Optimization of robot coordination using temporal synchronization

Carlos Rodríguez, Andrés Montaño and Raúl Suárez
Institute of Industrial and Control Engineering (IOC)
Universitat Politècnica de Catalunya (UPC) - Barcelona Tech
Barcelona, Spain, (carlos.rodriguez.p, andres.montano, raul.suarez@upc.edu).

***Abstract*—This work presents an optimization method applied to robot coordination using temporal synchronization. The coordination process considers the possibility of using multi-robot systems in which each robot executes individually planned tasks in a shared environment. The coordination process generates a curve in a discretized coordination space that contains the sequence of coordinated configurations of the robots, this curve can be optimized in order to minimize the backward movements of the robots during their path execution. The optimization method was implemented for a two arm robotic system, a comparison between the executions with and without optimization was performed, and two illustrative experiments are presented in this paper.**

I. Introduction

The efficient coordination of several robot arms to avoid collisions while they carry out independent tasks in a common workspace is a frequent problem in several robotic fields, both in industrial and service applications. This work proposes an approach to solve this problem modifying the temporal evolution of the robots along their precomputed geometrical paths and optimizing it to minimize the backward movements of the robots over their paths.

Coordination problems can be solved by simultaneously planning the trajectories of all the robots in the shared workspace (centralized approaches), or by independently planning the trajectories of each robot and then applying an additional coordination phase (either off-line or on-line) to prevent potential collisions between them (decoupled approaches) [1].

An analysis and classification of multiple robot coordination methods was presented by Todt et al. [2], showing that the motion coordination algorithms can be applied on different representations of the workspace (e.g. physical space, composite configuration space, composite configuration-time space, path-time space or coordination space).

The centralized approaches are complete but they involve a higher number of Degrees of Freedom (*DOF*) and therefore they are computationally more expensive than the decoupled approaches, which are then considered from the practical point of view. Sanchez and Latombe [3] presented a comparative study between centralized and decoupled planning for multi-robot systems using a PRM planner, which concludes that in applications with a rather tight robot coordination the use of a centralized planner is more desirable. Nevertheless, as mentioned above, centralized methods are not practical for on-line motion planning because they involve a large number of *DOF*, and therefore a decoupled approach should be used.

On the other hand, within the decoupled approaches, O'Donnell and Lozano-Peréz [4] addressed the motion coordination problem adding a precomputed time delay at the beginning of the movements that guarantees the collision avoidance between the robots. Lee et al. [5] and Yamamoto and Marushima [6] found an optimized coordination curve using dynamic programming. Their main goal is the minimization of the execution time of the tasks, considering the dynamics of the robots and the torque constraints. The obtained coordination curve is used to design the velocity profile for each robot so that collisions are avoided. Cheng [7] introduced an adjustment in the geometric paths identifying the regions of the space swept by the robots and then modifying the paths planned a priori so that the robots do not occupy these regions simultaneously, if it is not possible to modify the robot paths then their execution time is modified so that the conflictive regions are occupied by one robot at a time. Lee et al. [8] proposed an event-based approach for on-line and off-line collision-free trajectory planning for dual-arm assembly systems based on a fast geometric collision detection algorithm. More recently, Chiddarwar and Babu [9] introduced a method that solves the robot conflicts based on a path modification approach. The conflictive paths are modified based on the robot positions in a dynamically computed path modification sequence. In off-line approaches, as those mentioned above, the objective is to plan time or energy optimal motion trajectories because the computation time is not an important factor, but, in on-line approaches, this optimization cannot be achieved because the complete robot plan may be unknown and the computational time of the motion optimization is usually too large.

The optimization approach proposed here is based on the work done by Montaño and Suárez [10] that introduced an online coordination method for multi-robot systems using temporal synchronization, the method generates a curve in a discretized coordination space that contains the sequence of coordinated configurations of the robots, but it is not an optimal solution.

II. Discretized Coordination Space and Coordination procedure

This section summarizes the coordination procedure previously presented in [10] which is the base of the proposed approach. Consider n robots R_i, $i \in \{1,...,n\}$ which have

This work was partially supported by the Spanish Government through the projects DPI2010-15446, DPI2011-22471 and DPI2013-40882-P.

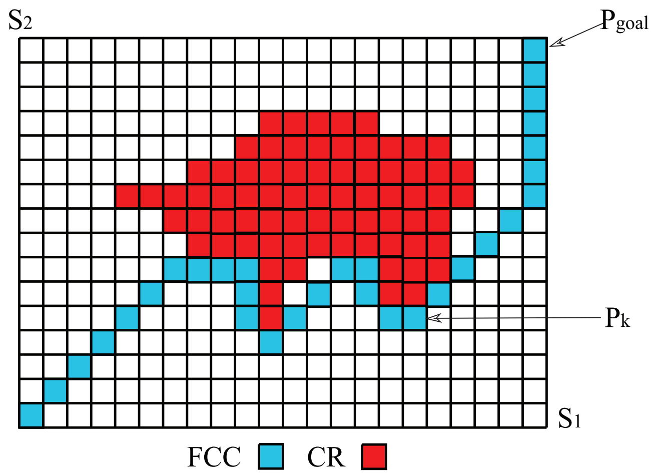

Fig. 1: Bidimensional Discretized Coordination Space DCS for two robots, showing the Collision Region CR and a Collision-free Coordination Curve FCC. In a real problem CR is not known a priori, but here, for illustrative purposes, it is shown as it would be completely known.

