

RDS - Realistic Dynamic Simulation of Robots

E. Roos, A. Behrens, S. Anton

Univ.-Prof. Dr.-Ing. Arno Behrens is Director of the Production Engineering and Design Institute and Head of the Production Engineering Laboratory at the German Federal Armed Forces University Hamburg.

Dipl.-Ing. Eberhard Roos is a scientific assistant and project manager at the same institute.

German Federal Armed Forces University

Tel: xx49-40-6541-3309

Production Engineering and Design Institute

Fax: xx49-40-6541-2839

Holstenhofweg 85

E-Mail: Eberhard.Roos@UniBw-Hamburg.de,

D-22043 Hamburg

Dipl.-Ing. Stefan Anton is a support engineer of Deneb Germany and specialised in robot dynamics.

Deneb Deutschland GmbH

Tel: xx49-6135 9105 0

Am Kümmerling 24-26

Fax: xx49-6135 9105 99

55294 Bodenheim

E-Mail: support@deneb.de, stefan.anton@easy-rob.de

Abstract:

The industrial robots (IR) of today are increasingly being used for assembling and manufacturing tasks. However, during the teach-in programming of these machines production facilities are not available. As this programming method is very expensive, off-line Programming (OLP) techniques are being used in more and more applications. Yet in most of the simulations being carried out in industry, the dynamics of the robot system are ignored. Therefore, on the basis of a common OLP system (IGRIP[®]) a concept was developed, which allows the dynamics of an IR to be analysed while still in the phase of system development. This paper shows the opportunities offered by this new development, which combines the original robot control software (RCS module) with an extended dynamics module in the OLP system. It also explains the fundamental procedure for designing the control loop and its parameters, as well as how to integrate the software modules into the simulation system. Several simulation examples are used to investigate the concept developed and its transferability.

1. State-of-the-art of off-line programming

According to a study from January 1997, the number of IRs being used for assembling and manufacturing tasks in Germany might reach 100,000 by the year 2000. The steady improvement of robot systems has also led to progressive programming methods. Besides classic, rather unsophisticated teach-in programming, off-line programming methods have also been established in several areas of industry. This study deals with graphic off-line programming systems which allow the motion of industrial robots to be planned and simulated. These systems allow the user not only to define working positions, but also to carry out collision and reachability studies to optimise the layout of the robot cell. Furthermore, under certain conditions the cycle time can be evaluated and precise robot programs generated. With the help of progressive OLP systems and using computers with above-average graphic performance, realistic working cells may be generated up to a certain degree (virtual reality). The use of OLP systems always proves to be cost-effective if the downtime of the production facility can be reduced, or if the time to generate the CAD data, both for programming and calibrating the production cell, is shorter than the teach-in programming time.

1.1 Error sources during off-line programming and possible solutions

The steadily increasing number of installed off-line programming and simulation systems confirms the interest of robot users in this new technology. Compared with the advantages during the optimization of robot cell layout or during tests for collision and reachability checks, there are difficulties in transferring user programs that have been generated and simulated off-line to the real production cell [1], [2], [6], [7].

Translation / Interpretation of the robot programming language

The first errors often occur in the language processing of the system, the so-called interpreter or translator. One reason for this is that the extent and semantics of the language differ between OLP system and real robot control. A typical example for this is the description of fly-by records (distance or velocity approximation records). Conversion or even simple syntax errors may seriously affect the execution of a user program. A fundamental condition for the transfer of simulated user programs is a post-processor which is free of syntax errors.

Motion Interpolation

Interpolation is used for instance in the motion modes PTP, linear and circular as well as in the transition between different motion modes (fly-by). Interpolation can be described mathematically by numerous parameters (distance, velocity parameters) and again exhibits its own variations in interpolating orientation. Even large industrially established OLP systems may not be able to cope with this large number of different algorithms with their fundamental path planning functions. Thus deviations cannot be avoided between simulation and real application in terms of:

- cycle time
- path accuracy
- path velocity or
- behaviour near singularities.

Transformation / Inverse kinematics

Calculation of the inverse kinematics is performed by ideal, simplified kinematic models. Deviations in the lengths of the robot axes and assembly errors are disregarded. The effector load, thermal influences or gear elasticity affect the static compliance and make the absolute accuracy of the robot worse. Even small changes in the axis angles may seriously affect the angular configuration and may cause collisions of the robot with peripheral components.

System dynamics

A large number of parameters affect the control systems behaviour, such as mass inertia, coulombic and viscous friction as well as elasticity. The system dynamics are generally disregarded in the simulation owing to the lack of dynamic data or because of inflexible dynamic interfaces.

Environment model

Simulation errors in this part of the error chain are caused by inaccurate modelling of the manufacturing environment. The following effects have to be taken into consideration:

- modelling of the tool and workpiece
- the arrangement of positioner, workpiece, tool, IR and other cell components in relation to the world coordinate system
- the model of the process (e.g. coating).

Efficient off-line programming does not aim to eliminate all of the above-mentioned error sources. It would be very time-consuming and expensive to reduce the errors in the kinematic chain by a reduction of manufacturing tolerances. Practice-oriented off-line programming requires at least

- an efficient post-processor, which is free of syntax errors
- a realistic motion interpolation for applications in which path accuracy or the determination of cycle time plays an important role.
- suitable, user-friendly calibration algorithms to avoid costly teach-in corrections

