Multi-Robot Task Sequencing & Automatic Path Planning for Cycle Time Optimization: Application for Car Production Line

Hicham Touzani, Hicham Hadj-Abdelkader, Nicolas Seguy, Samia Bouchafa

Multi-Robot Task Sequencing & Automatic Path planning for Cycle Time Optimization: Application for Car Production Line

Hicham Touzani[1,2], Hicham Hadj-Abdelkader[1], Nicolas Séguy[1], and Samia Bouchafa[1]

***Abstract*—Industrial robots are programmed to repeat a sequence of well-defined tasks, e.g. deburring, cutting and welding. The production cycle time is directly influenced by the task order as well as the way trajectories linking this tasks are generated. In this paper, we present an optimization approach that minimizes the production time as well as the overall movements duration of the robots. We propose a fast algorithm that generates a sequenced near-optimal solution for Multi-Robotic Task Sequencing Problem. We model the problem in the form of a new min (sum-max) Multiple Generalized Traveling Salesman Problem 'min(sum-max) MGTSP' model. Near-optimal solutions are obtained to automatically minimize cycle time in collision-free path. The originality of our method lies in its flexibility and ability to be integrated into industrial processes. In this study, we perform a double optimization in both the task and configuration space. Also, the proposed algorithm is able to automatically plan a collision-free trajectory between two robots' configurations by generating relevant *via points* which minimizes movements duration. Comparing to other approaches, experiment reveal positive results in terms of efficiency and demonstrate the ability of this method to be integrated into existing industrial software.**

***Index Terms*—Industrial Robots, Task Sequencing, Path Planning, Optimization, Intelligent and Flexible Manufacturing**

I. INTRODUCTION

INDUSTRIAL robots are massively used in automotive manufacturing well ahead of other industries such as electronics or food industry. They are among the most efficient machines in terms of reliability, versatility as well as profitability. A large majority of robot applications in the automotive industry consist in performing a set of welding points on multiple cells. This process often presents many optimization stages. For instance, in a car body assembly line, several robots are required to carry out welding sequences simultaneously on the same vehicle (see Fig.1). During the design of the line, assigning the welding points to the robots is a complex, costly expensive and time consuming task. The development phase may require 12 people during 8 months to reach the desired specifications.

Manuscript received: October 15, 2020; Revised: December 30, 2020; Accepted: January 23, 2021.

This paper was recommended for publication by Editor Jingang Yi upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Segula Technologies & Association Nationale Recherche Technologie ANRT.

[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

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

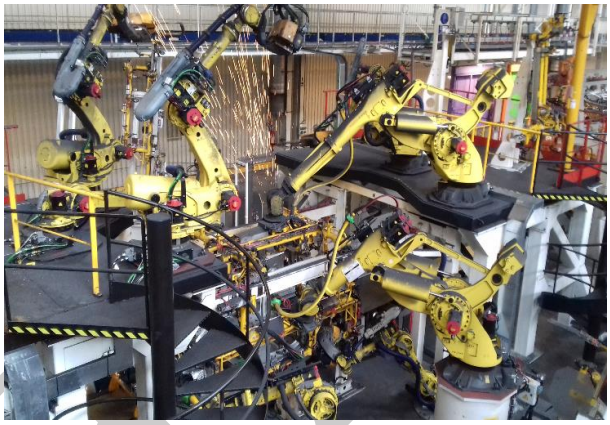

Fig. 1. Automotive production line with several robots. Segula Technologies / PSA line production.

Automatic tools can be helpful to reduce the installation and commissioning times, the production cycle time and also to improve the process implementation feasibility.

In this paper, our main focus is on reducing cycle time while keeping a reasonable robot total movement duration. Optimizing production cycle time can lead to real productivity gains. The relationship between cycle time and the number of products is direct.

The task sequencing problem groups several NP-hard problems together [1] such as determining the optimal sequence through different targets and determining the path with the minimum duration (respecting the constraints of the robot) without collision. Thus, we have divided the problem into sub-problems based on essential steps performed iteratively. Indeed, task sequencing is a phase which often remains manual but could take benefits from optimization especially with different additional robotics factors (Inverse Kinematics 'IK', number of robots, bases location, etc.) see Fig.2. The automation of multi-task sequencing integrating IK solutions in a collision-free way can be considered as a starting point for optimizing the installation phase of the production line. Focusing on this step will undoubtedly lead to real productivity gains.

Our scientific problematic include both "task sequencing" and "production scheduling" problems, since it covers how to assign work evenly between robots to increase the overall flow. In addition, the solution has been developed to be integrated into an existing industrial simulation software.

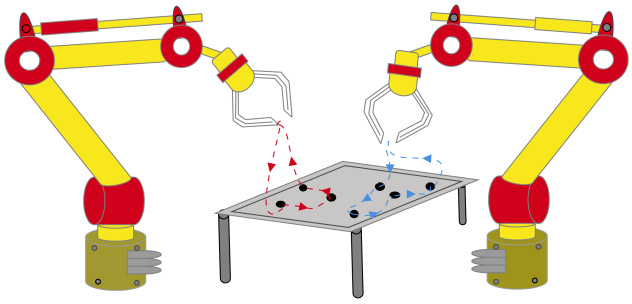

Fig. 2. Task sequencing solution for two robots.

