Efficient Industrial Solution for Robotic Task Sequencing Problem with Mutual Collision Avoidance & Cycle Time Optimization

Hicham Touzani[1,2], Nicolas Séguy[1], Hicham Hadj-Abdelkader[1], Raúl Suárez[3], Jan Rosell[3], Leopold Palomo-Avellaneda[3] and Samia Bouchafa[1]

***Abstract*—In the automotive industry, several robots are required to simultaneously carry out welding sequences on the same vehicle. Coordinating and assigning welding points between robots is a manual and difficult phase that needs to be optimized using automatic tools. The cycle time of the cell strongly depends on different robotic factors such as the task allocation among the robots, the configuration solutions and obstacle avoidance. Moreover, a key aspect, often neglected in the state of the art, is to define a strategy to solve the robotic task sequencing with an effective robot-robot collision avoidance integration. In this paper, we present an efficient iterative algorithm that generates a high-quality solution for Multi-Robotic Task Sequencing Problem. This latter manages not only the mentioned robotic factors but also aspects related to accessibility constraints and mutual collision avoidance. In addition, a home-developed planner (*RoboTSPlanner*) handling 6 axis has been validated in a real case scenario. In order to ensure the completeness of the proposed methodology, we perform an optimization in the task, configuration and coordination space in a *synergistic* way. Comparing to the existing approaches, both simulation and real experiments reveal positive results in terms of cycle time and show the ability of this method to be interfaced with both industrial simulation software and ROS-I tools.**

***Index Terms*—Robotic Task Sequencing, Coordination, Optimization, Motion Planning, Intelligent Manufacturing**

I. INTRODUCTION

In the last decades, the automotive industry has experienced an extensive use of industrial robots. Currently, they cooperate to produce a large batch of products in geometrically constrained environments (Fig. 1). Speed and quality of production are crucial factors for any industrial manufacturer, then, the use of reliable automatic tools are necessary.

For instance, when designing the line, distributing the welding points between the different robots is a complex task that cost time and personnel resources. In addition, robots often operate in shared workspaces. Therefore, the risk of collision must be considered in the optimization since it directly influences the cycle time.

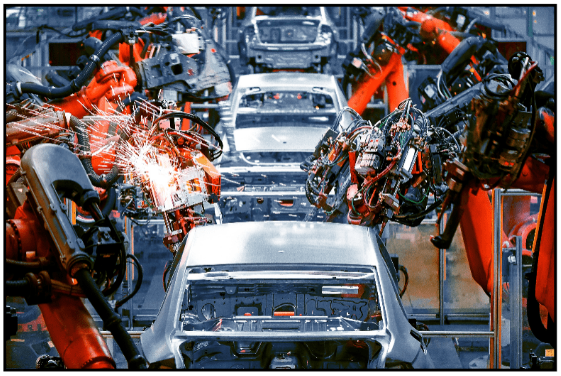

Fig. 1. Automotive BiW with several robots sharing the same workspace.

The objective of this work is to automate and optimize the robot programming process focusing on robotic task sequencing problem 'RTSP'. The RTSP consists in determining the trajectories for each robot in order to carry out welding points while minimizing the cycle time. This problem incorporates several considerations related to combinatorial optimization, path planning and robot coordination and leads to other synergistic sub-problems. The complexity of this problem has been discussed extensively [1]. There are studies that solve similar problems with exact methods such as [2] [3] [4]. However, the exact methods are only effective for small instances without considering different robotic factors (inverse kinematics 'IK' solutions , number of robots, accessibility constraints, obstacle avoidance, robot-robot collision, among others). Accessibility constraints refer to constraints of different types that make a task not accessible by a given robot, such as the robot kinematic limitations or the presence of obstacles.

The proposed approach works under the following assumptions. The robot IK are found beforehand by a solver assuming that there is only one predefined welding-gripper configuration for each welding point. The robot is assumed stationary during the task operation. Moreover, constraints due to predefined partial sub-sequences are not taken into account in the considered welding applications. Also, we assume that the TSP-heuristic based method provides a (near) optimal solution when the cost function evaluating the solutions no longer evolves over a fixed number of iterations. This last

Manuscript received: September 8, 2021; Revised: November 1, 2021; Accepted: December 29, 2021.

This paper was recommended for publication by Editor Youngjin Choi upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Segula Technologies & Association Nationale Recherche Technologie ANRT. Corresponding author: hicham259@gmail.com

[1] Authors are with Université Paris-Saclay, Univ Evry, IBISC, 91020, Evry-Courcouronnes, France. firstname.lastname@univ-evry.fr

[2] is with Segula Matra Automotive, Trappes, France. hicham.touzani@segula.fr

[3] Authors with the Institute of Industrial and Control Engineering (IOC) at Universitat Politècnica de Catalunya (UPC), Barcelona, Spain raul.suarez@upc.edu

Digital Object Identifier (DOI): see top of this page.

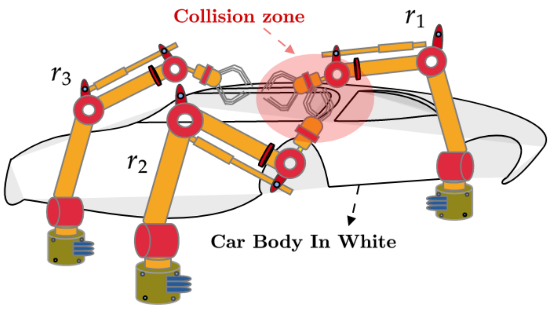

Fig. 2. Robots presenting a collision while performing their tasks.

assumption allows to present the optimality gap in complex problems when the optimal solution is unknown (hard to prove) even for straightforward comparisons. The term "high-quality" will be used for this purpose.

