A Programmable Inference Controller Meeting Highest Safety Requirements

Peter Vogrin, Wolfgang A. Halang

Abstract:

A programmable controller suited for automation applications of highest safety criticality is presented. Its features are input conditioning by low-resolution analogue-to-digital converters, inference by look-up in rule set tables, strictly periodic and jitter-free operation, high speed and simplicity of design. A fail-safe supervisor immediately initiates an emergency shut-down in case of a malfunction. The software in form of rule base tables can easily be verified by inspection. This device is used to implement control algorithms working with set-point pre-processors, which calculate internal set-point graphs of controlled variables in such a way that very high controller gains are attainable and, thus, stability is increased. The performance of these „SPP" controllers is closer to the „best physically possible", and much more predictable than the one of conventional control structures. The otherwise conficting design objectives stability, safety, high speed, small energy consumption, or steady and harmonic temporal controller output values can nearly all be achieved.

Keywords: *safety methoded control, safety-licensing, programmable electronic system, set-pair preprocessor, rule-based control, fuzzy control.*

1. Introduction

Safety-related devices and control systems are employed in many application areas of vital importance. With regard to the following two main reasons, the performance of state-of-the-art controllers employed in safety-critical systems is often very unsatisfying:

1. Owing to the overwhelming complexity of hardware and especially software, the thus restricted possibilities for safety-licensing, and the corresponding guidelines of the licensing authorities, the design principles of control equipment have to be quite elementary. In particular, according to the *List of Type Approved Programmable Electronic Systems [2], it is prohibited to use PID or other more complex control algorithms in safety-related applications*.
2. Another detriment to present controllers is their inherent lack of two aspects of speed:
   (a) Owing to their mathematical models, control systems react slowly after modifications of their set-points, and after unforeseen events, such as disturbances, defects and alterations of parameters of the technical processes being controlled.
   (b) Loop execution time.

In this paper, a novel programmable electronic system is introduced, which addresses safety issues by perfection. It is not suitable for any computing or industrial automation tasks, but for a large class of control tasks as typically found in applications having to meet the requirements of Safety Integrity Level 4 as defined in IEC 61508 [4]. The presented design fascinates by its simplicity as it does not involve any kind of sequential programs and arithmetic calculations. Nevertheless, the controller's behaviour is easily programmable by just inserting other memory modules containing different tables.

The programmable controller conditions its input domains by relatively coarse rasterising carried out in hardware, viz., by linear or non-linear low-resolution analogue-to-digital converters. This leads to an inference scheme which does not require any numerical or Boolean computations, but just look-ups in tables containing rule sets, and the name "inference" or "IF" controller.

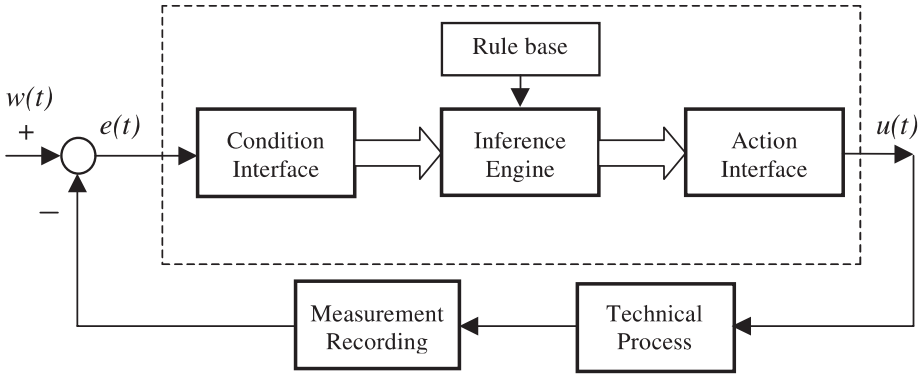

Fig. 1. Functional diagram of the inference controller.

Thus, the controller neither needs a general-purpose processor nor software in form of sequential computer programs, both of which would make the controller's safety-licensing practically impossible considering the present state-of-the-art. Instead, software only takes the form of rules in tables, lending itself to rigorous verification by inspection [1]. By design, the IF controller consists of a rather small number of relatively simple and long established hardware modules whose correct operation is permanently supervised by an inherently fail-safe circuitry. Upon any irregularity, this supervisor immediately causes an emergency shut-down of the controller and the technical process. Thus, safety-licensing of the hardware can follow well understood and already proven procedures. Generally, IF controllers have two major benefits as compared to controllers with conventional structures. Even though IF controllers are relatively simple to be safety-licensable, they can, in principle, approximate any control algorithm with sufficient precision. The second great advantage of IF controllers is their extremely short loop execution time.

Control systems often exhibit a big gap between real and desired performance, which has its reason in their mathematical models. Employing set-point pre-processors (SPP) as introduced in the second part of this paper promises a clear improvement of control performance in many application cases. A set-point pre-processor computes the internal set-point graphs of a controller in such a way, that its real behaviour is as close as possible to its desired behaviour. The effort to design such a controller is relatively small if the mathematical model of the technical process does not contain any considerable dead time or lag elements, as is the case for robots. The attainable speed and stability of SPP robot controllers are several times higher than for conventional controllers, even if the mathematical model of the technical process to be controlled could only approximately be taken into consideration in designing their set-point pre-processors. In a case study, an SPP robot control algorithm is designed as a fuzzy controller with rectangular input membership functions and implemented on an IF controller.

2. An Electronic System Programmed by Rule Sets

As shown in Fig. 1, the inference (IF) controller is designed with a condition interface producing rasterised values of several input variables. These are then subjected to an inference engine co-operating with a rule base. The outputs from the inference engine are directly provided to an action interface, which finally performs process actuation.

As inputs several analogue signals are provided to the controller in form of control errors, i.e., differences between actual measured values and desired values. For reasons of algorithmic simplification in the controller, and to use proven hardware as widely as possible, these differences are determined in analogue form with operational amplifiers. The transfer behaviour of the IF controller can be described by the static relations between its input and output values. Thus, if the raster intervals of the inputs are small enough, it is possible to approximate any static control algorithm with this type of controller. Moreover, dynamic controllers can be composed by adding integrators and differentiators. The main component of the IF controller is the inference engine. It operates under a strictly periodic regime. In contrast to industrial programmable logic controllers each loop execution tkes exactly the same amount of time, because the same operations are carried out in every iteration. Thus, the controller's real-time behaviour is fully deterministic and easily predictable. Every loop execution comprises three steps:

1. input data generation by analogue-to-digital conversion in the condition interface,
2. inference by determining appropriate control rules, and
3. control actuation via digital-to-analogue converters in the action interface.

These steps as well as the overall operation cycle are strictly synchronised with a system clock. The control errors are fed into the condition interface. The domains of the input variables are subdivided into intervals and, thus, a (coarse) discretisation of the input data is obtained. As the input values are given in analogue form, the most simple and straightforward way to directly obtain the corresponding digital coding is to employ analogue-to-digital converters. Thus, if the intervals are equally long, the condition interface reduces to a set of standard A/D converters, whose number equals the number of input variables. By discrete implementation of the A/D converters or by non-linear pre-amplification in the input stage, non-equidistant domain partitions can be realised, thus providing different precision in different ranges. Typically, higher precision is selected around reference points.

