To appear in the *International Journal of Production Research*
Vol. 00, No. 00, 00 Month 20XX, 1–17

Integrated Task Sequencing and Path Planning for Robotic Remote Laser Welding

András Kovács[a*]

[a]*Fraunhofer Project Center for Production Management and Informatics, Institute for Computer Science and Control, Hungarian Academy of Sciences*;

(*Received 00 Month 20XX; final version received 00 Month 20XX*)

This paper investigates the problem of integrated task sequencing and path planning in Remote Laser Welding (RLW). It is shown that finding the appropriate order of welding tasks is crucial for exploiting the efficiency of this new joining technology, and this can be achieved only if the robot path is considered already at the time of sequencing. For modelling the problem, a novel extension of the well-know Travelling Salesman Problem with neighbourhoods and durative visits, denoted as TSP-ND, is introduced. Basic properties of this problem are formally proven, and a GRASP meta-heuristic algorithm is proposed for solving it. Extensive computational experiments demonstrate that the novel approach solves efficiently industrially relevant problems, and it achieves substantial improvement in cycle time compared to the single earlier approach in the literature dedicated to RLW, as well as compared to a decomposition approach to solving the TSP-ND model.

Keywords: sequencing; path planning; travelling salesman problems; remote laser welding; manufacturing

1. Introduction

The recent development of a new generation of laser sources, such as Ytterbium fiber lasers, enabled laser welding with an operating distance (focal length) above one metre, using a laser scanner mounted on an industrial robot. The rotating mirrors in the scanner ensure extremely fast positioning of the laser beam even between distant points on the workpiece. Hence, the emerging technology, *Remote Laser Welding* (RLW) is extremely productive; a single RLW robot may replace up to 5 classical Resistance Spot Welding (RSW) robots (Buehrle, Bea, and Brockmann 2013). RLW also incurs a significantly lower cost-per-joint than RSW. In addition to the economic gain, RLW eliminates the most important limitation of earlier joining techniques, the accessibility issues between the welding gun and the workpiece. This, in turn, removes many earlier constraints on workpiece design, an advantage that can be turned easily into parts with reduced weight, yet higher stiffness (Park and Choi 2010).

Nevertheless, due to the high initial investment, the long-term profitability of RLW can be guaranteed only if the robot performs effective welding tasks continuously, with as little idle movement between them as possible. This requires efficient process planning and effective robot programming techniques. A crucial decision in process planning is sequencing the welding tasks. However, process planing, and specifically task sequencing algorithms tailored to the needs of RLW hardly exist (Reinhart, Munzert, and Vogl 2008).

Our general objective is developing an interactive off-line programming (OLP) toolbox with strong optimisation capabilities for RLW (Erdős et al. 2015). In this paper, we propose novel, effi-

*Corresponding author. Email: andras.kovacs@sztaki.mta.hu

cient methods for integrated task sequencing and path planning. A formal model is introduced that captures all important aspects of RLW, leading to a novel extension of the well-known Travelling Salesman Problem (TSP) with neighbourhoods and durative visits. It is shown that the straightforward decomposition approach to solving this problem may lead to arbitrarily poor solutions. Therefore, an efficient GRASP meta-heuristic algorithm is proposed for solving the two components of the problem in an integrated way. The approach is evaluated on real industrial data.

The adequacy of a model that ignores potential collisions for task sequencing stems from a common design guideline in RLW that the access volumes of the welding stitches must be left clear. Still, industrial data shows that this guideline is sometimes overridden by other design objectives, and hence, the calculated path might need to be adjusted to ensure a collision-free robot trajectory. Algorithms for collision-free path planning that adjust the path calculated by the techniques proposed below have been introduced in (Kovács 2014), and they are discussed in this paper, too.

This paper is partly based on earlier results published in the conference paper (Kovács 2013), however, it presents a new, formal model with a proof of its key characteristics, improved algorithms, and computational evaluation on real industrial data.

The paper is organised as follows. First, a brief review of the technological background and the related literature is given. Then, the task sequencing and path planning problem is defined (Section 2) and the formal characteristics of the proposed model are investigated (Section 3). The algorithms proposed for solving the problem are presented in Section 4. Computational experiments on industrial data are reported in Section 5. Finally, the application of the results in an OLP system is discussed (Section 6) and conclusions are drawn.

1.1 *Technological Background*

1.1.1 The Welding Process

An RLW operation consists in joining two or more sheet metal parts at various joints using a laser beam to transmit the required energy. In this paper, we assume stitch welds, i.e., disjoint linear or circular welding stitches with a length of 15-30 mm each. During the operation, the parts are held in a fixture, which is either stationary or attached to a rotating table. It is assumed that the operation is performed by a single RLW robot. A typical RLW robot consists of a robot arm with 4 rotational joints and a laser scanner. The robot arm moves the scanner with a maximum speed of 0.2-0.5 m/s, and due to the low scanner weight, with a rather high acceleration. The scanner contains two mirrors for the rapid positioning of the laser beam (up to 5 m/s), and a lens system to regulate the focal length. Hence, the robot is a redundant kinematic system with 7 degrees of freedom, in which the mirrors of the scanner move an order of magnitude faster than the mechanical joints of the robot arm. A typical RLW robot is depicted in Figure 1.

The robot can weld a stitch if the scanner is located within the *focus range* (e.g., 800-1200 mm) from the stitch, and the *inclination angle* (i.e., the angle between the laser beam and the surface normal) is not more than a specified technological parameter (e.g., $15 - 30^\circ$). Each stitch must be welded without interruption, at a pre-defined speed and laser power (e.g., 50 mm/s and 4kW), which depend on the thickness and the material of the parts to join. The robot can weld the stitch while in motion, therefore, the trajectory of the scanner must be a curve in the 3D space, such that sufficient time is spent over each stitch. Typically, there are 30-75 stitches to weld in an RLW operation in the automotive industry.

Using an industrial robotics analogy, the laser beam can be considered as the end effector of the robot. This allows us to define the *Tool Center Point* (TCP) as the end point of the beam, moving together with the robot and taking different positions over time on the surface of the workpiece. Furthermore, let us call the mid-point of the last, rotating mirror in the scanner head the *Scanner Center Point* (SCP). The SCP can be regarded as the origin of the laser beam, and the beam connects the SCP and the TCP in a straight line. In our work, we focus on planning the trajectory of the SCP, in contrast to earlier works that put emphasis directly on the trajectory of the TCP.

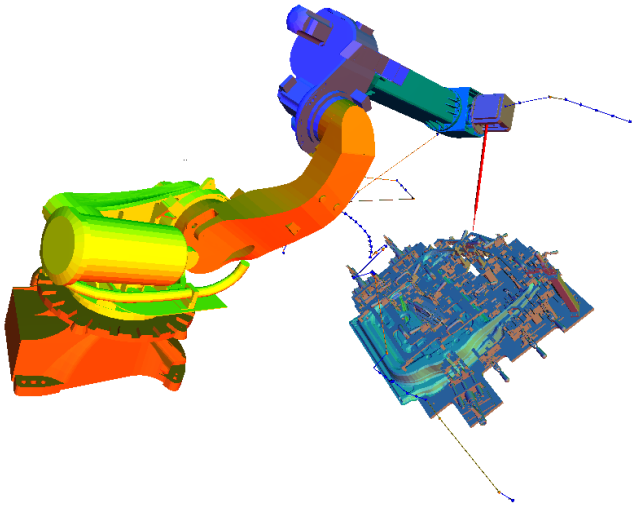

Figure 1. RLW robot welding a car front door, positioned in a fixture. The broken line indicates the path of the robot's scanner head, with blue sections standing for movement while welding and yellow sections for idle movement.

