework for learning geometry-based robot manipulation skills”. Robotics and Autonomous Systems. 2021; 141: 10376, https://www.scopus.com/pages/publications/85104929959. DOI: https://doi.org/10.1016/j.robot.2021.103761.

10. Sun, C., Gao, H., He, W. & Yu, Y. “Fuzzy neural network control of a flexible robotic manipulator using assumed mode method”. *IEEE Transactions on Neural Networks and Learning Systems*. 2018; 29 (11): 5214–5227, https://www.scopus.com/pages/publications/85041527429.
DOI: https://doi.org/10.1109/tnnls.2017.2743103.

11. Karaman, S. & Frazzoli, E. “Sampling-based algorithms for optimal motion planning”. *The International Journal of Robotics Research.* 2011; 30 (7): 846–894, https://www.scopus.com/pages/publications/80052218929. DOI: https://doi.org/10.1177/0278364911406761.

12. Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A. & Srinivasa, S. S. “CHOMP: Covariant Hamiltonian optimization for motion planning”. *The International Journal of Robotics Research.* 2013; 32(9-10): 1164–1193, https://www.scopus.com/pages/publications/84884264343. DOI: https://doi.org/10.1177/0278364913488805.

13. Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F. & Bertram, T. “Efficient trajectory optimization using a sparse model”. *Proceedings of the 6th European Conference on Mobile Robots (ECMR).* 2013; 138–143, https://www.scopus.com/pages/publications/84893335333.
DOI: https://doi.org/10.1109/ECMR.2013.6698833.

14. Pan, J. & Manocha, D. “GPU-based parallel collision detection for real-time motion planning”. *Springer Tracts in Advanced Robotics*. 2010. p 211–228, https://www.scopus.com/pages/publications/78650146189. DOI: https://doi.org/10.1007/978-3-642-17452-0_13.

15. Das, N., Gupta, N. & Yip, M. “Fastron: An online learning-based model and active learning strategy for proxy collision detection”. *Proceedings of the 1st Conference on Robot Learning (CoRL).* 2017. DOI: https://doi.org/10.48550/arXiv.1709.02316.

16. Das, N. & Yip, M. “Learning-based proxy collision detection for robot motion planning applications”. *IEEE Transactions on Robotics*. 2020; 36 (4): 1096–1114, https://www.scopus.com/pages/publications/85081315440. DOI: https://doi.org/10.1109/tro.2020.2974094.

17. Zhi, Y., Das, N. & Yip, M. “DiffCo: Auto-differentiable proxy collision detection with multi-class labels for safety-aware trajectory optimization”. *arXiv preprint*. 2021. DOI: https://doi.org/10.48550/arXiv.2102.07413.

18. Ortiz, J., Clegg, A., Dong, J., Sucar, E., Novotny, D., Zollhoefer, M. & Mukadam, M. “ISDF: Real-time neural signed distance fields for robot perception”. *Robotics: Science and Systems XVIII*. 2022, https://www.scopus.com/pages/publications/85177422862. DOI: https://doi.org/10.15607/rss.2022.xviii.012.

19. Joho, D., Schwinn, J. & Safronov, K. “Neural implicit swept volume models for fast collision detection”. *IEEE International Conference on Robotics and Automation (ICRA).* 2024; 15402–15408, https://www.scopus.com/pages/publications/85202452303.
DOI: https://doi.org/10.1109/icra57147.2024.10611687.

20. Medvid, A. & Yakovyna, V. “Robot self collision prediction using Kolmogorov-Arnold networks”. *Collection of scientific works of the XXIII International scientific conference ‘Neural network technologies and their application – (NNTA).* 2024. p 11–16, http://www.dgma.donetsk.ua/docs/news/2024/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA%20NNTA-2024.pdf#page=11.

21. Zhang, Y. & Wang, H. “Redundancy-based motion planning with task constraints for robot manipulators”. *Sensors.* 2025; 25 (6): 1900, https://www.scopus.com/pages/publications/105001107282. DOI: https://doi.org/10.3390/s25061900.

22. Wu, Y., Jia, X., Li, T. & Liu, J. “A real-time collision avoidance method for redundant dual-arm robots in an open operational environment”. *Robotics and Computer-Integrated Manufacturing*. 2025; 92: 102894. DOI: https://doi.org/10.1016/j.rcim.2024.102894.

23. Cui, Y., Xu, Z., Zhong, L., Xu, P., Shen, Y. & Tang, Q. “A task-adaptive deep reinforcement learning framework for dual-arm robot manipulation”. *IEEE Transactions on Automation Science and Engineering*. 2025; 22: 466–479, https://www.scopus.com/pages/publications/85182922771. DOI: https://doi.org/10.1109/tase.2024.3352584.

24. Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. & Wierstra, D. “Continuous control with deep reinforcement learning”. *arXiv preprint*. 2015. DOI: https://doi.org/10.48550/arXiv.1509.02971.

25. Wells, A. M., Dantam, N. T., Shrivastava, A. & Kavraki, L. E. “Learning feasibility for task and motion planning in tabletop environments”. *IEEE Robotics and Automation Letters*. 2019; 4 (2): 1255–1262, https://www.scopus.com/pages/publications/85065930131. DOI: https://doi.org/10.1109/lra.2019.2894861.

26. Kim, B., Wang, Z., Kaelbling, L. P. & Lozano-Pérez, T. “Learning to guide task and motion planning using score-space representation”. *The International Journal of Robotics Research*. 2019; 38 (7): 793–812, https://www.scopus.com/pages/publications/85067058496.
DOI: https://doi.org/10.1177/0278364919848837.