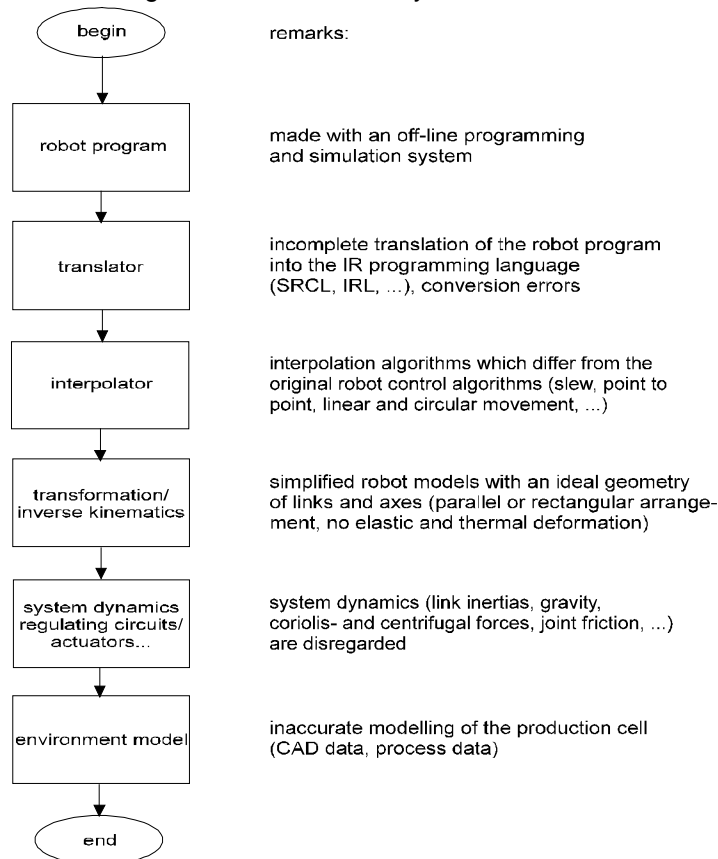


Fig. 1: Program flow chart: Conversion of a simulated user program

1.2 Measuring and calibration system with novel calibration algorithms - OPTIS

In many applications of graphic OLP systems it takes quite a long time to generate or to import CAD data of the production cell so as to analyse or to optimize the motion of the IR. To proceed consistently, the geometric data could be used directly to program the IR (Download). Instead of this, the user program is often completely reprogrammed in the production cell by using the time-consuming teach-in programming method. Valuable information about the cell geometry remains unused. Such a procedure raises questions about the economic efficiency of off-line programming methods in general. One reason for this may be found in measuring and calibration methods which are neither accurate nor user-friendly enough. These methods affect the position and orientation accuracy as well as the angular configuration of the IR, which should be almost identical in simulation and reality so as to avoid teach-in corrections.

With the help of the newly developed practice-oriented measuring and calibration system OPTIS (Off-line Programmed Tests for Industrial Robot Systems), which has been described in [1], [6], [7], it is possible to compensate almost entirely for position and orientation deviations. The following illustration shows the difference between the conventional and the novel calibration procedures schematically.

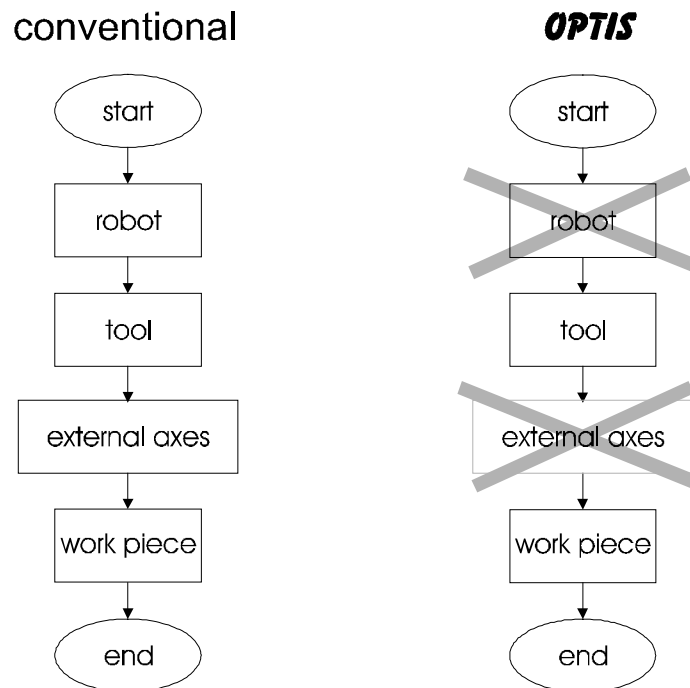


Fig. 2: Calibration procedures for simulated robot applications

Compared with conventional calibration techniques the novel calibration procedure has fewer calibration steps. Time-consuming measurements of the robot or of external axes can be completely avoided. Instead, the uncalibrated robot is used as a coordinate-measuring machine to adapt robot programs generated off-line. Reference positions are programmed by teach-in or are measured automatically by a sensor system (CCD camera, laser distance sensor), which is mounted at the mechanical interface of the IR during production cell set-up. The newly developed calibration algorithms automatically divide the workpiece up into subspheres and calculate corresponding correction frames. Thus it is possible to compensate for systematic errors resulting from the kinematic chain of the IR, the workpiece size and contour, and the relation between IR, external axes and workpiece. In 1996 the measuring and calibration system OPTIS left the test phase and was successfully introduced into industrial applications such as

- adaptation of off-line-generated user programs for high-precision laser welding
- set-up of identical robot workcells
- calibration of spot welding robots after repair or crash to improve exchangeability
- robot performance testing according to ISO 9283.

Conclusion

As the location of working positions, i.e. start, intermediate and end positions, of a test path affects the movement behaviour of an IR (for instance near singularities), the simulated and real test paths should coincide. This is a fundamental condition for further path accuracy investigations. This condition can be fulfilled with the calibration algorithms presented.

2. Causes of path deviations

Now that Section 1 has given a general idea of the error chain between simulated and real applications, the causes of path deviations are to be examined in more detail and further demands on a realistic dynamic simulation of robots are discussed. Path deviations are caused by errors in the kinematics, in the mechanics and in the dynamics of the system or by deviations between the motion planners in the OLP system and the robot control.

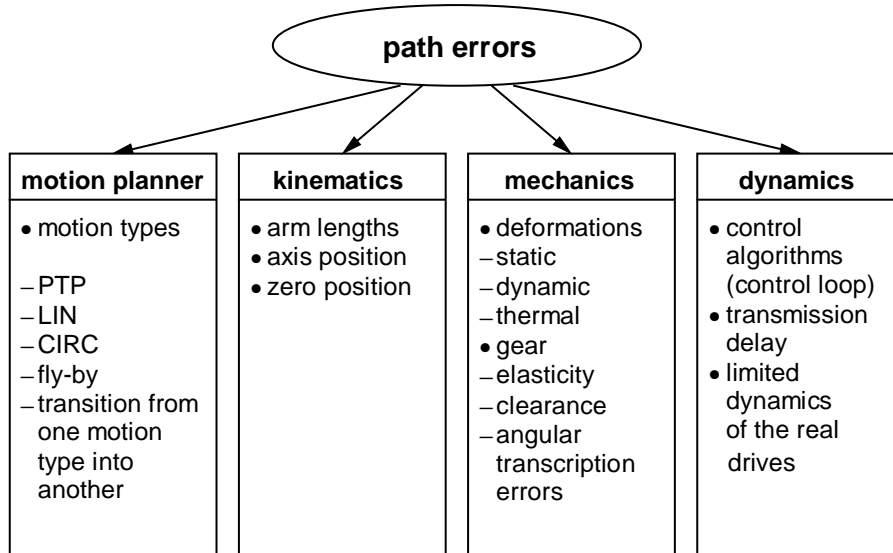


Fig. 3: Causes of path deviations

Motion planner

The motion planner causes considerable path deviations between simulation and reality. Even in the case of simple motion modes there are big differences between real and simulated paths. Figure 4 shows this effect using a sequence of PTP records.