The inference engine's rule base consists of one table for each output variable to be determined. Logically, these tables might be interpreted as cause effect tables, i.e., each rule has, in principle, the form:

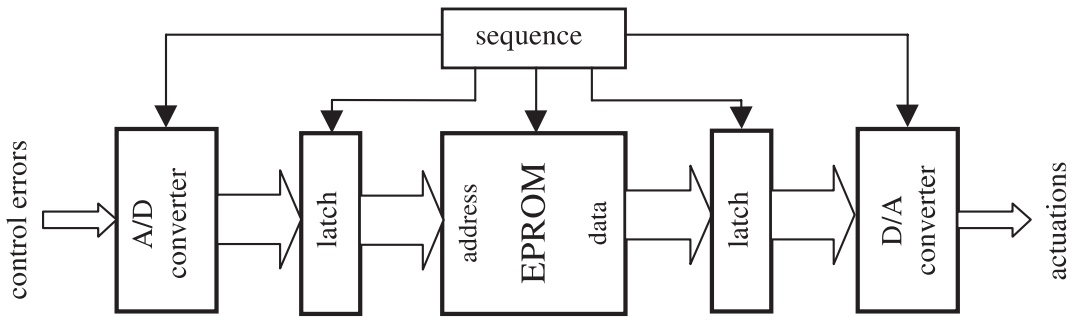

Fig. 2. Architecture of the inference controller.

if $\{cause_i\}$ **then** $\{effect_j\}$

The tables may also have the form as shown in Table 1: A value of an output variable is assigned to any conjunctive combination of the values corresponding input variables can assume. Such tables are most easily implemented as random access memories which, for safety reasons, should be readable only, i.e., as ROMs, PROMs, or EPROMs. Actually, Boolean conjunction of input variables is not performed. Instead, the binary codes of the input values are concatenated to form addresses of table entries, i.e., memory locations, containing digital equivalents of the desired actuations. Fig. 2 shows the IF controller's architecture. An input latch is needed to prevent jitter on address lines. The state of inputs is sampled and latched to provide the address bits of an EPROM. The thus read out data represent the output value associated with the given inputs. Latches hold the output values until new ones are provided. A sequencer, implemented with a PAL and consisting of a clock generator and dividers (counters), controls the latching of inputs, the generation of outputs, and their latching in a sequential way. The EPROM read-outs are finally provided to the action interface. This design avoids any further transformations by directly storing the digital equivalents of the desired actuations in the rule base tables (EPROM). Hence, the action interface reduces to a standard digital-to-analogue converter for each output signal to be generated by the controller.

In order to make the IF controller apt for utilisation in safety-critical or vital environments, it is endowed by a device supervising correct operation. In case of a malfunction this supervisor generates a signal which can be used to initiate an emergency shut-down of both the controller and the technical process. Owing to these requirements, the supervisor must be implemented in a fail-safe logic. To this end, a dynamisation principle is applied. As shown in Fig. 3, a detailed functional diagram of the IF controller, each functional unit provides a ready signal indicating successful operation. These signals are logically conjugated, by fail-safe And-gates, with the clock signals initiating the particular operations to form enable signals provided to the subsequent operation each. The last digital-to-analogue conversion performed in the action interface enables the first analogue-to-digital conversion in the condition interface to realise cyclic control operation. All enable signals are also input to a fail-safe Or-gate whose output drives an RC element. The temporal behaviour of the voltage at the capacitor C is depicted in Fig. 4. Only when the enable signals continue to permanently re-load the capacitor via the Or-gate and the resistor R, the voltage at C remains higher than a certain threshold. If the signals cease for any reason whatsoever, the capacitor discharges causing a relay to switch to emergency-off.

3. Software Safety-licensing

The contents of the rule base tables is the only "software" contained in IF controllers. All other functions are implemented in hardware. Here software does not mean executable sequential programs fetched from writable memory as in the classical Von Neumann computer architecture. Instead, it is better described as a parameterisation with which a general-purpose device is configured to perform a specific function. Since coded rule bases should always reside in some kind of read-only memories,

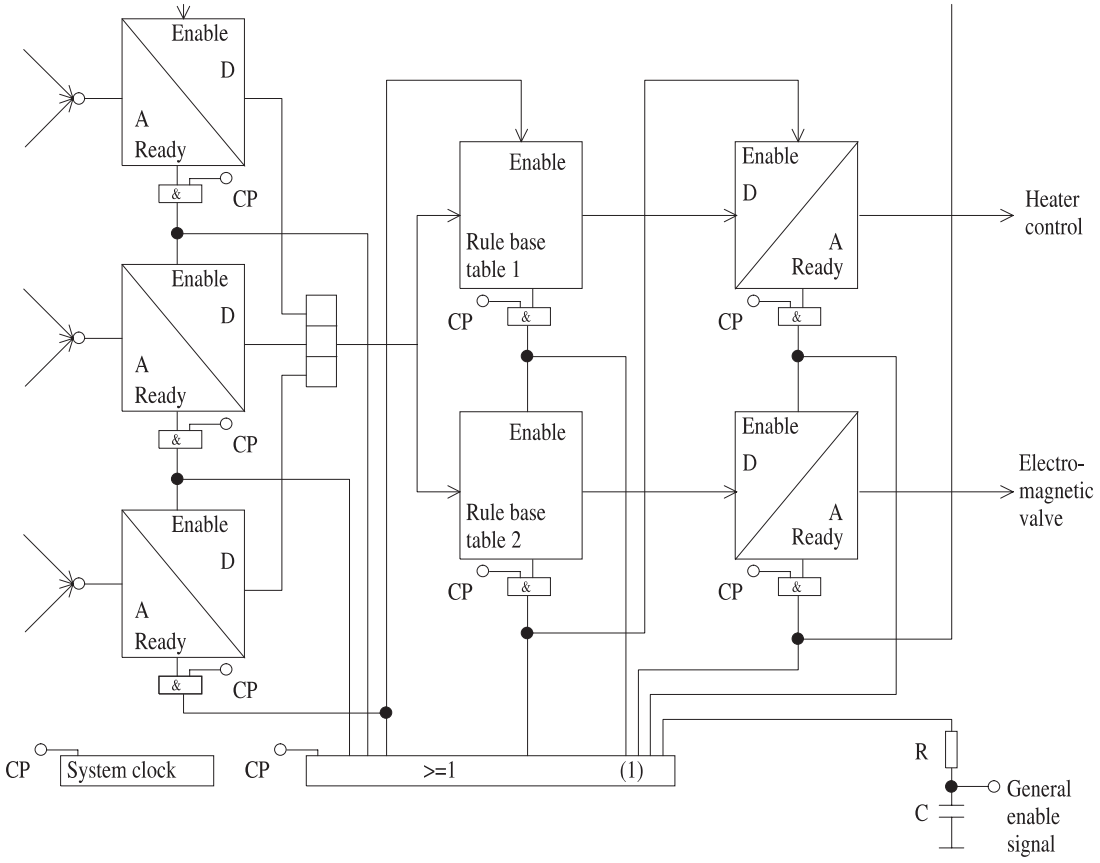

Fig. 3. Fail-safe supervision of the inference controller's operation.

the software takes on the form of firmware.

Rigorous software verification is, in general, still an unsolved problem due to the complexity of software. Moreover, object code, i.e., the only version of a program actually visible to and executed by a machine, must be considered for purposes of safety-licensing, since the transformation of a program's representation from source to object code by a compiler or assembler may introduce errors into the object code. The architecture of the IF controller greatly facilitates the rigorous verification of the software contained under the constraint that object code needs to be examined. Owing to this software's very limited complexity, it is feasible to employ the safety-licensing method of back translation. This method was developed by a licensing authority, viz., TÜV Rheinland, and consists of reading loaded object code out of a machine and having it inspected by human licensors [5]. If they work without mutual interaction, the process fulfills the requirements of diversity. Inspection is essentially informal, easily understandable, and immediately applicable with-out training. Its ease of understanding and use inherently fosters error-free application of the method.

