(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
06 December 2018 (06.12.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/219943 A1

(51) International Patent Classification:
B25J 9/16 (2006.01) *G05B 13/02* (2006.01)

(21) International Application Number:
PCT/EP2018/064059

(22) International Filing Date:
29 May 2018 (29.05.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10 2017 005 081.3
29 May 2017 (29.05.2017) DE

(71) Applicant: **FRANKA EMIKA GMBH** [DE/DE]; Infanteriestr. 19 Gebäude 5, 80797 München (DE).

(74) Agent: **RÖSLER, Frank**; RÖSLER RASCH VAN DER HEIDE & PARTNER, Bodenseestr. 18, 81241 München (DE).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,

WO 2018/219943 A1

(54) Title: SYSTEM AND METHOD FOR CONTROLLING ACTUATORS OF AN ARTICULATED ROBOT

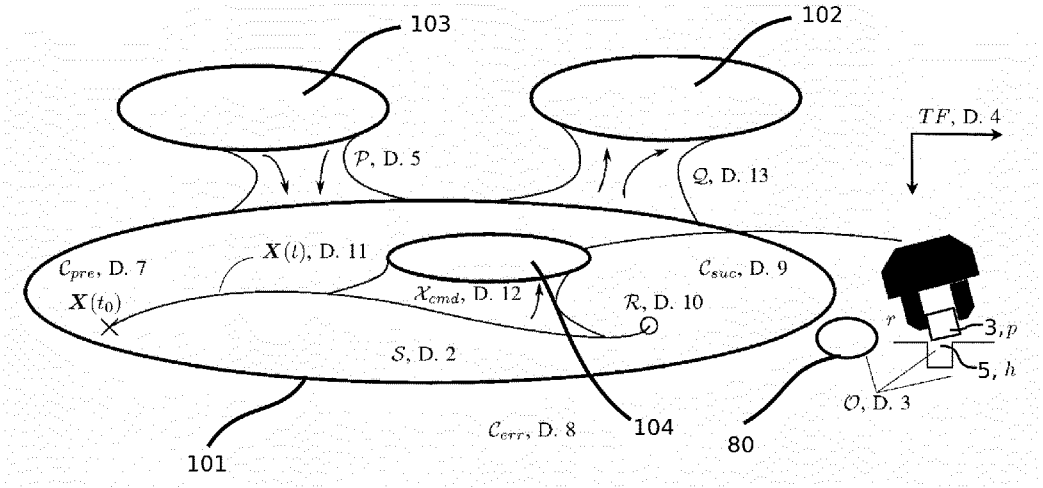


Fig. 2

(57) Abstract: The invention relates to a system for controlling actuators of an articulated robot (80) and for enabling the robot (80) executing a given task, comprising a first unit (101) providing a specification of robot skills s selectable from a skill space depending on the task, a second unit (102), wherein the second unit (102) is connected to the first unit (101) and further to a learning unit (103) and to an adaptive controller (104), wherein the adaptive controller (104) receives skill commands χ_{cmd}, wherein the skill commands χ_{cmd} comprise the skill parameters P_I, wherein based on the skill commands χ_{cmd} the controller (104) controls the actuators of the robot (80), wherein the actual status of the robot (80) is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller (104) and to the second unit (102), wherein based on the actual status, the second unit (102) determines the performance $Q(t)$ of the skill carried out by the robot (80), and wherein the learning unit (103) receives P_D, and $Q(t)$ from the second unit (102), determines updated skill parameters $P_I(t)$ and provides $P_I(t)$ to the second unit (102) to replace hitherto existing skill parameters P_I.

[Continued on next page]

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

System and method for controlling actuators of an articulated robot

The invention relates to a system and method for controlling actuators of an articulated robot.

The traditional way of programming complex robots is sometimes sought to become more intuitive, such that not only experts but also shopfloor workers, in other words laymen, are able to utilize robots for their work. The terms “skill-” and “task-based programming” are very important in this context. “Skills” are in particular some formal representation of predefined actions or movements of the robot. Several approaches to programming with skills exist, e.g. [1], [2], [3], and they are in particular mostly viewed independently from the controller, i.e. in particular the controller only executes commands calculated by the skills implementation. From this it can be seen that the underlying controller is a common factor for manipulation skills and thus provides a set of parameters that is shared by them. It is, however, according to general knowledge not efficient and often not even feasible to use the same parameter values for all manipulation skills. Typically, it is even not possible considering the same skill in different environments. Depending on the particular situation, the parameters have to be adapted in order to account for different environment properties such as rougher surfaces or different masses of involved objects. Within given boundaries of certainty the parameters could be chosen such that the skill is fulfilled optimally, or at least close to optimal with respect to a specific cost function. In particular this cost function and constraints are usually defined by the human user with some intention e.g. low contact forces, short execution time or a low power consumption of the robot. A significant problem in this context is the tuning of the controller parameters in order to find regions in the parameter space that minimize such a cost function or are feasible in the first place without necessarily having any pre-knowledge about the task other than the task specification and the robots abilities. Several approaches were proposed that cope with this problem in different ways such as [4], in which learning motor skills by demonstration is described. In [5] a Reinforcement Learning based approach to acquiring new motor skills from demonstration is introduced. The authors of [6], [7] employ Reinforcement Learning methods to learn motor primitives that represent a skill. In [8] a supervised learning by demonstration approach is used with dynamic movement primitives to learn bipedal walking in simulation. An early approach utilizing a stochastic real-valued reinforcement learning algorithm in combination with a nonlinear multilayer artificial neural network in order to learn robotic skills can be found in [9]. Soft robotics is shown in [10], while

impedance control to apply the idea to complex manipulation problems is shown in [11]. An adaptive impedance controller is introduced in [12]. Both are adapted during execution, depending on the motion error and based on four physically meaningful meta parameters. From this the question arises how these meta parameters are to be chosen with respect to the environment and the problem at hand.

It is the objective of the invention to provide a system and a method for improved learning of robot manipulation skills.

A first aspect of the invention relates to a system for controlling actuators of an articulated robot and for enabling the robot to execute a given task, comprising:
- a first unit providing a specification of robot skills *s* selectable from a skill space depending on the task, with a robot skill *s* being defined as a tuple (S, O, C_{pre}, C_{err}, C_{suc}, R_m, χ_{cmd}, X, P, Q) with
S: a Cartesian product of I subspaces ζ_i : $S=\zeta_{i=1}\times\zeta_{i=2}\times...\times\zeta_{i=I}$ with $i=\{1,2,....,I\}$ and $I\geq 2$,
O: a set of physical objects,
C_{pre}: a precondition,
C_{err}: an error condition,
C_{suc}: a succcess condition,
R: a nominal result of ideal skill execution,
χ_{cmd} : skill commands,
X: physical coordinates,
P: skill parameters, with P consisting of three subsets P_t, P_l, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_l being the parameters not known initially which need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_l,
Q: a performance metric, while $Q(t)$ is denoting the actual performance of the skill carried out by the robot,
- a second unit, wherein the second unit is connected to the first unit and further to a learning unit and to an adaptive controller, wherein the adaptive controller receives skill commands χ_{cmd}, wherein the skill commands χ_{cmd} comprise the skill parameters P_l, wherein based on the skill commands χ_{cmd} the controller controls the actuators of the robot, wherein the actual status of the robot is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller and to the second unit, wherein based on the actual status, the second unit determines the performance $Q(t)$ of the skill carried out by the robot, and wherein the learning unit receives P_D, and $Q(t)$ from

the second unit, determines updated skill parameters $P_l(t)$ and provides $P_l(t)$ to the second unit to replace hitherto existing skill parameters P_l.

Preferably the subspaces ζ_i comprise a control variable, in particular a desired variable, or an external influence on the robot or a measured state, in particular an external wrench comprising in particular an external force and an external moment.

A preferred adaptive controller is derived as follows:

Consider the robot dynamics:

$$M(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q)=\tau_u+\tau_{ext} \quad (1)$$

where $M(q)$ denotes the symmetric, positive definite mass matrix, $C(q,\dot{q})\dot{q}$ the Coriolis and centrifugal torques and $g(q)$ the gravity vector. The control law is defined as:

$$\tau_u(t)=J(q)^T(-F_{ff}-K(t)e-D\dot{e})+\tau_r \quad (2)$$

where $F_{ff}(t)$ denotes the feed-forward wrench, $K(t)$ the stiffness matrix, D the damping matrix and $J(q)$ the Jacobian. The position and velocity error are denoted by $e=[e_t,e_r]^T$ and $\dot{e}=\dot{x}^*-\dot{x}$, respectively. $e_t=x^*-x$ is the translational position error and $e_r=\theta^*-\theta$ the rotational angle axis error. The dynamics compensator τ_r is defined as:

$$\tau_r=M(q)\ddot{q}^*+C(q,\dot{q})\dot{q}^*+g(q)-sign(\epsilon)\bar{\nu} \quad (3).$$

The feed forward wrench F_{ff} is defined as:

$$F_{ff}=F_d(t)+\int_{t_s}^{t}\delta F_{ff}\,dt+F_{ff,0} \quad (4)$$

where $F_d(t)$ is an optional initial time dependent trajectory and $F_{ff,0}$ is the initial value of the integrator. The controller adapts feed forward wrench and stiffness via

$$\delta F_{ff}(t)=F_{ff}(t)-F_{ff}(t-T) \quad (5)$$
$$\ldots=\alpha(\epsilon-\gamma_\alpha(t)F_{ff}(t)) \quad (6)$$

and

$$\delta K(t) = K(t) - K(t-T) \quad (7)$$
$$... = \beta\left(diag(\epsilon \circ e) - \gamma_\beta(t) K(t)\right) \quad (8)$$

The adaptive tracking error is defined as

$$\epsilon = e + \kappa \dot{\epsilon} \quad (9)$$

with $\kappa > 0$. The positive definite matrices α, β, γ_a and γ_β represent the learning rates for the feed forward and stiffness and the forgetting factors, respectively. Damping *D* is designed according to [21] and *T* is the sample time of the controller.

