

Article

Not peer-reviewed version

Active Touch Sensing for Robust Hole Detection in Assembly Tasks

[Bojan Nemec](#)*, [Mihael Simonic](#), [Ales Ude](#)

Posted Date: 13 June 2025

doi: 10.20944/preprints202506.1105.v1

Keywords: Active Tactile Sensing; Peg In Hole Assembly; 3D Object Localization; Deterministic and Probabilistic Search Algorithms; Robustness to Sensor and Map Uncertainty



Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Article

Active Touch Sensing for Robust Hole Detection in Assembly Tasks

Bojan Nemec *, Mihael Simonič and Aleš Ude

Humanoid and Cognitive Robotics Lab, Jožef Stefan Institute, 1000 Ljubljana, Slovenia

* Correspondence: bojan.nemec@ijs.si

Abstract: In this paper, we propose an active touch sensing algorithm designed for robust hole localization in 3D objects, specifically aimed at assembly tasks such as peg-in-hole operations. Unlike general object detection algorithms, our solution is tailored for precise localization of features like hole openings, using sparse tactile feedback. The method builds on a prior 3D map of the object and employs a series of iterative search algorithms to refine localization by aligning tactile sensing data with the object's shape. It is specifically designed for objects composed of multiple parallel surfaces located at distinct heights; a common characteristic in many assembly tasks. In addition to the deterministic approach, we introduce a probabilistic version of the algorithm, which effectively compensates for sensor noise and inaccuracies in the 3D map. This probabilistic framework significantly improves the algorithm's resilience in real-world environments, ensuring reliable performance even under imperfect conditions. We validate the method's effectiveness for several assembly tasks, such as inserting a plug into a socket, demonstrating its speed and accuracy. The proposed algorithm outperforms traditional search strategies, offering a robust solution for assembly operations in industrial and domestic applications with limited sensory input.

Keywords: active tactile sensing; peg in hole assembly; 3D object localization; deterministic and probabilistic search algorithms; robustness to sensor and map uncertainty

1. Introduction

In robotics various sensors are employed to enable the execution of complex tasks [1]. These sensors include 2D and 3D cameras, force sensors, tactile sensors, laser scanners, and similar devices. Among these, cameras have proven to be particularly effective and cost-efficient for applications such as bin picking [2], automated robot assembly [3], and quality control [4]. However, challenges arise when objects are not visible due to overlap, poor lighting conditions, or suboptimal camera positioning [5,6]. Accurate hand-eye camera calibration is a critical yet frequently underestimated challenge in robotics, which can present problems for operations requiring high precision, such as when assembling objects with low tolerance [7].

In such cases, reliance must shift to alternative sensors, such as force and touch sensors, which do not provide information as rich or comprehensive as that from cameras [8,9]. This paper addresses this issue and proposes a tactile localization method that does not rely on cameras.

A typical assembly operation, such as inserting a peg into a hole, can be divided into two primary phases: positioning the peg near the opening and the actual insertion. This paper does not address the insertion phase, which requires force sensors and that has already been well studied in robotics [10]. Instead, we focus on the approach phase, i.e. the localization of the opening, using only force sensors or touch detection.

Several heuristic and statistical search methods are commonly employed for this purpose, including random search [11], spiral search [12], genetic algorithms based search [13], ergodic search [14] and others. Among them, only random and spiral searches do not require prior knowledge of the environment. In contrast, ergodic search utilizes a probability distribution indicating where the assembly object is likely to be located in space, making it more effective than the other two methods.

Rather than relying primarily on heuristics or statistical priors, our approach assumes the availability of an accurate geometric model of the environment, although its position and orientation relative to the robot are initially unknown. The goal is to localize this model through systematic exploration.

2. State of the Art

The problem of object localization in assembly operations has been widely studied in prior research, with diverse approaches proposed depending on available sensing modalities and application contexts. Early methods, such as those by [15,16], utilized pre-acquired contact maps combined with particle filters to enable precise localization using sparse tactile data. Similarly, [17] introduced a computationally efficient iterative Bayesian Monte Carlo technique for six degree-of-freedom (6-DOF) pose estimation, demonstrating robustness in tactile localization tasks. Other approaches, such as the Gaussian mixture model-based contact state detection method proposed by [18], leverage wrench signals to facilitate peg-in-hole assembly localization.

Building on these foundations, tactile sensing for object localization has been further advanced by [19], who introduced the Next Best Touch (NBT) strategy to identify the most informative subsequent contact for efficient pose estimation. Extensions of this concept to 2D visual maps were explored by [20] using recursive Bayesian filtering to estimate belief distributions over possible locations, with [21] refining this framework to address both localization and shape uncertainty in active tactile sensing. Recent works have incorporated deep learning techniques to process tactile data more effectively; for example, [22,23] demonstrated the use of deep neural networks (DNNs) for tactile object pose estimation from high-resolution sensor arrays, achieving significant accuracy improvements. Other studies, such as [8] and [24], have successfully applied tactile contact sensing for object recognition and classification, highlighting the growing capabilities of tactile perception.

In parallel, related research in robotic grasping and manipulation has emphasized the integration of multimodal sensory inputs, combining vision, force, and tactile data to enhance pose estimation accuracy and robustness under uncertainty [25,26].

Despite these significant advancements, the majority of existing work — apart from [15,16] — does not explicitly target the challenge of assembly pose search using sparse binary touch sensors, which provide extremely limited and discrete information. This sparse sensing modality imposes unique challenges in developing algorithms capable of robust, efficient localization under minimal sensory input. Consequently, this remains a critical open problem in automated assembly, motivating further research into probabilistic and adaptive methods tailored for sparse tactile feedback.

Binary touch sensing, despite its simplicity, offers several key advantages in constrained environments. Unlike visual-tactile sensing, which requires cameras with clear line-of-sight, adequate lighting, and often precise calibration between visual and robot coordinate frames, binary contact sensors can operate in complete darkness, through occlusions, and without complex setup. This makes them particularly well-suited for tasks where cameras cannot be reliably deployed, such as operations in enclosed fixtures, poorly illuminated areas, or behind physical obstructions. Furthermore, visual-tactile systems generally require high-fidelity calibration and often depend on higher-bandwidth communication and processing pipelines, whereas binary touch sensing enables lightweight, reactive implementations that are easier to deploy and maintain in industrial environments. These trade-offs motivate the development of efficient localization algorithms that rely solely on binary tactile feedback.

3. Materials and Methods

In this section, we present our original algorithms for detecting the 3D position of objects using touch sensing. We begin by introducing a basic search algorithm for 2D position detection and subsequently extend it to handle 3D position estimation. We then enhance these algorithms with a probabilistic search framework designed to robustly manage sensor noise, inaccuracies in the object map, and variations due to object rotation.

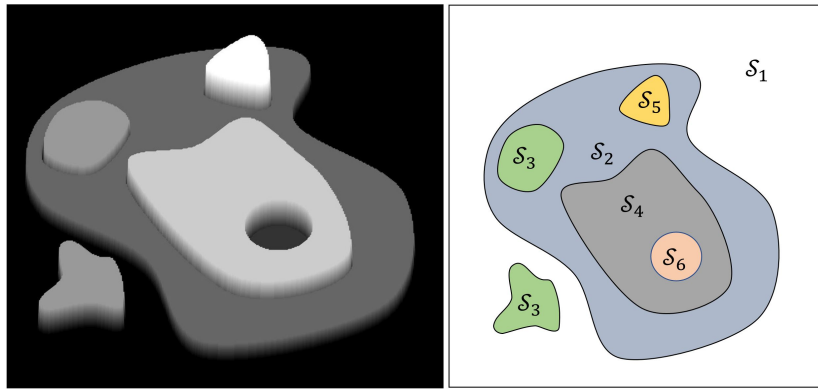


Figure 1. Left: 3D representation of the object's surface. Right: A 2D map with color-coded regions \mathcal{S}_i based on their height. Note that some regions may be disjoint (e.g., \mathcal{S}_3).

3.1. Map Registration

In this section, we present a deterministic 2D search method that serves as a foundation for 3D search introduced in Section 3.2 and its further enhancement into a probabilistic framework described in Section 3.3. Our approach shares similarities with Next Best Touch (NBT) methods [19], as it systematically refines the search region through geometric region elimination in consecutive steps.

We assume the availability of a 3D map of the object where the assembly operation takes place. Furthermore, we consider that the 3D object consists of a finite number of horizontal faces (quasi-isohight regions). These surfaces are represented as a 2D model map $\mathcal{M} = \{\mathcal{S}_i\}_{i=1}^N$ in the x - y plane, where \mathcal{S}_i denote the partitions of the map and N is the number of partitions. Each partition is defined as an area of the object having the same height z_i^m when put on the horizontal surface, $\mathcal{S}_i = \{\mathbf{p}_{i,j}^m\}_{j=1}^{N_i}$, $\mathbf{p}_{i,j}^m = [x_{i,j}^m, y_{i,j}^m, z_i^m]^T$ and N_i is the number of discrete points within the partition \mathcal{S}_i . In practice we obtain these points by discretization directly from a CAD model or, alternatively, using a scanner device. An example of such a region-based map is depicted in Fig. 1.