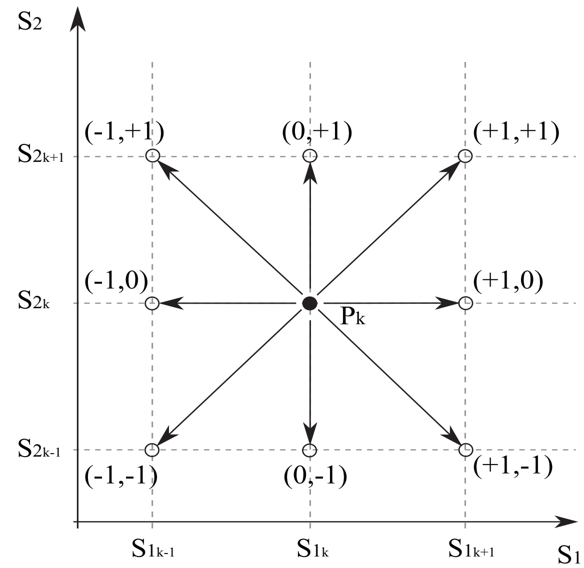

Fig. 2: The eight possible motion directions in a Discretized Coordination Space DCS for two robots.

to execute their tasks in a shared workspace following some assigned geometric paths (path_i) computed independently. The geometric path for each robot can be expressed using a path parameter that uniquely identifies the robot configuration along the path as $q_i = \mathrm{path}_i(s_i)$, where s_i denotes the traveling length along the path, with $s_{i_{\max}}$ being the entire path length. The space defined by the points $P = (s_1, ..., s_i, ..., s_n)$, with $0 \leq s_i \leq s_{i_{\max}}$, is called Coordination Space (CS) [11], i.e. CS is the n-dimensional space determined by the n path parameters s_i of the n robots. CS can be discretized considering a finite set of points $P_k = (s_{1_k}, ..., s_{i_k}, ..., s_{n_k})$, with $0 \leq s_{i_k} \leq s_{i_{k_{\max}}}$ obtaining a Discretized Coordination Space (DCS). The origin of DCS is the point $P_0 = (0, ..., 0)$ and the point at which the robots complete their tasks is $P_{\mathrm{goal}} = (s_{1_{k_{\max}}}, ..., s_{n_{k_{\max}}})$. The set of points in DCS representing collision configurations of the robots is called Collision Region (CR). The relative motion between the robots is described by a Coordination Curve (CC) in DCS; a CC may allow robots to move backward, which may be necessary for on-line collision avoidance [8]. If a CC does not pass through CR it is called a Collision-free Coordination Curve (FCC), i.e. a FCC is a set of sequential points $P_k \in$ DCS such that $\forall k \quad P_k \notin$ CR. Fig. 1 illustrates the DCS for two robots, a Collision Region CR and a Collision-free Coordination Curve FCC. It is assumed that if two consecutive configurations of DCS do not imply collisions then the transition between them are also free of collisions.

From a point P_k there are different possible movement directions in DCS, each of them is represented by a Motion Direction (MD). For n robots the number of possible MDs is $\mathrm{N_{md}} = 3^n - 1$. Fig. 2 shows a piece of DCS for two robots, at any generic point P_k there are eight different possible MDs to move to another point P_{k+1} in DCS (obviously, with the exception of points with coordinates s_{i_0} or $s_{i_{k_{\max}}}$). In this 2-dimensional DCS a diagonal MD going up and right indicates that both robots are moved forward, this is the default desired motion direction, i.e. direction (+1,+1). In the other diagonal cases one of the robots move forward while the other move backward i.e. directions (+1,-1), (-1,+1) and in the worst case both robots move backward, i.e. direction (-1,-1). The other cases of MD are equivalent to stop one robot while the other advances, i.e. directions (0,+1), (+1,0), or moves back, i.e. directions (0,-1) and (-1,0).

The coordination of the robots is performed through the generation of an FCC in DCS. Starting from the initial point, the next point in FCC is selected using a MD and a collision check is performed in order to test whether this point belongs to the free space in DCS or to CR. Then, if the tested point belong to the free space, it is stored in a sequence, generating an FCC. It is assumed that $\mathrm{N_{cc}} > 1$ collision checks can be done while each robot advances one step in it geometric path. The number of points in the portion of FCC between the last added point to FCC and the point representing the current position of the robots, called Explored Window EW, limits the number of points in DCS that can be analyzed in the optimization process before the robots reach the portion of FCC being optimized.