With the explanations above, the preferred adaptive controller is basically given.

A preferred γ_a and γ_β are derived via constraints as follows:
The first constraint of an adaptive impedance controller is the upper bound $\dot{K}_{max}$ on the speed of stiffness adaptation. Inserting $c_\alpha := \alpha \gamma_\alpha$ and $c_\beta := \beta \gamma_\beta$ into (8) leads, together with the bounded stiffness rate of change, to the relation:

$$\dot{K}_{max} = max_{t>0}\left[\left[\frac{1}{T}\beta\left(\epsilon(t) e(t) - \frac{c_\beta}{\beta} K(t)\right)\right]\right] \quad (10)$$

If it is assumed that $K(t=0)$ and $\dot{e}=0$, e_{max} is preferably defined as the amount of error at which $\dot{K}_{max} = \frac{\beta e_{max}^2}{T}$ holds. Furthermore, K_{max} denotes the absolute maximum stiffness, another constraint for any real-world impedance controlled robot. Then, the maximum value for β can be written as:

$$\beta_{max} = \frac{\dot{K}_{max} T}{e_{max}^2} \quad (11)$$

Since $\delta K(t) = 0$ and $\dot{e} = 0$ when K_{max} is reached, (10) can be rewritten as

$$0 = \beta_{max}\left(e_{max}^2 - \frac{c_\beta}{\beta_{max}} K_{max}\right) \quad (12)$$

Finally, the adaptation parameters become $c_\beta = \frac{\beta_{max} e_{max}^2}{K_{max}}$ and $\gamma_\beta = \frac{c_\beta}{\beta_{max}}$.

Finding the adaptation of feed forward wrench is preferably done analogously. This way, the upper limits for α and β are in particular related to the inherent system capabilities K_{max} and F_{max} leading to the fastest possible adaptation.

With the above explanations, preferred γ_a and γ_β are derived.

The introduced skill formalism focuses in particular on the interplay between abstract skill, meta learning (by the learning unit) and adaptive control. The skill provides in particular desired commands and trajectories to the adaptive controller together with meta parameters and other relevant quantities for executing the task. In addition, a skill contains in particular a quality metric and parameter domain to the learning unit, while receiving in particular the learned set of parameters used in execution. The adaptive controller commands in particular the robot hardware via desired joint torques and receives sensory feedback. Finally, the skill formalism makes in particular it possible to easily connect to a high-level task planning module. The specification of robot skills *s* are preferably provided as follows from the first unit:

The following preferred skill formalism is object-centered, in the sense that the notion of the manipulated objects are of main concern. The advantage of this approach is its simple notation and intuitive interpretability. The aspect of greater intuitiveness is based on the similarity to natural language:

Definition 1 (Skill): A skill *s* is an element of the skill-space. It is defined as a tuple (*S, O, C_{pre}, C_{err}, C_{suc}, R,* χ_{cmd} *, X, P, Q*).

Definition 2 (Space): Let *S* be the Cartesian product of *I* subspaces $\zeta_i = \mathbb{R}^{m_i \times n_i}$ relevant to the skill *s*, i.e.: $S = \zeta_{i=1} \times \zeta_{i=2} \times ... \times \zeta_{i=I}$ with *i={1,2,….,I} and* $I \geq 2$.

Preferably the subspaces ζ_i comprise a control variable, in particular a desired variable, or an external influence on the robot or a measured state, in particular an external wrench comprising in particular an external force and an external moment.

Definition 3 (Object): Let *o* represent a physical object with coordinates $^{\circ}X(t) \in S$ associated with it. *O* denotes the set of physical objects $o \in O$ relevant to a skill *s* with $n_o = |O|$ and $n_o > 0$. Moreover, *X(t)* is defined as $X(t) = ({}^{o_1}X(t), ..., {}^{o_n}X(t))^T$. Note that in these considerations the set *O* is not changed during skill execution, i.e. n_o = const.

Definition 4 (Task Frame): The task frame ${}^{\circ}R_{TF}(t)$ denotes the rotation from frame *TF* to the base frame ***0***. Note that we assume ${}^{\circ}R_{TF}(t)$ = const.

Definition 5 (Parameters): *P* denotes the set of all skill parameters consisting of three subsets P_t, P_I and P_D . The set $P_t \subset P$ contains all parameters resulting from a priori task knowledge, experience and the intention under which the skill is executed. In this context, it is referred to P_t also as task specification. The set $P_I \subset P$ contains all other parameters that are not necessarily known beforehand and need to be learned or estimated. In particular, it contains the meta parameters $(\alpha, \beta, \gamma_\alpha, \gamma_\beta)$ for the adaptive controller. The third subset $P_D \subset P$ defines the valid domain for P_I i.e. it consists of intervals of values for continuous parameters or sets of values for discrete ones. Thus, P_D determines the boundaries when learning P_I.

Conditions: There are preferably three condition types involved in the execution of a skill: preconditions, failure conditions and success conditions. They all share the same basic definition, yet their application is substantially different. Their purpose is to define the borders and limits of the skill from start to end:

Definition 6 (Condition): Let $C \subset S$ be a closed set and $c(X(t))$ a function $c: S \rightarrow B$ where $B = \{0, 1\}$. A condition holds iff $c(X(t)) = 1$. Note that the mapping itself depends on the specific type of condition.

Definition 7 (Precondition): C_{pre} denotes the chosen set for which the precondition defined by $c_{pre}(X(t))$ holds. The condition holds, i.e. $c_{pre}(X(t_0)) = 1$, *iff* $\forall\, x \in X : x(t_0) \in C_{pre}$. t_0 denotes the time at start of the skill execution. This means that at the beginning of skill execution the coordinates of every involved object must lie in C_{pre}.

Definition 8 (Error Condition): C_{err} denotes the chosen set for which the error condition $c_{err}(X(t))$ holds, i.e. $c_{err}(X(t)) = 1$. This follows from $\exists\, x \in X : x(t) \in C_{err}$. If the error condition is fulfilled at time *t*, skill execution is interrupted. Herein, assumptions about how this error state is resolved are not made herein since this depends in particular on the actual skill implementation and the capabilities of high-level control and planning agency.

Definition 9 (Success Condition): C_{suc} denotes the chosen set for which the success condition defined by $c_{suc}(X(t))$ holds, i.e. $c_{suc}(X(t)) = 1$ iff $\forall\, x \in X : x(t) \in C_{suc}$. If the

coordinates of all involved objects are within C_{suc} the skill execution can terminate successfully. With this it is not stated that the skill has to terminate.

Definition 10 (Nominal Result): The nominal result $R \in S$ is the ideal endpoint of skill execution, i.e. the convergence point. Although the nominal result R is the ideal goal of the skill, its execution is nonetheless considered successful if the success conditions C_{suc} hold. Nonetheless $X(t)$ converges to this point. However, it is possible to blend from one skill to the next if two or more are queued.

Definition 11 (Skill Dynamics): Let $X : [t_0,\infty] \rightarrow P$ be a general dynamic process, where t_0 denotes the start of the skill execution. The process can terminate if $(\forall\, c_{suc} \in C_{suc} : c_{suc}(X(t)) = 1) \vee (\exists\, c_{err} \in C_{err} : c_{err}(X(t)) = 1)$.
It converges to the nominal result R. This dynamic process encodes what the skill actually does depending on the input, i.e. the concrete implementation. This is preferably one of: a trajectory generator, a DMP, or some other algorithm calculating sensor based velocity or force commands. The finish time t_e is not necessarily known a priori. For example, for a search skill it can not be determined when it terminates because of the very nature of the search problem.

Definition 12 (Commands): Let $\chi_{cmd} \subset X(t)$ be the skill commands, i.e. a desired trajectory consisting of velocities and forces defined in TF sent to the controller.

Definition 13 (Quality Metric): Q denotes the set of all 2−tuples $(w,f_q(X(t))$ with $0 < w < 1$ and constraints $f_{c,i}(X(t))$. Furthermore let $q=\sum_i w_i f_{q,i}(X(t)) \forall (w_i, f_{q,i}(X(t))) \in Q$. The quality metric is a means of evaluating the performance of the skill and to impose quality constraints on it. This evaluation aims at comparing two different implementations of the same skill or two different sets of parameters P. The constraints can e.g. be used to provide a measure of quality limits for a specific task (e.g. a specific time limit). Note, that the quality metric reflects some criterion that is derived from the overall process in which the skill is executed or given by a human supervisor. Moreover, it is a preferred embodiment that a skill has several different metrics to address different demands of optimality.

