International Journal of Computer Integrated Manufacturing
Vol. 00, No. 00, Month 200x, 1–28

RESEARCH ARTICLE

Process planning and offline programming for robotic remote laser welding systems

Gábor Erdős[a,b], Csaba Kardos[a,b], Zsolt Kemény[a], András Kovács[a] and József Váncza[a,b]

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

[b] *Department of Manufacturing Science and Technology, Budapest University of Technology and Economics, Hungary*

(*Received 00 Month 200x; final version received 00 Month 200x*)

The paper introduces a complete offline programming toolbox for remote laser welding (RLW) which provides a semi-automated method for computing close-to-optimal robot programs. A workflow is proposed for the complete planning process, and new models and algorithms are presented for solving the optimization problems related to each step of the workflow: the sequencing of the welding tasks, path planning, workpiece placement, calculation of inverse kinematics and the robot trajectory, as well as for generating the robot program code. The paper summarizes the results of an industrial case study on the assembly of a car door using RLW technology, which illustrates the feasibility and the efficiency of the proposed approach.

Keywords: robot; remote laser welding; path planning; collision avoidance; offline programming; optimisation

ISSN: 0951-192x print/ISSN 1362-3052 online
© 200x Taylor & Francis
DOI: 10.1080/0951192YYxxxxxxxx
http://www.informaworld.com

1. Introduction

One of today's most significant technological trends in car body making is the spreading application of *remote laser welding* (RLW). In contrast to traditional *resistance spot welding* (RSW), RLW uses a laser beam projected by a scanner mounted on an industrial robot. This allows *contactless* welding from a *remote* point, circumventing many accessibility constraints and allowing faster operation. In view of high investment costs, however, RLW is only justified for a major reduction of cycle time or comparable gains (Ceglarek *et al.* 2004). Moreover, traditional, online robot programming methods are hardly applicable to RLW and cannot achieve the desired efficiency (Reinhart *et al.* 2008), while software support for offline programming for RLW is barely available.

This paper addresses the problem of generating close-to-optimal offline robot programs for RLW in a semi-automated way, built on the initial workpiece model, the welding task definitions, and the description of available equipment. A planning workflow is proposed that decomposes the overall problem into a hierarchy of sub-problems. Novel methods, tailored to the needs of RLW, are introduced for each step of the workflow. Computational experiments and a detailed case study on real industrial data, involving the assembly of a car door, demonstrate the feasibility and the efficiency of the approach.

The paper is organized as follows. After discussing the technological background and reviewing the related literature, Section 2 introduces particular assumptions and problem formulation. The overall workflow of planning is discussed in Section 3. Section 4 gives a detailed presentation of the methods proposed for solving the subproblems related to the different steps of the workflow. Afterwards, further details are given on the prototype implementation (Section 5) and the results of experiments are discussed (Section 6). Finally, conclusions are drawn and directions for future research are proposed.

1.1 *Technological background*

Laser welding can be regarded as a special way of applying heat to melt the materials to be joined. RLW is typically performed by a *single robot* with 4–5 revolute joints and a *laser scanner*. The robot arm moves the scanner with 0.2–0.6 m/s at most, the low scanner weight allowing rather high acceleration. The scanner head contains two mirrors, for the rapid positioning of the laser beam (up to 5 m/s at the laser-to-workpiece contact point), and lenses to set the focal length. Hence, the typical RLW robot is a redundant kinematic system with 7 degrees of freedom (DOF), in which the scanner mirrors and focus move an order of magnitude faster than the mechanical joints of the robot arm.

When joining sheet metal parts with RLW, product variation has been recognised to have crucial influence on product quality (Li *et al.* 2002, Franciosa *et al.* 2014). Galvanized steel, for instance, requires a part-to-part gap of 0.05–0.3 mm (Franciosa *et al.* 2014), warranted by *dimpling* one of the layers, and holding the assembly—including stitch locations—accurately in place with a *welding fixture* (Li *et al.* 2002). The RLW process is also affected by main process parameters such as laser power, welding speed, inclination angle of the laser beam as well as material stack-up. Figure 1 depicts a basic RLW setup in a simulated scenario in the software tool developed.

RLW has a number of *advantages*, such as extended useful workspace resulting from *contactless, single-sided and remote access* to the workpiece (Tsoukantas *et al.* 2007); easier access to tight corners due to narrow beam width; or faster welding and repositioning processes due to the large control bandwidth and small inertia of optical components. The coordinated motion of robot joints and optical components allows on-the-fly operation,

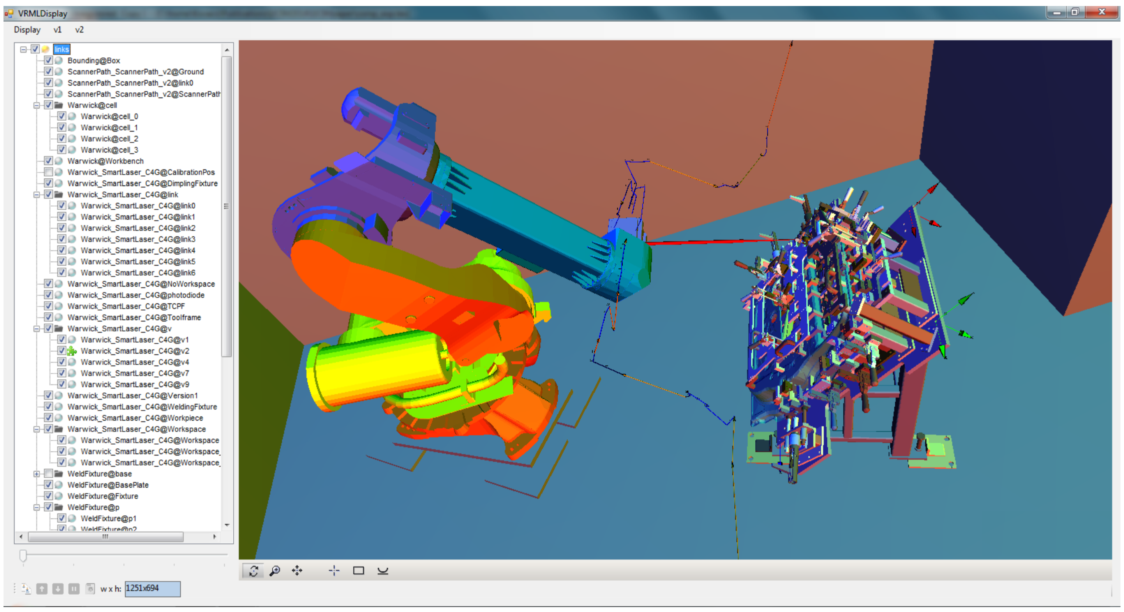

VRMLDisplay
Display
v1
v2
links
Bounding@Box
ScannerPath_ScannerPath_v2@Ground
ScannerPath_ScannerPath_v2@link0
ScannerPath_ScannerPath_v2@ScannerPath
Warwick@cell
Warwick@cell_0
Warwick@cell_1
Warwick@cell_2
Warwick@cell_3
Warwick@Workbench
Warwick_SmartLaser_C4G@CalibrationPos
Warwick_SmartLaser_C4G@DimplingFixture
Warwick_SmartLaser_C4G@link
Warwick_SmartLaser_C4G@link0
Warwick_SmartLaser_C4G@link1
Warwick_SmartLaser_C4G@link2
Warwick_SmartLaser_C4G@link3
Warwick_SmartLaser_C4G@link4
Warwick_SmartLaser_C4G@link5
Warwick_SmartLaser_C4G@link6
Warwick_SmartLaser_C4G@NoWorkspace
Warwick_SmartLaser_C4G@photodiode
Warwick_SmartLaser_C4G@TCPF
Warwick_SmartLaser_C4G@Toolframe
Warwick_SmartLaser_C4G@v
Warwick_SmartLaser_C4G@v1
Warwick_SmartLaser_C4G@v2
Warwick_SmartLaser_C4G@v4
Warwick_SmartLaser_C4G@v7
Warwick_SmartLaser_C4G@v9
Warwick_SmartLaser_C4G@Version1
Warwick_SmartLaser_C4G@WeldingFixture
Warwick_SmartLaser_C4G@Workpiece
Warwick_SmartLaser_C4G@Workspace
Warwick_SmartLaser_C4G@Workspace
Warwick_SmartLaser_C4G@Workspace_
Warwick_SmartLaser_C4G@Workspace_
WeldFixture@base
WeldFixture@BasePlate
WeldFixture@Fixture
WeldFixture@p
WeldFixture@p1
WeldFixture@p2
w x h: 1251x694

Figure 1. The basic setup of RLW technology: robot and path of the scanner head, workpiece in dedicated fixture.

reducing processing time and energy consumption (due to smaller control transients).

Nevertheless, laser welding does have its specific *application constraints* and *costs*. Most importantly, visibility has to be ensured for the entire length of the stitch. This has to be taken into consideration for the planning of fixtures, robot motion (avoiding occluding segments), and layout of potential visibility obstacles (even parts of the workpiece). Since the welding fixture should warrant a fit-up of the mating surfaces with tightly controlled gap tolerance, it is typically of a complex design that may easily impair stitch visibility. Due to surface penetration properties, the beam-to-surface inclination angle has to remain within technologically prescribed bounds. Further—possibly very tight—limitations may apply to beam deflection angle and focal length, requiring special care in calculating the rest of the robot motion, and precluding conventional online robot programming per se. Finally, the total first-cost of RLW technology is substantially higher than that of RSW, requiring laser source, complex and expensive fixturing, safety appliances and dedicated part moving devices in addition to the RLW robot itself.

Given its benefits and drawbacks, car manufacturing is the prevailing area of RLW deployment—nonetheless, even here, switching from RSW to RLW is subject to careful analysis of costs and expected advantages in the given manufacturing context (Zaeh *et al.* 2010, Erdős *et al.* 2013).

1.2 *Related works*

While offline programming for RLW robots has only been proposed recently (Reinhart *et al.* 2008, Hatwig *et al.* 2010, Munzert 2010), many of the underlying planning and execution control aspects have received significant attention in the scientific community. Shibata (2008), Tsoukantas *et al.* (2007), Singh *et al.* (2014) deal with the process of exposing the workpiece to the laser beam, giving insight into constraints of the physical process (e.g., timing, temperature control, dynamics of laser scanner tool) which have to be observed for feasible robot motion control and planning. The nature of laser welding, especially its high processing speed, may also affect long-term decisions regarding workpiece positioning (Mitsi *et al.* 2008), or welding cell layout (Iordachescu *et al.* 2011). Much of these has been subject to intuitive or empirical human planning, but recent trends point towards (semi-)automatic support for these once-per-type or once-per-facility decisions, too.

Low-level robot motion control and higher-level planning have to solve a number of typical problems related to welding applications, from efficient sequencing of welding tasks and the coordinated motion of robot and scanner with different control properties (Hatwig *et al.* 2010), all the way to optimized robot motion planning (Gasparetto and Zanotto 2010, Pashkevich *et al.* 2004). Algorithms for task sequencing and robot path planning are introduced by Reinhart *et al.* (2008), where task sequencing is performed by solving a travelling salesman problem (TSP) over the stitch positions in the Cartesian space. A drawback of the approach is that it ignores detailed geometry, accessibility constraints, and technological parameters. A similar model is applied and construction heuristics are proposed for path planning in laser cutting by Dewil *et al.* (2014). The applied model also observes ordering constraints among the contours to be cut.