Due to the complexity of the problem as a whole, we have subdivided it into well-defined objectives and consider some assumptions: the different possible configurations are solved beforehand by a solver, the duration of the task is identical in all points, and the collisions between robots are neglected at first glance. Determining a solution that does not initially include the robot-robot collision constraint (valid for the collision of obstacles) is essential in different aspects. First, it is necessary to model and validate the approach in a robot-robot collision-free case to test the performance of the solution. Second, this necessary step is essential to prepare a better integration of an additional step (industrial interlocking system). Third, in the best case, the first initial solution can be directly a global solution without collision. Finally, the velocity profiles can be readjusted to avoid collisions while keeping a high quality solution.

The paper is structured as follows. Section II presents state of the art existing solutions for task sequencing and path planning. Problem definition and formulation under a robotic framework, as well as proposed approaches are studied in section III. Section IV discusses the implemented algorithms. Finally, section V presents and comments the obtained results and gives some conclusions and perspectives for future studies.

II. Related works

There are two main methods for programming robots in an industrial environment: Online and Offline programming. For both methods, the solution depends on the expertise of the engineer. A significant number of research and industrial tools have been proposed to solve the task sequencing problem. However, most of existing approaches address only partially the problem with a limited performance. In addition, they did not provide automatic optimal solutions (implementable in an industry context) for this problem by including all the robotic factors [2]. Indeed, robotic softwares like Delmia, RobCad, Roboguide, etc. focus mainly on optimizing motion planning and neglect the task sequencing problem. The solutions developed today are not able to automatically generate a reliable optimal sequence and even less with additional robotic factors.

A. Task sequencing problem

The order of the task sequence to be performed is often determined manually in different industrial applications. The analogy between finding an optimal sequence and the well-known Travelling Salesman Problem 'TSP'[3] has often been established considering the similarity between the two problems. The goal of the TSP is to find an optimal sequence through a set of points and given distances such that each point is visited exactly once. Several efforts have been made to resolve the TSP problem, see comprehensive surveys [4] [5]. Deterministic approaches gives an exact solution to the problem, while a non-deterministic approach tries to find near-optimal solution.

By analogy to robotics, in TSP the metric to minimize is time rather than distance. The problem may seems similar to the TSP at first glance, but it is not completely identical since several robotics constraints must be considered like: the number of robots, IK, bases locations, obstacle avoidance, etc. All these factors strongly increase the search space and make the exact methods less efficient and only limited to small instances [1]. Multiple efficient heuristic solutions have been developed for this purpose to solve this problem.

In this article, we only refer to the research closest to the originally cited problem. Among the first to have established the relation between TSP-like problems and robotics one can cite Dubousky et al. [6]. Authors have proposed an approach based on temporal movements in order to deal with point-to-point tasks.

There are other TSP-based problems that allow for a closer modeling to the problem. In order to integrate the robot configurations, it is more realistic to formulate the problem as a set of points in each task. This problem refers to Generalized Traveling Salesman Problem 'GTSP' [7]. One of the first studies to establish the link between GTSP and robotics is [8], where a multi-goal path planning problem is presented. Each task is a set of IK solutions, and the goal is to find the optimal sequence at minimum cost visiting only one point of each group, see [9]. In [10], a generalized approach based on Genetic Algorithm 'GA' for manipulating a single robot has been proposed. This research involves the IK of the manipulator. Suárez-Ruiz et al. [11] have proposed a fast algorithm to solve the GTSP problem for a single robot with several targets in a 2D plane.

Given the complexity of the tasks to be performed in the automotive industry, one single robot cannot accomplish them in a limited workspace and time. It is then necessary to carry out an optimization involving several robots.

In a robotic context, several research may seem similar to our main issue such as sequential assembly operations problems [12][13]. The main difference is that these studies consider a logical constraint between operations and are often limited to a small number of instances (robots & operations). Spensieri et al. [14] have developed an iterative approach to solve min-max Multiple Generalized Traveling Salesman Problem '(min-max) MGTSP' and have planned robots trajectories in a collision-free way. This type of formulation does not fully cover the problem considering the different above-cited robotics factors. In general, two separate formulations of MTSP exist in the literature. The first approach minimizes the total cost of all agents (min-sum MTSP) and the second one minimizes the cost of the slowest agent (min-max MTSP).

Namely, the majority of cited approaches considers the cost to be minimized equal to the total duration of the sequences (min-sum) or the duration of the longest tour (min-max). However, in this paper, we intend to go a step further. The total duration and the duration of the robot that has the longest movement duration to perform one sequence of tasks must be considered and optimized simultaneously in order to have a realistic balanced workload while minimizing the cycle time.

B. Automatic path planning

Path planning can be performed in the task or configuration space to find a collision-free path. The complexity is exponential with respect to different robotic factors, as mentioned above. For that purpose, heuristic algorithms that sacrifice completeness for practical effectiveness, can be investigated.

Optimization of parameters can be established by focusing on the dynamics and kinematics of the robot considering *Kinodynamic* constraints such as acceleration limits, joint speeds, etc. We do not focus on this aspect since we consider that the majority of industrial robots already have optimized integrated controllers. The other aspect is to calculate an optimal path that the robot must follow by avoiding collisions in the environment. The robot can move through several *via points* between the start and end points. These *via points* can be chosen in different ways while avoiding obstacles and their choice has a direct impact on the cycle time.