Since rule base tables are machine executable on one hand, but also constitute formal problem specifications on the other, there is, *by design*, no semantic gap, except coding, in the IF controller's architecture between the levels relating to humans and to the machine. *Inspecting object code thus means to verify an implementation and to validate a problem specification at the same time*. The effort involved to verify rule base tables and their correct implementation is by orders of magnitude less than for licensing sequential software and is, therefore, also economically feasible.

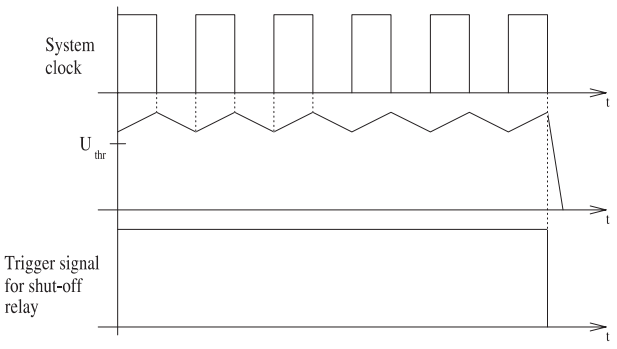

Fig. 4. Dynamisation principle for fail-safe supervision.

4. Case Study and Experimental Results

The following simple example shows the principle of constructing a rule-based controller. We consider control of a pointer's angle of rotation without friction or any other damping to provide for the most difficult case to control. As the pointer moves on a vertical plane, the gravity needs to be accounted for, and usually a torque is needed to hold the pointer at the desired angle. Let the pointer's mass be 1 kg and its length 1 m. Therefore, the rotation of the pointer is mathematically described by the following differential equation for the angle ϕ as a function of time:

$$\frac{d^2\phi}{dt^2} = 3 \cdot (M_C + M_D + 4.9 \cdot \sin\phi) \text{ and } M_C = 5 \cdot I \quad (1)$$

with M_C being the torque controlling the pointer, M_D the torque disturbing it, and I the current leading to the torque M_C. If the control deviation is larger than $2\ rad$ and M_D is not too big, the pointer moves to the desired rotation angle with an angular velocity between 10 and 15 $\frac{rad}{s}$, and the controller works as speed controller. After the acceleration period, the pointer moves to the desired angle nearly without actuations. Therefore, if the control error is $2\ rad$, the angular velocity is between 10 and 15 $\frac{rad}{s}$. It is easy to construct and to improve a rule-based controller for this narrow speed range. If the control deviation is smaller than $2\,rad$, the controller works as an angle of rotation controller. It is also possible to control the speed of the pointer dependent on the control deviation. This approach could reduce the regulating time. Higher speed, however, certainly leads to higher mechanical forces. These considerations lead to the structure of a rule-based controller as shown in Fig. 5. For the domains of the input variables non-equidistant partitionings with higher precision around zero as shown in Figs. 6 and 7 are selected. The number of words required in the EPROM corresponds to the product of the numbers of input intervals. In this example 63 words are needed. Table 1 shows a part of the rule base table, in which I_D is the digital equivalent of the current leading to the torque M_C. These values are stored in the EPROM.

Experiments and measurements were carried out to compare the performance of such rule-based controllers and of best possible classical PID controllers. They revealed that, in addition to the ones already mentioned, rule-based controllers have many advantages:

- The overshoot after a set-point jump is much smaller (less than 0.1 rad) independent of the set-point.
- The output error is smaller than 0.1 rad in a much shorter time.
- If the disturbing torque M_D is not too large, it is possible to freely determine the maximum speed of the pointer.

This case study also shows that it is possible to build a rule-based controller with an EPROM of a very small capacity, only. Nevertheless, its performance turns out to be much better than the performance of a conventional approach. Moreover, it is easily feasible to improve the performance of the rule-based controller. The range of permanent control errors can, for instance, be reduced by

- providing more input intervals, especially for the control deviation x_1, close to zero, or by
- providing an additional controller input for the integral of the rotation angle's control deviation.

5. Control with Set-point Pre-processors

The purpose of endowing a controller with a set-point pre-processor (SPP) is to calculate the internal set-point graph in such a way that the maximum difference between the set-point graph and the real position is much smaller than the maximum control error of a conventional controller. Then, the temporal graph of this difference is steady and harmonic, higher controller gains are possible and, therefore, feedback control systems become more stable. Another great advantage of such SPP controllers is

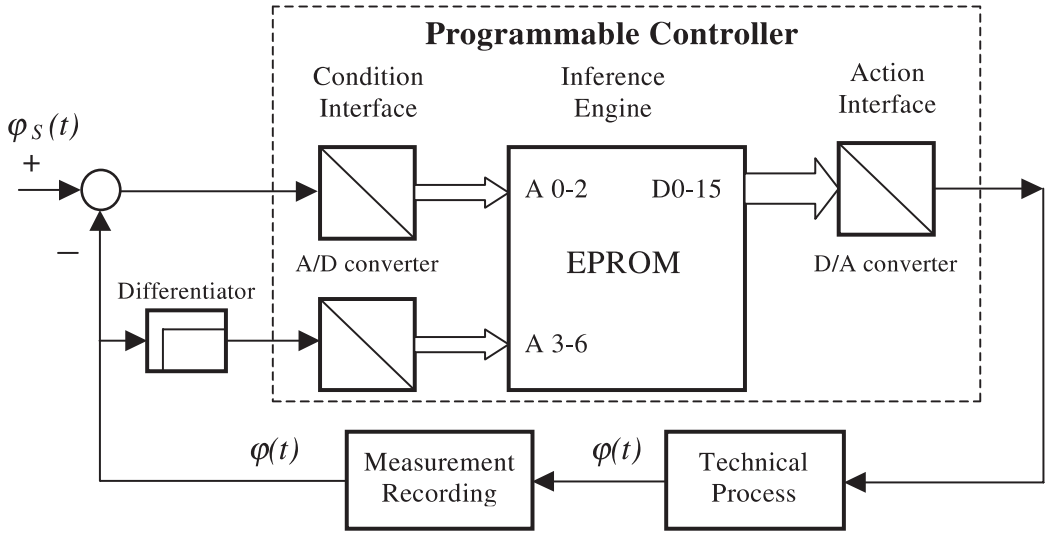

Fig. 5. Structure of a rule-based controller of a pointer's rotation.

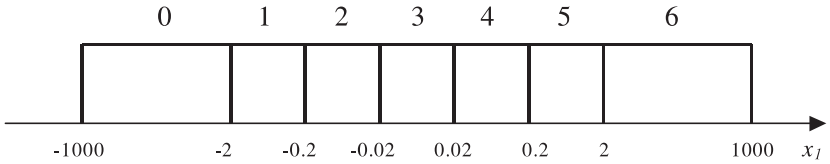

Fig. 6. Input intervals of the rotation angle's control deviation (A0–2).

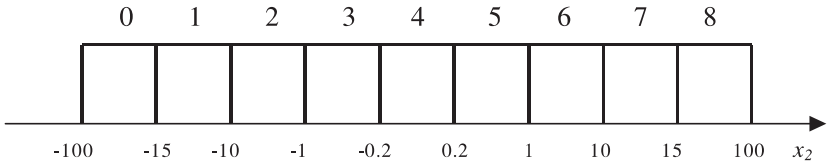

Fig. 7. Input intervals of the angular velocity (A3–6).

Table 1. Section of a rule base table for one output variable.