While the 2D map \mathcal{M} representing the object's geometry and the object's orientation is known, the position of the object in the robot's coordinate system is unknown. There are many practical examples in industry that satisfy these requirements, for example all objects that are rotationally invariant. There is also a common case when we can provide the exact orientation of the object, but not its position. We consider a scenario where the robot must determine any point on the target region \mathcal{S}_g , which centroid is denoted by \mathbf{p}_g^m in the map coordinate system. Initially, we are given an estimate of a point above the target region \mathcal{S}_g in the robot's coordinate system, which we denote here by $\tilde{\mathbf{p}}^r(0)$. However, due to uncertainty in this initial position, the robot might initially contact a different region. Note that we are not specifically looking for a centroid of \mathcal{S}_g but for any point in \mathcal{S}_g .

Next the robot moves in the $-z$ direction in the map coordinate system until it touches the surface of the object. By calculating the z coordinate of this initial contact point $\mathbf{p}^r(0)$ in the map coordinate system (see Eq. (4)), the robot can identify which of the regions has been touched.¹ We denote the touched region as $\mathcal{S}_s(0) = \mathcal{S}_i$, where i is the index determined by the measured height. The region \mathcal{S}_i is defined as $\{\mathbf{p}_{i,j}^m\}_{j=1}^{N_i}$, so each point $\mathbf{p}_j^m \in \mathcal{S}(0)$ corresponds to a point in \mathcal{S}_i .

In the following we apply notation where vectors with superscript $(.)^r$ are expressed in the robot's coordinate system, while the corresponding vectors with the superscript $(.)^m$ are expressed in the map's coordinate system.

¹ This assumes that a mapping between the map's z coordinate and the robot's z coordinate is available. However, this assumption is necessary only to explain the basic algorithm. In later sections, we extend the algorithm to cases where this mapping is initially unknown.

Initially, we determine the touched position in the map coordinate system as a point closest to the centroid of the touched region and ensure it is also contained in the that region. We denote this position as \mathbf{p}_e^m . The algorithm then computes the displacement vector:

$$\mathbf{d}^m = \mathbf{p}_g^m - \mathbf{p}_e^m. \quad (1)$$

Next the robot moves to the next estimate of the position above the target region:

$$\tilde{\mathbf{p}}^r(1) = \tilde{\mathbf{p}}^r(0) + \mathbf{R}_0 \mathbf{d}^m, \quad (2)$$

where the rotation matrix $\mathbf{R}_0 \in \mathcal{R}^{3 \times 3}$ accounts for the rotation between the robot and the map coordinate system. The robot then moves again along the $-z$ coordinate of the map coordinate system until it touches the surface of the object. The z coordinate of the new contact point $\mathbf{p}_r(1)$ in the map coordinate system determines the next touched region $\mathcal{S}_t(1)$.

To refine the estimate, we update the search region $\mathcal{S}_s(0)$ by selecting all points \mathbf{p}^m within \mathcal{S}_s that satisfy:

$$\mathcal{S}_s(1) = \left\{ \mathbf{p}_j^m \in \mathcal{S}_s(0) \mid \mathbf{p}_j^m + \mathbf{d}^m \in \mathcal{S}_t(1) \right\}. \quad (3)$$

The updated search region $\mathcal{S}_s(1)$ contains only the points that fulfill the above condition. The next estimate of \mathbf{p}_e^m is computed as the centroid of $\mathcal{S}_s(1)$. Like before, if the centroid is not contained within $\mathcal{S}_s(1)$, we take a random point from $\mathcal{S}_s(1)$ as the estimate of \mathbf{p}_e^m . This operation is repeated until the robot hits a point on the target region \mathcal{S}_g . We denote the iteration index by k .

In the appendix we show that the last touched position $\mathbf{p}^r(k)$ is guaranteed to lie within the target region \mathcal{S}_g .

The above procedure defines an iterative algorithm outlined in Algorithm 1.

Algorithm 1: Map Registration Algorithm Using Touch Sensing

Input: Map \mathcal{M} representing the object's geometry, initial estimate for the position above the target region $\tilde{\mathbf{p}}^r(0)$, the centroid of the target region \mathbf{p}_g^m , and rotation matrix \mathbf{R}_0 and height z_0 of the map in the robot coordinate system.

Output: Estimated position \mathbf{p}_e^m , robot touch position $\mathbf{p}^r(k)$ at goal region \mathcal{S}_g .

```

1  $z^m(0), \mathbf{p}^r(0) \leftarrow \text{TouchFloor}(\tilde{\mathbf{p}}^r(0))$ 
2  $\mathcal{S}_s(0) \leftarrow \text{GetRegion}(z^m(0))$ 
3  $k \leftarrow 0$ 
4 while  $z^m(k) \neq z_g^m$  do
5    $\mathbf{p}_e^m \leftarrow \text{GetPoint}(\mathcal{S}_s(k))$ 
6    $\mathbf{d}^m \leftarrow \mathbf{p}_g^m - \mathbf{p}_e^m$ 
7    $\tilde{\mathbf{p}}^r(k+1) \leftarrow \tilde{\mathbf{p}}^r(k) + \mathbf{R}_0 \mathbf{d}^m$ 
8    $z^m(k+1), \mathbf{p}^r(k+1) \leftarrow \text{TouchFloor}(\tilde{\mathbf{p}}^r(k+1))$ 
9    $\mathcal{S}_s(k+1) \leftarrow \text{RegisterRegions}(\mathcal{S}_s(k), \mathbf{d}^m, z^m(k+1))$ 
10   $k \leftarrow k+1$ 
11 return  $\mathbf{p}_e^m, \mathbf{p}^r(k)$ 

```

In Algorithm 1, we apply the following functions:

- **TouchFloor** is a function that involves the motion of the robot from the initial position $\tilde{\mathbf{p}}^r(k)$ along the $-z$ axes in the map coordinate system until it touches the surface of the object. It also computes the $z^m(k)$ coordinate of the touch point in the map coordinate system. This calculation involves the transformation

$$\begin{bmatrix} x^m(k) \\ y^m(k) \\ z^m(k) \end{bmatrix} = \mathbf{R}_0^T (\mathbf{p}^r(k) - \begin{bmatrix} 0 \\ 0 \\ z_0 \end{bmatrix}), \quad (4)$$

where z_0 is a constant that defines the z-component of the map coordinate system origin expressed in robot coordinates. Note that x and y coordinates of the map coordinate system origin are unknown.

- **GetRegion** returns the region index based on measured $z^m(k)$ coordinate at the contact point.
- **GetPoint** returns a point from $\mathcal{S}_s(k)$ closest to the centroid of $\mathcal{S}_s(k)$.
- **RegisterRegions** returns the region composed of all points \mathbf{p}_j^m that satisfy the condition $\mathbf{p}_j^m \in \mathcal{S}_s(k)$, $\mathbf{p}_j^m + \mathbf{d}^m \in \mathcal{S}_t(k+1)$.

To illustrate the proposed method, we apply it to the search for the socket where a robot should insert an audio jack plug, as shown in Fig. 2. Fig. 3 shows the iterative refinement process. The search area $\mathcal{S}_s(k)$ (marked in white) is gradually reduced until the estimated location \mathbf{p}_e^m is within the target region \mathcal{S}_g , ensuring successful insertion of the plug.



Figure 2. Audio plug and socket used in the example. The shaded square determines the search area for insertion of the pin into the socket and corresponds to the black area in Fig. 3

3.2. Map Registration with Unknown Object Base Plane Height

The algorithm presented in the previous section assumes that the z-coordinate of the object's surface can be directly determined from the touch sensor's reading. In other words, it requires prior knowledge of the height of the object's base plane in the robot coordinate system so that each touch immediately reveals which region was contacted. However, if the exact height is unknown, the robot cannot directly ascertain which region it has touched. In such scenarios, estimating the object's base z-coordinate (height) becomes a necessary step before proceeding with precise localization. The algorithm presented in this section overcomes this limitation by eliminating the need for prior height information, thus ensuring that the robot can still identify the contacted region.

We propose an iterative algorithm to estimate an object's base height using a 3D map and successive touch operations. As, before, we assume that the object consists of a finite number of uniform height regions, denoted as \mathcal{S}_i , where $i = \{1, \dots, N\}$ denotes the region index, each located at a distinct height z_i^m . From the 3D map, the algorithm first identifies the number of these regions, N , and their corresponding heights z_i^m in the robot coordinates.