Wurm et al. [15] have proposed an approach to generate an obstacle avoidance between several robots based on the Voronoi diagram. However, this approach is difficult to be applied to large-scale problems. The authors in [16] have planned the trajectory of an end effector using the GA. Xidias et al. [17] have studied the problem of determining the optimal collision-free trajectory by avoiding obstacles, one of the limitations of such an approach is that the considered environment is only 2D. The authors in [18] have used an approach allowing to generate safe *via points* around the obstacle in Cartesian space for a single robot in order to reduce the computational effort, but this method present limitations in crowed spaces for several robots taking into account additional factors. Sampling-based methods, especially the Probabilistic RoadMap 'PRM' and Rapidly exploring Random Tree 'RRT' algorithms, are considered to be the current state of the art for planning a collision-free trajectory in geometrically complex environments. PRM is a random sampling of configurations in C_{space}. A sample is rejected if it is in collision. These samples are then connected to each other forming a roadmap. Finally, the optimal path is calculated by graph-based methods. On the other hand, RRT iteratively builds a tree by expanding it to a randomly sampled configuration respecting the associated constraints. We consider that PRM is more adapted to our application since we will be able to reuse the constructed graph to re-plan different trajectories in the same robotic cell. In contrast, RRT is more suitable for single query applications. It is true that the research mentioned above shows promising results but very few of them consider realistic path planning for multi-robotic aspect and only use reduced models which do not represent the real behavior of the robot in a 3D environment. To overcome these limitations and make our application as realistic as possible, we consider a multi-robotic cell by taking into account the entire structure of the robot in the path planning. The path is generated in the configuration space C_{space} since our criterion is trajectory movement duration while considering the robot's limit joints. We also use an existing industrial simulation software *Roboguide* capable of managing the real controllers of the robot and thus generating an almost real trajectory in CAD environments.

III. Problem description and formulation

In order to solve the problem discussed above, we propose to model the problem in the form of the new min (sum-max) MGTSP. We consider n targets $\{p_1, p_2, .., p_i, .., p_n\}$ which must be assigned to a set of N robots $R = \{r_1, r_2, ..., r_k, ..., r_N\}$. Each target i must be visited once by one of the robots. Each robot r_k has a set of possible configurations m for each target. Every robot has a starting position and configuration $p_0^{r_k}$ and $J_0^{r_k}$. The robots start and return to their same starting point and configuration. One point must be assigned to at least one robot. Initially, the objective is to simultaneously optimize the total cost and the maximum cost traversed by the robots in the task space. Note that the input data come from points reachable by all robots (to test the performance of the algorithm in the worst case). Then, starting from the determined order of the sequence, we perform an other optimization in the configuration space C_{space}. So, our problem can be formalized as:

Input: a set of n targets $Data = \{p_1, p_2, .., p_i, .., p_n\}$ where each p_i is associated with a position (x_i, y_i, z_i) and orientation (W_i, P_i, R_i).
a set of IK for every robot in every point:
$J_{p_i}^{r_k} = \{J_1^{r_k}, J_2^{r_k}, ..., J_j^{r_k}, ..., J_m^{r_k}\}$
with $J_j^{r_k} = \{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6\}$
a set of starting position & configuration:
$I_{pos} = \{p_0^{r_1}, p_0^{r_2}, ..., p_0^{r_N}\}$ & $I_{config} = \{J_0^{r_1}, J_0^{r_2}, ..., J_0^{r_N}\}$
Goal: find an optimal sequence in the task and configuration space taking as a criterion the total movement duration and the cycle time in collision-free way.

Targets and configurations can be represented in the form of a complete and undirect graph $G = (V, E)$ where V is the set of vertices representing targets (or the sampled configurations for the path planning) and E is the set of edges representing distances (or time). It can be noted that this graph can be formulated into a *weighted* graph where the weight is a cost function $C : E \rightarrow \mathbb{R}^+$. The cost function $c(i, j)$ between two points i and j, models the time required for the robot to move from one task to another with a start configuration to the following one. The problem is divided into three steps which are performed iteratively:

Step 1: Calculation of a solution managing the assignment of tasks and returns the sequence order for each robot as:

$$S_{pos}^{r_k} = \{p_0^{r_k}, s_1, s_2, ..., s_n, p_0^{r_k}\}$$

where s_i is the solution point of order i.

Step 2: Optimization of the total movements duration in C_{space} based on the solution from the previous step

$$S_{config}^{r_k} = \{J_0^{r_k}, q_1, ..., q_n, J_0^{r_k}\}$$

where q_i is the configuration solution of order i.

Step 3: Generation of K relevant *via points* V between the configuration q_i and q_{i+1} if a collision is present in the path while minimizing robot movement:

$$Path_{\{q_i,q_{i+1}\}} = \{q_i, V_1, V_2, ..., V_K, q_{i+1}\}$$

In order to ensure the quality of the solution, these three steps are carried out iteratively. The quality of the initial solution may be affected when adding more via points. For this reason, the sequence is recalculated in step 1 when a collision occurs. Please note that only the duration of the trajectory presenting a collision is replaced by the new movement duration (including new via points). Then, we recalculate the configuration solutions in step 2 and check if there is a new collision. These steps are repeated until there is no collision.