III. Overview of the Proposed Approach

The aim of the proposed optimization process is the minimization of the backward movements of the robots during the execution of their tasks according to their independent plans, since this involves an extra energy consumption and may require more time. The fact that one or more robots move backward occurs because during the exploration of DCS searching for an FCC a CR was found, and at least one robot must move backward to avoid the collision with other robots in the environment.

Looking to the motion directions MD in FCC it can be known the exact point P_k in which one or more robots recede and, at this point the optimization of FCC can be launched. The optimization process is performed as follows.

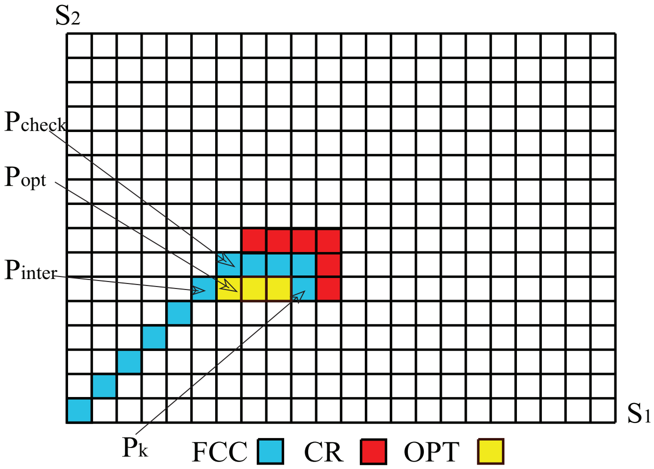

Fig. 3: Elements involved in the optimization process in DCS.

First, it is necessary to define the set of points OPT that contains possible new points of FCC that would replace some points in the current FCC (yellow points in Fig. 3). Once a point P_k involving a backward movement of a robot R_i was added to FCC, the set OPT is composed by the points in DCS linking P_k with a point $P_{inter} \in$ FCC that do not imply a movement of R_i (see Fig. 3). In case that more than one robot move backward P_{inter} is selected to avoid the backward movement of the robot with higher priority. In order to optimize FCC it is necessary to check whether the current robot configuration in FCC has not exceed P_{inter}, if this condition is true the optimization can be done, otherwise the optimization of this portion of FCC is not feasible. As for the coordination process, in the optimization process $\mathrm{N_{cc}}$ represents the number of collision checks that can be done while the robots advance one step in FCC. The optimization begins from the point $P_{opt} \in$ OPT closest to P_{inter}, if $P_{opt} \notin$ CR then it is added to FCC replacing the point P_{check} that follows P_{inter} in the original FCC (see Fig. 3). The process is repeated until OPT was completely included in FCC or the robots reach the point acting as P_{inter} in an optimization step.

In order to provide a visual support of the optimization process explained above, Fig. 4 shows a complete example of the optimization process of an FCC in DCS for two robots.

IV. Optimization procedure

The optimization approach introduced above for n robots is particularized here for a cell with two robots R_1 and R_2. In this case, DCS is 2-dimensional, and even when the coordination is done on-line, the corresponding paths path_1 and path_2 are independently computed off-line for the desired tasks assigned to each robot, thus $s_{1_{k_{\max}}}$ and $s_{2_{k_{\max}}}$ are known.

Algorithm 1 shows the procedure of the proposed approach, which must be executed by each robot R_i. It requires as input the geometric paths $\mathrm{path}_i, i = 1, 2$. P_{goal} is the point at which the tasks were completed and there were no more movements to be coordinated. In the algorithm there are two main parts, the coordination of movements and the optimization of the coordinated movements, which is the main contribution of this work.

The coordination implies the exploration of DCS [10], selecting a point P_{k+1}, checking it for collisions, and adding it to FCC if it is collision free. When P_{k+1} implies a backward movement of a robot an optimization flag is activated. In order to determine the next point P_{k+1} of FCC, a state diagram is used, with the nodes representing the MD and the transitions defined according to whether the resulting movement produces a collision configuration or not.

The optimization process is described from lines 17 to 29 in Algorithm 1. It determines and checks the point P_{opt} in OPT in order to replace the section of FCC that produces a backward movement of a robot. The subset OPT is generated as explained in Section III. Then, P_{inter} is determined and if the robots have not passed through P_{inter} the optimization can be started, otherwise the optimization is not feasible. The optimization procedure is executed as follows. P_{opt} and P_{ckeck} are identified, then P_{opt} is checked for collision and if it is in the free space in DCS then P_{ckeck} is replaced by P_{opt} in FCC. This process is repeated until OPT was completely included in FCC or the robots reach the point acting as P_{inter}.

The coordination and optimization actions mentioned above are executed until the goal of each robot is reached.