The minimization of processing time in milling is investigated by Castelino *et al.* (2002). A generalized TSP (GTSP) approach is proposed, where the nodes correspond to the candidate tool entry/exit points for machining a feature. Potential collisions are neglected. A TSP with neighbourhoods (TSPN) model is proposed by Alatartsev *et al.* (2013) for sequencing robotic tasks with start/end points chosen arbitrarily along open or closed contours. A multi-objective constraint optimization model is proposed by Kolakowska *et al.* (2014) for task sequencing in spray painting, for minimizing cycle time and maximizing paint quality at the same time. Process planning methods considering other criteria, e.g., reduced carbon emission, are presented by (Yin *et al.* 2014).

An efficient, generic task sequencing and collision-free path planning model, with illustrations from resistance spot welding (RSW) is presented by 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 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, reducing computationally 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 continuous space while welding.

The relative position and orientation (*placement*) of robot and workpiece can have much impact on performance or feasibility of the robot motion task associated to the workpiece—still, (semi-)automatic placement calculation is fairly rare. Good placement is both *feasible* (workspace constraints or other hard requirements are satisfied) and *optimal* (performance criteria are optimised). Related works commonly assume a feasible task right away, thereby removing many interdependencies. Some approaches operate with Cartesian positions and orientations only, and formulate the problem based on manipulator properties or workspace constraints (Mitsi *et al.* 2008, Pamanes and Zeghloul 1991, Tian and Collins 2005). Other works assume a-priori knowledge of a path (Zeghloul and Pamanes-Garcia 1993, dos Santos *et al.* 2010), some of them exploiting path-specific criteria, or calculating proper performance measures (dos Santos *et al.* 2010, Nektarios and Aspragathos 2010). Placement is a difficult—possibly constrained—optimisation problem with many interdependencies, typically solved using meta-heuristics. Most often, genetic algorithms are used (dos Santos *et al.* 2010, Nektarios and Aspragathos 2010, Tian and Collins 2005), but tunnelling algorithms (Levy and Gómez 1985) and exhaustive search for less demanding cases (Hwang and Watterberg 1996) are also known.

2. Problem statement

The paper focuses on the *welding cycle of an assembly cell* that includes the workpiece fastened in a fixture, a single RLW robot, and fixed equipment of the workcell, including ground and boundaries. Consequently, planning and synchronizing the operations of other equipment that may manipulate the workpiece or monitor the process are out of scope now. The workcell is assumed to be embedded in a production system, hence the set of tasks to be accomplished in a single cycle, as well as the maximal total cycle time of welding operations, are specified as input. Due to the fixture, most welds must be performed on a specific side of the workpiece, requiring welding tasks to be defined as a linear or circular stitch together with a (directed) surface normal. Also, fixture design is expected to provide the 3D model of a welding fixture that, besides warranting the part-to-part gap, allows sufficient visibility of each stitch. It is assumed that the fixture keeps heat-related distortion within limits and stitches can thus be welded in an arbitrary sequence. Finally, *dimpling* is considered a special case of RLW, forming a series of small, punctual welds on the workpiece surface, and is not discussed separately.

Altogether, the *process planning* problem is stated as follows:

- *Given* the specification of the problem in terms of
  - the models of the workpiece, its fixture, and the workcell's boundaries and static elements,
  - the definition of a set of welding tasks to be accomplished in a single cycle, with an upper limit of the cycle time, and
  - the specification of the robot and the laser source,
- *find* the placement of the workpiece in the workcell, together with an appropriate offline robot program,
- *such that* the program is executable, accomplishes each welding task, and minimizes the cycle time.

An executable robot program has to meet various types of constraints: it should fit the kinematic model of the robot, comply with all the technological constraints of the RLW technology, allow the laser beam to hit the stitches only, and avoid the collision of the robot with any object within the workcell.

2.1 *Requirements*

Solving the above problem requires the application of a number of generic *engineering principles* and an *integrated workflow*. First, the complexity of the problem calls for *decomposing* it into relatively independent subproblems: one responsible for static configuration, and one for dynamic planning and offline programming of the workcell. Aside from being inherently coupled, the two subproblems and tracks of problem solving cannot be handled in a single phase. Instead, the final solution evolves through a *refinement* process that adds more and more solution details. To this end, a *unified representation* of the workcell is required, including all relevant elements and capturing both their structure and behaviour. In any phase, the solution of subproblems should be supported, as far as possible, by general purpose methods. This is especially the case for physical (geometrical) interaction of objects—to handle the latter, uniform triangular mesh representation in the Standard Tessellation Language (STL) is used for modelling all objects in 3D. The recurrent issues of distance calculation and collision check are tackled with the same, generic apparatus, relying on the Proximity Query Package (PQP) (Larsen *et al.* 2000). Likewise, the target performance criteria of minimizing cycle time require the application of advanced *optimization* methods. However, the integrated solution of all subproblems is far from being fully automatic, and, in fact, there is no need to aim for this either. Instead, a system is needed that guides engineers through a problem solving process, presenting feasible alternatives at the key decision points, and supporting the direct involvement of the engineer in approving, rejecting or modifying (intermediate) results of the configuration and planning process. Such decision support with *mixed-initiative problem solving* implies that (1) the system has to have an easy-to-use graphical user interface (GUI) appropriate for presenting and manipulating all main intermediate and final results of the problem solving process, and (2) the computational methods should respond fast enough for use in interactive scenarios.

2.2 *Models*

The process planning problem is captured by the following models.

2.2.1 Models of the workpiece and the fixture

An RLW operation consists in assembling a number of sheet metal parts to form a single assembly. It is assumed that all parts are loaded into the fixture, and are joined in a single pass. Neither the workpiece nor the fixture move during the entire operation (e.g., no active clamping is applied), and therefore, they can be modelled as static objects, represented by their mesh models. This geometrical model is required to ensure that contact between the laser beam and the workpiece occurs exactly where and how it is required for welding, while all other types of collisions are avoided.

2.2.2 Model of welding tasks

The RLW robot has to weld a finite number of disjoint, linear or circular stitches to join the parts. Each stitch must be welded without interruption, with the scanner centre point being in the *access volume* of the stitch, a truncated cone in the 3D space determined by the maximum inclination angle (deviation from the surface normal) and the focal length range of the robot—see also Figures 2 and 5 for further explanation. The laser power and the welding speed are determined for each stitch individually.

2.2.3 Model of the RLW robot

Although dynamic properties and control behaviour of the robot arm and actuated optical components of the scanner head differ considerably, the system is modelled on the kinematic level as a single kinematic chain. For this reason, the following conventions are declared (see also Figure 2):

- Additional degrees of freedom (DOF) introduced by the scanner head are treated as further kinematic DOF extending the manipulator's chain. Actuated mirrors add *revolute* DOF, while the adjustable focal length of the laser beam is treated as a single *prismatic* joint. Therefore, the beam can be regarded as an extension of the "tangible" components of the robot.
- The contact point of the beam and the workpiece is analogous to a tool-to-workpiece contact point, and it is convenient to consider it the *tool centre point* (TCP).
- Also of interest is a *scanner centre point* (SCP)—this will be needed for calculations regarding accessibility or collision, as well as solving the inverse kinematics. It suits best for a number of calculations if the SCP is identical to the origin of one of the

last link frames, e.g., the intersection of mirror axes if this exists. In the case shown here, the SCP lies at the origin of the last mirror-DOF, and the laser beam is then represented by a straight segment (of the focal length) between the SCP and the TCP.

- For technological reasons, the scanner head may be mounted on the robot with a non-zero offset in another direction than the robot's last revolute axis. This results in the last three revolute axes (laser mirror DOF included) not intersecting in a common point, hampering the "wrist decoupling" frequently used in closed-form inverse kinematic solutions. In fact, there may be no other closed-form solution at all, as in the case shown in the paper. While a disadvantageous scanner head offset is not a binding implication of RLW technology altogether, it does occur in practice. Calculations may then require a zero-offset "virtual scanner centre point", as shown in further parts of the paper—given its further roles, it is referred to here as *calibration point* (CP).

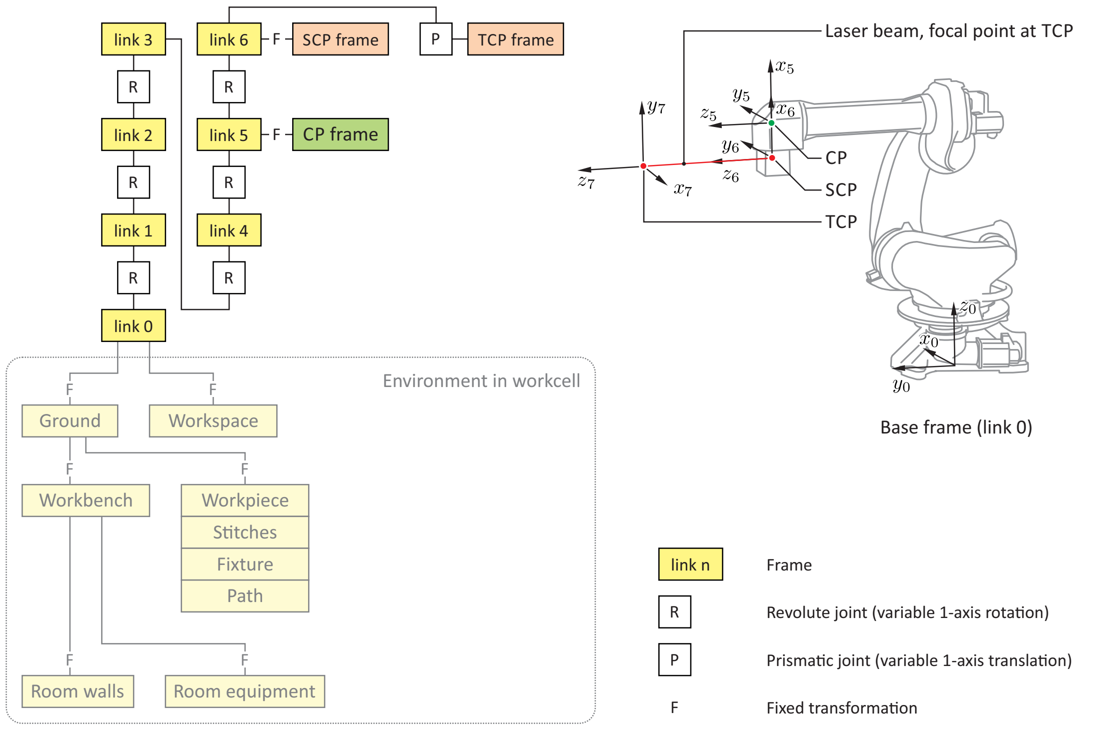


Figure 2. Linkage model of the robot and the workcell.

The welding robot in the case study (see Section 6.1) has the following kinematic DOF:

- *5 revolute DOF for the arm carrying the scanner head.*
- *1 revolute DOF added by the actuated mirror system in the scanner head.* As mentioned before, the mirror-DOF is attached to the carrying arm with a non-zero offset, not forming a single "wrist centre point". The mirror system has much less inertia than the arm links and is capable of much faster movements, nevertheless, its joint limits are much tighter.
- *1 prismatic DOF for the focal length.* Again, technological limits do not allow a large variation of the focal length, so that the joint limits of this DOF are also much closer than in a typical industrial manipulator.

