Robot Motion Control During Abrupt Switchings Between Manipulation Primitives

Torsten Kröger and Bernd Finkemeyer

Abstract— **Robot manipulation tasks can be subdivided into Manipulation Primitives — a promising concept that is being studied since three decades. Although the basic theory shows great potential to let robots perform useful manipulation tasks in human environments, we can hardly find successful implementations. To realize highly reactive sensor-guided and sensor-guarded robot behaviors, Manipulation Primitives consist of three fundamental parts: (1) a controller reference frame, (2) a set of feedback controllers and set-points, and (3) a termination condition. This paper describes, how stable and continuous robot motion control behaviors can be achieved during abrupt transitions of Manipulation Primitives. The core idea is based on an On-Line Trajectory Generation algorithm that generates set-points for multiple lower-level controllers from arbitrary states of motion within one control cycle (typically less than one millisecond). This features gives robot manipulation control systems the ability to very rapidly switch controllers, and to permit non-zero controller accelerations at start-up. Simulation and real-world experimental results are shown to underline the relevance for robot manipulation tasks.**

I. Introduction

Developing manipulation capabilities and collision-preventing free-space motions are keys to let mobile manipulators accomplish useful tasks in human environments. Realizing such systems is difficult because manipulation tasks in dynamic environments require the integration of sensors (force/torque, tactile, vision, distance, etc.) in high-level planning systems as well as in inner robot control loops. The approach of separating robot manipulation tasks into highly reactive sensor-based compliant Manipulation Primitives (MP) has been investigated by several research groups and seems very promising to let robots perform *useful* tasks in real-world environments.

To realize an MP-based robot motion control system, force/torque control, visual servo control, distance control, manipulator dynamics, and control software architecture issues have to be solved and integrated in one system. A single primitive may involve a number exteroceptive sensors and corresponding feedback controllers in the most inner motion control loops to simultaneously allow force/torque control, visual servo control, distance control, and trajectory-following control within a given robot task. In the control cycle, within which the termination condition of a single primitive becomes true, the next MP is immediately chosen based on the system state and currently perceived sensor signals. This new MP is executed from the very next control cycle on, such that the control system has to be switched from one MP to the next one within one control cycle, that is, all controllers may be fed with different set-points and may be used w.r.t. a different reference frame.

This paper briefly reviews the concept of Manipulation Primitives. The core of it describes an approach based on On-Line Trajectory Generation to guarantee *continuous* robot motions despite of abrupt transitions from one MP to another. Finally, simulation and real-world experimental results are shown to illustrate, how the proposed concepts behave in practice.

T. Kröger is with the Artificial Intelligence Laboratory at Stanford University, Stanford, CA 94305, USA, tkr@stanford.edu. B. Finkemeyer is with KUKA Laboratories GmbH, Zugspitzstraße 140, D-86165 Augsburg, Germany, bernd.finkemeyer@kuka.com.

II. Related Works

A. Manipulation Primitives

The concept of Manipulation Primitives is based on the works of Mason [1], De Schutter *et al.* [2], [3], and Bruyninckx *et al.* [4], [5]. Concrete works on the regarded concept were published by Mosemann *et al.* [6], [7], who introduced the concept of Skill Primitives for robot assembly tasks. The authors of this paper [8]–[11] focused on control aspects in this context and suggested a formal definition of the term *Manipulation Primitive*. Milighetti *et al.* [12]–[14] extended this definition with further elements that enable the control system to make a certain fuzzy- and/or probability-based switching decision during task execution. Thomas *et al.* [15], [16] applied the MP concept to robot assembly tasks, and Maaß *et al.* [17], [18], Dietrich *et al.* [19], and Reisinger [20] transferred it to parallel kinematic machines. A very recent implementation was presented by Zieliński *et al.* [21].

