Robotic path planning for non-destructive testing - a custom MATLAB toolbox approach

Carmelo Mineo* [a, b], Stephen Gareth Pierce[a], Pascual Ian Nicholson[b] and Ian Cooper[b]

[a] Department of Electronic and Electrical Engineering, University of Strathclyde
Royal College Building, 204 George Street, Glasgow G1 1XW, UK

[b] TWI Technology Centre (Wales), Harbourside Business Park, Harbourside Road, Port Talbot, SA13 1SB, UK

* Corresponding author: *carmelo.mineo@strath.ac.uk*

Abstract

The requirement to increase inspection speeds for non-destructive testing (NDT) of composite aerospace parts is common to many manufacturers. The prevalence of complex curved surfaces in the industry provides motivation for the use of 6 axis robots in these inspections. The purpose of this paper is to present work undertaken for the development of a KUKA robot manipulator based automated NDT system. A new software solution is presented that enables flexible trajectory planning to be accomplished for the inspection of complex curved surfaces often encountered in engineering production. The techniques and issues associated with conventional manual inspection techniques and automated systems for the inspection of large complex surfaces were reviewed. This approach has directly influenced the development of a MATLAB toolbox targeted to NDT automation, capable of complex path planning, obstacle avoidance, and external synchronization between robots and associated external NDT systems. This paper highlights the advantages of this software over conventional off-line-programming approaches when applied to NDT measurements. An experimental validation of path trajectory generation, on a large and curved composite aerofoil component, is presented. Comparative metrology experiments were undertaken to evaluate the real path accuracy of the toolbox when inspecting a curved 0.5 m^2 and a 1.6 m^2 surface using a KUKA KR16 L6-2 robot. The results have shown that the deviation of the distance between the commanded TCPs and the feedback positions were within 2.7 mm. The variance of the standoff between the probe and the scanned surfaces was smaller than the variance obtainable via commercial path-planning software. Tool paths were generated directly on the triangular mesh imported from the CAD models of the inspected components without need for an approximating analytical surface.

By implementing full external control of the robotic hardware, it has been possible to synchronise the NDT data collection with positions at all points along the path, and our approach allows for the future development of additional functionality that is specific to NDT inspection problems. For the current NDT application, the deviations from CAD design and the requirements for both coarse and fine inspections, dependent on measured NDT data, demand flexibility in path planning beyond what is currently available from existing off-line robot programming software.

Keywords
Robotics, path-planning, automated non-destructive inspections

1. Introduction

Non-destructive testing (NDT) is a highly multidisciplinary group of analysis techniques used throughout science and industry to evaluate the properties of materials, and/or to ensure the integrity of components/structures, without causing damage to them [1]. In civil aerospace manufacturing, the increasing deployment of composite materials demands a high integrity and traceability of NDT measurements, combined with a rapid throughput of data. Modern components increasingly present challenging shapes and geometries for inspection. Using traditional manual inspection approaches produce a time-consuming bottleneck in the industrial production [2] and hence provides the fundamental motivation for increased automation.

Modern Computer-Aided Design (CAD) is used extensively in composite manufacture. Additionally, where it was once necessary to construct large items from many smaller parts, Computer-Aided Manufacturing (CAM) now allows these large items to be produced easily from one piece of raw material (through traditional subtractive approaches, or built up using more recent additive manufacturing processes [3]). As a result, large components with complex geometries are becoming very common in modern structures, and the aerospace industry is a typical field, where wide complex shaped parts are very frequently used. Moreover the use of composite materials, which are notoriously challenging to inspect [4], is becoming widespread in the construction of new generations of civilian aircraft. To cope with future demand projections for these operations, it is therefore essential to overcome the current NDT bottleneck, which traditionally can be the slowest aspect in a production process.

A fundamental issue with composites manufacturing compared to conventional light alloy materials lies in the process variability. Often parts that are designed as identical, will have significant deviations from CAD, and also may change shape when removed from the mould. This presents a significant challenge for precision NDT measurement deployment which must be flexible to accommodate these manufacturing issues.

For these reasons, NDT inspection is often performed manually by technicians who typically have to position and move appropriate probes over the contour of the sample surfaces. Manual scanning requires trained technicians and results in a very slow inspection process for large samples. The repeatability of a test can be challenging in structures where complex setups are necessary to perform the inspection (e.g. orientation of the probe, constant standoff, etc.) [5]. While manual scanning may remain a valid approach around the edges of a structure, or the edges of holes in a structure, developing reliable automated solutions has become an industry priority to drive down inspection times. The fundamental aims of automation within the inspection process are to minimize downtimes due to the higher achievable speed, and to minimise variability due to human factors.

Semi-automated inspection systems have been developed to overcome some of the shortcomings with manual inspection techniques, using both mobile and fixed robotic platforms. The use of linear manipulators and bridge designs has, for a number of years, provided the most stable conditions in terms of positioning accuracy [6, 7]. The use of these systems to inspect parts with noncomplex shapes (plates, cylinders or cones) is widespread; typically, they are specific machines, which are used to inspect identically shaped and/or sized parts.

More recently, many manufacturers of industrial robots have produced robotic manipulators with excellent positional accuracy and repeatability. An industrial robot is defined as an automatically controlled, reprogrammable, multipurpose manipulator, programmable in three or more axes [8]. In the spectrum of robot manipulators, some modern robots have suitable attributes to develop automated NDT systems and cope with the challenging situations seen in the aerospace industry [9]. They present precise mechanical systems, the possibility to accurately master each joint, and the ability to export positional data at frequencies up to 500Hz. Some applications of 6-axis robotic arms in the NDT field have been published during the last few years and there is a growing interest in using such automation solutions with many manufacturers within the aerospace sector [2].

Exploring the current state of the art, RABIT is a group of systems developed by TECNATOM S.A., in collaboration with KUKA Robots Ibérica, that first approached the possibility of incorporating the use of industrial robots in NDT applications [9]. These systems boast the capability of using the potential of industrial robots and integrating them in an overall inspection apparatus, bringing together all the hardware and software required to plan and configure ultrasonic inspections. Off-the-shelf robotic arms were also used in the LUCIE (Laser Ultrasound for Composite InspEction) system, addressed to inspect large curved surfaces such as the inside of aircraft fuselage, by means of ultrasound generated by laser [10]. Genesis Systems Group has developed the NSpect family of Robotic Non-Destructive Inspection cells (Genesys, 2011). Incorporating the FlawInspecta technology (Lines, 2006) developed by Diagnostic Sonar in conjunction with National Instruments, the NSpect systems employ a KUKA 6DOF robot arm to perform ultrasonic inspection using either an immersion tank, or a recirculating water couplant. General Electric (GE) has also investigated the integration of phased array UT with off-the-shelf industrial robots for the inspection of aerospace composites [11, 12].

Despite these previous efforts, there remain challenges to be addressed before fully automated NDT inspection of complex geometry composite parts becomes commonplace. The key challenges include generation and in-process modification of the robot tool-path, high speed NDT data collection, integration of surface metrology measurements, and overall visualisation of measurement results in a user friendly fashion. Collaborations driving this vision include TWI Technology Centre (Wales), which is currently carrying out a 3-year project, called IntACom, on behalf of its sponsors; its objective is to achieve a fourfold increase in the throughput of aerospace components [2]. Additionally the UK RCNDE consortium conducts research into integration of metrology with NDT inspection [13, 14]. Both these consortia have identified the requirement for optimal tool path generation over complex curved surfaces, and the current article describes joint work between these groups to develop a novel approach to a flexible robotic toolpath generation using a user friendly MATLAB toolbox. This new toolbox provides a low cost research based approach to robot path planning for NDT applications, and provides a platform for future development of highly specific NDT inspection challenges.

2. Approaches to robotic path planning for NDT applications

2.1 Existing robotic path-planning software

Six-axis robotic arms have traditionally been used in production lines to move the robot end-effector from one position to another for repetitive assembly and welding operations. In this scenario, where the exact trajectory between two points in the space is not too important, the teach pendant of a robot is used to move manually the end-effector to the desired position and orientation at each stage of the robot task. Relevant robot configurations are recorded by the robot controller and a robot programme is then written to command the robot to move through the recorded end-effector postures. More recently, accurate mechanical joints and control units have made industrial robotic arms flexible and precise enough for finishing tasks in manufacturing operations [15]. Robotic manipulators are highly complex systems and the trajectory accuracy of a machining tool has a huge impact on the quality and tolerances of the finished surfaces. As a result, many software environments have been developed by manufacturers, academic researchers and also by the robot manufacturers themselves, in order to help technicians and engineers to program complex robot tasks [16]. The use of such software platforms to program robot movements is known as Off-Line Programming (OLP). It is based on the 3D virtual representation of the complete robot work cell, the robot end-effector and the samples to be manipulated or machined. Although some limited applications for inspection delivery have been demonstrated [17], in general conventional OLP is geared towards manufacturing applications where the task is the production of a specific component using conventional milling/ drilling / trimming operations. In contrast, the result of an automated NDT inspection requires a flexible and extensible approach that has the flexibility to allow future changes in the path planning to accommodate requirements of future NDT inspections. Building the NDT toolbox in Matlab provides an easy route to such future adaptation by the research community (e.g. fluid dynamics modelling of water-jets, compensation for part variability and conditional programming approaches).