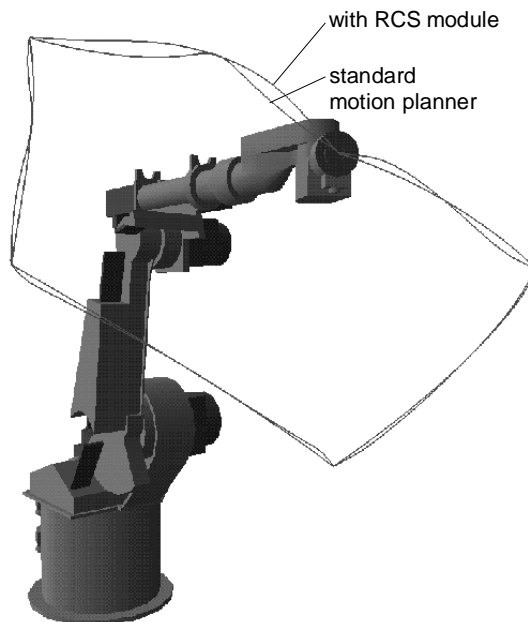


Fig. 4: Simulation deviations caused by the motion planner.
 Command path calculated with RCS module and standard motion planner.
 Kinematic robot structure

This type of error was explained in section 1 and a suitable calibration procedure was presented, which refers to the points of path support.

Mechanical structure

Static and thermal deformations can be compensated for by the calibration procedure mentioned in section 1, if comparable conditions (effector load, robot warmed up) apply during measurement of reference positions and production. The influence of gear elasticity and gear clearance on path accuracy were not taken into consideration in these investigations.

Robot dynamics

Dynamic modelling of an IR is used to determine forces and torques at each single link, which are caused by

- axis position,
- axis velocity,
- axis acceleration and
- forces and torques at the TCP.

These forces and torques correspond to the adjusting forces and torques required to move the robot arm along a given trajectory. The following differential equation system describes this connection:

$$\underline{T} = \underline{H}(\underline{\varphi}) \cdot \ddot{\underline{\varphi}} + \underline{C}(\underline{\varphi}, \dot{\underline{\varphi}}) + \underline{D}(\underline{\varphi}, \dot{\underline{\varphi}}) + \underline{G}(\underline{\varphi}) + \underline{M} \quad (1)$$

where:

\underline{T}	(n x 1) vector of the drive torques
$\underline{\varphi}$	(n x 1) vector of the axis angles
$\underline{H}(\underline{\varphi})$	(n x n) inertia tensor
$\underline{C}(\underline{\varphi}, \dot{\underline{\varphi}})$	(n x 1) vector of torques caused by Coriolis and centrifugal forces
$\underline{D}(\underline{\varphi}, \dot{\underline{\varphi}})$	(n x 1) vector of torques caused by coulombic and viscous friction
$\underline{G}(\underline{\varphi})$	(n x 1) vector of interference torque caused by gravity
\underline{M}	(n x 1) vector of interference torque caused by forces and torques at the TCP

In order to achieve realistic dynamic robot simulation, the data for the mechanical model (system of rigid bodies) and the model of the control loop have to be as exact as possible. The latter includes the controller model (control structure, control parameters) and the drive model (control, power output stage, motor, tacho generator, path measuring system) as well as possible interference.

The following factors are some that may result in model inaccuracies:

- incorrect control structure (PID control instead of cascade control)
- false assumptions in terms of the time behaviour of the drive (time-lag device of first order instead of a time behaviour of higher order)
- non-linearities of the drives (acceleration limit)
- non-linearities of the mechanical structure (gear clearance, friction)
- flexibility of the mechanical structure

Discussion:

Various investigations show that the main causes of path deviation between simulated and real applications may be found in the path planner [2]. The control structure (cascade control) and the control parameters affect the dynamics and the drive overshoot [8] and should be taken into consideration in the dynamic module. Dynamic deformation may be ignored because of the high stiffness of the robot arm [3].

3. The concept of realistic dynamic simulation

To minimise path deviations, the RCS module (RCS: Robot Control Simulation) is integrated into the simulation system and harmonized with the test robot by adapting the machine data. The RRS-interface allows parts of the original control software (the so-called RCS module, i.e. path planning and transformation algorithms) to be integrated into the simulation system. Among other functions, error and warning processing is also made possible. Experimental investigations show good agreement between simulation and real application with respect to the command path, cycle time and the angular configuration, if the RCS module is used [2].

The concept of realistic dynamic simulation includes the adaptation of the dynamic model (control, drive, mechanical structure) in the simulation system. There is a cascaded control structure in the case of the test robot (KUKA IR 364-15 with KRC 32 control).

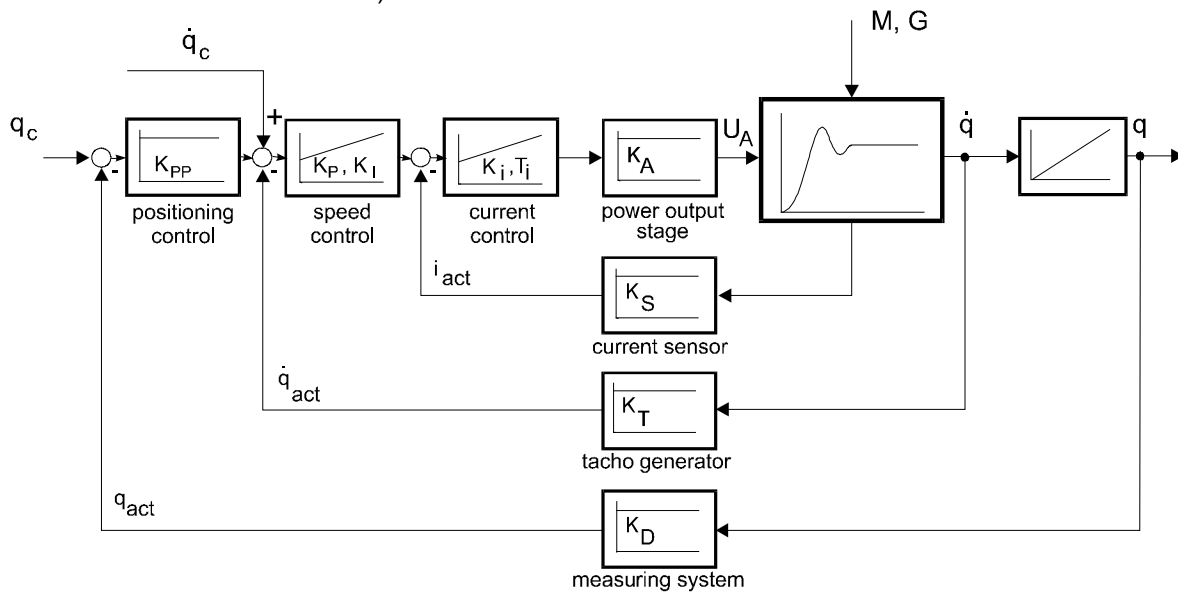


Fig. 5: Position control with cascaded speed and current control

The behaviour of the cascade control may be described as follows: cascaded control loops

- keep interference away from the positioning control variable.
- can partly compensate for non-linearities in the mechanical structure of the robot and the drives.

