Robot Path Planning over a Sequence of Points Specified in Task Space

Balázs Németh[1,2], **András Kovács**[1]
[1] Institute for Computer Science and Control (SZTAKI), Budapest, Hungary
[2] Trinity College, University of Cambridge, UK
bn273@cam.ac.uk, andras.kovacs@sztaki.hu

Abstract

This paper addresses path planning for articulated industrial robots over a sequence of points defined in the Cartesian task space, which is a common problem faced in assembly, manufacturing, and material handling applications. The particular challenge lies in the fact that for typical 6 degrees-of-freedom industrial robots, the inverse kinematic problem admits various solution branches, i.e., multiple candidate robot configurations for the same task-space point. Therefore, this problem is a combination of a discrete optimization problem (selecting the configurations) and a path planning problem in the robot joint configuration space (finding a collision-free path). Earlier methods address this problem by solving the two sub-problems sequentially, which may easily lead to sub-optimal solutions. This paper introduces a Benders decomposition approach to solve the two sub-problems jointly, by adapting standard techniques on each individual level: dynamic programming for the selection of configurations in the master problem, and probabilistic roadmaps for path planning in the sub-problem. The efficiency of the approach is demonstrated in a medium-size industrial case study.

1 Introduction

A key challenge in the planning of robotic assembly, manufacturing, or handling processes is that tasks specified in the task space, i.e., the Cartesian coordinate system attached to the workpiece or the workcell, have to be transformed into motion in the robot's joint configuration space. Since a robot end effector pose in the task space (task point) can be reached by various robot configurations, planning also involves the discrete optimization problem of selecting the most efficient combination of robot configurations for each task point. Obviously, this optimization problem is tightly related to the problem of robot path planning, which needs to be solved in the joint configuration space. Finding robot configurations and collision-free paths that minimise the cycle times is the key to maximising productivity.

In most previous approaches, the two levels are handled sequentially: either (1) the configurations are determined first using some heuristics, then collision-free paths are planned; or (2) the collision-free paths are pre-computed between every possible combination of configurations, and the configurations are selected based on the computed path lengths to minimize the cycle time (Ulrich et al. 2016; Villumsen and Kristiansen 2017). However, the first approach may lead to overly long paths in dense environments due to the big difference between the collision-free and heuristic path lengths, whereas the second approach may incur excessive computation times. Previous research addressed responding this challenge by integrating inverse kinematics into the path planning algorithm and using heuristic goal and obstacle distance measures to select candidate robot configurations (Bertram et al. 2006).

This paper presents an algorithm using Benders decomposition (Rahmaniani et al. 2017) which, given a sequence of points in task space, computes an efficient path over these points. Experimental results show that this approach effectively reduces cycle time in a reasonable amount of computational time even in crowded and difficult environments.

The rest of the paper is organised as follows. Section 2 gives a rigorous mathematical formulation of the problem. Section 3 presents how Benders decomposition can be applied to this problem. Section 4 contains the results of computational experiments in comparison with other approaches. Finally, in Section 5, conclusions are drawn.

2 Problem Definition

Let $\mathbf{x_0}, \mathbf{x_1}, \cdots, \mathbf{x_n}$ be robot poses, with fully defined positions and orientations of the robot's end effector in the Cartesian task space that should be visited in this particular order. It is assumed that the robot performs a cycle, i.e., $\mathbf{x_0} = \mathbf{x_n}$, although this assumption can be easily lifted. The joint configuration space of the robot is denoted by $\mathcal{C}$ and its subset of non-colliding configurations by $\mathcal{C}_{free}$. Each point $\mathbf{x_i}$ in the task space can be reached by finitely many ($k_i > 0$) robot joint configurations $\mathbf{q}_{i,1}, \cdots, \mathbf{q}_{i,k_i}$.

$\mathcal{C}$ is endowed with a metric $t(\mathbf{u}, \mathbf{v})$ which measures the time it takes for the robot to move from configuration $\mathbf{u}$ to $\mathbf{v}$ on the straight segment in the joint configuration space $\overline{\mathbf{uv}} \subset \mathcal{C}$, disregarding potential collisions. It is assumed that t is monotonous in the length of the movement of each individual joint. Experiments were performed using trapezoid joint speed profiles, corresponding to constant joint accelerations at the beginning and end of the motion, and a constant joint velocity in between.

