Advantages of Using Two Robotic Arms for Tight Assemblies

Anonymous Author(s)

***Abstract*— We demonstrate the advantages of using two robotic arms simultaneously in tight assembly operations. We provide a new end-to-end framework which outputs an execution plan for the robotic arms given their digital CAD models. Our method is implemented in both simulation and real-life, with theoretical and empirical guarantees on its performance. In particular, we show that simultaneous movement of two arms significantly reduces execution time, produces higher quality trajectories, and accelerates the search time for valid robot placements. Furthermore, we provide guarantees on the bounds of the required dimensions of the robotic cell. Video clips of demonstrations with real-life robotic arms can be found in our project page.**

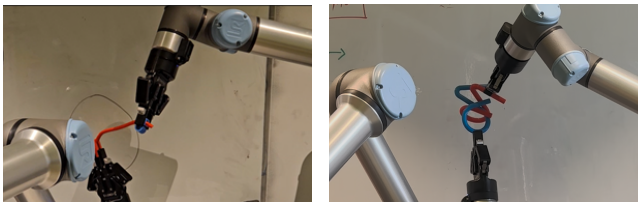
I. Introduction

In modern manufacturing, assembly consumes about 50% of the production time and 20% of total manufacturing costs [1], [2]. The growing demand for mass customization requires flexibility in production, often achieved during product assembly [3]. Consequently, Assembly Planning and robotic assembly have become prominent fields of research [4], [5].

Millions of industrial robots being one of the most fast growing industrial sectors [6], making the challenge of converting a digital plan of the parts into a final robotic arms trajectory a critical task. Yet, the control and programming of the commercially available industrial robots are limiting factors for their effective implementation, especially for dynamic production environments or when complex applications are required [7].

A recent work [8] presented a complete framework for converting a digital design of assembly parts into a single arm trajectory plan, coupled with the desired trajectory placement.

That work consisted of three main phases. First was the well studied free-flying objects motion planning [9]–[39], which to date is still challenging in tight environments. Second was coping with the Continuous path-wise IK (*CPW-IK*) which converted a continuous assembly trajectory in the work-space into a continuous trajectory in the joint-space overcoming challenges such as singularities, different IK branches, and collision avoidance. The IK itself is also a well studied challenge [40]–[49], however, there was a missing solution that could cope with the continuous task, perform it for the popular UR-like arms, and do so fast enough to enable the trajectory placement search. Hence a dynamic programming based solution coupled with analytic IK [42] was presented. Third was the challenge of *trajectory placement*. This is similar to the well studied *robot placement* problem [50], where valid locations of the robot's base are searched for each desired pose, or a set of poses, of the robot's end effector. Further work was done on mapping and testing robot placement for multiple isolated configurations [50] in order for the robot to be able to complete a path. However, the challenge solved in [8] was to place a full continuous trajectory in a complex $6D$ environment.



(a) Assembly 16505 (b) Alpha-z puzzle

Fig. 1: Simultaneous assembly using 2 robotic arms.

Other works provided a full cycle assembly planning for simpler tasks using a single robotic arm [8], [51], [52], or use 2 arms maintaining fixed transformation to hold an object [53]. And using 2 arms can boost the task dramatically as shown in [54].

However, to date there is no solution for coping with complex assembly trajectories in the workspace using two arms for actively manipulate the assembly.

In this work we demonstrate such a framework, enabling the advantages of using two robotic arms, such as further flexibility, accuracy, and speed, and enabling multiple assemblies within the same robotic cell.

The contribution of this work is as follows: (1) extending the framework presented in [8] into 2 robotic arms, including extension of its main algorithm *CPW-IK* (2) perform experiments indicating improvements in assembly time, trajectory accuracy, and search time (3) guarantee as to the space occupied by the solution (4) real world demonstrations of the above solutions in action.

Our project page is available at: [TDB].

II. Preliminaries and Problem Statement

In this section we summarize tools from [8] and present the problem statement.

A. Motion Planning of Free Flying Objects

The input to this phase is a free flying assembly problem. That is, digital models of two free-flying bodies B_1 and B_2, as well as start and goal positions $\bar{q}_{\text{start}}$ and $\bar{q}_{\text{goal}}$ for one of the bodies, assuming the other one stays still at the origin, as presented in Section II-D. We deploy an appropriate motion planning algorithm, e.g., as described in [26], [34], [38]. Specifically, in this work we use the TR-RRT algorithm from our previous work [26], as it best performs on the dataset presented in [34], which is also in use in the current work.