With the above, the specification of robot skills s is provided in a preferably way from the first unit.

The learning unit is preferably derived as follows:

The learning unit applies meta learning, which in particular means finding the right (optimal) parameters $p^* \in P_I$ for solving a given task. Requirements: In order to learn the controller meta parameters together with other parameters such as execution velocity, several potentially suitable learning methods are to be evaluated. The method will face the following issues:

-Problems have no feasible analytic solution,

-Gradients are usually not available,

-Real world problems are inherently stochastic,

-No assumptions minima or cost function convexity,

-Violation of safety, task or quality constraints,

-Significant process noise and many repetitions,

Therefore, a suitable learning algorithm will have to fulfill the subsequent requirements:

• Numerical black-box optimization,

• No gradients required,

• Stochasticity has to be regarded,

• Global optimizer,

• Handle unknown and noisy constraints,

• Fast convergence rates.

Preferably one of the following algorithms or a combination thereof for meta learning is applied in the learning unit: Grid Search, Pure Random Search, Gradient-descent family, Evolutionary Algorithms, Particle Swarm, Bayesian Optimization.

Generally, gradient-descent based algorithms require a gradient to be available. Grid search and pure random search, as well as evolutionary algorithms typically do not assume stochasticity and cannot handle unknown constraints without extensive knowledge about the problem they optimize, i.e. make use of well- informed barrier functions. The latter point also applies to particle swarm algorithms. Only Bayesian optimization in accordance to [25] is capable of explicitly handling unknown noisy constraints during optimization. Another and certainly one of the major requirements is little, if possible, no manual tuning to be necessary. Choosing for example learning rates or making explicit assumptions about noise would break with this intention. Obviously, this requirement depends to a great deal on the concrete implementation but also on the optimizer class and its respective requirements.

Considering all mentioned requirements, the spearmint algorithm known from [26], [27], [28], [25] is preferably applied. This particular implementation requires no manual tuning, it is only required to specify the prior and the acquisition function in advance once.

More preferably, a Bayesian Optimization is applied. Preferably it is realized and implemented as follows:

In general, Bayesian optimization (BO) finds the minimum of an unknown objective function *f(p)* on some bounded set *X* by developing a statistical model of *f(p)*. Apart from the cost function, it has two major components, which are the prior and the acquisition function. Prior: In particular a Gaussian process is used as prior to derive assumptions about the function being optimized. The Gaussian process has a mean function $m: \chi \rightarrow \mathbb{R}$ and a covariance function $K: \chi \times \chi \rightarrow \mathbb{R}$. As a kernel preferably the automatic relevance determination (ARD) Matérn 5/2 kernel is used, which is given by:

$$K_{M52}(p,p') = \theta_0\left(1+\sqrt{5r^2(p,p')}+\frac{5}{3}r^2(p,p')e^{-\sqrt{5r^2(p,p')}}\right) \text{ with}$$

$$r^2(p,p') = \sum_{i=1}^{d}\frac{(p_i - p_i')^2}{\theta_i^2}.$$

This kernel has *d+3* hyperparameters in *d* dimensions, i.e. one characteristic length scale per dimension, the covariance amplitude θ_0, the observation noise ν and a constant mean *m*. These kernel hyperparameters are integrated out by applying Markov chain Monte Carlo (MCMC) via slice sampling [29]. Acquisition function: Preferably a predictive entropy search with constraints (PESC) is used as a means to select the next parameters *x* to explore, as described in [30]. Cost function: Preferably a cost metric *Q* defined as above is used directly to evaluate a specific set of parameters P_l. Also, the success or failure of the skill by using the conditions C_{suc} and C_{err} can be evaluated. Bayesian optimization can make direct use of the success and failure conditions as well as the constraints in *Q* as described in [25].

The invention presents the following advantages: The adaptive controller from [12] is extended to Cartesian space and full feed forward tracking. A novel meta parameter design for the adaptive controller based on real-world constraints of impedance control is provided. A novel formalism to describe robot manipulation skills and bridge the gap between high-level specification and low-level adaptive interaction control is introduced.

Meta learning via Bayesian Optimization [14], which is frequently applied in robotics [16], [17], [18], is the missing computational link between adaptive impedance control and high-level skill specification. A unified framework that composes all adaptive impedance control, meta learning and skill specification into a closed loop system is introduced.

According to an embodiment of the invention the adaptive controller adapts feed forward wrench and stiffness via $\delta F_{ff} = F_{ff}(t) - F_{ff}(t-T)$.

According to another embodiment of the invention the learning unit carries out a Bayesian and/or a HiREPS optimization / learning.

HiREPS is the acronym of “Hierarchical Relative Entropy Policy Search”.

According to another embodiment of the invention the system comprises a data interface with a data network, and the system is designed and setup to download system-programs for setting up and controlling the system from the data network.

According to another embodiment of the invention the system is designed and setup to download parameters for the system-programs from the data network.

According to another embodiment of the invention the system is designed and setup to enter parameters for the system-programs via a local input-interface and/or via a teach-in-process, with the robot being manually guided.

According to another embodiment of the invention the system is designed and setup such that downloading system-programs and/or respective parameters from the data network is controlled by a remote station, and wherein the remote station is part of the data network.

According to another embodiment of the invention the system is designed and setup such that system-programs and/or respective parameters locally available at the system are sent to one or more participants of the data network based on a respective request received from the data network.

According to another embodiment of the invention the system is designed and setup such that system-programs with respective parameters available locally at the system

can be started from a remote station, and wherein the remote station is part of the data network.

According to another embodiment of the invention the system is designed and setup such that the remote station and/or the local input-interface comprises a human-machine-interface HMI designed and setup for entry of system-programs and respective parameters and/or for selecting system-programs and respective parameters from a multitude system-programs and respective parameters.

According to another embodiment of the invention the human-machine-interface HMI is designed and setup such that entries are possible via „drag-and-drop"-entry on a touchscreen, a guided dialogue, a keyboard, a computer-mouse, a haptic interface, a virtual-reality-interface, an augmented reality interface, an acoustic interface, via a body tracking interface, based on electromyographic data, based on elektroenzephalographic data, via a neuronal interface, or a combination thereof.

According to another embodiment of the invention the human-machine-interface HMI is designed and setup to deliver auditive, visual, haptic, olfactoric, tactile, or electrical feedback or a combination thereof.

Another aspect of the invention relates to robot with a system as shown above and in the following.

Another aspect of the invention relates to a method for controlling actuators of an articulated robot and enabling the robot executing a given task, the robot comprising a first unit, a second unit, a learning unit, and an adaptive controller, the second unit being connected to the first unit and further to a learning unit and to an adaptive controller, comprising the following steps:

- providing a specification of robot skills *s* selectable from a skill space depending on the task by a first unit (101), with a robot skill *s* being defined as a tuple (S, O, C_{pre}, C_{err}, C_{suc}, R, χ_{cmd}, X, P, Q) with

S: a Cartesian product of I subspaces ζ_i : $S = \zeta_{i=1} \times \zeta_{i=2} \times ... \times \zeta_{i=I}$

with $i=\{1,2,....,I\}$ and $I \geq 2$,

O: a set of objects,

C_{pre}: a precondition,

C_{err}: an error condition,

C_{suc}: a success condition,

R: a nominal result of ideal skill execution,
χ_{cmd} : skill commands,
X: physical coordinates,
P: skill parameters, with P consisting of three subsets P_t, P_l, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_l being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_l,
Q: a performance metric, while $Q(t)$ is denoting the actual performance of the skill carried out by the robot,
- an adaptive controller receiving skill commands χ_{cmd} from a second unit,
wherein the second unit is connected to the first unit and further to a learning unit and to the adaptive controller and wherein the skill commands χ_{cmd} comprise the skill parameters P_l,
- controlling the actuators of the robot by the controller and based on the skill commands χ_{cmd} , wherein the actual status of the robot is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller and to the second unit,
- determining by the second unit and based on the actual status, the performance $Q(t)$ of the skill carried out by the robot,
- the learning unit receiving P_D and $Q(t)$ from the second unit, and
- determining updated skill parameters $P_l(t)$ and providing $P_l(t)$ to the second unit and replacing hitherto existing skill parameters P_l.

Preferably the subspaces ζ_i comprise a control variable, in particular a desired variable, or a external influence on the robot or a measured state, in particular an external wrench comprising in particular an external force and an external moment.

Another aspect of the invention relates to a computer system with a data processing unit, wherein the data processing unit is designed and set up to carry out a method according to one of the preceding claims.

Another aspect of the invention relates to a digital data storage with electronically readable control signals, wherein the control signals can coaction with a programmable computer system, so that a method according to one of the preceding claims is carried out.

Another aspect of the invention relates to computer program product comprising a program code stored in a machine-readable medium for executing a method according to

one of the preceding claims, if the program code is executed on a computer system.

