

Placement Optimization of Flexible Proximity Sensors for Human-Robot Collaboration

Hongrui Wu¹, Xin Wang², *Member, IEEE*, and Yihao Li

Abstract—Flexible proximity sensors mounted on robot arms boost obstacle detection in human-robot collaboration (HRC). However, most of the flexible sensor placements lack further analysis to exploit the flexibility, leading to an inefficient and overpriced sensing system. In this letter, we propose a systematic method to optimize the placement of the flexible proximity sensor for HRC. To prepare for the optimization, the geometric model of a flexible sensor is established. Based on a global search algorithm, we obtain the optimized sensor placement with a sufficient detection ability and a minimum number of sensors. An experiment was conducted to verify the reliability of the method. The comparison between the optimized placement results and the conventional ones indicates that the proposed method could achieve better detection performance with much fewer sensors. This method also takes the flexibility into account by customizing the placement for different tasks. Emphasizing the merits of flexibility, this letter provides new insights into the application and the design of the flexible sensor for HRC.

Index Terms—Human detection and tracking, human-robot collaboration, sensor placement, soft sensors and actuators.

I. INTRODUCTION

HUMAN-ROBOT collaboration (HRC) is a growing field ensuring high-level security between people and robots [1]. To implement HRC, the robot and the working space need to be equipped with various high-performance sensors for powerful environment perception [2]. The proximity sensor is a key component of an HRC sensing system. It plays a major role especially when the user is close to the robot and blocks the detection of other types of sensors [3]. Recent trends towards HRC research have shown an increased interest in the flexible proximity sensor, which is widely used in the electronic skin to endow a robot with the proximity sense [4], [5]. The flexible proximity sensor generally comprises an array of ranging sensors, for example, time of flight (ToF) sensors, embedded in a

Manuscript received 20 December 2023; accepted 13 April 2024. Date of publication 1 May 2024; date of current version 7 May 2024. This letter was recommended for publication by Associate Editor D. Brscic and Editor A. Peer upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant U1913206, and in part by Shenzhen Science and Technology Program under Grant JSGG20211029095205007 and Grant JSGG20200701095404008. (*Corresponding author: Xin Wang.*)

The authors are with the School of Mechanical Engineering and Automation, Harbin Institute of Technology Shenzhen, Guangdong 518055, China, and also with the Guangdong Provincial Key Laboratory of Intelligent Morphing Mechanisms and Adaptive Robotics, Harbin Institute of Technology Shenzhen, Guangdong 518055, China (e-mail: hongrui_wu@163.com; wangxinsz@hit.edu.cn).

This letter has supplementary downloadable material available at <https://doi.org/10.1109/LRA.2024.3396105>, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3396105

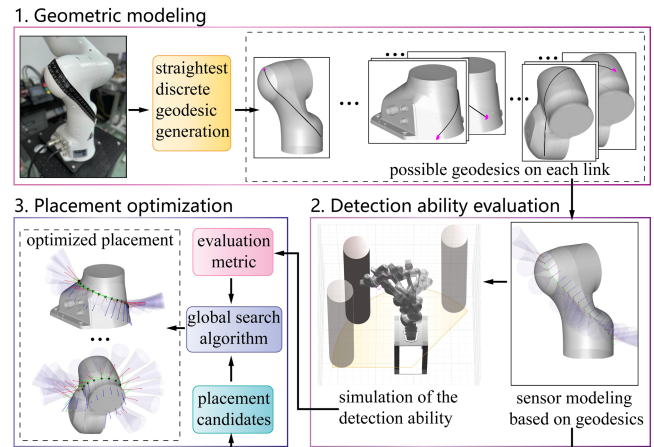


Fig. 1. Outline of the proposed placement optimization method.

flexible printed circuit (FPC) strip [6], [7], [8]. Such a flexible design boosts the compatibility and interchangeability of the sensor array.

Despite the advantages, the cost of a flexible proximity sensor is considerably high owing to the present manufacturing processes. It's necessary to consider the price when deploying it. Besides, the users often blindly mount excessive sensors on a robot, worsening the economy. For instance, proximity sensors are usually placed at the top and the bottom of every robot link like watches in many cases. Such a simple but expensive placement fails to exploit the sensor's most outstanding strength: flexibility, in which users are able to freely place the sensor in any allowable shape and position to pursue better detection performance. Therefore, it's significant to design an efficient method to guide the flexible sensor placement considering the trade-off between the performance and the cost.

To cope with the above issues, this study sets out to find the optimal placement of flexible proximity sensors on a robot arm of any shape. Based on the geodesic theory, we first express the geometric information of a flexible proximity sensor mounted on a robot link. The detection process of this sensor could then be modeled and the detection ability of the current sensor placement is evaluated using Monte Carlo method. With a global search algorithm, we could acquire the optimal placement location and the number of the flexible proximity sensors for a specific HRC task. Acceptable detection performance could therefore be achieved with as few sensors as possible. Fig. 1 outlines the proposed approach.

The main contributions of this work are summarized as follows:

- i) We build the model of a flexible proximity sensor strip mounted on a surface of an arbitrary shape, especially for a robot link.
- ii) We propose a systematic approach to the optimized placement of flexible proximity sensors for HRC, improving the detection performance and reducing the cost.
- iii) The proposed method exploits the compatibility and interchangeability of the flexible sensor, making the placement optimization customizable.

The rest of this letter proceeds as follows: Section II reviews the literature on relevant sensor placement methods. In Section III, we present the proposed method with technical details. An application case is introduced in Section IV and the results of the implementation are also discussed. Section V finally draws a conclusion on this study with suggestions for further work.