We define a binary variable $\delta^{r_k}_{(i,j)}$ where the value indicates whether the robot performs the next task or not. $\delta^{r_k}_{(i,j)} = 1$ if and only if the robot r_k moves from point i to point j, otherwise 0 which refers to the equation (1), the goal is to minimize the sum of the costs of each robot and the cost of the robot that have the longest movement duration.

The problem can be formulated mathematically as follows:

$$\delta_k(i,j) \begin{cases} 1, \text{if the robot leaves from a point i to j} \\ 0, \text{otherwise} \end{cases} \quad (1)$$

$$C_{r_k} = c(p^{r_k}_0, s_1) + \sum_{i=1}^{n-1} (\delta^{r_k}_{(i,i+1)} c(s_i, s_{i+1})) + c(s_{n-1}, p^{r_k}_0) \quad (2)$$

$$C_{minsum} = \min(\sum_{k=1}^{N} C_{r_k}) \quad (3)$$

$$C_{minmax} = \min(\max(C_{r_k})) \text{ for every } r_k \quad (4)$$

Equation (2) formulates the cost of a single robot r_k. The criterion objective is defined in equations (3) and (4).

IV. Proposed Approach

A comprehensive solution to the problem in robotics within realistic deadlines and IT resources is a difficult task, especially for large instances. So, the proposed approach is based on a non-deterministic GA due to the NP-hard nature of the task sequencing problem. The GA algorithm simulates the natural evolution process. The idea is to get a near-optimal solution to solve the min (sum-max) MGTSP problem. Based on the representation discussed in the previous section, the proposed approach offers flexibility to evolve the algorithm with other additional robotic factors. The proposed algorithm operates on a set of solutions and uses GA mechanisms to converge towards near-optimal solutions.

A. Step 1 - Algorithm development in task space

One of the essential factors influencing the quality of the solution obtained from GA is the diversity of the research space. Each chromosome is composed of a numeric vector representing the number of targets to be visited by the robot.

Since the l_{th} chromosome Ch_l consists of the target solution to be visited for each robot, it is formulated as follows:

$$Ch_l = \{S^{r_1}_{pos}, S^{r_2}_{pos}, ..., S^{r_N}_{pos}\}$$

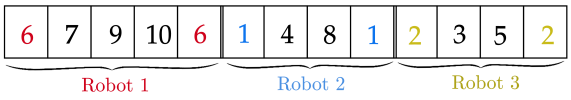

Fig. 3. Example of representation of the chromosome for 3 robots.

We can note that the starting positions for each robot are included in the representation in order to take into account the task cost to reach the following target. For example, the case of 10 targets and 3 robots as depicted in Fig.3, the first robot will perform tasks 6, 7, 9, 10 and 6, the second robot will perform tasks 1, 4, 8 and 1, and the third robot carry out the tasks 2, 3, 5 and 2.

The initial population containing p chromosomes is randomly generated in order to maintain the diversity of the population which is expressed as follows:

$$Pop = \{Ch_1, Ch_2, ..., Ch_p\}$$

We propose to estimate the quality of the solution by combining two criteria:

- Find the sequence for each robot while the total cost is minimized.
- Require the robot that have the longest movement duration (longest tour) to perform one sequence of tasks to be as short as possible.

Based on equations (2) and (3), the cost function can be written as follow:

$$Cost = \alpha . C_{minsum} + \beta . C_{minmax} \quad (5)$$

C_{minsum} and C_{minmax} are the two criteria of the sequence Ch_l. The coefficients α and β allow to promote a criterion over the other.

The detail of this procedure is presented in Algorithm 1.

B. Step 2 - Algorithm development in configuration space

IK solutions for each robot can be presented in the form of an undirect graph. This graph is characterized by a similar

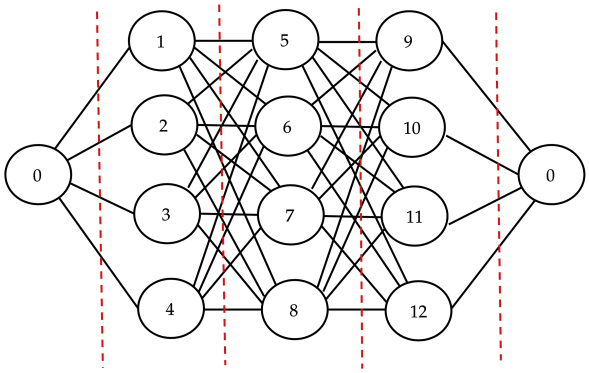

Fig. 4. Chromosome coding for a problem of 4 targets and 4 possible configurations in Cspace.

Algorithm 1 Optimization in task space

Input: a set of n targets $Data = \{p_1, p_2, .., p_i, .., p_n\}$, a set of starting position $I_{pos} = \{p_0^{r_1}, p_0^{r_2}, ..., p_0^{r_N}\}$

Output: A minimal cost cyclic $S_{pos}^{r_k}$ with double objective value

$t \leftarrow 0$
Initialize α, β
Initialize parameters of GA
Generate an initial population Pop_t of chromosomes Ch
Evaluate fitness (cost function equation (5))
while termination condition not met **do**
    Select individuals from Pop_t
    Random selection of indexes to be mutated
    Locally Mutate the solution of a single robot
    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 corresponding fitness value

structure of a Neural Network 'NN' consisting of n layers corresponding to the n points, each layer forms the possible (realistic) configurations of the robot to reach a point p_i in the task space. It can be noted that the layers are classified according to the results of Algorithm 1. The "input layer" and the "output layer" are the starting configuration $J_0^{r_k}$ since each robot must return to its initial configuration. The weight of NN can be given by the cost $c^{r_k}(q_i, q_{i+1})$ required to move from one configuration q_i to another q_{i+1} in the C_{space}.

