DESIGN AND IMPLEMENTATION OF A MOTION CONTROLLER FOR A SPHERICAL WELDING ROBOT SPECIALLY CONSTRUCTED FOR REPAIRING DAMAGED HYDRAULIC TURBINES USING A 3-D VISION SURFACE MAP

Gerardo Antonio Idrobo Pizo
University of Brasilia, Department of Mechanical Engineering, Automation and Control Group (GRACO), CEP 70910-900, Brasília, DF.
gerardo_idrobo@unb.br

Evandro Leonardo Silva Teixeira
Building 50, Cranfield Univesity, Cranfield, Bedfordshire
MK430AL, Cranfield, England
e.l.teixeira@cranfield.ac.uk

José Mauricio Santos Torres Motta
University of Brasilia, Department of Mechanical Engineering, Automation and Control Group (GRACO), CEP 70910-900, Brasília, DF.
jmmotta@unb.br

Abstract. This paper presents the development of a motion controller to be implemented on a welding spherical robot specially constructed for repairing hydraulic blades surfaces damaged by the cavitation phenomenon. The robot trajectory is to be planned according to the blade surface mapped from a scanning vision sensor in order to fill the cavities found on the damaged blade surface welding in layers. The robot has 5 d.o.f. with two rotary joints, a linear actuator and a pan-tilt wrist. The robot controller allows a 3-D trajectory to be completely specified, i.e., position, velocities and accelerations by using finite-state machines (FSM) to store state variables, a block of combinational logic which determines the state transition for motion synchronization. The FSM are built by using FPGA´s (Field Programmable Gate Arrays). The welding torch trajectories are calculated related to the robot base frame from selected positions taken from the 3D map stored on the robot controller memory by using coordinate transformations from the map coordinate system to the robot base frame, next to the workspace has been fully calibrated by proper procedures of robot calibration. The welding process requires a strict velocity control. Since the welding torch velocities are specified along the trajectory, forward and inverse kinematic models and the Jacobian matrix relating the robot joints velocities to the end-effector´s position, the robot motion can be fully transduced in logical functions to be sent to the FPGA. Simulation and experimental results show that the system developed is fully functional for the purposes aimed in terms of trajectory and velocity accuracy, motion repeatability and processing speed.

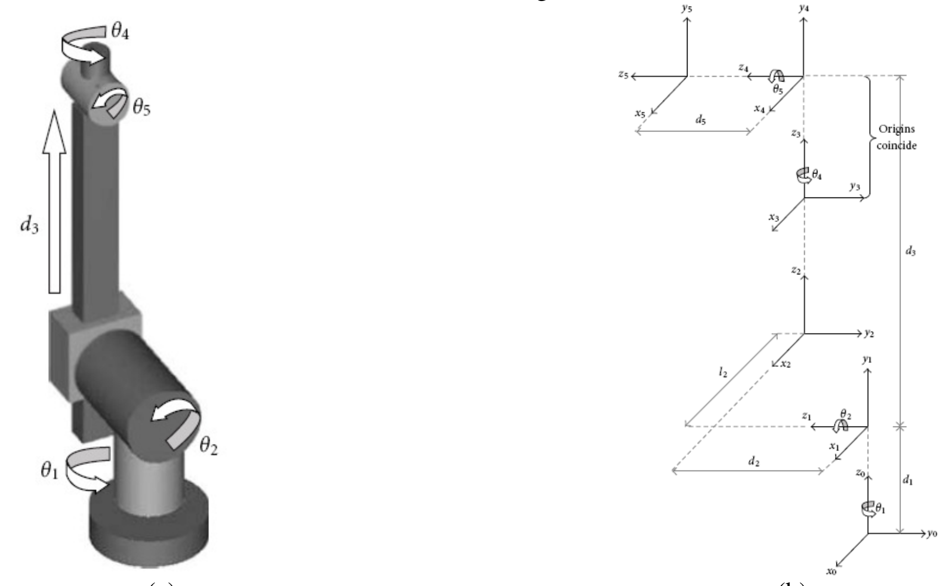
Keywords – Motion Controller, Robot Control, FPGA.

1. INTRODUCTION

Hydraulic turbines installed in hydroelectric plants are subject to several types of mechanical wearing. The sources of mechanical straining can range from operational conditions of the hydro generator, poor design characteristics, properties of the blade material employed and operation points out of specification as a consequence of overloading (Ecober *et. al.*, 2006). This paper presents parts of a R&D project aiming at designing and constructing a prototype of a specialized welding robotic system for recovering material damage on hydraulic turbine blades by welding in layers, with the purpose of repairing erosion produced by cavitation pitting and/or cracked by cyclic loading, reducing human risks and increasing the efficiency of the process. The construction of the robot as a complete system required the prior specification of several characteristics of the process, functions to be performed, constructive aspects, control systems and feedback, applications of electrical and electronic components to robotics, elaboration of mathematical models for actuators and development of hardware and software for control and supervision.

This article describes the steps of the construction of a welding robot control system with five degrees of freedom, driven by step motors. In addition, it presents a proposal for control by using FPGA´s (Field Programmable Gate Arrays) to process, plan and follow-up trajectories based on surface models reconstructed from point clouds. The article also discusses the kinematic model, the construction of 3-D maps, motion control systems on FPGA, the operation of the control system and experimental assessments.

The constructed robot model and schematics are in Fig. 1.