Using current OLP software to generate appropriate tool-paths for NDT purposes can appear quite straightforward at first inspection; however it is possible to list a series of serious inadequacies:

1. Path-planning for automated NDT inspections is a very specific task. As previously mentioned, much commercial software for off-line robot programming draws its origin from the need to use the advantageous flexibility of general robotic manipulators to replace the more traditional and usual machining tools (milling machines, lathes, etc.). As a result, many commercial software applications for off-line programming of robots are expensive tools, incorporating a lot of functionality specific for CAD/CAM purposes and machining features. Conventional OLP software has no easy provision for modification of toolpath based on specific characteristics of NDT inspection. For example, in water-jet coupled ultrasonic testing, the exact orientation and separation from part of the end effector, combined with water pressure and nozzle characteristics will influence the trajectory of the water jet. Since the

new toolbox is Matlab based, it is flexible and easy to accommodate such new features into the path planning algorithm in future.

2. Significant complications exist when two or more robotic arms need to be synchronized in order to perform a specific NDT inspection. The Ultrasonic Through-Transmission (UTT) technique, for example, uses two transducers: one emitter and one receiver; the receiver being placed on the opposite side of the component and facing the transmitting probe. Many commercial pieces of software (e.g. Delcam and Mastercam) do not offer any support for co-operating robots. FastSurf, an add-on from CENIT for Delmia (Dassault Systems), allows partial synchronization of robotic movements (e.g. at start or end points of complex paths) but not full synchronisation over the complete path, required for the UTT technique. Our new Matlab toolbox provides the capability for full point to point synchronisation between robots for situations where a change in material section is encountered – this is far more sophisticated than simple master-slave synchronisation implemented by typical robot equipment suppliers.

Contrary to currently available OLP software, our new Matlab toolbox provides a capability for full synchronisation (at all points on the path) with external instrumentation systems (in our case an ultrasonic NDT inspection system). Such synchronisation is fundamental [7] to building an accurate map of NDT results on an inspected part with the accuracy required (typically sub-millimetric).

2.2 Robot programming & simulation in MATLAB

MATLAB ® is a common platform for the modelling and simulation of various kinds of systems. It is a flexible programming software environment extensively used for matrix manipulations, plotting of data, implementation of algorithms and creation of user interfaces. It can also be interfaced with programs written in other languages, including C, C#, C++, Java and Fortran. As such, MATLAB is a popular choice for the simulation of robotics systems.

Specific toolboxes (collections of dedicated MATLAB functions) have been developed in the past few years for research and teaching in almost every branch of engineering, such as telecommunications, electronics, aerospace, mechanics and control. Several toolboxes have been presented for the modelling of robot systems [18-22]. These software tools have been inspired by various application scenarios, such as robot vision [21, 22], and space robotics [19], and have addressed both industrial [20] and academic/educational [18, 21, 22] targets. Off-line programming has been investigated by [23] using a combined Simulink/SimMechanics approach.

To date, no NDT specific path-planning software has been presented in the literature. The following section describes the architecture of a new MATLAB toolbox, developed to specifically address the current needs of robotic NDT related to effective tool-path generation taking into account the deficiencies of existing off-line programming as outlined above.

3. RoboNDT software

Originally a Matlab toolbox for robotic path planning targeted to ultrasonic NDT inspection was developed. A modular approach to the toolbox development was adopted throughout to allow for growth and progressive validation of a large-scale project. The toolbox was based on 4 main modules: *start-up*, *path-planning*, *evaluation* and *outputs*. The latest developments have led to a full software application, named RoboNDT, equipped with a Graphical User Interface (GUI) to enhance ease-of-use. The latest executable version of the application can be downloaded from http://www.strath.ac.uk/eee/research/cue/downloads/.

Figure 1 shows the schematic architecture of RoboNDT.

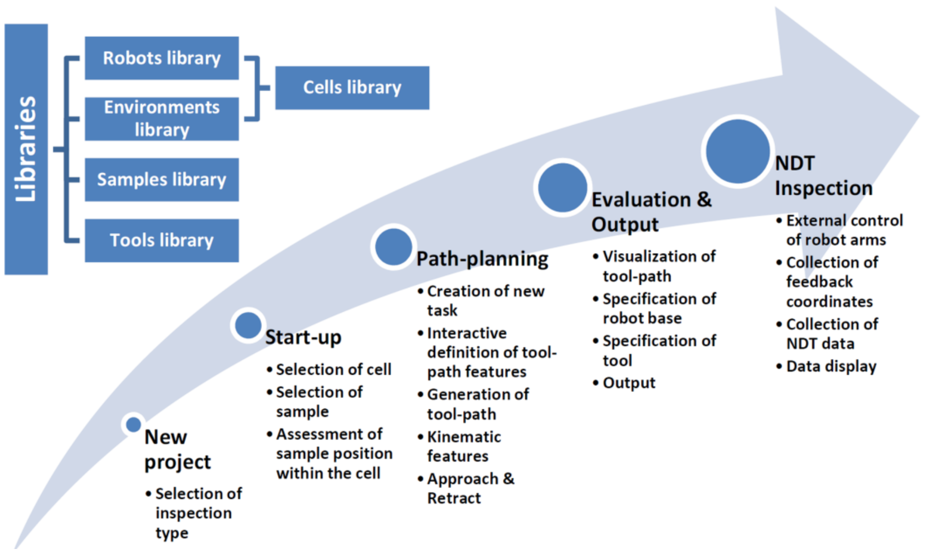


Figure 1 - Schematic architecture of RoboNDT.

Traditional commercial path-planning software generates specific robot language programmes that need to be transferred to the robot controller to be executed. RoboNDT generates output files suitable to be used through a C++ server application that has been developed to achieve external control of KUKA robots. This is a novel approach. The command packets of coordinates are sent from an external computer in real-time to the robot controller via Ethernet. Working with KUKA Robots, the external control is enabled by the KUKA Robot Sensor Interface (RSI) software add-on installed into the robot controllers [24]. The C++ application manages the reception of robot feedback positions and NDT data, whilst commending the tool-path to the robot manipulating the NDT probe. This approach allows sending of command coordinates to multiple robots from the same external server computer and enables the path synchronization mismatch to be maintained within the distance covered by the robots in a single interpolation cycle. For example for robots controlled in a 12 milliseconds interpolation cycle running at 100 mm/s, the maximum path mismatch would be equal to 1.2 mm. This worst case scenario is much improved over commercial solutions that use digital I/O signals for synchronization purposes. In addition this approach is more sophisticated than simple master-slave synchronisation approaches developed by robot suppliers. Our solution allows for a constant change in the relative path to be encoded with ease, such a situation arises in ultrasonic inspection when considering through transmission of ultrasound through materials with constantly changing thickness.

3.1 LIBRARIES

It was deemed that, in order to develop a flexible platform, the easy and appropriate definition of all elements involved in the path-planning operations has to be guaranteed. Five libraries have been implemented to allow the user to easily reproduce the real working environment for the robotic NDT inspection and use virtual models of the real equipment. The Libraries menu, accessible from the menu bar (Figure 2a), provides access to these important modules of the GUI. They are the robot library, the tool library (probes and sensors) the environment library and the appropriate contexts to manage the robot cells and the samples of interest.