II. RELATED WORK

Before the prevalence of the flexible sensor, earlier studies mainly concentrate on the placement of distributed proximity sensor modules for HRC. In relevant research, the regions where sensors are allowed to be deployed on a robot link are first selected and the detection efficiency of a specific placement setup is assessed. A classic approach is uniformly disposing all the placement candidates in a hexagon pattern and constructing a cost function to quantify the detection possibility based on the theory of danger field [9], [10], [11]. Taking limited times of Monte Carlo simulation, the varying robot configuration and obstacle position are considered in the optimization procedure. Another typical study focuses on the depth sensor placement for a legged robot, which is similar to the proximity sensing case [12]. Researchers first evenly generate a set of candidate points on each robot link and consult a veteran mechanical engineer about the locations which are too narrow to attach a sensor. After discarding the ineligible candidates, the detection ability is quantified as the ratio of detectable points on a spherical surface around the robot to all the uniformly distributed points on this surface. Considering the varying robot configuration, several representative tasks and the robot configurations in these tasks are taken into account when computing the detection ability. To ensure a limited total number of sensors, the optimization is defined as a multi-objective problem and solved by Non-dominated Sorting Genetic Algorithm II (NSGA-II). Further, the next position the end-effector plans to reach is highlighted by adding a penalty item into the objective function. In [13], a statistical method is taken to measure the perception capability of a sensor placement. Researchers employ the relative entropy to estimate the difference between the obstacle appearance distribution and the sensor detection distribution, which are defined in advance. An optimization process is then designed to reduce the difference for a wider detection coverage. Likewise, the diversity of robot configurations is considered by adopting Monte Carlo method.

More recently, researchers have paid attention to the rigid sensor ring. Encircling the robot link, the sensor ring typically comprises multiple off-the-shelf ToF modules [14]. It plays a part in the collision avoidance for HRC [15]. From a geometric viewpoint, the detection ability of a particular sensor ring is quantified with the sensing volume, which expresses the detection coverage of the desired workspace [16]. The effect of the ring number on each link and the module number on each ring is investigated as well as the tilt angle of each ToF module. To eliminate the blind spots in the detection zone, researchers double the number of ToF modules and adjust their spacing for a

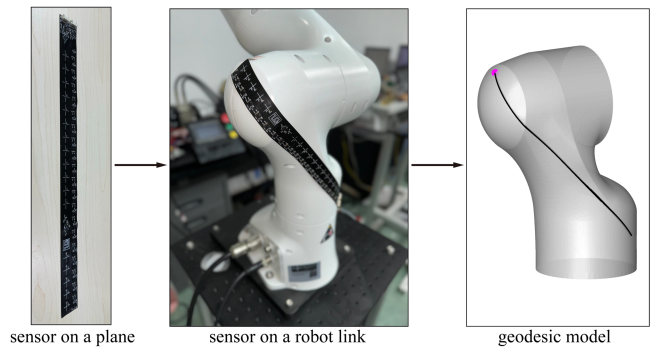


Fig. 2. Geodesic model of a flexible sensor's mounting location.

reasonable overlap [17]. In [18], a pair of sensor rings is mounted at the opposite ends of a robot link, directing the sensing region along the link. Optimizing the mounting angle, this research aims to form a detection zone that surrounds the whole body of the robot. Like [16], the sensing volume coverage is chosen as the criterion for detection ability. Despite the improvement of detection performance, the preceding studies excessively resort to the increase of the sensor number, neglecting the surging cost. Fixing the ring at the end of the robot link, they also fail to optimize the placement.

Nowadays, the flexible sensor has become a trending topic. Surprisingly, there is a relative paucity of research on the placement of the flexible proximity sensor. Thereby, we propose a systematic technique to find the optimal location and number of flexible sensors for an efficient proximity perception in the HRC.

III. METHOD

A. Background

To optimally place flexible proximity sensors on a robot link, the foremost step is to describe the possible mounting location of the sensor strip with an appropriate mathematical model. Thus the shape of the whole sensor strip as well as the position and orientation of each ranging module on the sensor could be specified.

Laying the flexible sensor on a plane, the edge of the sensor strip between its left side and right side is a straight line whereas it's one of the "straightest curves" on a surface if we lay the sensor on a robot link. The link is allowed to possess an arbitrary curved shape. Such a feature could be modeled by a classic concept in Riemannian geometry: geodesic [19], [20]. In fact, the geodesic is a generalization of the straight line from a plane to an arbitrary surface. Fig. 2 shows this concept.

Considering the features of geodesics, there are two major ways to generate a geodesic on a surface: the straightest curve generation and the shortest curve generation. Before applying the geodesic theory, it's essential to specify which feature is preferred and then use the appropriate technique. Compared with the shortest curve generation, the straightest curve enjoys the uniqueness. It's strictly defined by the initial point and direction. For example, from the South Pole to the North Pole, there is only one straightest curve in a given direction. However, in terms of the shortest curve, there are infinite ones (longitude lines) between the two poles. Based on the uniqueness of the straightest curve, we could go through all the possible geodesics on a robot link by changing the initial condition. Hence sufficient mounting location candidates could be prepared for the global

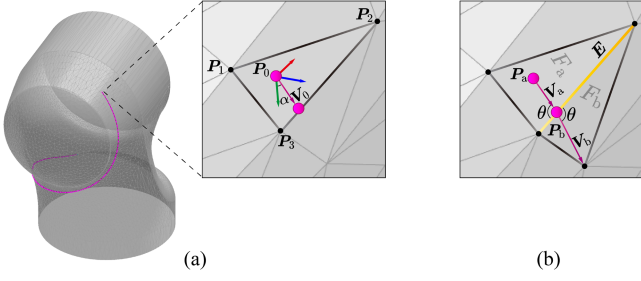


Fig. 3. (a) Initial value problem of the geodesic generation. P_1 , P_2 , and P_3 are the initial triangle's vertices, P_0 is the initial point, V_0 is the direction vector, and α is the direction angle corresponding to V_0 . (b) Edge case of the discrete geodesic update. E is the current edge, θ is the intersection angle, F_a and F_b are the current and next facets, P_a and P_b are the current and next points, V_a and V_b are the current and next direction vectors, respectively.

search optimization. Therefore, algorithms related to finding the straightest curve on the polyhedral surface are employed here.