(a) (b)

Figure 1. Constructed robot: (a) model sketch; (b) kinematic model

For a robot to realize all tasks needed in the application proposed it has to be able to fulfill the following requirements: a) capacity to operate in any position: horizontal, vertical or inverted; b) low weight: portability and fixation to the blades; c) rigidity to deflection: load on wrist occurs in any direction and arm extension; d) high motion accuracy: capacity to reach accurately welding regions from the mapped geometry; e) availability of parts in the market; f) control with component interfacing capability; g) topology making feasible to measure large areas with laser scanning and 3d geometry mapping; h) large workspace; i) easiness to be fixed to the turbine blades.

The robotic system proposed has a spherical topology with 5 degrees of freedom, electric stepper motors, rotary and linear actuators and a 2m-diameter workspace. The system has an embedded measurement system with a vision sensor especially built to produce range images by scanning laser beams on the blade surface. The range images are used to construct 3-D models of the blade surface and the location of the damaged spots are recorded into the robot controller in 3-D coordinates, thus enabling the robot to repair the flaws automatically by welding in layers. The robot controller and measurement system are built in FPGA-based reconfigurable microprocessors. The welding process is the GMAW (Gas Metal Arc Welding) using a composite GMAW electrode (tubular metal cored electrode) carried out with a pulsed arc welding machine.

The robotic system was designed to have high rigidity mechanics, easy assembly and fixing on the blade surface and hassle-free operation. Low cost, light weight, portability, high positioning accuracy and repeatability are also characteristics of the resulting robotic system.

The proposed robot has the following characteristics and capacities: spherical topology with 5 degrees-of-freedom, 3 in the manipulator (2 rotation and 1 translation) and 2 in the wrist; surface mapping with only one scanning pass; embedded electronics; MIG/MAG welding; fixation on blades by magnetic devices or by air suction (still to be defined); construction by assembling off-the-shelf parts, allowing construction of several low-cost prototypes using an assembly manual; low cost and time to design.

2. ROBOT KINEMATIC MODEL

A robot can be modeled as a series of links connecting its end-effector to its base, with each link connected to the next by an actuated joint. Robot construction by assembling modular off-the-shelf parts brings about flexibility in manufacturing and allows the construction of several prototypes in the shop-floor, but also produces many sources of assembling inaccuracies. So, model calibration shows up as a very important step in order to achieve the desired accuracy. Robot calibration procedures will not be discussed in this article.

Robot kinematic models are generally based on the well-known Denavit-Hartemberg convention (Paul, 1981) because of its simplicity and easiness to be geometrically represented. The elementary transformations can be formulated as (D-H convention):

$$T = f(\theta, \alpha, d, l) = Rz(\theta).Tz(d).Tx(l).Rx(\alpha) \tag{1}$$

, where T represents position and orientation coordinates of a link frame related to a previous one, θ and α are the rotation parameters, *d* and *l* are translation parameters. The robot (Fig. 1) has perpendicular and parallel axes. However, the D-H convention cannot be used in error parameter models when modeling parallel axes due to singularities that occur in the Jacobian matrix, which produces ill-conditioning in model calibration. This issue is discussed in details in by Motta (2005). A possible convention for parallel axes is the Hayati-Mirmirani (Hayati and Mirmirani, 1985), which cannot be used in perpendicular axes for the same reason. The Hayati-Mirmirani is a four-parameter convention that describes the transformation between two parallel axes as:

$$f(\theta, \alpha, \beta, l) = Rz(\theta).Tx(l).Rx(\alpha).Ry(\beta) \tag{2}$$

The elementary homogeneous transformations representing each of the elementary motions are:

$$Rz = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; Tz = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix}; Tx = \begin{bmatrix} 1 & 0 & 0 & l \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; Rx = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha & -S\alpha & 0 \\ 0 & S\alpha & C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

, where Cθ = cos(θ) and Sθ = sin(θ).

2.1. Forward Kinematics

The end-effector position is calculated from the product of homogeneous transformation matrices which depend on the robot joint coordinates, as well as its geometry, such as link lengths and type of joints (Paul, 1981). Table 1 shows the geometric parameters used to build the model.

Table 1. Kinematic parameters of the robot model. (Fig. 1b)

Joint Variable	Angle	Displacement	Length	Twist
θ	θ_n	d_n	l_n	α_n
θ_1	θ_1	d_1	0	90°
θ_2	θ_2 + 90°	d_2	l_2	-90°
θ_3	0	d_3	0	90°
θ_4	θ_4	0	0	-90°
θ_5	θ_5	d_5	0	0

Table 2. Manipulator Transformation Matrix

$$^0T_5 = \begin{bmatrix} Xx & Yx & Zx & px \\ Xy & Yy & Zy & py \\ Xz & Yz & Zz & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned}
Xx &= (C1.C2.C4 - C1.S2.S4).C5 - S1.S4 \\
Xy &= (S1.C2.C4 - S1.S2.S4).C5 + C1.S4 \\
Xz &= (S2.C4.C5 + C2.S4.C5) \\
Yx &= -(C1.C2.C4 - C1.S2.S4).S5 - S1.C4 \\
Yy &= -(S1.C2.C4 - S1.S2.S4).S5 + C1.C4 \\
Yz &= -(S2.C4 + C2.S4).S5 \\
Zx &= -C1.C2.S4 - C1.S2.C4 \\
Zy &= -S1.C2.S4 - S1.S2.C4 \\
Zz &= -S2.S4 + C2.C4 \\
Px &= (-C1.C2.S4 - C1.S2.C4).d5 - C1.S2.d3 - C1.C2.L2 + S1.d2 \\
Py &= (-S1.C2.S4 - S1.S2.C4).d5 - S1.S2.d3 - S1.C2.L2 - C1.d2 \\
Pz &= (-S2.S4 + C2.C4).d5 + C2.d3 - S2.L2 + d1
\end{aligned}$$