3.1 Simplified modelling of the control loop and robot axis

Dynamic simulation is characterized by numerous parameters to describe the models of control, drives and mechanics. A linear axis model is used to design the control loop. The simplified model is characterized by the following features:

- Cascaded speed control with analogue PI-control.
- The dynamics of the current control loop are ignored. The current control loop is supposed to be a time-lag device of first order with a small time constant.
- The robot axis is modelled without feedback torque of the gearbox on the motor.
- Simplified model of the gear box without angular transcription errors or non-linearities such as gear clearance and backlash.
- Simplified model of the mechanic structure (time element of first order) because the model of the controlled system is unknown or is too difficult to determine exactly.
- Non-linearities such as friction or limitations are ignored during control design.

In contrast to the simplified control design, the interference torque and non-linearities are taken into consideration during dynamic simulation in the OLP system.

4. Design of the axis control

The IR is modelled as a system of several stiff bodies connected by links (tree structure). An axis may be influenced by interference torques of the end effector and of other links. The axis controller, however, is designed using only isolated control systems. In order to determine the dynamic behaviour, the axis control only accesses state variables of one single axis. As the test robot (KUKA IR 364-15) has high gear ratios and comparatively short arm lengths, the coupling of the axes may be ignored in relation to the axis dynamics. With the simplifications mentioned in section 3.1 the following block diagram is obtained for the control design:

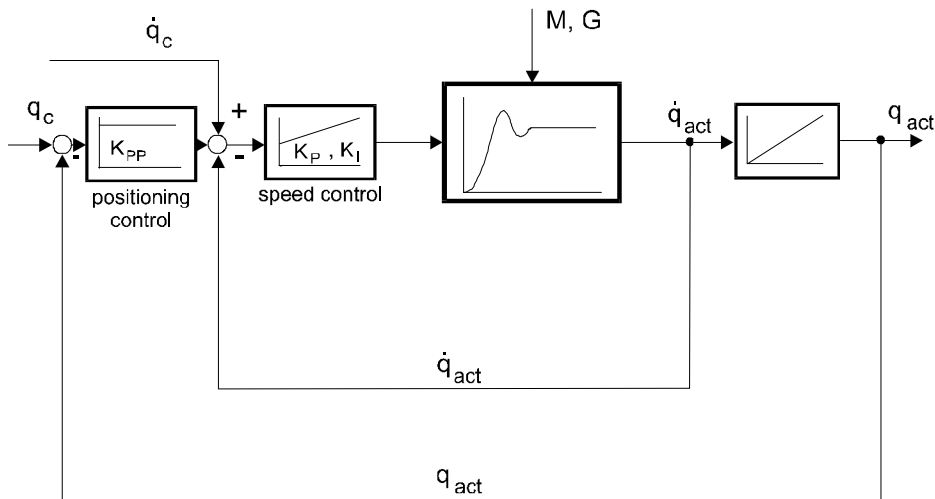


Fig. 6: Block diagram of the simplified cascade control

4.1 Design of the cascade control of one axis

The following equation describes the driving torque of the motor:

$$\tau = i \cdot J \cdot \ddot{\varphi} + i \cdot D \cdot \dot{\varphi} + \frac{G}{i} + \frac{M}{i} \quad (2)$$

- with
- τ driving torque
 - φ axis angle ($\varphi = q/i$)
 - i gear ratio
 - J mass inertia of the motor
 - D viscous friction
 - G interference torque caused by gravity
 - M interference torque caused by the effector load

Power is transmitted by high-ratio gears. This is the reason why gravity and interference torques may be ignored during the rough design calculation of the cascade control. Therefore the motor as well as the robot axis can be approximately described as PT1 elements.

Controlled system (PT1 element):

$$G_s = \frac{1}{D \cdot i \cdot T \cdot s + 1} = \frac{g}{i} \cdot \frac{1}{T \cdot s + 1} \quad (3)$$

with $g = \frac{1}{D}$ as amplification and $T = \frac{J}{D}$ as the time constant of the controlled system (4)

PI-speed control: $G_{PI}(s) = \frac{i \cdot K_P(K_I + s)}{s}$ (5)

Forward path: $L_v = G_{PI} \cdot G_s$ (6)

Return transfer function of the speed control loop:

$$T_v(s) = \frac{L_v}{1 + L_v} = \frac{i \cdot K_P \cdot (K_I + s) \cdot \frac{g}{i}}{s \cdot (T \cdot s + 1) + K_P \cdot (K_I + s) \cdot g} \quad (7)$$

$$T_v(s) = \frac{K_P \cdot g}{T} \cdot \frac{(K_I + s)}{s^2 + \frac{1 + K_P \cdot g}{T} \cdot s + \frac{K_P \cdot K_I \cdot g}{T}} = \frac{K_P \cdot g}{T} \cdot \frac{(K_I + s)}{s^2 + 2 \cdot \zeta \omega_n \cdot s + \omega_n^2} \quad (8)$$

The conjugate-complex pair of poles of $T_v(s)$ is at

$$s_{1,2} = -\zeta \cdot \omega_n \pm j \cdot \omega_n \cdot \sqrt{1-\zeta^2} \quad (9)$$

Using eqn. (8) the amplification factors K_P and K_I may be calculated by comparing the coefficients, if the damping factor (ζ) and the angular frequency (ω_n) are given:

$$2 \cdot \zeta \cdot \omega_n = \frac{1 + K_P \cdot g}{T} \quad (10)$$

$$\omega_n^2 = \frac{K_P \cdot K_I \cdot g}{T} \quad (11)$$

Chosen damping factor: $\zeta=0.5$

The real component of the conjugate-complex pair of poles of $T_V(s)$ is at $s=-\zeta \cdot \omega_n$. The distance of this pole from that of the controlled system, $s=-1/T$, shall be a multiple (n_T) of $-1/T$:

$$-n_T/T = -\zeta \cdot \omega_n \quad \text{or} \quad \omega_n = n_T / (\zeta \cdot T) \quad (12)$$

Inserting (12) into (10) gives the amplification factor of the speed control loop K_P :

$$K_P = \frac{2 \cdot n_T - 1}{g} \quad (13)$$

By inserting (12) and (13) into (11), the K_I factor of the speed control loop can be calculated:

$$K_I = \frac{(n_T)^2}{\zeta^2 \cdot T \cdot (2 \cdot n_T - 1)} \quad (14)$$

Superimposed positioning control: the speed transfer function $T_V(s)$ is a 2nd order system. Consequently, the controlled system of the positioning control is 3rd order.