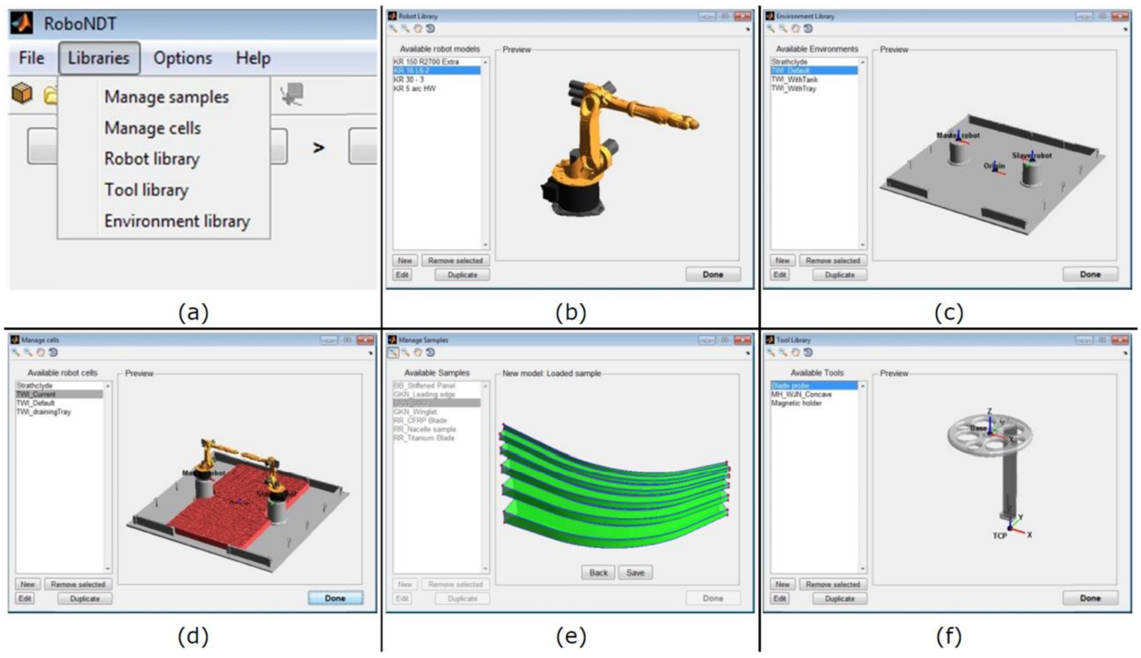
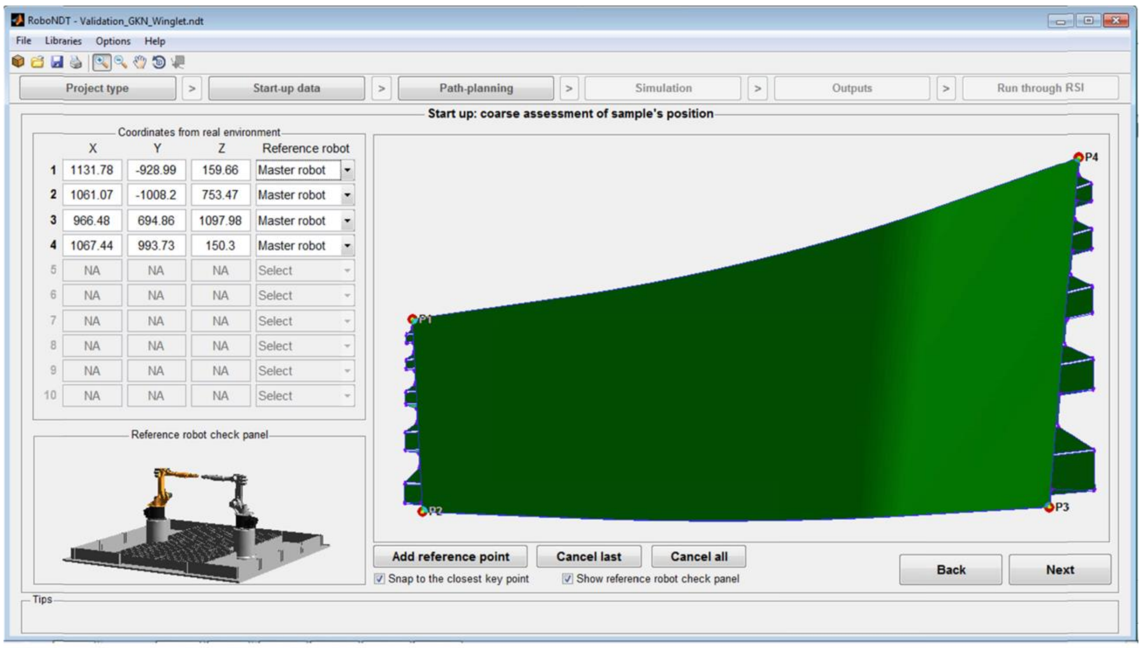


Figure 2 - Libraries menu (a), Robot library (b), Environment library (c), Cell management (d), Sample management (e) and Tool library (f).

The list of available items is placed on the left hand side of each library. The user can select any of the items to display the related CAD model. The push buttons on the bottom left corner can be used to remove, edit or duplicate the selected robot model or to create a new item. All libraries, except the cell management context, allow loading of STL (Standard Tessellation Language) CAD files and the specification of key properties for robots, environments, samples and tools (kinematic features, coordinate reference systems, etc…). The cell management context allows the user to create a new robotic working environment through assembling one or more robots into one selected environment.

3.2 START-UP MODULE

The triangular mesh of the sample, imported from the STL file, needs to be placed in the correct position within the virtual robotic cell. Existing software usually considers the CAD models strictly correspondent to the real parts; whilst this can be tolerated for well machined metallic samples, it is sometimes the source of unacceptable errors for large composite components. Therefore the sample's position calibration mode implemented in RoboNDT uses a positioning algorithm originally proposed for 3D point cloud data registration [25]. It calculates the optimum position of the STL mesh within the virtual robot cell in order to minimize the square errors of the distances between at least four points selected in the real sample and the relative points in the CAD model. The i-th point whose coordinates are measured through jogging the robot arm in the real environment is herein named as P_e^i; the relative point selected from the virtual CAD model of the sample is named as P_r^i. Figure 3 shows locating the four required reference points of a complex curved aerofoil sample. For the sake of describing the operation of the software, the carbon fibre composite component, shown in Figure 3, is used for the experimental validation described in this paper. The sample's wide surfaces, spanning across most of the available robot working envelope and curving in different directions, were chosen for testing and validating the path-planning software.



RoboNDT - Validation_GKN_Winglet.ndt
File
Libraries
Options
Help
Project type
Start-up data
Path-planning
Simulation
Outputs
Run through RSI
Start up: coarse assessment of sample's position
Coordinates from real environment
Reference robot check panel
P1
P2
P3
P4
Add reference point
Cancel last
Cancel all
Snap to the closest key point
Show reference robot check panel
Back
Next
Tips

	X	Y	Z	Reference robot
1	1131.78	-928.99	159.66	Master robot
2	1061.07	-1008.2	753.47	Master robot
3	966.48	694.86	1097.98	Master robot
4	1067.44	993.73	150.3	Master robot
5	NA	NA	NA	Select
6	NA	NA	NA	Select
7	NA	NA	NA	Select
8	NA	NA	NA	Select
9	NA	NA	NA	Select
10	NA	NA	NA	Select

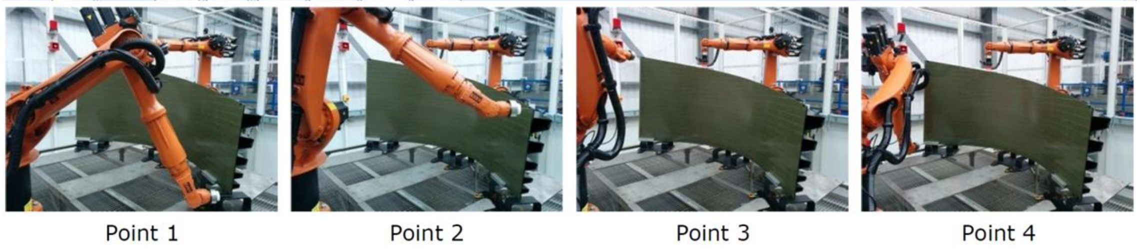


Figure 3 - Definition of four reference points for sample's position calibration.

The centroids of the two datasets (P_r and P_e), named cP_r and cP_e, are given by:

$$cP_r = [cP_{rx} \quad cP_{ry} \quad cP_{rz}] = \left[\frac{\sum_{i=1}^{4} P_{rx}^i}{4} \quad \frac{\sum_{i=1}^{4} P_{ry}^i}{4} \quad \frac{\sum_{i=1}^{4} P_{rz}^i}{4}\right]; \qquad (1)$$

$$cP_e = [cP_{ex} \quad cP_{ey} \quad cP_{ez}] = \left[\frac{\sum_{i=1}^{4} P_{ex}^i}{4} \quad \frac{\sum_{i=1}^{4} P_{ey}^i}{4} \quad \frac{\sum_{i=1}^{4} P_{ez}^i}{4}\right]; \qquad (2)$$

Therefore the point clouds with the centroids removed (mP_r and mP_e) are:

$$mP_r = P_r - cP_r; \tag{3}$$

$$mP_e = P_e - cP_e; \tag{4}$$

A correlation matrix (H) is calculated as:

$$\mathrm{H} = \sum_{i=1}^{4} mP_r(i,:)^T * mP_e(i,:); \tag{5}$$

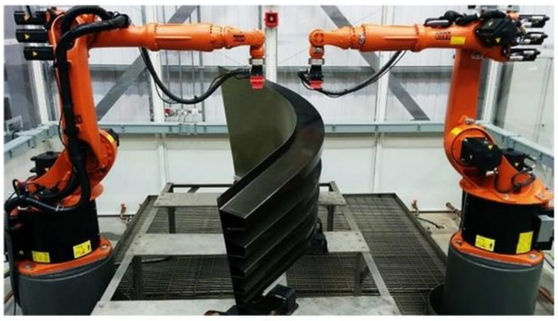
The singular value decomposition (SVD) function, applied to the correlation matrix, produces a diagonal matrix S of the same dimension as H, with nonnegative diagonal elements in decreasing order, and unitary matrices U and V so that $H = U * S * V'$.