We have developed a solution based on an industrial specification:

- A flexible approach dealing with a significant number of IK solutions for multi-robot systems.
- An approach that converges in reasonable time to find a near-optimal solution in the C_{space}.

To better exploit the functionalities of the GA, the solution has been formulated as a chromosome. Each gene of this chromosome represents the number of the neuron referring to a solution with a unique configuration of a point p_i. Our method is based on a modified genetic algorithm in order

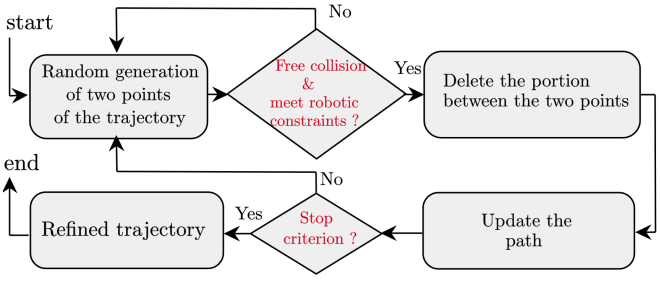

Fig. 5. Generation of relevant via points process.

to visit each layer v_i once in the C_{space}. For instance, the chromosome coding for a problem with four targets consisting of four possible configurations is illustrated in the Fig.4. This procedure is presented in Algorithm 2.

The choice of the metric for two given configurations q_i and q_{i+1} is an important criterion that has a direct impact on the final solution. Given the complexity of determining the optimal time between each edge of the graph, we consider the use of approximate metric. The cost is approximated as follows $c^{r_k}(q_i^j, q_{i+1}^j)$:

$$c^{r_k}_{(q_i^j, q_{i+1}^j)} = max\left(\frac{|q_{i+1} - q_i|}{V_{max}^{j,r_k}}\right) \tag{6}$$

where V_{max}^{j,r_k} is the maximum joint velocity for joint j corresponding to the robot r_k.

Algorithm 2 Optimization in C_{space}

Input: the best chromosome Ch_l calculated from the Algorithm 1
A set of IK $J_{p_i}^{r_k}$ for each robot in every position & starting configuration I_{config}

Output: A minimal cyclic cost $S_{config}^{r_k}$

Initialize parameters of GA
Import the order of the solution $S_{pos}^{r_k}$ given by algorithm 1
Split the configurations
Number the ordered configurations to create the neurons of the neural network
Define the neural network with the given layers
Define chromosome representation
Optimize the total movement duration for a robot r_k using the modified GA bellow:
    a: Generate an initial Pop_t' respecting the order of NN
    while termination condition not met
    b: Select individuals from Pop_t'
    c: Cross configuration solutions
    d: Evaluate fitness (total movements duration)
    e: $Pop_{t+1}' \leftarrow$ new individuals
    f: $t' \leftarrow t'+1$
    end while
Return the best IK solution $S_{config}^{r_k}$ for each robot and its corresponding fitness value

C. Step 3 - Algorithm development for path planning

The generation of new via-points between two tasks for a robot r_k is performed to avoid obstacles. This step consists of two essential sub-steps:

The first step 3.1 generates a roadmap and finds the optimal path using graph-based methods. Since we consider our industrial application to be part of a multi-query problem, where we have to define several trajectories in the same environment (industrial cell), we have decided to use an approach based on PRM. Having N robots r in a workcell, N graph construction are needed to sample every configuration space $C_{space}^{r_k}$ for each robot. We have built a roadmap from G^{r_k} where the vertices are configurations of $C_{free}^{r_k}$. The samples are randomly

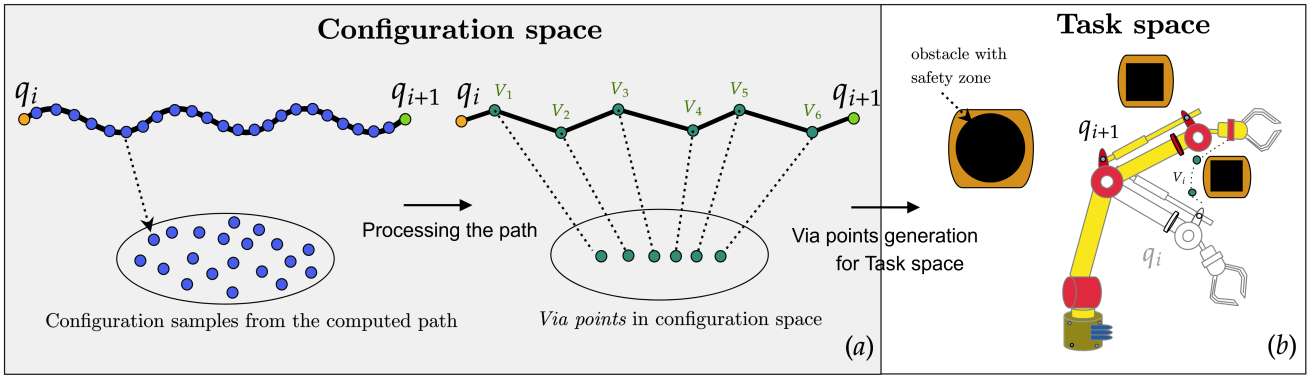

Fig. 6. Path planning process to generate relevant via points in configuration space (a) and its application in task space (b).

generated using a uniform distribution while respecting the permitted configurations of each robot. It should be noted that the obstacles have been amplified (safety zone) so that the robot remains at a safe distance from the obstacle during the avoidance. Then, we calculate the new collision-free path including several configurations using graph-based algorithms such as the well-known A*.