2.2. Inverse Kinematics

From the equations shown in Table 2 the inverse kinematic model can be developed, i.e., the robot joint coordinates can be obtained from the end-effector pose. For simplification, one can define k1 and k2 as

$$k1 = \sqrt{Px^2 + Py^2 - {d_2}^2}\,; \qquad k2 = \sqrt{Px^2 + Py^2 + (Pz - d_1)^2 - {d_2}^2 - {l_2}^2}\,; \tag{4}$$

Then, the joint coordinates can be formulated in explicit variables as:

$$\theta_1 = \tan^{-1}\left(\frac{D_2}{K1}\right) + \tan^{-1}\left(\frac{Py}{Px}\right); \qquad d3 = K2; \qquad \theta_4 = \tan^{-1}\left(\frac{-Zx}{\sqrt{Xx^2 + Xy^2}}\right); \tag{5}$$

$$\theta_2 = \frac{\pi}{2} + \tan^{-1}(\frac{Pz - D_1}{K1}) - \tan^{-1}(\frac{K2}{L2}); \qquad \theta_5 = \tan^{-1}(\frac{-Xz}{\sqrt{Xx^2 + Xy^2}});$$

2.3. Differential Kinematics

The kinematic control of a robotic manipulator requires not only the functional relationships between the robot poses expressed in the robot base coordinate system and joint coordinate system but the functional relationships between the pose derivatives and joint derivatives (i.e. velocities and accelerations). These functions are expressed by the Jacobian matrix (Sciavicco and Siciliano, 2000).

The Jacobian matrix can be formulated for a n degree-of-freedom manipulator by using a vector $p \in R^6 = [p_x, p_y, p_z, \gamma_x, \gamma_y, \gamma_z]$ representing a pose in the tool coordinate frame and $q \in R^n$ representing the robot joint variables, where $p(t) = p(q(t))$ and $q(t) = q(p(t))$. So,

$$\dot{p}(t) = \frac{\partial p(q)}{\partial q} \cdot \dot{q}(t) = J \cdot \dot{q}(t); \tag{6}$$

, where J is the Jacobian matrix

$$J_{6x5} = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} & \frac{\partial p_x}{\partial d_3} & \frac{\partial p_x}{\partial \theta_4} & \frac{\partial p_x}{\partial \theta_5} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \gamma_z}{\partial \theta_1} & \frac{\partial \gamma_z}{\partial \theta_2} & \frac{\partial \gamma_z}{\partial d_3} & \frac{\partial \gamma_z}{\partial \theta_4} & \frac{\partial \gamma_z}{\partial \theta_5} \end{bmatrix}; \tag{7}$$

However, since $n < 6$ it is not possible to find a solution for $\dot{q}(t)$ as a function of $\dot{p}(t)$. So, one of the components of the end-effector orientation vector, $\gamma = [\gamma_x, \gamma_y, \gamma_z]$, has to be kept constant so that J can be inverted. Thus, the joint velocities can be found from:

$$\dot{q}(t)_{5x1} = \frac{\partial q(q)}{\partial p} \cdot \dot{p}(t)_{5x1} = J^{-1}{}_{5x5} \cdot \dot{p}(t)_{5x1}\,; \tag{8}$$

, where J^{-1} is the inverse of J. To include accelerations in both systems one needs to derivate Eq. (6) and Equation (8) once more:

$$\ddot{p}(t) = J \cdot \ddot{q}(t) + \dot{J} \cdot \dot{q}(t); \qquad \ddot{q}(t) = J^{-1} \cdot \ddot{p}(t) + \dot{J^{-1}} \cdot \dot{p}(t); \tag{9}$$

3. 3-D SURFACE MAPPING

The vision sensor specially constructed and used to acquire cloud of points is a laser scanner based system with a high resolution camera and a laser diode to project one light plane on the object and uses computer vision algorithms to extract information directly from the image (Fig. 2) (Ginani and Motta, 2007, Idrobo-Pizo and Motta, 2009). From the geometric distortions of the projected light in the object and knowing the intrinsic camera parameters it is possible to calculate the 3-D coordinates of the object surface points, generating point clouds from the light strips scanning the object surface.

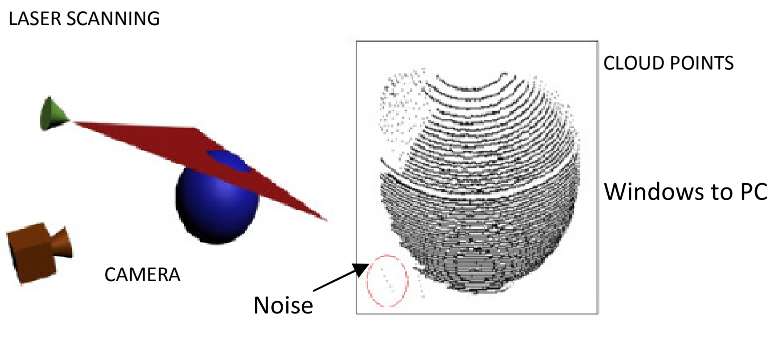


Figure 2. Vision sensor hardware

However, as in any other vision system, the camera and the environment generate noise. To reduce the surface noise the cloud of points can be modeled as the output of an input image I_p, containing the 3-D information from a specific region of the surface (Fig. 3). Image I_p is divided into three components $[x_p, y_p, z_p]$, and filtered by a FIR (finite-duration impulse response) filter with impulse response with finite length N, defined as:

$$h = \begin{bmatrix} h_{0x} & h_{0y} & h_{0z} \\ h_{1x} & h_{1y} & h_{1z} \\ h_{2x} & h_{2y} & h_{2z} \\ \vdots & \vdots & \vdots \\ h_{(N-1)x} & h_{(N-1)y} & h_{(N-1)z} \end{bmatrix}; \tag{9}$$

, where N corresponds to finite filter width.