V. Experimental Results

The simulated environment used in this work corresponds to a robotic cell of the Institute of Industrial and Control Engineering (IOC), where there are two robots Stäubli TX-90 with 6 *DOF* equipped with a Schunk Anthropomorphic Hand (SAH) with 13 *DOF*, and a Schunk Dexterous Hand (SDH2) [12] with 7 *DOF*.

The implementation is based on ROS for the communication layer, Qt libraries for the user interface, Coin3D for the graphical rendering and PQP for the collision detection. For the graphical simulations the robots were modeled using triangular meshes. The path planning is done using the home-developed path planning framework called *the Kautham Project* [13]. This framework provides the developer with several tools needed for the development of planners, like, for instance, direct and inverse kinematic models of the robots and hands, random and deterministic sampling methods [14], metrics to evaluate the performance of planners (number of generated samples, collision check callings, number of nodes in the graph solution, connected components) and simulation tools.

The following two examples illustrate the proposed optimization method for the robot coordination using temporal synchronization. Each robot path was independently computed off-line, by using a motion planner based on a PRM [15]. The optimization was done allowing two collision checks during each robot step movement (i.e. $\mathrm{N_{cc}} = 2$).

Fig. 5a shows the setups for the experiment 1. The robot R_1 is in charge of removing the yellow can O_4, and R_2 is in charge of removing the red can O_3. Fig. 6a shows a collision configuration when the two robots execute their paths without any coordination. Fig. 7 shows the obtained FCC, for the coordination without optimization (top) and with optimization (bottom). Note that when the optimization method is used

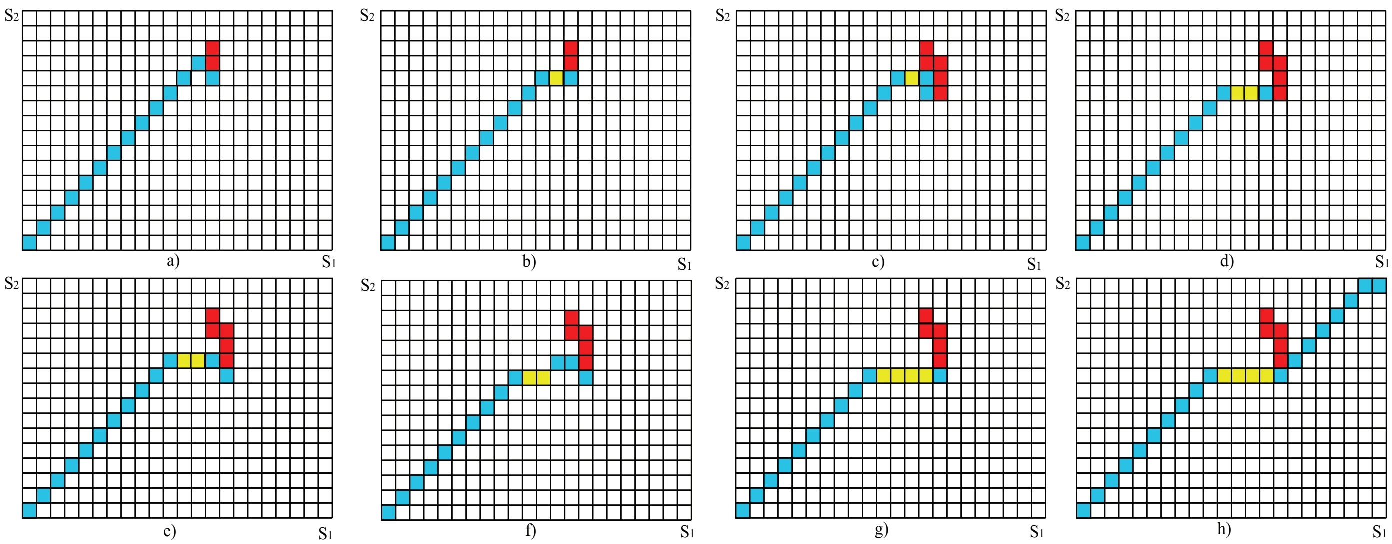

Fig. 4: Example of the optimization process of an FCC in a 2-dimensional DCS. a) An obstacle was found in DCS and one robot moves backward (last blue point). b) After the optimization of FCC the backward move was removed (the yellow point was included in FCC). c) A new backward move was found. d) Optimized FCC. e) Again, a new backward move was found. f) FCC partially optimized. g) FCC fully optimized. h) Complete optimized FCC.

the curve FCC does not surround exactly the obstacle and minimizes the backward movements of the robots (the blue points of the final FCC are not always together to the detected collision points of CR shown in red). There are still some backward movements in the optimized FCC, this is due to the fact that the robots arrive to the points where the optimization is being performed before the FCC was fully optimized. In order to have a general vision of the problem, a collision check was run for all the points in DCS and Fig. 8 shows the complete collision regions CR in DCS together with the optimized FCC. Fig. 9 shows the number of points in the Explored Window EW (in red without optimization and in blue with optimization). Note that EW is smaller when the optimization os applied, because no new points are added to FCC during the optimization process. Finally, Table I summarizes the results of the executions of experiment 1 with and without optimization (number of executed robot movements, execution time, number of calls to the collision check).

