Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 136 (2025) 302–307

www.elsevier.com/locate/procedia

35th CIRP Design 2025

6-DoF Robot Base Positioning Methodology in Manufacturing Cells Using Inverse Reachability (ARBP-IR)

Caner-Veli Ince[a]*, Jan Peter Niestroj[a], Annika Raatz[a]

[a]*Institute of Assembly Technology and Robotics, Leibniz University Hannover, An der Universität 2, 30823 Garbsen, Germany*

* Corresponding author. Tel.: +49 511 762 18247. *E-mail address:* ince@match.uni-hannover.de

Abstract

6-DoF robots are essential in factory automation, but determining their optimal base position relies on user intuition, making the process time-consuming, error-prone, and skill-dependent. There are discrete event simulation programs (DES) that support the shop floor layout design. However, DES often lack automated base positioning functions for industrial robots. This article presents a method for automated base placement of 6-DoF robots to reduce the effort for their implementation in manufacturing cells, specifically in automated layout creation of forging cells and processes. Forging cells present specific challenges, primarily due to the immobility of their components, which cannot be rearranged because of their significant mass. Given their long operational lifespans traditional forges are not inherently designed to integrate robotic systems. To address these challenges, an extensive literature review was conducted to assess the current state of automated robot base positioning methodologies. Although several methods and approaches exist, none sufficiently meet the needs of industrial applications within forging cells, as many originate from domains such as service or mobile robotics. The most promising methods were subsequently selected, merged, and refined to tailor them for the specific context of forging cell automation using industrial robots. The resulting positioning methodology is based on the principle of inverse reachability and has been implemented within the Visual Components software environment. Evaluation of the developed algorithm demonstrates a significant reduction in the time required to identify suitable base positions for complex cell design tasks.

Keywords: Automation; Tailored Forming; Forging

1. Introduction

Automation in production is one solution to address cost minimization and the upcoming labor shortage in European countries. In particular, the implementation of 6-DoF (Degrees of Freedom) robots has become well-established in this context [1]. The implementation of 6-DoF robots require the precise positioning of the robot base within the manufacturing cell. This strategic placement is critical for ensuring that the robot can effectively access all required positions for the designated task. Furthermore, an optimally positioned robot base contributes to enhanced operational efficiency by reducing cycle times, minimizing energy consumption, and decreasing mechanical wear [2]. The initial positioning is typically achieved through a process of trial-and-error, even in instances where digital tools are available to assist with the process of determining the optimal position [3]. The success and efficiency of position determination rely heavily on the engineer's skill and experience. Additionally, the process is time-intensive, which conflicts with the goals of cost reduction and addressing labor shortages.

Becoming independent of the engineer's skills and experience is highly desirable, especially considering the anticipated labor shortage. Therefore, this paper aims to develop a methodology that automates the robot base positioning of a 6-DoF robot within a given 3D model of a manufacturing cell. Furthermore, to enhance the usability of this methodology, it will be integrated into commercial 3D manufacturing simulation software. A challenging use case is the automation of forging cells, because they were not

2212-8271
10.1016/j.procir.2025.08.053

originally designed for automation. For the automation of these cells, one option is to rearrange the machines. However, this is challenging, as the forge or other components cannot be moved easily. Alternatively, robots, depending on their size, are easier to relocate, making this the preferable option. The automation of forging cells is one of the aims of the collaborative research center (CRC) 1153– 'Tailored Forming', which is why overall development took place within the CRC. The CRC's general goal is to develop a novel manufacturing process for hybrid bulk components by tailoring them especially for their use case and implementing further functionalities [4].

A brief overview of current methods for autonomous robot base placements is provided. On the basis of these fundamentals, the requirements for the development of the algorithm are outlined, and the algorithm itself is subsequently introduced. Next, the implementation within a software environment is demonstrated. To validate its functionality, the algorithm is tested using a digital model of a manufacturing cell from the CRC. Finally, the work is concluded with a summary and an outlook on future research directions.