B. Straightest Geodesic Generation

In most cases, the geometric information is depicted by a polyhedral model, in which discrete planar polygons are combined to approximate the smooth surface [21]. Likewise, we use the stereolithography (STL) model to express the shape of robot links. To generate the straightest geodesic on a polyhedral surface, a classic method is to give an initial point and a direction vector on a facet and trace the vector across the polyhedral surface from the initial point [22]. With the intersection between the vector and the edge of the passed triangle, a series of line segments could be generated as the discrete geodesic path. If the polyhedral surface could be unfolded into a plane, the final path would become a straight line on the plane. The determination of the initial point P_0 and the direction vector V_0 is commonly defined as the initial value problem for the discrete geodesic:

$$\gamma(0) = P_0, \quad (1a)$$

$$\gamma'(0) = V_0. \quad (1b)$$

By solving it, we can obtain a unique discrete geodesic γ .

In our case, each robot link is represented by a STL model. To begin the initial value problem, one triangle in the STL model is selected and the initial point is then determined based on barycentric coordinates of the triangle:

$$P_0 = \lambda_1 P_1 + \lambda_2 P_2 + \lambda_3 P_3, \quad (2)$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$; P_1 , P_2 , and P_3 are the Cartesian coordinates of the first triangle's vertices. Besides, a local coordinate system sitting in the initial triangle at P_0 is established and the direction vector V_0 could be defined by a direction angle α as

$$V_0 = [\cos \alpha \quad \sin \alpha \quad 0]^T. \quad (3)$$

Fig. 3(a) depicts the initialization of the geodesic generation.

With P_0 and V_0 , the discrete geodesic is generated by updating the path point and direction vector for a straight path. Extending the current direction vector, we can get an intersection point as the next path point P_b if the vector meets an edge E of the current triangle. To ensure the "straightest property", the current direction vector V_a and the updated one V_b on the next triangle, which is also attached to the current edge E ,

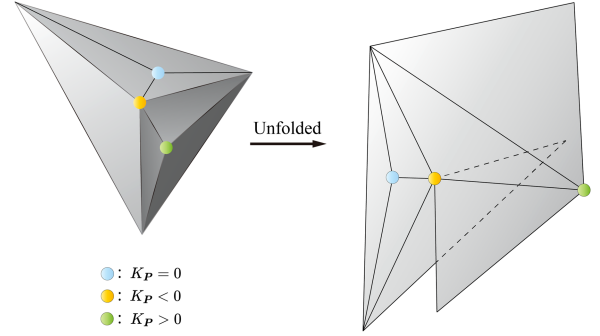


Fig. 4. Case that illustrates the relation between the symbol of a vertex P 's Gaussian curvature K_P and its location as well as the unfolded shape of the triangles attached to this vertex.

should share the equal angle with respect to E . According to this principle, the next path point and direction vector are generated. This idea is illustrated in Fig. 3(b). Such a process resembles a common scene on a plane, which is formed by unfolding the two attached facets: connecting a straight line between two points located at different sides of an edge.

Nevertheless, when the extended direction vector meets a vertex of the current triangle, the situation would become so complicated that we can't simply unfold two attached facets. In this case, every triangle that contains the specific vertex needs to be unfolded. What makes the issue more complex is that not every set of the attached triangles could be completely unfolded into a plane. It depends on the flatness of the vertex, which is generally quantified by the discrete Gaussian curvature [23].

In the polyhedral space, the Gaussian curvature K_P of a vertex P is defined by the angle defect:

$$K_P = 2\pi - \sum_{i=1}^n \theta_P^i, \quad (4)$$

which stands for the difference between 2π and the sum of each attached triangle i 's interior angles at P [24]. The latter item $\sum_{i=1}^n \theta_P^i$ is called the "total vertex angle" of the n triangles.

Generally, the symbol of K_P represents the location of the target vertex on the polyhedral model as well as the unfolded shape of the triangles attached to this vertex. Intuitively, a zero K_P represents a point on a flat plane. If $K_P > 0$, the vertex is at the apex of a spherical surface, and a missing piece exists in the unfolded triangles. Otherwise, if $K_P < 0$, the vertex is at the saddle point of a hyperbolic surface and there is an overlapping area in the unfolded triangles. This relation is delineated by a case in Fig. 4. As a result, simply drawing a straight line across the edge doesn't work when the direction vector meets a vertex where K_P is not zero.

To specify the next direction vector at a vertex, a modified approach should be used to adapt to this special case. A workable way is to search the vector that bisects $\sum_{i=1}^n \theta_P^i$ through the attached triangles. To implement this idea, we compute the initial angle between the current direction vector and one of the current triangle's edges attached to the vertex. Along a specific direction, we accumulate the initial angle with the next attached interior angle triangle by triangle. When the cumulative value θ_{sum} exceeds half of $\sum_{i=1}^n \theta_P^i$, the next direction vector is modified in the last searched triangle by α to meet the expected half value. This procedure is shown in Fig. 5.

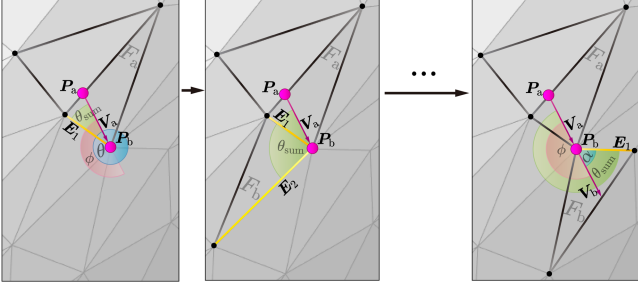


Fig. 5. Vertex case of the discrete geodesic update. F_a and F_b are the current and next facets; P_a and P_b are the current and next points; V_a and V_b are the current and next direction vectors; E_1 and E_2 are the current and next edges, respectively; θ is the sum of the interior angles at P_b , ϕ is half of θ , θ_{sum} is the cumulative interior angle, α is the difference between the final θ_{sum} and ϕ .