Controlled system: $L_P(s) = T_V(s)/s$ (15)

P-control: $G_{PP} = K_{PP}$ (16)

Forward path: $L_P = G_{PP} \cdot T_V/s$ (17)

with $T_V(s) = \frac{\omega_n^2 (1 + s/K_I)}{s^2 + 2 \cdot \zeta \omega_n \cdot s + \omega_n^2}$ (18)

Return transfer function: $T_P = L_P / (1 + L_P)$ (19)

$$T_P(s) = \frac{K_{PP} \cdot \omega_n^2 \cdot (1 + \frac{s}{K_I})}{s^3 + 2 \cdot \zeta \omega_n \cdot s^2 + \omega_n^2 \cdot s + \frac{K_{PP} \cdot \omega_n^2 \cdot s}{K_I} + K_{PP} \cdot \omega_n^2} \quad (20)$$

$$T_P(s) = \frac{K_{PP} \cdot \omega_n^2 \cdot (1 + \frac{s}{K_I})}{s^3 + 2 \cdot \zeta \omega_n \cdot s^2 + \omega_n^2 (1 + \frac{K_{PP}}{K_I}) \cdot s + K_{PP} \cdot \omega_n^2} \quad (21)$$

with $\alpha_0 = K_{PP} \cdot \omega_n^2$
 $\alpha_1 = \omega_n^2 (1 + K_{PP}/K_I)$
 $\alpha_2 = 2 \cdot \zeta \cdot \omega_n$ (22)

Calculation of the amplification factor of the positioning control loop: the denominator is a 3rd degree polynomial. Thus a conjugate-complex pair of poles and one real pole or three real poles will result.

$$(s^2 + 2 \cdot \zeta_p \cdot \omega_{pn} \cdot s + \omega_{pn}^2) \cdot (s + s_p) = s^3 + (2 \cdot \zeta_p \cdot \omega_{pn} + s_p) \cdot s^2 + (2 \cdot \zeta_p \cdot \omega_{pn} \cdot s_p + \omega_{pn}^2) \cdot s + \omega_{pn}^2 \cdot s_p \quad (23)$$

Comparison of the coefficients of (23) and (22):

$$1. \quad 2 \cdot \zeta_p \cdot \omega_{pn} + s_p = \alpha_2 = 2 \cdot \zeta \cdot \omega_n \quad (24)$$

$$2. \quad 2 \cdot \zeta_p \cdot \omega_{pn} \cdot s_p + \omega_{pn}^2 = \alpha_1 = \omega_n^2 \cdot \left(1 + \frac{K_{PP}}{K_I} \right) \quad (25)$$

$$3. \quad \omega_{pn}^2 \cdot s_p = \alpha_0 = K_{PP} \cdot \omega_n^2 \quad (26)$$

The coefficients ζ , ω_n and K_I are known from the cascaded speed control loop. K_P is included in ζ and ω_n .

Input of the real pole of the positioning transfer function:

With respect to the root locus curve the real pole of the positioning transfer function is between 0 and $-K_I$.

$$s_p \in]0, \dots, K_I[$$

Calculation of K_{PP} , ζ_p and ω_{pn} :

$$\text{from (26):} \quad \left(\frac{\omega_{pn}}{\omega_n} \right)^2 = \left(\frac{K_{PP}}{s_p} \right) \quad (27)$$

$$\text{from (24):} \quad 2 \cdot \zeta_p \cdot \omega_{pn} = 2 \cdot \zeta \cdot \omega_n - s_p \quad (28)$$

$$\text{from (25):} \quad \left(\frac{\omega_{pn}}{\omega_n} \right)^2 + \frac{2 \cdot \zeta_p \cdot \omega_{pn} \cdot s_p}{\omega_n^2} = 1 + \frac{K_{PP}}{K_I} \quad (29)$$

$$\text{with (27) and (28)} \quad \frac{K_{PP}}{s_p} + \frac{(2 \cdot \zeta \cdot \omega_n - s_p) \cdot s_p}{\omega_n^2} = 1 + \frac{K_{PP}}{K_I} \quad (30)$$

$$\text{transformed} \quad K_{PP} \cdot \left(\frac{1}{s_p} - \frac{1}{K_I} \right) = 1 - 2 \cdot \zeta \cdot \frac{s_p}{\omega_n} + \frac{s_p^2}{\omega_n^2} \quad (31)$$

$$\text{Thus } K_{PP} \text{ results:} \quad K_{PP} = K_I \cdot s_p \cdot \frac{1 - 2 \cdot \zeta \cdot \frac{s_p}{\omega_n} + \frac{s_p^2}{\omega_n^2}}{K_I - s_p} \quad (32)$$

ζ and ω_n are already known from the speed control loop, s_p must be entered into the equation.

Calculation of the angular frequency ω_{pn} and of the damping factor ζ_p of the positioning control loop:

$$\text{Using equation (26):} \quad K_{PP} = \left(\frac{\omega_{pn}}{\omega_n} \right)^2 \cdot s_p \quad (33)$$

$$\text{Inserting (33) into (32) gives } \omega_{pn}: \quad \omega_{pn} = \omega_n \cdot \sqrt{K_I \cdot \frac{1 - 2 \cdot \zeta \cdot \frac{s_p}{\omega_n} + \left(\frac{s_p}{\omega_n} \right)^2}{K_I - s_p}} \quad (34)$$

From (24):

$$\zeta_p = \frac{2 \cdot \zeta \cdot \omega_n - s_p}{2 \cdot \omega_{pn}} \quad (35)$$

5. Dynamic simulation in IGRIP®

A dynamic module in IGRIP® (Interactive Graphics Robot Instruction Program) is used to simulate the dynamic movement behaviour. This module directly generates the non-linear equation system from the kinematic chain of the robot (tree structure). A graphic user interface may be used to enter parameters, such as mass, centre of gravity, mass inertia, friction, additional forces and torques.

A moving robot may be affected by various forces. There are for instance forces due to gravity, mass inertia, friction as well as coupling forces between the axes, which are calculated by the Newton-Euler method. The drive torque is superimposed by interference torques at the robot axis. These torques affect the axis variables of position, velocity and acceleration. The differential equation system of second order (see Eqn. 1) is solved by using integration methods (Runge-Kutta, Adams-Bashforth, etc.), which are integrated into the simulation system. The software modules to describe the time behaviour of the axes as well as the algorithms of the cascade control can be implemented into the simulation system with the help of a C-interface (Shared Library).