1.1.2 Off-line Programming for RLW

In industrial practice, robot programming is still mostly performed by on-line programming, i.e., by manually guiding the robot from one position to the next, at very small steps. This approach is rather time consuming, it does not allow effective optimisation, and hence, easily results in severely sub-optimal robot paths.

Our goal is to implement a complete OLP toolbox for RLW, which can provide an automated method for computing close-to-optimal robot programs. This involves, after the validation of the input by accessibility analysis, the optimisation of the task sequence; robot path planning; the placement of the workpiece in the robot working area; the inverse kinematic transformation that converts the path from the workpiece coordinate system to the robot joint coordinate system; and the simulation of the robot path, including collision detection. Finally, the robot program is generated in an automated way. This paper focuses on the first planning step, as displayed in Figure 2. The complete workflow is presented in detail in (Erdős et al. 2015, 2014).

1.2 *Literature Review*

1.2.1 Mathematical Methods for Task Sequencing

Problems involving the sequencing of a set of robotic tasks are naturally modelled as a Travelling Salesman Problem (TSP) or one of its numerous extensions. Certain manufacturing technologies, such as RLW, painting, or inspection, allow some flexibility in choosing the robot positions for each task. If the candidate positions of the effective tasks form a finite set, the obvious model becomes the Generalised TSP (GTSP), in which nodes are ordered into classes, and each class must be visited exactly once. Methods for transforming a GTSP into an ordinary TSP with the same number of nodes have been introduced in (Noon and Bean 1993) and in (Behzad and Modarres 2002). Helsgaun (2015) showed that the resulting TSP instances, despite their degenerate structure with extreme edge weights, can be solved efficiently by using the state-of-the-art TSP solver LKH, with appropriate parameter settings and post-processing heuristics. Exact and approximation algorithms for solving GTSPs have been proposed by Rice and Tsotras (2013). The Lin-Kernighan heuristic has been adapted to the GTSP, and compared with other GTSP heuristics in (Karapetyan and Gutin 2011).

The continuous space variant of GTSP is called TSP with neighbourhoods (TSPN), where the shortest tour that visits the given spatial regions is looked for. TSPN was originally introduced

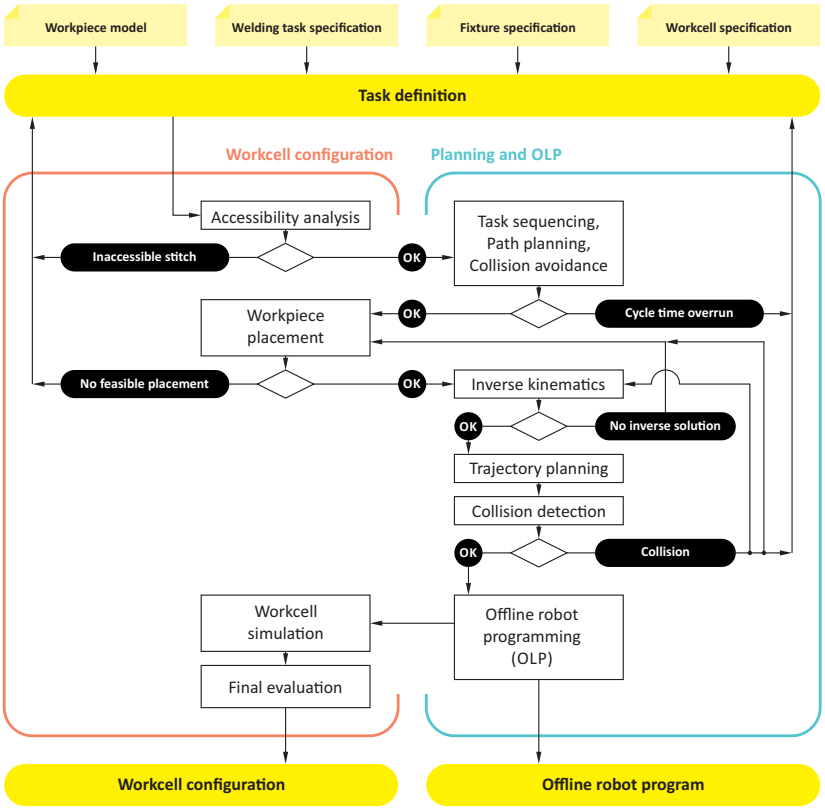

Figure 2. Workflow in the OLP system (Erdős et al. 2015).

by Arkin and Hassin (1994). The approximability of TSPN is investigated in (Safra and Schwartz 2006). Constant-factor approximation algorithms for TSPN over a set of disjoint, connected regions in the plane are proposed in (Elbassioni, Fishkin, and Sitters 2009). A mixed-integer non-linear programming approach to solving TSPN with polyhedral or ellipsoid regions in the 2D or 3D space is introduced in (Gentilini, Margot, and Shimada 2013).

With a pre-defined sequence, and the additional assumption that the regions to visit are 2D polygons, TSPN simplifies to the Touring (a sequence of) Polygons Problem (TPP). While TPP is NP-hard in general, it can be solved efficiently for convex polygons (Dror et al. 2003). TPP problems in robotics are classically addressed using the Rubberband Algorithm (RBA), which has been proven to achieve a constant approximation of the optimal solution for a number of TPP variants in (Pan, Li, and Klette 2010).

1.2.2 Task Sequencing and Path Planning in Manufacturing

We are aware of a single earlier approach to task sequencing and path planning specifically for RLW and remote laser cutting, introduced in a series of papers (Reinhart, Munzert, and Vogl 2008; Hatwig et al. 2012). The proposed algorithms are designed mostly for planar workpieces: task sequencing is performed by solving a TSP over the fixed welding stitch positions, and a robot path is computed in a 2D plane above the workpiece. A similar model is applied and heuristics are proposed for path planning in laser cutting in (Dewil, Vansteenwegen, and Cattrysse 2014; Dewil et al. 2015), with sophisticated ordering constraints among the contours to cut.

A generic task sequencing and collision-free path planning model, with illustrations from RSW is presented in (Saha et al. 2006). A critical assumption is that the robot can execute each effective task from a relatively small set of candidate configurations, e.g., at most 10 robot configurations per task, which can be generated a priori. An iterative algorithm is proposed that tries to compute as few point-to-point collision-free paths as possible, hence avoids solving unnecessary, computa-

tionally demanding subproblems. The difficulty in applying this approach to RLW stems from the fact that efficient paths in RLW exploit the free movement of the robot in the continuous space while welding.

The minimisation of processing time in a milling operation is investigated in (Castelino, D'Souza, and Wright 2002). A GTSP approach is proposed, where the nodes correspond to the candidate tool entry/exit points for machining a feature. The problem of 2D cutting path optimisation is modelled as a TSPN and solved using a two-step genetic algorithm in (Lee and Kwon 2006). The same problem in the context of nibbling on an NC turret punch press has been investigated by Veeramani and Kumar (1998), who proposed a heuristic that alternates between solving the two sub-problems related to sequencing and to pierce point determination. A similar TSPN model is proposed in (Alatartsev et al. 2013) for sequencing a set of robotic tasks whose start/end points can be chosen arbitrarily along open or closed contours. Task sequencing for a combined punching and laser cutting machine is addressed in (Wang and Xie 2005) using ant colony optimisation and in (Xie, Gan, and Wang 2009) using genetic algorithms. A multi-objective constraint optimisation model is proposed in (Kolakowska, Smith, and Kristiansen 2014) for task sequencing in spray painting, for minimising cycle time and maximising paint quality.