Most sampling based algorithms, including those in [26], [34], [38] suffer from two undesired properties. First, they

usually allow some small penetration between the sub-assemblies. This works perfectly in simulation, but is challenging for the physical world. Second is that it may be, as is often the case with sampling-based plans, unnecessarily long and jagged. We therefore follow the free-flying motion planning algorithm with a post processing phase. This includes the *retract* function from [26], to remove the penetrations, followed by a smoothing function as described in [8].

The output of this phase is a piecewise linear path $\bar{\gamma}$, that is, a collision free path from $\bar{q}_{\text{start}} \in \bar{\mathcal{C}}_{\text{free}}$ to $\bar{q}_{\text{goal}} \in \bar{\mathcal{C}}_{\text{free}}$, i.e., a path $\bar{\gamma}:[0,1] \to \bar{\mathcal{C}}_{\text{free}}$ such that $\bar{\gamma}(0) = \bar{q}_{\text{start}}$ and $\bar{\gamma}(1) = \bar{q}_{\text{goal}}$. The full sequence is

$$\bar{q}_{\text{start}} = \bar{q}_0, \bar{q}_1, \ldots, \bar{q}_{N-1}, \bar{q}_N = \bar{q}_{\text{goal}}, \tag{1}$$

and the segment between each pair of consecutive configurations in the sequence, such that $\bar{\gamma}$ is contained in $\bar{\mathcal{C}}_{\text{free}}$.

B. Continuous Path-wise IK for a Single Robotic Arm

In this section, we summarize the method presented in [8]. One major difficulty of transferring a trajectory from $\mathrm{SE}(3)$ to the joint-angles space is finding a consistent branch of inverse-kinematics [44]. For each end-effector pose in $\mathrm{SE}(3)$, the six-DoF UR-like robotic arm has up to eight possible different inverse-kinematics solutions, which one can compute analytically.

For each pose in the trajectory for the free flying objects, we calculate all possible IK solutions. We construct a layered directed acyclic graph $G=(V,E)$, where layer L_k contains the feasible IK solutions of pose k in the original trajectory. We connect a node $a \in L_k$ with a directed edge e to node $b \in L_{k+1}$ if and only if their L^1 distance in the joint-angle space is smaller than some predefined threshold δ. The weight of the edge is that same L^1 distance $w(e) = ||a-b||_{L^1}$. We also add two dummy nodes s,t to the graph, and connect them with zero-weight edges to the nodes of L_1 and L_N respectively, with N being the number of poses in the original trajectory. Denote by $\Pi_G(s,t)$ all paths from s to t in the graph G.

We find a path γ whose heaviest edge has the smallest weight. That is, we take γ:

$$\gamma = \mathrm{argmin}_{\rho \in \Pi_G(s,t)} \left\{ \max_{e \in \rho} w(e) \right\} \tag{2}$$

We define the *error* of the trajectory γ as follows: Let $p \in \partial A$ be any point on the boundary of the sub-assembly manipulated by the robotic arm. Let $\gamma_p, \bar{\gamma}_p : [0,1] \to \mathbb{R}^3$ be the trajectories of p after following the paths γ and $\bar{\gamma}$, respectively. The one-sided Hausdorff distance between γ_p and $\bar{\gamma}_p$ is $d(\gamma_p, \bar{\gamma}_p) = \max_{x \in \gamma_p} \min_{y \in \bar{\gamma}_p} ||x-y||$. Then the error of the path γ is the largest one-sided Hausdorff distance of any point $p \in \partial A$:

$$\mathrm{Err}(\gamma) = \max_{p \in \partial A} d(\gamma_p, \bar{\gamma}_p). \tag{3}$$

Our goal is to bound this error by some arbitrarily small ε.

Denote by D the the maximal distance of a point in our moving system (the robot and the dynamic body attached to it) from the robot's base. Then [8] shows that taking $\delta_\gamma \leq \frac{\varepsilon}{D}$ yield the desired error $\mathrm{Err}(\gamma) \leq \varepsilon$.