The path minimizing the duration of the movement is obtained in C_{space} including several configurations to avoid the obstacle. In practice, the robot needs only a few via-points to avoid an obstacle. Therefore, step 3.2 reduces the number of configurations in this new path (from step 3.1). The process (Fig.5) is done by randomly taking two configuration points from the path and testing whether the trajectory between these two configurations is free of collision and respects the constraints (acceleration limits, maximum articulation speeds, etc). The trajectory is shortcut when the latter conditions are met. This means the intermediate points between these two configurations are removed. When the previously mentioned conditions are not met, two new configurations are randomly taken from the path until the conditions are verified. After shortcutting the path, a new path is obtained with a few necessary configurations to avoid the obstacle. We then convert these configuration points into Cartesian space (forward kinematics) which determine the final via points (Fig.6).

V. Results

In this section, the performance of the developed algorithms has been assessed. The experiments focus on demonstrating the ability of such algorithms to solve a real industrial problem consisting of processing several welding points by multiple robots sharing the same workspace.

The algorithms demonstrated their ability to generate a point order for each robot. The latter considers the most adapted configuration in every point in a collision-free way, and minimizes the discussed optimization criteria. We applied our algorithms to a real industrial simulation software ROBOGUIDE integrating the real robot controllers for reliable robot simulations. The six-axis Fanuc R-2000-iC/165F manipulator robots were used for simulation on a PC powered by a Core i7-9750H at 2.6 GHz and 32 GB of RAM. In

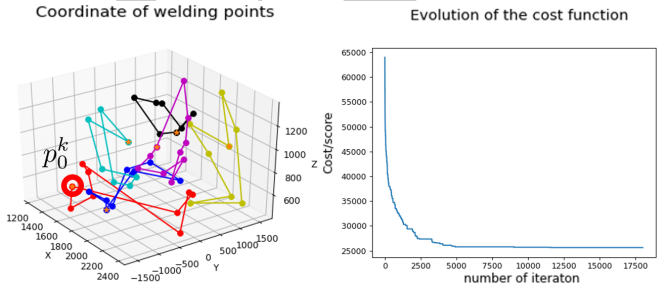

Fig. 7. Example of a solution proposed by the algorithm developed for problem 4 (left) & the evolution of the cost function as a function of the number of iterations (right).

addition, in order to manage collision information from the CAD environment, a communication was established between the algorithms coded in Python and the industrial software via KAREL scripts. Furthermore, for the purpose of reducing random fluctuations, more than one evaluation was performed for the same problem. All simulations were executed in three-dimensional (3D) space. Four problems have been studied, named respectively problem 1, 2, 3 and 4, dealing with 10, 20, 30 tasks for three robots and 50 tasks for six robots.

Table 1 shows the parameters chosen for our experiments as well as the movement duration for each robot. It should be noted that if the selection and mutation value (local or global) are very high, the algorithm will perform high steps following each iteration and will not converge to a good solution. An example of a proposed solution for problem 4 is illustrated in Fig.7 in 3D Cartesian space as well as the evolution of the cost function discussed in the section (4-a). In addition, the solutions of the possible configurations are communicated by the simulation software. Then, a selection of suitable configurations minimizing cycle time was established by the algorithm discussed in the section (4-b).

Different test data exist in order to perform a comparative analysis for TSP-based problems in the field of combinatorial optimization. Our work does not aim to do better than existing state-of-the-art approaches for solving a TSP problem, but rather wants to provide a global solution integrating different robotic factors to solve an industrial problem. In the field

	Parameters					Movement time (s)					
	Cnvg. value	Pop. size	Slct. prob	Mut. prob	Cross. prob	Rbt 1	Rbt 2	Rbt 3	Rbt 4	Rbt 5	Rbt 6
Problem 1	1500	150	60%	70%	70%	1.85	0.46	2.68			
Problem 2	2000	150	60%	70%	70%	Rbt 1	Rbt 2	Rbt 3			
						2.18	0.56	1.77			
Problem 3	4000	150	60%	70%	70%	Rbt 1	Rbt 2	Rbt 3			
						1.95	2.14	1.17			
Problem 4	7000	150	60%	70%	70%	Rbt 1	Rbt 2	Rbt 3	Rbt 4	Rbt 5	Rbt 6
						1.31	0.173	1.44	1.97	1.58	1.65