6. Realistic dynamic simulation by combination of RCS module and dynamics

Realistic dynamic simulation is based on the RCS module (RCS: Robot Control Simulation) which has already been mentioned in section 3. It consists of five basic services, which are called up in a defined order. According to [9] a control is initialized and identified with the service "Initialize". The angular position of the robot is defined by the command "Set_Initial_Position". Thus the start position is defined for the interpolator. After the motion target has been transmitted to the control by using the service "Set_Next_Target", intermediate positions calculated by the interpolator may be requested through the service "Get_Next_Step". Here the dynamic module of the IR and the RCS module can communicate with each other. In addition to its function of planning paths, the original control software also transmits information about the angular position of the robot and about error messages and warnings. After the command position has been reached, the service "Get_Next_Step" requests new motion targets. Using the service "Terminate" interrupts the connection between control and simulation system.

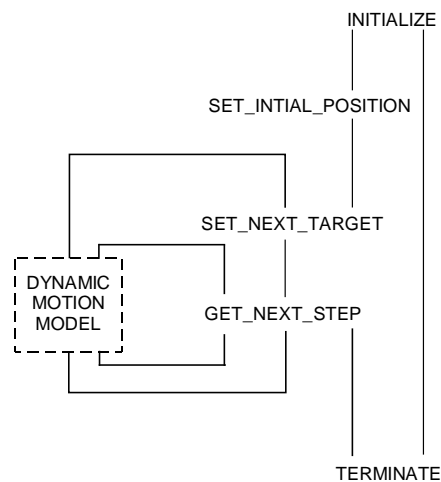


Fig. 7: Interaction of RCS module and dynamic module in IGRIP® (see also [9])

Experimental verification

On the basis of ISO 9283 [10] a test path was programmed to verify the system behaviour calculated by the software modules developed (cascade control, robot model). The test path consists of linear and circular sections of different lengths and may be used to determine robot characteristics such as

- path accuracy (AT)
- cornering overshoot (CO)

- cornering round of error (CR) and
- stabilization path length (SPL).

The investigations were carried out using the test parameters of path velocity, effector load and motion type mentioned in the standard. Figure 11 shows the experimental set-up for testing path accuracy. The actual path is visualized by integrating a plotter pen into the tool.

Parameters (amplification factors, time constants) calculated by the control design module (see section 4.1) as well as from the machine data file of the robot were used in the experiments. The formulae to calculate the amplification factors and explanations to the positioning control loop were supplied by the robot manufacturer. The amplification factors, which were calculated during the control design, allowed acceptable results with respect to the dynamic systems behaviour. A graphic user interface integrated into the simulation system allows the behaviour of the control systems to be improved interactively by changing the amplification factors, for instance. During the simulation the user is able to supervise variables such as drive torque or position and speed deviations.

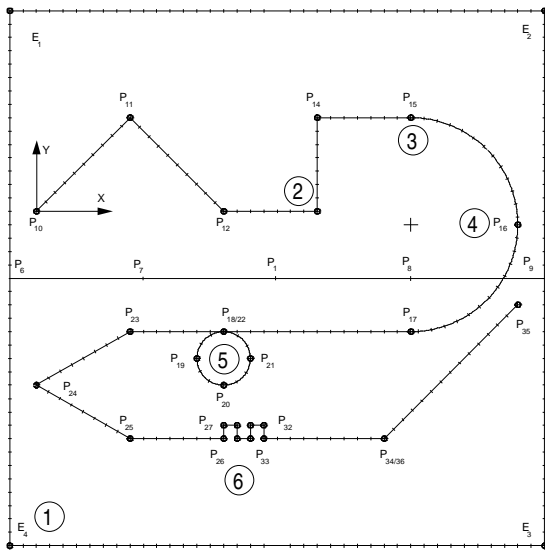


Fig. 8: Complex test path [10]

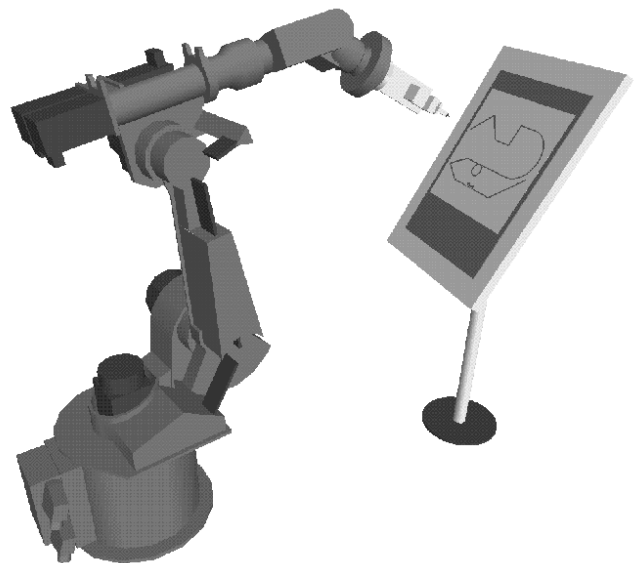


Fig. 9: Experimental set-up to test path accuracy

8. Results

8.1 Simulation with cascade control and RCS module

Agreement between simulation and reality for respect to the test robot (KUKA IR 364-15) with KRC control (RC 30/52):

Little influence of velocity and effector load

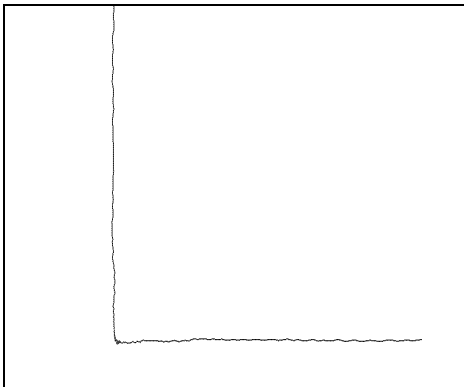
- on path accuracy in the case of linear paths.
- on overshoot at the end of linear paths.
- on cornering overshoot (CO).
- on path accuracy in the case of approximated linear path sections (fly-by).
- If path velocity is increased, there will be greater path deviations in the case of circular paths. The origin of this effect may be found in the interpolator, in the control loop (transmission delay between actual and calculated position and speed) as well as in the mechanics (clearance of axis no. 1). Thus an elliptical path results from a circular command path.
- If the effector load is increased, there will be a bigger overshoot at the end of circular paths.
- Good agreement between simulated and real cycle time (see [2]).

Deviations between simulation and reality (cascade control and RCS module):

- The maximum programmable path velocity is lower in the simulation than in the real application. The maximum velocity value in the simulation corresponds with the actual velocity on the test path, that can be reached by the robot. If a too high path velocity is programmed, the RCS module will transmit an error message and interrupt the simulation. This effect does not appear in the real control. It may for instance be noticed if the robot has to move around sharp corners.
- Velocity fly-by mode may not be carried out by the RCS module in the case of very short linear movements (see fig.10, section 6). In contrast to the real robot control the RCS module signals this with an error message.