This paper is structured as follows. Related works and our contributions are presented in Section II. The specific addressed problem is formulated in Section III, and the proposed solution discussed in Section IV. Section V presents experimental results that show the efficiency of the implemented approach. Finally, Section VI gives conclusions and directions for future studies.

II. Related Work

Most recent industrial tools focus on optimizing the scheduling of robotic cells. However, aspects as motion planning and collision avoidance are ignored. In this paper, we focus mainly on how to generate optimal movements between points for a multi-robot system. The problem of distributing the welding points among the robots presents the synergistic combination of managing the accessibility constraint, determining the best inverse kinematic solution at every point, avoiding obstacles and interferences between the robots. Various studies and industrial tools have been proposed to solve the RTSP. However, very few of these solutions are implementable in industry since the robotic factors are often included partially and treated independently.

A. Robotic Task Sequencing Problem

The problem of determining the optimal sequence for a robot has often been associated with tour-searching combinatorial problems such as Traveling Salesman Problem 'TSP' like models [5]. Their complexity classes have been formalized for decision problems whose version belongs to the class NP-hard [6]. A generalized approach based on Genetic Algorithm 'GA' for one single robot has been presented in [7]. Alatertsev et al. [8] have developed an effective heuristic to determine the optimal sequencing for an industrial robot. In [9], an approach to solve the the Airbus challenge sequencing for a single robot with several tasks in a 2D plane have been proposed. The main limitations of these approaches are that they address only the resolution of the sequencing in an abstract framework and some of them overlook the kinematics of the robot. In addition, these approaches focus mainly on applications with a single robot. Therefore, no consideration is given to multi-robot systems where robot-robot collision avoidance (see Fig. 2) is required and must be integrated in the overall solution.

B. Multi-Robotic Task Sequencing Problem with Conflicts

In offline programming, the question of coordinating trajectories in an industrial robotic cell with several robots sharing the same workspace is addressed frequently. The solutions currently proposed are highly dependent on the engineer's expertise. Several studies have been proposed in order to solve the problem of avoiding mutual collisions of a multi-robot systems. Some of them may seem similar to our main issue such as task allocation and motion scheduling problem in pick and place applications [10][11][12]. The main difference is that these studies solve the problem partially in a high-level, by focusing on motion scheduling to avoid robot-robot collisions. Moreover, they consider a logical and spatiotemporal constraints between the tasks and are limited to unobstructed environments. Our approach focuses on solving the task sequencing and the motion planning in an obstructed industrial environment. The majority of current solutions solve only partially this problem with insufficient performance for being integrated into the global solution. Also, current simulation software, such as RobCad, Delmia, or Roboguide, does not integrate an effective solution for the task sequencing problem, even less with the automatic robot-robot collision management. On the other hand, IPS software cited in [13] works on solving the task sequencing with conflicts.

There are two main approaches addressing the coordination of multi-robot systems: centralized and decoupled [14].

The centralized approach is characterized by coordinating robots in a combined configuration space C_{space}. Namely, the configuration spaces of all robots are merged into a single composite configuration space by performing a Cartesian product. The size of this space grows according to the degree of freedom 'DOF' of all robots. This approach is excessively expensive in term of computation since it requires the determination of the composite C_{space}. Evidently, this approach directly integrates robot-robot collision avoidance in the optimization process, but it is still inappropriate in applications where the cycle time must be minimized.

In contrast, a decoupled approach breaks down the problem into computationally tractable sub-problems. This approach consists of pre-planning the trajectories individually by considering only the static obstacles in the environment and neglecting the collisions between robots. Following this, a coordination strategy is established to avoid mutual collisions. Introducing an interlock system, path modification or velocity tuning are considered to be the current methods for avoiding robot-robot collisions especially for robotic arms [15][16]. It is worth noting that the completeness is lightly sacrificed for the purpose of tractability. Our method circumvents this problem since it integrates the impact of the coordination on the overall solution. Also, it should be noted that the decoupled approach is more effective to solve a problem including different NP-hard sub-problems [17]. In [18], a workspace partition strategy is carried out by ensuring that each robot works in its own area without having to manage the robot-robot collision. Spensiri et al. [13] have proposed an iterative method that performs the task assignement and then calculates the trajectories to avoid collisions without reconsidering the first assignment. However, when introducing waiting times or performing a velocity tuning to avoid mutual collisions, task allocations as well as the planned trajectories initially proposed may be affected. Hence, motion planning and robot-robot collision avoidance presents a critical impact on the overall cycle time and must be combined with the other robotics factors. The approach proposed in this work is an improvement of [19] and integrates the accessibility constraint of each robot as well as the coordination into the overall process. In addition, an improved robust planner handling 6 DOF has been proposed and validated on a real industrial robots.

C. Research and Industrial Contributions of the proposal

The proposed approach simultaneously addresses the problems related to combinatorial optimization, motion planning and multi-robot coordination. The contribution of this research aims to highlight the advantages of integrating automatic solutions in an industrial context. Also, this paper demonstrates the ability of such tools to improve the feasibility and the cycle time of a real industrial cell.

The study contributes to both research and industry as follows: First, it formalizes the problem based on both current state of the art methods and industrial practices. Second, it provides an analysis regarding the influence of the sequencing, motion planning and robot-robot coordination on the overall cycle time of the cell. Third, it resolves motion planning, coordination and production scheduling problems in a synergistic way. Fourth, it develops an iterative solution for high dimensional complexity for muti-robot systems.

Also, this study improves mainly the industry practices as follows: First, it provides both automation and time/resource saving to install a production line. Second, it creates repeatable solutions with a high performance in terms of cycle time. Third, it provides a solution that is independent of the engineer expertise.

III. Problem Definition

The RTSP can be formulated in the form of min (sum-max) Multi Generalized Traveling Salesman Problem 'MGTSP' to simultaneously optimize both the total and the maximum movement durations of the robot trajectories. This formulation has been adopted for the purpose of optimizing the cycle time while balancing the workload between the robots.