C. Trajectory Placement for a Single Robotic Arm

Note that we always assume that, without loss of generality, $\bar{q}_{static} = 1 \in \mathrm{SE}(3)$. We call this choice of initial pose the *placement* of the trajectory. However, not all placements have a valid corresponding trajectory in joint-angle space.

For a single robotic arm, choosing merely the initial pose decides the entire placement of the trajectory. Hence, for a single robotic arm, we begin by randomly sampling a six-dimensional starting pose, and run the CPW-IK algorithm. If successful, a valid placement for the trajectory was found, otherwise, we sample a new initial pose and repeat the process. We can further narrow down the search, as described in [26].

D. Problem Statement

The input to the framework is a CAD of two rigid bodies to be assembled, together with desired start and goal relative poses in $\mathrm{SE}(3)$. The output is detailed instructions, in joint-angle space, of how each of two robotic arms should move in order to perform the assembly. This includes placement of the pieces at the beginning of the assembly, grasping poses, and trajectory of each robotic arm.

More formally, we start with two rigid bodies $B_1, B_2 \subseteq \mathbb{R}^3$, which we wish to assemble. We assume that the bodies are placed in a pose aligned with their grasp by a robotic arm. In the CAD software however, they may be posed differently, with offsets of $q_{B_1}, q_{B_2} \in \mathrm{SE}(3)$ for B_1, B_2, respectively.

Hence, when planning for free flying objects, we can assume that B_1 is fixed at the pose q_{B_1}, and that B_2 is a dynamic object that we move from q_{B_2} to $\bar{q}_{\text{goal}} \cdot q_{B_2}$.

The configuration space (C-space) of a single UR-like robotic arm is $\mathcal{C} = [0, 4\pi)^6 \subseteq \mathbb{R}^6$. The forward kinematics function $f : \mathcal{C} \to \mathrm{SE}(3)$ maps a configuration $c \in C$ to its corresponding workspace pose $\bar{q} = f(c)$ of the end-effector.

When two robotic arms R_1, R_2 are in play, the combined configuration space is the product $\mathcal{C}^2 = \mathcal{C} \times \mathcal{C} \subset \mathbb{R}^{12}$. That is $c^2 = (c^{(1)}, c^{(2)}) \in \mathcal{C}^2$ is a pose such that the arm R_1 is at configuration $c^{(1)}$ and the arm R_2 at configuration $c^{(2)}$.

We say that $c^2 \in \mathcal{C}^2_{forbid} \subseteq \mathcal{C}^2$ if any two of R_1, R_2, B_1, B_2 are in intersection among themselves or with an environment obstacle.

We define the *free region* as all configurations that yield no intersection of the interiors of the robots, the objects, and the environment obstacles:

$$\mathcal{C}^2_{free} = \left(\mathcal{C}^2 \setminus \mathcal{C}^2_{forbid} \right). \tag{4}$$

We are now ready to define our problem.

Problem Statement: Assume we are given CAD of rigid bodies $B_1, B_2 \subseteq \mathbb{R}^3$, their offsets $q_{B_1}, q_{B_2} \in \mathrm{SE}(3)$, and the goal pose $\bar{q}_{\text{goal}}$ of B_2. We aim to find a collision-free path of motion $\gamma = (\gamma^{(1)}, \gamma^{(2)}) : [0,1] \to \mathcal{C}^2_{\text{free}}$ such that:

$$f\left(\gamma^{(2)}(0)\right)^{-1} \cdot f\left(\gamma^{(1)}(0)\right) = q_{B_2}^{-1} \cdot q_{B_1}, \tag{5}$$

$$f\left(\gamma^{(2)}(1)\right)^{-1} \cdot f\left(\gamma^{(1)}(1)\right) = (\bar{q}_{\text{goal}} \cdot q_{B_2})^{-1} \cdot q_{B_1}. \tag{6}$$

We note that a choice for such a path γ also incorporates a choice for the placement of the rigid bodies in the

workspace. That is, γ may be any valid trajectory in joint-angle space that starts with assembled pieces and ends with dis-assembled pieces.

In this work we restrict ourselves to paths γ that are piecewise linear, that is, a sequence:

$$c^2_{\text{start}} = c^2_0, c^2_1, \ldots, c^2_{N-1}, c^2_N = c^2_{\text{goal}}, \tag{7}$$

and the segment between each pair of consecutive configurations in the sequence, such that γ is contained in $\mathcal{C}_{\text{free}}$.