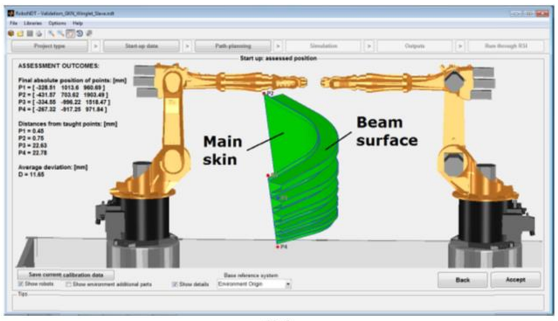
The rotation matrix (R) and the translation vector (t), necessary to match the reference cloud (P_r) with the experimental cloud of points (P_e), are then calculated from the SVD output.

$$R = V * U'; \tag{6}$$

$$t = cP_e' - R * cP_r'; \tag{7}$$

The calculated rotation matrix and translation vector are used to locate the meshed sample in the best position. The result of this operation is then displayed, so that the user can verify and accept before proceeding. Figure 4a shows a picture of the real setup; it shows the robotic hardware of an automatic inspection prototype system developed within the TWI led project IntACom. The system utilises two KUKA KR16 L6-2 robotic arms, with controllers that run KUKA System Software (KSS) 8.2. Mounted on the robot end-effectors are 3D-printed water jet nozzles, which encapsulate phased array ultrasonic probes. The water jets are used to transmit the ultrasonic waves to the specimen under inspection. The specimen under test in this work is a curved composite sample with 1.6m^2 surface inspection area. Figure 4b is a screen shot of the sample model in the virtual robot environment at the end of the calibration.


(a)



ASSESSMENT OUTCOMES:

Final absolute position of points: [mm]
P1 = [-328.51 1013.6 960.69]
P2 = [-431.57 703.62 1903.49]
P3 = [-334.55 -996.22 1518.47]
P4 = [-267.32 -917.25 971.84]

Distances from taught points: [mm]
P1 = 0.45
P2 = 0.75
P3 = 22.63
P4 = 22.78

Average deviation: [mm]
D = 11.65

(b)

Figure 4 - Calculated position of the sample, waiting for the user to approve.

3.3 PATH PLANNING MODULE

The easiest way to generate a tool-path following the contour of a meshed CAD surface would consist of approximating the mesh with a polynomial analytical surface. However, during initial development, this approach revealed its limitations. The approximation introduces an error by definition. The error can obviously be decreased by increasing the order of the polynomial fitting function, but at the expense of increased computation time.

More importantly, the approximation of a meshed surface with a polynomial surface is only possible when the surface can be mathematically described by a surjective function. A surjective function, $z = f(x, y)$ with X-Y domain and codomain in Z, is surjective (or a surjection) if every element z in Z has a corresponding x-y couple such that $z = f(x, y)$. The function *f* may map more than one couple of X-Y to the same element of Z, but not the opposite. The inverse of a surjective function is not surjective. As a result, the approximation of a meshed surface fails if the surface is not surjective and it is influenced by the orientation of the surface in the 3D Cartesian space.

Therefore a new path-planning approach was developed. Since the CAD files are imported as meshed objects, the basic idea is to compute to tool-path directly on the triangular mesh without need for an approximating analytical surface. For the sake of presenting the algorithm, let us consider the circumstance where we need to create a curve parallel to one edge of a given surface edge, maintaining the distance *d* from the edge and laying down on the same surface.

Because of the triangular mesh constituting the surface, the surface edge is formed by segments, whose extremities coincide with two corners of the adjacent triangles. For each segment of the surface edge it is possible to find the intersection points between the plane perpendicular to the segment for its middle point and the edges of the triangles of the surface mesh. Figure 5a shows the intersection points relative to the first segment of the surface edge.

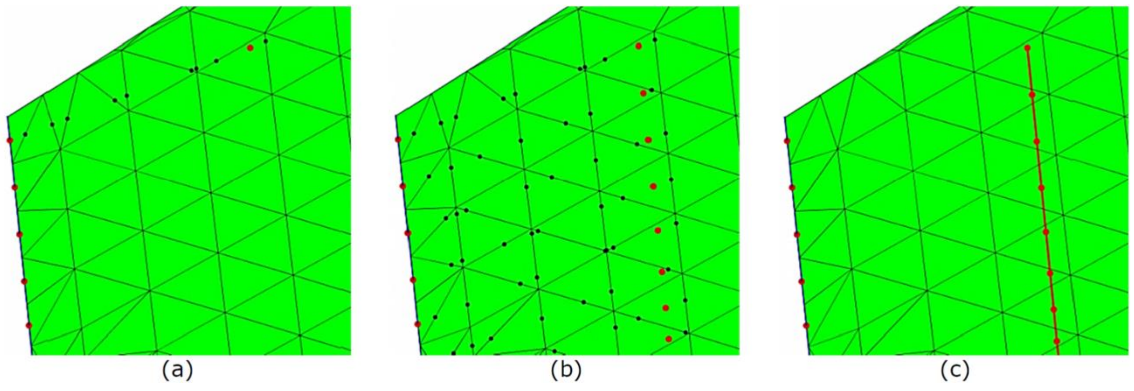


Figure 5 - Algorithm for the generation of a curve parallel to the reference edge, described step by step.

Starting from the point on the surface edge and following the succession of the intersection points, the curvilinear distance from the reference edge is calculated. The resulting distance is the proper distance along the surface contour. The integration of the distance stops when the distance exceeds the set distance *d*. The farther intersection points are ignored. Since the last remaining point is further than the set distance and the second-last point is closer, a point that is standing exactly at the set distance is calculated through interpolation between the two points. This point lies on one of the mesh triangles. The process is repeated for all the segments in the reference edge (Figure 5b). The final net result is a curve, parallel to the selected reference edge

(Figure 5c). Since all of the points of the curve lay on the mesh triangles, the accuracy with which the curve follows the contour of the surface is equal to the maximum deviation between the mesh and the sample surface. Since every point of the obtained curves lay on one (and only one) of the triangles of the mesh, the perpendicular direction associated to each point is given by the vector normal to the relative triangle.

For the creation a raster scan tool-path the algorithm described so far is iterated to generate other parallel lines, equally spaced, covering the entirety of the meshed surface. The row of crossed triangles shown in Figure 6a divides the mesh in two regions: the region that has already been swept, between the reference edge and the generated curve, and the remaining part of the mesh. The former region is identified (Figure 6b) and excluded from the domain of interest for the iteration of the algorithm. The generated curve is then replaced to the reference surface edge and a new parallel curve is computed. Figure 6c shows the result of the iteration of the algorithm to achieve 100 % coverage of the test surface, where three irregular shaped holes were introduced to test the algorithms under more challenging circumstances.