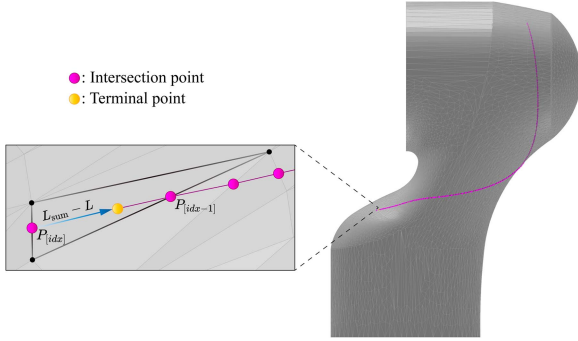


Fig. 6. Terminal point of the discrete geodesic is translated backward to meet the length condition. L and L_{sum} are the sensor length and the cumulative geodesic path length; $P_{[idx]}$ and $P_{[idx-1]}$ are the last and the next-to-last intersection points, respectively.

Integrating the two complementary cases, we could describe the shape of a flexible sensor strip mounted on a robot link by generating the straightest geodesic path starting with an initial point and a direction from the link's STL model. Algorithm 1 depicts the overall procedure. The resulting discrete geodesic comprises successive line segments between a series of intersection points. When the total length of the line segments exceeds the sensor's, the terminal point is translated backward against the last direction vector to meet the boundary condition of the sensor's length, as shown in Fig. 6.

To tightly place a flexible sensor, the surface it sticks to should be smooth and continuous. Thus in terms of two successive triangles containing the geodesic, the angle between their normal vectors shouldn't be too large. By defining a threshold value ($\pi/2$ in our case), we can exclude the undesired mounting location. Besides, the original STL model might contain some problematic features which hamper the generation of the straightest geodesic. A typical example is the isolated edge, which is exclusive to only one triangle. It incurs an error when searching the other facet attached to the isolated edge apart from the current one. Before the geodesic generation, we could avoid the negative effect of the STL file by using off-the-shelf mesh preprocessing algorithms to repair the polyhedral model.

Algorithm 1: Straightest Geodesic Generation.

Input: Sensor length L , Initial point P_0 , Direction angle α
Output: Intersection point sequence P , Facet sequence F

- 1: $P_a = P_0$, $V_a \leftarrow$ direction vector corresponding to α , $F_a \leftarrow$ facet containing P_0
- 2: $L_{\text{sum}} = 0$, $idx = 1$, $P_{[idx]} = P_a$, $F_{[idx]} = F_a$
- 3: **while** $L_{\text{sum}} < L$ **do**
- 4: **if** V_a meets a vertex of F_a **then**
- 5: $V_b, P_b, F_b \leftarrow$
 Geodesic Path Update : Vertex Case
- 6: **else**
- 7: $V_b, P_b, F_b \leftarrow$
 Geodesic Path Update : Edge Case
- 8: **end if**
- 9: **if** the angle between the normal vectors of F_a and F_b
 $\geq \frac{\pi}{2}$ **then**
- 10: $flag = 1$, **break**
- 11: **else**
- 12: $flag = 0$
- 13: $V_a = V_b$, $P_a = P_b$, $F_a = F_b$
- 14: $idx = idx + 1$, $P_{[idx]} = P_a$, $F_{[idx]} = F_a$
- 15: $l \leftarrow$ length of the line between $P_{[idx]}$ and $P_{[idx-1]}$
- 16: $L_{\text{sum}} = L_{\text{sum}} + l$
- 17: **end if**
- 18: **end while**
- 19: **if** $flag == 1$ **then**
- 20: Remove the current geodesic path
- 21: **else**
- 22: $P_{[idx]} \leftarrow$ the point on the line between $P_{[idx-1]}$ and $P_{[idx]}$, whose distance to $P_{[idx-1]}$ is $L_{\text{sum}} - L$
- 23: **end if**

C. Flexible Proximity Sensor Modeling

Once a straightest geodesic of a robot link is extracted, the model of a flexible sensor placed in this location could then be established. Generally, a flexible proximity sensor consists of multiple ranging modules. They are evenly mounted on a FPC strip and play a major role in the obstacle detection. During sensing the potential obstacle, the emitter of the ranging module will transit an illumination cone as the detection zone. If part of the obstacle is inside the cone, the distance from the ranging module to the obstacle could be measured.

To model the flexible proximity sensor, we evenly distribute ranging modules along the straightest geodesic and deem them to be a series of apexes that the illumination cones emit from. Having attached a coordinate system, the pose of a ranging module S mounted on the i th link is expressed by a constant transformation matrix ${}^i T_S$, which is relative to the i th link frame.

As regards the cone's dimensions that indicate the performance of the ranging module, the aperture and the slant height are two key parameters. They stand for the field of view (FOV) and the maximum ranging distance of the ranging module, respectively. With the pose and dimensions of each ranging module on a flexible sensor, we have established its model under a specific mounting location as shown in Fig. 7.

D. Detection Ability Evaluation

After modeling the flexible proximity sensor, the detection ability of a specific sensor placement is then estimated based on

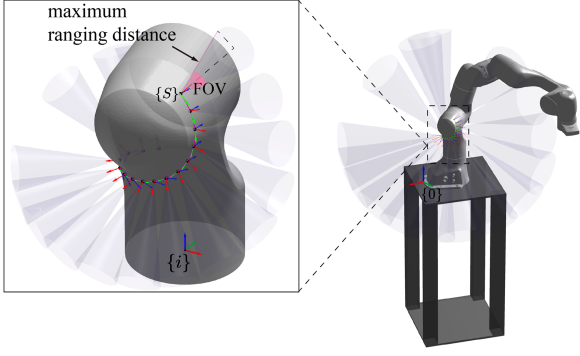


Fig. 7. Model of the flexible sensor under a specific mounting location. $\{S\}$ is the frame of one of the ranging modules, $\{i\}$ is the frame of the robot link the sensor is attached to, and $\{0\}$ is the frame of the robot base.