Within this general definition of γ, we explicitly implement three types of trajectories. First is a single arm trajectory as presented in [8], where arm R_1 keeps body B_1 static, and arm R_2 does all the work with B_2. Second is *simultaneous dual-arm*, where at each step both of the arms move. And third, which we refer to as *alternating dual-arm*, where at every step only R_1 moves, only R_2 moves, or both arms move.

III. Methods and Guarantees

In this section we show how the CPW-IK method can be extended for two arms, for the simultaneous dual-arm case as well as for the alternating dual-arm case. For all cases, we assume that we have a pre-computed path $\bar{\gamma}:[0,1]\to \mathrm{SE}(3)$ for the rigid part B_2, which constitutes a valid motion plan for the free flying parts (assuming B_1 stays put). We aim to find a corresponding path $\gamma:[0,1]\to\mathcal{C}^2$.

A. CPW-IK for Simultaneous Dual-Arm

First, we sample some random placement. That is, we sample some random $q \in \mathrm{SE}(3)$ which will be $q = f\left(\gamma^{(2)}(0)\right)^{-1} \cdot f\left(\gamma^{(1)}(0)\right)$. We multiply the path of the free-flying objects $\bar{\gamma}$ by this q.

For each step $\bar{q}_i \mapsto \bar{q}_{i+1}$ in the path $\bar{\gamma}$ we do as follows:

- Calculate the midpoint between $\bar{q}_i$ and $\bar{q}_{i+1}$.
- Find all IK solutions for that midpoint pose, and add those solutions as a layer in the dynamic DAG, for the rigid part B_2.
- Find the corresponding pose for part B_1 in $\mathrm{SE}(3)$ by multiplying the new pose of B_2 by the difference between that same pose in $\bar{\gamma}$ and q_{B_1}. Find all possible IK solutions for that pose and add the layer to the DAG, for the rigid part B_1.
- Each edge in these layers, corresponds to simultaneous movement of the robotic arms R_1 and R_2 towards that midpoint.

After building the DAG, we use the same dynamic programming and choose a path minimizing the Hausdorff distance or the makespan as in Section II-B.

B. CPW-IK for Alternating Dual-Arm

First, we decide the alternation intervals in advance. That is, we split the interval $[0,1]$ to $[0,1] = \bigcup_{j=0}^{m-1}[t_j, t_{j+1}]$, with $0 = t_0 < t_1 < \cdots < t_j < \cdots < t_{m-1} < t_m = 1$. On each interval $[t_j, t_{j+1}]$, exactly one of the following motion patterns is maintained: (i) only arm R_1 moves; (ii) only arm R_2 moves; or (iii) both arms R_1 and R_2 move simultaneously. In this work, we split $[0,1]$ into m equal intervals, although a more elaborate search can be performed.

We start with the first interval $[t_0, t_1]$, and evaluate all three options; pattern (i) and (ii) are calculated with our technique for a single robotic arm, while (iii) is calculated using the method shown in Section III-A. Note that each pattern yields a different pose in joint-angle space at time t_1.

We then recursively construct a search tree, connecting each node, that ends in time t_j, to all three possible motion patterns of the following interval $[t_j, t_{j+1}]$.

Finally, similar to Section II-B, we find the route in the search tree whose heaviest edge has the smallest weight. We can optimize over various metrics by choosing a weight function over the edges. In this work, we also optimize over makespan.

C. Robotic Cell Size Guarantee

In this section, we provide guarantees regarding the space that the dual-arm robotic cell will occupy. This is based solely on the components involved and the positions of the robotic arms, not the specific trajectory.

Understanding the dimensions of the robotic cell is crucial when deploying robotic arms in factories and workshops. It ensures safety, facilitates collision avoidance, and optimizes the use of floor space.