The algorithm begins by selecting an arbitrary position above the object, establishes the contact point \mathbf{p}_0^r using the TouchFloor procedure and records the z coordinate as height z_0^r . At this stage, it is unclear which of the map's regions $\mathcal{S}_i, i = \{1, \dots, N\}$ the robot has touched. Therefore, the algorithm initializes a candidate region $\mathcal{S}_{s,i}$ for each i , effectively treating all N regions as potential matches. In subsequent steps, the algorithm narrows down the feasible candidate regions by eliminating regions that are inconsistent with additional measurements.

The robot touches the object at another arbitrary point \mathbf{p}_t^r , and a displacement vector in the map frame is computed as:

$$\mathbf{d}^m = \mathbf{R}(\mathbf{p}_t^r - \mathbf{p}_0^r). \quad (5)$$

The new contact point yields a height measurement z_t^r . We calculate height difference:

$$dz = z_t^r - z_0^r, \quad (6)$$

For each candidate region, the height $z_{s,i}^m$ is updated as

$$z_{s,i}^m = z_i^m + dz, \quad (7)$$

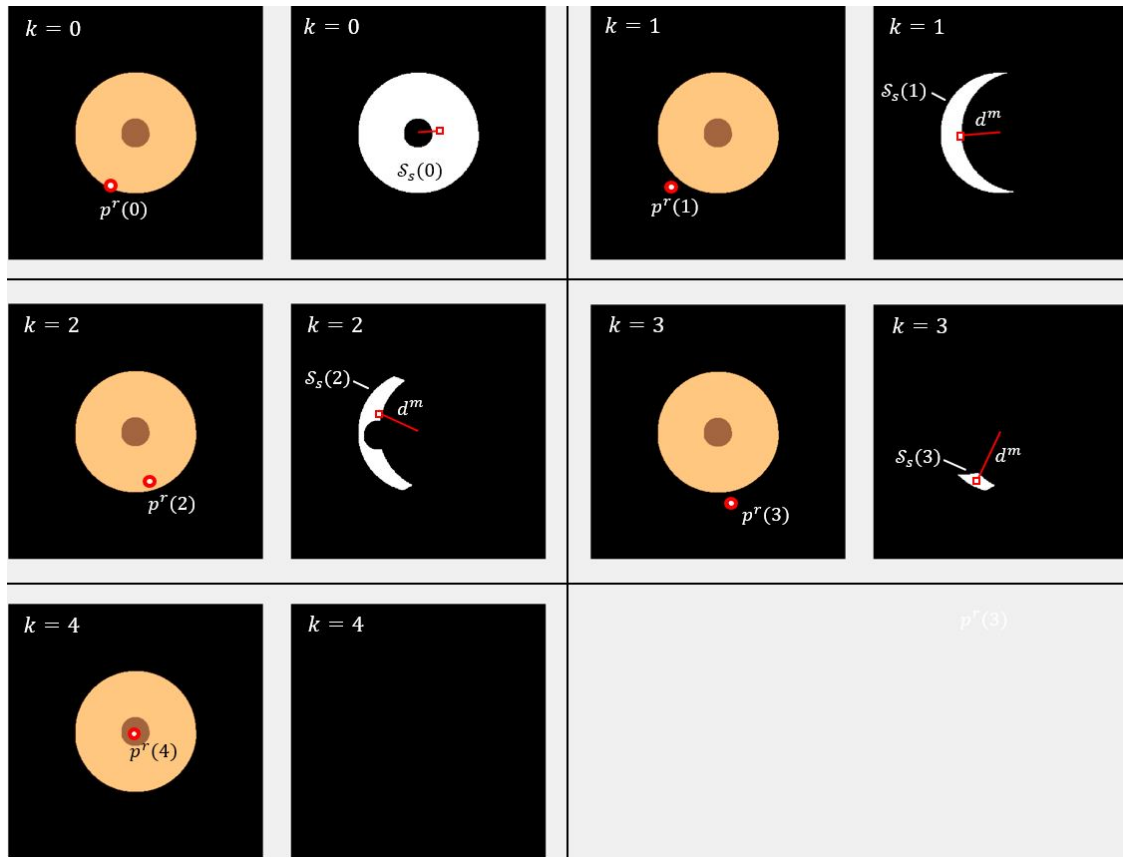


Figure 3. Example of the search process for the audio plug socket with progressive refinement of the search region over five steps ($k = 0, 1, 2, \dots$). Each step includes two views: the left shows the object map with the robot's contact point (red circle), which is unknown to the algorithm and displayed only for illustration; the right shows the current search region $\mathcal{S}_s(k)$ in white, the estimated point \mathbf{p}_m^e as a red square, and the direction vector \mathbf{d}_m . The map includes three regions: the dark brown socket hole (target), the light brown enclosure, and a black area where the robot misses the socket. At $k = 0$, the robot touches the object, and the algorithm identifies the touched region $\mathcal{S}_s(0)$. It selects \mathbf{p}_m^e near the centroid of $\mathcal{S}_s(0)$ and computes \mathbf{d}_m toward the goal point \mathbf{p}_m^g , located at the center. This guides the next move to $\mathbf{p}^r(1)$. The touched region is updated using Eq. (3), shrinking the search area to $\mathcal{S}_s(1)$. The process repeats, with the algorithm refining \mathbf{p}_m^e and \mathbf{d}_m at each step, until the robot reaches the goal region \mathcal{S}_g at $k = 4$, where the search area converges to zero.

where z_i^m is the height (z coordinate) of the i -th region \mathcal{S}_i . Additionally, each candidate region $\mathcal{S}_{s,i}(k)$ is updated by retaining only those points that satisfy the condition:

$$\mathbf{p}^m \in \mathcal{S}_{s,i}(k), \quad \mathbf{p}^m + \mathbf{d}^m \in \mathcal{S}_{h_{s,i}}, \quad (8)$$

where $\mathcal{S}_{z_{s,i}} \in \mathcal{M}$ represents the set of regions at height $z_{s,i}$.

This process is repeated until all but one of the candidate region have been eliminated (i.e., their $\mathcal{S}_{s,i}$ areas are reduced to zero). The remaining candidate region is then identified as the correct match for the initial contact point \mathbf{p}_0^r and the height associated with \mathcal{S}_i can be used to estimate the base z -coordinate of the object.

The algorithm is outlined in Algorithm 2. In addition to the functions already used in Algorithm 1, the following new functions are defined ²:

- $\text{Area}(\mathcal{S}_i)$ returns the area of the region \mathcal{S}_i .
- $\text{rand}(m, n)$ returns a $m \times n$ matrix with random numbers.

² In function TouchFloor, an unknown value z_0^r appears. However, since the results of this function are subtracted in algorithm 2, the value of z_0^r does not affect the result and can be set to 0.

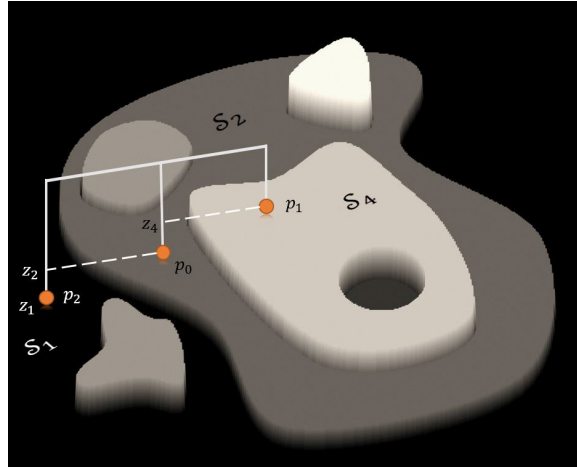


Figure 4. A set of contact points uniquely determines the identity of each plane. In this example, p_0 was the first touch, p_1 the second, and p_2 the third. This sequence, along with the detected height differences that identify the planes, is consistent only if p_0 belongs to region S_2 .

- `CountFeasibleRegions` returns the number of feasible candidate regions, i.e. regions with area greater than 0.