Interval of input x_1	Interval of input x_2	Actuation I_D
2	6	1.3
2	7	3
2	8	5
3	0	-5
3	1	-4
3	2	-1.8
3	3	-0.3
3	4	0

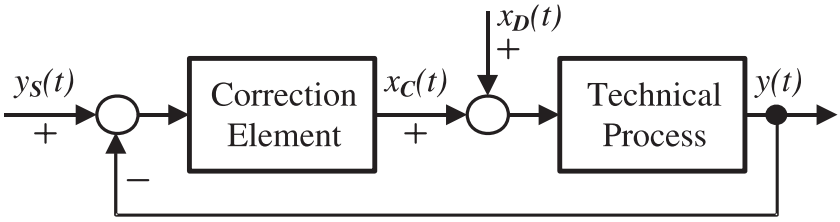

Fig. 8. Principle of a common conventional control system.

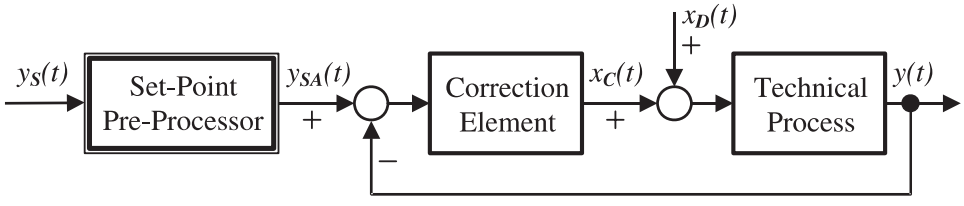

Fig. 9. Principle of a controller working with a set-point pre-processor (SPP controller).

the continuous temporal behaviour of their actuations. Moreover, compared with conventional controllers such as PD or PID, the influence of disturbances on control performance is very small. The only differences between the design principle of a conventional controller for a single loop feedback system as shown in Fig. 8 and the one of an SPP controller as depicted in Fig. 9 are that the SPP controller contains a set-point pre-processor and that its correction element gains are higher.

The set-point pre-processor input $y_S(t)$ is the desired temporal graph of the controlled value. Its output is the internal set-point graph $y_{SA}(t)$. Using the difference $y_{SA}(t) - y(t)$ instead of the control error $y_S(t) - y(t)$ as static controller relation inputs leads to the great advantage that the behaviour of the controlled system is much more predictable as the one of a conventional controller. Moreover, the real performance of a properly designed SPP controller is usually much closer to the "best physically possible" controller behaviour. Thus, in controller design priorities and objec-tives are almost freely selectable.

As displayed in Fig. 10, we consider the elementary example of controlling the position of a sleigh on a horizontal line. We shall only show the basic design principle of controllers working with set-point pre-processors. Design and behaviour of a conventional PD controller (PD_1) is compared with the ones of two PD controllers working with set-point pre-processors (SPP_2 and SPP_3). The mathematical model of the control system is characterised by the following differential equation:

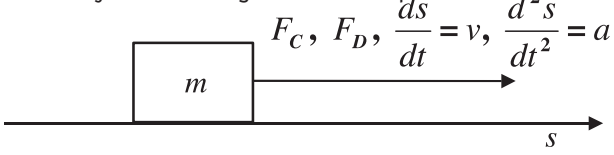

Fig. 10. Sleigh on a horizontal line.

$$F_C(t) + F_D(t) = m \cdot \frac{d^2s(t)}{dt^2} = m \cdot a(t) \qquad (2)$$

The total force $F_C + F_D$ and the speed of the sleigh v have to be zero in the stationary state. Discrete degrees of freedom of robots often have similar properties. The control problem is to guide the sleigh's position s by adjusting the force F_C. The controlled system has an integrative behaviour. Thus, the performance of the best PD controller is better than the performance of the best PID controller, if the disturbing force F_D is zero.

Design of the Conventional* PD *Controller PD_1. In this example, the force controlling the sleigh $F_C(t)$ is limited between -25N and 25N. It was tried to design controllers with the best performance assuming that the maximum amount of the set-point jump is 100m. Moreover, the control errors due to disturbing forces $F_D(t)$ should be as small as possible. The control deviation $s_S(t) - s(t)$ of the conventional PD controller PD_1 may be very large. The force $F_C(t)$ is about $(s_S(t) - s(t))$ multiplied by the proportional amplification P. Thus, P has to be rather small, otherwise the controlled system will be unstable. The chosen mathematical algorithm of the correction element is described by the following equation:

$$\begin{aligned} F_D(t) &= P \cdot (s_S(t) - s(t)) + PD \cdot v(t) \\ &= 3.5Nm^{-1} \cdot (s_S(t) - s(t)) - 3Nsm^{-1} \cdot v(t) \end{aligned} \qquad (3)$$

These assumptions lead to some great disadvantages, viz., inherent tardiness after a set-point jump and after a sudden change of $F_D(t)$. Another detriment is the large permanent control error if a permanent force disturbs the sleigh.

Construction of the* SPP *Controllers SPP_2 and SPP_3. As described above, controller design objectives are nearly freely selectable. In this example, we consider the desired temporal graph of the force controlling the sleigh $F_A(t)$. The task of the set-point pre-processor is, therefore, to compute the internal set-point graph $s_{SA}(t)$ by assuming the desired temporal graphs $s_S(t)$ and $F_A(t)$, whereas s_S is the set-point of the controlled value. There is a static relationship between the position of the sleigh $s(t)$ and the force accelerating it. So, lag or dead time elements are not contained in the mathematical model of the controlled system. We do not consider occasionally occurring disturbing forces ($F_D = 0$). The sleigh mass is m =1kg. These considerations lead to a simple equation of the set-point pre-processor's calculator:

$$s_{SA}(t_1) = s_{SA}(t_0) + \frac{1}{m} \cdot \int_{t_0}^{t_1} \int_{t_0}^{t} F_A(\tau) \cdot d\tau dt \qquad (4)$$

Set-point pre-processor of the SPP *controller SPP_2:* The set-point pre-processor calculates the fastest internal set-point graph physically possible by considering that the controlling force is within the controller's actuation range (-25N ≤ $F_C(t)$ ≤ 25N). Therefore, $F_A(t)$ is either the maximum or the minimum of the controlling force $F_C(t)$ or zero. This property has the great disadvantage that the controller has small force reserves if the real technical process is different from the one considered in the design of the controller, e.g., due to disturbances or aging processes. An example of an internal set-point graph $s_{SA}(t)$ is shown in Fig. 11.

Set-point pre-processor of the SPP *controller SPP_3:* The design principles of the SPP controllers SPP_2 and SPP_3 resemble each other. The only difference is that $F_A(t)$ is limited between -6.25 N and 6.25 N. Thus, the theoretical reserve force of the SPP controller SPP_3 is 25 N - 6.25 N = 18.75 N. On the other hand, as displayed in Fig. 11, the time after which the internal set-point graph $s_{SA}(t)$ reaches the set-point $s_S(t)$ is twice as long.

Correction element of the SPP *controllers:* Generally, the design principle of controllers working with set-point pre-processors is to distinguish between the desired temporal graph of the controlled value (in this example, $s_S(t)$) and its internal set-point graph ($s_{SA}(t)$). The mathematical algorithm of a set-point pre-processor is designed in such a way that the difference between the internal set-point graph $s_{SA}(t)$ and the actual controlled value $s(t)$ is rather small. Due to this property, it is both possible and essential that the amplifications of the controller are high. Owing to the small difference $s_{SA}(t)$-$s(t)$, it is possible to design a controller with high amplification, even though its actuations are limited. The controller's amplification also has to be large in order to decrease this difference. The temporal graph $s_{SA}(t)$-$s(t)$ and, thus, the performance of the controller primarily depends on the controller's proportional amplification P_P. Therefore, SPP controllers should be designed according to the following procedure:

1. *Design of the set-point pre-processor:* The controlled values should be able to follow the internal set-point graphs even if the controlled technical process is disturbed by any anticipated fault.
2. *Coarse determination of the controller's proportional amplification P_P:* Afterwards, its differential and integral amplifications P_D and P_I are set. Meaningful values of P_P, P_D and P_I mutually affect each other.