2. Literature Survey of Robot Base Positioning

The state of the art will give a brief overview of current methods for robot positioning to identify strengths and weaknesses for the described use case of the CRC. Robot base placement is a field that has been extensively researched. As early as the 1990s, Hwang & Watterberg [5], and Barral et al. [6] laid the foundation for further studies with their work on cycle time reduction through robot placement. However, an established automated methodology for robot placement has yet to emerge.

Asokan et al. investigated how to position an underwater robotic vehicle so that all points of interest would be within the robot's workspace [7]. A point of interest (POI) defines the specific coordinates that the robot must reach in order to successfully execute a handling task. They describe the general positioning strategy as follows: Firstly, defining manipulator kinematics, workspace, and boundaries. Secondly, moving the vehicle so that all POIs are within the workspace. Thirdly, optimizing the position and finally checking for singular positions and adjusting the base position as needed. For the optimization, they consider the manipulator's dexterity, which is calculated from the kinematic conditioning index Ψ of the manipulator. A limitation of this approach is that it does not account for collisions, based on the assumption that the robot can position itself freely.

Mitsi et al. develops another method that combines a genetic algorithm (GA) and a quasi-Newton algorithm (QNA) [8]. They aim to find the ideal spot for the robot to reach predefined POIs while not violating joint limits and avoiding singularities. Furthermore, the method tries to maximize the manipulability ω with J as the Jacobian matrix.

$$\omega = \sqrt{det(J * J^T)} \quad (1)$$

A GA is effective in finding solutions without getting stuck in local extrema, but they are only efficient when applied to problems with a limited scope or a small number of variables. In contrast, QNA is effective at identifying local extrema when considering a variety of parameters. The GA is employed to explore the global solution space in a broad manner, where the parameters it identifies serve as initial values for QNA. The QNA then precisely identifies local extrema. This combined approach proves to be more efficient than employing either algorithm in isolation. The combination of a GA and QNA also brings together their advantages

Zacharias et al. proposed an alternative method [9], which utilizes cross-correlation and a reachability map. The reachability map defines the robotic arm's workspace by discretizing points in relation to the robot's base coordinates, specifying the directions from which each point can be accessed. This data is combined into a 3D map that shows the robot arm's capabilities. After creating this map, the necessary path is calculated to complete a desired handling task. The reachability map is then overlaid onto the path and adjusted until the path is fully covered by reachable points. By examining the position and orientation of the reachability map, the best base positioning be determined. This method was specifically developed for accurately positioning a mobile service robot.

In contrast to the hybrid method of Mitsi et al. [8], Nektarios & Aspragathos [2] use a purely genetic algorithm for position determination. Their goal is to determine the optimal relative position and orientation between the robot base and the desired path. In doing so, the robot's end effector is intended to move along the path at the highest possible speed, thereby reducing cycle time. The optimization criterion is the Approximation of the Minimal Manipulator Velocity Ratio (AMMVR), which is employed to define a fitness function within the genetic algorithm. The proposed algorithm can be deployed on any given robot path that can feature any shape and is represented by an arbitrary curve. However, due to the nature of the interpolation, the algorithm can only handle convex or concave curves. For paths that contain both convex and concave segments, the interpolation algorithm must be applied separately to each convex and concave segment. The method can be applied to any non-redundant manipulator and has been verified for the PUMA 560 industrial robot with 6 degrees of freedom. Although, Nektarios & Aspragathos [2] do not address the issue of the limited scalability of genetic algorithms mentioned by Mitsi et al. [8]. As their studies focus on maximizing velocity values, they do not discuss the computational duration of their genetic algorithm.

Son & Kwon describe the robot base placement as a convex optimization problem [3]. The advantage of this approach lies in its ability to perform reachability checks without solving the inverse kinematics, which allows for significant time savings. This approach takes into account the conditions that all POIs are reachable, obstacles are considered, and singular configurations are avoided. Convex programming is a subfield of mathematical optimization that deals with the minimization or maximization of a convex objective function over a convex set. The key property of convex optimization problems is that every local minimum is also a global minimum. This simplifies the solution of such problems compared to non-convex optimization problems. Son & Kwon tested their method using four different motion sequences. The algorithm was

implemented in MATLAB using the CVX package. To verify the results, they utilized the KUKA KR6 R900-2 robot in the KUKA.Sim Pro simulation environment. The determined base positions include the coordinates, but not the orientation.