Consider the following general nomenclature (see Fig. 3):

- $R = \{r_1, r_2, .., r_k, .., r_N\}$ set of N available robots in the manufacturing cell.
- $P = \{p_1, p_2, .., p_i, .., p_n\}$: set of n welding points p_i on a workpiece where a welding action must be done by a robot; each welding point p_i has an associate welding position and orientation that fix the configuration of the robot end-effector to perform the welding action.
- p_i^k: welding point at p_i assigned to robot r_k; for simplicity and coherence, it will also be used to represent the location of the end-effector of r_k when performing a welding action on p_i.

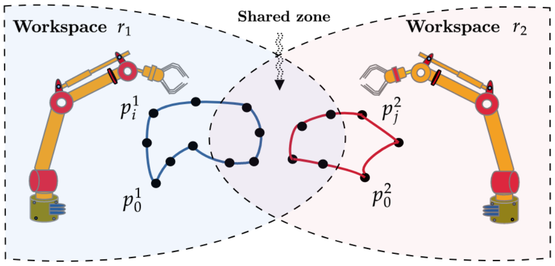

Fig. 3. Two robots with their accessibility constraints and the corresponding task assignment solution

- Q^k: a generic configuration of robot r_k; Q_0^k is used to represent a predefined starting and ending configuration of r_k (which produces the end-effector location p_0^k), and $Q_{p_i}^k$ the robot configurations that produces p_i^k. Note that more than one configuration Q^k can generate the same p^k.
$P_{acc}^k \subset P$: set of welding points p_i accessible to r_k.

Now, the **goal of this work** is the determination of sequences $S_p^k = \{p_0^k, s_1^k ..., s_j^k, ..., s_{n_k}^k, p_0^k \mid s_j \in P_{acc}^k\}$, i.e. s_j^k represents a point p_i^k assigned to r_k in the solution sequence S_p^k. Equivalently, $S_Q^k = \{Q_0^k, J_1^k, ..., J_j^k, ..., J_{n_k}^k, Q_0^k \mid s_j \in P_{acc}^k\}$ with J_j^k being a configuration of r^k that correspond to s_j^k. This must satisfy $\bigcap_{\forall k} S_p^k = \emptyset$ and $\bigcup_{\forall k} S_p^k = P$ (i.e. $\sum_{k=1}^{N} n_k = n$) while optimizing the cell cycle time and respecting the robotic constraints (such as kinodynamics and accessibility constraints).

Considering current industry practices, the proposed methodology has the following input and output data:

Input:

- The set of n welding points $P = \{p_1, p_2, .., p_i, .., p_n\}$.
- A set of robot starting configurations $R_0 = \{Q_0^1, ..., Q_0^k, ..., Q_0^N\}$, and the set of corresponding configurations of the end-effectors $EE_0 = \{p_0^1, p_0^2, ..., p_0^N\}$.
- A set of m IK solutions for each robot at point p_i, $Q_{p_i}^k = \{Q_1^k, Q_2^k, ..., Q_j^k, ..., Q_m^k\}$, $k = 1, ..., N$ (note that Q_j^k may not exist and that $Q_{p_i}^k$ could be empty when p_i is not reachable by r_k.
- The set P_{acc}^k, $k = 1, ..., N$ of accessible welding points p_i for each robot r_k:
$P_{acc}^k = \{p_i \mid p_i \in P \text{ and } \exists Q_j^k\}$.

Output:

- The optimal sequence S_p^k for each robot and the corresponding movements to execute it, such that: 1) The robot kinodynamics and the accessibility constraints are satisfied. 2) Each welding point p_i must be reached only once by only one robot. 3) Each robot must leave from and return to its initial configuration Q_0^k. 4) No collision occurs with the predefined trajectories in the execution of the sequences. 5) The cycle time and the overall movement duration are minimized.

The problem can be formulated in a formal way as follows:

let $G = (V, E)$ be the representation of the problem defined above as a complete and undirected graph, where V is the set of vertices representing points p_i and E is the set of edges between the vertices, each edge with a weight w indicating the travel time of the optimal collision-free trajectory of a robot going from one point p_i to another one.

The path of the robot r_k between two consecutive points s_j^k and s_{j+1}^k is given by a number K of *via configurations* Q' (or via points) resulting from the motion planner, i.e. $\text{Path}^k_{(J_j^k, J_{j+1}^k)} = \{J_j^k, Q'_1, Q'_2, ..., Q'_K, J_{j+1}^k\}$.

Then, taking into account the kinodynamic constraints of the robots (acceleration, velocity, and torque/force), the optimal traveling time of $\text{Path}^k_{(J_j, J_{j+1})}$ will be considered as the cost $c(s_j^k, s_{j+1}^k)$ of the movement from s_j^k to s_{j+1}^k. The path plus the time parametrization defines the robot trajectory, i.e. a time-dependent path with a defined speed and acceleration at each point.

The following optimization problem can be stated:

$$C^k = c(p_0^k, s_1) + \sum_{i=1}^{n_k} c(s_i, s_{i+1}) + c(s_{n_k}, p_0^k) \tag{1}$$

$$C_{minsum} = \min(\sum_{k=1}^{N} C^k) \tag{2}$$

$$C_{minmax} = \min(\max_{k \in \{1,...,N\}} (C^k)) \tag{3}$$

$$Cost = \alpha C_{minsum} + \beta C_{minmax} \tag{4}$$

$$c^k_{(J_j^k, J_{j+1}^k)} = \max_{d \in \{1,...,DOF^k\}} \left(\frac{|\theta_{j+1}^{k,d} - \theta_j^{k,d}|}{\dot{\theta}_{max}^{k,d}} \right) \tag{5}$$

where equation (1) indicates the cost C^k of each robot r_k; equations (2), (3) and (4) indicate the objective criterion $Cost$, being C_{minsum} and C_{minmax} the two criteria considered to evaluate a sequence with the coefficients α and β promoting one criterion over the other; and, finally, we use an approximate metric (5) given the complexity of determining the optimal time between two tasks, with $\theta_j^{k,d}$ the joint d of J_j^k where $d \in \{1, \ldots, DOF^k\}$ whose maximum joint velocity is $\dot{\theta}_{max}^{k,d}$. Robots have to avoid mutual conflicts, thus, any two arcs implying a robot-robot collision can not be simultaneously executed.