These considerations lead to the following actuation function of the SPP controllers:

$$F_D(t) = P_P \cdot (s_{SA}(t) - s(t)) + P_D \cdot v(t) \quad (5)$$
$$= 250\,Nm^{-1} \cdot (s_S(t) - s(t)) - 26\,Nsm^{-1} \cdot v(t)$$

Their proportional amplification P_P is 250 Nm^{-1} instead of P =3.5 Nm^{-1} by the conventional PD controller PD_1.

Performance of the controllers: As shown in Figs. 11 and 12, at the moment t_1 (in this example, t_1 = 0) a set-point jump immediately leads to a sudden change of the conventional controller's control error. At the time t_1 the control deviation jump $s_S(t_1) - s(t_1)$ corresponds to the amount of the set-point jump-with regard to the stability of the controlled system, the amplification of the controller has to be small, particularly if the maximum amount of the set-point jump is high. Therefore, conventional controllers are usually slow and not very stable. The aperiodic border case is assumed in the design of the conventional PD controller PD_1. Thus, the sleigh reaches its desired values $s_S(t)$ without overshoots if it is not greatly disturbed. However, only small enlargements of the controller's amplifications result in big overshoots, and still, the time after which the set-point is reached for the first time is much longer than the one for an SPP controller which exhibits a minute overshoot, at most.

In contrast to the control deviation of a conventional

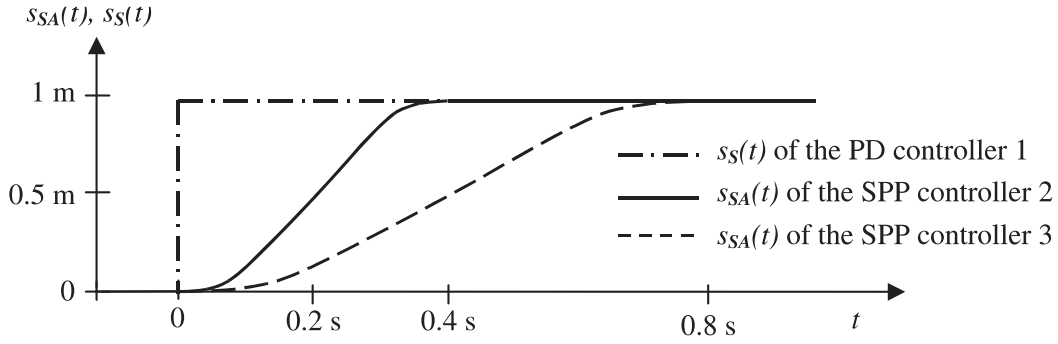

Fig. 11. Correction element inputs leading to a set-point jump from s(t) = 0 to 1 m.

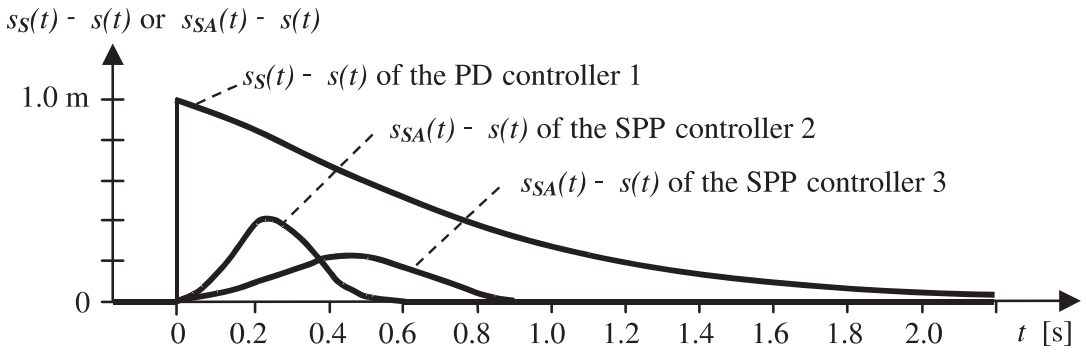

Fig. 12. Control errors after a set-point jump from zero to s = 1 m.

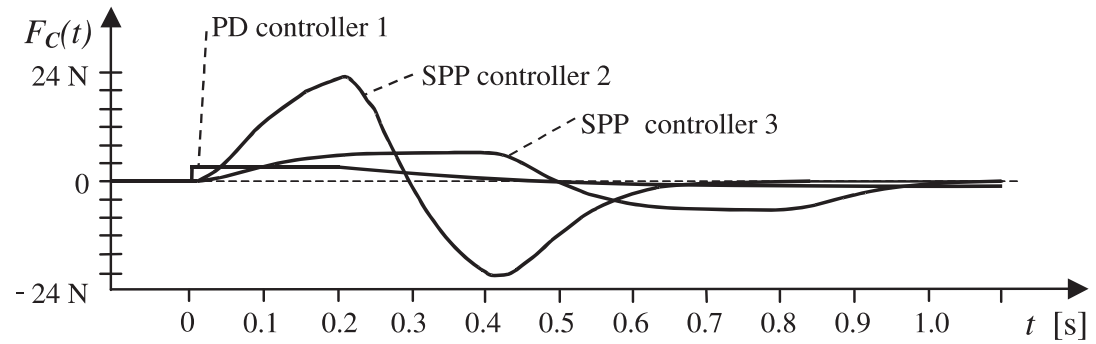

Fig. 13. Controlling force $F_C(t)$ after a set-point jump from zero to s = 1 m.

Table 2. Comparison of the performance of the PD controller PD_1, the behaviour of the SPP *controller SPP_2, and the performance of the* SPP *controller SPP_3 after a set-point jump from zero to s_S =1 m or to s_S = 100 m, respectively. The amount of the first overshoot is s_B, t_D is the time after which the output error is smaller than 0.1 m (t_E: $|\Delta s| < 0.01$ m; t_F: $|\Delta s| < 0.001$ m).*

Type	s_S	Over-shoot	$\|\Delta s\| < 0.1$	$\|\Delta s\| < 0.01$	$\|\Delta s\| < 0.001$	Range of F [N]	
	[m]	s_B [m]	t_D [s]	t_E [s]	t_F [s]	F_{C-}	F_{C+}
PD_1	1	–	1.4	2.6	3.6	-0.76	3.5
SPP_2	1	–	0.43	0.56	0.67	-20.92	22.92
SPP_3	1	0.0004	0.73	0.88	0.96	-6.28	6.27
PD_1	100	13.3	7.0	8.0	9.0	-25	25
SPP_2	100	–	4.03	4.16	4.27	-25	25
SPP_3	100	–	7.9	8.14	8.35	-6.25	6.25