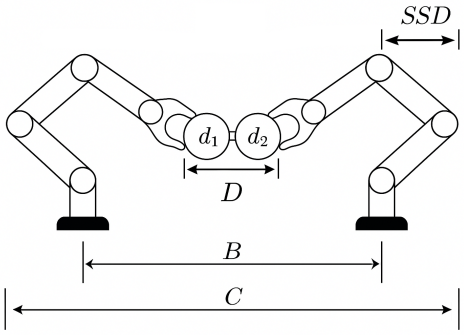


Fig. 2: Dimensions of a robotic cell. B is the distance between the bases of the robotic arms. C is the maximal width occupied by the robotic cell. SSD is the guaranteed maximal deviation of a single arm beyond its base. d_1 and d_2 are the maximal diameters of sub-assembly 1 and sub-assembly 2. D is the sum of d_1 and d_2

As demonstrated in Section IV, valid trajectory placements using two robotic arms occur much more frequently than when using a single arm. Therefore, we can limit our analysis to trajectory placements on the plane that is normal to the line connecting the bases of the two arms and passes through the midpoint between them. We assume that after each step, the entire assembly is translated back to this plane (in practice, we combine this back-shift with the step itself). This allows the analysis of the robotic cell width to depend solely on the lengths of the robotic arms (arm_length) (see Figure 2), the distance between the bases of the arms (B), and the combined diameter of the assembly pieces (D). The diameter D can be calculated simply as the sum of the

TABLE I: The minimum total running time (makespan) over 100 valid placements, expressed in degrees (as angular speed can be decided), for each assembly. The *Single-arm* row refers to settings where one object is static, and only a single arm moves. The *Dual-arm* row refers to settings where both arms are moving simultaneously. *Time saved* is the time saved when using two robotic arms, compared to a single one.

Assembly	az	abc	16505	06397	*Average*
Single-arm	425.5	389.3	81.7	886.7	—
Dual-arm	402.0	338.6	62.7	745.8	—
Time saved	**5.5%**	**13.0%**	**23.3%**	**15.9%**	**14.4%**

diameters of the two sub-assemblies (d_1,d_2), though a tighter bound may assess the diameter of the assembly at each step. Let C represent the width of the robotic cell. Then:

$$C = \frac{B}{2} + \frac{D}{2} + arm\_length, \tag{8}$$

whose minimum over the arms distance B, under the assumption that $D \leq B$ (otherwise further restrictions on the trajectory may be required), is achieved at $B = D$:

$$C_{min} = D + arm\_length. \tag{9}$$

Further, in case the placements of the arms in the robotic cell are pre-defined and we are only interested in the single-side deviation (SSD) of the robotic arms out of the given robotic cell, we receive:

$$SSD = \frac{D}{4} - \frac{B}{4} + \frac{arm\_length}{2} \tag{10}$$

IV. Experiments and Results

In this section we demonstrate the advantage of using our framework with two robotic arms over a single arm. Video clips for execution of all assemblies are available in the project page. The link will be available at the end of Section I.

A. Dataset

We used the tight assembly dataset presented in [8], and compared the results of using two arms to those using a single arm. The six assemblies in that dataset were selected from a larger dataset [34], keeping 1-2 challenging samples from each family. These are tight assemblies that require non-trivial combination of translation and rotation in order to be assembled, hence proved to be the hardest for the algorithms tested. In this work, as a proof of concept, we demonstrate the solutions, and measure the results, of four representative samples.

TABLE II: The average Hausdorff distance over 10 valid placements, presented in total radians traveled per time steps, for each assembly. *Error reduction* is the percentage of average reduction in terms of Hausdorff distance using two robotic arms, compared to a single one

Assembly	az	abc	16505	06397	*Average*
Single-arm	0.035	0.041	0.028	0.121	—
Dual-arm	0.027	0.023	0.017	0.103	—
Error reduction	**22.9%**	**43.9%**	**39.3%**	**14.9%**	**30.2%**

TABLE III: Successful placements ratio. *Improvement* is the ratio between the two-arms success and the single-arm success.

Assembly	az	abc	16505	06397	*Average*
Single-arm	0.021	0.124	0.194	0.033	—
Dual-arm	0.225	0.253	0.286	0.235	—
Improvement	**x10.71**	**x2.04**	**x1.47**	**x7.12**	**x5.34**

B. Implementation details

We used the TR-RRT [26] motion planning algorithm for the free-flying sub-assemblies.

We demonstrate our method on a pair of *Universal Robots UR5e* robotic arms, equipped with *Robotiq 2F-85* grippers.

