A programming system for welding robots

Nikoleris, Giorgos

Published in:
International Journal for the Joining of Materials

1990

A programming system for welding robots

Dr. Giorgos Nikoleris
Robotics Group, Dept. of Production and Materials Engineering,
Lund University, Lund, Sweden

(*Int. J. for the Joining of Materials*, 2(2):55-61, 1990. ref:nikoleris90a)

A Programming System for Welding Robots

G. Nikoleris
Lund University, Lund, Sweden

(received in final form: May 8, 1990)

Abstract

This article covers some aspects on the programming of welding robots emphasizing on the process requirements. A programming system based on the Pascal programming language is presented as well as an interface to CAD/CAM-systems.

1 Introduction

Since the introduction of the first teach-and-repeat manipulators in the early 1960's, industrial robots have been used for many different manufacturing tasks, including spot and arc-welding, spray painting, deburring and automatic assembly. These tasks are defined by programs that instruct the system controller to move the end-effector of the robot along described paths and to take several peripheral actions. The robot program defines manipulator motions and includes statements to control the process equipment in use, to activate devices like fixtures and clamps, to make the necessary interactions with the environment through sensors and to communicate with other computer controlled equipment.

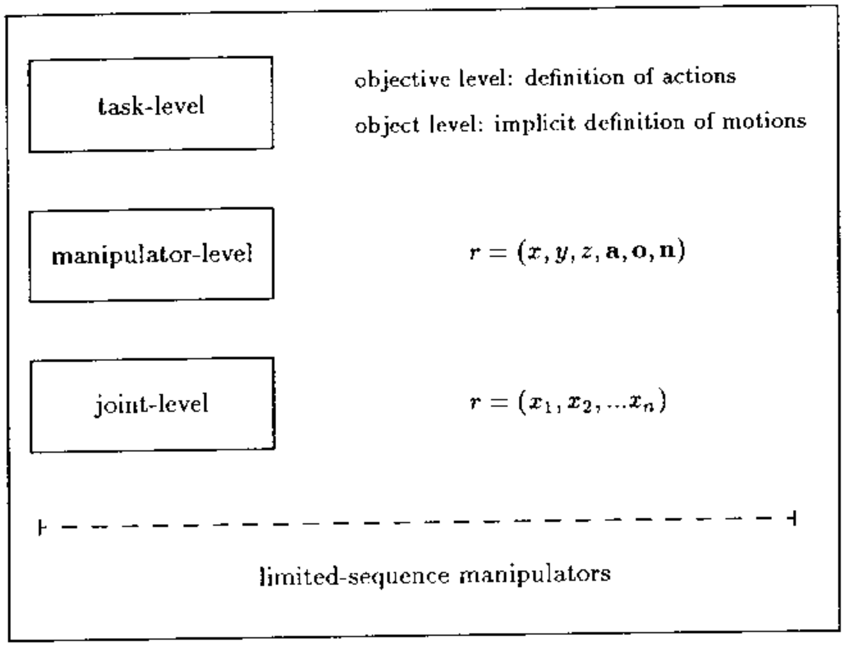
Robot programming is a major research topic in robotics. Different languages have been developed in order to make robot programming easier and to help programmers to exploit the inherent versatility and flexibility of the industrial robots. The industrial robot is an essential manufacturing unit in the factory of the future, where high flexibility and short set-up times are necessary. Thus, the methods for producing robot programs must be as flexible and effective as the industrial robot itself.

The difficulties in programming an industrial robot originate in the dual nature of the robot program: a logical structure with controlling statements that has to be developed at the same time as spatial relations or motions are defined. The majority of industrial robots are currently programmed by taking them out of production and leading them through their new task, a technique known as *on-line* or *teach* programming. In this way the robot controller is used by a low-level programming system that mostly takes care of the description of robot movements.

The design of robot languages has been strongly influenced by the potential users and the application area of the robotic system. The users of robot programming languages can be either *end users* or *application developers*. End users such as robot operators usually describe the operations to be carried out by the robot. End users have normally little programming skill and make best use of high-level systems that do not include operations like robot modelling or sensor system specification. Application developers on the other hand develop modules with specific facilities in order to increase the ease of use for a particular class of end users, while decreasing the generality of the system. Application developers combine greater programming skill with a deeper knowledge of the application area. The systems that are best used by application developers must allow for low-level system-specific facilities and at the same time employ the programming environment of a high-level language. A typical problem area that application developers are likely to work with is the programming of coordinated motions or the integration of tactile and vision sensors. This mix of users and the way they work have had an important influence on the language design, see Fig. 1.