2.2.4 Linkage model of the RLW workcell

In the research and its deployable implementation presented here, all kinematic and geometric modelling relies on the *linkage* structure which is capable of describing a wide range of mechanisms. A linkage is a graph whose nodes denote *links*, while edges are *constraints* between the links. Three kinds of constraints are considered: revolute and prismatic joints, as well as fixed transformation (see Figure 2). Two links and a constraint between them define together what is usually referred to as a *kinematic pair*. The linkage of the workcell defines an open kinematic chain. A linkage describes the following properties of a mechanism:

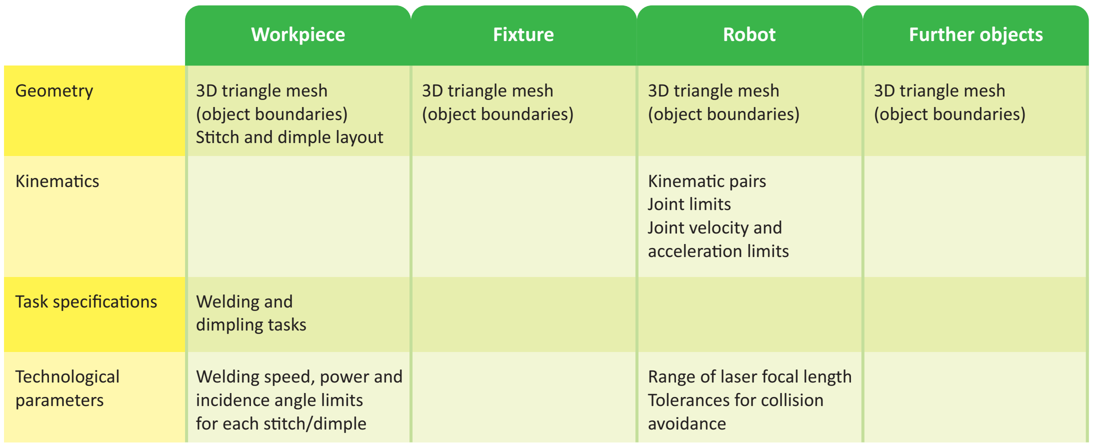
- Description of links:
  - link volume (via 3D triangle mesh);
  - local link reference frame;
  - link inertia parameters.
- Kinematic constraints between the links:
  - fixed transformations;
  - parametrised variable transformations representing the moving kinematic pairs.

Well noted, the *linkage* structure can accommodate more information than required directly for the welding robot—in fact, it can model a complete workcell including robot, workpiece, fixture, stationary equipment, feeder devices (e.g., turntable), and other technologically relevant volumes. Also, it is capable of modelling spatial relations (e.g., relative location within the workcell), as well as spatial and kinematic constraints (e.g., robot workspace boundaries for the former, and motion limits for various moving objects for the latter). Due to its versatility, the *linkage* can serve as the pivotal data structure for the entire planning process. Figure 2 presents the linkage model of the workcell: beyond the linkage mechanism of the actual 7-DOF robot used in the case study (highlighted), the overall linkage also represents the operating environment, the workspace of the robot, the workpiece and its fixture, the stitches, as well as the path of the robot.

In the case presented here, the linkage structure thus serves as a central repository supporting subsequent phases of the planning process—these, in turn, enrich the linkage with new information later planning steps will need. In other words, the linkage structure evolves and supports gradual refinement in the fashion of a *blackboard* that is shaped by a variety of planning components. In order to do this, the original *linkage* concept was generalised to accommodate all necessary information. Figure 3 summarises the objects modelled in the linkage and initial information associated with them.

It is important to distinguish between *representation* and *presentation* of the linkage, and clarify the relation of these in the planning process. The representation of the linkage is, by definition, the specific way in which objects, their relations, as well as task and solution data depict the given workcell, welding task and solution. The presentation of the linkage is a specific view that shows the entire structure (or its relevant subset only) in a way that best serves the activities (both calculations and human intervention) carried out in the given planning step. Therefore, different forms of presentation are used, always adapting to the requirements in the given step of the workflow.

The *presentation*—actually, a model in Virtual Reality Modelling Language (VRML)—is derived from the *representation* of the linkage. Nevertheless, a *presentation→representation* information flow can also occur if an operation (e.g., manual placement of objects) is carried out in the derived model of presentation, and the original representation is updated in accordance with these changes.



	Workpiece	Fixture	Robot	Further objects
Geometry	3D triangle mesh (object boundaries) Stitch and dimple layout	3D triangle mesh (object boundaries)	3D triangle mesh (object boundaries)	3D triangle mesh (object boundaries)
Kinematics			Kinematic pairs Joint limits Joint velocity and acceleration limits	
Task specifications	Welding and dimpling tasks			
Technological parameters	Welding speed, power and incidence angle limits for each stitch/dimple		Range of laser focal length Tolerances for collision avoidance	

Figure 3. Initial contents of the linkage model.

3. The workflow of planning

The RLW problem exposed in Section 2 involves two types of subproblems: (1) a *configuration* subproblem when one has to decide whether and in what setting the available technology is capable of performing the RLW tasks, and (2) a *planning* subproblem when the behaviour of the workcell is to be determined, all the way to offline robot programming. These static and dynamic aspects of the core problem are closely related to each other, and are, therefore, solved in an integrated *workflow* that augments and refines the linkage model of the workcell in a step-by-step manner. In this refinement-based solution process, configuration and planning decisions are made hand in hand, by taking more and more geometrical, kinematic and technological constraints into consideration.

The workflow starts with the initial task definition comprising the workpiece and fixture geometries, the specification of welding tasks as well as that of the workcell around the welding robot. By taking all these input data (see also Section 2.2), the initial model of the linkage is generated, as shown in Figure 3. Figure 4 presents the integrated workflow that results in the two main components of the final solution: a detailed configuration of the RLW cell, and the executable offline program of the welding robot. Main phases of the workflow are as follows:

(1) *Accessibility analysis* checks whether all the welding (or dimpling) tasks are accessible by the laser beam given the workpiece and fixture geometries, welding parameters and the capabilities of the RLW robot.
(2) *Task sequencing and path planning* generates a collision-free path for the scanner head with shortest possible cycle time.
(3) *Workpiece placement* is responsible for finding a posture of workpiece (embedded in its fixture) relative to the robot. No collisions may occur and the path of the scanner head has to be completely included in the working area of the robot.
(4) *Inverse kinematics* generates the *motion plan* for the joints of the robot, including the synchronized control of the laser beam.
(5) *Trajectory planning* adjusts the motion plan to the precise joint velocity and acceleration limits and generates the final path.
(6) *Collision detection* is performed against all elements of the workcell (including its boundaries) while the robot is moving along its prescribed path.
(7) *Offline programming* transforms the motion plan into an offline robot program

that is executable by the specific controller of the RLW robot at hand.

(8) *Workcell simulation* presents all components of the workcell linkage model (completely defined by this stage). The entire operation of the RLW robot is simulated.

(9) *Final evaluation* gives account of the key performance indicators. In this paper, it is cycle time, but energy demand, cost, etc. can also be considered here.

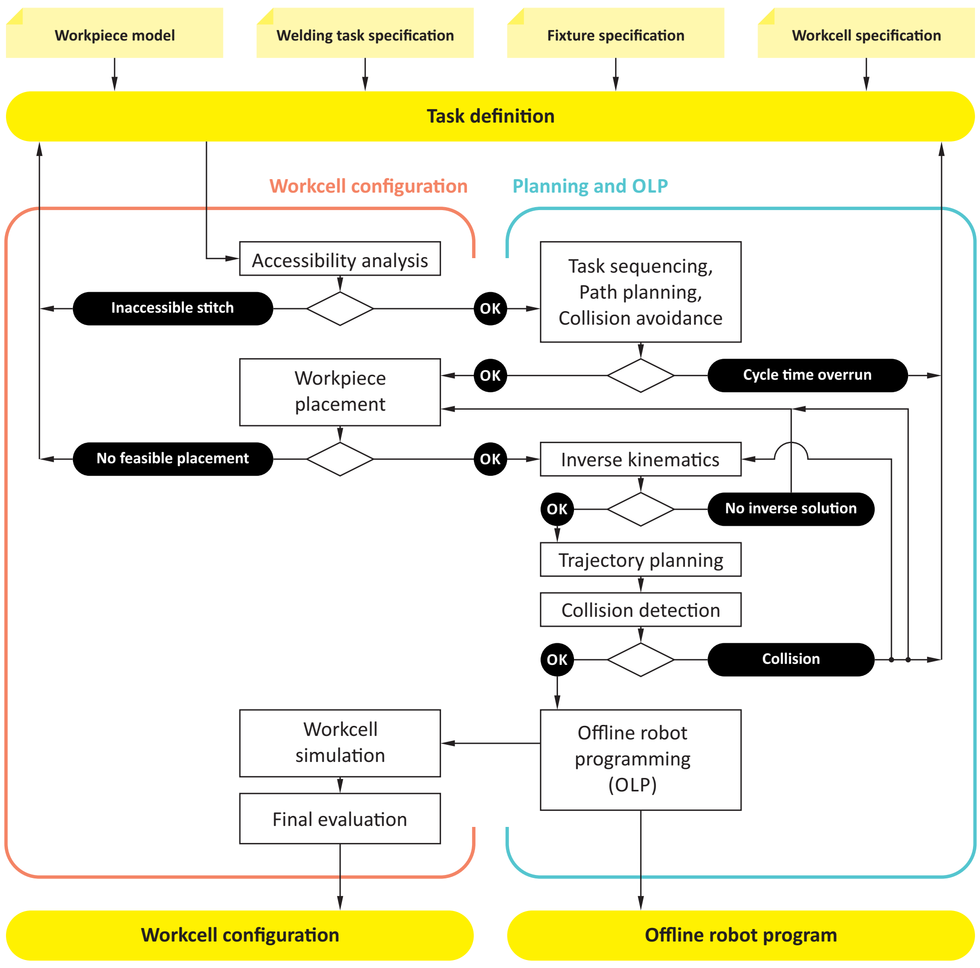


Figure 4. The integrated workflow.

The workflow has a number of *checkpoints* evaluating the *feasibility* of the interim solution generated that far (i.e., a partially specified linkage model of the workcell). Since the latter undergoes constant enrichment during subsequent workflow stages, it is always a *partial solution*, matched with constraints known up to the given checkpoint. Conflicts may thus surface at later stages, requiring *backtracking* to an earlier phase of the workflow (see Figure 4). Specifically, backtracking is needed in the following cases:

- Any of the welding tasks proves to be inaccessible given the workpiece and fixture geometries, welding parameters, as well as the capabilities of the RLW robot.
- Optimized task sequencing and path planning generates a scanner path whose cycle time exceeds the cycle time limit provided by system level design.
- No feasible placement of the workpiece can be found: some tasks cannot be performed

within the working area of the welding robot.

- Given a workpiece placement, no inverse kinematic solution can be generated for the scanner path.
- Upon simulated execution of the robot program generated by inverse kinematics, the robot collides with any element of the workcell.