Algorithm 2: Map registration with unknown object base plane height using touch sensing

Input: Map of the surface \mathcal{M} with N regions, region heights z_i^m , initial point in robot coordinates $\mathbf{p}^r(0)$

Output: Estimated z-coordinate of the region $S(0)$ and the region index ix

```

1  $z_0^r \leftarrow \text{TouchFloor}(\mathbf{p}^r(0))$ 
2  $\mathcal{S}_{s,i} \leftarrow \mathcal{S}_i, i = \{1 \dots N\}$ 
3  $k \leftarrow 0$ 
4 while CountFeasibleRegions()  $\neq 1$  do
5    $\mathbf{d}^m \leftarrow \text{rand}(2, 1)$ 
6    $\mathbf{p}_t^r \leftarrow \mathbf{p}^r(0) + \begin{bmatrix} \mathbf{d}^m \\ 0 \end{bmatrix}$ 
7    $z_t^r \leftarrow \text{TouchFloor}(\mathbf{p}_t^r)$ 
8    $dz \leftarrow z_t^r - z_0^r$ 
9   for  $1 \leq i \leq N$  do
10     $z_{s,i}^m \leftarrow z_i^m + dz$ 
11     $\mathcal{S}_{s,i}(k+1) \leftarrow \text{RegisterRegions}(\mathcal{S}_{s,i}(k), \mathbf{d}^m, z_{s,i}^m)$ 
12   $k \leftarrow k + 1$ 
13  $ix \leftarrow \text{indexOf}(\text{Area}(\mathcal{S}_{s,i}(k) > 0))$ 
14 return  $z_{ix}^m + z_0^r, ix$ 

```

The underlying intuition behind this approach is that once the robot touches all planes constituting the object, we can uniquely determine the identity of each plane. In practice, the identity of a certain plane can often be determined by touching only some of the planes. By tracking the sequence of detected planes and their relative displacements, the algorithm ensures reliable plane identification. This process is illustrated in Fig. 4.

By estimating the z-coordinate before searching for the x- and y-coordinates (as described in Section 3.1), the algorithm significantly reduces the initial search space, minimizing computational complexity. Experimental results in Section 4 show that this additional step of determining z coordinate of the object's base plane only marginally increases the total number of search iterations.

3.3. Probabilistic Map Registration

In an idealized, noise-free environment, deterministic map registration can achieve arbitrarily high accuracy. However, real-world robotic assembly is affected by measurement noise, actuation errors, and map inaccuracies. These factors can lead to inconsistencies in which the search area \mathcal{S}_s

is prematurely reduced to zero before reaching the goal. To address these challenges, we propose a probabilistic extension of the deterministic approach.

Unlike the deterministic approach, where the map \mathcal{M} is partitioned in regions \mathcal{S}_i , we now model the likelihood that a point belongs to the search region. Let $P(\mathbf{p}^m \in \mathcal{S}_s(k))$ denote the probability that a point \mathbf{p}^m belongs to the search region $\mathcal{S}_s(k)$ at k -th iteration. Rather than assuming \mathbf{p}_e^m is in the center of $\mathcal{S}_s(k)$, the algorithm selects $\mathbf{p}_e^m \in \mathcal{S}_s(k)$ with the highest probability P .

In the deterministic map registration algorithm, the displacement vector \mathbf{d}^m is computed according to Eq. (1). In the probabilistic framework, we instead model the displacement length $\|\mathbf{d}\|$ as a random variable with a continuous probability distribution. It is sampled from the range $[\|\mathbf{d}\|_{\min}, \|\mathbf{d}\|_{\max}]$ assuming a normal distribution $\mathcal{N}(\mu_d, \sigma_d)$, where μ_d is taken as the displacement length calculated with Eq. (1) (see Fig. 5). The parameter σ_d models the uncertainty in the robot's position and map inaccuracies by controlling the spread of the Gaussian distribution used to sample the displacement length $\|\mathbf{d}\|$. Intuitively, σ_d defines the width of this distribution, determining how broadly the search region is updated around the expected displacement. Typically, it is chosen such that the Gaussian covers approximately 20–30% of the nominal displacement vector length $\|\mathbf{d}\|$. At this scale, the Gaussian falls to about 5% of its peak height at the distribution's edges, ensuring that the probabilistic update accounts for realistic positional errors without overly broadening the search space. This setting balances robustness against robot and map uncertainties with the efficiency of the search, and while the exact choice can be tuned experimentally, the described range provides a principled guideline.

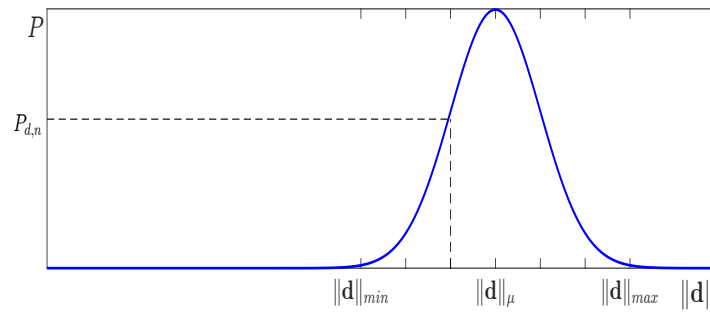


Figure 5. A distance $\|\mathbf{d}\|$ is modeled to be normally distributed. We sample the probability for each discrete distance $\|\mathbf{d}_n\|$ from the interval from $\|\mathbf{d}_{\min}\|$ to $\|\mathbf{d}_{\max}\|$.

The search region is updated accordingly. It is obtained by marginalizing over all possible displacement lengths. That is, instead of using a single displacement vector, we integrate the effect of sampled displacements weighted by their probability.

For computational reasons the length \mathbf{d} on the interval $[\|\mathbf{d}\|_{\min}, \|\mathbf{d}\|_{\max}]$ is divided into N_d intervals, \mathbf{d}_n , $n = 1 \cdots N_d$, each with an associated probability $P_{d,n}$, providing that

$$\sum_{n=1}^{N_d} P_{d,n} = 1.$$

In each k -th search step, for each length \mathbf{d}_n , we obtain region $\mathcal{S}_{s,n}(k+1)$ using the Eq. (3), following the same procedure as in the deterministic case. This way, we obtain N_d regions $\mathcal{S}_{s,n}(k+1)$ and compute:

$$\mathcal{S}_s(k+1) = \bigcup_{n=1}^{N_d} \mathcal{S}_{s,n}(k+1) \quad (9)$$

The probabilities are updated recursively as:

$$P(\mathbf{p}^m \in \mathcal{S}_s(k+1)) = P(\mathbf{p}^m \in \mathcal{S}_s(k)) \cdot \sum_{n=1}^{N_d} (P(\mathbf{p}^m \in \mathcal{S}_{s,n}(k+1))), \quad (10)$$

Similar as before in the deterministic approach, the algorithm narrows the search region $\mathcal{S}_s(k)$ until the robot hits the goal region \mathcal{S}_g .

Fig. 6 illustrates the probabilistic map registration process in an audio pin insertion task, similar to the deterministic approach in Section 3.1.