Another aspect of the invention relates to computer program with program codes for executing a method according to one of the preceding claims, if the computer program runs on a computer system.

The sources of prior art mentioned above and additional sources are as follows:

[1]: M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, “Robot skills for manufacturing: From concept to industrial deployment,” *Robotics and Computer-Integrated Manufacturing*, 2015.

[2]: U. Thomas, G. Hirzinger, B. Rumpe, C. Schulze, and A. Wortmann, “A new skill based robot programming language using uml/p state- charts,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 461–466.

[3]: R. H. Andersen, T. Solund, and J. Hallam, “Definition and initial case- based evaluation of hardware-independent robot skills for industrial robotic co-workers,” in *ISR/Robotik 2014; 41st International Sympo- sium on Robotics; Proceedings of*. VDE, 2014, pp. 1–7.

[4]: P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, “Learning and gener- alization of motor skills by learning from demonstration,” in *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on*. IEEE, 2009, pp. 763–768.

[5]: P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, and S. Schaal, “Skill learning and task outcome prediction for manipulation,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3828–3834.

[6]: J. Kober and J. Peters, “Learning motor primitives for robotics,” in *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on*. IEEE, 2009, pp. 2112–2118.

[7]: J. Kober and J. R. Peters, “Policy search for motor primitives in robotics,” in *Advances in neural information processing systems*, 2009, pp. 849–856.

[8]: S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, “Learning movement primitives,” in *Robotics Research. The Eleventh International Symposium*. Springer, 2005, pp. 561–572.

[9]: V. Gullapalli, J. A. Franklin, and H. Benbrahim, “Acquiring robot skills via reinforcement learning,” *IEEE Control Systems*, vol. 14, no. 1, pp. 13–24, 1994.

[10]: A. Albu-Schaffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimbock, S. Wolf, and G. Hirzinger, “Soft robotics,” *IEEE Robotics & Automation Magazine*, vol. 15, no. 3, 2008.

[11]: S. Part, “Impedance control: An approach to manipulation,” *Journal of dynamic systems, measurement, and control*, vol. 107, p. 17, 1985.

[12]: C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schäeffer, and E. Burdet, “Human-like adaptation of force and impedance in stable and unstable interactions,” *Robotics, IEEE Transactions on*, vol. 27, no. 5, pp. 918–930, 2011.

[13]: E. Burdet, R. Osu, D. Franklin, T. Milner, and M. Kawato, “The central nervous system stabilizes unstable dynamics by learning optimal impedance,” *NATURE*, vol. 414, pp. 446–449, 2001. [Online]. Available: http://dx.doi.org/10.1038/35106566

[14] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the human out of the loop: A review of bayesian optimization,” *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[15]: M. D. McKay, R. J. Beckman, and W. J. Conover, “Comparison of three methods for selecting values of input variables in the analysis of output from a computer code,” *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[16]: R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, “Bayesian optimization for learning gaits under uncertainty,” *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 1-2, pp. 5–23, 2016.

[17]: J. Nogueira, R. Martinez-Cantin, A. Bernardino, and L. Jamone, “Unscented bayesian optimization for safe robot grasping,” *arXiv preprint arXiv:1603.02038*, 2016.

[18]: F. Berkenkamp, A. Krause, and A. P. Schoellig, “Bayesian optimiza- tion with safety constraints: safe and automatic parameter tuning in robotics,” *arXiv preprint arXiv:1602.04450*, 2016.

[19]: G. Ganesh, A. Albu-Schäffer, M. Haruno, M. Kawato, and E. Burdet, “Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks,” in *Robotics and Au- tomation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2705–2711.

[20]: J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*. Prentice-hall Englewood Cliffs, NJ, 1991, vol. 199, no. 1.

[21]: A. Albu-Schäffer, C. Ott, U. Frese, and G. Hirzinger, “Cartesian impedance control of redundant robots: Recent results with the DLR- light-weight-arms,” in *IEEE Int. Conf. on Robotics and Automation*, vol. 3, 2003, pp. 3704–3709.

[22]: G. Hirzinger, N. Sporer, A. Albu-Schäffer, M. Hahnle, R. Krenn, A. Pascucci, and M. Schedl, “Dlr’s torque-controlled light weight robot iii-are we reaching the technological

limits now?" in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1710–1716.

[23]: L. Johannsmeier and S. Haddadin, "A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 41–48, 2017.

[24]: R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "An experimental comparison of bayesian optimization for bipedal locomotion," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1951–1958.

[25]: J. Snoek, "Bayesian optimization and semiparametric models with applications to assistive technology," Ph.D. dissertation, University of Toronto, 2013.

[26]: J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[27]: E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[28]: K. Swersky, J. Snoek, and R. P. Adams, "Multi-task bayesian optimiza- tion," in *Advances in neural information processing systems*, 2013, pp. 2004–2012.

[29]: R. M. Neal, "Slice sampling," *Annals of statistics*, pp. 705–741, 2003. [30] J. M. Herna´ndez-Lobato, M. A. Gelbart, M. W. Hoffman, R. P. Adams, and Z. Ghahramani, "Predictive entropy search for bayesian optimization with unknown constraints." in *ICML*, 2015, pp. 1699–1707.

Brief description of the drawings:

Fig. 1 shows a peg-in-hole skill according to a first embodiment of the invention,

Fig. 2 shows a conceptual sketch of skill dynamics according to another embodiment of the invention,

Fig. 3 shows a method for controlling actuators of an articulated robot according to a third embodiment of the invention,

Fig. 4 shows a system for controlling actuators of an articulated robot and enabling the robot executing a given task according to another embodiment of the invention,

Fig. 5 shows the system of Fig. 4 in a different level of detail, and

Fig. 6 shows a system for controlling actuators of an articulated robot and enabling the robot executing a given task according to another embodiment of the invention.

Detailed description of the drawings:

In Fig. 1 the application of the skill framework for the standard manipulation problem, i.e. the skill “peg-in-hole“ is shown. On the left half of the picture the robot 80 is located in a suitable region of interest ROI 1, with the grasped peg 3 being in contact with the surface of an object with a hole 5. The skill commands velocities resulting from a velocity based search algorithm, aiming at finding the hole 5 with according alignment and subsequently inserting the peg 3 into the hole 5. In the alignment phase a feed forward force is applied downwards-vertical (downwards in Fig. 1) and to the left. Simultaneously, the alignment movement consists of basic rotations around two horizontal axes (from left to right and into the paper plane in Fig. 1). During the insertion phase, the skill commands $\dot{x}_{d,z}$ until x_d reached a desired depth. At the same time, perpendicular Lissajous velocities $\dot{x}_{d,x}, \dot{x}_{d,y}$ are overlaid. If the peg 3 reaches the desired depth the skill was successful. The skill is defined as follows:

$S = \{\boldsymbol{x}, R, F_{ext}, \tau_{ext}\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is the position in Cartesian space, $R \in \mathbb{R}^{3x3}$ is the orientation, $F_{ext} = [f_{ext}, m_{ext}]^\top \in \mathbb{R}^6$ is the wrench of the external forces and torques and $\tau_{ext} \in \mathbb{R}^n$ is the vector of external torques where n denotes the number of joints. Objects $O = \{r, p, h\}$, where r is the robot 80, p the object or peg 3 grasped with the robot 80 and h the hole 5. $C_{pre} = \{X \in S \mid f_{ext,z} > f_{contact}, x \in U(x), g(r, p) = 1\}$ states that the robot 80 shall sense a specified contact force $f_{contact}$ and the peg 3 has to be within the region of interest ROI 1, which is defined by $U(.)$. The function $g(r,p)$ simplifies the condition of the robot r 80 having grasped the peg p 3 to a binary mapping. $C_{suc} = \{X \in S \mid x_z > x_{z,0} + d\}$ which states that the peg 3 has to be partially inserted by at least d into the hole 5 for the skill to terminate successfully. Ideally d is the depth of the hole 5.

$C_{err} = \{X \in S \mid x \notin U(x), \tau_{ext} > \tau_{max}\}$ states that the skill fails if the robot 80 leaves the ROI 1 or the external torques exceed some specified safety limit component wise. $P = \{P_t, P_l\}$ with $P_t = \{a, d, \tilde{T}, r\}$ and $P_l = \{\alpha_t, \alpha_r, \beta_t, \beta_r, F_{ff,0}, v_t, v_r\}$. a is the amplitude of the Lissajous curves, d is the desired depth, $\tilde{T}$ is the pose estimation of the hole 5 and r is the radius of the region of interest ROI 1. The controller parameters α, β and $F_{ff,0}$ are applied as in the above shown general description. v is a velocity and the indices t, r

refer to translational and rotational directions, respectively. $Q_{time} = \{t_e - t_s, f_{z,max} = max_t f_{ext,z}\}$, where t_e and t_s are the start and end time of the skill execution and $f_{ext,z}$ is the external force in z-direction. This metric aims to minimize execution time and comply to a maximum level of contact forces in the direction of insertion simultaneously.