Backtracking can also be initiated as an engineering decision to modify the partial solution generated so far. Finally, if no solution to the problem is found, the original task definition needs to be revised. The resolution of such conflicts goes beyond the scope of the workflow, by changing the input data for process parameters, product or system design, or the applicable RLW technology. Fixture (re)design is a typical point where several iterations outside the RLW process planning workflow are needed. The latter can, namely, reveal occluding fixture elements that hamper the welding process, requiring clamps, locators, and the corresponding stitches to be relocated. Finally, if iterations fail to simultaneously satisfy part-to-part gap control and accessibility constraints, the product design has to be modified.

4. Solution methods

Below, all main steps of the integrated planning workflow are presented, each employing techniques or considerations that are a new contribution to the state of the art.

4.1 *Accessibility analysis*

Process planning must start with the verification of the design to ensure the manufacturability of the workpiece using the selected technology and resources. In RLW, the key question is collision-free access to the workpiece fastened in the fixture. In this sense, process planning provides feedback to workpiece and fixture geometric design. Below, the concept of *accessibility* for RLW is defined, then a pragmatic method is presented for verifying the accessibility of the welding stitches.

4.1.1 Technological access volumes

Assume that a set of n stitches is given to be welded on the workpiece, denoted by $\{s_1, s_2, \ldots, s_n\}$. Each stitch s_i is characterised by its *surface normal* N_i and a *maximum inclination angle* φ_i, i.e., the angle between the laser beam and the surface normal when the stitch is welded. Furthermore, let C_i denote the geometrical centre point of the stitch, i.e., the midpoint of a linear stitch or the centre of a circular stitch. Finally, the welding time associated to stitch s_i is denoted by t_i. The robot is characterised by the maximum velocity of the scanner head, v (assumed to be independent of the position in the working area), and the range of the focal length of the laser beam, $\underline{f}$ and $\overline{f}$.

Given these preliminaries, the *technological access volume* (TAV) of a welding stitch is the region of the space from where the stitch can be welded, respecting the technological constraints on inclination angle and focal length. It is a truncated cone centred in C_i, with half opening angle φ_i, inner radius $\underline{f}$, and outer radius $\overline{f}$, as shown in Figure 5. Strictly speaking, this definition would require spherical outer and inner bases for the truncated cone. However, to benefit from convex TAVs, this shape is approximated with a planar inner base, while leaving the outer base spherical. Note that the definition exploits that the size of the stitch is an order of magnitude smaller than other characteristic dimensions in RLW technology, hence, it can be assumed that the whole stitch can be welded from where the centre point C_i can be welded (this assumption is *not* exploited in the definition of the collision-free access volume of the stitch).

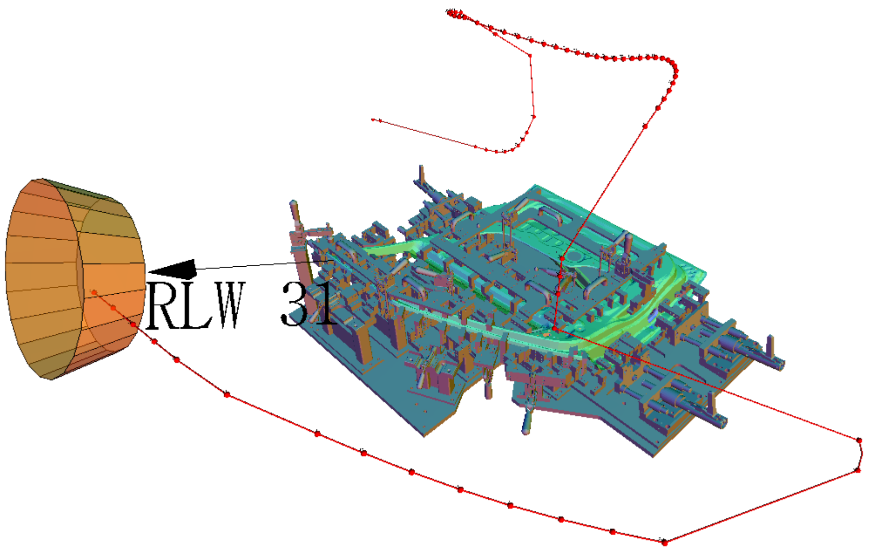


Figure 5. Technological access volume (TAV) of a welding stitch.

4.1.2 Collision-free access volumes

The *collision-free access volume* (CFAV) of a stitch is the subset of the TAV from where the stitch can be welded without collision. Below, collision is formally defined for welding and idle movement, focusing on the types of collisions that can be detected at this phase of the workflow, i.e., that are independent of decisions made in later phases (see Figure 4). These are the collisions between the *laser beam* vs. the *workpiece* and the *fixture*, as well as the *scanner head* vs. the *workpiece* and the *fixture*. It is noted that these are the most critical types of collisions in RLW.

The following definitions assume that welding can only be performed when the entire stitch is visible from the laser emission point, and therefore, ignore the theoretical possibility of certain sections of a stitch becoming gradually visible as the scanner head moves along its path. This assumption is common in stitch welding (Hatwig *et al.* 2012).

Collision detection must ensure that the specified minimum distance is maintained between the above pairs of objects while the robot moves along its *continuous path*. The set of tolerance parameters have been defined to provide this guarantee based on collision checks performed in an appropriately selected, finite set of *discrete positions*. For each pair of relevant objects, a *lower tolerance* and an *upper tolerance* distance is introduced, denoted by d_l and d_u, respectively, with $d_l < d_u$, as shown in Table 1. Collision checks in the selected positions are performed with a required minimum distance of d_u, which ensures that a minimum distance of d_l is maintained throughout the continuous path.

Collision detection is performed on a triangle mesh representation of the 3D objects involved, by executing PQP distance queries (Larsen *et al.* 2000) for the investigated pairs of objects. The mesh representation of the workpiece and the fixture are given as input, whereas the representation of the laser beam and the scanner head is constructed at runtime. Since the orientation of the *scanner head* is unknown at this phase of the

Parameters for collision detection	
d_l^S	Lower tolerance distance for the scanner head
d_u^S	Upper tolerance distance for the scanner head
d_l^L	Lower tolerance distance for the laser beam
d_u^L	Upper tolerance distance for the laser beam
e^L	Laser beam end truncation
r^S	Radius of the scanner head model

Table 1. Parameters for collision detection.

workflow, it is approximated by a circumscribed sphere. Technically, this is achieved using a mesh model that represents the scanner head as a single point P, and specifying $r^S + d_u^S$ as the distance threshold value in the PQP distance query.

The mesh model of the laser beam for welding a *linear stitch* consists of a single triangle, as shown in Figure 6, defined by the scanner head position (laser emission point), P, and the stitch start and end points, S_1 and S_2. To reflect that collisions between the very end of the laser beam and the workpiece are operational in welding, the height of the triangle is truncated by e^L when testing for collisions against the workpiece, which results in the light grey collision zone for the fixture and the dark grey zone for the workpiece. In case of a *circular stitch* with radius r, the mesh consists of a single line between the laser emitting point and the stitch centre point, leading to a narrow cylindrical volume that must be clear of any collisions. Finally, the collision-free volume for the idle movements of the robot, with the laser beam switched off, is denoted by CF_0.

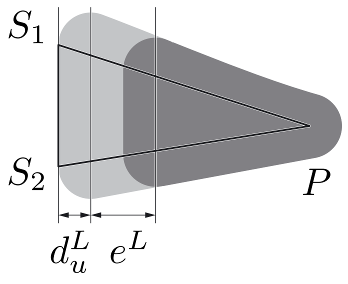


Figure 6. Mesh model of the laser beam for welding the linear stitch $\overline{S_1S_2}$ from robot position P. The approach results in the light grey collision zone for fixture, and the dark grey collision zone for both the fixture and the workpiece.

4.1.3 Accessibility ratios

In order to provide a simple quantitative measure of the accessibility of the welding stitches, the accessibility ratio of stitch s_i is defined as $r_i = \frac{|\mathrm{CFAV}_i|}{|\mathrm{TAV}_i|}$. This ratio is computed by sampling TAV_i on a rectangular grid, and verifying the collision-free accessibility of each sample point using the above methods. It is noted that separate accessibility measures can be computed for collisions with the scanner head and with the laser beam, which helps identify the reason of inaccessibility. Computational experiments have confirmed that the proposed techniques construct high quality robot paths if r_i exceeds 10% for every stitch. Otherwise, the geometric design, i.e., the fixture, the stitch layout, or, as a last resort, the workpiece itself must be modified to improve accessibility.

4.2 *Task sequencing and path planning*

The problems of welding task sequencing and path planning are strongly related: the robot path must conform to the computed sequence of the stitches, while evaluating a candidate stitch sequence is hardly possible without computing an associated robot path. The procedure proposed below solves the two problems in an integrated way. The method exploits that RLW does not suffer from the serious accessibility issues that characterize many other joining technologies, and that general guidelines for RLW require that the TAVs of the stitches are left clear. At the same time, experience with industrial data suggests that the above guidelines are sometimes overridden by other design objectives, potentially resulting in collisions along the path.

For this reason, a two-step approach is proposed, consisting of a simultaneous task sequencing and rough-cut path planning step that efficiently minimizes the cycle time while ignoring potential collisions, and a subsequent collision-free path planning step that fixes collisions while preserving the stitch sequence computed before. The proposed methods are illustrated in Figure 7, showing a colliding rough-cut path and the corresponding collision-free path for a sample workpiece. A detailed description of the proposed task sequencing and rough-cut path planning algorithm is given by Kovács (2013), whereas the algorithm for collision-free path planning is presented by Kovács (2014).

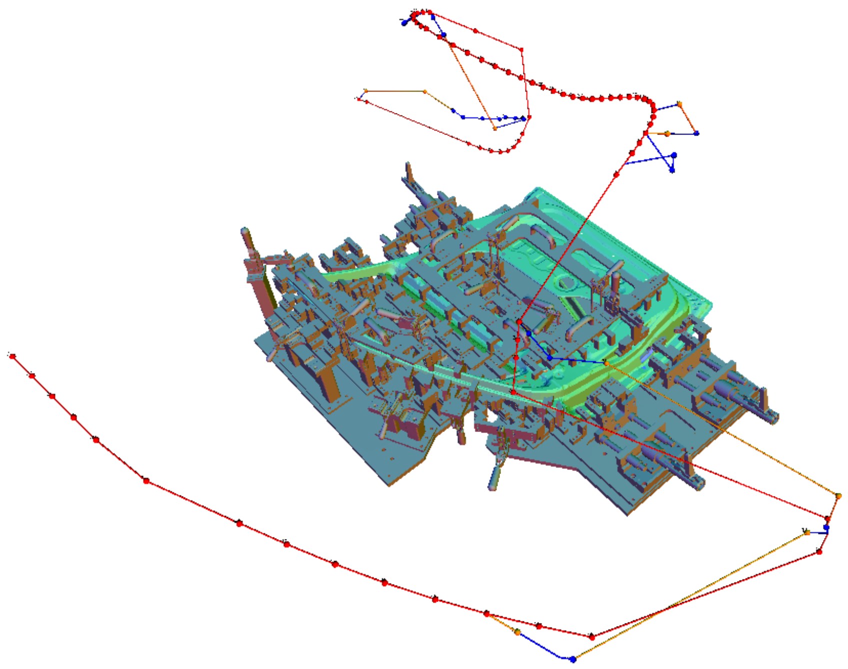


Figure 7. Comparison of the rough-cut and the collision-free paths. Blue sections denote welding, while yellow section correspond to idle movement.