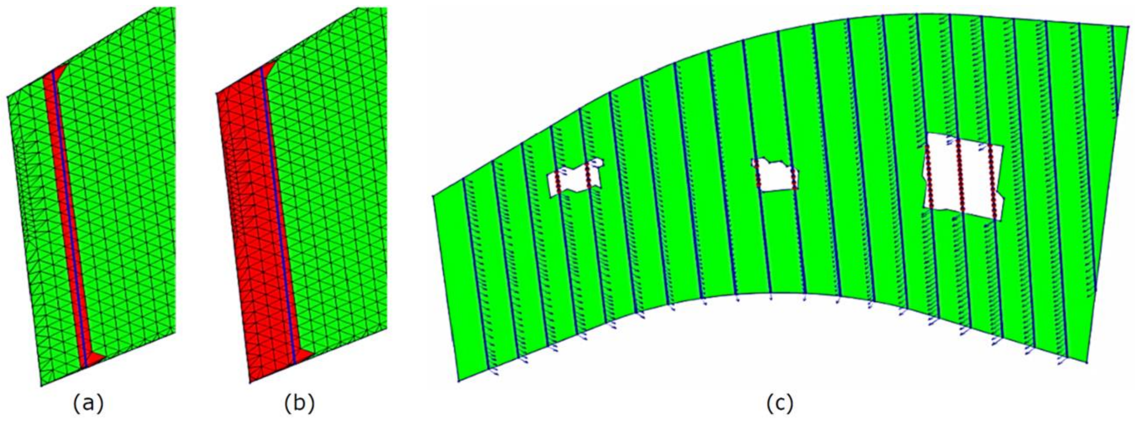
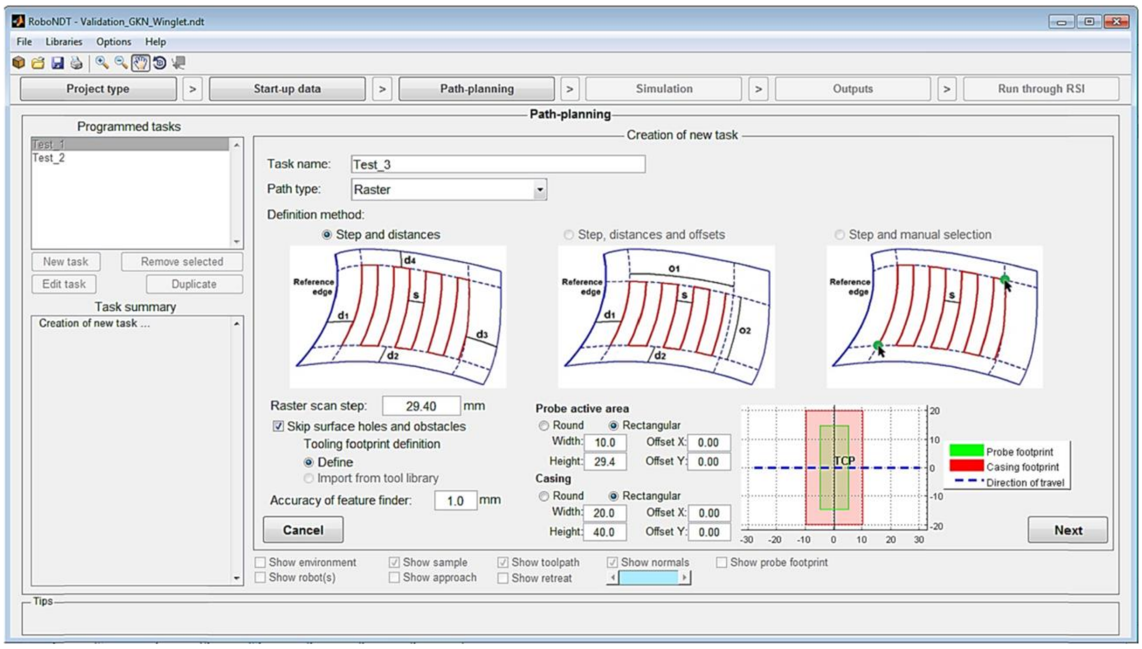


Figure 6 - Triangles crossed by the parallel curve (a), meshed region to be excluded (b) and parallel curves given by the iteration of the algorithm (c).

It is important to optimize the NDT coverage around surface voids and obstacles, and rule out any risk of collisions. The software is able to recognize holes and obstacles in the surface of interest. The footprint of the ultrasonic probe active area and its casing is definable in the GUI. The user can specify the footprints during the creation of a new inspection tool-path (Figure 7).



Path-planning

Programmed tasks: Test_1, Test_2

Task summary: Creation of new task ...

Creation of new task

Task name: Test_3

Path type: Raster

Definition method: [✓ Step and distances] Step, distances and offsets Step and manual selection

Raster scan step: 29.40 mm

[✓] Skip surface holes and obstacles

Tooling footprint definition: [✓ Define] Import from tool library

Accuracy of feature finder: 1.0 mm

Probe active area: Round [✓ Rectangular]
Width: 10.0 Offset X: 0.00
Height: 29.4 Offset Y: 0.00

Casing: Round [✓ Rectangular]
Width: 20.0 Offset X: 0.00
Height: 40.0 Offset Y: 0.00

☐ Show environment ✓ Show sample ✓ Show toolpath ✓ Show normals ☐ Show probe footprint
☐ Show robot(s) ☐ Show approach ☐ Show retreat

Figure 7 - Creation of a new task. First phase with the selection of the tool-path type and the definition of the tool footprints.

If the desired tool-path type is a raster, the individual generated curves have to be linked to generate a single scanning raster path. The end of each line is linked with the first point of the next line, inserting a connecting path. During this step, the software adds the kinematic features to the tool-path. Acceleration and deceleration ramps characterize the robot end-effector speed pattern at the start and at the end point of each continuous portion of the tool-path. If α is the duration of the acceleration and deceleration ramps in a normalized time scale (t) and $v(t)$ is the normalized speed as a function of time, the following conditions are applied to obtain a continuous speed pattern:

$$\begin{cases} v(0) = 0 \\ v(\alpha) = 1 \\ v(1-\alpha) = 1 \\ v(1) = 0 \end{cases} \quad \begin{cases} v'(0) = 0 \\ v'(\alpha) = 0 \\ v'(1-\alpha) = 0 \\ v'(1) = 0 \end{cases} \tag{8}$$

The typical speed pattern is given in Figure 8a. It is described by the function:

$$\begin{cases} v(t) = -\frac{2}{\alpha^3}t^3 + \frac{3}{\alpha^2}t^2 & \text{for } 0 \le t \le \alpha \\ v(t) = 1 & \text{for } a \le t \le (1-\alpha) \\ v(t) = \frac{2}{\alpha^3}(t+\alpha-1)^3 - \frac{3}{\alpha^2}(t+\alpha-1)^2 + 1 & \text{for } (1-\alpha) \le t \le 1 \end{cases} \tag{9}$$

When the tool-path continuous portion is not long enough to allow reaching of the regime speed, e.g. for short distances between two consecutive parallel curves, the following conditions replace the former ones:

$$\begin{cases} v(0) = 0 \\ v(0.5) = \beta \\ v(1) = 0 \end{cases} \quad \begin{cases} v'(0) = 0 \\ v'(0.5) = 0 \\ v'(1) = 0 \end{cases} \tag{10}$$

where β is a percentage of the target speed used for the raster scan. This parameter spans between 0 and 1 according to the length of the trajectory linking two consecutive lines of the scan path. Small values of β are used for short trajectories, to let the robot quietly abandon the end point of the finished line and reach the starting point of the next line. The speed function results:

$$v(t) = 16\beta t^4 - 32\beta t^3 + 16\beta t^2 \qquad \text{for } 0 \leq t \leq 1 \tag{11}$$

The typical pattern of the speed is given in Figure 8b.

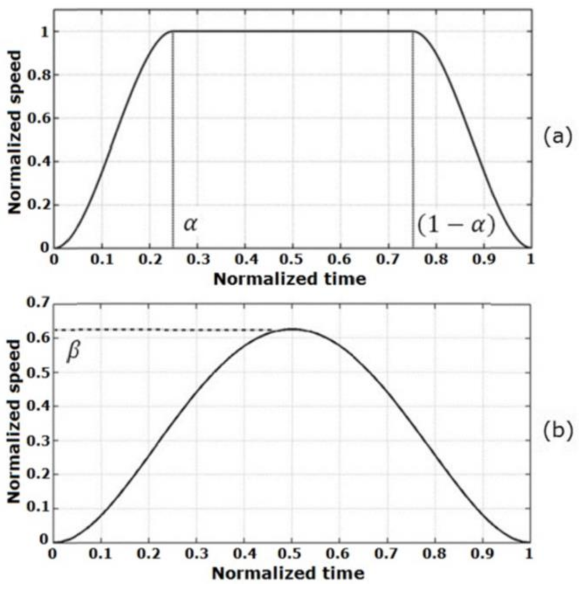


Figure 8 - Typical speed pattern for a long continuous curve (a) and a short one (b).

3.4 EVALUATION & OUTPUT

Figure 9 shows the inspection tool-paths generated through RoboNDT for the experimental tests described in section 4.