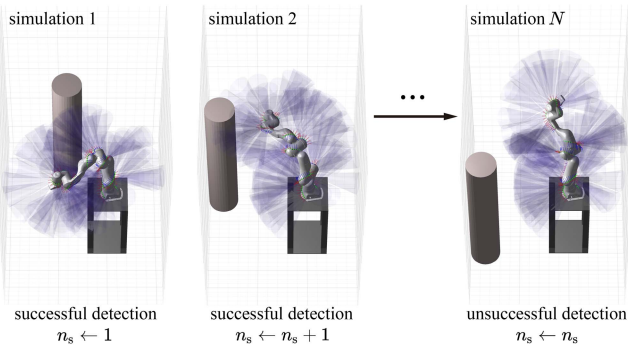


Fig. 8. Monte Carlo simulation case for the successful detection ratio. n_s is the cumulative successful detection times, N is the total simulation number.

Monte Carlo method. The basic idea is to count the successful detection times n_s among a large number of simulations N and calculate the success ratio:

$$\zeta(\%) = \frac{n_s}{N} \times 100\%. \quad (5)$$

Considering the motion of the robot as well as the obstacle, the robot's pose and the obstacle's position vary from simulation to simulation. Taking plenty of samples into account, the detection ability could then be comprehensively evaluated. Fig. 8 shows a case of the Monte Carlo method.

In every simulation, the pose of the robot is described by joint variables within the range of the joint constraint:

$$\mathbf{q} = [q_1, q_2, \dots, q_m], \quad (6)$$

where m is the degree of freedom (DoF) of the robot. When the robot's pose changes every simulation, the pose of the ranging module follows the motion of the robot link. Multiplying ${}^i T_S$ by the transformation matrix ${}^0_i T$ from the i th link frame to the base one, the homogeneous matrix ${}^0 T_S$ updates, indicating a varying pose of a ranging module, which is relative to the base frame:

$${}^0 T_S = {}^0_i T {}^i T_S, \quad (7a)$$

$${}^0_i T = {}^0_1 T {}^1_2 T \dots {}^{i-1}_i T. \quad (7b)$$

Regarding the obstacle, it's modeled by a cylindrical surface. Along with the varying robot configuration, the position of the obstacle randomly updates within a circle region around the

robot. If any part of the cylindrical surface is inside one of the illumination cones, the successful detection times n_s increase by one and a new simulation follows. Using (5), the successful detection ratio ζ is obtained as the evaluation metric of the detection ability of a specific sensor placement.

E. Sensor Placement Optimization

With the geometric model and the evaluation metric of a sensor placement, we set about searching for the optimal arrangement that enjoys sufficient detection performance with as few sensors as possible. Considering candidates for the optimal placement, all the possible sensor mounting locations on each robot link are generated. It's based on traversing the initial points and the direction vectors at appropriate intervals to go through the geodesics on a certain robot link via the technique in Section III-B. However, this method would cause repeated results when a geodesic's last intersection point and the reverse of the last direction vector are the initial values of another geodesic. To search for the surplus geodesics, we compare the positions of each geodesic's first and last intersection points as well as the middle point among the evenly distributed ranging module origins. Based on the comparison, the redundant ones from a group of similar geodesics could be removed.

As mentioned in Section III-B, a generated geodesic is comprised of path points. After generating all the geodesics on a robot link as mounting location candidates, each array that stores path points as a geodesic is indexed by a number. Thus we use the indexes to stand for the corresponding mounting locations. Based on the indexes, we define a placement using n sensors as $\mathcal{X}_n$:

$$\mathcal{X}_n := \{a_i, b_i\}_{i=1}^n, \quad (8)$$

where a_i is the index of a utilized robot link, and b_i is the index of a sensor mounting location on link a_i . In our case, each link is equipped with one flexible sensor at most.

To consider the trade-off between the detection performance and the costs, i.e. the number of sensors, the original problem is multi-objective. However, owing to the limited number of sensors, we convert the multi-objective problem into a single-objective one by specifying the sensor number in each optimization and comparing the results from different numbers of sensors:

$$\mathcal{X}_n = \arg \max \zeta(\mathcal{X}_n), \quad (9)$$

where ζ is the successful detection ratio defined in Section III-D.

When selecting $\mathcal{X}_n$ as the variable, the successful detection ratio is a discrete function. Meanwhile, with the same variable, the function would output different values with noise due to the Monte Carlo method. Confronted with the discontinuous and stochastic issue, Bayesian optimization, a global search method, is utilized for solving this problem [25]. Bayesian optimization exploits Gaussian process to approximate the indeterministic objective function based on several sample points. Further, an acquisition function is introduced to update these sample points. The beauty of Bayesian optimization is the efficiency when coping with a lengthy objective function and integer variables, which is suitable for this case.

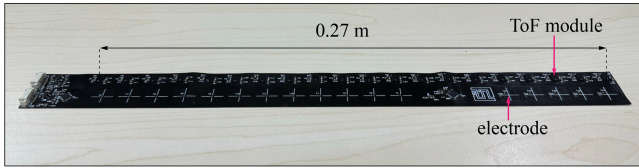


Fig. 9. Flexible sensor, whose length of the ToF module array is 0.27 m, comprises 16 ToF modules for regular detection and 16 electrodes for short-distance detection.

IV. CASE STUDY

A. Case Setup

We use a Franka Emika Panda 7-axis robot arm and flexible proximity sensor strips developed by ourselves to test the proposed method. Despite the fact that Franka Emika Panda has been equipped with force sensors for HRC, we don't utilize them in this case. In fact, our method is compatible with other common robot arms and flexible sensors.

The sensor strip comprises 16 VL6180X ToF modules. The length of the ToF ranging module array is 0.27 m. The FOV and the maximum ranging distance of VL6180X are 25° and 0.5 m, respectively. Because of the limited minimum ranging distance of VL6180X, we embed 16 electrodes in the sensor strip to fulfill the short-distance human detection based on capacitive sensing. Fig. 9 shows this sensor.