IV. Proposed Approach

Finding the optimal solution for the RTSP is a complex problem that requires a lot of resources, especially when the number of welding points increases in complex environments, as frequently are the industrial cells, and it becomes even more complex when it has to deal with a multi-robot system, as all the robots actions influence each other.

For instance, the choice of a particular task order depends on the location of the robot bases and the accessibility constraints, and the choice of the robot configuration depends on the task order. In addition, path planning avoiding robot-robot collisions can influence the choice of the order of tasks and vice-versa. A lot of computational effort is necessary to solve this problem because all these factors significantly increase the dimension of the search space to look for a global solution, and it quickly becomes computationally intractable. For these reasons, the proposed strategy is based on an iterative approach decoupling the problem into different stages. The approach directly takes into account the synergistic relationship between the robotic factors and considers their impact on the overall cycle time. The whole problem can be decoupled into four major stages as follows:

- *Stage 1:* Determine the robot sequences S_p^k considering the accessibility constraints.
- *Stage 2:* Find the proper robot configurations S_Q^k for each welding point p_i (see [19] for more details).
- *Stage 3:* Solve the motion planning problem with movement time optimization; this can be done using the home-developed *RobTSPlanner*, which uses a PRM and a via point generation method with storing graph capabilities (see [19] and Section IV-C for more details).
- *Stage 4:* Avoid the robot-robot interferences using the coordination optimization.

In this work, we present the implementation of new features, mainly related to the integration of accessibility constraints (Stage 1) in section IV-A and to robot-robot coordination (Stage 4) in section IV-B. Moreover, our major contribution focus on describing a new strategy integrating the four stages in a synergistic way (section IV-C).

A. Task Assignment with Accessibility Constraints

The algorithm proposed in Stage 1 is based on a non-deterministic genetic algorithm 'GA' for several reasons. First, our work aims to provide a flexible global solution for the min (sum-max) MGTSP for a robotic problem that can easily integrate several constraints like multiple number of robots, inverse kinematics and accessibility constraints. Thus, this study does not aim to do better than existing state-of-the-art approaches with a tailored heuristic for a specific TSP-based problem like GTSP in combinatorial optimization. Second, min (sum-max) MGTSP is NP-hard. So, the use of exact method is less effective when it comes to integrate all the robotic factors. Third, the goal of the first two stages is not necessarily to find the "exact" optimal sequence but high-quality solutions that facilitate the synergistic evaluation between the other stages. For instance, the optimal sequence from the Stages 1 and 2 is not necessarily the global solution after the integration of Stage 3 and 4.

The algorithm of Stage 1 has been modified to include accessibility constraints and, thus, ensure an adapted solution for an industrial real case scenario. Then, the scope of the proposed approach in Stage 1 and 2 is confined to determine a good starting estimation regarding the assignment of tasks to each robot r_k.

The proposed algorithm operates on a set of solutions defining the sequences to be performed by N robots. Each solution is defined in a single chromosome $Ch_l = \{S_p^1, ..., S_p^k, ..., S_p^N\}$ evaluated using the objective function of the Eq. 4. In fact, this function quantitatively measures the suitability of a candidate assignment. The initial guess is evaluated by assigning a score given by summing the duration of all the robot movements (Eq. 2) and the highest robot movement duration (Eq. 3) for Ch_l. This evaluation is required for the selection phase of the new population Pop. Each individual is generated by selection, mutation and crossing operators and then evaluated. Note that, in order to include the accessibility constraints, the following additional condition is imposed in the generation of the chromosomes: $\forall p_i \in S^k : p_i \in P^k_{acc}$. In addition, in order to avoid assigning a non accessible point p_i to a robot r_k during the global mutation operations, a specific algorithm has been developed to modify the sequences that lead to non valid solutions. The details of this procedure are shown in Algorithm 1 where the differences with respect to a standard GA can be appreciated.

Algorithm 1 Optimization with accessibility constraint

Input: a set of welding points $P = \{p_1, p_2, .., p_i, .., p_n\}$, a set of accessible points for each robot $P^k_{acc} = \{p_i \mid p_i \in P \text{ and } \exists Q^k_j\}$, a set of starting end effector position for each robot $EE_0 = \{p^1_0, p^2_0, ..., p^N_0\}$

Output: A minimal cost cyclic S^k_p for each robot with a dual objective value satisfying the robotic factors.

```
$t \leftarrow 0$
Initialize the promoting factors $\alpha$, $\beta$
Initialize parameters of GA
Generate a random $Ch_l$ satisfying the accessibility constraints ((for every $S^k_p$ in $Ch_l$) $\in P^k_{acc}$)
Generate an initial population $Pop_t$ of chromosomes $Ch$
Evaluate fitness (cost function Eq. (5))
while termination condition not met do
    Select individuals from $Pop_t$
    Randomly select the indexes to be mutated
    Locally Mutate the solution of a single robot
    Generate the mutant candidate satisfying the accessibility constraint
    Mutate robot solutions globally
    Cross robot solutions
    Re-evaluate fitness (cost function equation (5))
    $Pop_{t+1} \leftarrow$ new individuals
    $t \leftarrow t + 1$
end while
Return the best chromosome solution $Ch_l$ and its fitness value satisfying the accessibility constraint
```

B. Robot-Robot Coordination

The coordination in the Stage 4 is an essential part to ensure robot-robot collision avoidance. Indeed, in practical cases, simulation engineers need to determine manually the synchronization signals guaranteeing that two robots do not enter in the same area simultaneously. Also, another limitation is that these signals must be recalculated if the initial program is modified, as for instance, those related to *Carry-Over* operations (i.e. adding new functionalities). For these reasons, we propose a tool that automates the generation of essential signals for the robot-robot collision avoidance.