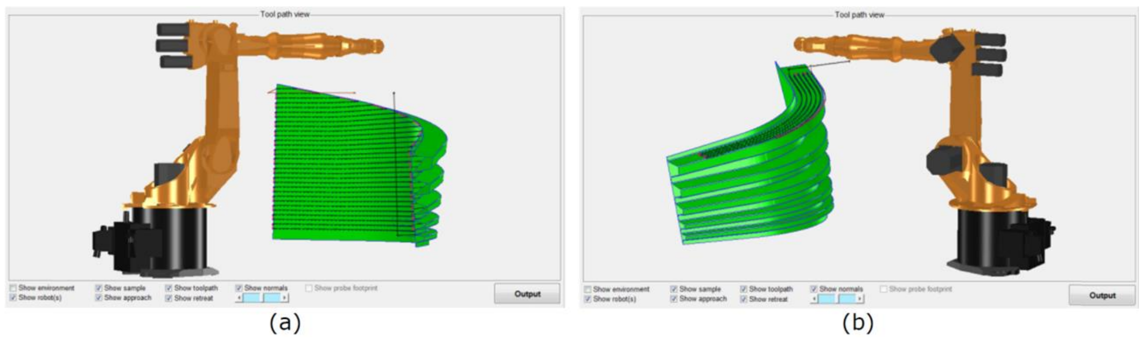


Figure 9 - Evaluation of generated tool-paths.

The tool-paths and the approaching and retracting trajectories are displayed relative to the virtual model of the robot arm. For the sake of testing the software with surfaces curving in different directions, the main skin of the winglet and the top surface of one of its back wall beams (Figure 4b) were considered for path-planning. The main skin surface has an area of 1.6 m^2; the beam surface has an area of 0.5 m^2. The generated tool-paths are raster scans with a 29.4 mm raster step. This step is suitable for phased-array ultrasonic inspection (PAUT) when 64 elements, 0.6 mm pitch phased-array probe is employed and its elements are fired with focal law that uses a sub-aperture of 14 elements.

The output function of the software translates the generated tool-path into a set of command coordinates packets that can be interpreted by the robot controllers. Each robot pose is represented by a vector, $p=[x, y, z, A, B, C]^T$, containing the three Cartesian coordinates of a given position and the roll (A), pitch (B) and yaw (C) angles of the end-effector orientation for that position. The conversion of the normal vector components (N_X, N_Y, N_Z) into the angular coordinates is based on the following rotational matrix:

$$\mathbf{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} = -\begin{bmatrix} V_X & T_X & N_X \\ V_Y & T_Y & N_Y \\ V_Z & T_Z & N_Z \end{bmatrix} \tag{12}$$

The normal vector components populate the third column of the matrix. The second column contains the tangential vector, representing the direction of travel calculated as:

$$\mathbf{T} = \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix} = \begin{bmatrix} \frac{dx}{\sqrt{dx^2+dy^2+dz^2}} \\ \frac{dy}{\sqrt{dx^2+dy^2+dz^2}} \\ \frac{dz}{\sqrt{dx^2+dy^2+dz^2}} \end{bmatrix} \tag{13}$$

where dx, dy and dz are the gradients of the trajectory in the three dimensions.
The first column contains the bi-normal vector:

$$\mathbf{V} = \begin{bmatrix} V_X \\ V_Y \\ V_Z \end{bmatrix} = \begin{bmatrix} N_Y T_Z - N_Z T_Y \\ N_Z T_X - N_X T_Z \\ N_X T_Y - N_Y T_X \end{bmatrix} \tag{14}$$

Thus the angular coordinates (A, B, C) are calculated through the following general formulation:

$$\mathrm{B} = \operatorname{atan2}\left(\frac{-\mathrm{R}_{31}}{\sqrt{\mathrm{R}_{11}^2+\mathrm{R}_{21}^2}}\right) \tag{15}$$

$$A = \begin{cases} 0, & for\ |\mathrm{B}| = \pi/2 \\ \operatorname{atan2}\left(\frac{\mathrm{R}_{21}}{\mathrm{R}_{11}}\right), & for\ |\mathrm{B}| \neq \pi/2 \end{cases} \tag{16}$$

$$C = \begin{cases} \mathrm{C} = \frac{B}{abs(B)} \cdot \operatorname{atan2}(\mathrm{R}_{12}), & for\ |\mathrm{B}| = \pi/2 \\ \mathrm{C} = \operatorname{atan2}\left(\frac{\mathrm{R}_{32}}{\mathrm{R}_{33}}\right), & for\ |\mathrm{B}| \neq \pi/2 \end{cases} \tag{17}$$

The software generates two output text files: the first contains all command coordinates the robot needs to receive to inspect the target surface, and a second short log file containing the points to set the initial and final motion to approach the starting point of the inspection and to abandon the endpoint. These two files have very simple syntax; each line merely contains 6 coordinates (*x, y, z, A, B, C*) to drive the robotic arm to a specific pose. The two text files can be used by the aforementioned C++ server application and the packets of coordinates are sent one by one to the robot controller via Ethernet communication.

4. Validation experiments – Path accuracy and NDT results

Tests were carried out to validate the accuracy of the tool-paths generated through RoboNDT and prove the reliability of the new approach, based on external control of the robot motion and simultaneous collection of feedback coordinates and NDT data carried out by the C++ server application. Since one single application manages the command and the feedback packets of coordinates, the new approach allows monitoring of the dynamic accuracy of the tool-paths. This type of investigation is not possible with the traditional approach.

Two sets of tool-paths were created to execute the NDT inspection at 100 mm/s and 300 mm/s, maintaining the same robot acceleration equal to 500mm/s^2. The positional error is calculated as the distance between the commanded tool centre points (TCPs) and the reached points as measured by the robot encoders. The orientation error is calculated as the mismatch angle between the commanded rotation matrix and the rotation matrix computed from the feedback roll, pitch and yaw angles.

Table 1 shows the maps of position and orientation errors for all tool-paths generated through RoboNDT. For the sake of helping the comparison, the same colour scale has been maintained where possible.

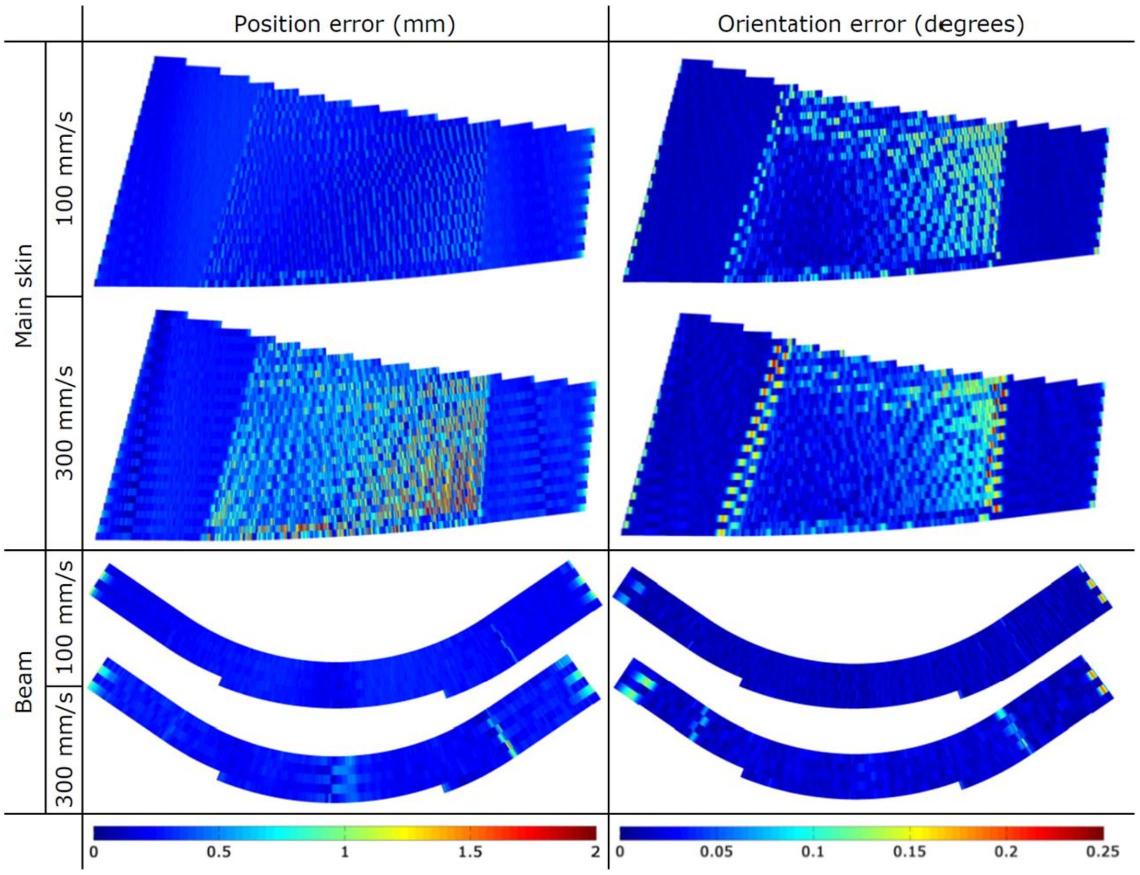


Table 1 - Maps of position and orientation errors.