Fig. 2 shows a conceptual sketch of skill dynamics. At execution start, all coordinates, i.e. all physical objects *O*, reside in C_{pre} of *S* for which the preconditions hold. The skill dynamics then drive the system through skills space towards the success condition C_{suc} and ultimately to the nominal result *R*. The valid skill space is surrounded by C_{err}. The abbreviation ”D.<Number>” refers to the following definitions, such that e.g. “D.4” refers to Definition 4 from the upcoming description. The skill provides desired commands and trajectories to the adaptive controller 104 together with meta parameters and other relevant quantities for executing the task. In addition, a skill contains a quality metric and parameter domain to the learning algorithm of the learning unit 103, while receiving the learned set of parameters used in execution. The adaptive controller 104 commands the robot hardware via desired joint torques and receives sensor feedback. Finally, the skill formalism makes it possible to easily connect to a high-level task planning module. The following preferred skill formalism is applied:

Definition 1 (Skill): A skill *s* is an element of the skill-space. It is defined as a tuple $(S, O, C_{pre}, C_{err}, C_{suc}, R, \chi_{cmd}, X, P, Q)$.

Definition 2 (Space): Let *S* be the Cartesian product of *I* subspaces $\zeta_i = \mathbb{R}^{m_i x n_i}$ relevant to the skill *s*, i.e.: $S = \zeta_{i=1} \times \zeta_{i=2} \times ... \times \zeta_{i=I}$ with *i={1,2,….,I} and* $I \geq 2$, wherein the subspaces ζ_i comprise a control variable and an external wrench comprising an external force and an external moment.

Definition 3 (Object): Let *o* represent a physical object with coordinates $^{\circ}X(t) \in S$ associated with it. *O* denotes the set of all objects $o \in O$ relevant to a skill *s* with $n_o = |O|$ and $n_o > 0$. Moreover, *X(t)* is defined as $X(t) = ({}^{o_1}X(t), ..., {}^{o_n}X(t))^T$. In these considerations the set *O* is not changed during skill execution, i.e. n_o = const.

Definition 4 (Task Frame): The task frame $^{\circ}R_{TF}(t)$ denotes the rotation from frame *TF* to the base frame *O*. It is assumed that $^{\circ}R_{TF}(t)$ = const.

Definition 5 (Parameters): *P* denotes the set of all skill parameters consisting of three subsets P_t, P_I and P_D. The set $P_t \subset P$ contains all parameters resulting from a priori task knowledge, experience and the intention under which the skill is executed. It is referred to P_t also as task specification. The set $P_I \subset P$ contains all other parameters that are not necessarily known beforehand and need to be learned or estimated. In particular, it contains the meta parameters $(\alpha, \beta, \gamma_\alpha, \gamma_\beta)$ for the adaptive controller 104. The third subset $P_D \subset P$ defines the valid domain for P_I i.e. it consists of intervals of values for

continuous parameters or sets of values for discrete ones. Thus, P_D determines the boundaries when learning P_I.

Conditions: There are three condition types involved in the execution of a skill: preconditions, failure conditions and success conditions. They all share the same basic definition, yet their application is substantially different. Their purpose is to define the borders and limits of the skill from start to end:

Definition 6 (Condition): Let $C \subset S$ be a closed set and $c(X(t))$ a function $c : S \rightarrow B$ where $B = \{0, 1\}$. A condition holds iff $c(X(t)) = 1$. The mapping itself depends on the specific type of condition.

Definition 7 (Precondition): C_{pre} denotes the chosen set for which the precondition defined by $c_{pre}(X(t))$ holds. The condition holds, i.e. $c_{pre}(X(t_0)) = 1$, *iff* $\forall\, x \in X : x(t_0) \in C_{pre}$. t_0 denotes the time at start of the skill execution. This means that at the beginning of skill execution the coordinates of every involved object must lie in C_{pre}.

Definition 8 (Error Condition): C_{err} denotes the chosen set for which the error condition $c_{err}(X(t))$ holds, i.e. $c_{err}(X(t)) = 1$. This follows from $\exists\, x \in X : x(t) \in C_{err}$. If the error condition is fulfilled at time t, skill execution is interrupted. No assumptions are made about how this error state is resolved since this depends on the actual skill implementation and the capabilities of high-level control and planning agency.

Definition 9 (Success Condition): C_{suc} denotes the chosen set for which the success condition defined by $c_{suc}(X(t))$ holds, i.e. $c_{suc}(X(t)) = 1$ iff $\forall\, x \in X : x(t) \in C_{suc}$. If the coordinates of all involved objects are within C_{suc} the skill execution can terminate successfully.

Definition 10 (Nominal Result): The nominal result $R \in S$ is the ideal endpoint of skill execution, i.e. the convergence point.

Although the nominal result R is the ideal goal of the skill, its execution is nonetheless considered successful if the success conditions C_{suc} holds. Nonetheless $X(t)$ converges to this point.

Definition 11 (Skill Dynamics): Let $X : [t_0,\infty] \rightarrow P$ be a general dynamic process, where t_0 denotes the start of the skill execution. The process terminates if

$(\forall\, c_{suc} \in C_{suc} : c_{suc}(X(t)) = 1) \vee (\exists\, c_{err} \in C_{err} : c_{err}(X(t)) = 1)$.

It converges to the nominal result R. This dynamic process encodes what the skill actually does depending on the input, i.e. the concrete implementation. This is a trajectory generator, a DMP, or some other algorithm calculating sensor based velocity or force commands. The finish time t_e is not necessarily known a priori. For a search skill it cannot be determined when it terminates because of the very nature of the search problem.

Definition 12 (Commands): Let $\chi_{cmd} \subset X(t)$ be the skill commands, i.e. a desired trajectory consisting of velocities and forces defined in *TF* sent to the controller.

Definition 13 (Quality Metric): *Q* denotes the set of all 2−tuples $(w, f_q(X(t))$ with $0 < w < 1$ and constraints $f_{c,i}(X(t))$. Furthermore let $q = \sum_i w_i f_{q,i}(X(t)) \forall (w_i, f_{q,i}(X(t))) \in Q$. The quality metric is a means of evaluating the performance of the skill and to impose quality constraints on it. This evaluation aims at comparing two different implementations of the same skill or two different sets of parameters *P*. The constraints are used to provide a measure of quality limits for a specific task (e.g. a specific time limit). The quality metric reflects some criterion that is derived from the overall process in which the skill is executed or given by a human supervisor.

Fig. 3 shows a method for controlling actuators of an articulated robot 80 and enabling the robot 80 executing a given task, the robot 80 comprising a first unit 101, a second unit 102, a learning unit 103, and an adaptive controller 104, the second unit 102 being connected to the first unit 101 and further to a learning unit 103 and to an adaptive controller 104, comprising the following steps:

- Providing S1 a specification of robot skills s selectable from a skill space depending on the task by a first unit 101, with a robot skill s being defined as a 2-tuple out of (S, O, C_{pre}, C_{err}, C_{suc}, R, χ_{cmd}, X, P, Q) with

S: a Cartesian product of I subspaces ζ_i : $S = \zeta_{i=1} \times \zeta_{i=2} \times ... \times \zeta_{i=I}$ with $i=\{1,2,....,I\}$ and $I \geq 2$,

O: a set of all objects,

C_{pre}: a precondition,

C_{err}: an error condition,

C_{suc}: a success condition,

R: a nominal result of ideal skill execution,

χ_{cmd} : skill commands,

X: physical coordinates,

P: skill parameters, with P consisting of three subsets P_t, P_I, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_I being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_I,

Q: a performance metric, while $Q(t)$ is denoting the actual performance of the skill carried out by the robot 80,

- an adaptive controller 104 receiving S2 skill commands χ_{cmd} from a second unit 102, wherein the second unit 102 is connected to the first unit 101 and further to a learning unit 103 and to the adaptive controller 104 and wherein the skill commands χ_{cmd}

comprise the skill parameters P_l,

- controlling S3 the actuators of the robot 80 by the adaptive controller 104 and based on the skill commands χ_{cmd}, wherein the actual status of the robot 80 is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller 104 and to the second unit 102,
- determining S4 by the second unit 102 and based on the actual status, the performance *Q(t)* of the skill carried out by the robot 80,
- the learning unit 103 receiving S5 P_D and *Q(t)* from the second unit 102, and
    - determining S6 updated skill parameters $P_l(t)$ and providing $P_l(t)$ to the second unit 102 and replacing hitherto existing skill parameters P_l, wherein the subspaces ζ_i comprise a control variable and an external wrench comprising in particular an external force and an external moment.