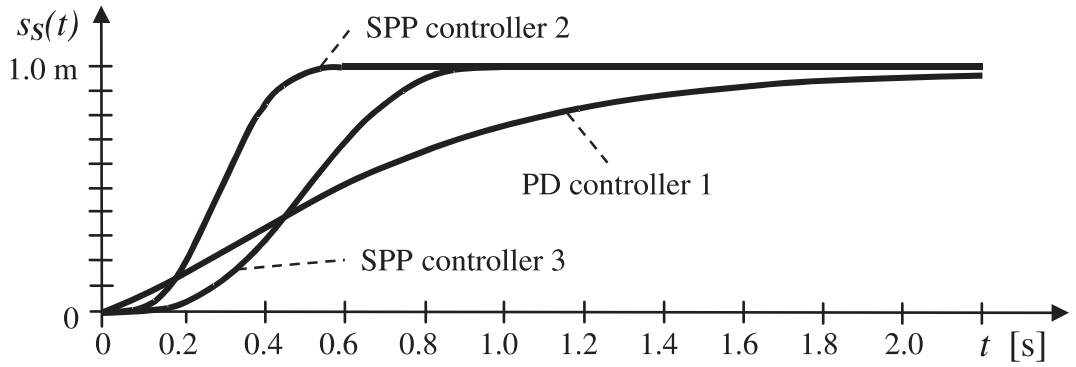

Fig. 14. Controlled value s(t) after a set-point jump from zero to s =1 m.

controller, the SPP controller's temporal graph of its difference $s_{SA}(t) - s(t)$ is continuous. Therefore, the actuations $F_C(t)$ of an SPP controller are very continuous. In this example, the actuation graph $F_C(t)$ roughly has the shape of one sine wave. The positive force half-wave speeds up the sleigh towards the set-point. Afterwards, the negative force half-wave decreases the sleigh's speed and guides it to the desired position. The temporal graph of the controlling force $F_C(t)$ and, therefore, its maximum and minimum amounts can be well determined. Due to these properties, the maximum amount of $F_C(t)$ and the reaction speed of SPP controllers can be much larger than the ones of conventional controllers as depicted in Figs. 13 and 14 and Table 2. Nevertheless, the sleigh reaches its desired position without overshoots.

6. Case Study: Robot Control

We consider *position control of a robot with two degrees of freedom*. The robot arm rotates on a horizontal plane around a pivot with the angular velocity ω. The distance r between the burden and the pivot of the robot is changeable with the velocity v. Thereby, the axis of the translation movement runs through the pivot. The task is to control the position of the burden with the mass m_L, which is located at the end of the robot arm, by adjusting the force F_C and the torque M_C.

Design of a Conventional Robot Controller Cv - RC. The principle of non-linear decoupling of systems as presented in [3] is used as design principle for this controller. Its controlling force $F_C(t)$ and its controlling torque $M_C(t)$ are not limited. It was tried to find the parameters of the fastest controller that leads to the assumption of the desired values without considerable overshoots by considering that m_L = 20 kg. Thus, the conventional controller *Cv - RC* is described by the following equations for the distance r and the angle of rotation φ as functions of time:

$$F_C(t) = 70Nm^{-1} \cdot (r_S(t) - r(t)) - 119Nsm^{-1} \cdot v(t) - \{70Nm^{-1}s^2 \cdot r(t) - 25Ns^2\} \cdot \omega^2(t) \quad (6)$$

$$M_C(t) = \{17.67kgm^2 - 50kgm \cdot r(t) + 70kg \cdot r^2(t)\} \times \{1s^{-2}(\varphi_S(t) - \varphi(t)) - 1.75s^{-1}\omega(t)\} + \{140kg \cdot r(t) - 50kgm\} \cdot v(t) \cdot \omega(t) \quad (7)$$

Design of an SPP Robot Controller SPP-RC. The conventional controller *Cv-RC* and the SPP controller *SPP-RC* compensate for the centrifugal force and the Coriolis torque of the robot. Thus, the translation dynamics controller only affects the dynamics of the robot arm translation, and the dynamics of the robot rotation is only influenced by the rotation dynamics controller. As shown in Fig. 15, apart from the facts that the dynamics controllers work with a set-point pre-processor, and that the control algorithm is approximated by an IF controller, *SPP-RC* has the same design as *Cv-RC*. The set-point pre-processor of *SPP-RC* calculates the internal set-point graphs of the controlled values $r_{SA}(t)$ and $\varphi_{SA}(t)$ by assuming that the necessary actuations controlling the robot are within the ranges of possible actuations of the control system. Thus, the position of the burden $r(t)$ and $\varphi(t)$ is, in principle, able to follow the internal set-point graphs.

With this example, we shall show that SPP controllers exhibit excellent performance under any type of stress, even if the technical process to be controlled is insufficiently understood when constructing the set-point preprocessor. Thereby, the following simplifications are made: The non-linearities of the mathematical process model, the disturbing force $F_D(t)$ and the disturbing torque $M_D(t)$, and the influences of the controllers' differential parts on the actuations are all not considered. Thus, $r_{SA}(t)$ and $\varphi_{SA}(t)$ have similar shapes as $s_{SA}(t)$ depicted in Fig. 11. An example of $r_{SA}(t)$ is shown in Fig. 16. As described above, the real position of the burden is very close to $r_{SA}(t)$ and $\varphi_{SA}(t)$. Hence, the burden motion is of utmost continuity. The mathematical model of the SPP robot controller *SPP-RC* is characterised by the following equations, in which the first part each describes the dynamics controller and the final product the decoupling of the centrifugal force or of the Coriolis torque, respectively:

$$F_C(t) = P_{Pr}(r_{SA}(t) - r(t)) + P_{Dr} \cdot v(t) - \{70\,Nm^{-1}s^2 \cdot r(t) - 25\,Ns^2\} \cdot \omega^2(t) \quad (8)$$

$$M_C(t) = P_{P\varphi}(\varphi_{SA}(t) - \varphi(t)) + P_{D\varphi} \cdot \omega(t) + \{140\,kg \cdot r(t) - 50\,kgm\} \cdot v(t) \cdot \omega(t) \quad (9)$$

It is both possible and essential that the constants P_{Pr}, $P_{P\varphi}$, P_{Dr} and $P_{D\varphi}$ of the dynamics controllers are high. Furthermore, meaningful values of the proportional amplification, e.g., P_{Pr}, and of the differential amplification, e.g., P_{Dr}, of a single dynamics controller mutually affect each other. These amplifications are, however, otherwise almost freely selectable. Since the controller compensates for the centrifugal force and the Coriolis torque of the robot, it is possible to design a single dynamics controller without consideration of the robot's other degrees of freedom.

The compared controllers are optimised for simultaneous set-point jumps from zero to r =0.1 m and φ =1 rad. The rotation and the translation set-point jumps need about the same build-up periods, and start at the same time to provide for the most difficult case to control. It was tried to find the parameters of the best controller with the properties described above. They are: P_{Pr} = 5800 Nm^{-1}, $P_{P\varphi}$ = 4000 Nm, P_{Dr} = -1260 Nsm^{-1} and $P_{D\varphi}$ = -450 Nms. In this example, the proportional amplifications of the SPP controller *SPP - RC* are more than 80 times larger than the proportional amplifications of the conventional controller *Cv - RC*.

Design of IF Controllers as Execution Platforms. The dynamics controllers and the decouplings of the controllers (centrifugal force and Coriolis torque) are implemented on safety-licensable IF controllers. The rule base table controlling the translation is stored in EPROM 1, and the rule base table controlling the rotation is stored in EPROM 2. A section of the rule base table in EPROM 1 is shown in Table 3.