A recent survey on task sequencing in robotics has been presented in (Alatartsev, Stellmacher, and Ortmeier 2015). The integration of task planning and motion planning has received significant attention in the robotics community recently, with special attention to applications in navigation and manipulation. Various approaches combine symbolic planners as high-level solvers and motion planners as subproblem solvers, see, e.g., (Kaelbling and Lozano-Perez 2011; Srivastava et al. 2014).

2. Problem Definition

Briefly, the investigated process planning problem consists in sequencing the individual welding stitches and computing a corresponding SCP path, in such a way that the cycle time of the complete welding operation is minimised.

Formally, let there be given a set of n welding tasks, denoted by $\{s_1, s_2, ..., s_n\}$, to be executed by a single RLW robot in one operation. Each task corresponds to welding a single linear or circular stitch. Accordingly, with a slight abuse of the notation, the stitch corresponding to task s_i will also be denoted by s_i. Each task is characterised by its associated welding time, t_i, and its access volume, AV_i, the set of SCP positions in the 3D Cartesian space from where s_i can be welded.

The technological constraints on the maximum inclination angle and the focus range imply that AV_i is a truncated cone above stitch s_i, with its axis corresponding to the surface normal at the stitch, as shown in Figure 3. Strictly speaking, this definition would require spherical outer and inner bases for the truncated cone. However, to benefit from convex AVs, we approximate this shape by using a planar inner base, while leaving a spherical outer base. Since the length of a stitch is significantly smaller than other characteristic dimensions in the welding process, it is reasonable to assume that all points of a stitch can be processed from the AV belonging to the mid-point of the stitch.

The maximum robot speed (speed of the SCP) is denoted by v. The objective is *minimising the cycle time*, i.e., the time required for the robot to travel the computed path while welding the stitches. The following assumptions are made:

- The stitches must be welded by a single welding robot that is able to position the laser beam at a single stitch at a time.
- Each stitch must be welded without interruption.
- The robot has a sufficiently large working area that includes the complete AV of all stitches.
- The maximum SCP speed, v, is independent of the position in the working area.
- The robot has low inertia, and therefore, acceleration limits can be disregarded.
- The scanner can position the laser beam in zero time.
- Any stitch sequence is feasible.

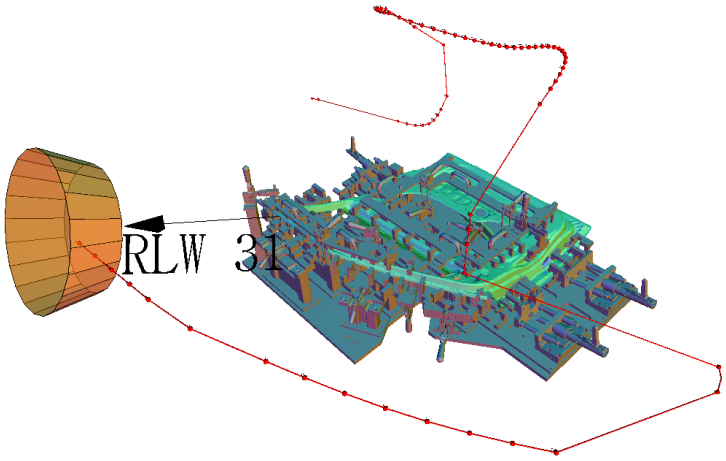

Figure 3. Access Volume (AV) of a welding stitch. The robot trajectory, denoted by a red broken line in this figure, must spend sufficient time in the AV of each stitch to weld that stitch.

- Potential collisions are disregarded at this phase of the planning hierarchy.

It should be observed that in areas where the time required for welding dominates the time required for travelling the path, a path with minimal cycle time may include unnecessary zigzags. Such a path may increase the energy consumed by the robot due to the greater path lengths and unnecessary curves, and ultimately, may be unacceptable for human experts. To avoid such paths, we add both the SCP path length and the TCP path length to the objective, with appropriately chosen low multipliers.

3. A TSP Model with Neighbourhoods and Durative Visits

3.1 *Model Definition*

In order to give a formal model for the integrated task sequencing and path planing problem, we define an extension of TSP called the *Travelling Salesman Problem with neighbourhood and Durative visits* (TSP-ND) as follows:

Definition 3.1 (TSP-ND)**.** *In an instance of TSP-ND, there are given n potentially overlapping, compact regions in the (2D or 3D) Euclidean space. A minimum-duration trajectory is sought that visits all regions in an arbitrary order, and spends t_i time visiting region i. Visits are non-interruptible and exclusive (i.e., two visits cannot overlap in time), but a trajectory may spend additional time inside regions, e.g., for traversing the region or for visiting other, overlapping regions. The maximum travel speed along the trajectory is a given constant v.*

Additionally, the location where the trajectory starts (ends) visiting region i is called the *start point* (*end point*) of visit i. These points will be denoted by a_i and b_i, respectively. It is assumed that a_i and b_i is located along the trajectory in such a way that the visit between them takes exactly t_i time. Accordingly, due to the potential additional time spent in the region, a_i and b_i may not coincide with the points where the trajectory first enters or at last leaves the region. With a given ordering of the visits, $[i]$ will be used to denote the index of the ith visit according to the given sequence.

Now it is easy to see that there exists a mapping from the integrated task sequencing and path planning problem for RLW to TSP-ND in the 3D space with convex regions as follows. Region i in TSP-ND corresponds to the AV of stitch i. The trajectory must spend t_i time visiting region i, and the maximum travel speed is v.

3.2 *Structure of the Optimal Solution*

In this section, we limit our attention to TSP-ND with convex regions. For such instances, the optimal trajectories can be characterised by the following two lemmas. Below, $\overline{ab}$ denotes a straight linear path between two points a and b, $\widetilde{ab}$ is an arbitrary curve between the same points, and $|\cdot|$ is the Euclidean distance norm.

Lemma 3.2. *Any TSP-ND instance with convex regions admits an optimal solution where, for each visit, the start and end points $a_{[i]}$ and $b_{[i]}$ are connected by a straight line segment $\overline{a_{[i]}b_{[i]}}$ (1). Moreover, in every optimal solution, the end point of a visit $b_{[i]}$ and the start point of the subsequent visit $a_{[i+1]}$ are connected by a straight line segment $\overline{b_{[i]}a_{[i+1]}}$ (2).*

Proof. Let there be given an optimal TSP-ND solution. Now, replacing for each region $[i]$ the curve $\widetilde{a_{[i]}b_{[i]}}$ by a straight line segment $\overline{a_{[i]}b_{[i]}}$ maintains the feasibility of the solution, since the regions are convex. Moreover, this modification does not increase the solution cost, from which (1) follows. Assume that in this solution, the end point of one visit $b_{[i]}$ and the start point of the subsequent visit $a_{[i+1]}$ are connected by a curve $\widetilde{b_{[i]}a_{[i+1]}}$ that differs from the straight line segment $\overline{b_{[i]}a_{[i+1]}}$. Then, replacing $\widetilde{b_{[i]}a_{[i+1]}}$ by $\overline{b_{[i]}a_{[i+1]}}$ in the solution strictly decreases the solution cost, which contradicts the optimality of the original solution, and hence proves statement (2). □

Consequently, the optimal trajectory for a TSP-ND with convex regions is a broken line path with $2n$ (potentially coinciding) breakpoints a_i and b_i.