One of the main contribution, compared to the previous studies, is that we consider the influence of Stage 4 on the overall cycle time. Let's number the configurations of robot r_k along its trajectory by $b^k = \{0, 1, 2, ...b^k_i, ..., b^k_{max}\}$.

Let's define now the Coordination Space C_{crd} of the set of robots as the space determined by $C_{crd} = b^1 \times b^2 \times ... \times b^N$. Thus, a point in C_{crd} univocally refers to the configurations of all the robots in the workspace.

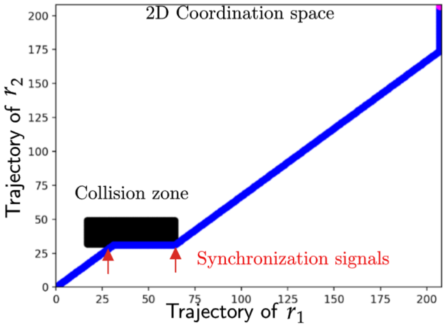

Fig. 4. Example of the coordination of two robots r_1 and r_2 in C_{crd} avoiding collisions. In this example $b^1_{max} = b^2_{max} = 200$.

In order to perform their trajectories, all the robots have to go from $b^k_i = 0$ to $b^k_i = b^k_{max}$, since the robot paths were planned separately, some points in C_{crd} may imply a robot-robot collision, which must be avoided by properly defining a path in C_{crd}. This formulation makes it possible to express the coordination problem as a path planning problem in C_{crd}. In addition, it raises another optimization problem: find the required synchronization signals that ensure the minimum robot waiting times. These are necessary to avoid robot-robot collisions. This requires the representation of all the "obstacles" in the coordination space. Graph-based methods [14] [15] are then used to find the path in C_{crd}, including the minimum waiting times to avoid robot-robot collisions. Based on the pre-calculated trajectories of the robots, the approach consists of adjusting their velocity profiles so that no robot-robot collision occurs. The collision zones act as "obstacles" in the C_{crd}, which leads to plan a path in this space where the axes correspond to the samples of each trajectory. Since no backward movement is allowed, the obstacle shapes are adjusted to their bounding boxes aligned with the axis to avoid deadlocks.Fig. 4 illustrates an example of Stage 4, the algorithm solves the robot-robot interference by adding synchronizations points automatically. However, this may have an impact on the cycle time. Our new strategy solves this problem as detailed below.

C. New Overall Strategy

The iterative approach adopted is a major improvement with respect to previously developed algorithms. To the best of our knowledge, our solution is the first to attempt to solve the problem in its entirety by analyzing the influence of each stage on the final cycle time. In addition, this approach has been directly implemented in the industrial software Fanuc ROBOGUIDE. Moreover, the algorithms have been also interfaced with ROS-I capabilities to ensure more generalized simulations. Fig. 5 illustrates the strategy on which our solution is based.

The complexity of RTSP is due to an increasing number of parameters introduced by the considered robotic factors, as has been previously discussed. For instance, it is complicated to achieve sufficient performance by calling the planner between each two consecutive robot actions to avoid collisions, especially when dealing with an NP-hard problem with, for

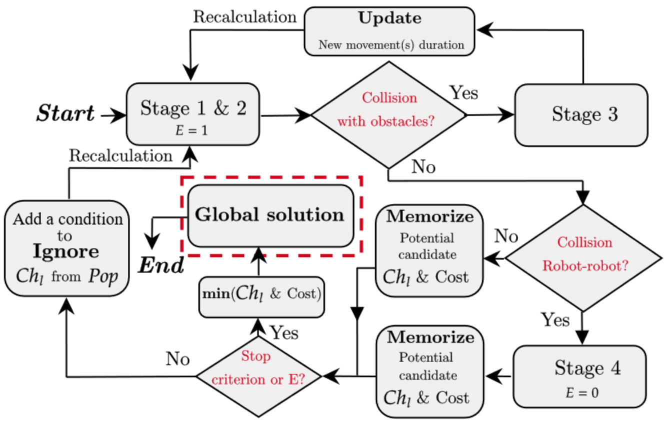

Fig. 5. Strategy of the proposed approach.

instance, up to 50 tasks including multiple IK solutions in multi-robot systems. To cover this limitation, the Stage 1 considers only the tasks allocation in a free-collision environment and uses an approximate function for the initial guess. In addition, a major advantage is that, in the best case scenario, if the determined sequences are directly collision-free, the approach converges to the overall solution without having to calculate all the trajectories in advance. Otherwise, the home-developed *RoboTSPlanner* of Stage 3 is called. Indeed, a roadmap is created, being its vertices collision-free configurations. Then, a path shortcutting is performed to keep the relevant via points and avoid the obstacles. It should be noted that the *RoboTSPlanner* of Stage 3 is only called for segments presenting a collision. However, a crucial side-effect to be mentioned is that the quality of the initial solution may be affected when the planner is used to find a new trajectory avoiding collisions. For this reason, the sequence is recalculated in Stage 1 when a collision occurs by updating the new time duration of the trajectory presenting a collision.