For the signals $r_{SA}(t)$ - $r(t)$, $v(t)$, $\varphi_{SA}(t)$ - $\varphi(t)$ and $\omega(t)$ analogue-to-digital converters with unequally spaced input intervals featuring higher precision around zero as shown in Fig. 17 are selected. Due to this input interval spacing of the A/D converters, the EPROM storage space required is under the given conditions by orders of magnitude less than the storage space required for a controller with equally spaced input intervals. The IF controller implementing *SPP - RC* contains two 16 bits wide 0.5 Mwords EPROMs. A further effective way to reduce the storage space required is to cascade IF controllers (rule base tables). For instance, it is also possible to implement *SPP - RC* on an IF controller with one 16 Kwords EPROM, two 1 Kwords EPROMs and one 0.5 Kwords EPROM, only.

It was tried to design the best controllers by considering that m_L = 20 kg. In Table 4, we compare the performance of the conventional robot controller *Cv - RC* with the one of the SPP robot controller *SPP - RC* under a number of circumstances: mass of the burden m_L = 20 kg (a); m_L =0 kg (b); m_L = 50 kg (c); m_L = 20 kg by considering damping due to springs (d); m_L = 20 kg by assuming coincident disturbing impulses $F_D(t)$ and $M_D(t)$ (e). Performance is expressed in terms of the size of the first overshoot r_0 (φ_0), the time t_r (t_φ) after which the output error is smaller than 0.0001m (rad), and the permanent control error $\Delta r_{(t=\infty)}$ ($\Delta\varphi_{(t=\infty)}$).

The desired actuation functions of the SPP robot controller *SPP - RC* are approximated by rule bases and implemented on a safety-licensable IF controller, which works very well and shows excellent performance. This case study reveals that *SPP - RC* has, in addition to the ones mentioned in the Conclusion and the Introduction, many advantages as compared to the conventional robot controller *Cv - RC*:

- The time after which the control error is smaller than 0.0001 m after a translation set-point jump is between 4 and 8 times shorter, and the time after which the control error is smaller than 0.0001 rad after a rotation set-point jump is between 6 and 8 times shorter.
- A permanent force disturbing the robot arm leads

Table 3. Part of the rule base table (cause effect table) stored in EPROM 1.

Cause				Effect
if	**and**	**and**	**and**	**then**
Interval of input ω	Interval of input r	Interval of input v	Interval of input $r_{SA} - r$	Actuation F_C [N]
30	11	5	28	122.69
30	11	5	29	234.05
30	11	5	30	591.33
30	11	6	0	-570.18
30	11	6	1	-212.90
30	11	6	2	-101.54

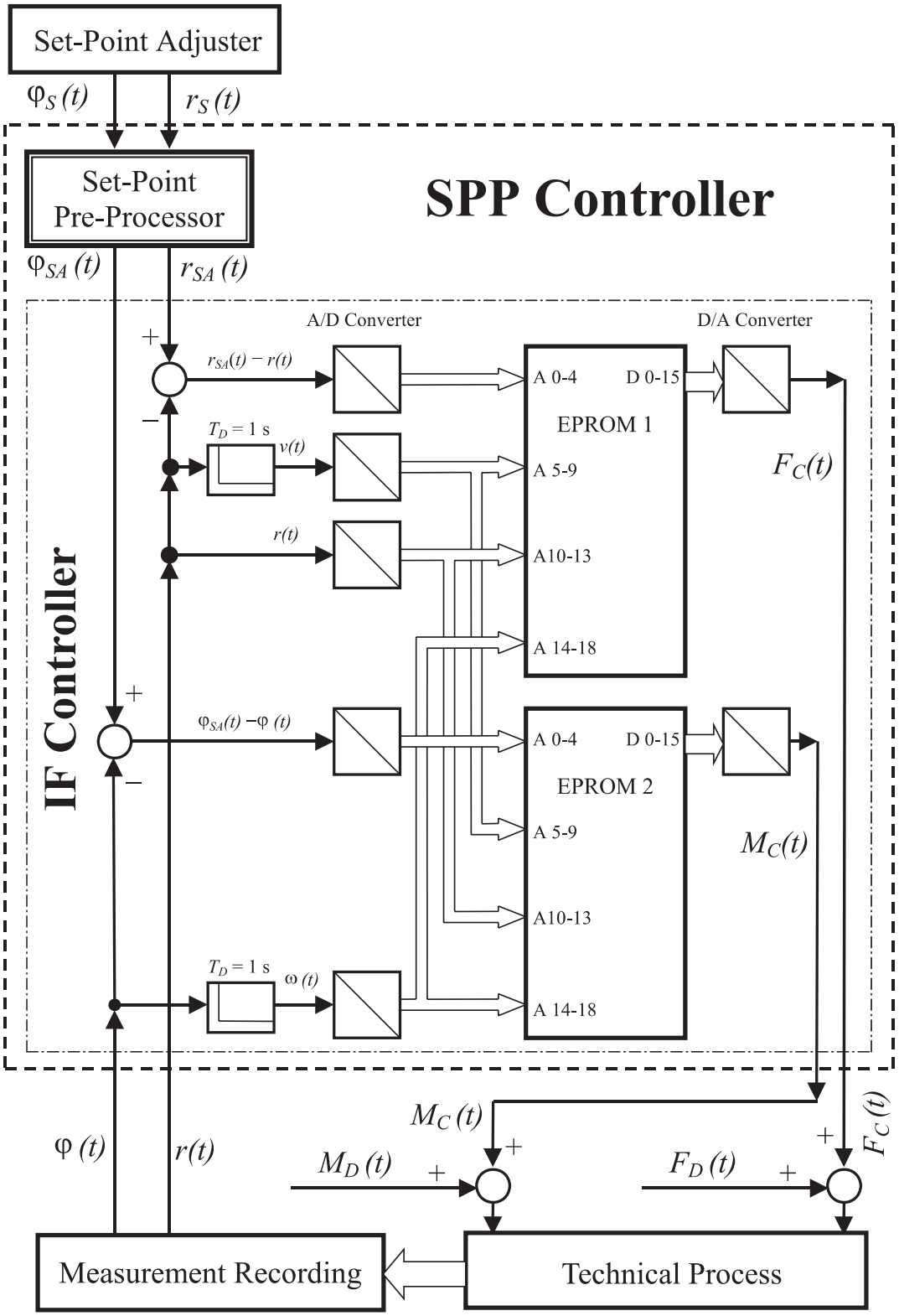

Fig. 15. Principle of the SPP robot controller SPP - RC.

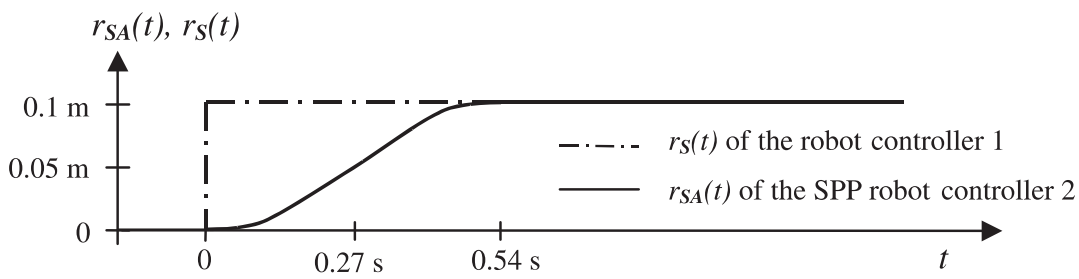

Fig. 16. Correction element inputs leading to a set-point jump from r(t)=0 to 0.1 m.