Lemma 3.3. *In every optimal solution of a TSP-ND with convex regions, for each i, the start point $a_{[i]}$ is the nearest point to the end point $b_{[i-1]}$ in the intersection of region $[i]$ and the sphere around $b_{[i]}$ with radius $vt_{[i]}$. The symmetric statement holds for the location of end points $b_{[i]}$.*

Proof. By the definition of TSP-ND, $a_{[i]}$ must be located in region $[i]$, at most $vt_{[i]}$ far from point $b_{[i]}$. Among this set of feasible points, the location that minimises the solution cost is the point that minimises the distance $|\overline{b_{[i-1]}a_{[i]}}|$. □

3.3 *Performance of the Decomposition Approach*

The classical approach in manufacturing is solving the task sequencing and the path planning problems sequentially, one after the other. This decomposition approach involves solving the sequencing problem as a TSP over a graph with one vertex corresponding to an appropriately selected point in the 3D space for each stitch. The critical issue then is selecting the appropriate 3D points in the AVs, and in the lack of good heuristics, the straightforward solution is using the mid-point of the AV. Below we show that the performance of this approach can be arbitrarily bad.

Lemma 3.4. *The performance of the decomposition approach to solving TSP-ND using the mid-points of the neighbourhoods can be arbitrarily bad.*

Proof. We prove this lemma by giving an appropriate construction scheme, for the ease of understanding, in 2D. Let the TSP-ND instance contain $2n$ rectangular neighbourhoods, located in two rows of n embedded rectangles as shown in Figure 4. The height of each rectangle is 1 unit, the widths are 6, 12, 18, ... units in each row, and the two rows are located 1 unit away from each other. The mid-points of the rectangles are denoted by P_i. The vertical distance between two neighbouring mid-points is then 2 units, whereas the horizontal distances are 3 units. The optimal TSP solution according to the decomposition approach is $(P_1, P_{n+1}, P_{n+2}, P_2, P_3, P_{n+3}, ...,$, as shown by the red dotted line in the figure. With the assumption that the travel speed is 1 unit and visits take ε time, this results in a TSP-ND solution where the path crosses between the two columns of rectangles n times, which takes $n + 2n\varepsilon$ time.

In contrast, the optimal TSP-ND solution is a path that visits all the rectangles in the same column at once, crossing between the two columns only once, illustrated by a blue dashed line in

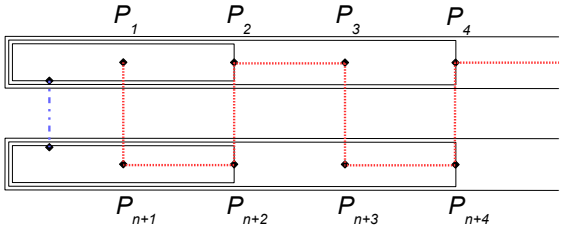

Figure 4. An instance where the performance of the decomposition approach can be arbitrarily bad. The red dotted line illustrates the solution of the TSP over the neighbourhood mid-points, whereas the blue dashed line shows an optimal TSP-ND solution.

the figure. Travelling this path takes only $1 + 2n\varepsilon$ time. Hence, with $n \to \infty$, the performance of the decomposition approach is indeed arbitrarily bad. □

It is emphasised that the above construction scheme can be easily adapted to 3D instances for RLW as well, with coplanar AV midpoints located exactly as above, and the axes of all truncated cones corresponding to horizontal lines in that plane.

4. A GRASP Meta-heuristic Algorithm

This section proposes a meta-heuristic approach based on *Greedy Randomised Adaptive Search Procedures* (GRASP) for solving the task sequencing and path planning problem in an integrated way. GRASP is a widely applied and efficient meta-heuristic for solving hard combinatorial optimisation problems (Pitsoulis and Resende 2002). It successively computes heuristic solutions by a randomised greedy algorithm, and improves them by a hill climbing local search. Upon reaching a local optimum, i.e., a solution that cannot be improved further, a new randomised solution is constructed. This procedure is iterated until a time limit is hit. The pseudo-code of GRASP for a minimisation problem can be sketched as follows, with $f(x)$ standing for the objective value of solution x, x^* for the best known solution, and f^* for its objective value:

```
PROCEDURE Grasp
  f* := ∞
  WHILE the time limit is not hit
   Compute a greedy randomised solution → x
   Local search from x until local opt. → x'
   IF f(x') < f* THEN
       x* := x'
       f* := f(x*)
  RETURN x*
```

In successful applications of GRASP, both the embedded randomised greedy algorithm and the local search relies on powerful operators for the specific problem domain. Below, we propose an algorithm that combines adaptations of classical local search operators for TSP for modifying the task sequence and geometric computation techniques for path planning. At all times, the algorithm maintains a stitch sequence (or partial sequence during the construction phase) and a close-to-optimal path corresponding to that sequence as the current solution. Below we review the algorithms associated to each step of the GRASP procedure, i.e., the construction heuristic, the improvement heuristic, and the path planning algorithm that computes the path for each new candidate sequence constructed by the two heuristics.

4.1 *Greedy Solutions*

Initial greedy solutions are constructed using an adapted version of the *farthest insertion* heuristic (Rosenkrantz, Stearns, and Lewis 1977) for TSP. The algorithm starts with a partial solution consisting of two stitches whose AV mid-points are the farthest from each other. The path corresponding to these two stitches is a linear section that connects the closest points of the two AVs. Then, in each iteration, for each stitch not yet sequenced, the algorithm looks for the best position for inserting the given stitch into the current path, by calling the path planner algorithm for evaluating all possible insertion positions, including at the beginning or at the end of the current path. The stitch whose *best* insertion causes the *greatest* increase in the cycle time is selected, and it is inserted at its *best* position. *Randomisation* is achieved by perturbing the cost of each candidate insertion. The intuition behind farthest insertion is that the approximate shape of the path is formed as early as possible, which renders it one of the most efficient heuristics for TSP with Euclidean distance measures.

The pseudo-code of the algorithm is displayed below, where π is the current path, and set U contains, at all times, the stitches not yet sequenced. Function $d_{\text{mid}}(i,j)$ denotes the distance between the midpoints of AV_i and AV_j, whereas $C(\pi[p,k])$ stands for the cycle time computed by the path planner for path π with stitch k inserted into it at position p. Both $d_{\text{mid}}(\cdot)$ and $C(\cdot)$ include a random perturbation in the form of a multiplier drawn from the uniform random distribution over $[1.0, 1.05)$ in the current implementation.

```
PROCEDURE GreedySolution
  LET (i, j) := arg max_{i,j∈U} d_mid(i, j)
  LET π = [i, j]
  LET U := {1, ..., n} \ {i, j}
  WHILE U ≠ ∅
   LET (k*, p*) := arg max_{k∈U} min_p C(π[p, k])
   Insert k* into π at position p*
   U := U \ {k*}
  RETURN π
```

4.2 *Improvement by Local Search*

The adopted *hill climbing* search modifies the initial solution iteratively using a *best improvement* strategy. In each iteration, it scans the neighbourhood of the current solution, and selects the neighbouring solution (task sequence and corresponding path) characterised by the lowest cycle time. If the neighbour improves on the cycle time of the current solution, the search continues by moving to this best neighbour. Search terminates when no further improvement can be made, which indicates that a local optimum is found.