To make programming more efficient researchers have taken different approaches. Research is divided into three areas: *Textual programming languages, simulation systems* and *implicit programming.* Textual languages focus on the description of robot motions as relations between objects, the interface of multiple sensor systems and the execution of peripheral actions using a formal programming language. Simulation systems are used to validate and to verify the generated programs as well as to choose an optimal robot configuration. Finally, research in implicit robot programming takes a top-down approach to the logical structure of the robot programs, using high-level statements that specify the execution of tasks rather than explicit operations.

Most of the research in implicit robot programming is



task-level
objective level: definition of actions
object level: implicit definition of motions
manipulator-level
$r = (x, y, z, \mathbf{a}, \mathbf{o}, \mathbf{n})$
joint-level
$r = (x_1, x_2, ... x_n)$
limited-sequence manipulators

Fig. 1. Abstraction levels of robot programming languages.

devoted to the development of high-level assembly operations. In these robot tasks are specified by the spatial relations of the objects to be assembled prior and after the assembly operation. Less attention has been paid to the programming of process robots. In many process applications the logical structure of the robot program is in essence quite simple. The robot is programmed to move along a path while the process equipment is activated with the proper process parameters. Leadthrough programming is sufficient in such applications. However, the tasks that a robot program must perform may increase significantly if the industrial robot becomes a part of an integrated manufacturing unit. These tasks include general administrative functions, coordinated operation between several manipulators and integration with external sensor systems. Complexity may increase even if only small units are integrated, as in the case of a welding robot and a positioner. In this example, in order to define relatively simple motions, such as circular welds on an arbitrary plane, the operator has to teach a large number of poses to the controller.

In the case of welding robots operating in a computer integrated environment programs have to provide for communication with other computers for program selection and data retrieval, for part identification, for quality assurance functions and for general statistics. The robot controller has to be able to communicate with sensor systems and other computer controlled manufacturing units. This administrative overhead can be reduced using higher level languages that support these operations as language statements or functions. In the case of coordinated motions between two computer controlled devices, the robot language must support object defined motion, e.g. motions defined in an object coordinate system rather than in the robot coordinate system. Another area where high level programming may be beneficial is in the proper choice of process parameters.

The motions of a industrial robot as well as the welds that have to be accomplished are usually programmed as a series of linear movements. Additional information as travelling speed, welding current and voltage as well as signals to automated jigs and welding positioners is also added to the program.

2 Robot Programming System

2.1 The structure of the system

The programming system (see Fig. 2) is based on procedures that can be called in order to create a robot program or monitor the system under program execution. A more limited user-friendly robot language interface is provided for the casual user or the robot operator. Robot motions are defined either from a graphical system or using the robot system.

The system provides facilities to define movements using local object frames or frame combinations and object information such as position vectors and surface normals.