Fig. 10a shows the setups for the experiment 2. The robot R_1 is in charge of removing the yellow cans O_2 and O_4, and R_2 is in charge of removing the red cans O_1 and O_3. Fig. 11a shows a collision configuration when the two robots execute their paths without any coordination. Fig. 7 shows the obtained FCC, for the coordination without optimization (top) and with optimization (bottom). In this case the optimization is executed in two parts of FCC and the effect in each one is the same than in experiment 1. Again, not all the backward movements were removed from FCC because the robots arrives to the points where the optimization is being performed before the FCC was fully optimized. For illustrative purpose, the collision check was run again for all the points in DCS and Fig. 13 shows the complete collision regions CR in DCS together with the optimized FCC. Fig. 14 shows the number of points in the Explored Window EW (in red without optimization and in blue with optimization). Note that

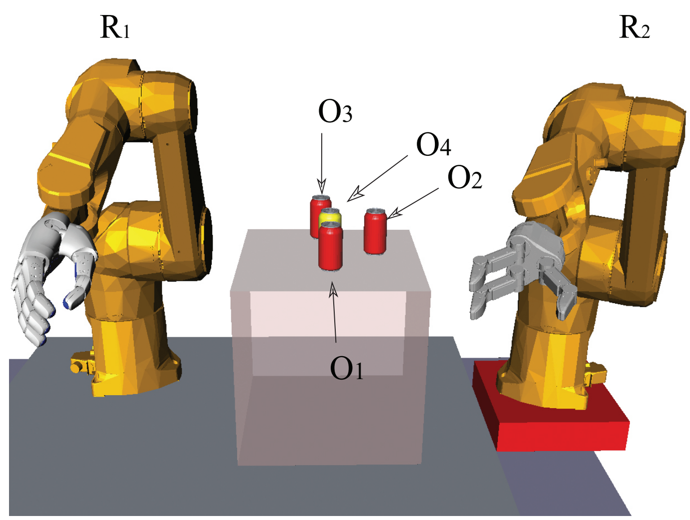

Fig. 5: Setup of the workcell for Experiment 1.

EW is smaller when the optimization os applied, because no new points are added to FCC during the optimization process. Finally, Table II summarizes the results of the executions of experiment 2 with and without optimization (number of executed robot movements, execution time, number of calls to the collision check).

VI. Summary

This work presents an optimization method applied to robot coordination with temporal synchronization. The goal of the proposed method is to minimize the backward movements of the robots in order to keep them away from potential collisions with other robots and avoid an unnecessary use of energy and

Algorithm 1: Main

input : $\text{path}_i, i = 1, 2$

FCC $\leftarrow \emptyset$, $P_k \leftarrow O$
MakeOpt = false
while *Task is not finished* **do**
    $i = 0$
    while $i \leq N_{cc}$ **do**
        if *MakeOpt = false* **then**
            Determine P_{k+1} using MD$_k$
            if $P_{k+1} \neq P_{\text{goal}}$ **then**
                if P_{k+1} *does not imply collision* **then**
                    Add P_{k+1} to FCC
                    $P_k \leftarrow P_{k+1}$
                    if P_{k+1} *means a backward movement* **then**
                        MakeOpt = true
                else
                    Select a new MD$_k$
            $i = i + 1$
        if *MakeOpt = true* **then**
            generate OPT using P_k and the portion of FCC already computed
            P_{inter} = FCC $\cap$ OPT
            if *the robots have not reached* P_{inter} **then**
                P_{opt} = point after P_{inter} in OPT
                P_{check} = point after P_{inter} in FCC
                if P_{opt} *does not imply collision* **then**
                    replace P_{check} by P_{opt} in FCC
                if *no backward movement* **then**
                    MakeOpt= false
                $i = i + 1$
            else
                MakeOpt= false
    Move the robots from its current positions to the next ones according to FCC

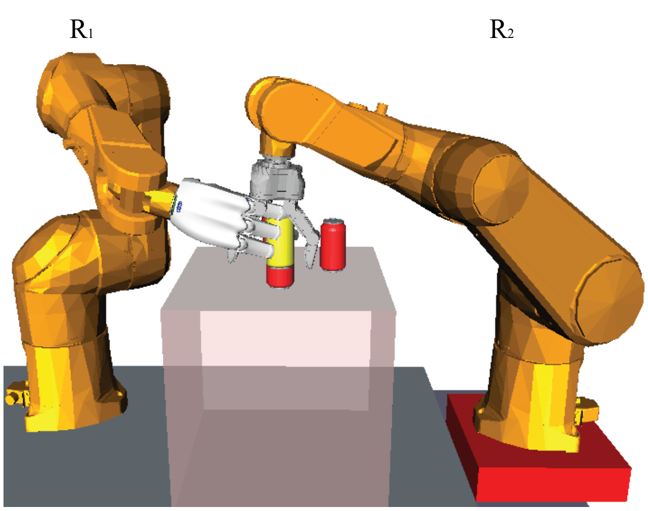

Fig. 6: Collision configuration when the two robots execute their paths without coordination in experiment 1.