Simulated path behaviour

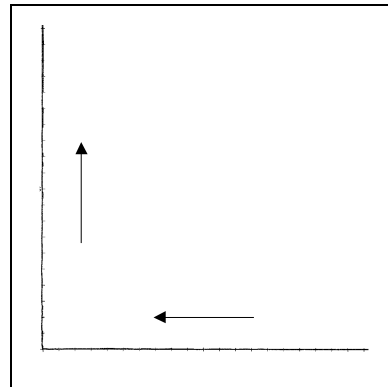
Real path behaviour



Cornering behaviour

transition
line-line

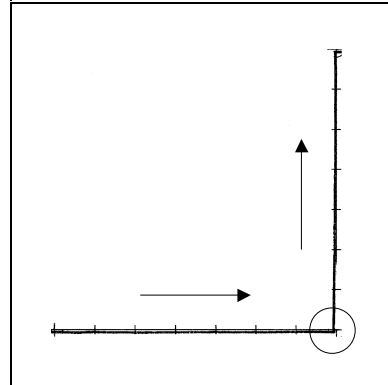
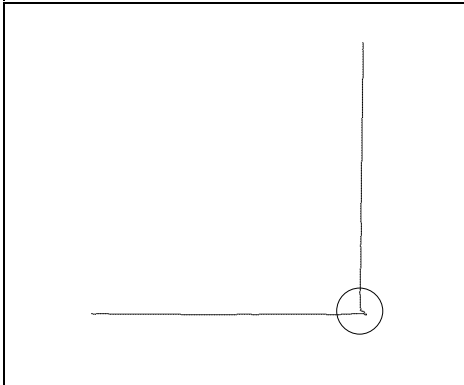
fig. 8, section 1



Cornering behaviour

transition
line-line

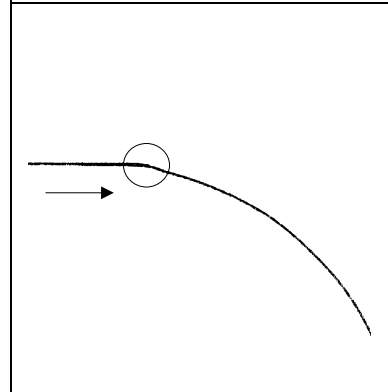
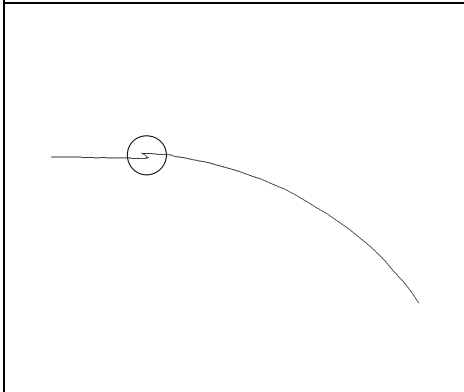
fig. 8, section 2



path behaviour

transition
line-circle

fig. 8, section 3



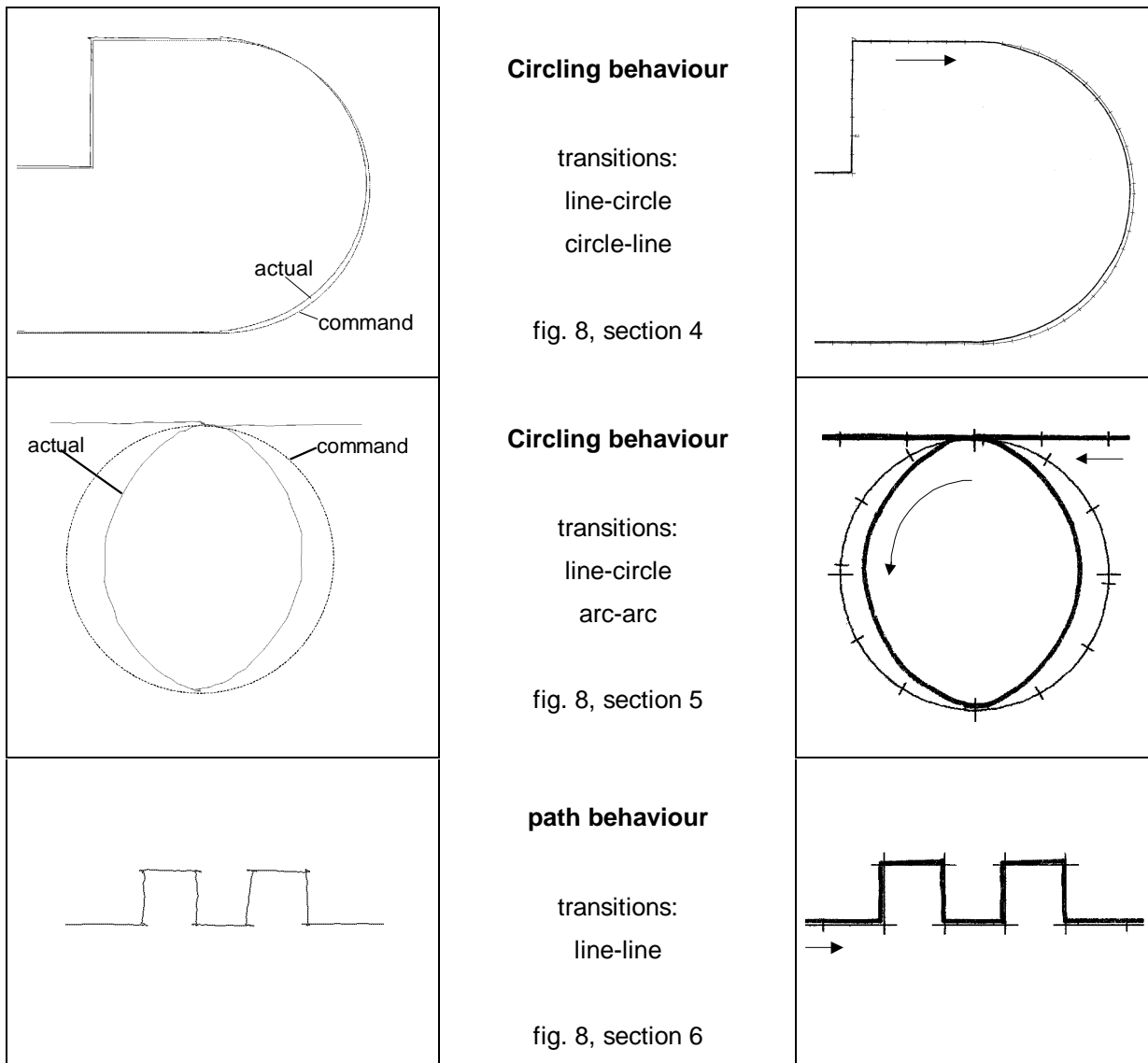


Fig. 10: Simulated and real path behaviour (RCS module and cascade control)
 Path velocity in m/s respectively deg/s: \$VEL={CP 0.3, ORI1 200, ORI2 200}
 Payload: 13 kg

The test series showed good agreement between simulated and real dynamic movement behaviour in the combination of cascade control and RCS module.