The output components $[x, y, z]$ are:

$$x_p(n) = h_x I_p(n) + s(n); \qquad y_p(n) = h_x I_p(n) + s(n); \qquad z_p(n) = h_x I_p(n) + s(n); \tag{10}$$

, where $s(n)$ is the random noise that contaminates the I_p point cloud.

To attenuate the random noise, $s(n)$, h can be estimated through performance criteria (w) generated by an adaptive filter:

$$x\acute{}_p(n) = w_x I_p(n); \qquad y\acute{}_p(n) = w_y I_p(n); \qquad z\acute{}_p(n) = w_z I_p(n) \tag{11}$$

, where

$$w = \begin{bmatrix} w_{0x} & w_{0y} & w_{0z} \\ w_{1x} & w_{1y} & w_{1z} \\ w_{2x} & w_{2y} & w_{2z} \\ \vdots & \vdots & \vdots \\ w_{(N-1)x} & w_{(N-1)y} & w_{(N-1)z} \end{bmatrix}; \tag{12}$$

The estimated random noise signal $s(n)$ can be calculated from the contaminated signal from the point cloud (Eq. 10) minus the signal generated by the adaptive filter (Eq.11) as:

$$\begin{aligned} e_x(n) &= x_p(n) - x\acute{}_p(n) = (h_x - w_x) I_p(n) + s_p(n) \\ e_y(n) &= y_p(n) - y\acute{}_p(n) = (h_y - w_y) I_p(n) + s_p(n) \\ e_z(n) &= z_p(n) - z\acute{}_p(n) = (h_z - w_z) I_p(n) + s_p(n) \end{aligned} \tag{13}$$

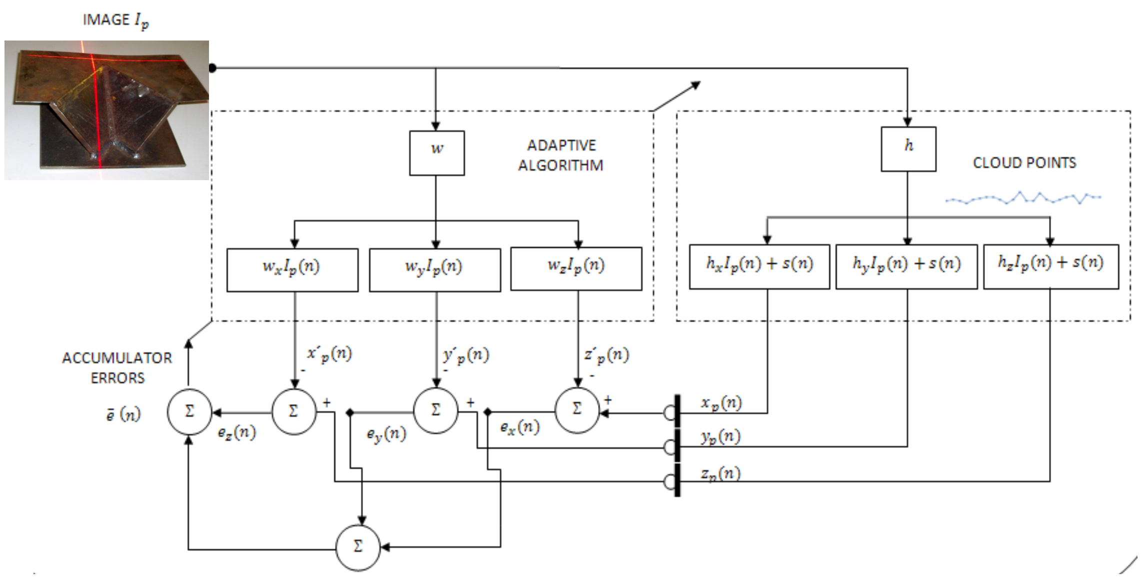


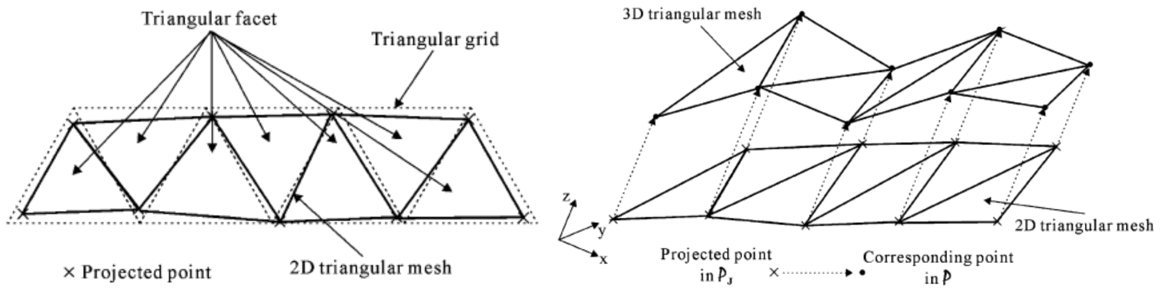
Figure 3. Diagram of the process for attenuating noise in the components $[x, y, z]$ of image (I_p)

In an ideal case, $w = h$ and $e(n) = s(n)$, such as the noise can be completely eliminated. However, the approximation of h by a filter w of finite length restricts full noise elimination.