4.2.1 Task sequencing and rough-cut path planning

The problem consists in determining a task sequence and a corresponding scanner centre point (SCP) path that welds all the stitches $\{s_1, \ldots, s_n\}$ and minimizes the cycle time. The following assumptions are exploited during task sequencing and rough-cut path planning (a part of these will be relaxed in subsequent steps of the workflow):

- 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;
- The scanner can position the laser beam in zero time;
- Any stitch sequence is feasible;
- Robot acceleration limits are disregarded at this phase of planning;
- Potential collisions are ignored at this phase of planning.

The SCP path is represented as a broken line, encoded in the form $((P_1, a_1), (P_2, a_2), \ldots, (P_k, a_k))$, where segment (P_i, a_i) denotes that the robot moves from point P_i to point P_{i+1} along a linear section while performing action a_i. Action a_i can be of two types: $a_i = (s_{[i]}, -)$ standing for idle movement after welding $s_{[i]}$, or $a_i = (s_{[i]}, +)$, denoting welding stitch $s_{[i]}$, where $s_{[i]}$ corresponds to one of the stitches $\{s_1, \ldots, s_n\}$. Obviously, for $(P_i, (s_{[i]}, +))$, P_i and P_{i+1} must be inside $\text{TAV}_{[i]}$, which also implies that the complete section $\overline{P_i P_{i+1}}$ is inside the convex $\text{TAV}_{[i]}$.

Since collisions are ignored and the TAVs are convex, an optimal rough-cut path contains exactly one segment for welding each stitch. For convenience, two welding segments are assumed to enclose exactly one idle movement segment $(P_i, (s_{[i]}, -))$, potentially with zero length, having a duration of $d(P_i, P_{i+1})/v$. Furthermore, it can be observed that there exists an optimal path such that for each welding segment $(P_i, (s_{[i]}, +))$, it holds that $d(P_i, P_{i+1}) \leq t_{[i]}v$, and motion between P_i and P_{i+1} takes exactly $t_{[i]}$ time. Hereafter, the search will be restricted to such paths.

The problem in scope corresponds to the direct product of a TSP (for optimizing the task sequence) and a path planning problem in the 3D space. For solving this problem, a randomized restart local search algorithm has been developed. The algorithm combines adaptations of classical search operators for TSP for modifying the task sequence, and a path planning heuristic that computes a close-to-optimal SCP path for each candidate task sequence. The algorithm terminates when it hits the defined time limit.

In each run of the randomized restart procedure, an initial solution is constructed using an adapted version of the farthest insertion heuristics (Johnson and McGeoch 1997). The algorithm inserts the stitches one-by-one into the sequence: in each iteration cycle, it considers the stitch that is the farthest from the current path, and inserts it into its locally best position, with a small random perturbation on the costs of insertion. This initial solution is improved using a hill climbing search with the so-called 2-opt (deleting two edges and re-connecting the tour) and or-opt (relocating a segment of the tour of maximal length k to another position, in forward or backward orientation) neighbourhoods (Johnson and McGeoch 1997), until it reaches a locally optimal solution that cannot be improved. In such a case, search is restarted with a new initial solution. Several filtering techniques have been implemented to eliminate members of these neighbourhoods that cannot improve the solution, see Kovács (2013).

The evaluation of a neighbour involves computing a new SCP path for the modified task sequence. An incremental algorithm is applied that departs from the path computed in the previous iteration, and adapts it to the changes performed by the neighbourhood function. The algorithm sweeps along the segments of the path for a fixed number of iterations, and adjusts a single corner point of the broken line at a time. For welding segments $(P_i, (s_{[i]}, +))$, the new position of the starting point P_i is the position inside $\text{TAV}_{[i]}$ with $d(P_i, P_{i+1}) \leq t_{[i]}v$ that minimizes the distance $d(P_i, P_{i-1})$. The new position of the next point, P_{i+1}, is computed in a similar way, exploiting the symmetry that it is the starting point of the same welding segment in the opposite direction.

4.2.2 Collision-free path planning

The rough-cut path computed above may contain collisions that must be detected and corrected by a collision-free path planning algorithm. The procedure presented here achieves this, while preserving the task sequence and minimizing the cycle time—however, the complexity of mesh model collision queries justifies its restriction to the final rough-cut path only. The procedure removes the colliding segments and their at most Nth degree neighbours from the rough-cut path. Resulting gaps must be bridged by a series of new, collision-free path segments that connect two given points, P_α and P_β, while welding stitches $s_{\{1\}}, \ldots, s_{\{m\}}$ in this sequence.

The state space for collision-free path planning is represented as a four-dimensional *collision map* of discrete vertices, with three spatial dimensions and one additional dimension for the action performed in the vertex. The map contains $(P, a = (s, +))$ as a vertex if and only if P is contained in the CFAV of stitch s, whereas $(P, a = (s, -))$ is contained in the map if $P \in \mathrm{CF}_0$. The points included in the map are the points of a discretized, rectangular 3D grid with a resolution of $\varrho = \min(d_u^S - d_l^S, d_u^L - d_l^L)$. The application of this resolution and collision checks in the grid points with tolerance d_u^S and d_u^L ensure that movement between two neighbouring grid points is collision-free with d_l^S and d_l^L (Kovács 2014). The map is created for a finite rectangular area, obtained by extending the envelope of the deleted segments of the rough-cut path in all directions, as requested by the input parameter B, the maximum by-pass parameter. Possible transitions between states are captured by directed arcs between the vertices, according to the following rules. Let $N(P)$ denote the 6-neighbourhood of point P in the 3D grid. From vertex $(P, (s_{\{i\}}, +))$, there are arcs to $(P', (s_{\{i\}}, +))$ with $P' \in N(P)$, i.e., continuing the welding operation in a neighbouring point, and to $(P, (s_{\{i\}}, -))$, i.e., finishing the current welding task and continuing with idle movement. From the vertex encoding $(s_{\{i\}}, -)$ in P, there are arcs to $(P', (s_{\{i\}}, -))$ with $P' \in N(P)$, i.e., continuing the idle movement, and to $(P, (s_{\{i+1\}}, +))$, i.e., welding the next stitch. To save computation time by omitting unnecessary collision checks, vertices of the collision map are generated on the fly, as they are explored by the search procedure. Moreover, the results of collision detection are inferred from the results for the neighbouring points whenever possible.

In order to compute a collision-free path, an A* search is performed on the above defined collision map. The search also maintains in each node the time spent welding the current stitch. The source node corresponds to starting the welding operation $(P_\alpha, s_{\{1\}})$, whereas the goal state is $(P_\beta, s_{\{m\}})$ with welding time $t_{\{m\}}$. A relaxed section at the beginning (or end) of the rough-cut path presents a special case, since here, the collision-free path can be started (finished) at any point of the corresponding CFAV. The cost function of the search is the time spent travelling the path, whereas the heuristic estimate of the remaining cost is $h(P, (s_{\{i\}}, \cdot)) = \max\left(r + \sum_{j=i+1}^{m} t_{\{j\}}, \frac{d(P, P_\beta)}{v}\right)$, where r is the remaining welding time of stitch $s_{[i]}$. The first term encodes the total remaining welding time, while the second term is the time for travelling from the current location to the goal P_β. The second term is ignored for multiple goal locations. The search terminates with an optimal collision-free source-to-goal path over neighbouring grid points.

The path computed by the A^* search consists of small, axial sections in the Cartesian coordinate system. This path is smoothed by an algorithm that considers the corner points of the path one-by-one, and eliminates the corner points whenever this keeps the path collision-free. The procedure is illustrated in Figure 8. Finally, the smoothed collision-free path segments are inserted at the place of the original, colliding path segments, and the cycle time is re-calculated.

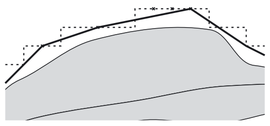


Figure 8. Comparison of the initial (dotted) and the smoothed (solid) collision-free paths.

4.3 *Workpiece placement*

The relative placement of welding robot and workpiece has significant impact on the feasibility of the welding task, i.e., whether—and how well—the welding robot is able to trace over all specified stitches or dimples within its operating area. Two groups of aspects can be identified and associated with different stages of the planning workflow:

- *Rough placement* satisfies two groups of constraints, i.e., (1) inclusion of all task points $\{P_i\}$ in the accessible volume, and (2) absence of collisions of the bodies involved (including laser beam) for all $\{P_i\}$. If any of the constraints cannot be satisfied simultaneously for all specified points $\{P_i\}$ of the path, one may either modify the task (e.g., re-allocating points or changing design parameters), or partition it to subtasks, each having a different suitable placement.
- *Fine placement* optimises some—typically continuous—performance-related measure, mostly for an entire path (of which $\{P_i\}$ are a true subset).

The planning workflow concentrates on *rough placement* for a number of reasons:

- Carrying out the welding task in one pass—if possible, within a given cycle time—is of utmost priority. Non-compliance with this requirement calls for mandatory redesign. A clear go / no-go answer is, therefore, of highest importance.
- A feasible placement of the workpiece needs to be determined before an entire joint trajectory is generated, leaving the placement step without information on robot configuration (left-handed vs. right-handed), or paths connecting the inverse kinematic solutions for the discrete SCP points. Fine placement is thus impractical at this stage.
- In its current form, placement is elaborated manually by moving the workpiece under visual feedback of path point feasibility. This requires short response times (and a quickly calculated measure), suggesting, again, easy-to-comprehend predicates associated with rough placement.
- Influencing performance measures only, fine placement has much less of an impact on the success of the planning procedure.

In this specific workflow, no inverse solution is known yet when placement feasibility is examined. Therefore, including robot-to-obstacle collision tests into the placement evaluation has little justification, leaving inclusion of path points into reachable workspace—in essence a series of point-to-body collision checks—the only type of test to be performed.

In the case dealt with here, prescribed SCP points have to lie in the (dexterous) workspace of the SCP, which may differ for left-handed and right-handed configuration (not yet specified at this point). A computationally "lean" approximation is going back to a frame in the kinematic chain where different configuration pairs have no effect on the reachable workspace, and employing a combination of optimistic and pessimistic estimations as described below. For the robot in this paper, the frame of a fictitious zero-offset scanner head has such an origin, coinciding here with the calibration point (CP, see Section 2). It is also known that the true scanner centre point (SCP) is always a fixed offset away from the CP in some direction. Therefore, two sweep volumes can be defined based on manufacturer-supplied workspace data (see also Figure 9):

- *Pessimistic workspace approximation*—this is the manufacturer-defined workspace uniformly shrunk by the scanner head offset.
- *Optimistic workspace approximation*—this is the manufacturer-defined workspace uniformly inflated by the scanner head offset.

Based on inclusion in these volumes, a discretised (3-step) predicate holds for all P_i:

- P_i is certainly outside the feasible volume if it lies outside the optimistic boundary,
- P_i is conditionally feasible if it lies between optimistic and pessimistic boundaries, and
- P_i is certain to be feasible if it lies inside the pessimistic boundaries.