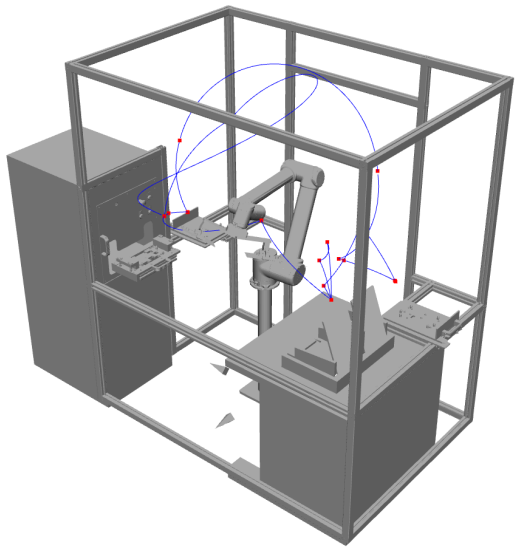

Figure 1: Robotic workcell of the working example, with a robot end effector path in blue. Each segment of the path is linear in the joint configuration space.

The problem then consists in finding a minimum time collision-free robot path that visits the above poses in the given order, i.e., a function f and a sequence of configurations $\{\mathbf{s}_i\}_{i=0}^{m}$ in $\mathcal{C}_{free}$ such that:

$$f : \{0, \cdots, n\} \to \mathbb{N}, \quad 1 \leq f(i) \leq k_i$$

The sequence $\{\mathbf{q}_{i,f(i)}\}_{i=0}^{n}$ is a sub-sequence of $\{\mathbf{s}_i\}_{i=0}^{m}$ with $\mathbf{s}_0 = \mathbf{q}_{0,f(0)}$, $\mathbf{s}_m = \mathbf{q}_{n,f(n)}$ and for each $i \in [1, m]$ the segment $\overline{\mathbf{s}_{i-1}\mathbf{s}_i}$ lies entirely in $\mathcal{C}_{free}$.

$$T = \sum_{i=1}^{m} t(\mathbf{s}_{i-1}, \mathbf{s}_i) \text{ is minimal}$$

2.1 Working Example

The working example of this paper consists of a real industrial workcell and a fixed-base UR10 articulated robot with 6 rotational degrees of freedom. Seven points of the task space must be visited in the given order, and three or four collision-free configurations could be obtained for each of them using an analytical inverse kinematics solver. Collisions may occur not only between the end effector and the workcell, but between any two robot links or a robot link and the workcell as well. A particular challenge stems from the fact that task points are located in hardly accessible regions all around the workcell. Accordingly, connecting the task points to other robot configurations with collision-free segments required sophisticated sampling and local planning in the probabilistic path planner applied, and subsequent task points could not be connected in a trivial way.

3 Benders Decomposition Solution Approach

3.1 Overview

From now on, a problem instance is considered where $\mathbf{x}_0 = \mathbf{x}_n$, i.e., a cycle is looked for.

The separation of the two levels of the problem is not difficult: if the configurations are chosen, a multi-query path planner can be called to compute a collision-free path through those points, details are in section 3.3. If distances between all possible pairs of robot configurations are given, configurations can be chosen using a simple dynamic programming algorithm as described in Section 3.2.

In Benders terminology, the choice of configurations is called the *master problem*, while the *sub-problem* addresses planning a collision-free path through the fixed configurations. In one iteration, the master problem is solved first. To be able to accomplish this, the distances are needed between configurations which are either heuristic estimates $t(\mathbf{q}_{i,f(i)}, \mathbf{q}_{i+1,f(i+1)})$ (taking a straight line segment regardless of collisions) or previously computed collision-free path lengths (every such path is calculated only once). For this choice of configurations let us denote the estimated path time (which is only an approximate value as there might be collisions along the way) by T^*. Observe that, as a consequence of the triangle inequality, T^* is a lower bound on the optimal path since every other path is longer even if they collide and by making them collision-free they cannot get any shorter.

This lower bound is called an *optimality cut*. At the end of the iteration, the sub-problem is solved, edge weights are updated, and unless a path is found whose cycle time is equal to the cost of master solution, the procedure is continued. When equality is reached the optimal path has been found. Observe that with a probabilistically complete and optimal path planner for solving the sub-problem, the overall approach is probabilistically complete and optimal as well.