During the test, the robot carries out a pick-and-place task with human collaboration. This task is expressed by 80 discrete robot configurations looping through 3200 Monte Carlo samples in the simulation. Meanwhile, the position of the obstacle model randomly changes every Monte Carlo simulation within the working area, a 145° sector centered on the origin of the robot base with a radius of 0.855 m, the horizontal length of Franka Emika Panda's spherical workspace. In terms of the obstacle modeled by a cylindrical surface, we set its height and diameter to 1.75 m and 0.311 m, which are close to the 50 th percentiles of an adult male's height and width. This implementation case is depicted in Fig. 10. Apart from the pick-and-place task, the sensor's flexibility enables us to customize the placement for different robot tasks. Regarding a given task, we could adjust the robot configuration and the obstacle's position when estimating the detection ability and output an exclusive optimal arrangement for a specific task.

For the initial values of the discrete geodesic, we regard the STL facet's centroid as the initial point and extend the direction vector from each centroid at a direction angle interval of 30° . Using the method in Section III-B, we generate all the possible sensor placement candidates for each link.

In terms of the Bayesian optimization, we initialize it with 4 sample points. The expected improvement (EI) is selected as the acquisition function of Bayesian optimization to update these sample points. The exploration and the exploitation are equivalently considered during the optimization.

As mentioned in Section III-E, the sensor placement $\mathcal{X}_n$ is expressed by the utilized robot link indexes a and the corresponding mounting location indexes b . To simplify the multi-objective problem, we search for the optimal placement under a specific sensor number n and compare the results from different sensor numbers. In this case, $\mathcal{X}_n$ is searched one by one, which means that when n increases, a and b update based on the latest result. Therefore, each optimization only searches one utilized

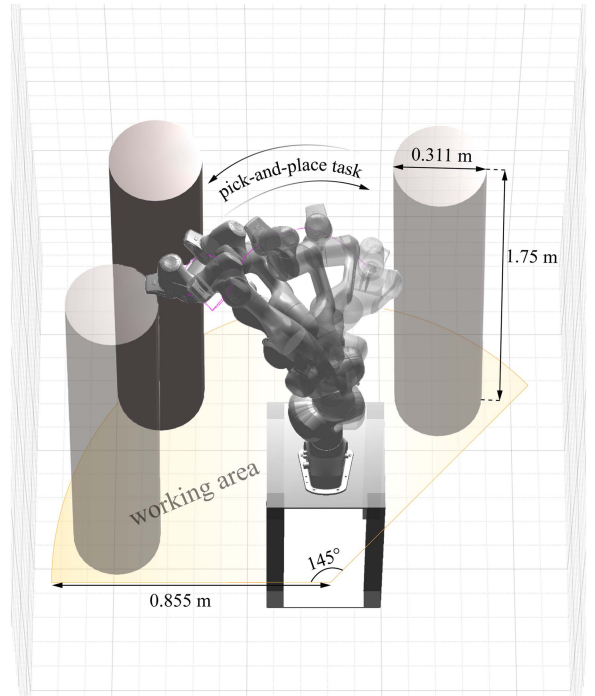


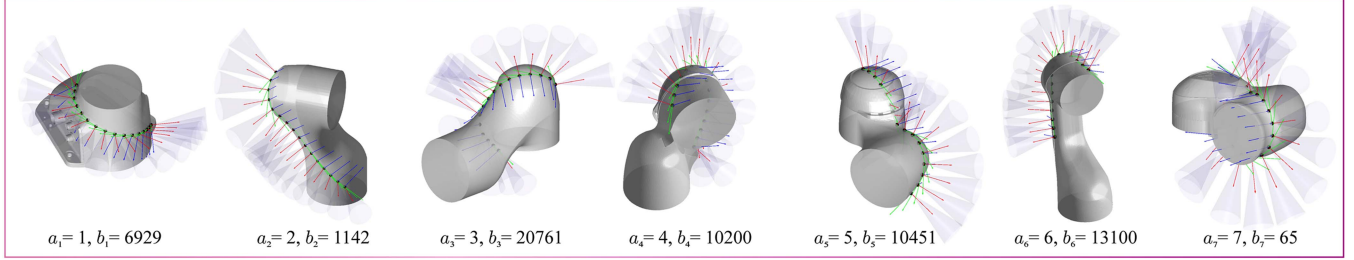
Fig. 10. Pick-and-place task and the obstacle model's working area.

link and the related placement. Compared with just specifying the sensor number and searching through more variables, trial results indicate that the one-by-one process is more efficient and could output better results.

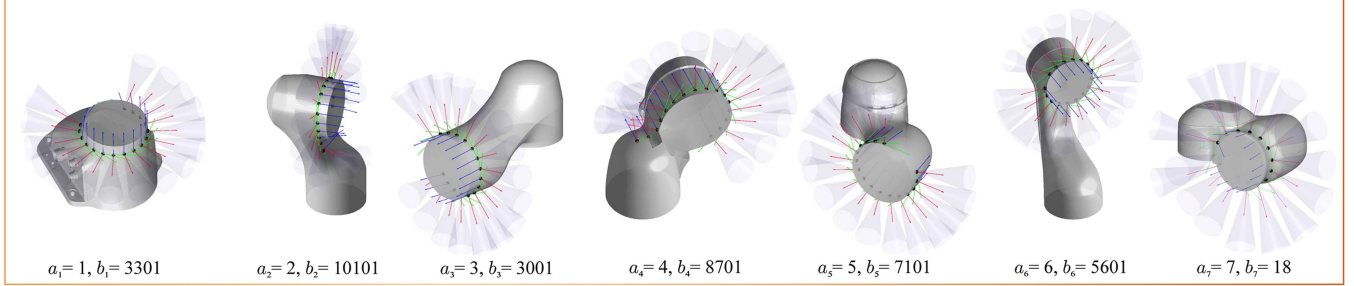
B. Simulation Analysis

For the 7 links of a Franka Emika Panda, we output 7 optimized placements $\mathcal{X}_1$ to $\mathcal{X}_7$ using 1 sensor to 7 sensors, respectively. To avoid breaking the sensor due to the relative velocity between the two attached links of a rotational joint, one flexible sensor could only be placed on one robot link, instead of crossing two links. Further, we compute the successful detection ratio of the non-optimized conventional placement where the flexible sensors are mounted around each robot joint like watches. Such a conventional placement is common in previous studies on rigid sensor rings, in which the placement optimization is neglected [14], [15], [16], [17], [18]. The poses of flexible sensors in the optimized placement $\mathcal{X}_7$ and the conventional placement are shown in Fig. 11. Considering the indeterministic results of Monte Carlo method, the corresponding ζ for each placement is calculated 32 times and averaged to obtain the value for comparison. Table I shows the results. Besides, a box plot is drawn to delineate the trend of ζ against the increasing sensor number as shown in Fig. 12(a).