B. On-Line Motion Generation

The control approach to guarantee continuous robot motions during abrupt switchings of Manipulation Primitives is based on the On-Line Trajectory Generation (OTG) framework of [22], [23]. The works mostly related to this concept are [24]–[30]. Macfarlane *et al.* [24] present a jerk-bounded, near-time-optimal trajectory planner that uses quintic splines, which are also computed on-line but only for *one-degree-of-freedom systems*. In [25], Cao *et al.* use rectangular jerk pulses to compute trajectories, but initial accelerations different from zero cannot be applied. Compared to the multi-degree-of-freedom approach used in this paper, the latter method has been developed for *one-dimensional problems* only. Broquère *et al.* [26], [27] published a method that uses an On-Line Trajectory Generator for an arbitrary number of independently acting degrees of freedoms. The approach is very similar to the one of Liu [28] and is based on the classic seven-segment acceleration profile [31]. With regard

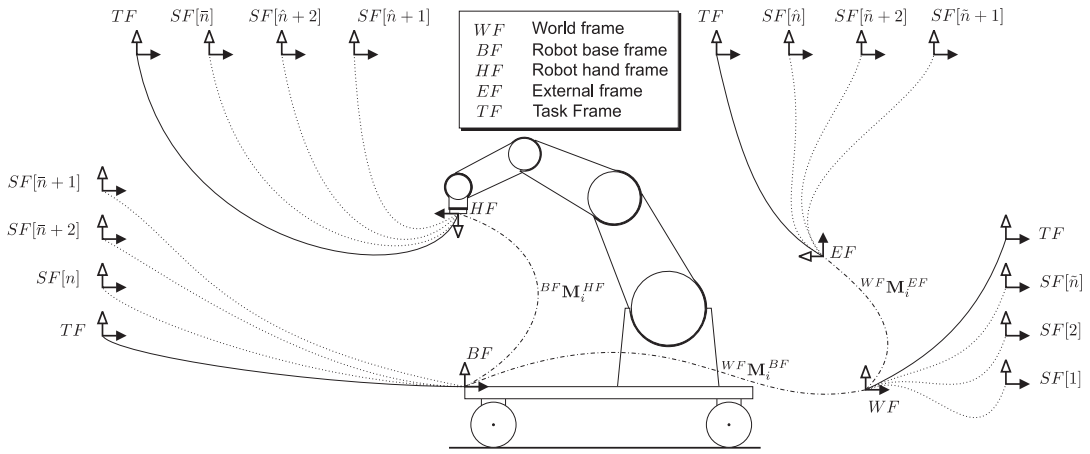

Fig. 1. Frame assignments according to [10].

to [23], it is a Type V OTG approach designed for handling several degrees of freedom individually. Real-world results are presented and described at [32].

A disadvantage of [24], [25], [28] is that they cannot cope with initial acceleration values unequal to zero. A further, very recent work of Haschke *et al.* [29] presents an on-line trajectory planner in the very same sense as [22], [23] do. The proposed algorithm generates jerk-limited trajectories from arbitrary states of motion, but it suffers from numerical stability problems, that is, it may happen, that no jerk-limited trajectory can be calculated. In such a case, a second-order trajectory with infinite jerks is calculated. Furthermore, the algorithm only allows target velocities of zero. Ahn *et al.* [30] proposed a work for the on-line calculation of one-dimensional motion trajectories for any given state of motion *and* with arbitrary target states of motion, that is, with target velocities and target accelerations unequal to zero. Sixth-order polynomials are used to represent the trajectory, which is called arbitrary states polynomial-like trajectory (ASPOT). The major drawback of this work is that no kinematic motion constraints, such as maximum velocity, acceleration, and jerk values, can be specified.

III. Manipulation Primitive Framework

The robot motion control and robot motion specification framework descried in the following was proposed by Finkemeyer [8] in 2004. A summary can be found in [11].

An MP at time instant T_i is formally defined as

$$\mathcal{MP}_i := \{\mathcal{HM}_i, \tau_i, \lambda_i\} \tag{1}$$

where $\mathcal{HM}_i$ defines a hybrid motion, τ_i contains tool commands, and the termination condition λ_i determines the end of execution of a single MP. In this brief review, we only focus on $\mathcal{HM}_i$ and λ_i.