The value of w is calculated through the LMS (Least Mean Squares) algorithm that uses the Steepest Descent method to find filter weights from the equation given by

$$w(n+1) = w(n) + \frac{1}{2}\mu[-\nabla E\{|e_i{}^2(n)|\}]; \tag{14}$$

, where μ is the step-size parameter and controls the convergence characteristics of the LMS algorithm, $e_i^2(n)$ is the RMS (Root Mean Square) error between the output of each component $i \rightarrow \{x', y', z'\}$ and the reference signal $\{x, y, z\}$. The LMS adaptive algorithm is preferred because of a higher convergence speed (about 10 times higher) than other known algorithms such as Kalman and Wiener filters (Idrobo-Pizo, 2009).



(a) (b)

Figure 4. (a) 2-D triangular mesh constructed from the projected points next to nodal points repositioning. (b) 3-D triangular mesh is constructed based on its corresponding 2-D triangular mesh.

The algorithm used to organize the point clouds in triangular meshes is divided into three steps, and each step is divided into several sub-steps (Chui et al, 2008): (a) Projection of the cloud of points on the mesh grid, (b) Nodal point repositioning, (c) Reconstruction of the 3-D triangular mesh from the 2-D triangular mesh. Details of the first two steps can be found in Idrobo-Pizo and Motta (2009) and consist in projecting the 3-D point cloud on a plane and constructing a 2-D triangular mesh among the projected points eliminating all points but the closest to the mesh nodes. The mesh nodes are moved to the closest point of each node. The 3-D triangular mesh can be reconstructed next to the process of moving the nodes of the 2-D mesh (Fig. 4a) by using the depth values of each displaced nodal point (Fig. 4b).

3.1. Path Generation

Path generation aims to relate joint positions as a function of time to ensure an adequate control of velocities and accelerations of the end-effector during the robot motion. Good quality welding procedures require a strict speed control of the welding torch motion.

Path generation algorithms can model a trajectory by using polynomial functions (Tonetto, 2007). In this project polynomial functions of the fifth order were used, which ensures that the motion along the current trajectory points have continuity at the velocity specified. The motion of the joint *i* is defined by the polynomial below, i.e., its position, velocity and acceleration as a function of time:

$$\theta_i(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5;$$
$$\dot{\theta}_i(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4;$$
$$\ddot{\theta}_i(t) = 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3; \tag{15}$$

, where θ_i is the joint *i* position, *t* is the time and $a_0 \ldots a_5$ are coefficients. With six parameters available one can impose six boundary conditions between two consecutive points in the trajectory:

- Initial condition: position $\theta_0(t)$, velocity $\dot{\theta}_0(t)$ and acceleration $\ddot{\theta}_0(t)$
- Final condition: position $\theta_f(t)$, velocity $\dot{\theta}_f(t)$ and acceleration $\ddot{\theta}_f(t)$

The six equations arising from the boundary conditions solve all six coefficients, a_0,…, a_5 as in Eq. (16) and define the function that describes the path that the joint must follow as a function of time when inserted in Eq. (15).

To program a desired trajectory the three-dimensional map is used to specify a starting point and a final point in the path and the correspondent time associated to each point to be reached by the welding torch during the robot motion according to the specified velocity. The positions of each joint at the correspondent time are calculated from the inverse kinematics from Eqs. (5-8).

$$a_0 = \theta_{i0}(t); \qquad a_1 = \dot{\theta}_{i0}(t); \qquad a_2 = \frac{\ddot{\theta}_{i0}(t)}{2};$$

$$\begin{aligned} a_3 &= \frac{[20(\theta_{if} - \theta_{i0}) - (8\,\theta_{if} + 120\dot{\theta}_{i0})t + (\ddot{\theta}_{if} - 12\ddot{\theta}_{i0})t^2)]}{2t^3} \\ a_4 &= \frac{[-30(\theta_{if} - \theta_{i0}) + (14\,\dot{\theta}_{if} - 16\dot{\theta}_{i0})t + (-2\ddot{\theta}_{if} + 3\ddot{\theta}_{i0})t^2)]}{2t^4} \\ a_5 &= \frac{[12(\theta_{if} - \theta_{i0}) - (6\,\dot{\theta}_{if} + 6\dot{\theta}_{i0})t + (\ddot{\theta}_{if} - \ddot{\theta}_{i0})t^2)]}{2t^5} \end{aligned} \tag{16}$$