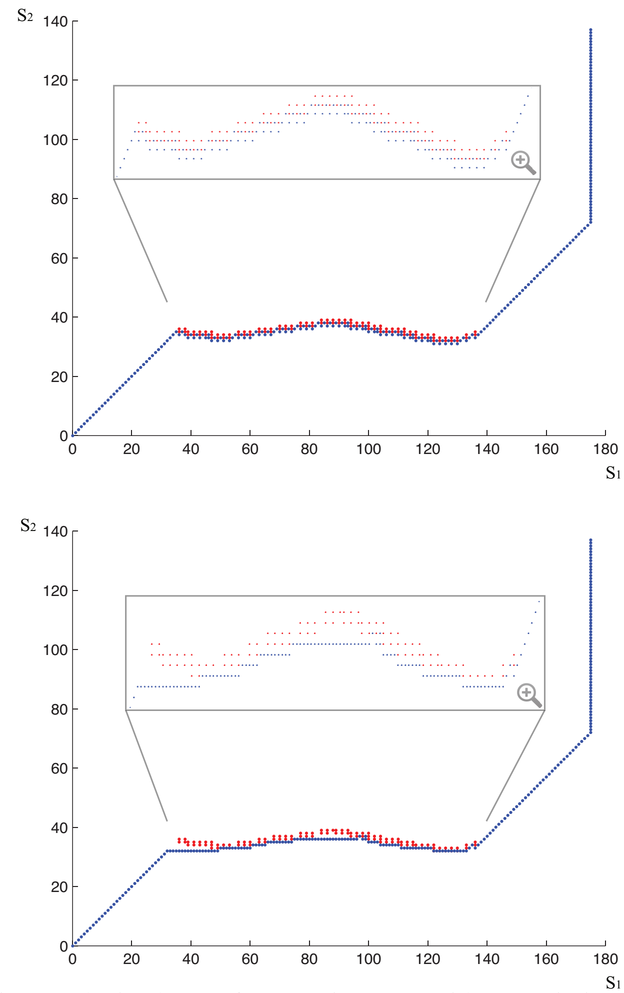

Fig. 7: Obtained FCC for experiment 1, without optimization (top) and with optimization (bottom). The square into the figure shows a zoom of the optimized part of FCC.

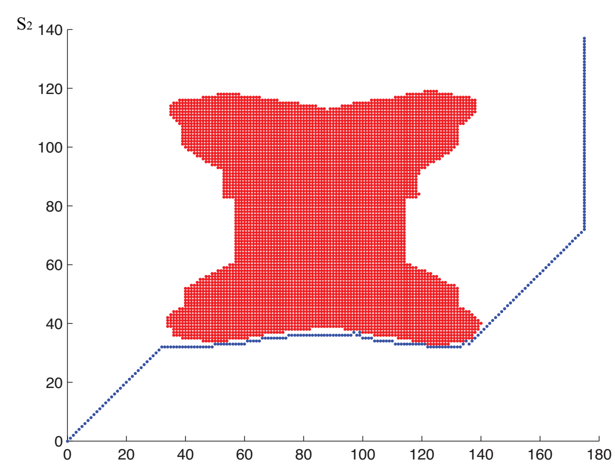

Fig. 8: DCS with the complete collision region CR and the optimized FCC for experiment (left) and for experiment 1.

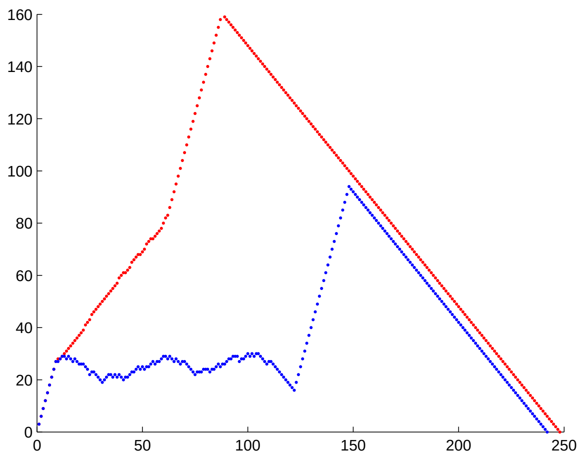

Fig. 9: Number of points in EW for experiment 1, without optimization (red) and with optimization (blue).

	Experiment 1	
	Without optimization	With optimization
Movements	248	242
Exploration time	88	148
Collision check	351	348 + 257

TABLE I: Result of the execution of the first experiment.