Vahrenkamp et al. describe another approach to determining base positions using reachability maps [10]. In their method, the perspective is inverted compared to Son & Kwon [9]: instead of analyzing which positions the TCP can reach from a given base position, they investigate where the robot base can be located for a given pose to reach. To achieve this, the Inverse Reachability Distribution (IRD) is first calculated in an offline step by iterating through all voxels of the reachability map and determining the base-to-TCP transformation and the voxel entry. By inverting the transformation and storing the result in the corresponding IRD voxel, the IRD is created. When online robot positioning is required, an Oriented Reachability Map (ORM) is calculated from the given POI and the IRD. By creating ORMs for each POI and overlaying them, a map is generated, from which all possible robot positions can be derived and qualitatively evaluated. From this map, the position with the best reachability can be selected. Subsequently, the chosen base position is checked for collisions. The approach allows the positioning in the X and Y coordinates only.

The review indicates that methodologies for automated robot base positioning already exist in the field. However, each methodology has its specific limitations and assumptions and will be evaluated in Section 3.2. To assess the applicability of these methodologies for use in forging cells, assumptions need to be defined, which will be addressed in the next chapter.

3. Automated Robot Base Positioning Approach

This chapter outlines the assumptions made for developing a novel method for automated robot base positioning (ARBP). Furthermore, the described approaches will be evaluated to choose the best one, which will then be adjusted to fit our use case.

3.1. Assumptions

The positioning of a 6-DoF robot within the cell design process diverges from the examined use case. During the layout design phase, starting from the initial sketch, each component of the cell can be placed as desired. Conversely, the automation of existing cells is subject to more stringent constraints regarding the layout. These constraints include:

- The cell is fully modeled in the simulation environment, except for the robot. As a result, all obstacles and environmental constraints are known.
- The robotics application is defined by POIs that specify both the pose and the configuration of the robot.
- The specific robot to be positioned has been determined.

Further, the ARBP method must be capable of automatically determining base positions in the three dimensions X, Y, and Z based on predefined POIs. This search should be feasible for arbitrary orientations of the robot base to ensure that the user is not restricted in the arrangement of other components in the cell. This means that the user has the option of determining the mounting location of the robot, e.g., for mounting to the ceiling. Existing solutions for base positioning typically do not limit themselves to a single position. Therefore, the system must be able to intelligently select an optimal position when multiple locations are possible. To ensure the feasibility and safety of the robotic application, the methodology must be capable of performing collision checks. Equally important is the ability to check for singularities and ensure compliance with the robot's joint limits. Process-related factors must also be taken into account when designing the cell. It may be necessary for the robot to be positioned in specific areas. Therefore, the implementation of the methodology should provide the capability to adjust the search area for base positions.

3.2. Evaluation Literature Survey

The most appropriate methodologies to the mentioned requirements are convex programming, inverse reachability, and cross-correlation. A significant advantage of convex programming is the detailed description for avoiding singular positions. Since the approach does not require solving inverse kinematics, Son & Kwon argue that the method can be faster than methods with inverse reachability [3]. A disadvantage of convex programming is that the orientation of the base is not considered at all. Additionally, while the method describes how to manage collisions, the processing of complex geometries as convex objects is computationally intensive.

The methods of cross-correlation [9] and inverse reachability [10] rely on the use of reachability maps. Cross-correlation is a well-established method for identifying patterns and relationships within large data sets. Nonetheless, the inverse reachability method performs better than cross-correlation. It exhibits lower computation times and does not require a predefined trajectory, leading to greater efficiency.

The main advantage of inverse reachability over convex programming is that it is well-generalized and thus applicable to other types of robots. Additionally, it takes base orientations into account. The potential drawback of a slightly longer computation time compared to convex programming can be offset by the simpler consideration of collisions.

Based on the evaluation of the three methods, the inverse reachability (IR) method is selected for further development and implementation in the ARBP, whereby ARBP becomes ARBP-IR. The IR method offers a balanced combination of robustness, flexibility, and comprehensive criteria fulfillment, making it appear to be the most suitable for meeting the requirements.