4. FPGA-BASED MOTION CONTROL SYSTEM

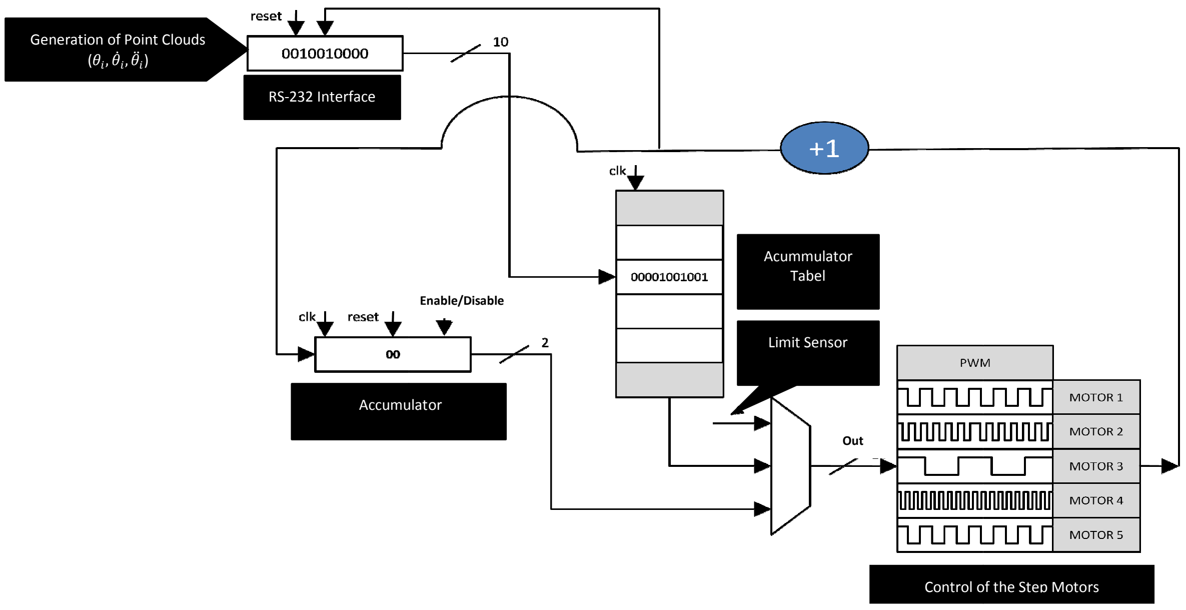


Figure 5. Block Diagram of the Robot Controller System.

The robot controller was built with reconfigurable architectures based on FPGAs. The main advantage to use FPGAs is their capacity for embedding complex systems, including microprocessor and specific application hardware using only one device. This allows the system to improve its performance to execute complex algorithms. They provide arrays of configurable logical cells (CLBs) that can be configured to perform given functions by means of configuration bitstreams. The bitstreams are generated by a software tool, and usually contains the configuration information for all the components. Configuration of the FPGA can be realized by using several design/synthesis tools provided by companies such as Altera (Altera, 2011) and Xilinx (Xilinx, 2011). The circuit can be described using a high level description language such as VHDL or Verilog. The tool makes the synthesis of the description file producing a binary configuration file (bitstream file). This file is used to configure the FPGA device.

FPGA's allows the implementation of algorithms directly in hardware instead of in software (for instance, using microcontrollers). Implementation in software has limitations due to the sequential nature of von Newmann architectures that run the software model. In contrast, in FPGA implementations the potential of parallelism of the algorithms can be explored in order to improve the performance of the system.

The control system design of the five degrees of freedom robot in FPGA was carried out by using the programming tools Visual C++ and VHDL (Very High Speed Integrated Circuit Hardware Description Language) in the Project Navigator from Xilinx, where the synthesis, simulation and implementation of the software are performed on a FPGA device (Field Programmable Gate Array).

The robot programming process is carried out from a PC-computer by using its RS-232 serial port. Since the control unit receives data of position and velocity, it activates the step motors.

Each step motor has limit Hall effect-based sensors (magnetic proximity sensors). This sensory subsystem is very important since it sends to the control system a reference point to start motion and a range bound that the motor can actuate.

A block diagram of the system developed to control the robot's motion is showed below (Fig. 5).

The robot programming process starts with the selection of the path to be followed by the welding torch, from the initial to the end points. The path is modeled into the computer using the 3-D surface map generated by robot vision system (Feng and Teng, 2005).

The path specified is then sampled in short intervals, Δx (Fig. 6) so that the robot displacement is performed linearly point-to-point using differential kinematic equations. The value of Δx depends exclusively on the motion accuracy needed.

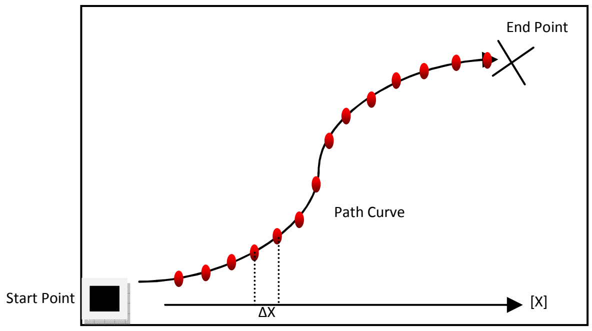


Figura 6. Sketch of a robot path curve.

The functions of each subsystem and the I-O data involved are discussed in the next sections.

a. Generation of Point Clouds

This module receives the 3-D spatial coordinates of the points belonging to the selected path in the surface 3-D map coordinate system and calculates the transformation from the point coordinates represented in the robot base reference system into the robot joint coordinate system, as angles θ_{in} and displacement joint variables, making use of inverse kinematics described in item 2.3.