Table 2 reports the maximum and Root Mean Square (RMQ) errors. The maximum position error is equal to 2.70 mm and the maximum orientation error is equal to 0.29 degrees. As it was expected, faster speeds produce bigger errors because of the inertial effects affecting the robotic motion.

		Inspection speed			
		100 mm/s		300 mm/s	
		Main skin	Beam surface	Main skin	Beam surface
Position error (mm)	Max	1.37	1.18	2.70	1.53
	RMS	0.30	0.27	0.52	0.33
Orientation error (degrees)	Max	0.21	0.20	0.29	0.24
	RMS	0.04	0.02	0.05	0.03

Table 2 - Maximum and Root Mean Square (RMQ) errors.

The variability of the standoff between the probe and the surfaces is shown in Table 3 with Time-Of-Flight (TOF) maps of the ultrasonic wave reflected from the scanned surface to the probe. The TOF values have been divided by the speed of sound in the sample to quantify the standoff variability in millimetres. The standoff relative to the RoboNDT tool-path is compared to that relative to the tool-path generated through leading aerospace commercial path-planning software based on the Dassault Delmia V5 platform. The comparison is made for the same travelling speed of 300 mm/s and acceleration of 500 mm/s^2. The phased array probe focal law and the setting of the ultrasonic receiver (a Micropulse 5PA from PeakNDT) were set to acquire C-scans with resolution of 1.2 mm in all directions. The tool-paths accuracy is evaluated through comparing the feedback coordinates received from the robot encoders and the command coordinates.

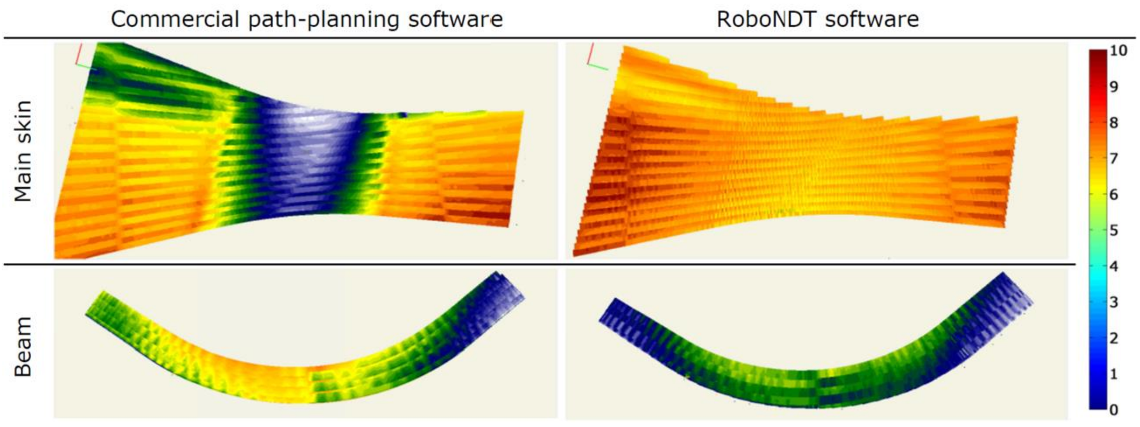


Table 3 - Maps of standoff between probe and scanned surface.

The variability of the standoff is within 10 mm for the tool-paths created with the commercial software and within 4.5 mm for the RoboNDT tool-paths.

The experimental data demonstrates that the path errors, achievable through externally controlled, robots are lower than the errors given by the traditional approach. The development of RoboNDT has enabled a new viable and innovative approach for robotic NDT; however the software is not yet optimized in terms of computation times. The path-planning tasks executed by RoboNDT take around 5 times longer than the time taken by the commercial software. An Intel® Xeon® CPU computer with 24Gb of RAM, running the 64-bit Windows 7 operating system, was used to test both approaches.

4.1 NDT RESULTS

The raster scan of the main skin and of the beam surface took respectively 205sec and 38sec with RoboNDT tool-paths and 200sec and 35sec with the tool-paths generated through the commercial software. Previous manual scans of the same surfaces were respectively completed in 2 and 0.4 hours; this results in robotic inspection being around 40 times faster than manual inspection (in addition to being much more reliable, repeatable and accurate).

It is clear from Table 4 that the path accuracy of RoboNDT derived tool-paths exceeds those obtained from the commercial software. For the current application, the level of accuracy for both approaches is sufficient as the intended NDT delivery is accomplished using a water jet coupling approach [9, 26]. The water path from water nozzle to sample surface can easily accommodate such tool-path inaccuracies. The bottom row of Table 4 shows the close-up of an array of artificial squared delaminations embedded within the thickness of the winglet main skin. The smallest delaminations have a size of 3 mm and are visible in both cases. For other NDT inspection applications the improvements in path accuracy are more significant, for example if implementing eddy current inspections, then a tight control of standoff distance is required throughout the path to avoid false defect indications.

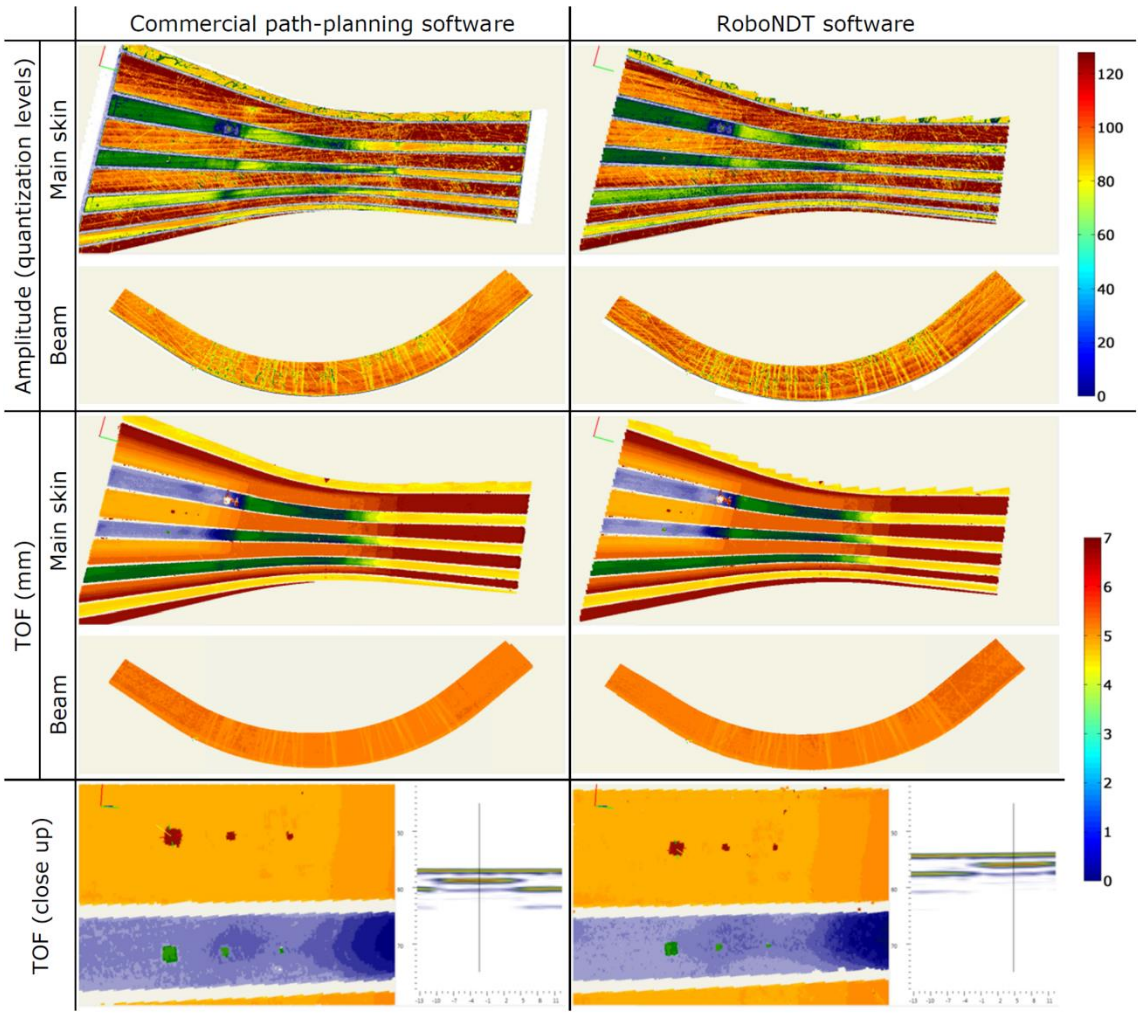


Table 4 - NDT results.