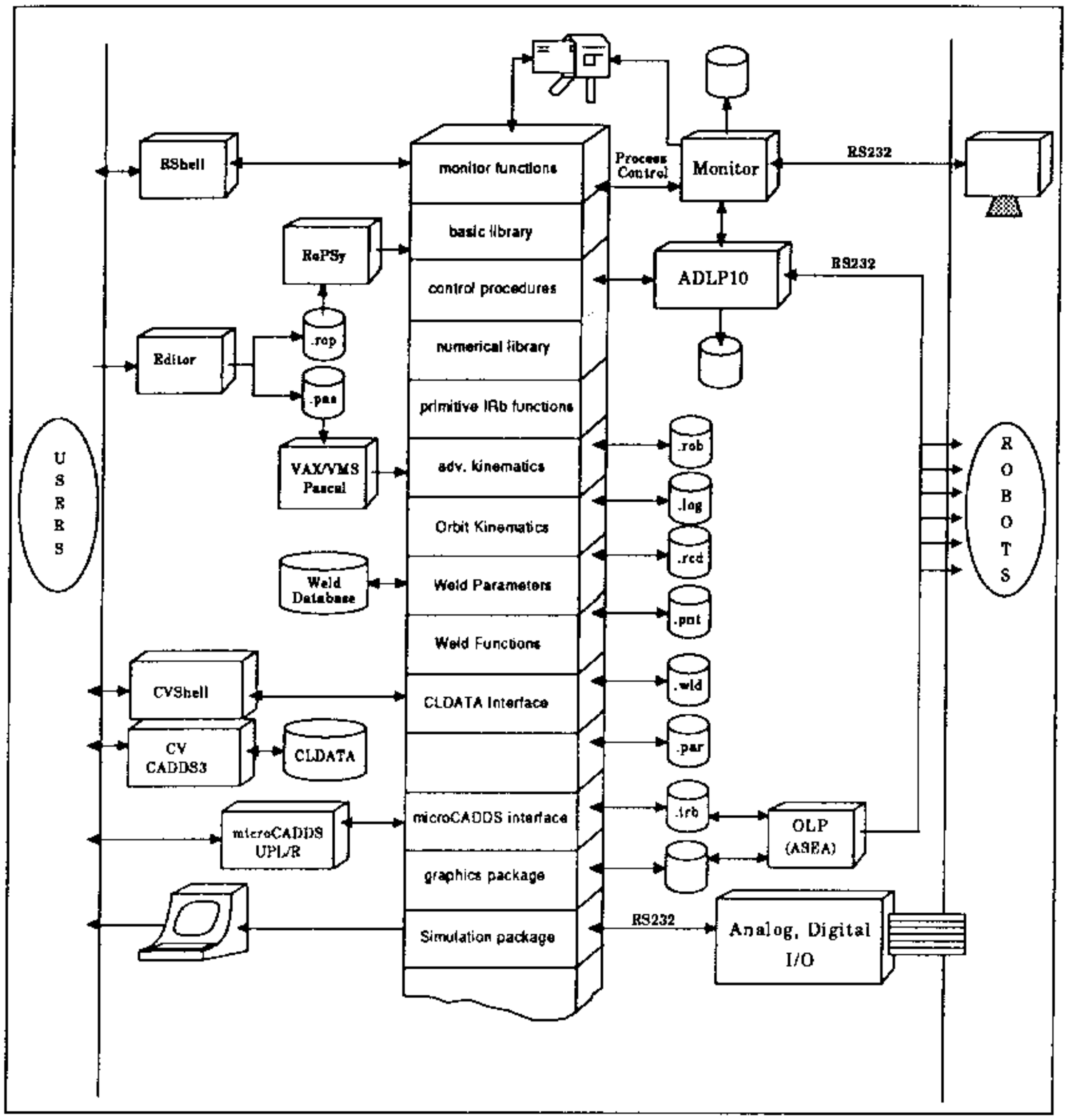


Fig. 2. The structure of the robot programming system.

2.2 Defining the geometry of the welds

The motion of a welding robot has to be defined using two vectors. The position vector describes the movement of the electrode tip and the approach vector describes the orientation of the welding gun in space. Two vectors and five degrees of freedom is sufficient for welding since the welding gun is an axially symmetrical body.

Robot motion is then accomplished as a series of linear movements of the electrode tip along a path defined by a number of position vectors. The orientation of the welding gun is defined by an equal number of approach vectors. Robot motion is accomplished by interpolation between two defined orientations.

In order to use a graphical system, robot motion has to be defined using relations to objects rather than explicit robot movements. The position and the orientation of the welding gun are therefore derived from a surface that describes the welds to be made by the robot. The use of a modelled surface is not imperative. The surface may be defined implicitly by two parallel lines that describe the weld. The surface is then divided into a number of plane, polygon surfaces that approximate the actual weld path. The orientation vector is derived using a relation to the normal to the defined planes, see Fig. 3.

3 Kinematics

One of the fundamental problems in robot programming is the efficient representation, analysis and solution of the

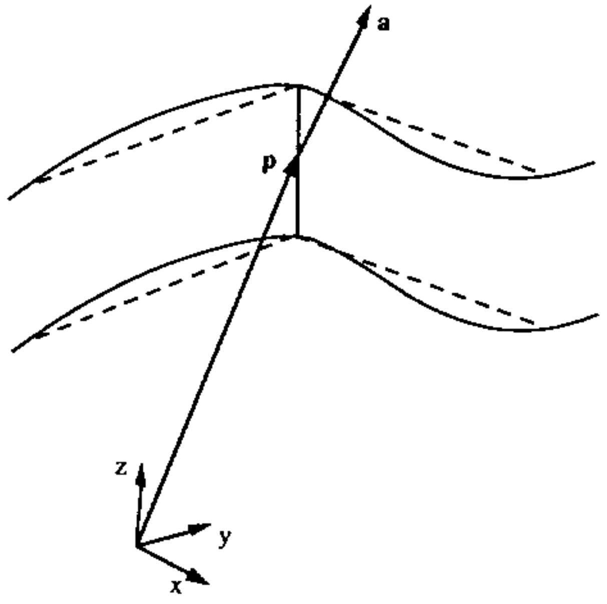


Fig. 3. The position and orientation vector of the welding gun.