Moreover, the Jacobian matrix relating spatial coordinate velocities to joint coordinate velocities are calculated by using the kinematic differential equations. These values are stored in an array of 1024 bits and sent to the FPGA through the asynchronous serial port RS-232 with a baud rate of 9600 bps.

b. FPGA-Computer Interface

SPARTAN-3 module has a MAX-232 integrated circuit whose function is to convert the input/output signals of the FPGA to the RS_232 standard. Through this Protocol the FPGA receives an array of 1024 bits corresponding to the values of joint angles (θ_{1n}, θ_{2n}, θ_{4n}, θ_{5n}) and displacement (d_{3n}) and their respective velocities ($\dot{\theta}_{1n}$, $\dot{\theta}_{2n}$,$\dot{d}_{3n}$, $\dot{\theta}_{4n}$, $\dot{\theta}_{5n}$). In addition, it performs real-time monitoring of point-to-point positions of the robot joints during the planned path.

c. Accumulator

The function of this circuit is to store the data from the computer in the FPGA memory and save them in a database address. This module is needed to eliminate delays in communication between the computer and the drivers of the motors, so that the data are stored successively in four registers, each one of 1024 bits.

d. Control of the Step Motors

This module is responsible for generating control signals for the step motors through a function that converts joint angles values (θ_{1n}, θ_{2n}, θ_{4n}, θ_{5n}) and joint displacements (d_{3n}) and their respective velocities ($\dot{\theta}_{1n}$, $\dot{\theta}_{2n}$,$\dot{d}_{3n}$, $\dot{\theta}_{4n}$, $\dot{\theta}_{5n}$) and accelerations ($\ddot{\theta}_{1n}$, $\ddot{\theta}_{2n}$,$\ddot{d}_{3n}$, $\ddot{\theta}_{4n}$, $\ddot{\theta}_{5n}$) in signals at intervals of time and frequency to activate the motors.

5. ROBOT OPERATION STEPS

The robot operation steps can be described sequentially in the following steps:

1. The affected region of the turbine blade is selected visually.
2. The welding speed is selected according to the welding plan.

3. The robot is moved for the vision sensor to scan the surface of the affected area.
4. The trajectory of the welding torch is defined by using welding strategy special algorithms (not developed yet) for welding in layers.
5. Few milliseconds are needed so that the computer transforms the spatial coordinates from the 3-D map to angles (θ_n) and displacements (d_n) of the robot joints using inverse and differential kinematics.
6. An instruction is sent to the FPGA with coordinate information to activate the motors further.
7. Each instruction of position, velocity and acceleration is sent to the motors and a new package of instructions is demanded from the computer.
8. Once the welding torch gets to the trajectory end point the robot joints are halted and new spatial coordinates are awaited corresponding to a new welding bead trajectory.

6. EXPERIMENTAL RESULTS

This section shows experimental assessments to illustrate the system operation. Figure (7) shows the robot moving to a selected area for the vision system to scan the object. In Fig. (8) a trajectory is chosen for the robot welding torch to follow. Fig. (9) shows the robot following the planned trajectory using a pen in the place of the welding torch.

Figures (10) and (11) show a plot of a sample of a robot trajectory and error projected on plane X-Z.

It is possible to verify that there are position errors between the planned trajectory from the image (T_v) and the trajectory followed on the object piece (T_p); the error between the two paths point-to-point can be calculated by the sum of the squared differences (SSD) as:

$$\Delta E = \frac{1}{N}\sum_{i=1}^{N}(T_{vi} - T_{Pi})^2 \tag{17}$$

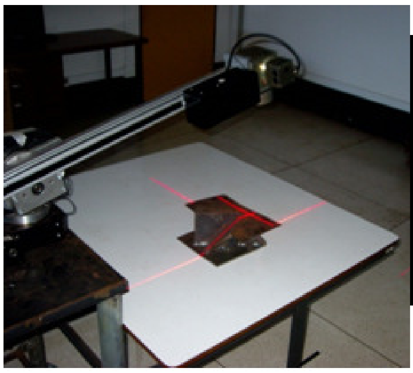


Figura 7. Robot Constructed

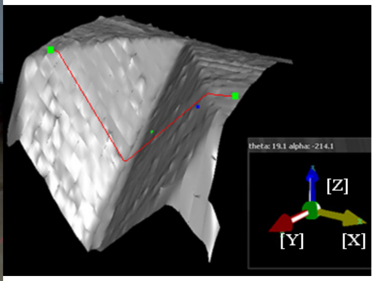


Figura 8. Selected Path.

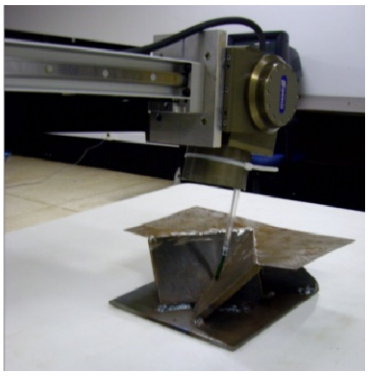


Figura 9. Robot path following.