3.2 Master Problem: Selection of Robot Configurations

Let $G = (V, E)$ be a weighted digraph with vertices $\mathbf{q}_{i,j}$ (robot joint configurations) and two vertices $\mathbf{q}_{i_1,j_1}$ and $\mathbf{q}_{i_2,j_2}$ connected by a directed edge if and only if $i_2 - i_1 = 1$. The weight of the edge $(\mathbf{q}_{i,j}, \mathbf{q}_{i+1,k})$ is denoted by $w_{i,j,k}$.

The objective is to find a function g such that:

$$g : \{0, \cdots, n\} \to \mathbb{N}, \quad 1 \leq g(i) \leq k_i, \quad g(0) = g(n)$$

$$W = \sum_{i=0}^{n-1} w_{i,g(i),g(i+1)} \text{ is minimal}$$

This problem can be solved by dynamic programming as follows. Let $d_{i,j,k}$ be the length of the shortest path starting at $\mathbf{q}_{0,k}$ and ending at $\mathbf{q}_{i,j}$. Observe that $d_{i,j,k}$ satisfies the following recurrence relation:

$$d_{i,j,k} = \begin{cases} 0, & \text{if } i = 0, j = k \\ \infty, & \text{if } i = 0, j \neq k \\ \min_{1 \leq m \leq k_{i-1}} (d_{i-1,m,k} + w_{i-1,m,j}), & \text{if } i > 0 \end{cases}$$

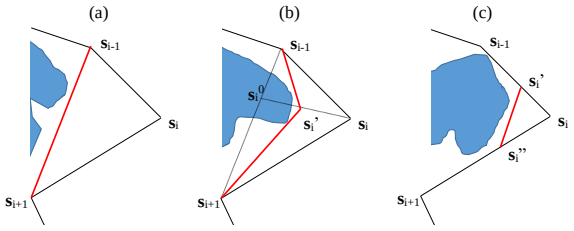

Figure 2: Smoothing by deleting (a), shifting (b), and adding (c) intermediate path points.

After calculating all intermediate values the final result is simply:

$$W = \min_{j \in \{1,\cdots,k_0\}} d_{n,j,j}$$

The exact path can be traced back by maintaining a parent array $p_{i,j,k}$ which keeps track of the minimal index m during computation. If every k_i can be bounded from above by k then the running time of the master-problem solver is $O(nk^2)$.

3.3 Sub-problem: Robot Path Planning

For a fully instantiated choice function f the sub-problem is simply finding collision-free paths between $\mathbf{q}_{i-1,f(i-1)}$ and $\mathbf{q}_{i,f(i)}$. This is done using a standard probabilistic roadmap (PRM) planner as described in (Kavraki et al. 1996) with minor differences compared to the original paper.

First of all, we found it useful to construct a dense graph rather than a forest, as not only an arbitrary collision-free path is looked for but one with length as short as possible. Similarly, we favoured Dijkstra's algorithm over a simple breadth-first search as it gives the shortest path between two vertices in the graph. In addition, the sampling of PRM nodes was enhanced by choosing some configurations closer to obstacles and narrow passages governed by a Gaussian distribution as described in (Boor, Overmars, and van der Stappen 1999) and (Hsu et al. 2003).

Moreover, paths returned by PRM queries can be significantly improved using heuristic path smoothing algorithms which are not discussed in detail in the literature. Three such algorithms were employed (see Fig. 2):