Fig. 4 and 5 show each a system for controlling actuators of an articulated robot 80 and enabling the robot 80 executing a given task in different levels of detail. The system each comprising:

- a first unit 101 providing a specification of robot skills *s* selectable from a skill space depending on the task, with a robot skill *s* being defined as a tuple out of (S, O, C_{pre}, C_{err}, C_{suc}, R, χ_{cmd}, X, P, Q) with
    - S: a Cartesian product of I subspaces ζ_i : $S=\zeta_{i=1}\times\zeta_{i=2}\times...\times\zeta_{i=I}$ with $i=\{1,2,....,I\}$ and $I\geq 2$,
    - O: a set of all physical objects,
    - C_{pre}: a precondition,
    - C_{err}: an error condition,
    - C_{suc}: a success condition,
    - R: nominal result of ideal skill execution,
    - χ_{cmd} : skill commands,
    - X: physical coordinates,
    - P: skill parameters, with P consisting of three subsets P_t, P_l, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_l being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_l,
    - Q: a performance metric, while $Q(t)$ is denoting the actual performance of the skill carried out by the robot 80,
- a second unit 102, wherein the second unit 102 is connected to the first unit 101 and further to a learning unit 103 and to an adaptive controller 104,wherein the adaptive controller 104 receives skill commands χ_{cmd}, wherein the skill commands χ_{cmd} comprise

the skill parameters P_l, wherein based on the skill commands χ_{cmd} the controller 104 controls the actuators of the robot 80, wherein the actual status $X(t)$ of the robot 80 is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller 104 and to the second unit 102, wherein based on the actual status $X(t)$, the second unit 102 determines the performance *Q(t)* of the skill carried out by the robot 80, and wherein the learning unit 103 receives P_D, and *Q(t)* from the second unit 102, determines updated skill parameters $P_l(t)$ and provides $P_l(t)$ to the second unit 102 to replace hitherto existing skill parameters P_l, wherein the subspaces ζ_i comprise a control variable and an external wrench comprising an external force and an external moment. The skill commands χ_{cmd} comprise the skill parameters *P* within the desired force F_d, and F_d, which is dependent on *P* via $F_d = f_F(X, P)$, wherein P_l is one of the three subsets of *P.* Likewise the desired velocity $\dot{x}_d$ is dependent on P and therefore also on P_l with $\dot{x}_d = f_X(X, P)$. The parameter P_t is herein received from a database of a planning and skill surveillance unit, symbolized by a stacked cylinder.

Fig. 6 shows a system for controlling actuators of an articulated robot 80 and enabling the robot 80 executing a given task, comprising:

- a first unit 101 providing a specification of robot skills *s* selectable from a skill space depending on the task, with a robot skill *s* being defined as a tuple from (*S, O, C_{pre}, C_{err}, C_{suc}, R,* χ_{cmd} *, X, P, Q*). The expressions of this tuple are defined as follows:

  S: a Cartesian product of *I* subspaces ζ_i: $S = \zeta_{i=1} \times \zeta_{i=2} \times ... \times \zeta_{i=I}$ with *i={1,2,….,I} and* $I \geq 2$,

  O: a set of all physical objects,

  C_{pre}: a precondition,

  C_{err}: an error condition,

  C_{suc}: a success condition,

  R: a nominal result of ideal skill execution,

  χ_{cmd} : skill commands,

  X: physical coordinates,

  P: skill parameters, with *P* consisting of three subsets P_t, P_l, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_l being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_l,

  Q: a performance metric, wherein *Q(t)* is denoting the actual performance of the skill carried out by the robot 80,

- a second unit 102, wherein the second unit 102 is connected to the first unit 101

and further to a learning unit 103 and to an adaptive controller 104,
wherein the adaptive controller 104 receives skill commands χ_{cmd},
wherein the skill commands χ_{cmd} comprise the skill parameters P_I,
wherein based on the skill commands χ_{cmd} the controller 104 controls the actuators of the robot 80 via a control signal τ_d, wherein the actual status $X(t)$ of the robot 80 is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller 104 and to the second unit 102, wherein based on the actual status $X(t)$, the second unit 102 determines the performance *Q(t)* of the skill carried out by the robot 80, and wherein the learning unit 103 receives P_D, and *Q(t)* from the second unit 102, determines updated skill parameters $P_I(t)$ and provides $P_I(t)$ to the second unit 102 to replace hitherto existing skill parameters P_I.

Reference-list

1	Region of interest ROI
3	Peg
5	Hole
80	Robot
101	First unit
102	Second unit
103	Learning unit
104	Adaptive controller
S1	Providing
S2	Receiving
S3	Controlling
S4	Determining
S5	Receiving
S6	Determining

Claims

1. System for controlling actuators of an articulated robot (80) and enabling the robot (80) executing a given task, comprising:

- a first unit (101) providing a specification of robot skills *s* selectable from a skill space depending on the task, with a robot skill *s* being defined as a tuple (S, O, C_{pre}, C_{err}, C_{suc}, R, χ_{cmd}, X, P, Q) with

S: a Cartesian product of I subspaces ζ_i : $S=\zeta_{i=1}\times\zeta_{i=2}\times...\times\zeta_{i=I}$ with $i=\{1,2,....,I\}$ and $I\geq 2$,

O: a set of physical objects,

C_{pre}: a precondition,

C_{err}: an error condition,

C_{suc}: a success condition,

R: nominal result of ideal skill execution,

χ_{cmd} : skill commands,

X: physical coordinates,

P: skill parameters, with P consisting of three subsets P_t, P_I, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_I being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_I,

Q: a performance metric, wherein $Q(t)$ is denoting the actual performance of the skill carried out by the robot (80),

- a second unit (102), wherein the second unit (102) is connected to the first unit (101) and further to a learning unit (103) and to an adaptive controller (104),
wherein the adaptive controller (104) receives skill commands χ_{cmd},
wherein the skill commands χ_{cmd} comprise the skill parameters P_I,
wherein based on the skill commands χ_{cmd} the controller (104) controls the actuators of the robot (80),
wherein the actual status of the robot (80) is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller (104) and to the second unit (102),
wherein based on the actual status, the second unit (102) determines the performance value $Q(t)$ of the skill carried out by the robot (80), and
wherein the learning unit (103) receives P_D, and $Q(t)$ from the second unit (102), determines updated skill parameters $P_I(t)$ and provides $P_I(t)$ to the second unit

(102) to replace hitherto existing skill parameters P_l.

2. System according to claim 1,
wherein the adaptive controller (104) adapts feed forward wrench and stiffness via $\delta F_{ff} = F_{ff}(t) - F_{ff}(t-T)$.

3. System according to one of the claims 1 or 2,
wherein the learning unit (103) carries out a Bayesian and/or a HiREPS optimization / learning.

4. System according to one of claims 1 to 3,
wherein the system comprises a data interface with a data network, and wherein the system is designed and setup to download system-programs for setting up and controlling the system from the data network.

5. System according to one of the claims 1 to 4,
wherein the system is designed and setup to download parameters for the system-programs from the data network.

6. System according to one of the claims 1 to 5,
wherein the system is designed and setup to enter parameters for the system-programs via a local input-interface and/or via a teach-in-process, with the robot (80) being manually guided.

7. System according to one of the claims 1 to 6,
wherein the system is designed and setup such that downloading system-programs and/or respective parameters from the data network is controlled by a remote station, and wherein the remote station being part of the data network.

8. System according to one of the claims 1 to 7,
wherein the system is designed and setup such that system-programs and/or respective parameters locally available at the system are sent to one or more participants of the data network based on a respective request received from the data network.

9. System according to one of the claims 1 to 8,
wherein the system is designed and setup such that system-programs with

respective parameters available locally at the system can be started from a remote station, and wherein the remote station is part of the data network.

10. System according to one of the claims 1 to 9,
wherein the system is designed and setup such that the remote station and/or the local input-interface comprises a human-machine-interface HMI designed and setup for entry of system-programs and respective parameters and/or for selecting system-programs and respective parameters from a multitude system-programs and respective parameters.

11. System according to claim 10,
wherein the human-machine-interface HMI is designed and setup such that entries are possible via „drag-and-drop“-entry on a touchscreen, a guided dialogue, a keyboard, a computer-mouse, a haptic interface, a virtual-reality-interface, an augmented reality interface, an acoustic interface, via a body tracking interface, based on electromyographic data, based on elektroenzephalographic data, via a neuronal interface, or a combination thereof.

12. System according to claim 10 or 11,
wherein the human-machine-interface HMI is designed and setup to deliver auditive, visual, haptic, olfactoric, tactile, or electrical feedback or a combination thereof.