3.3. Adapted Approach

The method of inverse reachability must be modified to suit the use case. In the first step, the reachability map is calculated and then transformed into the inverse reachability distribution. This step is specific for each robot, but independent of any specific use case. For positioning the robot base, the Oriented Reachability Map (ORM) is required. The ORM takes the desired orientation of the robot base, the surrounding

environment, and the points of interest (POIs) that the robot must reach into account. The ORM identifies potential base positions from which the robot can reach all POIs. Subsequently, for each potential base position, it is checked whether the robot collides, violates joint limits, or reaches a singular configuration. If none of these criteria are met, the base position is confirmed; otherwise, the next potential base position is evaluated. A flowchart of the described adapted approach is given in Fig. 1.

Compared to Vahrenkamp et al., the ARBP-IR method allows for base positioning in the X, Y, and Z coordinates, making it more flexible in finding an ideal base position [10]. Additionally, the implementation in a software environment enables a more detailed and precise consideration of the cell environment, improving collision detection. It also checks for singularities, which was not possible before. However, this approach focuses solely on base positioning and does not determine the robot's path, therefore there is no optimization related to cycle times.

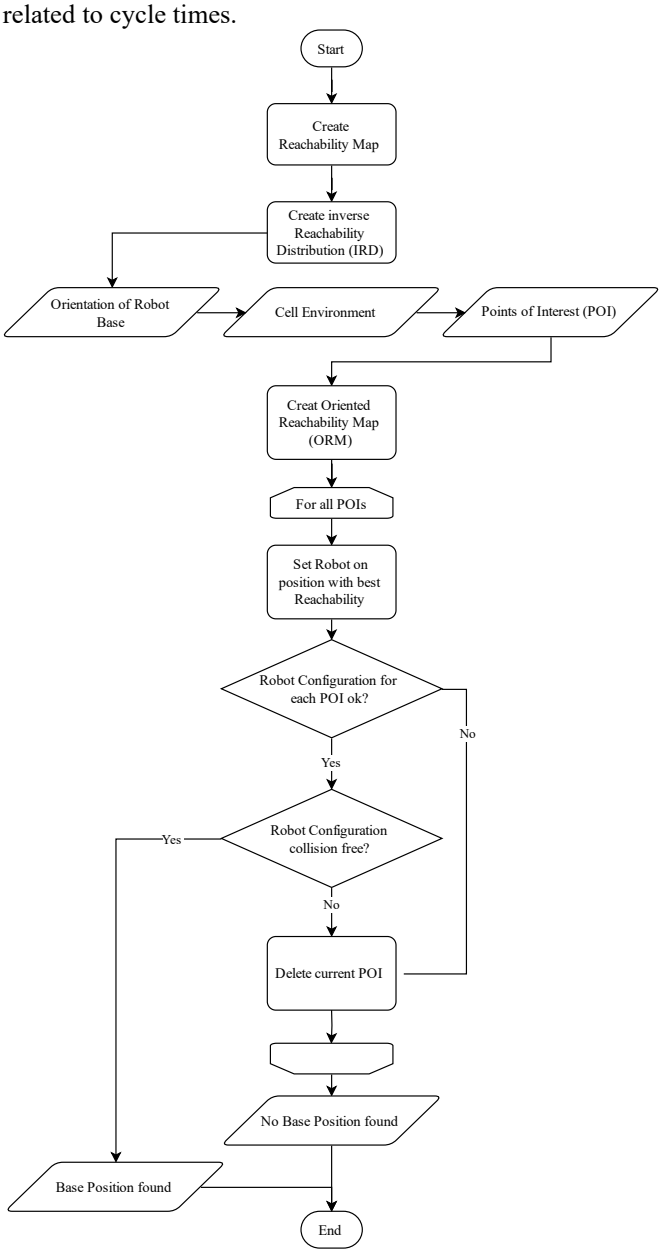

Fig. 1: Flowchart of the methodology for automated robot base positioning (ARBP-IR).