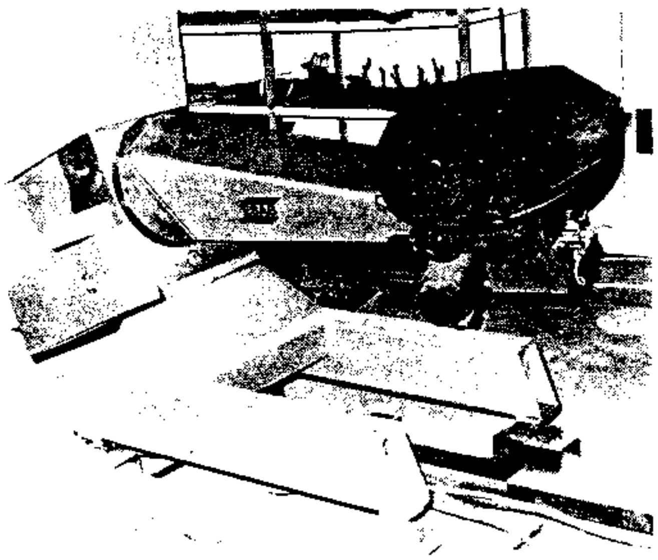


Fig. 4. The Esab positioner, type MHA 500 A.

kinematics of a robotic system with one or more manipulators. Several approaches in this problem are described in the literature. The most known are analytical and geometrical solutions based on the Denavit-Hartenberg representation parameters. Another approach is an iterative solution which provides more generalized solutions for different types of robots. Other authors have presented solutions based upon special polygon closures and 3×3 dual matrices.

3.1 The Kinematics of a Welding Positioner

The use of welding positioners increases the potential of a welding station. Positioners are used in manual and semi-automatic welding in order to obtain better weld quality as well as higher deposition rate and safety.

A welding positioner is constructed so that it can handle bulky and heavy objects. In order to avoid large load moments *universal balance* positioners are constructed so that the intersection point of their axes lies near the center of gravity of the welding object (Fig. 4). Other positioner types are the *turning rolls* type with one revolute horizontal axis and the gear-driven tilt and roll type (Fig. 6).

A computer controlled positioner that can move synchronously with the welding robot permits the non-stop welding of geometrically more complicated joints. However, the programming of such welds is a difficult and time consuming operation that involves the manual specification of a large number of points. An efficient programming method should make use of the description of the motions of the welding gun relative to the welding object. In order to implement such a programming method we must derive the coordinated movements of the manipulator and the positioner. The kinematics of a welding positioner have been studied in [1]. In this section an analytical solution for a universal positioner is presented.

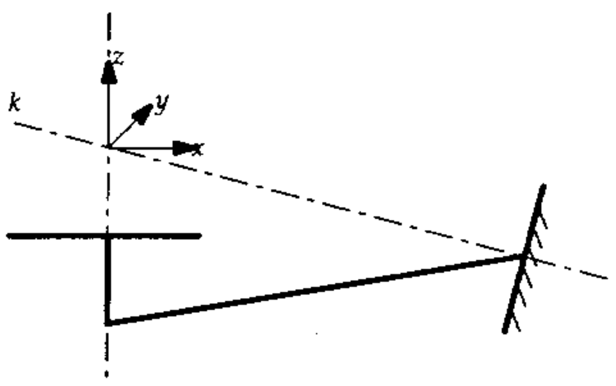


Fig. 5. The base frame of a universal positioner.

The base frame of the universal positioner is located in the intersection of its two axes and is aligned as shown in Fig. 5. The kinematics of a welding positioner describe the transformation of the weld joint parameters (the position vector, the normal vector and the velocity) between the local coordinate system of the object and the base frame of the positioner. The weld joint parameters describe the location and the orientation of the *weld frame*.