The rest of the process is as follows. We begin with a free-flying trajectory $\bar{\gamma}$ and continue with running the various algorithms seeking for valid placements. We then design the parts themselves in Blender, add a small mark or cutout at the desired grasping point, and print them using *Creality Ender-3 S1* 3D printer with 0.2 mm resolution. Finally, we grasp the parts using the end effectors and execute the generated trajectory γ.

C. Results

The experiments demonstrated three main findings. First, the total assembly time (*makespan*) is consistently reduced when using two arms compared to using a single arm as can be seen in table I.

Second, the quality of the paths, in terms of uni-directional Hausdorff distance, is improved, as can be seen in table II.

Third, the trajectory placement algorithm performs faster, as demonstrated in table III in terms of frequency of valid trajectories.

V. Discussion

In this work we present a framework that converts challenging assembly tasks into real world assembly plans using two robotic arms. We provide quality guarantees for the executed trajectory, as well as guarantee for the space this robotic cell will occupy during assembly.

We have also presented quantitative advantages of using two robotic arms over a single arm, in terms of execution speed, trajectory accuracy, and search time.

We continue with this research in two directions. First is further optimizing the simultaneous work of two arms by better balancing the work between them. Second is using this ability to plan manipulation of two arms simultaneously, into solving three handed assembly problems (see figure 3).

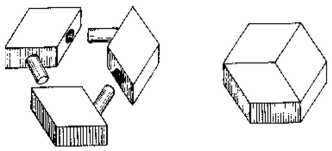


Fig. 3: Three handed assembly, source: johnrausch.com

REFERENCES

[1] J. Fan and J. Dong, "Intelligent virtual assembly planning with integrated assembly model," in *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*, vol. 5, 2003, pp. 4803–4808 vol.5.

[2] K.-C. Ying, P. Pourhejazy, C.-Y. Cheng, and C.-H. Wang, "Cyber-physical assembly system-based optimization for robotic assembly sequence planning," *Journal of Manufacturing Systems*, vol. 58, pp. 452–466, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612521000042

[3] J. Michniewicz, G. Reinhart, and S. Boschert, "CAD-based automated assembly planning for variable products in modular production systems," *Procedia CIRP*, vol. 44, pp. 44–49, 2016.

[4] S. Ghandi and E. Masehian, "Review and taxonomies of assembly and disassembly path planning problems and approaches," *Computer-Aided Design*, vol. 67, pp. 58–86, 2015.

[5] Y. Jiang, Z. Huang, B. Yang, and W. Yang, "A review of robotic assembly strategies for the full operation procedure: planning, execution and evaluation," *Robotics and Computer-Integrated Manufacturing*, vol. 78, p. 102366, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584522000540

[6] A. Dzedzickis, J. Subačiūtė-Žemaitienė, E. Šutinys, U. Samukaitė-Bubnienė, and V. Bučinskas, "Advanced applications of industrial robotics: New trends and possibilities," *Applied Sciences*, vol. 12, no. 1, p. 135, 2021.

[7] P. Bilancia, J. Schmidt, R. Raffaeli, M. Peruzzini, and M. Pellicciari, "An overview of industrial robots control and programming approaches," *Applied Sciences*, vol. 13, no. 4, p. 2582, 2023.

[8] D. Livnat, Y. Lavi, and D. Halperin, "A full-cycle assembly operation: From digital planning to trajectory execution using a robotic arm," in *Accepted to Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2025*, 2025, accepted for publication, to be presented at ICRA 2025.

[9] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-only robot navigation in a neural radiance world," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022.

[10] M. J. Bency, A. H. Qureshi, and M. C. Yip, "Neural path planning: Fixed time, near-optimal path generation via oracle imitation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3965–3972.

[11] F. Dai, A. Wahrburg, B. Matthias, and H. Ding, "Robot assembly skills based on compliant motion," in *Proceedings of ISR 2016: 47st International Symposium on Robotics*. VDE, 2016, pp. 1–6.