27. Yang, Z., Garrett, C., Lozano-Perez, T., Kaelbling, L. & Fox, D. “Sequence-based plan feasibility prediction for efficient task and motion planning”. *Robotics: Science and Systems XIX*. 2023. DOI: https://doi.org/10.15607/rss.2023.xix.061.

28. Zesch, R. S., Witemeyer, B. R., Xiong, Z., Levin, D. I. W. & Sueda, S. “Neural collision detection for deformable objects”. CoRR, abs/2202.02309. 2022. DOI: https://doi.org/10.48550/arXiv.2202.02309.

29. Pulikottil, T., Boix Rodríguez, N. & Peeters, J. R. “Robotic ease of disassembly metric (re-dim) for human robot cooperative disassembly: A case study for a vacuum cleaner”. *Procedia CIRP*. 2024; 122: 677–682, https://www.scopus.com/pages/publications/85193542045.
DOI: https://doi.org/10.1016/j.procir.2024.02.021.

30. “The difference between UFACTORY xArm5, UFACTORY xArm6 and UFACTORY xArm7”. – Available from: https://help.ufactory.cc/en/articles/4491842.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article
Author Vitaliy S. Yakovyna is a member of the Editorial Board of this journal. This role had no influence on the peer review process or editorial decision regarding this manuscript

Received 30.09.2025
Received after revision 18.11.2025
Accepted 28.11.2025

DOI: https://doi.org/10.15276/hait.08.2025.27
УДК 004.89

Оцінювання спеціалізованих показників маніпульованості роботизованого маніпулятора в сценаріях пилососання

Медвідь Андрій Ярославович[1)]
ORCID: https://orcid.org/0009-0001-4128-4973; andrii.y.medvid@lpnu.ua
Яковина Віталій Степанович[1),2)]
ORCID: https://orcid.org/0000-0003-0133-8591; vitaliy.s.yakovyna@lpnu.ua
[1)] Національний університет «Львівська політехніка», вул. С. Бандери, 12. Львів, 79013, Україна
[2)] Вармінсько-Мазурський університет в Ольштині, вул. Очаповського, 2. Ольштин, 10-719, Польща

АНОТАЦІЯ

Надійне планування руху роботизованої руки під час вакуумного прибирання потребує вибору таких конфігурацій суглобів, з яких маніпулятор може продовжувати рух у потрібних напрямках без самоколізій, виходу за межі обмежень

суглобів чи потрапляння у сингулярності. Класичні показники маніпульованості описують здатність маніпулятора рухати енд-ефектор у різних напрямках, однак не враховують геометричні обмеження та характерні для прибирання напрямні рухи.

У роботі запропоновано метод оцінювання спеціалізованого напрямного показника маніпульованості для семиступеневої роботизованої руки з підлоговою вакуумною насадкою. Показник відображає, наскільки легко маніпулятор може продовжувати рух у шести елементарних напрямках (вперед, назад, вліво, вправо, поворот за годинниковою стрілкою, поворот проти годинникової стрілки). Було створено набір даних із ста сімдесяти шести тисяч двадцяти коректних конфігурацій, отриманих шляхом аналізу чотиривимірної ґратки поз енд-ефектора, обчислення до трьох розв'язків оберненої кінематики на кожну позу та покрокового моделювання локального зсуву з перевіркою на колізії. Для кожної конфігурації сформовано шість напрямних показників, що враховують досяжність інверсної кінематики, зіткнення та відстань від вихідної пози.

Для апроксимації цих значень було навчено повнозв'язну нейронну мережу. Модель досягла середньої абсолютної похибки приблизно дві цілих нуль десятих для трансляційних і від однієї цілої п'ятдесят п'ять сотих до однієї цілої шести десятих для обертальних напрямків (близько восьми процентів від повного діапазону) та забезпечила дуже швидку інференцію – близько дев'яноста п'яти тисяч восьмисот вісімдесяти конфігурацій/с.

Підхід переносить трудомісткі обчислення у офлайн-етап і забезпечує легку апроксимацію для онлайн-планування. Метод обмежений відсутністю інформації про зовнішні перешкоди та стохастичністю семплінгу інверсної кінематики. Подальші дослідження мають бути спрямовані на інтеграцію середовищних обмежень і поєднання показника з оптимізаційним плануванням траєкторій.

Ключові слова: маніпульованість; роботизована рука; робот-пилосос; нейронні мережі; уникнення колізій; планування траєкторій; передбачення колізій

ABOUT THE AUTHORS

Andrii Y. Medvid – PhD student, Artificial Intelligence Systems Department. Lviv Polytechnic National University, 12, Bandera Str., Lviv, 79013, Ukraine
ORCID: https://orcid.org/0009-0001-4128-4973; andrii.y.medvid@lpnu.ua
***Research field*:** Robotics, Motion Planning, Imitation Learning

Медвідь Андрій Ярославович – аспірант кафедри Систем Штучного Інтелекту. Національний університет «Львівська політехніка», вул. С. Бандери, 12, Львів, 79013, Україна

Vitaliy S. Yakovyna – Doctor of Engineering Sciences, Professor, Artificial Intelligence Systems Department. Lviv Polytechnic National University, 12, Bandera Str., Lviv, 79013, Ukraine
ORCID: https://orcid.org/0000-0003-0133-8591; vitaliy.s.yakovyna@lpnu.ua. Scopus Author ID: 6602569305
***Research field*:** Software Reliability, Machine Learning, Ensemble Learning, Computational Intelligence

Яковина Віталій Степанович – Доктор технічних наук, професор кафедри Систем Штучного Інтелекту. Національний університет «Львівська політехніка», вул. С. Бандери, 12, Львів, 79013, Україна.