The transformation of the weld frame is given by

$$W_{base} = \mathrm{Rot}(k,\phi_1)\,\mathrm{Rot}(z,\phi_2)W_{object} \tag{1}$$

or taking into account that $k_y = 0$ and that $k_x^2 + k_y^2 + k_z^2 = 1$

$$W_{base} = \begin{bmatrix} k_x^2 C\phi_2 + k_z^2 C\phi_1 C\phi_2 - k_z S\phi_1 S\phi_2 & -k_x^2 S\phi_2 + k_z^2 C\phi_1 S\phi_2 - k_z S\phi_1 C\phi_2 & k_z k_x V\phi_1 \\ k_z S\phi_1 C\phi_2 + C\phi_1 S\phi_2 & -k_z S\phi_1 S\phi_2 + C\phi_1 C\phi_2 & -k_x S\phi_1 \\ k_x k_z V\phi_1 C\phi_2 + k_x S\phi_1 S\phi_2 & -k_x k_z V\phi_1 S\phi_2 + k_x S\phi_1 C\phi_2 & k_z^2 V\phi_1 + C\phi_1 \end{bmatrix} W_{object} \tag{2}$$

where $V\phi = \mathrm{vers}\phi = 1 - \cos\phi$, $C\phi = \cos\phi$ and $S\phi = \sin\phi$.

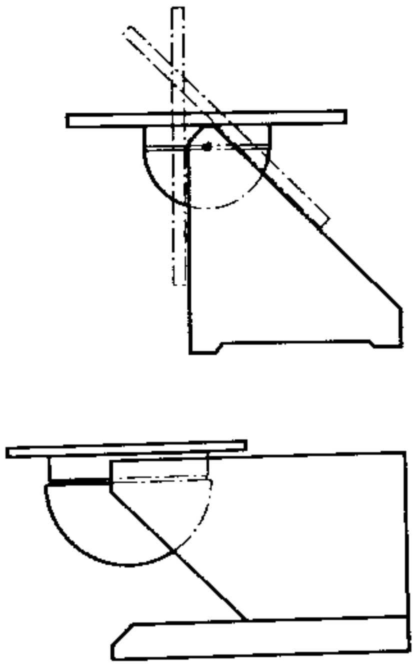


Fig. 6. Gear-Driven Roll-Tilt Positioners.

We can now formulate the direct kinematics problem of a positioner as follows:

> *Given the values of the joint parameters of a positioner, find the the position vector, the normal vector and the velocity vector of a weld in the positioner base frame.*

Since the positioner has only two axes we can not expect to solve the inverse problem in a general way. However, we are able to align the weld joint in some known direction, under restrictions that occur when the normal vector of the weld is aligned with the last axis of the positioner. The inverse kinematics problem can now be formulated as follows:

> *Given the position vector and the normal vector of a weld in the positioner base frame, find the values of the joint parameters that align the normal vector* **a** *of the weld in a new direction* **a**$'$.

The joint angles can be found from equation 1. Premultiplying with $\mathrm{Rot}(k,\phi_1)^{-1}$ yields

$$\begin{bmatrix} k_x k_x \,\mathrm{vers}\,\phi_1 + \cos\phi_1 & -k_z \sin\phi_1 & k_z k_x \,\mathrm{vers}\,\phi_1 \\ k_z \sin\phi_1 & \cos\phi_1 & k_x \sin\phi_1 \\ k_x k_z \,\mathrm{vers}\,\phi_1 & -k_x \sin\phi_1 & k_z k_z \,\mathrm{vers}\,\phi_1 + \cos\phi_1 \end{bmatrix} \begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} \tag{3}$$

$$= \begin{bmatrix} \cos\phi_2 & -\sin\phi_2 & 0 \\ \sin\phi_2 & cos\phi_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

From equation 3 we obtain

$$a'_x k_x k_z \,\mathrm{vers}\,\phi_1 - a'_y k_x \sin\phi_1 + a'_z(k_z^2 \,\mathrm{vers}\,\phi_1 + \cos\phi_1) = a_z \tag{4}$$

Substituting

$$\cos\phi_1 = \frac{1 - \tan^2\frac{\phi_1}{2}}{1 + \tan^2\frac{\phi_1}{2}} \quad , \quad \sin\phi_1 = \frac{2\tan\frac{\phi_1}{2}}{1 + \tan^2\frac{\phi_1}{2}}$$

yields

$$\tan^2\frac{\phi_1}{2} - 2\frac{k_x a'_y}{\beta} - \frac{a_z - a'_z}{\beta} = 0 \tag{5}$$

where $\beta = 2k_x(k_x a'_z + k_z a'_x) - (a'_z + a_z)$.