- **Smoothing by deletion:** The simplest and perhaps the most effective out of the three, this path smoother tries to shortcut the path by deleting redundant path points. Given a path $\{\mathbf{s}_i\}_{i=0}^{m}$ entirely in $\mathcal{C}_{free}$, it looks for a minimum-length sub-sequence $\{\mathbf{s}'_j\}_{j=0}^{\ell}$ in $\mathcal{C}_{free}$ with identical endpoints. To find this sub-sequence, it constructs a weighted graph $G = (V, E)$ with vertices $\{\mathbf{s}_i\}_{i=1}^{m}$. An edge goes from $\mathbf{s}_i$ to $\mathbf{s}_j$ $t(\mathbf{s}_i, \mathbf{s}_j)$ iff $i < j$ and $\overline{\mathbf{s}_i\mathbf{s}_j} \subset \mathcal{C}_{free}$. Observe that the problem is finding a shortest path in the directed acyclic graph G, which has a well-known dynamic programming solution.
- **Smoothing by shifting:** When an intermediate node cannot be deleted, this path smoother tries to lower path cost by reducing the detour made to avoid the obstacle. For this purpose, it takes each intermediate path point $\mathbf{s}_i$ and attempts to replace it with a point $\mathbf{s}'_i$ selected from the section $\overline{\mathbf{s}_i\mathbf{s}_i^0}$, where $\mathbf{s}_i^0$ divides section $\overline{\mathbf{s}_{i-1}\mathbf{s}_{i+1}}$ according to $\frac{|\overline{\mathbf{s}_{i-1}\mathbf{s}_i^0}|}{|\overline{\mathbf{s}_i^0\mathbf{s}_{i+1}}|} = \frac{|\overline{\mathbf{s}_{i-1}\mathbf{s}_i}|}{|\overline{\mathbf{s}_i\mathbf{s}_{i+1}}|}$. The heuristic tries a pre-determined number of points on the section $\overline{\mathbf{s}_i\mathbf{s}_i^0}$, and selects the one leading to collision-free path sections closest to $\mathbf{s}_i^0$. Since metric t is monotonous, the shifted path is shorter than the original one.
- **Smoothing by adding:** This algorithm aims to add appropriate points inside the intermediate segments to cut off sharp corners of the path. It checks consecutive configurations $\mathbf{s}_{i-1}$, $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$ of the path and looks for $\mathbf{s}'_i \in \overline{\mathbf{s}_{i-1}\mathbf{s}_i}$ and $\mathbf{s}''_i \in \overline{\mathbf{s}_i\mathbf{s}_{i+1}}$ such that $\overline{\mathbf{s}'_i\mathbf{s}''_i} \subset \mathcal{C}_{free}$ and $\overline{\mathbf{s}_{i-1}\mathbf{s}'_i\mathbf{s}''_i\mathbf{s}_{i+1}}$ is shorter than $\overline{\mathbf{s}_{i-1}\mathbf{s}_i\mathbf{s}_{i+1}}$. Note that $\overline{\mathbf{s}_{i-1}\mathbf{s}'_i}$ and $\overline{\mathbf{s}''_i\mathbf{s}_{i+1}}$ are automatically collision-free.

4 Computational Experiments

Three different approaches to solving the problem were compared using the collision detection and path planning library presented in (Zahorán and Kovács 2019), on 12 different sequences (including the real industrial point sequence and 11 random sequences) in the robotic cell presented in Section 2.1. In a pre-processing step, a closed-form inverse kinematics solver was run to obtain joint configurations, and a PRM was built for the workcell with about 30000 vertices, which took 43 minutes on a personal computer with Intel i5 1.80 GHz CPU and 16 GB RAM. Since robotic process planning is an iterative procedure by nature, the time of building the PRM was not considered as a part of the planning time.The approaches were as follows:

- **Sequential:** The baseline approach in industry first determines the robot configurations based on estimated distances, and then calculates the collision-free path between them. It corresponds to the optimality cut calculated in the first iteration of the Benders approach.
- **Pre-compute:** This method first computes all collision-free paths between configurations, and then runs the algorithm explained in Section 3.2 to find the shortest cycle.
- **Benders:** The approach proposed in this paper.

Table 1 summarizes the results of the computational experiments. Column *Obj* contains the cycle time obtained using the given algorithm, while *Impr* shows the improvement compared to the baseline Sequential algorithm. Column *Time* refers to the average computation time, while column *Iter* shows the number of iterations performed (what an iteration means is described in Section 3). Finally, column *PRM* gives the amount of calls made to the path planner.