Some approaches, like [20], use coordination methods as a last step to avoid mutual collision but do not consider that conflicts can affect the overall solution, which complicates the problem. Our approach consider it. The mutual collision avoidance is iteratively integrated into the optimization process and calls into question the previous choices according to the evolution of the cycle time. In fact, after coordinating the trajectories in Stage 4, we record the solution and its cost in a list of "potential candidates", hoping to converge towards a new assignment/order integrating the coordination with a better cycle time. This is done by excluding the solution in the list in the next global iterations. For instance, if the solution exists in the list, it is ignored and the second ranked new solution is selected to be evaluated in the process, and so on. This process is repeated until the stop criterion is reached (no improvement of the quality value after a given number of iteration). Finally, the solution with the minimum cycle time from the list of "potential candidates" is selected as the global solution.

V. Experimental Results

The proposed approach has been tested for cases of different complexity in simulation as well as with real robots. A realistic industrial environment has been simulated considering that several tasks must be performed by multiple robots in a shared workspace. These experiments highlight the ability of the approach to adapt and solve a real industrial problem with the objective of minimizing the cycle time. The developed approach suggests a task assignment respecting the accessibility constraints and avoiding mutual collisions interferences. Moreover, it defines a sequence order as well as collision-free configuration solution for each robot. Fig. 6 illustrates an example of the proposed solution for a problem of 12 tasks to be solved by two robots. Fig. 6-a shows the high-quality solution when no accessibility constraints were set and Fig. 6-b shows the high-quality solution when the following constraint are considered: points '3', '5' and '8' are not reachable for robot r_1 and points '7', '9', '11' are not reachable for robot r_2. The solution in the constrained case demonstrates the capacity of our algorithm to adapt and find a new solution respecting the accessibility constraints.

As previously stated, the optimality gap is very complex to be determined for the studied problem. However, it can be determined for a special case of a single robot without considering obstacles where the problem can be reduced to GTSP. Fig. 7 shows the evolution over time of our cost function for 30 tasks solving GTSP. It can be appreciated that the cost converges to similar values as the exact solver Gurobi (gurobi.com) with a small distribution around it. Moreover, for this particular example, the computation time is of a similar order of magnitude to the one cited in [9], even for tasks with 3D coordinates. Please note that this comparison is a special case since it only refers to Stage 1 and 2 for one robot.

Multiple runs were performed for each problem in order to reduce random fluctuations (Fig. 7). Five relevant problems are discussed in this paper, named respectively Cases 1, 2, 3, 4 and 5, dealing with 8, 12 and 17 tasks for two robots and 30 and 50 tasks for three robots. These instances were constructed based on the available simulations and real environment, and they have been chosen by industrial experts in order to emulate a real robotic welding cell. Note that Cases 1 to 5 include a total of 84, 138, 164, 331 and 521 possible configurations, and 4, 5, 4, 10 and 14 shared feasible number of tasks, respectively.

Note that all the cases have been simulated and Case 3 has been validated on real robots for demonstrative purposes. For real validation, we have constructed a scenario with two six-axis Stäubli TX-90 robots, each one equipped with Schunk grippers, both in simulation and in a real robotic cell.

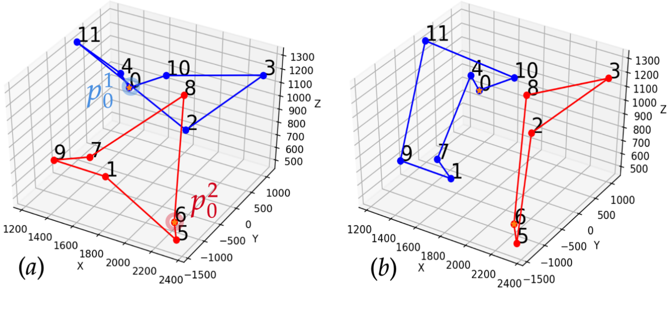

Fig. 6. Solution with and without constraint on accessibility (high quality sequence in blue for r_1 and in red for r_2): a) both robots can reach all the welding points; b) points 3, 5 and 8 are not reachable by robot r_1, and points 7, 9 and 11 are not reachable by robot r_2.

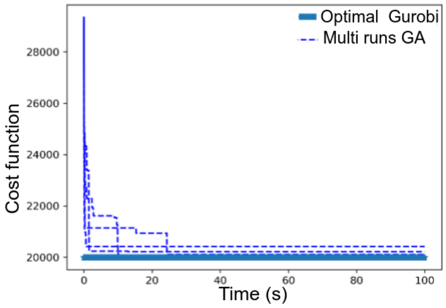

Fig. 7. Solutions of multiple runs for GTSP using our approach compared to an optimal solution for 30 tasks.

The implementation was based on Robot Operating System (ROS) (ros.org). By exploiting ROS-Industrial capabilities, the implementation has the main advantage of, not only encompassing all types of robots, but also making the solution easily embeddable into any industrial simulation software. To the best of our knowledge, our solution is the first to solve RTSP using completely open source tools. Also, as mentioned above, our approach integrates all the robotic factors in a synergistic way.

The evaluation was performed on a PC powered by i7-9750H at 2.6 GHz and 32 GB of RAM. Regarding ROS tools and packages, MoveIt! (moveit.ros.org) was used for environment creation and interaction, manipulation, collision checking, solving the IK and retrieving robot joint states via the *Move Group Interface*. The approach is based on pre-planning trajectories individually to optimize the cycle time.

Since our industrial applications imply multi-query problems, our home-developed *RoboTSPlanner* was directly interfaced with Moveit!.

The computational costs for the studied cases are plotted in Fig. 8-a. Note that even for 50 tasks, the computational time is much lower than the time spent by an offline programmer engineer (2-3 days). For instance, Fig. 8-b presents the evolution of the minimum cost of the “potential candidates” list for the overall approach for Case 4.