Due to the high complexity of constructing a neighbour, the emphasis was given to relatively small-size neighbourhoods: the *2-opt* (deleting two edges and re-connecting the sequence) and *or-opt* (relocating a sub-sequence of max. length k to another position, in forward or backward orientation) neighbourhoods, with sizes $O(n^2)$ and $O(kn^2)$, respectively (Johnson and McGeoch 1997). In the actual implementation, the value of $k = 5$ was applied. The two neighbourhood functions are illustrated in Figure 5.

In order to reduce the computational burden of constructing paths for every neighbour, the cycle time of each neighbour is heuristically estimated first, without running the path planner on the modified sequence. Let $R = \{(i,j)\}$ denote the set of edges (i,j) *removed* from the current sequence by the applied neighbourhood function. The time of travelling along these edges on the current path then equals $\frac{1}{v}\sum_{(i,j)\in R}|\overline{b_i a_j}|$. Analogously, let $A = \{(i,j)\}$ stand for the set of edges *added* to the sequence by the neighbourhood function. A lower bound estimate of the time of travelling these edges is given by $\frac{1}{v}\sum_{(i,j)\in A} d(AV_i, AV_j)$, where $d(AV_i, AV_j)$ is the minimum distance of AV_i

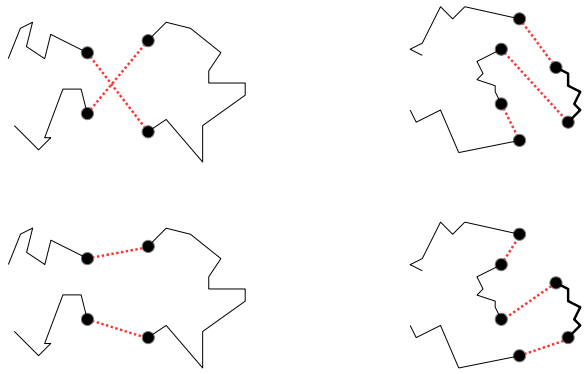

Figure 5. Illustration of the 2-opt (left) and the or-opt (right) neighbourhood functions. In this example, or-opt relocates the sub-sequence highlighted with thick line.

and AV_j. Finally, let C_0 denote the cycle time of the current solution. Hence, the heuristic

$$C_H = C_0 \; - \; \frac{1}{v}\left(\sum_{(i,j)\in R} |\overline{b_i a_j}| \; + \; \sum_{(i,j)\in A} d(AV_i, AV_j) \right)$$

under-estimates with high confidence the cycle time of the neighbour. Still, C_H is not a valid lower bound, since the path segments corresponding to edges not affected by the neighbourhood function may be modified by path planning. The neighbourhood is filtered by fathoming all neighbours that have $C_H > C_0$ without planning a path for these sequences.

4.3 *Path Planning*

The path planner computes a close-to-optimal SCP path for each task sequence modified by one of the above heuristics. This is performed by the Rubberband Algorithm (RBA), an efficient greedy method which has been proven to converge to the optimal path for various related problems, see, e.g., (Pan, Li, and Klette 2010). Nevertheless, it is possible to construct instances of TSP-ND where RBA gets stuck in a local optimum.

For computing an SCP path for a neighbour, RBA departs from the path in the current solution, and adapts it to the changes performed by the neighbourhood function. The algorithm sweeps along the path, and adjusts a single corner point of the broken line at a time to its locally optimal position. According to Lemma 3.3, the new position of a start point $a_{[i]}$ must be the position in the intersection of $\mathrm{AV}_{[i]}$ and a sphere around $b_{[i]}$ that minimises the distance $d(b_{[i-1]}, a_{[i]})$. This position $a'_{[i]}$ can be computed using closed-form geometric calculations.

The new positions of end points $b_{[i]}$ can be calculated in a similar way, exploiting the symmetry that these are starting points belonging to the same stitch along a reversed path. Finally, the start and end points of the broken line path are determined as $a_{[1]} := b_{[1]}$ and $b_{[n]} := a_{[n]}$. Sweeps along the path are iterated until the reduction of the cycle time between two subsequent iterations is lower than a given threshold, 0.03% in the current implementation.

5. Experimental Results

The proposed models and algorithms have been evaluated on problems involving the assembly of a car front door using RLW. Experiments have been performed on real industrial data, containing a single door geometry with different stitch layouts and realistic technological parameters. The experiments investigated the solution quality achieved using the proposed techniques, and they involved computing a task sequence and a robot path using three different algorithms:

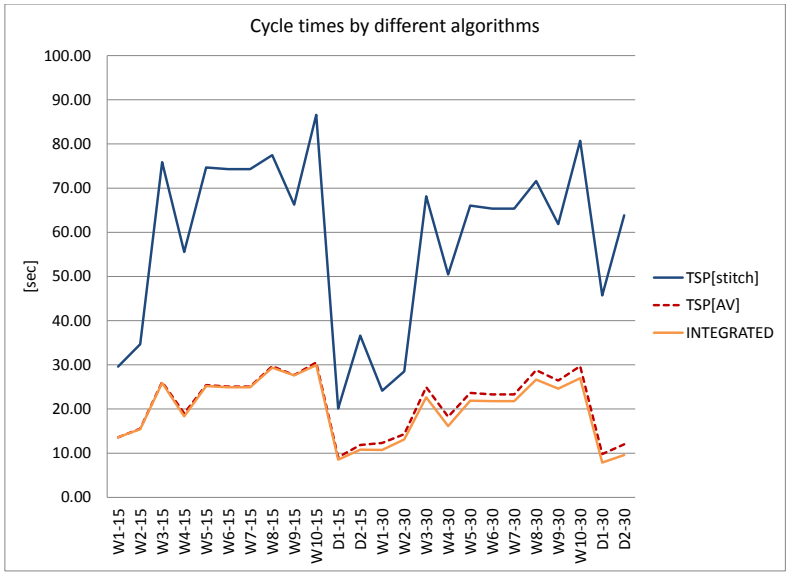

Figure 6. Comparison of the cycle times achieved by the three different algorithms for the 24 test instances.

- TSP[stitch], the single sequencing algorithm dedicated to RLW from the literature (Reinhart, Munzert, and Vogl 2008), which solves the sequencing problem as a TSP over the stitch positions. Implicitly, the approach minimises the length of the TCP path, which is only loosely related to the cycle time. A robot path is computed for the given sequence by using the algorithm described in Section 4.3.
- TSP[AV], an algorithm that solves a TSP over the AV mid-points for computing the stitch sequence. While from the computational aspect it is very similar to TSP[stitch], it can be considered as the straightforward decomposition approach to solving the TSP-ND model of the problem, see Section 3.3;
- INTEGRATED, the algorithm proposed above for integrated task sequencing and path planning, adopting the TSP-ND model.

The available industrial data contained 10 different stitch layouts for welding (instance names W1-W10), as well as two layouts for laser dimpling operations (instances D1 and D2). The original data set was cleaned to remove inconsistencies and to ensure that all stitches can be welded along a collision-free robot path. The experiments were performed using a maximum inclination angle of 15° and 30° as well (instance names postfixed with -15 and -30), which resulted in 20 different welding instances and 4 dimpling instances altogether. The number of stitches in an instance varied between 28 and 71.