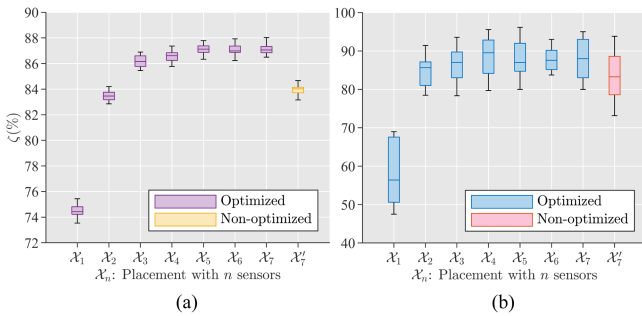
According to the results, the growth rate of ζ slows down with the increase of the sensor number. Despite 4 fewer sensors, $\zeta(\mathcal{X}_3)$ is merely 0.92% less than $\zeta(\mathcal{X}_7)$. It indicates that we could reduce costs by deploying fewer sensors under the condition of enough detection ability. Besides, the result of mounting 7 sensors in the conventional way, 83.93%, is only 0.46% more than $\zeta(\mathcal{X}_2)$, the optimized result of using only 2 sensors. It demonstrates the effectiveness of the sensor placement optimization.

Optimized placement $\mathcal{X}_7$ 

Conventional placement

Fig. 11. Optimized placement $\mathcal{X}_7$ and conventional placement. a and b are indexes of the robot links and the mounting locations, as mentioned in (8).TABLE I
RESULTS OF SENSOR PLACEMENT OPTIMIZATION

n	$\mathcal{X}_n$		$\zeta(\%)$
	a	b	
1	7	65	74.48
2	[1,7]	[6929, 65]	83.47
3	[1,6,7]	[6929, 13100, 65]	86.20
4	[1,2,6,7]	[6929, 1142, 13100, 65]	86.55
5	[1,2,3,6,7]	[6929, 1142, 20761, 13100, 65]	87.14
6	[1,2,3,5,6,7]	[6929, 1142, 20761, 10451, 13100, 65]	87.09
7	[1,2,3,4,5,6,7]	[6929, 1142, 20761, 10200, 10451, 13100, 65]	87.12
Conventional placement			
7	[1,2,3,4,5,6,7]	[3301, 10101, 3001, 8701, 7101, 5601, 18]	83.93

Fig. 12. Box plots of the successful detection rate ζ from optimized placements with different sensor numbers n and the non-optimized conventional one with 7 sensors. (a) Simulation data. (b) Experimental data.

C. Experimental Analysis

To verify the validity of the proposed method, we conducted an experiment under a real HRC environment as shown in Fig. 13 and the video attachment. The optimized placements and the conventional placement from the simulation were deployed in

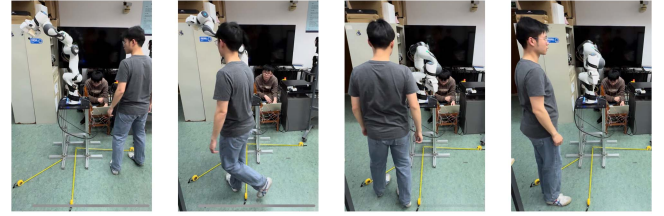


Fig. 13. Snapshots of the detection experiment under a real HRC environment. More details could be found in the video attachment.

the experiment for comparison. They were in accordance with Table I and Fig. 11. When the robot implemented the pick-and-place task, a participant, whose measurements were close to the size of the obstacle in the simulation, wandered within the working area. In the meantime, the ranging data of each mounted sensor were sent by the corresponding ROS publisher, whilst a ROS subscriber received and processed the data from all the publishers. An experimental successful detection ratio was then calculated after counting the successful detection times in 3000 sensing circles and excluding the self-occlusion detection in which the sensor detected the robot instead of the obstacle.

For statistical analysis, we repeated this experiment 32 times for each placement. Fig. 12(b) shows the box plot of the experimental results. Compared with Fig. 12(a), the variability of the experimental data is more significant owing to various factors in the real-world scenario such as the route of the participant, the scanning frequency of the flexible sensor, and the misalignment between the sensor mounting location in the simulation and that in the experiment. However, the general trend of the experimental data is in keeping with the simulation data. Despite an acceptable error from the simulation results, the experimental results show that the placement optimization could bring about better detection performance with much fewer

sensors, and this is consistent with the simulation results. It proves the effectiveness of our method.

V. CONCLUSION

This letter has investigated the optimal placement of flexible proximity sensors mounted on arm-type robots for obstacle detection in HRC. We model the flexible sensor based on the straightest geodesic and comprehensively measure the detection performance under a particular task and a sensor placement. It considers the change of the environment using Monte Carlo method. The optimal placement is searched via Bayesian optimization. After the optimization, the detection efficiency of flexible sensors has considerably improved.

Having analyzed the flexible sensor placement, we might shed light on broader aspects. Apart from optimizing the sensor placement, the proposed methodology could also be applied to facilitate the design process of the flexible sensor through modifying the optimization variables. For instance, we could optimize the sensor length and the number of ToF modules for the balance between the performance and costs. Further, it could play a part in the design of electronic skins, which are generally made up of flexible sensors.

Limited by the size of the flexible sensor, this letter only focuses on placing one sensor for one link. When the sensor becomes compacter with advanced manufacture techniques, more research is needed to discuss the case in which multiple sensor strips are mounted onto one link and intersecting strips should be excluded. Besides, corresponding collision avoidance methods also need to be studied.