5. Conclusions

In modern aerospace manufacturing, the increasing role of composite materials is introducing new challenges to the inspection and verification procedures employed to ensure safe deployment of the components in the finished structure. Traditional NDT methods such as ultrasonic testing are fundamental for such inspection. However the complex part shapes employed in aerospace structures, combined with complex material properties of composites, present significant challenges. Traditional manually delivered NDT is time consuming and manufacturers are increasingly demanding decreased cycle times for the inspections undertaken. Although some part geometries lend themselves to bespoke Cartesian or Cartesian plus rotation stage mechanical scanners, there are many instances of complex geometry that make the use of 6 axis robot positioners highly attractive. Most existing commercial off-line programming approaches are geared towards manufacturing processes, and lack the required flexibility for application to delivery of NDT measurements. In particular the lack of full point by point synchronisation, between multiple robots and the external measurement system, has been identified as one of the key shortcomings in existing software. Future flexibility to accommodate part variability through conditional programming approaches, and ability to build additional path modification due to effects such as water jet orientation are also key attributes of the new software tool developed.

The software, named RoboNDT, has been tailored to the generation of raster scan paths for the inspection of curved surfaces by 6-axis industrial robots, and in its current form represents the first iteration of a system designed to overcome the issues with current OLP packages. RoboNDT is intended to be flexible and extendable to accommodate future system and robot developments. It has been explained how the execution of the calculated path by a robotic arm, externally controlled through a C++ server application, can be beneficial for NDT inspections. The developed NDT robot toolbox will ultimately assist NDT technicians to move from a component CAD file to the actual physical inspection, without the need to use multiple pieces of software not optimised for robotic NDT inspections. The commercial driver for this work is the need to decrease NDT inspection times – this has been identified clearly as a bottleneck in existing composite parts manufacture in aerospace industries. The software contains specific functions tailored to generate a tool-path able to follow the contours of curved surfaces, according to a raster scan. The features of this type of path (surface reference edge, offsets, raster step, speed, acceleration, etc.) are fully customizable by the user. Tool paths were generated directly on the triangular mesh imported from the CAD models of the inspected components without need for an approximating analytical surface.

Comparative metrology experiments were undertaken to evaluate the real path accuracy of the toolbox when inspecting a curved 0.5 m^2 and a 1.6 m^2 surface using a KUKA KR16 L6-2 robot. The results have shown that the deviation of the distance between the commanded TCPs and the feedback positions is within 2.7 mm. The variance of the standoff between the probe and the scanned surfaces was smaller than the variance obtainable via commercial path-planning software.

In the future, more versatile versions of the software with additional features could be realised. The ultimate goal of the authors remains the simultaneous management of command coordinates, robot positional feedback and NDT data by an integrated server application running on a single dedicated PC. This paves the way to introducing intelligent novelty factors to the robotic NDT inspections; on-line monitoring and data visualization, real-time path correction and versatile path amending approaches are just some of the possible opportunities.

The current version of the software is available for download for research purposes from http://www.strath.ac.uk/eee/-research/cue/downloads/.

Acknowledgements

This work has been developed in partnership with TWI Technology Centre (Wales), University of Strathclyde (Glasgow), the Prince of Wales Innovation Scholarship Scheme (POWIS) and by IntACom, a project funded by Welsh Government, TWI, Rolls-Royce, Bombardier Aerospace and GKN Aerospace. Additional support was provided with assistance from UK Research Centre in NDE (EP/F017332/1) and EPSRC Equipment Grant “New Imaging Systems for Advanced Non-Destructive Evaluation” (EP/G038627/1).

References

[1] L. Cartz, *Nondestructive testing*, United States: ASM International, 1995.

[2] I. Cooper, P. I. Nicholson, D. Yan, B. Wright, and C. Mineo, “Development of a Fast Inspection System for Aerospace Composite Materials - The IntACom Project,” in 9th International Conference on Composite Science and Technology (ICCST-9), Sorrento (Italy), 2013.

[3] I. Gibson, D. W. Rosen, and B. Stucker, *Additive manufacturing technologies: rapid prototyping to direct digital manufacturing*, New York: Springer, 2010.

[4] Y. Bar-Cohen, “Emerging NDE Technologies and Challenges at the Beginning of the 3 rd Millennium--Part II, Part I,” 2000.

[5] T. Sattar, “Robotic non-destructive testing,” *Industrial Robot: An International Journal*, vol. 37, no. 5, 2010.

[6] M. Schwabe, A. Maurer, and R. Koch, “Ultrasonic Testing Machines with Robot Mechanics - A New Approach to CFRP Component Testing,” in 2nd International Symposium on NDT in Aerospace, Germany, 2010.

[7] P. Louviot, A. Tachattahte, and D. Garnier, “Robotised UT Transmission NDT of Composite Complex Shaped Parts,” in 4th International Symposium on NDT in Aerospace, Berlin (Germany), 2012.

[8] B. Djordjevic, “Remote Non-Contact Ultrasonic Testing of Composite Materials,” in 15th World Conference on Nondestructive Testing, Roma (Italy), 2000.

[9] E. Cuevas, M. López, and M. García, “Ultrasonic Techniques and Industrial Robots: Natural Evolution of Inspection Systems,” in 4th International Symposium on NDT in Aerospace, Berlin (Germany), 2012.

[10] F. Bentouhami, B. Campagne, E. Cuevas, T. Drake, M. Dubois, T. Fraslin, P. Piñeiro, J. Serrano, and H. Voillaume, “LUCIE - A flexible and powerful Laser Ultrasonic system for inspection of large CFRP components.,” in 2nd International Symposium on Laser Ultrasonics, Talence (France), 2010.

[11] A. Maurer, W. D. Odorico, R. Huber, and T. Laffont, “Aerospace composite testing solutions using industrial robots,” in 18th World Conference on Nondestructive Testing, Durban, South Africa, 2012.

[12] J. T. Stetson, and W. D. Odorico, “Robotic inspection of fiber reinforced aerospace composites using phased array UT,” in 40th Annual Review of Progress in Quantitative NDE, Baltimore, Maryland, 2013.

[13] S. G. Pierce, G. Dobie, R. Summan, L. Mackenzie, J. Hensman, K. Worden, and G. Hayward, "Positioning challenges in reconfigurable semi-autonomous robotic NDE inspection." p. 76501C.

[14] C. Mineo, D. Herbert, M. Morozov, S. G. Pierce, P. I. Nicholson, and I. Cooper, “Robotic Non-Destructive Inspection,” in 51st Annual Conference of The British Institute of Non-Destructive Testing, Daventry (UK), 2012.

[15] R. Bogue, “Finishing robots: a review of technologies and applications,” *Industrial Robot: An International Journal*, vol. 36, no. 1, pp. 6-12, 2009.

[16] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, “Recent progress on programming methods for industrial robots,” *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87-94, 2012.
[17] W. Haase, “Automated non-destructive examination of complex shapes,” in 14th Asia-Pacific Conference on NDT (APCNDT), Mumbai, India, 2013.
[18] P. I. Corke, “A robotics toolbox for MATLAB,” *Robotics & Automation Magazine, IEEE,* vol. 3, no. 1, pp. 24-32, 1996.
[19] K. Yoshida, "The SpaceDyn: a MATLAB toolbox for space and mobile robots." pp. 1633-1638.
[20] A. Breijs, B. Klaassens, and R. Babuška, “Automated design environment for serial industrial manipulators,” *Industrial Robot: An International Journal,* vol. 32, no. 1, pp. 32-34, 2005.
[21] G. L. Mariottini, and D. Prattichizzo, “EGT for multiple view geometry and visual servoing: robotics vision with pinhole and panoramic cameras,” *Robotics & Automation Magazine, IEEE,* vol. 12, no. 4, pp. 26-39, 2005.
[22] P. I. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB.*, Berlin: Springer, 2011.
[23] A. Hamzah, “Feasibility study on robot off-line programming and simulation using matlab tools: simmechanics and simulink packages,” Doctoral dissertation, Universiti Malaysia Pahang, 2004.
[24] KUKA, *KUKA.RobotSensorInterface 3.1 Documentation*, 2010.
[25] P. J. Besl, and N. D. McKay, "A method for registration of 3-D shapes." pp. 586-606.
[26] A. Maurer, W. Haase, and W. De Odorico, “Phased array application in industrial scanning systems,” in ECNDT, Berlin (Germany), 2006.