The detailed comparison of the performance of the three algorithms is presented in Table 1. Each row stands for a separate problem instance. Column n contains the number of stitches. For each algorithm, columns *time* and *len* contain the cycle time and the path length belonging to the stitch sequence computed by the given algorithm. The best values encountered over the different algorithms are highlighted by bold font for each instance. The cycle time achieved by the different algorithms are also visualised in Figure 6.

The experiments were run on a 2.40 GHz Intel Core i5 computer with 4GB RAM. TSP[stitch] and TSP[AV] used ILOG CP as a TSP solver. It is noted that ILOG CP terminated in local optima typically in less then 1 second, whereas INTEGRATED was run for 600 seconds on each instance. This time limit allowed typically 100-200 iterations in the GRASP meta-heuristic, involving a total of 5000-8000 iterations during the hill climbing searches and evaluating $4\text{-}6\cdot10^6$ sequences by path planning.

The results unambiguously indicate the dominance of proposed TSP-ND model (algorithms INTEGRATED and TSP[AV]) over the classical minimisation of the TCP path (algorithm TSP[stitch]), see Figure 7. For workpieces with complex geometry, TSP[stitch] is completely inad-

Table 1. Comparison of the TSP[stitch], TSP[AV], and INTEGRATED algorithms.

		TSP[stitch]		TSP[AV]		INTEGRATED	
	n	time	len	time	len	time	len
W1-15	28	29.61	14.24	**13.60**	5.35	13.62	**5.26**
W2-15	34	34.66	15.97	15.54	**5.72**	**15.41**	6.07
W3-15	62	75.87	35.33	26.13	**7.11**	**25.91**	7.33
W4-15	44	55.54	25.98	19.02	**6.19**	**18.34**	6.59
W5-15	64	74.66	34.51	25.42	7.07	**25.21**	**6.95**
W6-15	63	74.29	34.49	25.08	7.14	**24.91**	**6.67**
W7-15	63	74.29	34.49	25.08	7.14	**24.91**	**6.67**
W8-15	71	77.47	33.90	29.70	**6.98**	**29.35**	7.30
W9-15	67	66.27	28.74	27.67	7.01	**27.63**	**6.67**
W10-15	71	86.60	39.40	30.53	7.61	**29.96**	**7.12**
D1-15	59	20.11	11.80	9.10	5.40	**8.54**	**5.06**
D2-15	62	36.63	21.51	11.84	7.02	**10.79**	**6.36**
W1-30	28	24.14	11.54	12.31	4.69	**10.72**	**4.40**
W2-30	34	28.53	12.81	14.30	5.11	**13.11**	**4.38**
W3-30	62	68.17	30.81	24.96	6.65	**22.65**	**5.43**
W4-30	44	50.46	22.82	18.24	5.88	**16.15**	**4.55**
W5-30	64	66.05	29.84	23.64	6.68	**21.87**	**5.38**
W6-30	63	65.36	29.64	23.30	6.55	**21.78**	**5.32**
W7-30	63	65.36	29.64	23.30	6.55	**21.78**	**5.32**
W8-30	71	71.61	30.68	28.79	6.71	**26.67**	**5.66**
W9-30	67	61.85	26.35	26.44	6.54	**24.60**	**5.71**
W10-30	71	80.72	36.10	29.70	7.30	**26.98**	**6.01**
D1-30	71	45.72	26.65	9.81	5.80	**7.88**	**4.64**
D2-30	71	63.83	37.34	12.02	7.15	**9.58**	**5.53**
Avg.	58	58.24	27.27	21.06	6.47	**19.93**	**5.85**

equate, since it moves the scanner head in a zigzag above stitches that have nearby positions but different surface normals. In case of a car door, this phenomenon is the most spectacular around the window frame, where the stitches on the inner and the outer sides are close to each other, but must be welded from opposite directions. Consequently, in our experiments, TSP[stitch] resulted in 2-6 times higher cycle times and path lengths than INTEGRATED and TSP[AV].

Regarding the performance of the algorithms working on the TSP-ND model, INTEGRATED computed on average 5.7% lower cycle times and 10.7% shorter paths than TSP[AV]. INTEGRATED also found the lowest cycle time for 23 instances, and the shortest robot paths for 20 out of the total 24 problem instances. The difference was particularly significant on the instances with 30° inclination angle, where INTEGRATED could efficiently exploit the large access volumes to weld and move the robot in parallel, resulting in an average gain of 12% in cycle time and 21.2% in path length. The most significant difference (24.5-25.5% in cycle time and 25.0-29.2% in path length) was encountered on the dimpling instances D1-30 and D2-30, in which the duration of the effective tasks is dominated by the duration of robot motion, and hence, nearly all tasks could be executed while the robot was moving. The reduction of the cycle time and the path length also decreases the energy consumption of the RLW workstation and the RLW robot itself.

Obviously, various other approaches might be successful in solving TSP-ND, including iterative methods or algorithms based on the discretisation of the 3D space. Additional experiments have been carried out on the latter approach, by encoding a subset of the above TSP-ND instances into GTSPs and solving them using GLKH (Helsgaun 2015) as a black-box solver. Asymmetric GTSP instances have been generated by sampling the AVs along a rectangular grid and creating two clusters of nodes for each stitch, corresponding to the candidate start (resp. end) points for the given stitch. Non-Euclidean edge weights had to be specified in full matrix format, which imposed a strict limit on the applicable grid resolution. Accordingly, GTSPs with 6800–7200 nodes, corresponding to file sizes of 210–240MB, were considered. The stitch sequence was extracted from the GTSP solution, and the path planning algorithm presented in Section 4.3 was run on this sequence. The GTSP approach achieved 1.6-10.7% (on average 5.2%) worse results than INTEGRATED; and at most 1.2% worse, at most 2.0% (on average 0.4%) better results than TSP[AV].

Table 2. Cycle times obtained with different algorithms, in various phases of the OLP workflow.

	TSP[stitch]	TSP[AV]	INTEGRATED
Rough-cut path	89.07	41.35	39.99
Collision-free path	93.80	43.90	43.65
Robot motion plan	93.87	57.69	56.43

6. Discussion on the Application of the Results in an OLP System

The path computed by the proposed approach undergoes various transformations in an OLP system before it is actually executed on a physical robot (see a potential workflow in Figure 2). The two steps of the OLP workflow that may result in considerable changes in the SCP path or in its cycle time are collision avoidance and the inverse kinematic transformation that converts the SCP path from a Cartesian coordinate system to the robot joint configuration space. Below we overview the effects of these transformations, using as illustration a case study involving a car door that has between physically welded using a robot program automatically generated by the OLP system proposed in (Erdős et al. 2015). This OLP system includes an implementation of the investigated TSP[stitch], TSP[AV], and INTEGRATED algorithms for task sequencing and path planning. The case study is based on the same car door design as in the computational experiments above, however, laser power (and hence, the welding speed) had to be reduced due to technological reasons. For this reason, the cycle times in the case study (see Table 2) are higher than above, yet, the absolute difference between the cycle times obtained with different algorithms is comparable.