[12] L. E. Kavraki, M. N. Kolountzakis, and J. Latombe, "Analysis of probabilistic roadmaps for path planning," in *Proceedings of the 1996 IEEE International Conference on Robotics and Automation, Minneapolis, Minnesota, USA, April 22-28, 1996*. IEEE, 1996, pp. 3020–3025. [Online]. Available: https://doi.org/10.1109/ROBOT.1996.509171

[13] O. Salzman, M. Hemmer, B. Raveh, and D. Halperin, "Motion planning via manifold samples," *Algorithmica*, vol. 67, no. 4, pp. 547–565, 2013. [Online]. Available: https://doi.org/10.1007/s00453-012-9736-1

[14] M. Kleinbort, K. Solovey, Z. Littlefield, K. E. Bekris, and D. Halperin, "Probabilistic completeness of RRT for geometric and kinodynamic planning with forward propagation," *IEEE Robotics Autom. Lett.*, vol. 4, no. 2, pp. 277–283, 2019. [Online]. Available: https://doi.org/10.1109/LRA.2018.2888947

[15] O. Salzman, M. Hemmer, and D. Halperin, "On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages," *IEEE Trans Autom. Sci. Eng.*, vol. 12, no. 2, pp. 529–538, 2015. [Online]. Available: https://doi.org/10.1109/TASE.2014.2331983

[16] J. De Schutter and H. Van Brussel, "Compliant robot motion I. A formalism for specifying compliant motion tasks," *The International Journal of Robotics Research*, vol. 7, no. 4, pp. 3–17, 1988.

[17] J. Friedman, J. Hershberger, and J. Snoeyink, "Efficiently planning compliant motion in the plane," *SIAM Journal on Computing*, vol. 25, no. 3, pp. 562–599, 1996.

[18] H. Ha, J. Xu, and S. Song, "Learning a Decentralized Multi-arm Motion Planner," in *Conference on Robotic Learning (CoRL)*, 2020.

[19] D. Hsu, T. Jiang, J. Reif, and Z. Sun, "The bridge test for sampling narrow passages with probabilistic roadmap planners," in *2003 IEEE international conference on robotics and automation (cat. no. 03CH37422)*, vol. 3. IEEE, 2003, pp. 4420–4426.

[20] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1478–1483.

[21] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[22] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[23] S. Klemm, J. Oberländer, A. Hermann, A. Roennau, T. Schamm, J. M. Zollner, and R. Dillmann, "RRT*-connect: Faster, asymptotically optimal motion planning," in *2015 IEEE international conference on robotics and biomimetics (ROBIO)*. IEEE, 2015, pp. 1670–1677.

[24] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[25] T. Lefebvre, J. Xiao, H. Bruyninckx, and G. De Gersem, "Active compliant motion: a survey," *Advanced Robotics*, vol. 19, no. 5, pp. 479–499, 2005.

[26] D. Livnat, M. M. Bilevich, and D. Halperin, "Tight motion planning by riemannian optimization for sliding and rolling with finite number of contact points," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 14 333–14 340.

[27] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng, "A fast RRT algorithm for motion planning of autonomous road vehicles," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 1033–1038.

[28] M. A. Peshkin, "Programmed compliance for error corrective assembly," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 473–482, 1990.

[29] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2118–2124.

[30] S. Ruan, K. L. Poblete, H. Wu, Q. Ma, and G. S. Chirikjian, "Efficient path planning in narrow passages for robots with ellipsoidal components," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 110–127, 2022.

[31] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2016.

[32] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. H. Reif, "Narrow passage sampling for probabilistic roadmap planning," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1105–1115, 2005.

[33] G. Thomas, M. Chien, A. Tamar, J. A. Ojea, and P. Abbeel, "Learning robotic assembly from cad," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3524–3531.

[34] Y. Tian, J. Xu, Y. Li, J. Luo, S. Sueda, H. Li, K. D. Willis, and W. Matusik, "Assemble them all: Physics-based planning for generalizable assembly by disassembly," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 6, pp. 1–11, 2022.

[35] W. Wang, L. Zuo, and X. Xu, "A learning-based multi-RRT approach for robot path planning in narrow passages," *Journal of Intelligent & Robotic Systems*, vol. 90, pp. 81–100, 2018.

[36] J. Wang, T. Zhang, N. Ma, Z. Li, H. Ma, F. Meng, and M. Q.-H. Meng, "A survey of learning-based robot motion planning," *IET Cyber-Systems and Robotics*, vol. 3, no. 4, pp. 302–314, 2021.