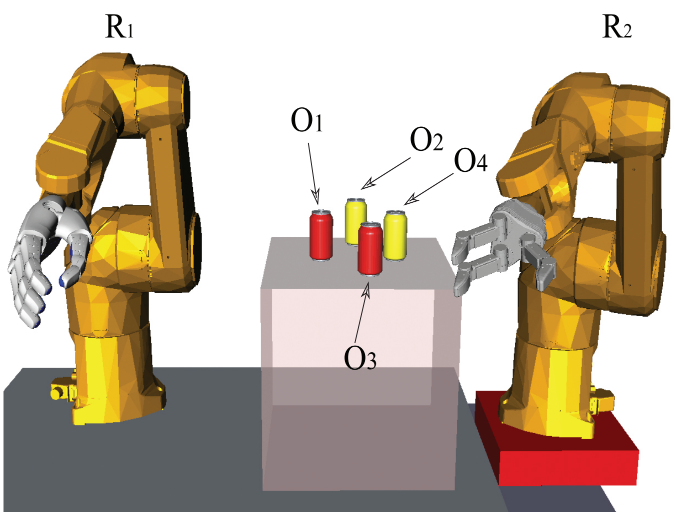

Fig. 10: Setup of the workcell for experiment 2.

	Experiment 2	
	Without optimization	With optimization
Movements	508	496
Exploration time	178	409
Collision check	712	715 + 930

TABLE II: Result of the execution of the second experiment.

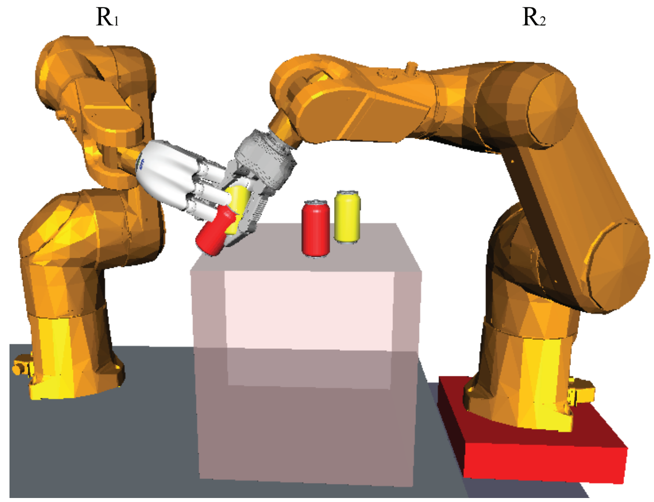

Fig. 11: Collision configuration when the two robots execute their paths without coordination in experiment 2.

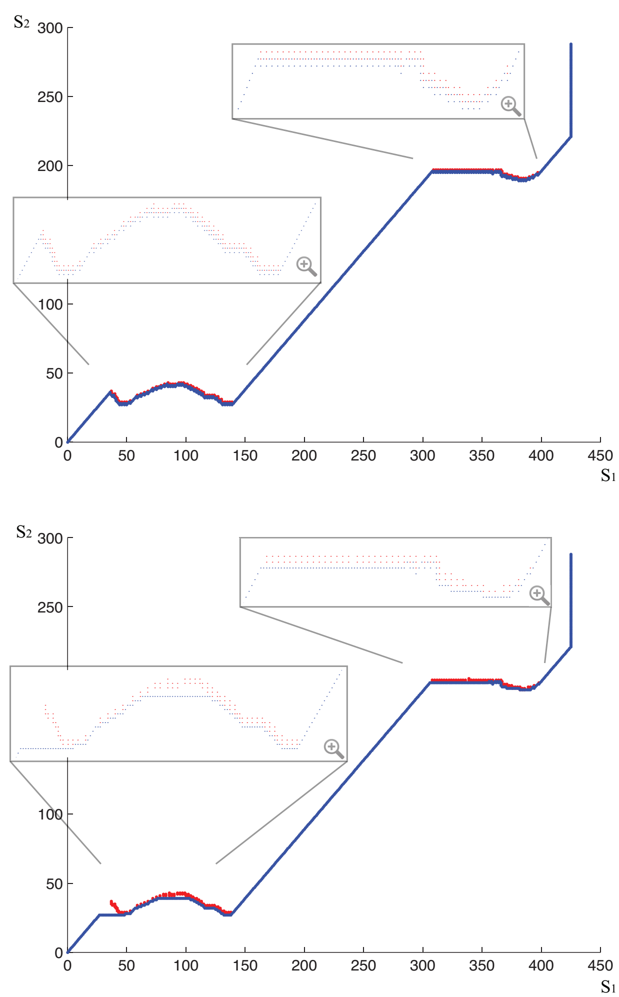

Fig. 12: Obtained FCC for experiment 2, without optimization (top) and with optimization (bottom). The squares into the figure show a zoom of the optimized parts of FCC.