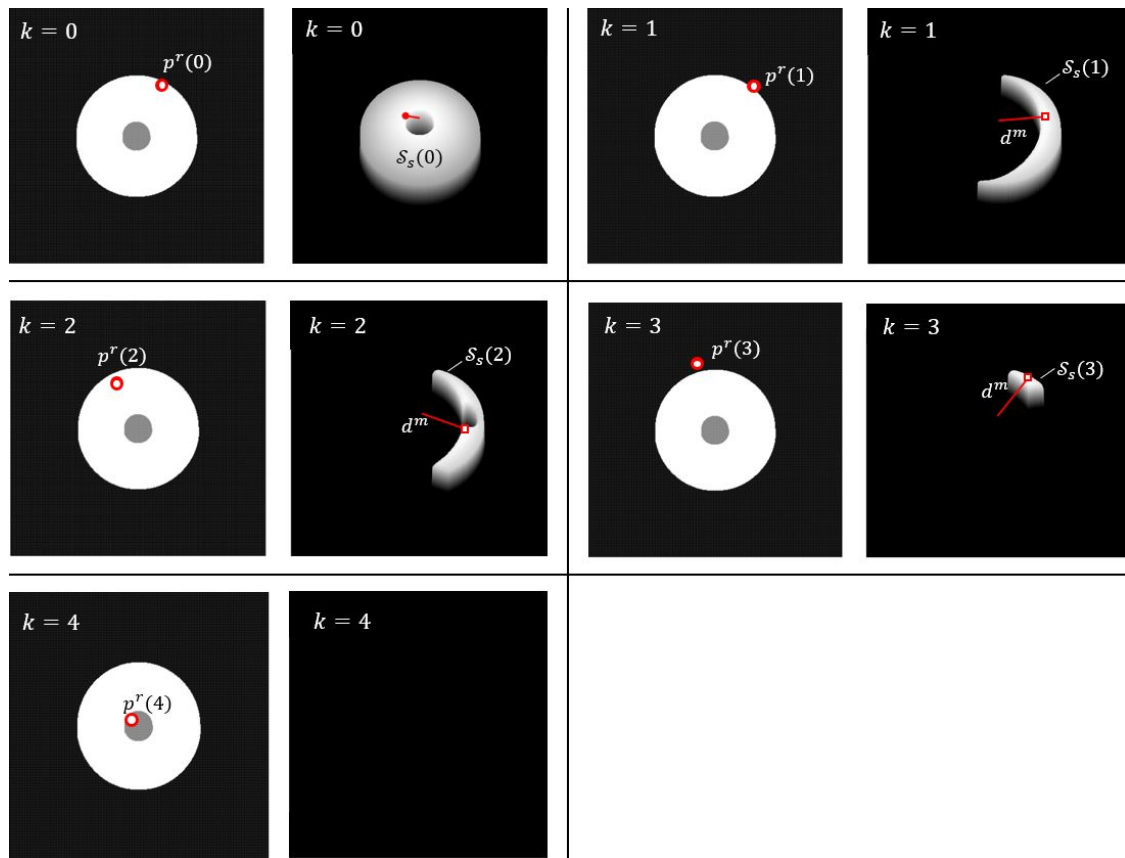


Figure 6. Example of probabilistic map registration for inserting an audio pin into a socket. The registration process is illustrated across sub-figures for $k = 0 \dots 4$. In the left sub-images, the gray region represents the socket center, the white region denotes the socket body, and the black region indicates the exterior. Red dots mark contact points, which are unknown to the algorithm. In the right sub-images, the search region \mathcal{S}_s is shown as a shaded 3D area tilted by 30° around the x -axis, where shading intensity represents the probability estimates $P(\mathbf{p}^m \in (\mathcal{S})_s(k))$. The red vector represents \mathbf{d}_μ^m , while the red square indicates \mathbf{p}_e^m . In this probabilistic case, the search region is represented with varying probabilities of robot position, accounting for sensor noise and map inaccuracies. The transition between steps (k to $k+1$) shows how the search space is adjusted dynamically, with increasing confidence in \mathbf{p}_e^m .

4. Experimental Results

In this section, we experimentally validate the performance of the proposed algorithm and compare it with a random search strategy. For all experiments, we used a 7-DOF Franka Research 3 robot controlled by an enhanced Cartesian impedance control law. The applied control law is detailed in [27] and [28]. Enhancements to the original control law include bidirectional friction compensation, which improved positional accuracy for small displacements with low stiffness. The touch motion was implemented by setting the velocity command in the direction of the surface normal of the object and monitoring the force in the same direction. Motion was halted whenever the force exceeded a predefined threshold, and impact forces were mitigated by setting low stiffness in the impedance control law in the direction of the surface normal. A touching probe with a known geometry is attached to the tip of the robot. This allows us to determine the height of the touched point in the robot base coordinate system.

4.1. Inserting the Pin into the Socket

To validate the efficiency and robustness of our algorithm, we first replicated the experiment of inserting an audio pin into a socket, as described in Section 3.1. The experimental setup is depicted in Fig. 7, where the socket was positioned on a table with its normal aligned along the z-axis. The socket was installed within a housing of 2 cm in diameter, with a socket hole measuring 3.5 mm.



Figure 7. Experimental setup for testing the insertion of the audio pin into the socket.

The search area was confined to a 4×4 cm square, and the map \mathcal{M} was encoded as a 400×400 matrix. Therefore, each point in the map corresponds to 0.1 mm. Since the coordinate frames of the map and the robot were aligned, the rotation matrix \mathbf{R} was set to the identity matrix. The robot's initial search position in robot coordinates was randomly selected within the defined search area.

In a set of 100 experimental trials, the algorithm successfully located the socket opening within one to ten attempts. Fig. 8 illustrates the convergence behavior and standard deviation of the search process. In this experiment, the results are virtually identical when using deterministic or probabilistic search.

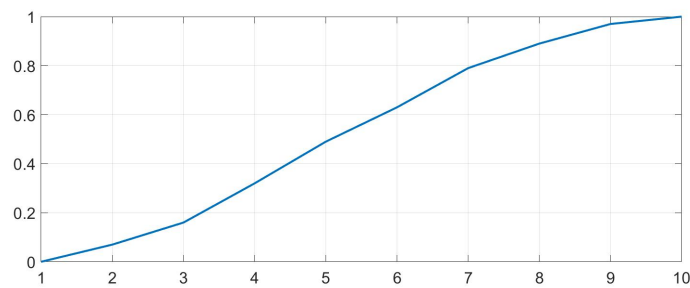


Figure 8. Convergence analysis of the proposed search algorithm. The x-axis represents the number of attempts, while the y-axis shows the probability of locating the target. The mean number of attempts is 5.83 and the standard deviation 2.04.

To further evaluate the algorithm's performance under more challenging conditions, we considered a scenario where the object's height relative to the robot is unknown. In this case, the height measurement alone is not sufficient to identify which of map partitions has been touched. Therefore, the algorithm first estimates the correct z-position before proceeding with the x- and y-coordinate search, following the procedures outlined in Sections 3.2 and 3.1, respectively. The convergence characteristics and standard deviation of this extended search process are illustrated in Fig. 9.

A comparative analysis between Figs. 8 and 9 reveals that incorporating the additional z-coordinate search increases the maximum number of attempts by only two, while the average number of attempts increases marginally. This demonstrates that the added dimensional complexity does not significantly degrade the efficiency of the search algorithm.

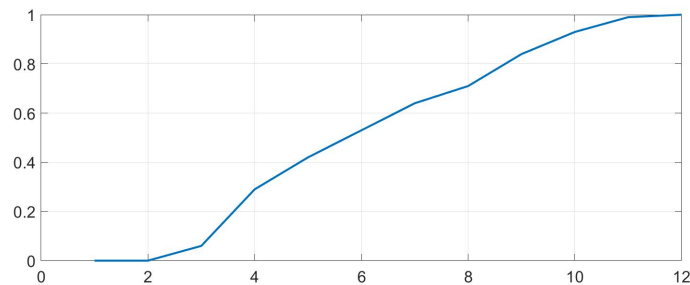


Figure 9. Convergence behavior of the combined search algorithm, which first determines the z-coordinate before localizing the x- and y-coordinates. The x-axis denotes the number of attempts and the left y-axis represents the probability of hitting the target. The mean and standard deviation are 6.37 and 2.53, respectively.

As a benchmark, we conducted an additional 100 trials using a purely random search strategy within a 4×4 cm grid with a 0.2 mm resolution. To ensure fairness, no points were tested more than once. Fig. 10 presents the convergence and standard deviation of the random search. The results

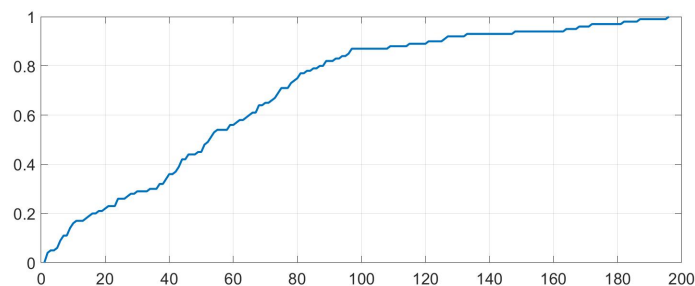


Figure 10. Convergence behavior of the enhanced random search. The figure shows the probability of hitting the target related to the number of attempts. The mean and standard deviation are 60 and 45.5 .

clearly highlight the superiority of our proposed search algorithm compared to random search. The algorithm demonstrates an average convergence speed more than six times faster than random search and exhibits significantly lower variance. In worst-case scenarios, our approach achieves over twenty times faster convergence, further validating its efficiency and reliability for real-world robotic assembly tasks.

In our final experiment, we evaluated the performance advantages of the probabilistic search algorithm under conditions of imprecise object mapping and positional inaccuracies of the robot. To simulate these uncertainties, we increased the distance parameter \mathbf{d}_m in Eq. (2) by a factor of 1.2, while retaining the original value of \mathbf{d}_m in the registration process described by Eq. (3). We then conducted 100 experimental trials of inserting an audio pin into a socket, comparing the success rates and convergence behavior of the deterministic and probabilistic search algorithms. The deterministic algorithm successfully inserted the pin into the socket in 85 out of 100 attempts, whereas the probabilistic algorithm achieved a success rate of 100 out of 100. Parameters N_d and σ_d were set to 20. These results, presented in Fig. 11, clearly demonstrate the superiority of the probabilistic search algorithm in noisy environments, highlighting its robustness in handling uncertainties.