While the algorithms proposed in this paper ignore potential collisions along the path, in industrial practice the robot path must be free of any collisions. We have proposed a collision-free path planning algorithm for RLW in (Kovács 2014) based on the common fixture design guideline in RLW that the access volumes of the stitches must be kept clear. Respecting this guideline guarantees that the above computed path is free of collisions. On the other hand, our experience suggests that the guideline is sometimes overridden by other design objectives, possibly resulting in collisions between the *robot* and the *workpiece* or the *fixture*, or between the *laser beam* and the *workpiece* or the *fixture*. In such cases, the proposed collision-free path planning algorithm detects all collisions along the above computed rough-cut path, and fixes them by recalculating the colliding path segments and their neighbouring segments, while leaving the task sequence and the path segments far from collisions unchanged. The difference of the initial path and the subsequent collision-free path for a real industrial workpiece is illustrated in Figure 8. The experimental results presented in (Kovács 2014) confirm that the above algorithms lead to high-quality collision-free paths, and the proposed INTEGRATED approach outperforms other investigated algorithms regarding the cycle time of the collision-free path as well. Collision avoidance resulted in an average 5.6%, and in at most 14.4% increase in cycle times, with strong correlation to the accessibility of the stitches in the given instance. It must be noted that for a few instances characterised by stitches with extremely

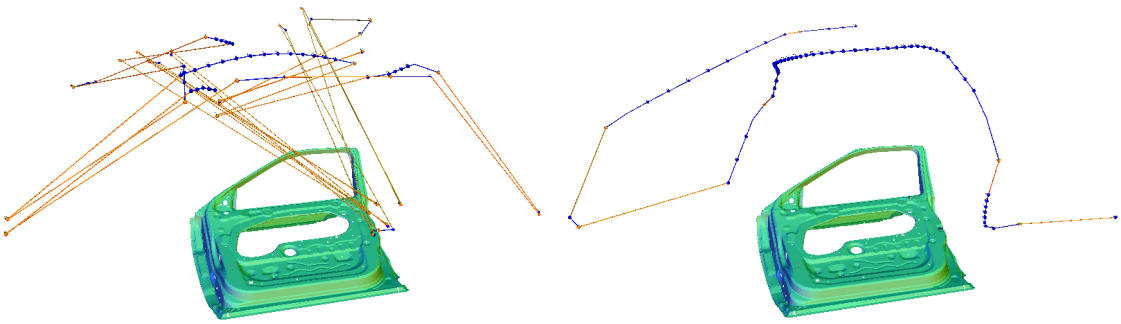

Figure 7. Comparison of the paths computed by the TSP[stitch] (above) and the proposed INTEGRATED (below) methods. TSP[stitch] focuses merely on the TCP path, while INTEGRATED is able to consider the path of the SCP as well.

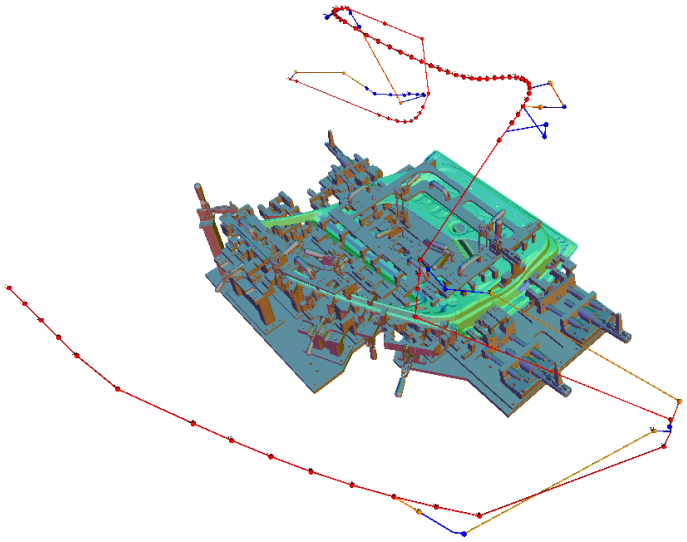

Figure 8. Comparison of the rough-cut path (red) and the collision-free path (blue-yellow).

poor accessibility, TSP[AV] resulted in more efficient paths than INTEGRATED. This suggests that the assumption made by TSP[AV] during sequencing, i.e., that stitches must be welded from the center of their access volume, is closer to reality than the assumption of INTEGRATED that the whole access volume can be used. Therefore, the refinement of the proposed algorithms for problems with extremely poor accessibility is an interesting topic for future research. The difficulty of considering potential collisions already during task sequencing resides in the free form geometry of the collision-free access volumes, although the explicit consideration of this geometry can be avoided, e.g., by sampling the access volumes and solving a GTSP over the sample points.

The inverse kinematic transformation was solved by a slightly improved version of the algorithm presented in (Erdős et al. 2015). This transformation is of particular interest because of the redundancy of the kinematic chain of a typical RLW robot (i.e., a path can be mapped to different robot joint motion plans with different cycle times) and because the transformation cannot be solved in a closed form. In the case study, the robot motion plan had significantly larger cycle time than predicted by the earlier planning phases. The increase originated from two sources. First, the time of repositioning the laser beam by moving the mirrors and lenses in the scanner was assumed to be zero in the proposed model; in reality, it was 3.76 seconds for the 72 stitches on the sample door for the INTEGRATED path. Second, due to the finite acceleration of the robot's mechanical joints, robot motion often took longer than predicted by the path planner based on the fixed velocity v specified in the robot's manual. This affected mostly the shorter movements, resulting in an increase of 9.02 seconds for INTEGRATED. Long idle movements were not affected, since sometimes the robot could reach higher speeds than v, which led to negligible increase of cycle time for the TSP[stitch] path. Nevertheless, the performance order of the three investigated algorithms coincides on the rough-cut path and on the robot motion plan.

7. Conclusions

This paper proposed a new model and an efficient algorithm for solving the problem of integrated task sequencing and path planning for RLW. The proposed approach encompasses two key novelties. First, a formal model has been introduced that, using a new extension of the well-known TSP, effectively captures the relevant aspects of this technological planning problem, including the coupled movement of the quick tool (laser beam) and the relatively slow robot, and allows exploiting the high degree of freedom in choosing the robot path when welding a well-defined stitch position.

Then, an efficient local search algorithm has been proposed for solving the sequencing and the path planning subproblems together, in a tightly integrated way. The model and the algorithm apply a continuous space representation, and hence, circumvent the losses stemming from sampling that characterises many other approaches working on a discretised space representation.

The efficiency of the proposed approach has been demonstrated in computational experiments on real industrial data, where it reduced significantly the cycle time of the welding operation. Compared to the classical approach in industry, which focuses on the minimisation of the TCP path length, the cycle time decreased by a factor of 2-6, showing clearly the inadequacy of the techniques originally developed for other technologies, requiring physical contact. Furthermore, the proposed algorithm resulted in an up to 25.5%, on average 5.7% reduction of the cycle time compared to the more straightforward decomposition approach. It has also been shown that the decomposition approach may perform arbitrarily poor on artificially constructed instances. The reduction of the cycle time not only increases the throughput of the RLW workstation, but also decreases the energy consumed by the welding robot and the workstation.

The approach may find applications in process planning for other technologies as well, where relatively slow robot motion is coupled with quick positioning of the tool, including monitoring, inspection, or spray painting. The techniques have already been applied to planning the dimpling operations in an RLW workstation.

The presented algorithms are components of a recently developed OLP toolkit for RLW in the automotive industry. This toolbox is expected to help production engineers generate efficient robot programs from the description of the workpiece and the available resources in a reproducible way, much quicker and more efficiently than it is done manually today. In addition, automating this level of planning supports the verification of decisions made on higher levels of the planning hierarchy, e.g., workpiece and fixture design, or the configuration of the welding cell. The OLP system, including the proposed sequencing and path planning algorithms, has been validated and evaluated in simulation scenarios and in physical experiments involving the assembly of a real car door.