[37] N. Widulle and O. Niggemann, "Using Reverse Reinforcement Learning for Assembly Tasks," in *PRL Workshop Series –Bridging the Gap Between AI Planning and Reinforcement Learning*, 2023.

[38] X. Zhang, R. Belfer, P. G. Kry, and E. Vouga, "C-space tunnel discovery for puzzle path planning," *ACM Trans. Graph.*, vol. 39, no. 4, aug 2020. [Online]. Available: https://doi.org/10.1145/3386569.3392468

[39] S. Zickler and M. M. Veloso, "Efficient physics-based planning: Sampling search via non-deterministic tactics and skills." in *AAMAS (1)*, 2009, pp. 27–33.

[40] S. Kucuk and Z. Bingul, "Inverse kinematics solutions for industrial robot manipulators with offset wrists," *Applied Mathematical Modelling*, vol. 38, no. 7-8, pp. 1983–1999, 2014.

[41] J. Zhao and N. I. Badler, "Inverse kinematics positioning using nonlinear programming for highly articulated figures," *ACM Transactions on Graphics (TOG)*, vol. 13, no. 4, pp. 313–336, 1994.

[42] J. Villalobos, I. Y. Sanchez, and F. Martell, "Singularity analysis and complete methods to compute the inverse kinematics for a 6-dof ur/tm-type robot," *Robotics*, vol. 11, no. 6, 2022. [Online]. Available: https://www.mdpi.com/2218-6581/11/6/137

[43] ——, "Alternative inverse kinematic solution of the ur5 robotic arm," in *Advances in Automation and Robotics Research*, H. A. Moreno, I. G. Carrera, R. A. Ramírez-Mendoza, J. Baca, and I. A. Banfield, Eds. Cham: Springer International Publishing, 2022, pp. 200–207.

[44] L.-T. Schreiber and C. Gosselin, "Determination of the Inverse Kinematics Branches of Solution Based on Joint Coordinates for Universal Robots-Like Serial Robot Architecture," *Journal of Mechanisms and Robotics*, vol. 14, no. 3, p. 034501, 11 2021. [Online]. Available: https://doi.org/10.1115/1.4052805

[45] T. Ho, C.-G. Kang, and S. Lee, "Efficient closed-form solution of inverse kinematics for a specific six-dof arm," *International Journal of Control, Automation and Systems*, vol. 10, pp. 567–573, 2012.

[46] J. Li, H. Yu, N. Shen, Z. Zhong, Y. Lu, and J. Fan, "A novel inverse kinematics method for 6-dof robots with non-spherical wrist," *Mechanism and Machine Theory*, vol. 157, p. 104180, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X20303979

[47] M. H. FarzanehKaloorazi and I. A. Bonev, "Singularities of the typical collaborative robot arm," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 51814. American Society of Mechanical Engineers, 2018, p. V05BT07A086.

[48] D. Rakita, B. Mutlu, and M. Gleicher, "Stampede: A discrete-optimization method for solving pathwise-inverse kinematics," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3507–3513.

[49] Y. Wang, C. Sifferman, and M. Gleicher, "Iklink: End-effector trajectory tracking with minimal reconfigurations," 2024. [Online]. Available: https://arxiv.org/abs/2402.16154

[50] A. Makhal and A. K. Goins, "Reuleaux: Robot base placement by reachability analysis," in *2018 second IEEE international conference on robotic computing (IRC)*. IEEE, 2018, pp. 137–142.

[51] W. Wan, K. Harada, and K. Nagata, "Assembly sequence planning for motion planning," *Assembly Automation*, vol. 38, no. 2, pp. 195–206, 2018.

[52] B. Tang, I. Akinola, J. Xu, B. Wen, A. Handa, K. Van Wyk, D. Fox, G. S. Sukhatme, F. Ramos, and Y. Narang, "Automate: Specialist and generalist assembly policies over diverse geometries," *arXiv preprint arXiv:2407.08028*, 2024.

[53] T. Cohn, S. Shaw, M. Simchowitz, and R. Tedrake, "Constrained bimanual planning with analytic inverse kinematics," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 6935–6942.

[54] R. Shome, K. Solovey, J. Yu, K. Bekris, and D. Halperin, "Fast, high-quality two-arm rearrangement in synchronous, monotone tabletop setups," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 888–901, 2021.