TABLE I
USED PARAMETERS AND THE CORRESPONDING MOVEMENT TIMES FOR EACH ROBOT

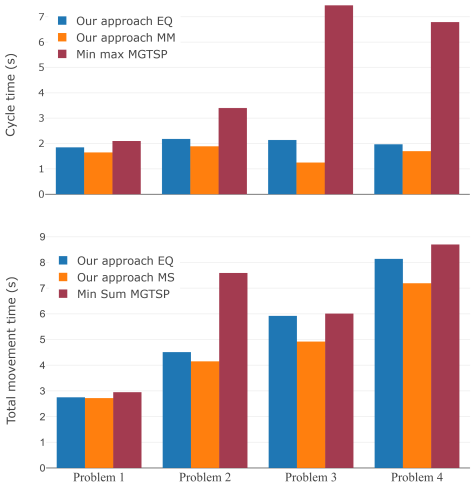

Fig. 8. Maximum cycle time and total movement time for different methods.

of robotics, there are studies that primarily evaluate the performance of path planners. In our case, it remains difficult to make a fair comparison between studies that are similar to our problem. The complexity lies in the way in which the problem was approached, the robotic constraints, the type of industrial robot used, the industrial environment, etc. No global reference exists therefore for multi-task sequencing problem. Thus, we decided to compare our approach to other research that has been based on two formulations min-sum MGTSP and min-max MGTSP.

Since these approaches do not provide any public implementation, we decided to reproduce the results for the same problem where we consider separately the cost of the total robot movement and the cost of the robot that have the longest

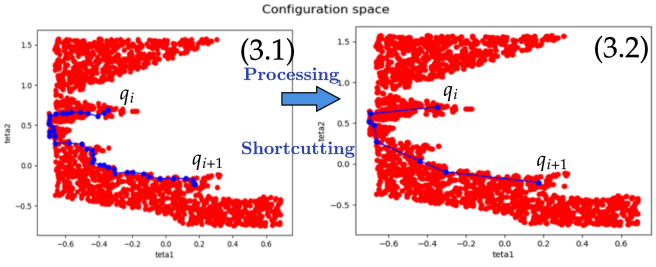

Fig. 9. PRM construction (red) and the proposed trajectory (blue) in the configuration space before and after the processing phase (3.1) and (3.2).

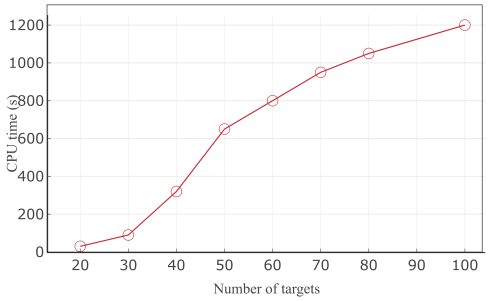

Fig. 10. CPU calculation time as a function of target numbers.

movement duration to perform one sequence of tasks. Fig.8 shows the cycle time and movement time provided by our method compared to other formulations. We have studied the case where the cycle time and movement duration must be fairly minimized (EQ), and the case where we favor cycle time over total movement duration (MM) by increasing their corresponding factors and vice versa (MS). The cycle time values were improved by 11.9%, 35.9%, 71.2% and 70.98% for the respective problems 1, 2, 3 and 4 compared to the results of the other formulations. We have also shown the ability of our approach to adapt and propose a new solution with regards to the favored criterion in a reasonable time (Fig.10). For example, we save $1s$ of cycle time in (MM) compared to (EQ) by sacrificing and increasing the total robot movement duration by 5%.

Fig.9 illustrates an example of automatic path planning from q_i to q_{i+1} for a single robot considering joint limitations (Joint 2 and 3 are plotted). The samples in red correspond to the PRM of the robot r_k, $7,000$ nodes of the graph were tested in $1,220sec$ and the trajectory in blue corresponds to the one calculated by our algorithm. Then, the processing phase is carried out to generate the relevant via points. The steps are performed iteratively to ensure the quality of the obtained solution.

Subsequently, we implemented our algorithm on ROBOGUIDE to simulate the real trajectory of the robot. Fig.11 shows the capture sequences for a problem of two robots where several obstacles are present between the points. The algorithm succeeded in assigning points for each robot, chose the most suitable configuration and planned a smooth trajectory without collision while minimizing cycle time. Since our solution is incremental, the robot-robot collision is managed like in industrial cells by an optimized interlock system.

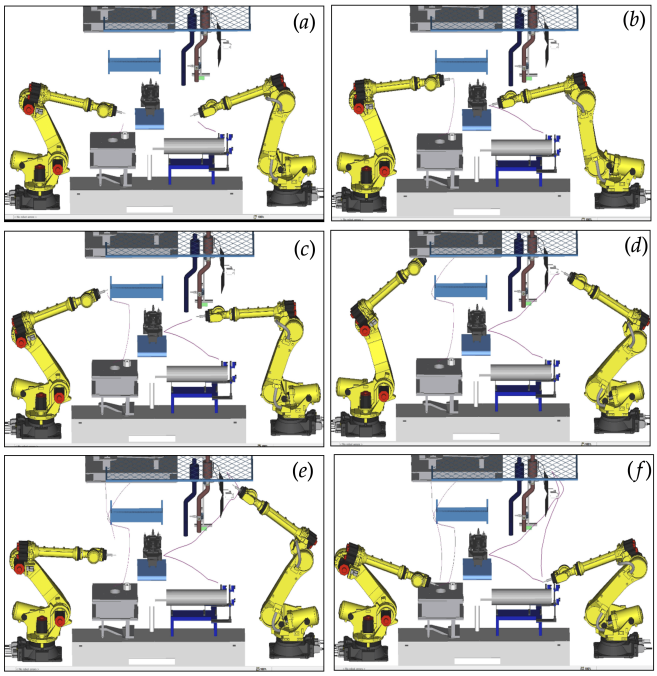

Fig. 11. Motion captures of the global solution proposed by the algorithm for a problem of two robots on Roboguide.