Note that this check is only observing the inclusion of P_i in the reachable workspace—known body collisions are to be checked additionally for the same position of P_i. If a collision is detected, the predicate *infeasible* must be issued.

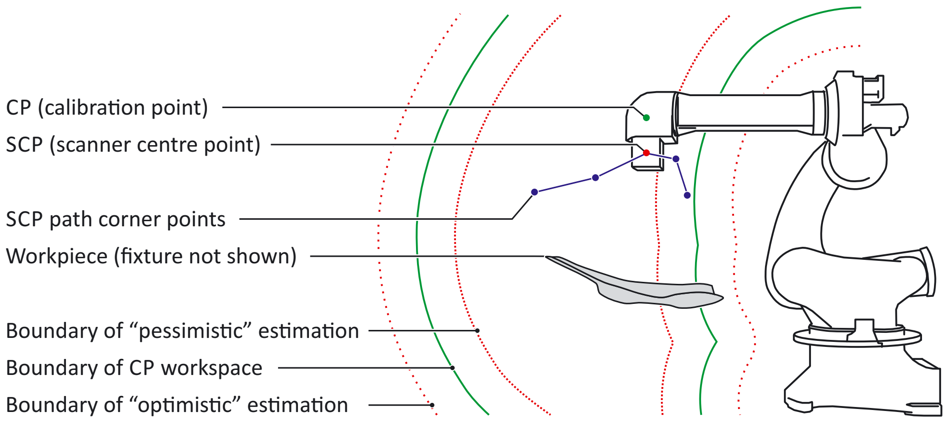

CP (calibration point)
SCP (scanner centre point)
SCP path corner points
Workpiece (fixture not shown)
Boundary of "pessimistic" estimation
Boundary of CP workspace
Boundary of "optimistic" estimation

Figure 9. Workspace boundaries for the CP, and optimistic / pessimistic boundaries for a simplified approximation of SCP feasibility.

4.4 *Inverse kinematics*

Once an SCP path has been generated, a corresponding series of—unambiguously specified—robot joint vectors is calculated, along with laser control values for each point. Referred to commonly as inverse kinematics, this—usually straightforward—task introduces a number of specific issues that have to be addressed in the case shown here:

- The entire mechanism is *kinematically redundant*, due to the addition of two laser-DOF (mirror tilt and focal length) to the 5 manipulator-DOF. Given the analogy of laser beam pose and 5-axis machining, the case shown here involves two redundant DOF.
- *Joint limits* at the laser-DOF are much closer together than it is typical for the kinematic DOF of a robot arm. This is most restrictive for the focal length of the welding beam, and values of the laser-DOF are best kept mid-range.
- A *closed-form inverse* is not possible due to kinematic properties (the laser mirror offset in particular), and an *iterative* solution routine has to be used instead.

Designing the motion plan properly, the first two issues can cancel each other out by using the focal length prescribed by the SCP path and the mid-range value for the last mirror angle. The mirror offset, however, poses a further, more intricate challenge: its unfavourable direction does not allow the use of a closed-form inverse solution for the remaining mechanism. It is, nevertheless, possible to combine closed-form and iterative elements instead of a fully iterative solution for the entire robot, and cut down computational demands by reducing the number of search space dimensions.

The clue to this is a fictitious robot with zero mirror offset which does allow a closed-form inverse. Now, compare the zero offset case with the actual robot, as shown in Figure 10. Recall that the last revolute mirror-DOF is fixed at midrange—in this case, the laser beam of the actual robot (marked with red in the figure) and the mirror offset (d_6) are perpendicular to each other. Also, it is easy to see that the beam of the fictitious zero-offset robot (marked green in the figure) is always at a distance of d_6 from the actual beam. Knowing this, the search space of the iterative inverse calculation can be narrowed down to a circle of radius d_6 around the SCP.

Having decoupled the last prismatic DOF, the procedure for the SCP is described at this time. Consider a prescribed SCP position and orientation, i.e., it is known where the SCP is required to lie, and in what direction the laser beam should exit. Knowing the direction of the non-zero mirror offset, the following can be stated about a fictitious zero-offset mirror centre point: (1) it is located in the plane which contains the actual SCP and is perpendicular to the laser beam, and (2) its distance from the actual SCP equals d_6. Having specified the SCP position and the aforementioned plane (via the orientation of the laser beam), it is certain that the zero-offset mirror centre point has to lie on a circle of known radius drawn around the prescribed location of the SCP. Now, a one-dimensional iterative search can proceed as follows:

(1) Given the prescribed SCP position, laser beam orientation and mirror offset, specify the circle where the zero-offset mirror centre has to lie.
(2) Trace this circle with the zero-offset mirror centre using a closed-form inverse solution. (In the case shown here, the zero-offset mirror centre coincides with the CP of the manipulator, therefore, existing kinematic models do not need to be extended for the purpose of inverse kinematic calculations.)
(3) Calculate forward kinematics with the joint values obtained in the previous step and observe where the actual SCP would lie.
(4) The solution for the actual SCP is found when the above forward solution is identical to the prescribed SCP position.

Closed-form inverse solutions usually deliver a finite number of solution branches—is this the case, steps 2 to 4 have to be repeated for each solution branch separately. It should be noted, however, that inverse solutions within a motion sequence are preferred to minimise transients and would stick to the same solution branch for longer intervals.

The symbolic closed-form solution for the remaining manipulator DOF was obtained semi-automatically, relying on manual selection of *template equations* from an automatically generated set of equations for the specific mechanism(Paul *et al.* 1981). Here, the fact is exploited that the kinematic equations of industrial manipulators can be represented by trigonometric polynomials that boil down to a limited pattern set. Expressing one joint variable after the other consists in symbolic steps which the user can manually select from an automatically generated set of matching template equations in alternating phases of human intervention and automatic processing (Müller *et al.* 2004). The use of the template equation technique resembles a decision tree with the original kinematic equations being at the root, automatic steps represented by edges, nodes standing for human decisions, and complete symbolic solutions being located at leaves. At points where different solution branches enter (e.g., left-handed or right-handed, elbow-up or elbow-down), a pair of template sets has to be selected, each standing for its own solution branch. Having obtained symbolic solutions, numeric deployment of inverse calculations can proceed without further symbolic operation (i.e., no additional symbolic processing is needed during routine inverse calculation at run-time).

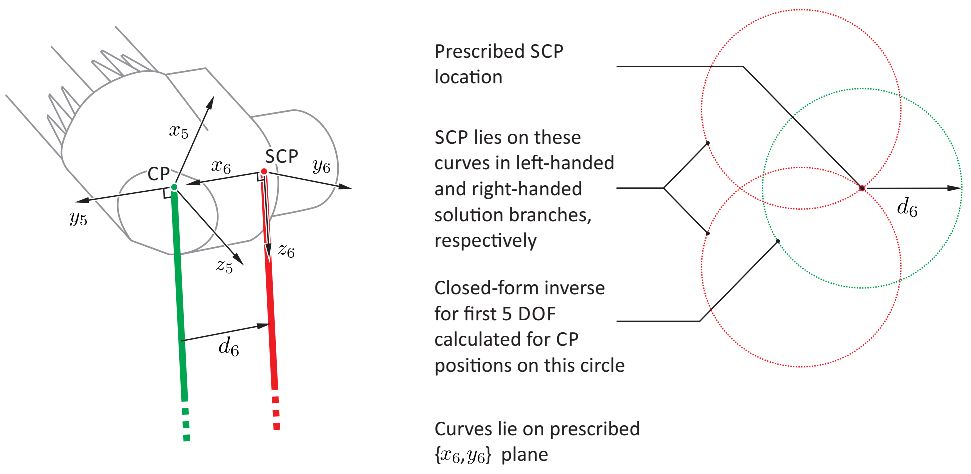

x_5
CP
x_6
SCP
y_6
y_5
z_5
z_6
d_6
Prescribed SCP location
SCP lies on these curves in left-handed and right-handed solution branches, respectively
Closed-form inverse for first 5 DOF calculated for CP positions on this circle
Curves lie on prescribed $\{x_6, y_6\}$ plane
d_6

Figure 10. Combination of closed-form and iterative methods for inverse kinematics calculation—this inverse calculation is made possible by fixing the last mirror-DOF so that the mirror offset and the laser beam are perpendicular to each other.

4.5 *Trajectory planning and offline robot programming*

The last two planning workflow phases transform the motion plan generated for the robot (including mirrors and laser beam) into a form executable by a specific RLW robot.

4.5.1 Trajectory planning

The motion plan calculated by inverse kinematics defines the joint paths where each joint variable is assigned a parametrised curve. In its initial form, the latter still ignores joint velocity and acceleration limits. The goal of trajectory planning is to find a suitable *re-parametrisation* of the motion plan to comply with these limitations.

Trajectory planning is specified with the following input constraints:

- welding speed of the TCP;
- maximum joint velocities;
- maximum joint accelerations.

In order to calculate the trajectory, the movement of every joint axis should be determined as a scalar function of time. Velocity and acceleration constraints imply a *trapezoidal velocity profile* limited (1) by the maximum joint velocities for idle movement segments, and (2) by the combination of the prescribed welding speed and the joint velocity limits for the welding segments. Having determined the joint velocity profiles, the *time* versus *TCP position* function can be calculated using the direct kinematic mapping of the robot.

The complete tool path of the TCP is calculated as follows. The velocity profiles $v(t) : \mathbb{R} \to \mathbb{R}^4 \times \mathbb{R}^4$ of the TCP tool path are known. The time parametrisation of the TCP tool path $p(t) : \mathbb{R} \to \mathbb{R}^4 \times \mathbb{R}^4$ is defined by the integral function of the velocity profile $v(t)$ function, and is calculated numerically by evaluating the t_i, P_i ordered pairs where $t_i \in \mathbb{R}, P_i \in \mathbb{R}^4 \times \mathbb{R}^4$. Calculating the inverse kinematic solution to every P_i, the $(t_i, q_{1,i}), (t_i, q_{2,i}), \ldots, (t_i, q_{7,i})$ ordered pairs are determined, from which the time parametrised joint functions are calculated using interpolation functions.

4.5.2 Offline robot programming

Having the inverse kinematics and the corresponding trajectory calculated enables moving to the next step of the workflow which is the generation of the offline robot program. The robot program is a sequence of commands written in a programming language provided by the robot controller. After analysing the structure of robot programming languages, a pattern-based code generation method was designed and implemented which uses an abstract representation of the robot program (see Figure 11 for its class structure).

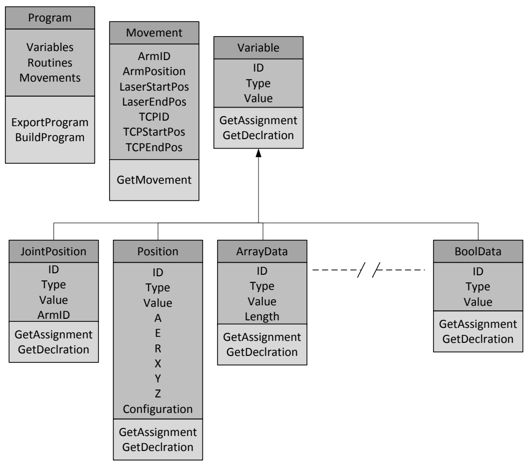


Figure 11. The class structure defined for generating the robot program automatically.