4.2. Inserting the Task Board probe into the socket

The subsequent experiment pertains to the *Task Board*, an internet-connected device designed to assess real-world robot manipulation skills [29]. Following the trial protocol, one of the operations involves extracting a probe from its socket, measuring the probe's voltage level, wrapping the cable,

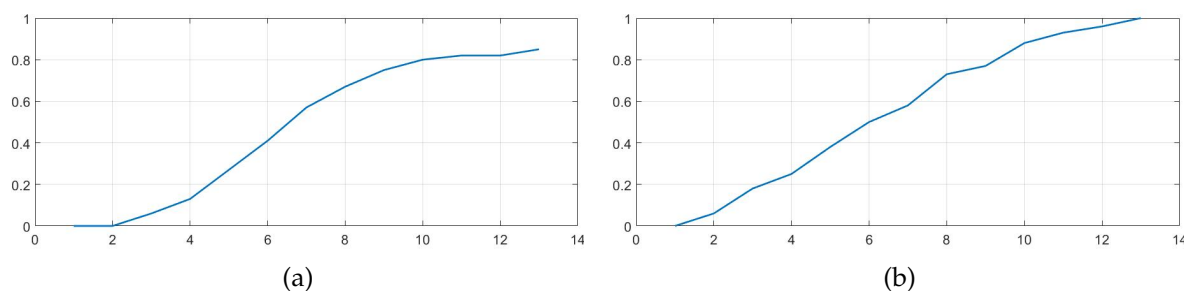


Figure 11. Convergence analysis of the deterministic search algorithm (a) and Convergence analysis of the deterministic search algorithm (b) in noisy environment. In this case, the deterministic and probabilistic search algorithms had success rate of 85% and 100%, respectively

and then stowing the probe. The last operation often fails due to factors such as incomplete grasp of the probe, significant movements during manipulation, environmental contact with the probe, and the effects of pulling the probe cable. As part of the euRobin project³, numerous Task Board manipulation solutions employing both in-hand and overhead cameras were introduced. However, these camera placements are inadequate for monitoring the probe-stowing operation. Consequently, an alternative solution utilizing touch detection was implemented for this purpose.

Initially, a 400 x 400 map with depth information of the socket housing was provided, as depicted in Fig. 12. In this case, the insertion is along the robot's x -axis, therefore the rotation matrix was

$\mathbf{R} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$. Each unit represented 0.1 mm in robot coordinates, with the socket hole having a

diameter of 4 mm. Following the protocol of previous experiments, 100 attempts were made to insert the probe into the socket, introducing random displacements of the starting point within the search area. In all attempts, the robot successfully inserted the probe into the socket in two to six attempts. The convergence and standard deviation of the search algorithm for this scenario are shown in Fig. 13.

As demonstrated, the algorithm identified the target more quickly than in the previous example. This increased efficiency is attributed to the more complex environment, which provides additional information about the location during exploration.

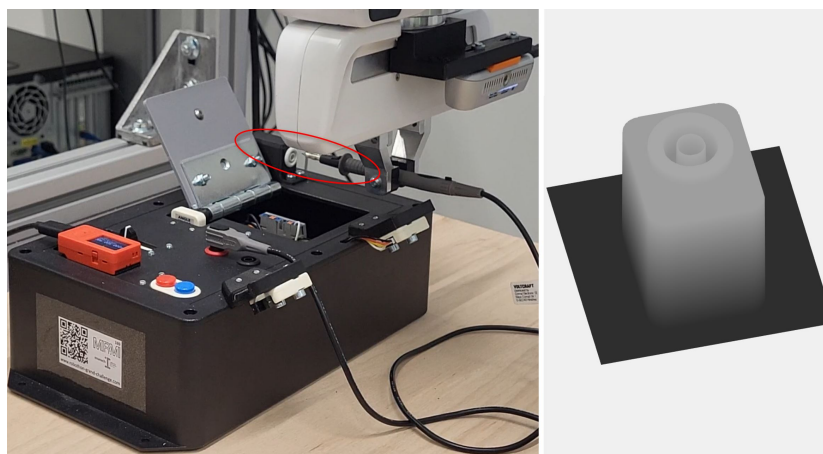


Figure 12. Left: Experimental setup for testing the stowing of the probe in the Task Board. The red oval highlights the socket and the probe. Right: A 3D map of the socket used for registration in the corresponding experiment. Note that the left and right images are intentionally shown from different viewpoints to emphasize the rotation \mathbf{R} between the robot's coordinate system and the map's coordinate system.

³ <https://www.eurobin-project.eu/>

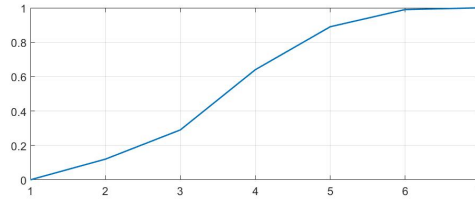


Figure 13. The convergence of the proposed search algorithm, showing the the probability to hit the target vs number of attempts. Mean and the standard deviation are 4.07 and 1.18, respectively

4.3. Inserting the Task Board Connector into the Socket with Continuous Search

In the previous examples, we evaluated the proposed search algorithm on objects with top surfaces that were not sufficiently smooth to allow for continuous trajectory-based search. However, the proposed procedure is also applicable and efficient in cases where a continuous trajectory can be employed to systematically sweep a designated search area. To demonstrate this capability, we again utilize the Task Board, this time focusing on the insertion of the termination connector of the test probe, as shown in Fig. 14.

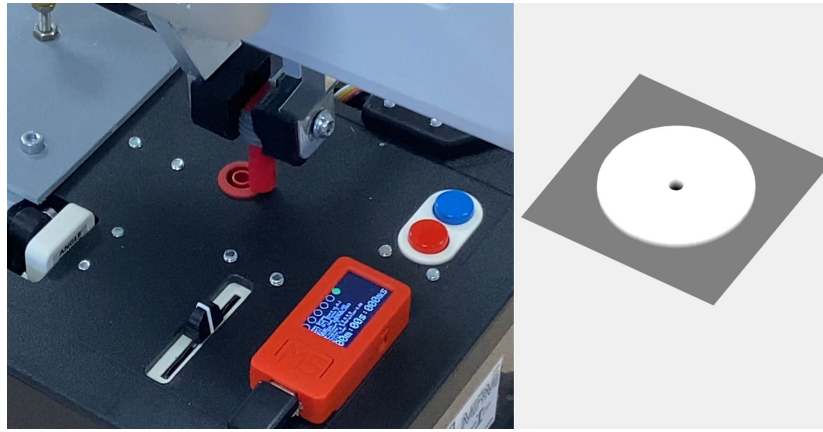


Figure 14. Left: Robot inserting the termination connector with combined spiral search and map registration algorithm. Right: Model of the socket, as used by the search algorithm

The search procedure is initiated using a spiral search strategy, where the trajectory is continuously updated at each sampling interval $t = k\delta t$ according to the following equation:

$$\mathbf{p}_t^r(k) = \mathbf{p}_0^r + (\delta r k) \begin{bmatrix} \sin(2\pi\gamma k) \\ \cos(2\pi\gamma k) \\ 0 \end{bmatrix}, \quad (11)$$

where \mathbf{p}_0^r represents the initial search position, δr defines the radial increment per step, and γ is the angular frequency governing the spiral motion. The parameters δr and γ must be carefully tuned to ensure that the generated trajectory sufficiently covers the search area and reliably intersects the goal region from any starting position \mathbf{p}_0^r .

During the spiral search, the robot applies a controlled force in the z-direction while maintaining compliance along this axis. This allows it to smoothly traverse the surface and conform to any variations in height. When the probe encounters the socket opening, it slides into place, marking the successful termination of the search. Further details on controlling the robot's stiffness and force at the tool center point can be found in [28].

To further improve the search efficiency, we integrate the spiral search with the map registration algorithm introduced in Section 3.1. First, we construct an appropriate model of the socket. Given that the plug is a cylinder with a radius of 4 mm, we account for its insertion by increasing the socket's radius accordingly. Additionally, considering the insertion tolerance of $\epsilon = 2mm$, the total radius

of the socket hole is adjusted to accommodate this clearance. For simplification, we model the plug as a point mass while ensuring that its physical constraints within the socket are maintained (See Fig. 14 right). The map registration algorithm runs concurrently with the spiral search, refining the position estimate dynamically. Specifically, an update is triggered whenever the distance between two consecutive points exceeds a predefined threshold:

$$\|\mathbf{p}_k^r - \mathbf{p}_{k-1}^r\| > \delta_{min}. \quad (12)$$

The algorithm continuously tracks the area of the current search region \mathcal{S}_s , which contains the initial search point \mathbf{p}_0^r . If the area of \mathcal{S}_s shrinks below the area of the goal region (i.e., the required region for successful insertion), the next command position is determined using Eq. (2). In this experiment we applied probabilistic map registration algorithm. This adaptive refinement significantly enhances search efficiency.

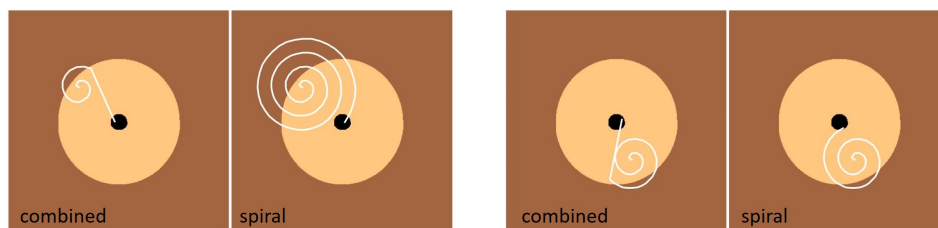


Figure 15. Comparison between the combined search algorithm and the spiral search algorithm for inserting the TaskBoard connector into the socket, evaluated for two different starting points. The white line represents the trajectory of the connector's center during the search. The dark brown area indicates regions where the connector fails to engage with the socket, while the light brown area represents regions where the connector glides over the socket. The black region marks the goal.