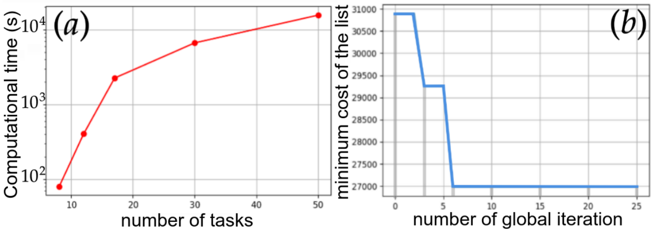

Fig. 8. a) Running times in the studied cases for the global approach. b) Evolution of the minimum cost in Case 4 for the global approach.

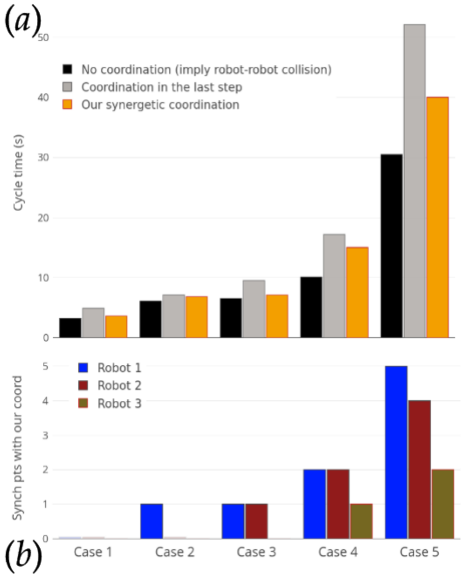

Fig. 9. a)- Comparison of our synergistic coordination with the coordination in the last step. b)- Number of synchronization points in our cases.

It is difficult to make a fair comparison between studies dealing with similar problems. The complexity of this comparison lies in the different ways in which the problem was approached and the robotic constraints considered, in the number and the type of the industrial robot used and the industrial environment considered. Moreover, no global reference exists for the multi-robotic task sequencing problem (see [1] for benchmarks and evaluation limitations). Despite these limitations, we compared our approach to [20] by reproducing its principle for the same problem (as they use the coordination without considering the previous choices). Fig. 9-a shows that our approach is able to obtain a better cycle time with coordination by calling into question the previously defined assignment in Stage 1. For example, in Case 1, it is obvious to note the advantage of our algorithm compared to a simple coordination not considering the synergistic influence of the other robotic factors. Thanks to its iterative form, the proposed approach is able to obtain a better cycle time with a decrease of 26.27% by generating a new solution with no synchronization points (Fig. 9-b). For reference, the black bars in Fig. 9-a corresponds to the cycle times without coordination, which implies a robot-robot collision. It is also relevant to point out that the cycle time has been increased only by 12.77% after solving the robot-robot interferences. Regarding

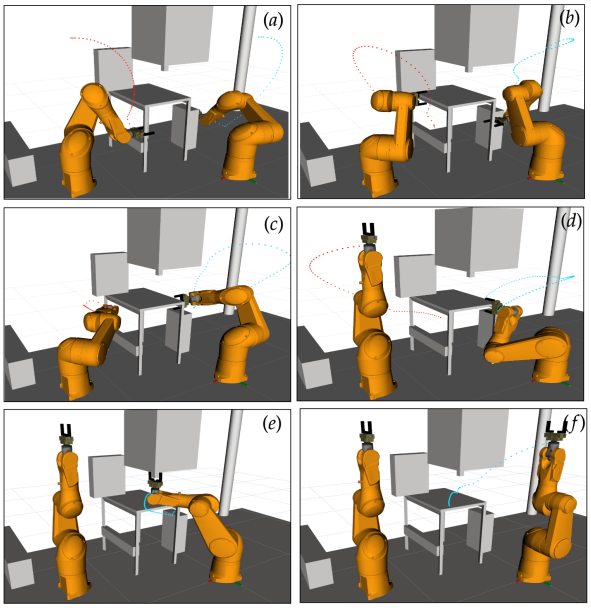

Fig. 10. Snapshots of the simulation for Case 3. The trajectories are plotted in blue and red. Four tasks are shared and accessible by the two robots.

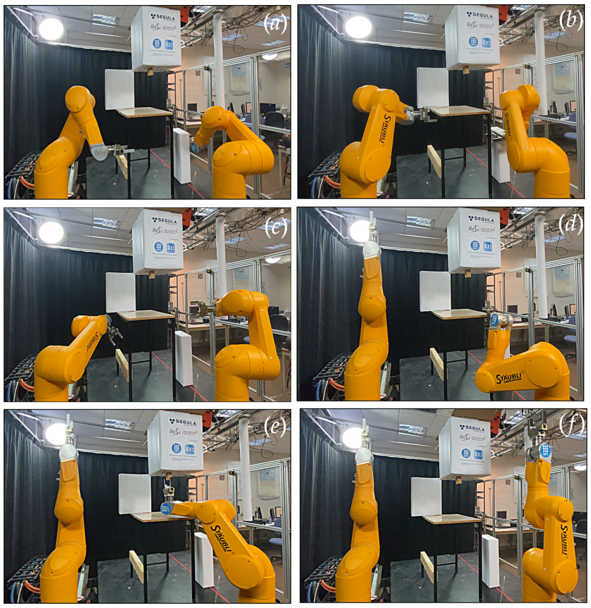

Fig. 11. Snapshots of the real execution for case 3. Four tasks are accessible by the two robots. This case was only selected for a presentative purposes.

Cases 2, 3, 4 and 5, the cycle time was improved by 18.34%, 12.78% and 6.69% 23.17%, respectively.

We also applied our approach on real robots for Case 3 including 17 tasks for the 2 Stäubli robots with several obstacles in the environment. Fig. 10 presents some snapshots of the simulation of the sequences of each robot.