3.4. Implementation

Implementing the methodology in a software environment that includes a graphical user interface (GUI) will enhance its usability. The GUI eliminates the need for programming experience or skills, thereby significantly lowering the barrier to utilizing the methodology [11]. Visual Components (VC) is primarily designed for conducting manufacturing simulations. While calculations can also be done in VC using Python, the software is not specialized for extensive mathematical computations. Even tough to realize the robot base positioning in VC, MATLAB is utilized. MATLAB is an established and well-documented program with powerful functions and toolboxes for numerical calculations and data analysis. The combination of both programs allows for easier execution of computations and integration of advanced analyses directly into VC. MATLAB's extensive libraries provide the flexibility and capability to extend algorithms designed to work with the basic search algorithm. Therefore, the MATLAB Engine API was installed in Python within Visual Components to enhance the capabilities of VC and realize the required calculations.

To create the inverse reachability distribution, it is necessary to construct the reachability map of the robot. Zacharias et al. describe how this process works, and the implementation is carried out based on their approach [9]. It is applicable to all types of robots that are capable of positioning and orienting in three axes. First, the workspace of the robot is discretized through voxelization. This is done in the shape of a cube, with the robot arm located in the center of the cube at the origin of the X, Y and Z coordinates. The length of the edges of a single voxel is equal to the resolution of the map, and this resolution can be freely selected. For unambiguous identification, each voxel is described by an i, j, and k index. This type of indexing corresponds to a three-dimensional MATLAB storage array, which is also referenced in this way. For each voxel, the reachability value is calculated by virtually moving the robot arm to multiple poses within the volume element. The poses are distributed equidistantly on a spherical surface and are oriented normally to the spherical surface, with the resolution serving as the diameter and the voxel center as the center of the sphere. If all poses on the spherical surface N are reached with any configuration, the reachability value R for the voxel is 1. However, if the robot arm cannot validly reach any of the test poses, the number of reachable poses P is 0, and the voxel receives a reachability value of 0. Thus, the reachability is determined by (2)

$$R = \frac{P}{N} \tag{2}$$

The value obtained by sampling the voxel represents the probability that a point within the voxel can be reached with a randomly chosen orientation. The calculated reachability map for a robot is shown in Fig. 2.

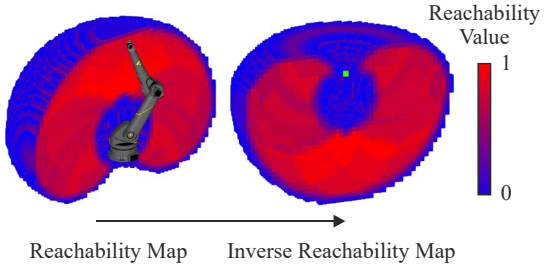

Fig. 2: Reachability Map of an exemplary industrial robot in Visual Component that is transformed into an Inverse Reachability Map. The map has an resolution of 10 mm. The coloration shows the Reachability Value of each point.

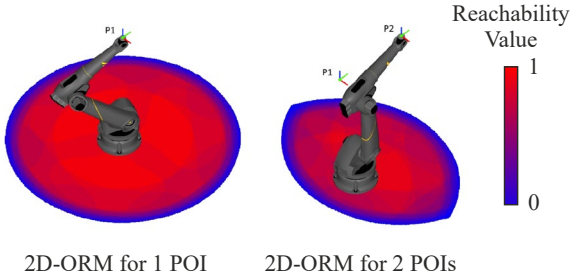

Fig. 3: Two dimensional ORM for one POI on the left side and the resulting overlapping ORM for two POIs.

The determination of the base position is achieved through the generation of the oriented reachability distribution. This is a point cloud consisting of potentially possible base positions for reaching the POIs and defines the search space for confirmed valid base positions. The reachability value R serves as a measure of the quality of each point. To find a valid base position as a solution to the method, the robot is virtually positioned on the point of the ORM (Oriented Reachability Map) with the highest reachability value. It may occur that multiple points have the same reachability value. In this case, the search grid point that appears first in the Python list representing the ORM is selected. At this grid point, the robot configurations and collision freedom are evaluated. If both are confirmed, a valid base position is identified. Otherwise, the next grid point is examined. If all grid points are invalid, it can be inferred that either no valid base position exists or that the set of potential robot base positions was not fully sampled. The ORM cannot be calculated in advance. It is based on the previously created Inverse Reachability Map, therefore this step has to be part of each evaluation on a single point. Additionally, the algorithm's inputs include the defined POIs, the user-specified orientation of the robot base – which distinguishes this approach from Vahrenkamp et al. [10] – and the search grid resolution.