The advantages of the combined search approach are illustrated in Fig. 15, which compares the performance of the combined algorithm with the standard spiral search for two different initial positions. In both cases, the combined algorithm exhibited faster convergence to the goal region. However, the efficiency of the combined approach depends on the amount of information gained about different regions during the search. If the robot does not encounter new regions while searching, the combined algorithm performs similarly to the standard spiral search. Consequently, when the initial position is close to the goal region, there are no performance difference between the two methods. On the other hand, the spiral search requires precise tuning of free parameters to successfully complete search. In contrast, the proposed combined search algorithm is successful even with poorly set spiral search parameters.

4.4. Summary of Experimental Results

To provide a concise overview of the method's performance across different use cases, we summarize the key experimental results in Table 1. The table reports the number of trials, success rate, average number of attempts, standard deviation, average search time and qualitative notes on search characteristics. This summary highlights the efficiency and robustness of the proposed method across a variety of practical scenarios, including both discrete and continuous search strategies, and under both ideal and noisy conditions.

We observe that across all tested scenarios, the proposed algorithm performs consistently and robustly, with probabilistic extensions offering added resilience under uncertainty. The combined continuous and probabilistic method further improves convergence efficiency in smooth surface conditions.

All experimental use cases are additionally described in the attached videos, where the exploration of the algorithm and the process of evaluating the starting point can be observed. The Matlab source code of the registration algorithms in the simulated environment and videos are available via repo.ijs.si/nemec/3d-object-pose-detection-using-active-touch-sensing.

Table 1. Summary of experimental results across different use cases.

Experiment	Trials	Success Rate	Avg. Attem.	Std. Dev.	Avg. Time	Notes
Audio Pin Random Search (Baseline)	100	100%	37.37	36.55	71.0 s	No prior knowledge used
Audio Pin Insertion (Deterministic)	100	100%	5.83	2.04	11.1 s	Basic algorithm with known object height
Audio Pin + Height Estimation	100	100%	6.37	2.53	12.8 s	Includes z-height search step
Audio Pin (Noisy, Deterministic)	100	85%	6.78	3.0	12.1 s	Sensitive to uncertainty, occasional failure
Audio Pin (Noisy, Probabilistic)	100	100%	6.76	2.32	12.8 s	Robust under position and map uncertainty
Task Board Probe	100	100%	4.07	1.18	8.7 s	Rich geometry improves convergence
Task Board Connector (Combined)	20	100%	—	—	7.8 s	Spiral + map registration, robust to par. settings

5. Conclusion

In this study, we introduced a novel algorithm for locating openings in peg-in-hole assembly tasks using sparse tactile feedback. Building upon principles from NBT techniques, particle filters, iterative tactile probing, and active hypothesis testing, the method leverages prior geometric knowledge of the target object to enable efficient search in environments with limited sensory data. Our experimental results demonstrate two key insights: (1) the algorithm achieves rapid convergence, particularly in complex environments, and (2) environmental complexity paradoxically enhances search efficiency by providing richer tactile cues that accelerate hypothesis elimination. This phenomenon arises because intricate geometries introduce distinct contact signatures, enabling the algorithm to discard incorrect hypotheses faster than in simpler, less informative settings.

The core algorithm, designed for 2D localization, was extended to 3D through innovative hypothesis confirmation and rejection protocols. By decoupling positional and orientational search dimensions, our 3D implementation avoids the curse of dimensionality, achieving comparable computational complexity to the 2D case while improving robustness. Furthermore, we developed a probabilistic framework to address real-world challenges such as sensor noise and imperfect prior maps to maintain reliability under practical conditions. The probabilistic algorithm has demonstrated much greater resilience to noisy data and the environment compared to the deterministic approach. Apart from its higher computational demands, it exhibits no deflections relative to the deterministic algorithm. The method’s compatibility with established time-continuous search techniques was also demonstrated, enabling hybrid strategies that combine the precision of tactile search with the efficiency of motion-planning approaches.

The proposed algorithm is specifically designed for objects that contain multiple parallel surfaces at a distinct height. This design choice reflects common constraints in many real-world tasks, particularly in industrial and domestic assembly settings where such geometries are prevalent (e.g., sockets, ports, jigs, or fixture-based systems). Thus, this case is practically the most relevant.

While the current implementation focuses on positional localization, the architecture naturally extends to full 6-DOF pose estimation through systematic expansion of the hypothesis space. Future work will address orientational search by developing adaptive sampling strategies that exploit mechanical constraints in peg-hole interactions, potentially using hierarchical search spaces to manage computational complexity. Additional directions include integration with force-torque sensing for contact-rich environments and validation in industrial assembly tasks with variable friction and material properties.

The algorithm's ability to transform environmental complexity into a computational advantage suggests broad applicability beyond peg-in-hole scenarios. Potential applications range from micro-surgical robotics, where tactile feedback is critical, to space-constrained maintenance tasks in aerospace systems. By providing a framework that effectively bridges geometric priors, probabilistic reasoning, and tactile exploration, this work advances the frontier of contact-based robotic perception, offering a principled approach to manipulation in information-constrained environments.

Funding: The research leading to these results has received funding from the European Union's Horizon Europe Framework Programme under grant agreement No 101070596 (euROBIN).

Appendix

If the estimated map point \mathbf{p}_e^m converges towards the initial touch point \mathbf{p}_0^m , both given in the map coordinate system, then the final touch point $\mathbf{p}_r(k)$ is guaranteed to lie within the target region \mathcal{S}_g . The convergence of \mathbf{p}_e^m towards \mathbf{p}_0^m can be proven by showing that the search area $\mathcal{S}_s(k)$ is monotonically decreasing in each iteration step k and that the estimated point $\mathbf{p}_e^m \in \mathcal{S}_s(k)$. In the following we consider three regions: the selection region $\mathcal{S}_s(k)$ at iteration step k , the region touched by the robot $\mathcal{S}_t(k)$, and the target region \mathcal{S}_g . The proof relies on the following theorem:

Theorem 1 (Monotonic Convergence). *Let \mathcal{M} be a map consisting of disjoint regions \mathcal{S}_i , each corresponding to a unique horizontal face in 3D space. Suppose:*

- *The initial touch point $\mathbf{p}_0^m \in \mathcal{S}_s(0)$, which defines the initial region $\mathcal{S}_s(0) = \mathcal{S}_t(0)$.*
- *At each iteration step $k \geq 0$, the algorithm computes the displacement $\mathbf{d}^m = \mathbf{p}_g^m - \mathbf{p}_e^m$, $\mathbf{p}_e^m \in \mathcal{S}_s(k)$, and the robot touches the new region $\mathcal{S}_t(k+1)$.*
- *The candidate region is updated as:*

$$\mathcal{S}_s(k+1) = \{\mathbf{p}_j \in \mathcal{S}_s(k) \mid \mathbf{p}_j + \mathbf{d}^m \in \mathcal{S}_t(k+1)\}.$$

Then, for all $k \geq 0$:

1. $\mathcal{S}_s(k+1) \subsetneq \mathcal{S}_s(k)$ (strict subset property),
2. $\mathbf{p}_0^m \in \mathcal{S}_s(k)$ (the initial touch point given in the map coordinate system is contained in the selection region).

Proof.

We prove the lemma by induction on k , demonstrating that the selection region $\mathcal{S}_s(k)$ is strictly decreasing while always containing the initial touch point \mathbf{p}_0^m .

Base Case ($k = 0$)

By definition, an algorithm determines

$$\mathbf{p}_0^m \in \mathcal{S}_s(0), \mathcal{S}_s(0) = \mathcal{S}_t(0).$$

Thus, the lemma holds for $k = 0$.

Inductive Step

We now show these properties hold for $k+1$.