The first joint parameter can now be obtained as

$$\phi_1 = 2\arctan\frac{k_x a'_y \pm \sqrt{k_x^2 a'^2_y + (a_z - a'_z)\beta}}{\beta} \tag{6}$$

The second joint parameter can now be obtained from equation 3 as

$$\phi_2 = 2\arctan\frac{a_x \pm \sqrt{a_x^2 - a_y^2 + \gamma^2}}{a_x + \gamma} \tag{7}$$

where $\gamma = a'_x k_z \sin\phi_1 + a'_y \cos\phi_1 + a'_z k_x \sin\phi_1$. Valid solutions must satisfy the third equation in the system 3.

3.2 The Jacobian of the Positioner

The next stage involves the calculation of the velocity and the acceleration of a point on the positioner if the rates of change of the joint parameters $(\dot{\phi}_1, \dot{\phi}_1)$ are known. The velocities of a manipulator are often presented in the form of a multidimensional derivative that is called the *Jacobian* matrix of the manipulator. The Jacobian relates the velocities of the joints to the Cartesian velocity of a moving point of the manipulator.

In the case of the welding positioner the Jacobian is a 6×2 matrix, relating the six components of the Cartesian velocity vector $(\boldsymbol{v}, \boldsymbol{\omega})$ of the weld to the 2×1 velocity vector of the joints.

$$\begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} J_{v1} & J_{v2} \\ J_{\omega 1} & J_{\omega 2} \end{bmatrix} \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix} \tag{8}$$

The angular velocity of a point of the moving plate of the positioner as seen by an observer in fixed space can be written as

$$\boldsymbol{\Omega}_2^0 = \boldsymbol{\Omega}_1^0 + \boldsymbol{\Omega'}_2^1 \tag{9}$$

where $\boldsymbol{\Omega}_1^0$ is the angular velocity of the first link of the positioner and $\boldsymbol{\Omega'}_2^1$ is the velocity component that is caused by the rotation of link 2. Using rotation matrices equation 9 becomes

$$\boldsymbol{\Omega}_2^0 = \boldsymbol{\Omega}_1^0 + \mathrm{Rot}_2^0\,\boldsymbol{\Omega'}_2^2\,\mathrm{Rot}_0^2 \tag{10}$$

Equation 9 can be written in vector form as

$$\boldsymbol{\omega}^0 = \begin{bmatrix} k_x\dot{\phi}_1 \\ 0 \\ k_z\dot{\phi}_1 \end{bmatrix} + \begin{bmatrix} k_x^2 \operatorname{vers}\phi_1 + \cos\phi_1 & -k_z\sin\phi_1 & k_z k_x \operatorname{vers}\phi_1 \\ k_z\sin\phi_1 & \cos\phi_1 & -k_x\sin\phi_1 \\ k_x k_z \operatorname{vers}\phi_1 & k_x\sin\phi_1 & k_z^2\operatorname{vers}\phi_1 + \cos\phi_1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\phi}_2 \end{bmatrix} \tag{11}$$

where $\dot{\phi}_i$ are the rate changes of the joint parameters. Finally

$$\boldsymbol{\omega}^0 = \begin{bmatrix} k_x\dot{\phi}_1 + \dot{\phi}_2(k_x k_z \operatorname{vers}\phi_1) \\ -\dot{\phi}_2 k_x \sin\phi_1 \\ k_z\dot{\phi}_1 + \dot{\phi}_2(k_z^2 \operatorname{vers}\phi_1 + \cos\phi_1) \end{bmatrix} \tag{12}$$

The linear velocity $\boldsymbol{v}$ of a point $\boldsymbol{p}$ on the positioner is given by

$$\boldsymbol{v}_p^0 = \boldsymbol{\omega}^0 \times \boldsymbol{p}^0 = \boldsymbol{\Omega}^0\boldsymbol{p}^0 \tag{13}$$

where $\boldsymbol{p}^0$ is the position vector expressed in the base coordinate system and

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{14}$$

is the angular velocity matrix [2].