VI. CONCLUSIONS

This work was motivated by the industrial need to automate and optimize the task sequencing problem in automotive stations. To the best of our knowledge, no existing automatic approach has provided a reliable solution that applies to current industrial software for the whole problem. Existing studies do not allow the resolution of a large number of targets while integrating multiple configurations in a collision-free way for a multi-robot system. To solve the multi-task sequencing problem, we have developed a new approach based on the genetic algorithm. The developed approach can solve up to 100 targets, while taking into account several robotic factors such as sequence order, configuration solutions (up to 10 per target) and the movement duration of each robot. All these parameters have a direct impact on the cycle time. The discussed method is able to generate via points to provide trajectory without collision in cluttered environment by reducing movement duration. The obtained results reduce both cycle time and total robot movement duration. Our approach presents a better cycle time compared to other single-criterion optimization, especially in more complex problems. We have noticed that the procedure to generate the configuration spaces for each robot can be slow for a complex environment. However, we considered that the pre-processing phase did not present any computation time concerns since this generation is done only once and the industrial environment is often unchanged when the production line is started.

Our future studies will focus mainly on the integration of an optimized interlock system adapted to the industry. In addition, we will integrate the robot-robot collision system into our algorithm with a reevaluation of the discussed steps, iteratively. Moreover, we will test the possibility of obtaining a better solution by taking into account several additional robotic factors such as the direction of the sequence execution, different task duration, accessibility constraint, etc.

ACKNOWLEDGMENT

This work has been supported by Segula Matra Automotive & ANRT. 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] Pan, Z., Polden, J., Larkin, N., Van Duin, S., & Norrish, J. (2012). Recent progress on programming methods for industrial robots. Robotics and Computer-Integrated Manufacturing, 28(2), 87-94.
[3] Applegate, D. L., Bixby, R. E., Chvatal, V., & Cook, W. J. (2006). The traveling salesman problem: a computational study. Princeton university press.
[4] J. S. B. Mitchell. Shortest paths and networks. In J. E. Goodman and J. O'Rourke, editors, Handbook of Discrete and Computational Geometry (2nd Edition), chapter 27, pages 607–641. Chapman & Hall/CRC, Boca Raton, FL, 2004.
[5] Arora, S. (2003). Approximation schemes for NP-hard geometric optimization problems: A survey. Mathematical Programming, 97(1-2), 43-69.
[6] Dubowsky, S., & Blubaugh, T. D. (1989). Planning time-optimal robotic manipulator motions and work places for point-to-point tasks. IEEE Transactions on Robotics and Automation, 5(3), 377-381.
[7] Laporte, G., & Nobert, Y. (1983). Generalized travelling salesman problem through n sets of nodes: an integer programming approach. INFOR: Information Systems and Operational Research, 21(1), 61-75.
[8] Wurll, C., Henrich, D., Wörn,, H.: Multi-goal path planning for industrial robots. In: International Conference on Robotics and Application (RA99). Santa Barbara, USA (1999).
[9] Laporte, G., & Nobert, Y. (1983). Generalized travelling salesman problem through n sets of nodes: an integer programming approach. INFOR: Information Systems and Operational Research, 21(1), 61-75.
[10] Zacharia, P. T., & Aspragathos, N. A. (2005). Optimal robot task scheduling based on genetic algorithms. Robotics and Computer-Integrated Manufacturing, 21(1), 67-79.
[11] Suárez-Ruiz, F., Lembono, T. S., & Pham, Q. C. (2018, May). Robotsp–a fast solution to the robotic task sequencing problem. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1611-1616). IEEE.
[12] Dogar, M., Spielberg, A., Baker, S., & Rus, D. (2019). Multi-robot grasp planning for sequential assembly operations. Autonomous Robots, 43(3), 649-664.
[13] Marvel, J. A., Bostelman, R., & Falco, J. (2018). Multi-robot assembly strategies and metrics. ACM Computing Surveys (CSUR), 51(1), 1-32.
[14] Spensieri, D., Carlson, J. S., Ekstedt, F., & Bohlin, R. (2015). An iterative approach for collision free routing and scheduling in multirobot stations. IEEE Transactions on Automation science and Engineering, 13(2), 950-962.
[15] Wurm, K. M., Stachniss, C., & Burgard, W. (2008, September). Coordinated multi-robot exploration using a segmentation of the environment. In 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1160-1165). IEEE.
[16] Tian, L., & Collins, C. (2003). Motion planning for redundant manipulators using a floating point genetic algorithm. Journal of Intelligent and Robotic Systems, 38(3-4), 297-312.
[17] Xidias, E.K., Zacharia, P.T., Aspragathos, N.A.: Time optimal task scheduling for articulated manipulators in environments cluttered with obstacles. In: Robotica, vol. 28, pp. 427–440. Cambridge University Press (2010).
[18] Bottin, M., & Rosati, G. (2019). Trajectory optimization of a redundant serial robot using cartesian via points and kinematic decoupling. Robotics, 8(4), 101.