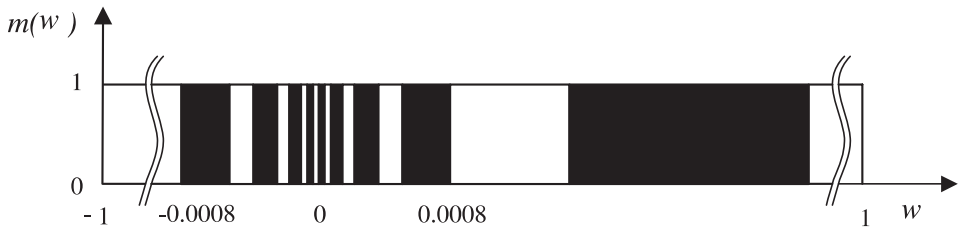

Fig. 17. Typical unequally spaced input intervals (e.g., angular velocity ω in rads^{-1})

Table 4. Behaviour after simultaneous set-point jumps from zero to r_S =0.1 m and φ_S =1 rad (a–d), and after disturbing impulses (100 N and 100 Nm) with the impulse width 1 s (e).

	Translation movement					
	r_0 $[10^{-3}m]$		t_r $[s]$		$\Delta r_{(t=\infty)}$ $[10^{-3}m]$	
Simul.	Cv-RC	SPP-RC	Cv-RC	SPP-RC	Cv-RC	SPP-RC
(a)	0.09	0	5.8	1.21	0	0
(b)	0	0	9.3	1.49	0	0
(c)	3.7	3.8	11.6	1.40	0	0
(d)	0	0	never	never	59	1.7
(e)	570	13.2	10.48	0.96	0	0

	Rotation movement					
	φ_0 $[10^{-3}rad]$		t_φ $[s]$		$\Delta\varphi_{(t=\infty)}$ $[10^{-3}rad]$	
Simul.	Cv-RC	SPP-RC	Cv-RC	SPP-RC	Cv-RC	SPP-RC
(a)	0.16	0	9.7	1.21	0	0
(b)	0.08	0	7.8	1.27	0	0
(c)	0.32	0	10.7	1.34	0	0
(d)	0	0	never	never	113	0.4
(e)	3340	26.3	12.08	0.41	0	0

Table 5. Comparison of the properties of IF controllers and of conventional controllers.

Properties of **IF controllers**	Properties of **conventional controllers**
They are licensable for the highest safety requirements, viz., Safety Integrity Level 4 of IEC 61508, although they can approximate any arbitrarily complex control algorithm.	According to the authorities, it is not admitted to use PID or other complex control algorithms in safety-related applications.
The use of appropriate, complex control algorithms enhances the inherent safety of technical processes to be controlled.	Very simple safety techniques lead to reductions of the inherent safety of controlled systems.
Operating in a strictly cyclic fashion, IF controllers exhibit fully predictable real-time behaviour.	The time needed for operation is usually not predictable. Thus, proper real-time behaviour cannot be guaranteed for these controllers.
Their loop execution times are extremely short, e.g., the one of the prototype is 0.8 μs.	The loop execution times normally exceed 1000 μs.

to an about 80 times smaller control error, and a permanent disturbing torque leads to a 200 times smaller control error.

- The permanent control error due to damping by springs $\Delta r_{(t=\infty)}$ is about 35 times smaller, and $\Delta\varphi_{(t=\infty)}$ is more than 200 times smaller.
- The over-shoot after a force impulse disturbing the robot arm is 43 times smaller, and a disturbing torque impulse leads to an about 120 times smaller overshoot. Furthermore, the duration between the time after which the disturbing force F_D (torque M_D) vanishes after an F_D (M_D) impulse and the time after which the control error is smaller than 0.0001 m (rad) is about 10 times (30 times) shorter.

7. Conclusion

The major advantages of the IF controller are its inherent safety and speed. Its main characteristics are input conditioning by analogue-to-digital converters and inference by look-up in rule tables. The controller consists of a few relatively simple, industry standard hardware

Table 6. Comparison of the properties of SPP and of conventional control algorithms.

Properties of **SPP control**	Properties of **conventional algorithms**
The control behaviour can be determined almost freely within the physical limits.	The control behaviour can be determined only coarsely within small ranges.
The objectives of controller design can be selected almost freely, e.g., stability, speed, reduction of expenses and environmental damages due to a technical process.	Often the main design problem is the stability of control systems, and it is not possible to meet other design objectives as well.
The controller gain is very high. Thus, the stability of control systems against disturbances and changes of technical processes is extremely high. Furthermore, the process speed attainable is high.	The controller gain is small. Control system are, therefore, very sensitive to disturbances and parameter changes of the processes to be controlled, and control systems react slowly.
The temporal graphs of actuations are steady and continuous.	The actuation shapes are unsteady. This leads to clear performance degradation.

modules. Hence, safety-licensing of the hardware can follow well understood and long established procedures. The main task of a safety proof is to verify an implemented rule set's correctness by inspection. This method leads back in one easy step from machine code to problem specifications. For the architecture presented, the effort required to utilise inspection to safety-license control software is by several orders of magnitude smaller than to verify sequential programs running on Von Neumann computers. Owing to its simple construction and its software form's utmost simplicity, the IF controller can be licensed for applications with the highest safety requirements, i.e., those of Safety Integrity Level 4. Working in a strictly periodic fashion with no jitter, the controller's real-time behaviour is predictable in full. Its hardware operation is supervised by a fail-safe logic immediately initiating an emergency shut-down in case of a malfunction. Thus, the use of IF controllers for safety-related functions eliminates the disadvantages due to the restrictions imposed by the licensing authorities [2].

Basically, it is possible to approximate any control algorithm to any sufficient precision and with reasonable effort by a rule-based one. In general, the better the approximation and the larger the number of inputs are the higher is the EPROM capacity needed in an executing IF controller, which may be very large. In many practical cases and especially if the structure of an IF controller is customised to the mathematical model of a given control system, however, the memory requirements for the corresponding rule base table often become surprisingly small. Therefore, difficult control tasks can very easily and transparently be solved by IF controllers. Table 5 summarises the benefits achieved with them. A prototype of an IF controller with a 64 Kword EPROM was built. It is cheap and small, and runs very fast with a loop execution time of 800 ns. As controllers often need to provide certain functions such as timers, counters, internal storage as well as digital and analogue inputs and outputs, a range of such modules was implemented which can be plugged into the prototype.

The approach of set-point pre-processors makes use of an advanced mathematical control algorithm. As described above and in Table 6, the use of SPP control normally leads to essential performance improvements, even if the mathematical model of a technical process assumed in the design of a controller is very coarse and inaccurate. The SPP robot controller *SPP - RC* described above combines the advantages of SPP control algorithm and IF controller as execution platform.

AUTHORS
Peter Vogrin, Wolfgang A. Halang* - Chair of Computer Engineering, Fernuniversität 58084 Hagen, Germany. E-mail: wolfgang.halang@fernuni-hagen.de.
* Corresponding author

References

[1] Fagan M.E., „Design and Code Inspection to Reduce Errors in Program Development". *IBM Systems Journal*, 1976, vol. 15, no. 3, 182-211.

[2] „TÜV Cooperation Functional Safety: List of Type Approved Programmable Electronic Systems (PES, PLCs)", 2008, *http://www.tuv-fs.com*.

[3] Hoyer H., Freund E., „The Principle of Non-linear Decoupling of Systems with Application to Industrial Robots". *Regelungstechnische Praxis rtp*, 22, 1980, pp. 80-87 and pp. 116-126.

[4] „International Standard IEC 61508-1: *Functional Safety of Electrical/Electronic/Programmable Electronic Systems: Generic Aspects - Part 1:* General Requirements". Geneva: International Electrotechnical Commission 1998.

[5] Krebs H., Haspel U., „Ein Verfahren zur Software-Verifikation". *Regelungstechnische Praxis rtp*, 1984, 26, pp. 73-78.