Acknowledgments

The author thanks J. Váncza and G. Erdős for the helpful discussions. This work has been supported by grants EU FP7 No. 285051 “RLW Navigator” and NFÜ ED-13-2-2013-0002.

References

Alatartsev, S., V. Mersheeva, M. Augustine, and F. Ortmeier. 2013. “On Optimizing a Sequence of Robotic Tasks.” In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems,* 217–223.

Alatartsev, S., S. Stellmacher, and F. Ortmeier. 2015. “Robotic Task Sequencing Problem: A Survey.” *J. of Intelligent & Robotic Systems* in print.

Arkin, E. M., and R. Hassin. 1994. “Approximation algorithms for the geometric covering salesman problem.” *Discrete Applied Mathematics* 55 (3): 197–218.

Behzad, A., and M. Modarres. 2002. “A New Efficient Transformation of Generalized Traveling Salesman Problem into Traveling Salesman Problem.” In *Proc. 15th Int. Conf. of Systems Engineering,* 6–8.

Buehrle, J., M. Bea, and R. Brockmann. 2013. “Laser Remote Process Technology on Automotive Manufacture.” In *Proceedings of the FISITA 2012 World Automotive Congress,* Vol. 199 of *Lecture Notes in Electrical Engineering*89–97. Springer.

Castelino, K., R. D’Souza, and P. K. Wright. 2002. “Toolpath Optimization for Minimizing Airtime During Machining.” *J. of Manufacturing Systems* 22 (3): 173–180.

Dewil, R., P. Vansteenwegen, and D. Cattrysse. 2014. “Construction heuristics for generating tool paths for laser cutters.” *Int. J. of Production Research* 52 (20): 5965–5984.

Dewil, R., P. Vansteenwegen, D. Cattrysse, Manuel Laguna, and Thomas Vossen. 2015. “An improvement

heuristic framework for the laser cutting tool path problem." *Int. J. of Production Research* 53 (6): 1761–1776.

Dror, M., A. Efrat, A. Lubiw, and J. S. B. Mitchell. 2003. "Touring a sequence of polygons." In *35th Annual ACM Symp. Theory of Computing,* 473–482.

Elbassioni, K. M., A. V. Fishkin, and R. Sitters. 2009. "Approximation Algorithms for the Euclidean Traveling Salesman Problem with Discrete and Continuous Neighborhoods." *Int. J. of Computational Geometry and Applications* 19 (2): 173–193.

Erdős, G., Cs. Kardos, Zs. Kemény, A. Kovács, and J. Váncza. 2014. "Workstation configuration and process planning for RLW operations." *Procedia CIRP* 17: 783–788.

Erdős, G., Cs. Kardos, Zs. Kemény, A. Kovács, and J. Váncza. 2015. "Process planning and programming for robotic remote laser welding systems." *Int. J. of Computer Integrated Manufacturing* in print.

Gentilini, I., F. Margot, and K. Shimada. 2013. "The travelling salesman problem with neighbourhoods: MINLP solution." *Optimization Methods and Software* 28 (2): 364–378.

Hatwig, J., P. Minnerup, M. F. Zaeh, and G. Reinhart. 2012. "An automated path planning system for a robot with a laser scanner for remote laser cutting and welding." In *2012 IEEE Int. Conf. Mechatronics and Automation,* 1323–1328.

Helsgaun, K. 2015. "Solving the equality generalized traveling salesman problem using the Lin-Kernighan-Helsgaun Algorithm." *Mathematical Programming Computation* in print.

Johnson, D. S., and L. A. McGeoch. 1997. "The Traveling Salesman Problem: A Case Study in Local Optimization." In *Local Search in Combinatorial Optimization,* edited by E. H. L. Aarts and J. K. Lenstra. 215–310. John Wiley and Sons, Ltd.

Kaelbling, L.P., and T. Lozano-Perez. 2011. "Hierarchical task and motion planning in the now." In *2011 IEEE Int. Conf. Robotics and Automation,* 1470–1477.

Karapetyan, D., and G. Gutin. 2011. "Lin-Kernighan Heuristic Adaptations for the Generalized Traveling Salesman Problem." *European J. of Operational Research* 208 (3): 221–232.

Kolakowska, E., S. F. Smith, and M. Kristiansen. 2014. "Constraint optimization model of a scheduling problem for a robotic arm in automatic systems." *Robotics and Autonomous Systems* 62 (2): 267–280.

Kovács, A. 2013. "Task Sequencing for Remote Laser Welding in the Automotive Industry." In *Proc. 23rd Int. Conf. Automated Planning and Scheduling,* 457–461.

Kovács, A. 2014. "Collision-free path planning for remote laser welding." In *2nd ICAPS Workshop on Planning and Robotics (PlanRob2014),* 172–181.

Lee, M-K., and K-B. Kwon. 2006. "Cutting path optimization in CNC cutting processes using a two-step genetic algorithm." *Int. J. of Production Research* 44 (24): 5307–5326.

Noon, C. E., and J. C. Bean. 1993. "An efficient transformation of the generalized travelling salesman problem." *INFOR* 31: 29–44.

Pan, X., F. Li, and R. Klette. 2010. "Approximate shortest path algorithms for sequences of pairwise disjoint simple polygons." In *Proc. 22nd Annual Canadian Conf. Computational Geometry,* 175–178.

Park, H-S., and H-W. Choi. 2010. "Development of digital laser welding system for car side panels." In *Laser Welding,* edited by X. Na. 181–192. InTech.

Pitsoulis, L., and M. G. C. Resende. 2002. "Greedy randomized adaptive search procedures." In *Handbook of Applied Optimization,* edited by P. M. Pardalos and M. G. C. Resende. 168–181. Oxford University Press.

Reinhart, G., U. Munzert, and W. Vogl. 2008. "A programming system for robot-based remote-laser-welding with conventional optics." *CIRP Annals – Manufacturing Technology* 57 (1): 37–40.

Rice, M. N., and V. J. Tsotras. 2013. "Parameterized algorithms for generalized traveling salesman problems in road networks." In *Proc. 21st ACM SIGSPATIAL Int. Conf. Advances in Geographic Information Systems,* 114–123.

Rosenkrantz, D.J., R.E. Stearns, and P.M. Lewis. 1977. "An analysis of several heuristics for the traveling salesman problem." *SIAM J. on Computing* 6 (3): 563–581.

Safra, S., and O. Schwartz. 2006. "On the complexity of approximating TSP with neighborhoods and related problems." *Computational Complexity* 14 (4): 281–307.

Saha, M., G. Sánchez-Ante, T. Roughgarden, and J-C. Latombe. 2006. "Planning Tours of Robotic Arms Among Partitioned Goals." *Int. J. of Robotics Research* 25 (3): 207–223.

Srivastava, S., E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel. 2014. "Combined Task and Motion Planning Through an Extensible Planner-Independent Interface Layer." In *2014 IEEE Int. Conf. Robotics and Automation,* 639–646.

Veeramani, D., and S. Kumar. 1998. "Optimization of the nibbling operation on an NC turret punch press."

Int. J. of Production Research 36 (7): 1901–1916.

Wang, G. G., and S. Q. Xie. 2005. "Optimal process planning for a combined punch-and-laser cutting machine using ant colony optimization." *Int. J. of Production Research* 43 (11): 2195–2216.

Xie, S. Q., J. Gan, and G. G. Wang. 2009. "Optimal process planning for compound laser cutting and punch using Genetic Algorithms." *Int. J. of Mechatronics and Manufacturing Systems* 2 (1/2): 20–38.