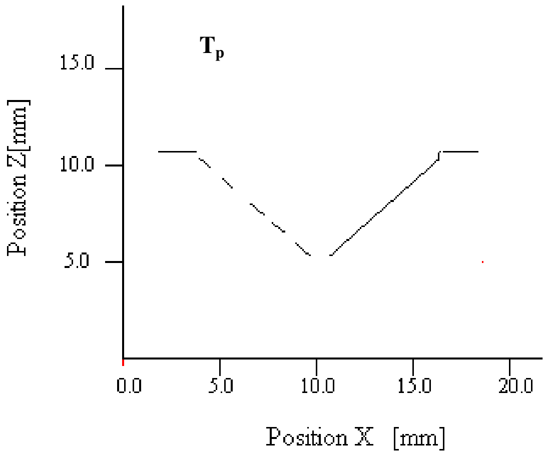


Figure 10. Path followed by the robot in the X-Z plane(Tp)

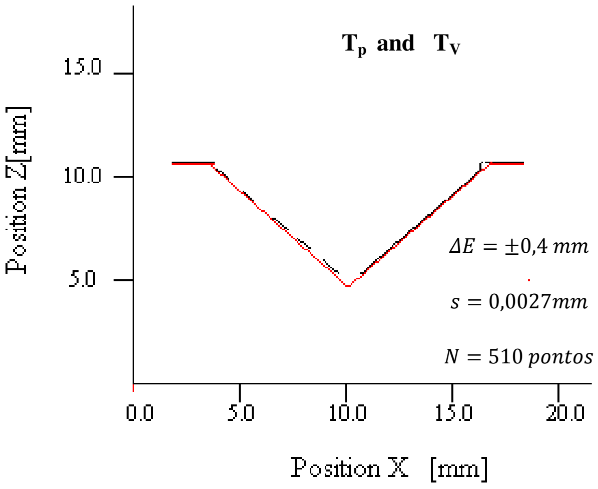


Figure 11. RMS error in the X-Z plane. ΔE(squared differences), s(standard deviation) and N(degrees of freedom)

7. CONCLUSIONS

The paper presents a proposal for a methodology for the path planning process and motion control of a robot manipulator constructed to repair hydraulic turbine blades by welding in layers and using FPGA-based logic processors. Besides, it was briefly discussed a computer application for surface reconstruction and tool path planning from a set of points specially developed for this application.

In this project a trajectory graphical simulator was built specially to simulate the robot motion before the real task. This was very useful because of the possibility of collisions and singularities. Since each joint position can be checked in simulation when the robot follows a specified trajectory the computer code can be generated with confidence for the robot to perform the real motion.

The use of FPGA devices allows more flexible control characteristics and to describe the hardware required to program the functions. The system is not limited to the capacities of the resources of the standard programmable devices such as microprocessors, micro-controllers, signal processors, among others. Additional enhancements are being investigated in order to refine the parameters calculated and to improve conditions for variable quantization in FPGA. In addition, other techniques are being studied to improve the robustness of the synchronization scheme developed.

The project is still in the testing phase. Just by checking if the RMS of the positional error was sufficiently small is not enough to determine whether the proposed control scheme can be used to successfully perform a welding task. In principle, good results for position and motion accuracy were obtained ensuring confidence in repairing damaged turbine blades by welding in layers with the robotic system developed.

8. ACKNOWLEDMENTS

ELETRONORTE (Electrical Power Plants of the North of Brazil) and FINATEC (Scientific and Technological Enterprises Foundation) are thanked for their financial support and collaboration. Our gratitude is especially conveyed to the staff of ELETRONORTE for their technical support and interest.

9. REFERENCES

Altera, 2011. <www.altera.com>.

Chui, K.L., Chiu, W.K., Yu, K.M., 2008 "Direct 5-Axis Tool-path Generation from Point Cloud Input Using 3D Biarc Fitting", Robotics and Computer-Integrated Manufacturing, Vol.1, pp 270–286.

Ecober, X., Egusquiza, E., Farhat, M., Avellan, F., Coussirat, M., 2006, "Detection of Cavitation in Hydraulic Turbines", Mechanical Systems and Signal Processing, Vol. 20, Issue 4, p.p.- 983-1007.

Feng, H.-Y., Teng, Z., 2005, "Iso-planar piecewise linear NC tool path generation from discrete measured data points". Computer-Aided Design, Vol. 37, Elsevier Ltd, pp. 55–64

Ginani, L. S. and Motta, J. M. S. T., 2007, "A Computer System to Calibrate Industrial Robots Using a Measurement Arm", Proceedings of the 19th International Congress of Mechanical Engineering, Vol. 1, Brasilia-DF, Brazil, pp. 1-10.

Hayati, S. and Mirmirani, M., 1985, "Improving the Absolute Positioning Accuracy of Robots Manipulators". Journal of Robotic Systems, Vol. 2, No.4, 397-413.

Idrobo-Pizo, G. A and Motta, J. M. S. T., 2009, "3D Surface Generation from Point Clouds Acquired from a Vision-Based Laser Scanning Sensor", Proceedings of the 20th International Congress of Mechanical Engineering, Vol. 1, Gramado-RS, Brazil, pp. 1-10.

Motta, J.M.S.T., 2005, "An Investigation of Singularities in Robot Kinematic Chains Aiming at Building Robot Calibration Models for Off-line Programming", Journal of the Brazilian Society of Mechanical Sciences and Engineering, Vol. 2, No. 2, pp. 200-204.

Paul, R. P., 1981, "Robot Manipulators - Mathematics, Programming, and Control", Boston, MIT Press, Massachusetts, USA, 279 p.

Sciavicco, L. and Siciliano B., 2000, "Modelling and Control of Robot Manipulators", 2nd Ed., Springer, London, UK, 377p.

Tonetto, C. P., 2007, "Uma proposta de sistematização do processo de planejamento de trajetórias para o desenvolvimento de tarefas de robôs manipuladores". Master Dissertation, Federal University of Santa Catarina, UFSC, Brazil.

10. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.