Usually, there are multiple POIs. Therefore, the ORMs of each individual POI are overlapped. For this, the previously stored grid points are revisited and checked in the same way to determine if they fall within the reach of the other POIs. Consequently, each grid point provides a reachability value between 0 and 1, which are multiplied to yield the final reachability value. In contrast, Vahrenkamp et al. select the lowest value per grid position [10]. The chosen method prioritizes the search for base positions that have high reachability values for all POIs. Fig. 3 (left) shows the ORM for a single POI P1 for base positions on the ground and an additional POI P2 (right), which is located +1000 mm in the y-direction. As a result, the identical ORMs overlap to form an elliptical distribution. Due to the high resolution of 10 mm in the ORM from Fig. 3, the map appears as a continuous distribution. However, space of potentially possible base positions is presented as a discrete grid.

4. Validation

The objective of this research was to develop an automated methodology for positioning robot bases to make the cell design process more efficient. To assess whether the goal of this method development was accomplished, a robot placement was conducted. The environment used for the validation test is the virtual representation of an existing cell and therefore allows a comparison with the real-world setup. The automated base positioning identifies a similar position in the X and Y directions as chosen in the real-world application. A significant discrepancy occurs in the Z-direction, where the algorithm recommends raising the robot base by 200 mm from the ground. The calculation process was performed three times with different initial positions of the robot. The algorithm completed the task in 3.63 ± 0.01 s, consistently selecting the same position.

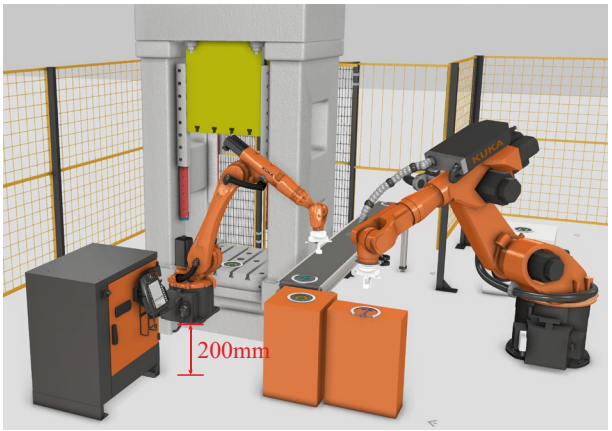

Fig. 4: Optimal positioning of the smaller robot on the left side of the forging cell. The algorithm recommends raising the robot by 200 mm compared to its manual positioning to enhance the reachability value.

Further, the efficiency of the ARBP-IR is tested. According to Makhal and Goins, there is no standardized method for efficiency testing of creating reachability maps [12]. For efficiency testing, they utilize the required time for testing the points of the reachability maps. Makhal and Goins' method tries to optimize the method of Vahrenkamp et al. and will be compared to the ARBP-IR. Therefore, the setting from Makhal and Goins was rebuilt in VC, and the same resolution for the

reachability map was set. Depending on the methods, a varying number of points were created. For achieving better comparability, the test rate is used. It indicates how many robot poses are checked on average per second by the respective method. The results are shown in Table 1. The ARBP-IR is able to process 1364 poses per second, whereas Makhal and Goins' method achieves 89 poses per second.

Table 1: Test rate for reachability map creation

	Number of processed poses	Time (s)	Test rate (poses/s)
ARBP-IR	352,000	258	1364
Makhal and Goins	763,600	8596.2	89

In conclusion, the ARBP-IR method offers considerable time savings in planning by reducing the effort required for base placement, which is particularly beneficial when designing complex robotic cells. This automation leads to a substantial increase in efficiency compared to manual methods. Additionally, the approach allows for the substitution of user expertise, minimizing the need for in-depth specialist knowledge to effectively plan and execute base placements. Furthermore, by integrating the add-on into the simulation software Visual Components, practical access to the method is facilitated, enhancing its usability. However, future work should expand the evaluation to include more use cases to increase comparability.