The algorithm succeeded in assigning the 17 tasks to the robots while respecting their accessibility constraints, in choosing the most suitable configurations and in planning smooth trajectories while avoiding obstacles and robot-robot potential collisions. Finally, Fig. 11 presents snapshots of the real execution of the robot trajectories for the same case. A video of the experimental results showing the different robot movements is available in https://youtu.be/sFWYrUe9CZQ.

VI. CONCLUSIONS

This work was conducted with the aim of automating and optimizing the robotic task sequencing problem, particularly in the automotive industry. In this paper, we have presented an approach that integrates four main stages in a *synergistic* way. Given the high complexity of the problem, the proposal relies on a decoupled approach. Thanks to its iterative form, this method compensates the scarified completeness produced by its decoupled structure. The performance of the implemented method was demonstrated in simulations as well as in a real scenario emulating the behavior of an automotive industrial cell. Compared to current state-of-the-art methods, our approach presents better results in terms of cycle time. Moreover, the used objective function demonstrates its ability to minimize both the total robot movement duration and the cycle time, converging to high-quality solutions. As far as we can tell, our approach solves the RTSP in its entirety considering the impact of each robotic factor on the overall cycle time. In addition to the implementation on an industrial software, our method was also interfaced with ROS-I in order to converge towards a generalized industrial solution. A direction for future work is to apply this methodology in a real industrial scenario. Furthermore, we will evolve our solution with more robotic factors, such as different task duration and logical constraints.

ACKNOWLEDGMENT

This work has been supported by Segula & ANRT, and by the Spanish Government project PID2020-114819GB-I00. The authors thank Daniel Lautram and Mokrane Abdiche, from Segula, for their precious support.

REFERENCES

[1] Alatartsev, S., Stellmacher, S., & Ortmeier, F. (2015). Robotic task sequencing problem: A survey. Journal of intelligent & robotic systems, 80(2), 279-298.

[2] Hömberg, D., Landry, C., Skutella, M., & Welz, W. A. (2017). Automatic reconfiguration of robotic welding cells. In Math for the Digital Factory (pp. 183-203). Springer, Cham.

[3] Skutella, M., & Welz, W. (2011). Route planning for robot systems. In Operations research proceedings 2010 (pp. 307-312). Springer, Berlin.

[4] Rambau and C. Schwarz. Solving a vehicle routing problem with resource conflicts and makespan objective with an application in car body manufacturing. Optimization Methods and Software, 29:353–375, 2014.

[5] Applegate, D. L., Bixby, R. E., Chvatal, V., & Cook, W. J. (2006). The traveling salesman problem: a computational study. Princeton.

[6] Garey, M. R., & Johnson, D. S. (1979). Computers and intractability (Vol. 174). San Francisco: freeman.

[7] Zacharia, P. T., & Aspragathos, N. A. (2005). Optimal robot task scheduling based on genetic algorithms. Robotics and Computer-Integrated Manufacturing, 21(1), 67-79.

[8] Alatartsev, S., Mersheeva, V., Augustine, M., & Ortmeier, F. (2013). On optimizing a sequence of robotic tasks. IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 217-223).

[9] F. Suárez-Ruizz, T. S. Lembono, and Q.-C. Pham, ”Robotsp–a fast solution to the robotic task sequencing problem,” in Proc. IEEE ICRA, 2018, pp. 1611–1616

[10] K. Behrens, R. Lange, and M. Mansouri, ”A constraint programming approach to simultaneous task allocation and motion scheduling for industrial dual-arm manipulation tasks,” in 2019 International Conference on Robotics and Automation (ICRA), May 2019, pp. 8705–8711.

[11] Kimmel, A., & Bekris, K. E. (2016, June). Scheduling pick-and-place tasks for dual-arm manipulators using incremental search on coordination diagrams. In ICAPS workshop on planning and robotics.

[12] Choudhury, S., Gupta, J. K., Kochenderfer, M. J., Sadigh, D., & Bohg, J. (2021). Dynamic multi-robot task allocation under uncertainty and temporal constraints. Autonomous Robots, 1-17.

[13] D. Spensieri, J. S. Carlson, F. Ekstedt, and R. Bohlin, An iterative approach for collision free routing and scheduling in multirobot stations, IEEE Trans. Autom. Sci. Eng., 13, 2, pp. 950–962, 2016.

[14] L.E. Kavraki , S.M. LaValle , Motion planning, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, 2008, pp. 109–131.

[15] D. Spensieri, R. Bohlin, J. S. Carlson, Coordination of robot paths for cycle time minimization, IEEE International Conference on Automation Science and Engineering, Madison, USA, 2013

[16] Montano, A., & Suárez, R. (2016). Coordination of several robots based on temporal synchronization. Robotics and Computer-Integrated Manufacturing, 42, 73-85.

[17] Chiddarwar, S. S., & Babu, N. R. (2011). Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence. Robotics and Autonomous Systems, 59(7-8), 508-518.

[18] Åblad, E., Spensieri, D., Bohlin, R., & Carlson, J.S. 2017. Intersection-free geometrical partitioning of multirobot stations for cycle time optimization. IEEE Trans. on Aut. Sc. & Eng., 15(2), 842-851.

[19] Touzani, H., Hadj-Abdelkader, H., Séguy, N., & Bouchafa, S. (2021). Multi-Robot Task Sequencing & Automatic Path Planning for Cycle Time Optimization: Application for Car Production Line. IEEE Robotics and Automation Letters, 6(2), 1335-1342.

[20] Pellegrinelli S, Pedrocchi N, Tosatti LM, Fischer A, Tolio T (2017) Multi-robot spot-welding cells for car-body assembly: design and motion planning. Robot Comput Integr Manuf 44(C):97–116.