Step 1: Shrinking of Selection Region

The algorithm computes the displacement vector:

$$\mathbf{d}^m = \mathbf{p}_g^m - \mathbf{p}_e^m,$$

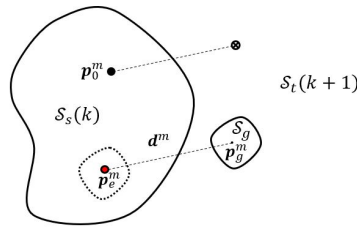


Figure 16. Registration process in k -th step. The region with dashed lines denotes the part that will be removed from $\mathcal{S}_s(k)$ in the next iteration.

where $\mathbf{p}_e^m \in \mathcal{S}_s(k)$ is the estimated point at iteration k . The newly detected region is denoted by $\mathcal{S}_t(k+1)$ and the candidate selection region is updated as:

$$\mathcal{S}_s(k+1) = \{\mathbf{p}_j \in \mathcal{S}_s(k) \mid \mathbf{p}_j + \mathbf{d}^m \in \mathcal{S}_t(k+1)\}.$$

Since by definition $\mathbf{p}_e^m \in \mathcal{S}_s(k)$ and $\mathbf{p}_e^m + \mathbf{d}^m \in \mathcal{S}_g$ and since $\mathcal{S}_g \cap \mathcal{S}_t(k+1) = \emptyset$, at least the point \mathbf{p}_e^m is excluded from $\mathcal{S}_s(k+1)$, $\mathbf{p}_e^m \notin \mathcal{S}_s(k+1)$. Thus at least one element is guaranteed to be excluded from the set $\mathcal{S}_s(k)$, which ensures

$$\mathcal{S}_s(k+1) \subsetneq \mathcal{S}_s(k).$$

Thus, the strict subset property holds.

Step 2: Containment of the Initial Touch Point

By induction assumption, the initial touch point satisfies $\mathbf{p}_0^m \in \mathcal{S}_s(k)$. If the newly touched region $\mathcal{S}_t(k+1)$ is equal to \mathcal{S}_g , the algorithm finishes as the correct point has been identified. If this is not the case, $\mathcal{S}_g \neq \mathcal{S}_t(k+1)$ and since $\mathcal{S}_g \cap \mathcal{S}_t(k+1) = \emptyset$, the initial touch point \mathbf{p}_0^m remains in $\mathcal{S}_s(k+1)$. This concludes the proof of the theorem.

Conclusion

By induction, we conclude that for all $k \geq 0$:

1. $\mathcal{S}_s(k+1) \subsetneq \mathcal{S}_s(k)$, ensuring monotonic shrinkage.
2. $\mathbf{p}_0^m \in \mathcal{S}_s(k)$, ensuring the true initial point is never eliminated.

The algorithm continues until the robot touches the goal region \mathcal{S}_g . Therefore, the algorithm converges to a sufficiently small search region \mathcal{S}_s containing \mathbf{p}_0^m , completing the proof. \square

References

1. Navarro, S.E.; Mühlbacher-Karrer, S.; Alagi, H.; Zangl, H.; Koyama, K.; Hein, B.; Duriez, C.; Smith, J.R. Proximity Perception in Human-Centered Robotics: A Survey on Sensing Systems and Applications. *IEEE Transactions on Robotics* **2022**, *38*, 1599–1620.
2. Zhuang, C.; Li, S.; Ding, H. Instance segmentation based 6D pose estimation of industrial objects using point clouds for robotic bin-picking. *Robotics and Computer-Integrated Manufacturing* **2023**, *82*. <https://doi.org/10.1016/j.rcim.2023.102541>.
3. Nottensteiner, K.; Sachtler, A.; Albu-Schäffer, A. Towards autonomous robotic assembly: Using combined visual and tactile sensing for adaptive task execution. *Journal of Intelligent & Robotic Systems* **2021**, *101*, 49.
4. Lončarević, Z.; Gams, A.; Reberšek, S.; Nemec, B.; Škrabar, J.; Skvarč, J.; Ude, A. *Robotics and Computer-Integrated Manufacturing* **2023**, *82*. <https://doi.org/10.1016/j.rcim.2023.102541>.
5. Saleh, K.; Szénási, S.; Vámosy, Z. Occlusion Handling in Generic Object Detection: A Review. In Proceedings of the IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMII), 2021, pp. 477–484.
6. Yi, A.; Anantrasirichai, N. A Comprehensive Study of Object Tracking in Low-Light Environments. *Sensors* **2024**, *24*, 4359. <https://doi.org/10.3390/s24134359>.
7. Eneuse, I.; Ibrahim, B.K.S.M.K.; Foo, M.; Matharu, R.S.; Ahmed, H. Accuracy evaluation of hand-eye calibration techniques for vision-guided robots. *PLOS One* **2022**, *17*, e0273261.

8. Liu, H.; Wu, Y.; Sun, F.; Guo, D. Recent progress on tactile object recognition. *International Journal of Advanced Robotic Systems* **2017**, *14*, 1–12.
9. Galaiya, V.R.; Asfour, M.; Alves de Oliveira, T.E.; Jiang, X.; Prado da Fonseca, V. Exploring Tactile Temporal Features for Object Pose Estimation during Robotic Manipulation. *Sensors* **2023**, *23*.
10. Abu-Dakka, F.; Nemec, B.; Jørgensen, J.A.; Savarimuthu, T.R.; Krüger, N.; Ude, A. Adaptation of manipulation skills in physical contact with the environment to reference force profiles. *Autonomous Robots* **2015**, *39*, 199–217.
11. Abu-Dakka, F.; Nemec, B.; Kramberger, A.; Buch, A.; Krüger, N.; Ude, A. Solving peg-in-hole tasks by human demonstration and exception strategies. *Industrial Robot* **2014**, *41*, 575–584.
12. Chen, F.; Cannella, F.; Sasaki, H.; Canali, C.; Fukuda, T. Error recovery strategies for electronic connectors mating in robotic fault-tolerant assembly system. In Proceedings of the IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA), 2014, pp. 1–6.
13. Marvel, J.A.; Newman, W.S. Assessing internal models for faster learning of robotic assembly. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2010, pp. 2143–2148.
14. Shetty, S.; Silvério, J.; Calinon, S. Ergodic Exploration Using Tensor Train: Applications in Insertion Tasks. *IEEE Transactions on Robotics* **2022**, *38*, 906–921.
15. Chhatpar, S.; Branicky, M. Localization for robotic assemblies with position uncertainty. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2003, pp. 2534–2540.
16. Chhatpar, S.; Branicky, M. Particle filtering for localization in robotic assemblies with position uncertainty. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2005, pp. 3610–3617.
17. Petrovskaya, A.; Khatib, O. Global Localization of Objects via Touch. *IEEE Transactions on Robotics* **2011**, *27*, 569–585.
18. Jasim, I.F.; Plapper, P.W.; Voos, H. Position Identification in Force-Guided Robotic Peg-in-Hole Assembly Tasks. *Procedia CIRP* **2014**, *23*, 217–222.
19. Hebert, P.; Howard, T.; Hudson, N.; Ma, J.; Burdick, J.W. The next best touch for model-based localization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2013, pp. 99–106.
20. Luo, S.; Mou, W.; Althoefer, K.; Liu, H. Localizing the object contact through matching tactile features with visual map. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 3903–3908.
21. Hauser, K. Bayesian Tactile Exploration for Compliant Docking With Uncertain Shapes. *IEEE Transactions on Robotics* **2019**, *35*, 1084–1096.
22. Bauza, M.; Valls, E.; Lim, B.; Sechopoulos, T.; Rodriguez, A. Tactile Object Pose Estimation from the First Touch with Geometric Contact Rendering. *arXiv:2012.05205* **2020**, [cs.RO], [arXiv:cs.RO/2012.05205].
23. Bauza, M.; Bronars, A.; Rodriguez, A. Tac2Pose: Tactile object pose estimation from the first touch. *The International Journal of Robotics Research* **2023**, *42*, 1185–1209.
24. Xu, J.; Lin, H.; Song, S.; Ciocarlie, M. TANDEM3D: Active Tactile Exploration for 3D Object Recognition. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 10401–10407.
25. Calandra, R.; Owens, A.; Upadhyaya, M.; Yuan, W.; Lin, J.; Adelson, E.H.; Levine, S. The Feeling of Success: Does Touch Sensing Help Predict Grasp Outcomes? *arXiv:1710.05512* **2017**, [cs.RO], [arXiv:cs.RO/1710.05512].
26. Yuan, W.; Dong, S.; Adelson, E.H. GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force. *Sensors* **2017**, *17*. <https://doi.org/10.3390/s17122762>.
27. Nemec, B.; Hrovat, M.M.; Simonič, M.; Shetty, S.; Calinon, S.; Ude, A. Robust Execution of Assembly Policies Using a Pose Invariant Task Representation. In Proceedings of the 20th International Conference on Ubiquitous Robots (UR), 2023, pp. 779–786.

28. Simonič, M.; Ude, A.; Nemec, B. Hierarchical Learning of Robotic Contact Policies. *Robotics and Computer-Integrated Manufacturing* **2024**, *86*. <https://doi.org/10.1016/j.rcim.2023.102657>.
29. So, P.; Sarabakha, A.; Wu, F.; Culha, U.; Abu-Dakka, F.J.; Haddadin, S. Digital Robot Judge: Building a Task-centric Performance Database of Real-World Manipulation With Electronic Task Boards. *IEEE Robotics & Automation Magazine* **2024**, *31*, 32–44.

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