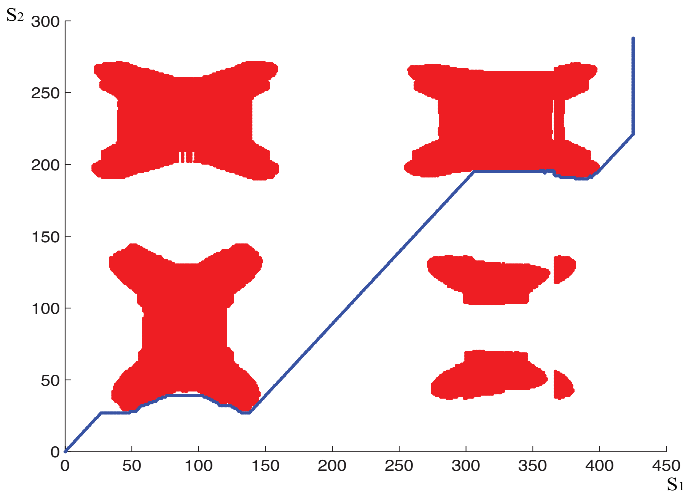

Fig. 13: DCS with the complete collision region CR and the optimized FCC for experiment (left) and for experiment 2.

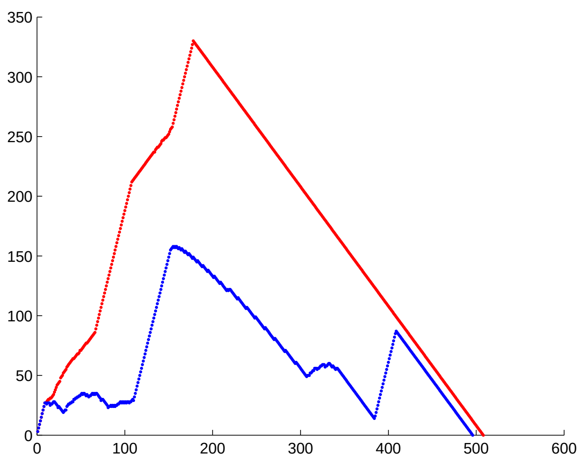

Fig. 14: Number of points in EW for experiment 2, without optimization (red) and with optimization (blue).

save time. Another advantage of the proposed optimization is that the robot paths became smoother, and this minimizes the robot vibrations that appear when they move back and forward just one (or a very small number) of steps in their paths, case that appears very often when using a standard temporal coordination without optimization.

References

[1] J.-C. Latombe, *Robot motion planning*. Kluwer Academic Publishers, 1991.

[2] E. Todt, G. Rausch, and R. Suárez, "Analysis and classification of multiple robot coordination methods," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 4, 2000, pp. 3158–3163.

[3] G. Sanchez and J.-C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 2, 2002, pp. 2112 – 2119.

[4] P. O'Donnell and T. Lozano-Peréz, "Deadlock-free and collision-free coordination of two robot manipulators," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, 1989, pp. 484 – 489.

[5] J. Lee, H. S. Nam, and J. Lyou, "A practical collision-free trajectory planning for two robot systems," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 3, 1995, pp. 2439–2444.

[6] A. Mohri, M. Yamamoto, and S. Marushima, "Collision-free trajectory planning for two manipulators using virtual coordination space," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 2, 1993, pp. 674 –679.

[7] X. Cheng, "On-line collision-free path planning for service and assembly tasks by a two-arm robot," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 2, 1995, pp. 1523 –1528.

[8] S. Lee, H. Moradi, and C. Yi, "A real-time dual-arm collision avoidance algorithm for assembly," in *Proc. IEEE Int. Symp. Assembly and Task Planning*, aug 1997, pp. 7–12.

[9] S. S. Chiddarwar and N. Ramesh Babu, "Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence," *Robotics and Autonomous Systems*, vol. 59, no. 7-8, pp. 508–518, 2011.

[10] A. Montaño and R. Suárez, "An On-Line Coordination Algorithm for Multi-Robot Systems," in *18th Proc. IEEE Int. Conf. Emerging Technologies and Factory Automation, ETFA*, September 2013.

[11] Y. Shin and Z. Bien, "Collisionfree trajectory planning for two robot arms," *Robotica*, vol. 7, pp. 205–212, 6 1989.

[12] Schunk, "SCHUNK GmbH & Co. KG – Shunck Dexterous Hand - SDH2," Site: www.schunk.com, Sep 2011.

[13] A. Pérez and J. Rosell, "A Roadmap to Robot Motion Planning Software Development," *Computer Applications in Engineering Education*, vol. 18, pp. 651–660, September 2010.

[14] J. Rosell, M. Roa, A. Prez, and F. Garca, "A general deterministic sequence for sampling d-dimensional configuration spaces," *J. Intelligent and Robotic Systems*, vol. 50, no. 4, pp. 361–373, 2007.

[15] C. Rodríguez, A. Montaño, and R. Suárez, "Manipulation tasks with a dual arm system including obstacles removing," in *18th Proc. IEEE Int. Conf. Emerging Technologies and Factory Automation, ETFA*, September 2013.