Each item in the class structure can be extended with methods for describing it in a particular programming language. An abstract program is symbolised by an instance of the *Program* class, which is a container for holding all variables, routines and *movements* used during the generation of the offline program. While variables and routines are only wrappers, a movement is designed to represent the smallest element of the trajectory (e.g., a stitch) and by doing so it has to describe the position of the destination; the ID of the robot arm; the speed of the TCP of the arm along the movement; the type of the applied interpolation; and the laser beam activity. All these data are extracted from the calculated trajectory. As the calculated inverse kinematics and its corresponding trajectory uses an arbitrary (controller-specific) arm decomposition during the calculation, the abstract representation of the robot movements has to apply the same arm decomposition. During the operation of the robot, movements of the two virtually decomposed arms are executed in a synchronised way.

5. Implementation

The workflow was implemented in a configurator and planner system that relies on the services of a toolbox of software modules dedicated to solving the subproblems. The main system design principles were as follows:

- Provide support for executing every step of the workflow.
- Allow the application of external software like solver modules, technical computation or proximity query packages.
- Support further extension of the system with additional modules.
- Use graphical environment for interactive, mixed-initiative problem solving and simulation.

The software toolbox was developed in Microsoft .NET using Visual C# which offers a wide range of tools for creating a graphical interface and facilitates the integration of different modules. The application guides the user through the workflow, allowing adjustment of all parameters and features of modules that can apply to the given context.

The entire workflow relies on a linkage structure created and manipulated by *LinkageDesigner*, an application package of Mathematica (see Erdős 2011), accessible via a .NET interface (*NETLink*). Being part of a symbolic computation environment, *LinkageDesigner* enables creating and maintaining a parametrised and extendible linkage model of the workcell. Moreover, it can also generate a VRML representation of linkages. The exported VRML file (transferred via file interface) is displayed in the software toolbox by a custom-tailored VRML viewer module, complying with the structure of the linkage. Owing to a one-to-one correspondence of VRML and linkage models, elements of the workcell subject to collision test or simulation can also be selected in the graphical presentation. This feature makes the system easy to use in interactive scenarios.

Accessibility analysis, task sequencing and path planning, as well as offline programming are implemented as standalone C++ applications. Proximity query and collision test are performed by the *PQP* package that also supports finding a feasible manual solution of the workpiece placement problem. Inverse kinematics and trajectory planning are solved using *LinkageDesigner*. Finally, simulation of robot operation as it moves along its path, from making one weld to another, uses the VRML model of the workcell. At any moment, collision can be tested against any object in the workcell.

6. Experiments

The methods have been verified and validated in two kinds of experiments: the complete workflow has been applied in a physical demonstrator case study, while the task sequencing and path planning algorithms have been tested in extensive comparative computational experiments.

6.1 *Case study*

The complete workflow and the methods developed have been tested in a physical demonstrator case study for the *automotive industry*. The study is aimed at replacing traditional RSW technology by RLW in the assembly process of a car door. Since the required changes in product, as well as issues of fixture design, are not in the scope of this paper, the description below addresses the problems of workcell configuration and planning only. Figure 1 shows a representation of this problem in the implemented system. After performing some preprocessing, planning started with the following *input data*:

- CAD models of the assembly (i.e., the components of the door), in STL format;
- CAD models of the fixture components, in STL format;
- the welding task specification (71 stitches), in a predefined XML format;
- the layout of the workcell, including CAD models of the cell components (i.e., walls, power source, robot base, chiller), in STL format, and
- kinematic and CAD model of a Comau Smartlaser C4G robot, as a linkage.

The linkage model and workspace of this 7DOF robot have been presented earlier, in Figures 2 and 9, respectively. In the experiments, a 4 kW laser beam with a focus range of 1000.5–1143 mm was used. Executable code had to be generated in the PDL2 language of the robot controller.

The accessibility analysis of the initial stitch layout and fixture revealed the infeasibility of the original problem specification. Iterative modification of both fixture and product design finally resulted in a specification where each stitch was accessible. After successful path planning, a feasible placement for the workpiece and the fixture was sought. The workpiece was placed so that the complete welding (SCP) path was included in the workspace of the robot.

Finally, after calibrating the experimental setup and generating the inverse kinematics, the automatically generated RLW programs have also been successfully executed both for dimpling and welding the specified stitches on the test door. Figure 12 shows a simulated and a corresponding real welding situation. Physical testing of identical car doors welded using this robot program will soon complete the experiments.

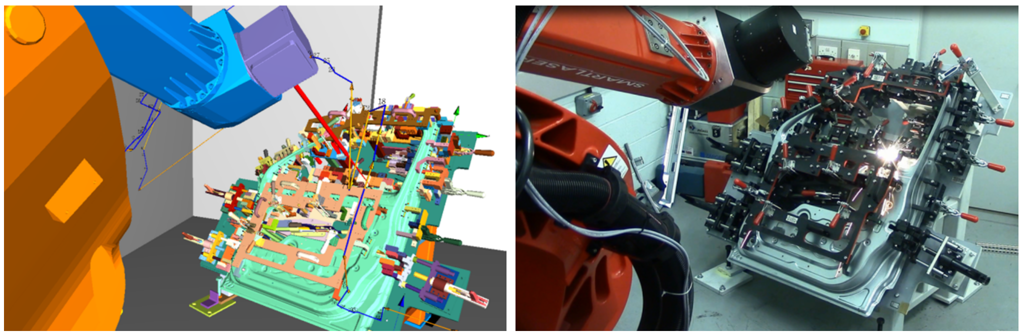


Figure 12. Snapshots of the simulated and the physical RLW process.

6.2 *Comparative computational experiments*

Computational experiments have been performed to compare the algorithms proposed above for task sequencing and path planning to the single algorithm for the same purpose in the literature (Reinhart *et al.* 2008). The problem instances involved the assembly of car doors using RLW. The set contained a single door geometry with different stitch layouts and realistic technological parameters, as well as various fixture designs, as these evolved during the early stages of the procedure sketched above. The stitch layouts consisted of 28–71 welding stitches. The mesh model of the door geometry included ca. 10^5 triangles, while the fixture model contained $5 \cdot 10^5$ triangles.

The experiments included computing a task sequence and a rough-cut path by three different algorithms for each instance, and converting each solution to a collision-free path by the algorithm presented in Section 4.2.2. The three sequencing algorithms were:

- INTEGR, the algorithm proposed above for integrated task sequencing and rough-cut path planning.
- TSP[TCP], the single sequencing algorithm dedicated to RLW from the literature (Reinhart *et al.* 2008), which solves a TSP over the stitch positions. Hence, this algorithm minimizes the length of the *TCP path.*
- TSP[SCP], a modified version of TSP[TCP] that solves a TSP over the TAV midpoints, instead of the stitch position. This way, the algorithm addresses the minimization of the *SCP path* length.

All algorithms have been implemented in C++, the latter two algorithms using ILOG CP as a TSP solver. The experiments were run on a 2.66 GHz Intel Core 2 Duo computer.

The proposed algorithms computed a feasible, collision-free robot path for every instance with all three task sequencing methods. The detailed comparison of the three algorithms is presented in Table 2, where each row stands for a separate problem instance. Instance names beginning with W and WF refer to welding without fixture and with fixture, respectively. Column n contains the number of stitches, while *min. accessibility* and *avg. accessibility* present the minimum and average accessibility ratio, r_i, measured over individual stitches in percent. For each algorithm, columns *cycle1* and *cycle2* contain the cycle times of the rough-cut path and the collision-free path. The best cycle times are shown in bold for each instance. Columns *run* contain the run time of the algorithm in seconds. It is noted that for TSP[TCP] and TSP[SCP], the TSP solver terminated with a locally optimal sequence in less than 1 second, hence, *run* is practically the time required for collision avoidance. In contrast, INTEGR was run for 120 seconds on each instance, plus the time of collision avoidance. These response times comply with industrial requirements, and enable the use of the algorithm in an iterative configuration and planning process.

The results show a notable difference among the instances depending on stitch accessibility. For the WF instances, accessibility was very poor (minimum accessibility around 10%, average accessibility of 60–70%). For the W instances, collision avoidance was run with the workpiece geometry only, resulting in 24–50% minimal, and around 90% average accessibility. Being optimized for RSW, the door design required a complicated fixture to ensure gap control—this can be accounted for poor accessibility in the WF case. Several instances had to be pre-processed to eliminate inaccessible stitches, since otherwise, the path planning problem would have no feasible solution. With all these findings, it is expected that a production car door would pose a task between the W and WF instances regarding stitch accessibility and collision avoidance complexity.

Regarding algorithm performance, INTEGR reduced cycle times drastically compared to TSP[TCP]. The reduction was on average 63% on the rough-cut path, and 61% on the collision-free path. This was mostly due to the joint consideration of TCP and SCP movement, instead of optimizing the TCP path only. For workpieces with complex geometry, TSP[TCP] leads to moving the scanner head in a zigzag above stitches that have nearby positions but different surface normals, as illustrated in Figure 13. In case of a car door, this phenomenon is most evident around the window frame, where the stitches on the inner and the outer sides are close to each other, but must be welded from the opposite sides of the robot working area.

INTEGR also outperformed TSP[SCP] regarding the cycle time of the rough-cut path

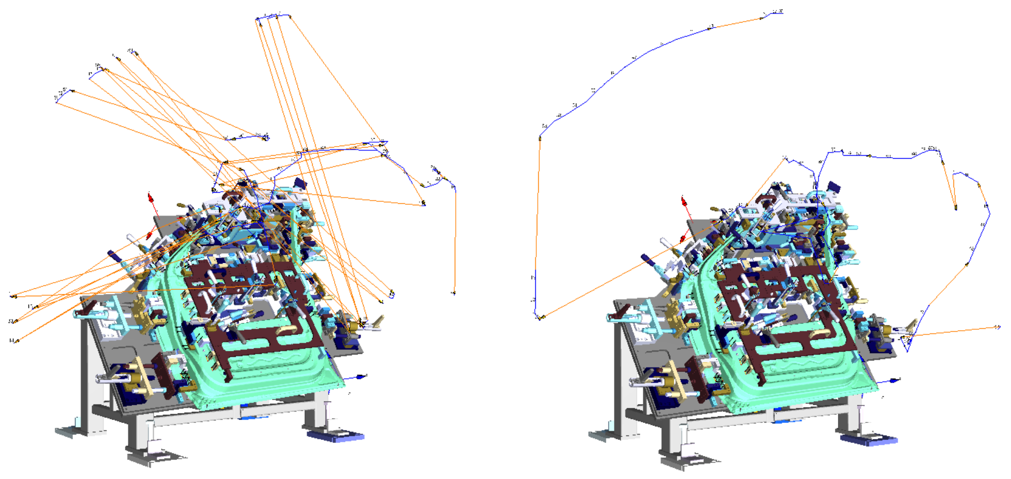


Figure 13. Comparison of the paths computed by the TSP[TCP] (left) and the proposed INTEGR (right) methods. TSP[TCP] focuses merely on the TSP path, while INTEGR is able to consider the path of the scanner head as well, resulting in significantly shorter SCP paths.

on every instance, by computing up to 6.1%, on average 2.9%, more efficient paths. However, this did not automatically translate to improvement on the collision-free path for each individual instance. The perturbation of the rough-cut paths by collision avoidance resulted in a situation where INTEGR computed better collision-free paths for 11 out of 15 instances, by up to 4.3%. However, TSP[SCP] outperformed INTEGR in 4 of the 15 cases, by 2.1–4.9% for different instances. This typically occurred for the WF instances with the worst accessibility. Beyond the random perturbation caused by the modifications to the rough-cut path, a possible explanation of this phenomenon comes from differences in the underlying assumptions made by the algorithms for sequencing. Implicitly, TSP[SCP] assumes that each stitch is welded from the mid-point of the technological access volume, whereas INTEGR assumes that the complete technological access volume can be used. In these problematic instances, the assumption of TSP[SCP] appears to be closer to reality. Initial experiments on sequencing using reduced TAVs confirm this hypothesis, and with an appropriate choice of parameters, the method resulted in INTEGR outperforming TSP[SCP] for all instances. Adequate heuristics are subject to future work.