5. Summary and Outlook

In this article, a new method for automatic robot base positioning based on inverse reachability (ARBP-IR) was developed and examined. The aim was to reduce the time required for planning robot cells and to find an automated, reproducible solution that is independent of expert knowledge. The methodology was integrated as an add-on into the simulation program Visual Components to evaluate its practical impact on the cell design process. As part of the research, various methods for algorithm-supported base placement were presented.

The inverse reachability method best met the requirements, although it does not offer the option to freely choose the base orientation and is limited to positions in a two-dimensional space. This method served as the foundation for the new approach. To perform the calculation in a three-dimensional space, the workspaces were discretized using voxels, and subsequently, the reachability of each individual voxel was determined.

Subsequently, the ARBP-IR method was assessed in a realistic scenario based on the cell design of the CRC 1153 research facility. For this purpose, a robot was virtually positioned with predefined target points. This test demonstrated that the new method enables an easy base placement, in that way achieving the research goal.

Next, the ARBP-IR needs to be evaluated against existing methods to evaluate its performance and benefits, if necessary, make optimizations to the newly developed algorithm. For this purpose, the methods from the literature survey must first be implemented in Visual Components (VC) to generate comparable results. Here, the required computational effort and quality of results will be compared.

Acknowledgement

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – CRC 1153 – subproject C07 – 252662854.

References

[1] N. C. N. Doan und W. Lin, „Optimal robot placement with consideration of redundancy problem for wrist-partitioned 6R articulated robots,“ Robotics and Computer-Integrated Manufacturing, Volume 48, p. 233–242, December 2017.

[2] A. Nektarios und N. A. Aspragathos, „Optimal location of a general position and orientation end-effector’s path relative to manipulator’s base, considering velocity performance,“ Robotics and Computer-Integrated Manufacturing, Volume 26, p. 162–173, April 2010.

[3] S.-W. Son und D.-S. Kwon, „A convex programming approach to the base placement of a 6-DOF articulated robot with a spherical wrist,“ The International Journal of Advanced Manufacturing Technology, Volume 102, p. 3135–3150, February 2019.

[4] B.-A. Behrens und J. Uhe, „Introduction to tailored forming,“ Production Engineering, Volume 15, p. 133–136, January 2021.

[5] Y. K. Hwang und P. A. Watterberg, „Optimizing robot placement for visit-point tasks,“ Artificial intelligence and research planning , p. 81-87, 1996.

[6] D. Barral, J.-P. Perrin, E. Dombre und A. Liegeois, „Development of Optimisation Tools in the Context of an Industrial Robotic CAD Software Product,“ The International Journal of Advanced Manufacturing Technology , p. 822-831, 1999.

[7] T. Asokan, G. Seet, M. Lau und E. Low, „Optimum positioning of an underwater intervention robot to maximise workspace manipulability,“ Mechatronics, Volume 15, p. 747–766, July 2005.

[8] S. Mitsi, K.-D. Bouzakis, D. Sagris und G. Mansour, „Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm,“ Robotics and Computer-Integrated Manufacturing, Volume 24, p. 50–59, February 2008.

[9] F. Zacharias, W. Sepp, C. Borst und G. Hirzinger, „Using a model of the reachable workspace to position mobile manipulators for 3-d trajectories,“ in 2009 9th IEEE-RAS International Conference on Humanoid Robots, 2009.

[10] N. Vahrenkamp, T. Asfour und R. Dillmann, „Robot placement based on reachability inversion,“ in 2013 IEEE International Conference on Robotics and Automation, 2013.

[11] W. L. Martinez, „Graphical user interfaces,“ WIREs Computational Statistics, Volume 3, p. 119–133, February 2011.

[12] A. Makhal und A. K. Goins, „Reuleaux: Robot Base Placement by Reachability Analysis,“ in 2018 Second IEEE International Conference on Robotic Computing (IRC), 2018.