13. Robot (80) with a system according to one of claims 1 to 12.

14. Method for controlling actuators of an articulated robot (80) and enabling the robot (80) executing a given task, the robot (80) comprising a first unit (101), a second unit (102), a learning unit (103), and an adaptive controller (104), the second unit (102) being connected to the first unit (101) and further to a learning unit (103) and to an adaptive controller (104), comprising following steps:
- providing (S1) a specification of robot skills s selectable from a skill space depending on the task by a first unit (101), with a robot skill s being defined as a tuple ($S, O, C_{pre}, C_{err}, C_{suc}, R, \chi_{cmd}, X, P, Q$) with

S: a Cartesian product of I subspaces ζ_i : $S=\zeta_{i=1}\times\zeta_{i=2}\times...\times\zeta_{i=I}$

15. with $i=\{1,2,....,I\}$ and $I\geq 2$,

O: a set of physical objects,

C_{pre}: a precondition,
C_{err}: an error condition,
C_{suc}: a success condition,
R: nominal result of ideal skill execution,
χ_{cmd} : skill commands,
X: physical coordinates,
P: skill parameters, with P consisting of three subsets P_t, P_l, P_D, with P_t being the parameters resulting from a priori knowledge of the task, P_l being the parameters not known initially and need to be learned and/or estimated during execution of the task, and P_D being constraints of parameters P_l,
Q: a performance metric, while $Q(t)$ is denoting the actual performance of the skill carried out by the robot (80),

- an adaptive controller (104) receiving (S2) skill commands χ_{cmd} from a second unit (102),

wherein the second unit (102) is connected to the first unit (101) and further to a learning unit (103) and to the adaptive controller (104) and wherein the skill commands χ_{cmd} comprise the skill parameters P_l,

- controlling (S3) the actuators of the robot (80) by the controller (104) and based on the skill commands χ_{cmd}, wherein the actual status of the robot (80) is sensed by respective sensors and/or estimated by respective estimators and fed back to the controller (104) and to the second unit (102),
- determining (S4) by the second unit (102) and based on the actual status, the performance value $Q(t)$ of the skill carried out by the robot (80),
- the learning unit (103) receiving (S5) P_D and $Q(t)$ from the second unit (102), and
- determining (S6) updated skill parameters $P_l(t)$ and providing $P_l(t)$ to the second unit (102) and replacing hitherto existing skill parameters P_l.

16. Computer system with a data processing unit, wherein the data processing unit is designed and set up to carry out a method according to one of the preceding claims.

17. Digital data storage with electronically readable control signals, wherein the control signals can coaction with a programmable computer system, so that a method according to one of the preceding claims is carried out.

18. Computer program product comprising a program code stored in a machine-readable medium for executing a method according to one of the preceding claims,

if the program code is executed on a computer system.

19. Computer program with program codes for executing a method according to one of the preceding claims, if the computer program runs on a computer system.

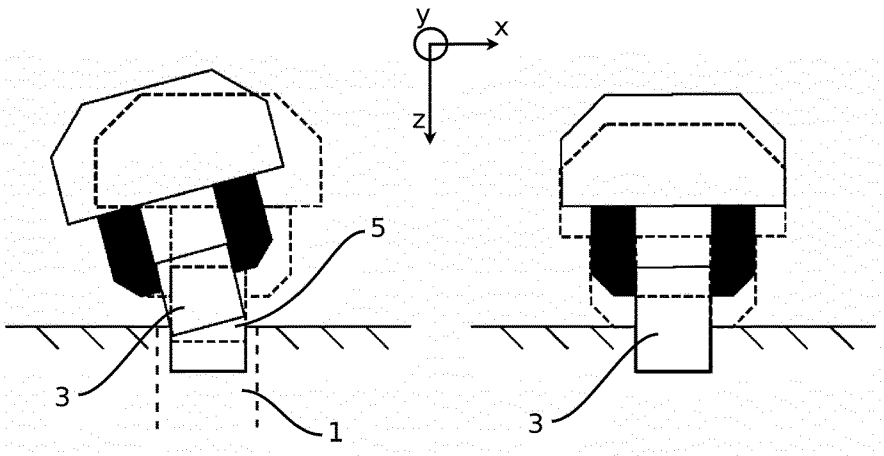

y
x
z
5
3
1
3

Fig. 1

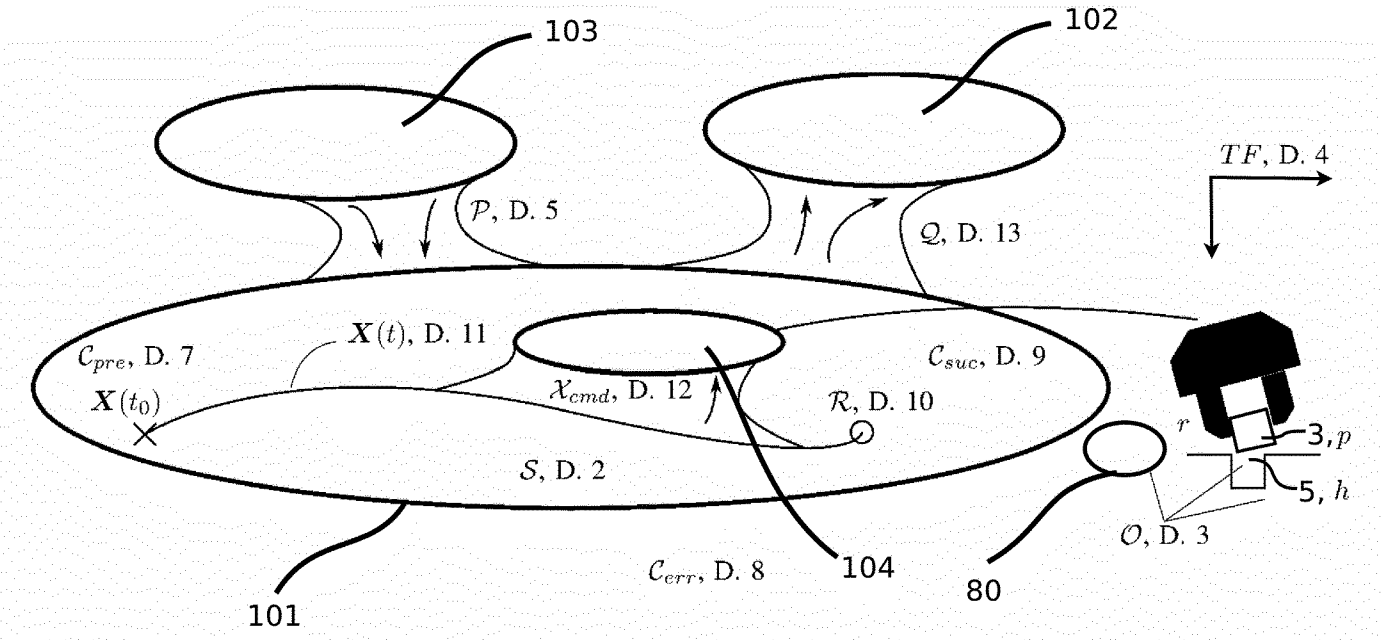

103
102
TF, D. 4
$\mathcal{P}$, D. 5
$\mathcal{Q}$, D. 13
$\boldsymbol{X}(t)$, D. 11
$\mathcal{C}_{pre}$, D. 7
$\boldsymbol{X}(t_0)$
$\mathcal{X}_{cmd}$, D. 12
$\mathcal{C}_{suc}$, D. 9
$\mathcal{R}$, D. 10
$\mathcal{S}$, D. 2
r
3, p
5, h
$\mathcal{O}$, D. 3
$\mathcal{C}_{err}$, D. 8
104
80
101