REFERENCES

- [1] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human-robot collaboration," *Auton. Robots*, vol. 42, pp. 957–975, 2018.
- [2] X. Ding, J. Guo, Z. Ren, and P. Deng, "State-of-the-art in perception technologies for collaborative robots," *IEEE Sensors J.*, vol. 22, no. 18, pp. 17635–17645, Sep. 2022.
- [3] S. E. Navarro et al., "Proximity perception in human-centered robotics: A survey on sensing systems and applications," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1599–1620, Jun. 2022.
- [4] S. Tsuji and T. Kohama, "Proximity skin sensor using time-of-flight sensor for human collaborative robot," *IEEE Sensors J.*, vol. 19, no. 14, pp. 5859–5864, Jul. 2019.
- [5] J. Liang, J. Wu, H. Huang, W. Xu, B. Li, and F. Xi, "Soft sensitive skin for safety control of a nursing robot using proximity and tactile sensors," *IEEE Sensors J.*, vol. 20, no. 7, pp. 3822–3830, Apr. 2020.
- [6] S. Tsuji, "String-like time of flight sensor module for a collaborative robot," *IEEE Trans. Elect. Electron. Eng.*, vol. 18, no. 10, pp. 1576–1582, 2023.
- [7] D. Hughes, J. Lammie, and N. Correll, "A robotic skin for collision avoidance and affective touch recognition," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1386–1393, Jul. 2018.
- [8] Z. Tong et al., "An ultrasonic proximity sensing skin for robot safety control by using piezoelectric micromachined ultrasonic transducers (PMUTs)," *IEEE Sensors J.*, vol. 22, no. 18, pp. 17351–17361, Sep. 2022.
- [9] B. Lacevic, P. Rocco, and A. M. Zanchettin, "Safety assessment and control of robotic manipulators using danger field," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1257–1270, Oct. 2013.
- [10] N. M. Ceriani, G. B. Avanzini, A. M. Zanchettin, L. Bascetta, and P. Rocco, "Optimal placement of spots in distributed proximity sensors for safe human-robot interaction," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 5858–5863.
- [11] G. B. Avanzini, N. M. Ceriani, A. M. Zanchettin, P. Rocco, and L. Bascetta, "Safety control of industrial robots based on a distributed distance sensor," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 6, pp. 2127–2140, Nov. 2014.
- [12] M. Brandao, R. Figueiredo, K. Takagi, A. Bernardino, K. Hashimoto, and A. Takanishi, "Placing and scheduling many depth sensors for wide coverage and efficient mapping in versatile legged robots," *Int. J. Robot. Res.*, vol. 39, no. 4, pp. 431–460, 2020.
- [13] Y. Wang, L. Zhang, and G. Chen, "Optimal sensor placement for obstacle detection of manipulator based on relative entropy," in *Proc. IEEE 14th Conf. Ind. Electron. Appl.*, 2019, pp. 702–707.
- [14] S. Kumar, C. Savur, and F. Sahin, "Dynamic awareness of an industrial robotic arm using time-of-flight laser-ranging sensors," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2018, pp. 2850–2857.
- [15] S. P. Kumar, *Dynamic Speed and Separation Monitoring With On-Robot Ranging Sensor Arrays for Human and Industrial Robot Collaboration*. Rochester, NY, USA: Rochester Inst. Technol., 2020.
- [16] S. Kumar and F. Sahin, "Sensing volume coverage of robot workspace using on-robot time-of-flight sensor arrays for safe human robot interaction," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2019, pp. 378–384.
- [17] O. A. Adamides, A. S. Modur, S. Kumar, and F. Sahin, "A time-of-flight on-robot proximity sensing system to achieve human detection for collaborative robots," in *Proc. IEEE 15th Int. Conf. Automat. Sci. Eng.*, 2019, pp. 1230–1236.
- [18] K. Qi, Z. Song, and J. S. Dai, "Safe physical human-robot interaction: A quasi whole-body sensing method based on novel laser-ranging sensor ring pairs," *Robot. Comput.-Integr. Manuf.*, vol. 75, 2022, Art. no. 102280.
- [19] G. Peyré and L. D. Cohen, "Geodesic methods for shape and surface processing," in *Advances in Computational Vision and Medical Image Processing: Methods and Applications*. Dordrecht, The Netherlands: Springer, 2009, pp. 29–56, doi: [10.1007/978-1-4020-9086-8_2](https://doi.org/10.1007/978-1-4020-9086-8_2).
- [20] P. Bose, A. Maheshwari, C. Shu, and S. Wuhler, "A survey of geodesic paths on 3D surfaces," *Comput. Geometry*, vol. 44, no. 9, pp. 486–498, 2011. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925772111000459>
- [21] X. Wang, Z. Fang, J. Wu, S.-Q. Xin, and Y. He, "Discrete geodesic graph (DGG) for computing geodesic distances on polyhedral surfaces," *Comput. Aided Geometric Des.*, vol. 52/53, pp. 262–284, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167839617300390>
- [22] K. Polthier and M. Schmies, "Straightest geodesics on polyhedral surfaces," in *Proc. ACM SIGGRAPH Courses*, Series SIGGRAPH '06, 2006, pp. 30–38, doi: [10.1145/1185657.1185664](https://doi.org/10.1145/1185657.1185664).
- [23] N. Sharp, M. Gillespie, and K. Crane, "Geometry processing with intrinsic triangulations," in *Proc. ACM SIGGRAPH Courses*, 2021, doi: [10.1145/3450508.3464592](https://doi.org/10.1145/3450508.3464592).
- [24] L. Alboul and G. Echeverria, "Polyhedral Gauss maps and curvature characterisation of triangle meshes," in *Proc. Math. Surfaces XI*, R. Martin, H. Bez, and M. Sabin, Eds., 2005, pp. 14–33.
- [25] K. Yuan, I. Chatzinikolaïdis, and Z. Li, "Bayesian optimization for whole-body control of high-degree-of-freedom robots through reduction of dimensionality," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2268–2275, Jul. 2019.