However, it is more convenient to use the coordinates of a point in the moving coordinate system of the rotating plate of the positioner. Equation 13 becomes

$$\boldsymbol{v}_p^0 = \boldsymbol{\Omega}_1^0\,\mathrm{Rot}(k, -\phi_1)\,\mathrm{Rot}(z, -\phi_2)\boldsymbol{p}^2 \tag{15}$$

If the velocity of a point $\boldsymbol{p}$ is defined in the base coordinate system, then the change rates of the two joint parameters ϕ_1 and ϕ_2 can be derived from equation 13.

$$\dot{\phi}_1 = \frac{\alpha}{k_x} - \dot{\phi}_2 k_z \operatorname{vers}\phi_1 \tag{16}$$

$$\dot{\phi}_2 = \frac{\beta}{k_x\sin\phi_1} = \frac{\alpha k_z - \gamma k_x}{k_x\cos\phi_1} \tag{17}$$

where

$$\alpha = \frac{v_z p_z + v_y p_y + v_x p_x}{2p_x p_y}$$

$$\beta = \frac{v_z p_z + v_y p_y - v_x p_x}{2p_x p_y}$$

$$\gamma = \frac{v_z p_z - v_y p_y + v_x p_x}{2p_x p_y}$$

Similarly the angular acceleration of the moving plate of the positioner, in the base coordinate system, is given by [3]

$$\dot{\omega}_2 = \omega_1 + \omega_1 \times \mathrm{Rot}(k, \phi_1)\,\dot{\omega}_2' + \ddot{\phi}_2 z \tag{18}$$

where $\omega_1 = \dot{\phi}_1\boldsymbol{k}$ and $\omega_2' = \dot{\phi}_2\mathbf{z}$. The linear acceleration of a point $\boldsymbol{p}$ on the moving plate of the positioner is given by

$$\dot{\mathbf{v}}_2 = \dot{\omega} \times \mathrm{Rot}(k, \phi)\boldsymbol{p} + \omega \tag{19}$$

Link i	Twist α_i	Length a_i	Offset d_i	Joint Variable ϕ_i
1	α_1	0	0	ϕ_1
2	0	0	0	ϕ_2

Table 1. Link parameters of the universal positioner.

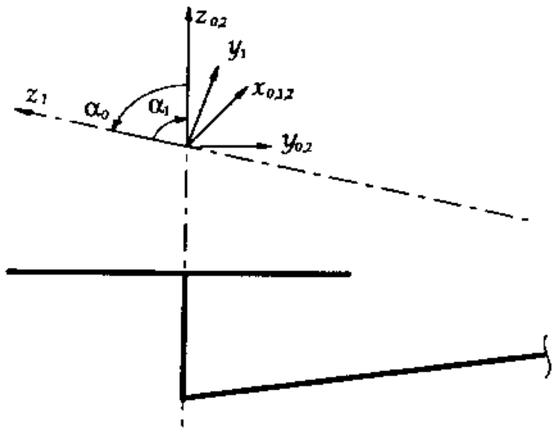


Fig. 7. Coordinate Frames for a Universal Positioner.

3.3 The Denavit-Hartenberg Representation of a Universal Positioner

The Denavit-Hartenberg frames of the universal positioner are shown in Fig. 7. The Link parameters for the universal positioner are shown in table 1. The kinematics of the positioner are solved in a similar way using the D-H representation. The main difference in the case of revolute axes is that the axis of revolution is now always z_i. See [4] for the derivation of the kinematic equations of a manipulator using generalized coordinates and the position vector of the origin of the joint frames.

4 Conclusions

In this report some aspects of the programming of welding robots have been studied. Solutions for the definition of the coordinated motion of a universal welding positioner and an industrial robot have been presented.

The programming of welding robots is a dual problem. It consists of a complex geometrical definition part and an equally complex process parameter part. There is typically a cyclic relationship between algorithm and language development. Algorithms are developed when the need is apparent. In this way robot language development and the specification and generation of motions are interconnected. A future topic that deserves much attention in the programming of welding robots is the choice of suitable process paramcters for a specified material-motion combination.

References

(1) G. Bolmsjö. *Robotsystems for snall series production — arc welding* (in swedish). PhD thesis, Dept of production and materials engineering, Lund University, Lund, Sweden, 1986.

(2) R. Bottema and B. Roth. *Theoretical Kinematics*. North-Holland, Netherlands, 1979.

(3) J.J. Craig. Anatomy of an off-line programming system. *Robotics Today*, 45–47, 1985.

(4) K.H. Low and R.N. Dubey. A comparative study of generalized coordinates for solving the inverse-kinematics problem of a 6r robot manipulator. *International Journal of Robotics Research*, 5(4):69–88, Winter 1986.