Fig. 2

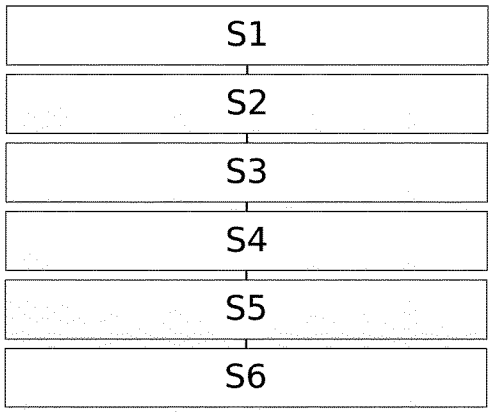

S1
S2
S3
S4
S5
S6

Fig. 3

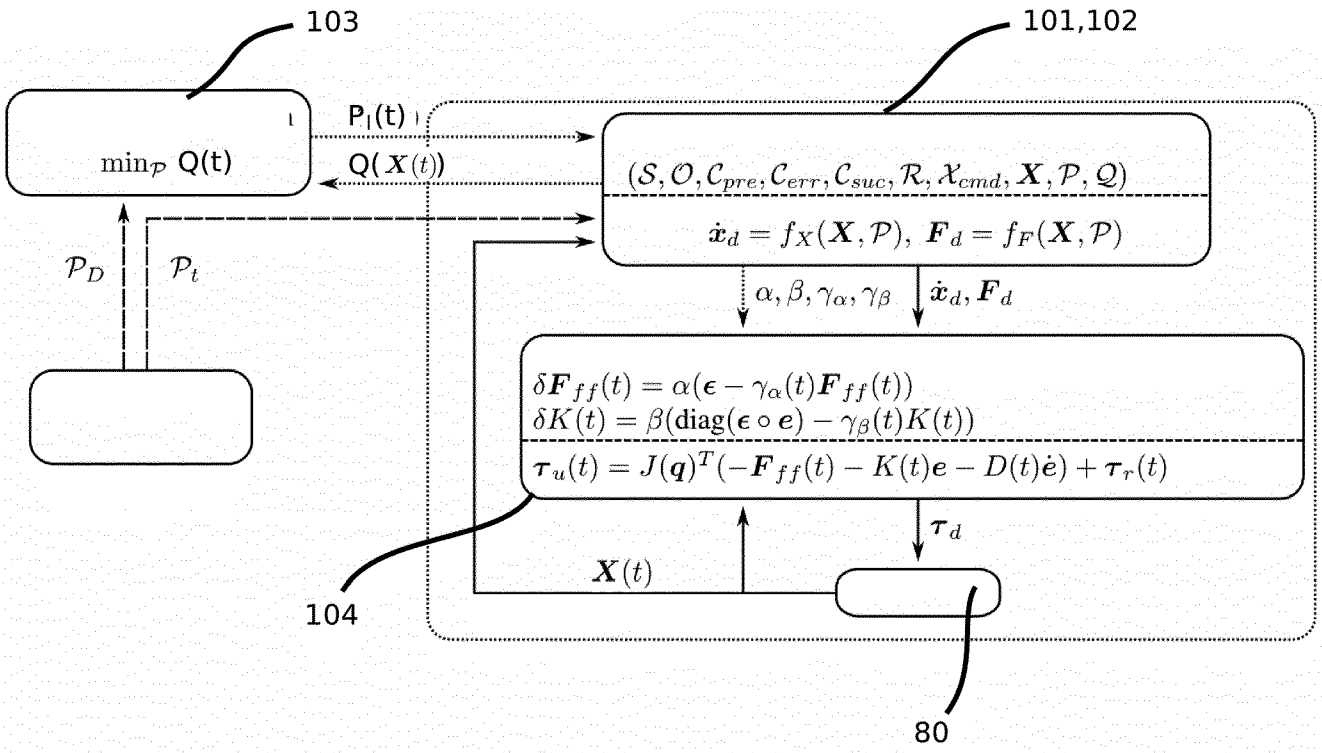

103
101,102
$\min_{\mathcal{P}}$ Q(t)
$P_l(t)$
$Q(\boldsymbol{X}(t))$
$(\mathcal{S}, \mathcal{O}, \mathcal{C}_{pre}, \mathcal{C}_{err}, \mathcal{C}_{suc}, \mathcal{R}, \mathcal{X}_{cmd}, \boldsymbol{X}, \mathcal{P}, \mathcal{Q})$
$\dot{\boldsymbol{x}}_d = f_X(\boldsymbol{X}, \mathcal{P}),\ \boldsymbol{F}_d = f_F(\boldsymbol{X}, \mathcal{P})$
$\mathcal{P}_D$
$\mathcal{P}_t$
$\alpha, \beta, \gamma_\alpha, \gamma_\beta$
$\dot{\boldsymbol{x}}_d, \boldsymbol{F}_d$
$\delta \boldsymbol{F}_{ff}(t) = \alpha(\boldsymbol{\epsilon} - \gamma_\alpha(t)\boldsymbol{F}_{ff}(t))$
$\delta K(t) = \beta(\text{diag}(\boldsymbol{\epsilon} \circ \boldsymbol{e}) - \gamma_\beta(t)K(t))$
$\boldsymbol{\tau}_u(t) = J(\boldsymbol{q})^T(-\boldsymbol{F}_{ff}(t) - K(t)\boldsymbol{e} - D(t)\dot{\boldsymbol{e}}) + \boldsymbol{\tau}_r(t)$
$\boldsymbol{\tau}_d$
$\boldsymbol{X}(t)$
104
80

Fig. 4

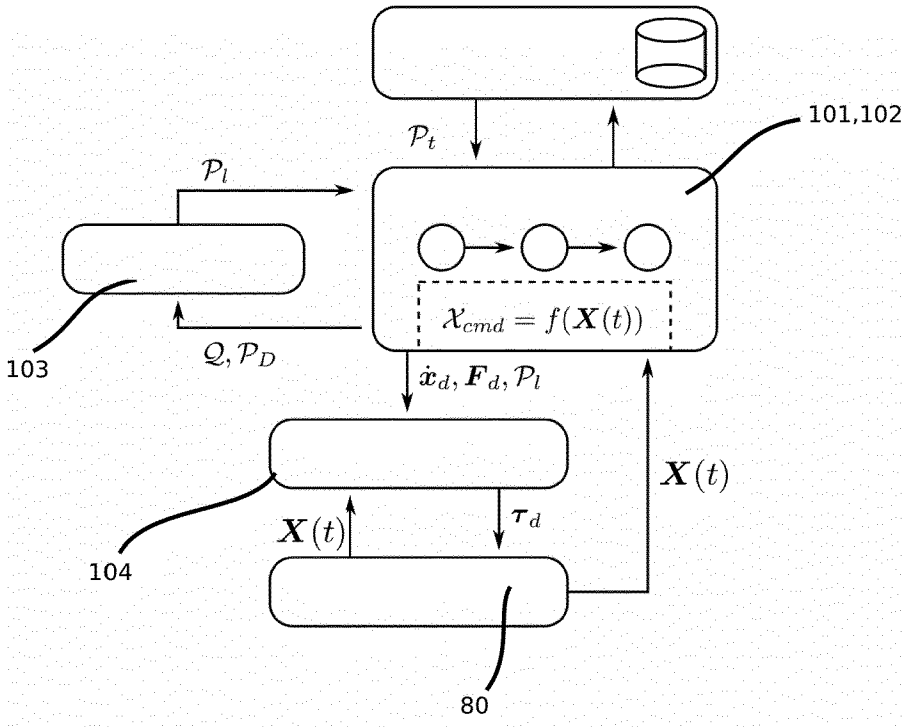

$\mathcal{P}_t$
$\mathcal{P}_l$
101,102
$\mathcal{X}_{cmd} = f(\boldsymbol{X}(t))$
103
$\mathcal{Q}, \mathcal{P}_D$
$\dot{\boldsymbol{x}}_d, \boldsymbol{F}_d, \mathcal{P}_l$
$\boldsymbol{X}(t)$
$\boldsymbol{X}(t)$
$\boldsymbol{\tau}_d$
104
80

Fig. 5

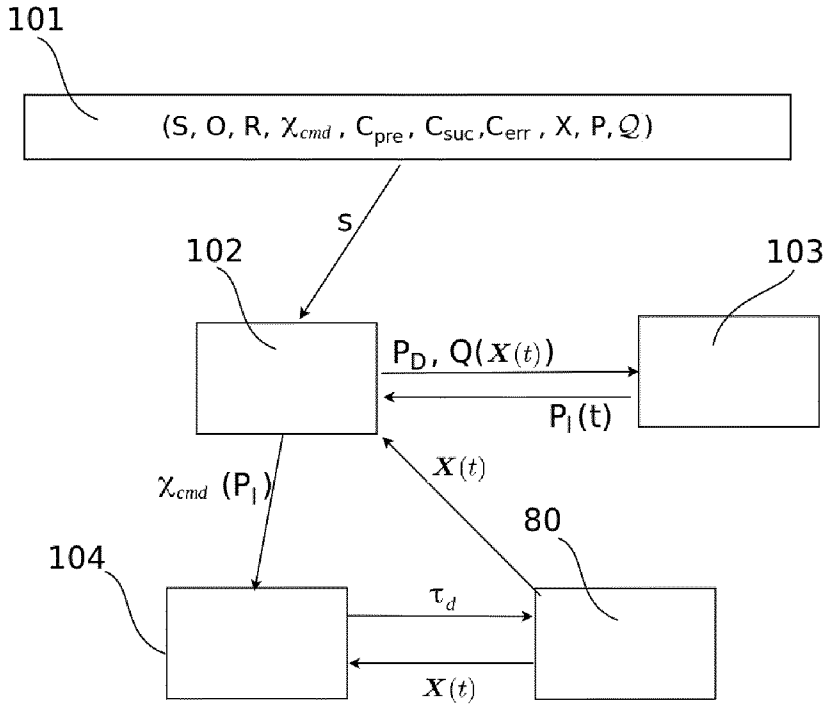

101
(S, O, R, χ_{cmd} , C_{pre} , C_{suc} ,C_{err} , X, P, $\mathcal{Q}$)
S
102
103
P_D, $Q(\boldsymbol{X}(t))$
$P_l(t)$
χ_{cmd} (P_l)
$\boldsymbol{X}(t)$
80
104
τ_d
$\boldsymbol{X}(t)$

Fig. 6

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2018/064059

A. CLASSIFICATION OF SUBJECT MATTER
INV. B25J9/16 G05B13/02
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G05B B25J

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CHENGUANG YANG ET AL: "Human-Like Adaptation of Force and Impedance in Stable and Unstable Interactions", IEEE TRANSACTIONS ON ROBOTICS, IEEE SERVICE CENTER, PISCATAWAY, NJ, US, vol. 27, no. 5, 1 October 2011 (2011-10-01), pages 918-930, XP011361213, ISSN: 1552-3098, DOI: 10.1109/TRO.2011.2158251 cited in the application abstract page 919 - page 921 page 924 - page 927 figures 1(b),7,8 ----- -/--	1-18

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 September 2018	05/10/2018

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	De Santis, Agostino

1

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	Mikkel Rath Pedersen ET AL: "On the Integration of Hardware-Abstracted Robot Skills for use in Industrial Scenarios", Second International Workshop on Cognitive Robotics Systems: Replicating Human Actions and Activities, 3 November 2013 (2013-11-03), XP055507180, Tokyo Retrieved from the Internet: URL:http://renaud-detry.net/events/crs2013/papers/Pedersen.pdf [retrieved on 2018-09-14] abstract page 1 - page 4 figure 2 -----	1-18