The results are quite satisfactory: even though the Sequential approach leads to good paths, it could be improved by up to 16.03%, on average 4.65% by the optimal planners. Moreover, for the actual industrial sequence, the optimal path was 14.73% faster than the path designed by an experienced engineer with the help of simulation software, and 10.85% faster than the Sequential path, which is extremely valuable

	Obj [s]	Impr [%]		Time [s]	Iter	PRM
	Avg.	Avg.	Max.	Avg.	Avg.	Avg.
Sequential	10.8	-	-	19.6	1	7
Pre-compute	10.3	4.65%	16.03%	325.8	1	69
Benders	10.3	4.65%	16.03%	89.2	7.5	24

Table 1: Results of the computational experiments

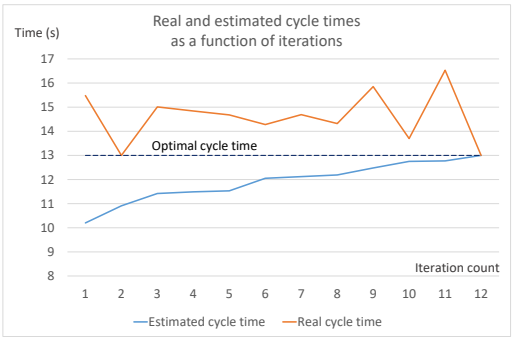

Figure 3: Evolution of estimated cycle time (solution of the master problem) and real cycle time (solution after collision-free path planning).

in the given application. Observe that Benders had to compute collision-free paths for only 34.78% of the edges, thus the computation time could be reduced by 72.62% compared to Pre-compute. We expect that this gain will increase further on larger problems or in less crowded environments. It is noted that the 97.8% of the planning time is taken by the smoothing operators, since they have to perform computation on the actual workcell geometry. Moreover, the proposed Benders approach can be interrupted with a valid solution when hitting a time limit. The feasibility and the cycle time of the computed paths was verified in experiments in the real robotic cell as well.

5 Conclusions

This paper presented an algorithm based on Benders decomposition that computes an efficient path in robot joint configuration space over a sequence of points specified in task space. For this purpose, it selects robot configurations for each point and computes a collision-free path through these configurations iteratively, until an optimum (in Benders context) is reached. The efficiency of the approach was demonstrated in an industrial case study, where it produced paths with up to 16.03%, on average 4.65% lower cycle times than the commonly applied sequential algorithm, and did it in less than a third of the time that is required to pre-compute all collision-free edges, even in a rather dense workcell.

However, we are aware of the fact that there are obvious limitations to the proposed solutions. First of all, we have worked under the assumption that the order of points in the task space is given. In many environments this is not true, hence the problem of sequencing must be tackled as well. It would be intriguing to see if this extension could be handled at the same time, just as it was done in our case. Moreover, robot joint intervals larger than 2π induce a rotational symmetry that could be exploited to increase the efficiency of the proposed approach. Finally, the extension of the method to smooth, e.g., spline trajectories would be relevant in many industrial applications.

Acknowledgments

The authors are grateful to László Zahorán for granting access to the robot collision library used in this research. This work has been supported by grants 2018-2.1.7-UK_GYAK-2019-00002 and GINOP-2.3.2-15-2016-00002. A. Kovács acknowledges the support of the János Bolyai Research Fellowship.

References

Bertram, D.; Kuffner, J.; Dillmann, R.; and Asfour, T. 2006. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *Proc. ICRA 2006, IEEE International Conference on Robotics & Automation*, 1874–1879.

Boor, V.; Overmars, M. H.; and van der Stappen, A. F. 1999. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proc. ICRA 1999, IEEE International Conference on Robotics and Automation*, 1008–1023.

Hsu, D.; Jiang, T.; Reif, J.; and Sun, Z. 2003. The bridge test for sampling narrow passages with probabilistic roadmap planners. In *Proc. ICRA 2003, IEEE International Conference on Robotics & Automation*.

Kavraki, L. E.; Svestka, P.; Latombe, J. C.; and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4):566–580.

Rahmaniani, R.; Crainic, T. G.; Gendreau, M.; and Rei, W. 2017. The benders decomposition algorithm: A literature review. *European Journal of Operational Research* 259(3):801–817.

Ulrich, M.; Lux, G.; Jürgensen, L.; and Reinhart, G. 2016. Automated and cycle time optimized path planning for robot-based inspection systems. *Procedia CIRP* 44:377–382.

Villumsen, S. L., and Kristiansen, M. 2017. A framework for task sequencing for redundant robotic remote laser processing equipment based on redundancy space sampling. *Procedia Manufacturing* 11:1826–1836.

Zahorán, L., and Kovács, A. 2019. Efficient collision detection for path planning for industrial robots. In *Proc. StuCoSReC 2019, 6th Student Computer Science Research Conference*.