8.2 Simulation with cascade control and standard motion planner

Further experiments were carried out to compare the dynamic movement behaviour without the original path planning algorithms of the KRC-control. The following results were obtained:

Agreement between simulation and reality:

- Little effect of velocity and effector load on path accuracy in the case of linear paths.
- Increased overshoot at the end of circular paths as a function of the effector load.

Differences between simulation and reality:

- Large overshoot at the beginning and end of simulated linear path sections as a function of the velocity and effector load. The control loop may even become unstable.
- Large cornering round-off error depending on the velocity and effector load.
- Circular path sections do not show an elliptical contour in the simulation.
- Large deviations between the simulated and real linear approximations of path sections (velocity fly-by) (see also [2]).
- Greater differences between simulated and real cycle time compared with the results achieved by the RCS module (see also [2]).

Altogether there are much greater differences between simulation and reality in the combination of standard motion planner of IGRIP® and the cascade control. The reason for this may be found in the path planning of IGRIP®, which uses ramp speed-time curves. The real control uses parabolic speed-time curves with smooth transitions between the curve sections. Thus an infinite jerk affecting the control systems behaviour may be avoided.

8.3 Discussion

Dynamic simulation requires information about a large number of parameters that can only be determined with good co-operation between the manufacturers of robot, control and OLP system. If time is to be saved, it is difficult to avoid simplification of the dynamic robot model or the control loops. A combination of RCS module and cascade control provides a realistic description of the interpolator and the control structure of the robot control. In contrast to the simulation with the standard motion planner of the OLP system, the results achieved by the original path interpolator (RCS module) and the cascade control can be used for analysing the IR movement behaviour. According to information from the robot manufacturer, the dynamic systems behaviour is optimized by potentiometers or by editing the machine data. Depending on the application planned, the robot can be adjusted to work according to a contour-optimizing or time-optimizing procedure. The new software tools already enabled control design and improvement of the dynamics system behaviour for the test robot to be performed interactively in the simulation system.

9. Summary and future developments

RDS (Realistic Dynamic Simulation of Robots) stands for a new concept aimed at realistically simulating the dynamic movement behaviour of industrial robots. It is a combination of cascade control, dynamic robot model as well as original path planning algorithms (RCS module), integrated into the simulation system IGRIP®. It allows the dynamics of a robot system to be simulated and analysed in real time. RDS can also be used to control dynamic parameters in the simulation system, such as the drive torque. Critical path sections or overcharging of the mechanical structure and the drives may be recognized in the planning phase of a robot application. If the controlled system is known approximately, the control design and improvement of control systems behaviour can then be carried out in the simulation by modifying the amplification factors. A graphic user interface in the simulation system may be used for this. Path deviations caused by the system dynamics can be recognized with the help of RDS in the simulation and may, for instance, be decreased by correction of the trajectory. Thus time-consuming optimizations in the real cell, for instance near singularities, may be avoided. It is conceivable that the system dynamics could be tuned during the design phase of an IR system, so as to optimize either the motion for contours or paths or the cycle time. To achieve this, additional algorithms are to be integrated in the software module to optimize the control. Experimental investigations with the control KRC 32 and the robot KUKA IR 364-15 showed good agreement between simulated and real movement behaviour. Additional investigations, using industrial robots with much higher payloads, are now needed to see whether the concept is transferable and generally valid. The experiments showed that realistic dynamic simulation will only be possible with cooperation of the robot manufacturers. Therefore it would be advantageous if the control algorithms were supplied by the robot manufacturer as software modules and could be integrated into the simulation system, using a defined interface. The RRS interface was the first step. RDS is on its way to becoming the second.

Acknowledgements:

The investigations in this paper were supported by KUKA Roboter GmbH. The authors would like to thank Mr. W. Angerer and Mr. D. Engelhardt for their patience in giving the authors information about the mechanics and control of the test robot. Special thanks go to Prof. Dr.-Ing. Dr.-Ing. E.h. R. Lunderstädt of the Automation

Technology Institute of the German Federal Armed Forces University, Hamburg, for his advice in the field of control engineering. Last but not least the authors would like to thank Dr. Scott E. Walter from Deneb Germany, who gave valuable hints on how to combine the RCS module with the dynamic module in IGRIP®.

References:

- [1] Behrens, A.; Roos, E.: A Method to Adapt Off-Line Programmed Manufacturing Tasks to the Real Environment Using High Resolution Sensor Devices. Proceedings of the 28th Intern. Symp. on Automotive Technology and Automation, Stuttgart 1995, pp. 121/129.
- [2] Bernhardt, R.: Deviation of simulation and reality in robotics: Causes and counter-measures. Proceedings of the 28th Intern. Symp. on Automotive Technology and Automation, Stuttgart 1995, pp. 139/144.
- [3] Dalacker, M.; Horn, A.: Moderne Regelungsverfahren für Industrieroboter. AT-Automatisierungstechnik Vol. 41 (1993) No. 10, pp. 363/371.
- [4] Harkort, R. P.: Ein Beitrag zur Steigerung der Verfah- und Positioniergenauigkeit von Industrierobotern im Rahmen eines Offline-Programmiersystems. Düsseldorf: VDI-Verlag 1988.
- [5] Pritschow, G.; Gronbach, H.: Simulation der Dynamik von Industrierobotern. Roboteranwendung für die flexible Fertigung. München, Wien: Hanser-Verlag 1994, pp. 48/62.
- [6] Roos, E.; Behrens, A.: Practice-oriented adaptation of simulated manufacturing tasks to the real workcell environment. International Journal of CAD CAM and Computer Graphics (1-2/1996), Vol. 11, pp. 185/198.
- [7] Roos, E.; Behrens, A.: A Method to Adapt Off-Line Programmed Manufacturing Tasks to the Real Environment. Proceedings of the 27th Intern. Symp. on Industrial Robots , Milan 1996, pp. 841/846.
- [8] Spur, G.: Die Genauigkeit von Maschinen: eine Konstruktionslehre. München, Wien: Hanser-Verlag 1996.
- [9] RRS-Interface Specification, Berlin: Fraunhofer-Institute for Production Systems and Design Technology (IPK), 1994.
- [10] ISO WD 11031. Manipulating industrial robots - Application oriented test - Path oriented applications. Revision 1, 1993.