		Accessibility		TSP[TCP]			TSP[SCP]			INTEGR		
	n	min.	avg.	cycle1	cycle2	run	cycle1	cycle2	run	cycle1	cycle2	run
W1	28	47.32	91.63	30.05	30.53	22	14.01	14.01	7	**13.69**	**13.69**	128
W2	34	47.32	95.16	35.50	35.50	2	15.93	15.93	2	**15.48**	**15.49**	135
W3	62	49.29	93.61	76.39	76.64	11	26.91	26.91	2	**26.11**	**26.11**	122
W4	44	34.38	87.21	56.33	57.23	19	19.55	19.84	8	**18.36**	**18.98**	146
W5	71	24.46	90.76	78.64	78.64	3	30.29	30.29	3	**29.85**	**29.85**	123
W6	67	24.46	90.84	67.70	67.70	3	28.50	28.50	3	**27.75**	**27.75**	123
WF1	28	10.97	64.21	30.05	31.26	229	14.01	15.04	113	**13.69**	**14.69**	299
WF2	34	14.63	69.49	35.50	35.86	286	15.93	16.92	192	**15.48**	**16.42**	337
WF3	62	11.14	68.49	76.39	78.42	294	26.91	**27.51**	219	**26.11**	28.08	353
WF4	44	10.89	58.81	56.33	58.23	163	19.55	21.16	196	**18.37**	**21.02**	283
WF5	64	9.79	65.03	75.37	77.18	270	26.15	**27.58**	249	**26.10**	28.93	448
WF6	63	9.79	63.55	74.92	76.40	399	25.81	**27.25**	365	**25.07**	27.91	404
WF7	63	6.90	60.98	74.92	76.59	504	25.81	**27.38**	334	**25.07**	27.95	421
Avg.	51	21.04	72.18	59.08	60.01	170	22.26	22.95	130	**21.63**	**22.84**	256

Table 2. Comparison of the three different sequencing algorithms.

7. Conclusions and future work

The paper presented a novel approach for generating offline programs for RLW robots with the objective of minimizing the cycle time. The backbone of the approach—and one of the key contributions in the solution—is a unified, linkage-based representation of all relevant objects and actors in the RLW workcell. The integrated workflow built upon this structure supports workcell configuration and OLP, the application of generic collision detection methods, and the simulation of the operation of the workcell behind a single user interface, presenting a consistent view of the evolving solution to the user whose supervision is a vital part in the mixed-initiative solution process. The unified nature of the approach still allows reasonable decomposition into subproblems, yet, it poses no hindrance to tighter combination of related subproblems—this is the key to the effective deployment of several novelties, integrated task sequencing and path planning being the most important achievement. Another key contribution is motion planning optimised for the scanner centre point (as opposed to earlier approaches planning for the laser-to-workpiece contact point). Thanks to this new approach, the cycle time of the robot program could be considerably improved in comparison to both state-of-the-art planning algorithms and the industrial practice. The automatic calculation of inverse kinematics and generation of the executable robot program alleviates the heavy burden of programming and manual teaching characteristic to current practice. A detailed case study in a real industrial setting, involving the assembly of a car door, demonstrated the feasibility of the approach.

The linkage model of the workcell is open to include any other equipment that is able to move, such as a turntable or feeder for loading and holding the workpiece together with its fixture. A possible point for further research is the coordination of interaction between the fixture design and process planning workflows. An automatic solution of the workpiece placement problem could facilitate this process, especially in cases when the solution space defined by the various technological, geometrical and kinematic constraints is very tight. Finally, the extension of the method to welding body-in-white will also be considered. This problem not only involves more complex geometries and collision tests, but also the cooperation of multiple welding robots. Notwithstanding, both the core linkage model and a number of elements of the solution process can be transferred from manufacturing to other domains where visibility is to be warranted while moving equipment along a complex three-dimensional path.

Acknowledgements

This work has been supported by the EU FP7 grant RLW Navigator No. 285051, and the NFÜ grant ED-13-2-2013-0002. The authors are grateful to Dariusz Ceglarek and Pasquale Franciosa (WMG, University of Warwick), Charles Marine (Stadco), Luca Bolognese (Comau) and Anil Mistry (Jaguar Land Rover) for the helpful discussions.

References

Alatartsev, S., *et al.*, 2013. On optimizing a sequence of robotic tasks. *In*: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2013)*, 217–223.

Castelino, K., D'Souza, R., and Wright, P.K., 2002. Toolpath optimization for minimizing airtime during machining. *Journal of Manufacturing Systems*, 22 (3), 173–180.

Ceglarek, D., *et al.*, 2004. Time-based competition in manufacturing: stream-of-variation analysis (SOVA) methodology – review. *International Journal of Flexible Manufacturing Systems*, 16 (1), 11–44.

Dewil, R., Vansteenwegen, P., and Cattrysse, D., 2014. Construction heuristics for generating tool paths for laser cutters. *International Journal of Production Research*, in print.

dos Santos, R.R., Steffen Jr, V., and de Fátima Pereira Saramago, S., 2010. Optimal task placement of a serial robot manipulator for manipulability and mechanical power optimization. *Intelligent Information Management*, 2 (9).

Erdős, G., 2011. Linkage Designer 2.0, `http://www.wolfram.com/products/applications/linkagedesigner/`. [online] [30 March, 2014].

Erdős, G., *et al.*, 2013. Planning of remote laser welding processes. *Procedia CIRP*, 7, 222–227.

Franciosa, P., *et al.*, 2014. Design synthesis methodology for dimensional management of assembly process with compliant non-ideal parts. *In*: *Proc. of International Joint Conference on Mechanical, Design Engineering and Advanced Manufacturing.*

Gasparetto, A. and Zanotto, V., 2010. Optimal trajectory planning for industrial robots. *Advances in Engineering Software*, 41 (4), 548–556.

Hatwig, J., *et al.*, 2012. An automated path planning system for a robot with a laser scanner for remote laser cutting and welding. *In*: *Proc. of the 2012 IEEE International Conference on Mechatronics and Automation (ICMA)*, 1323–1328.

Hatwig, J., Reinhart, G., and Zaeh, M.F., 2010. Automated task planning for industrial robots and laser scanners for remote laser beam welding and cutting. *Production Engineering*, 4 (4), 327–332.

Hwang, Y.K. and Watterberg, P.A., 1996. Optimizing robot placement for visit-point tasks. *In*: *Proc. AAAI Workshop on Artificial Intelligence for Manufacturing*, 81–86.

Iordachescu, D., *et al.*, 2011. Development of robotized laser welding applications for joining thin sheets. *In*: *Proc. of the 2011 International Conference on Optimization of the Robots and Manipulators*, 1–5.

Johnson, D.S. and McGeoch, L.A., 1997. The traveling salesman problem: a case study in local optimization. *In*: E.H.L. Aarts and J.K. Lenstra, eds. *Local search in combinatorial optimization.* John Wiley and Sons, Ltd., 215–310.

Kolakowska, E., Smith, S.F., and Kristiansen, M., 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. of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-2013)*, 457–461.

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

Larsen, E., *et al.*, 2000. Fast proximity queries with swept sphere volumes. *In*: *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2000)*, 3719–3726.

Levy, A.V. and Gómez, S., 1985. The tunneling method applied to global optimization. *Numerical optimization*, 1981, 213–244.

Li, B., Shui, B., and Lau, K., 2002. Fixture configuration design for sheet metal assembly with laser welding: a case study. *The International Journal of Advanced Manufacturing Technology*, 19 (7), 501–509.

Mitsi, S., *et al.*, 2008. Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm. *Robotics and Computer-Integrated Manufacturing*, 24 (1), 50–59.

Müller, M., Erdős, G., and Xirouchakis, P., 2004. High accuracy spline interpolation for 5-axis machining. *Computer-Aided Design*, 36 (13), 1379–1393.

Munzert, U., 2010. *Bahnplanungsalgorithmen für das robotergestützte Remote-Laserstrahlschweien.* Herbert Utz Verlag.

Nektarios, A. and Aspragathos, N.A., 2010. Optimal location of a general position and orientation end-effector's path relative to manipulator's base, considering velocity performance. *Robotics and Computer-Integrated Manufacturing*, 26 (2), 162–173.

Pamanes, G. and Zeghloul, S., 1991. Optimal placement of robotic manipulators using multiple kinematic criteria. *In*: *Proc. IEEE International Conference on Robotics and Automation*, 933–938.

Pashkevich, A.P., Dolgui, A.B., and Chumakov, O.A., 2004. Multiobjective optimization of robot motion for laser cutting applications. *International Journal of Computer Integrated Manufacturing*, 17 (2), 171–183.

Paul, R.P., Shimano, B., and Mayer, G.E., 1981. Differential kinematic control equations for simple manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11 (6), 456–460.

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

Saha, M., *et al.*, 2006. Planning tours of robotic arms among partitioned goals. *International Journal of Robotics Research*, 25 (3), 207–223.

Shibata, K., 2008. Recent automotive applications of laser processing in Japan. *The Review of Laser Engineering*, 36, 1188–1191.

Singh, A., *et al.*, 2014. Modelling of weld-bead geometry and hardness profile in laser welding of plain carbon steel using neural networks and genetic algorithms. *International Journal of Computer Integrated Manufacturing*, 27 (7), 656–674.

Tian, L. and Collins, C., 2005. Optimal placement of a two-link planar manipulator using a genetic algorithm. *Robotica*, 23 (2), 169–176.

Tsoukantas, G., *et al.*, 2007. On optical design limitations of generalized two-mirror remote beam delivery laser systems: the case of remote welding. *The International Journal of Advanced Manufacturing Technology*, 32 (9–10), 932–941.

Yin, R., *et al.*, 2014. A process planning method for reduced carbon emissions. *International Journal of Computer Integrated Manufacturing*, 27 (12), 1175–1186.

Zaeh, M.F., *et al.*, 2010. Material processing with remote technology revolution or evolution?. *Physics Procedia*, 5, Part A, 19–33.

Zeghloul, S. and Pamanes-Garcia, J., 1993. Multi-criteria optimal placement of robots in constrained environments. *Robotica*, 11 (2), 105–10.