A. Hybrid Move $\mathcal{HM}_i$

$\mathcal{HM}_i$ defines a hybrid move in the sense of the Task Frame Formalism [5], which was extended by [8] in order to take multiple sensors into consideration. For this purpose, the hybrid motion command $\mathcal{HM}_i$ is composed of a task frame specification $\mathcal{TF}_i$ and of a set of set-points $\mathcal{D}_i$:

$$\mathcal{HM}_i := \{\mathcal{TF}_i, \mathcal{D}_i\} \ . \tag{2}$$

Figure 1 illustrates the frame assignments in correspondence to [10]. n sensors, whose signals can be used in the feedback loops of the robot control scheme, may be mounted w.r.t. different coordinate frames. In correspondence to Fig. 1, the task frame $\mathcal{TF}_i$ at instant T_i is defined as

$$\mathcal{TF}_i := \{\vec{\theta}_i, RF_i, ANC_i, FFC_i\} \tag{3}$$

with

$$\vec{\theta}_i = \left({}_x\theta_i, {}_y\theta_i, {}_z\theta_i, {}_{\textcircled{x}}\theta_i, {}_{\textcircled{y}}\theta_i, {}_{\textcircled{z}}\theta_i\right)^T \in \mathbb{R}^6 \tag{4}$$

$$RF_i, ANC_i \in \{HF, WF, BF, EF\} \tag{5}$$

$$FFC_i \in \{WF, BF, EF\} \ . \tag{6}$$

The vector $\vec{\theta}_i$ specifies the position $({}_x\theta_i, {}_y\theta_i, {}_z\theta_i)^T$ and orientation $({}_{\textcircled{x}}\theta_i, {}_{\textcircled{y}}\theta_i, {}_{\textcircled{z}}\theta_i)^T$ of the task frame w.r.t. the reference frame RF_i, which can either be the robot's hand frame HF, the world frame WF, the robot base frame BF, or an external frame EF (e.g., a second mobile manipulator). The frame ANC_i serves as anchor and rigidly connects the Task Frame to another frame. The transformation ${}^{ANC}\mathbf{T}_{TF}$ is constant during the entire execution of one single MP. If compliant motions are to be executed w.r.t. an external moving coordinate system, the FFC_i frame (i.e., its pose, velocity, and acceleration) enables the internal computation of a feed-forward compensation (FFC_i) signal, such that sensor-based motion commands can be executed in dynamic systems in the same way as in static ones [16].

The set of set-points $\mathcal{D}_i$ at T_i is defined as[1]

$$\mathcal{D}_i \ := \ \left\{ {}^l_k D_i^c \ \forall \ (k, l, c) \in \left(\mathcal{K} \times \mathcal{L} \times \mathcal{C}\right)\right\} \tag{7}$$

[1]This representation of $\mathcal{D}_i$ was simplified in order to keep the summary about MPs short; for a full description, please refer to [10].

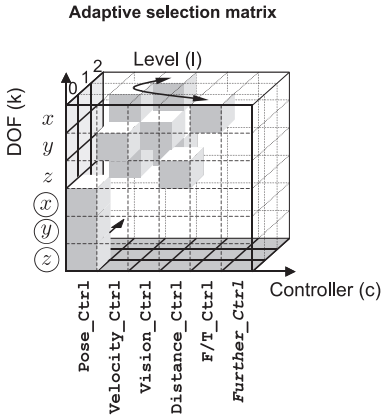

Fig. 2. The *adaptive selection matrix* is spanned by the elements of $\mathcal{K}$, $\mathcal{L}$, and $\mathcal{C}$ (cf. [10]).

with

$$\mathcal{K} := \{ x,\, y,\, z, ⓧ, ⓨ, ⓩ \} \tag{8}$$

$$\mathcal{L} := \{0, \ldots, (m-1)\} \tag{9}$$

$$\mathcal{C} := \big\{\texttt{Pose\_Ctrl}, \texttt{Velocity\_Ctrl}, \texttt{Distance\_Ctrl}, \texttt{F/T\_Ctrl}, \texttt{Vision\_Ctrl}, \ldots\big\} \tag{10}$$

$$|\mathcal{L}| \;=\; |\mathcal{C}| \;. \tag{11}$$

In eqns. (7)–(11), $\mathcal{K}$ represents the set of degrees of freedom, $\mathcal{L}$ the set of all control levels, and $\mathcal{C}$ the set of available controllers. These three parameters span the discrete space of the adaptive selection matrix as shown in Fig. 2 (cf. [10]). Each control module $c \in \mathcal{C}$ generates an availability flag vector

$$\vec{f}_i^{\,c} \;=\; \left({}_x f_i^{\,c},\, {}_y f_i^{\,c},\, {}_z f_i^{\,c},\, {}_ⓧ f_i^{\,c},\, {}_ⓨ f_i^{\,c},\, {}_ⓩ f_i^{\,c}\right)^T \;\in\; \mathbb{B}^6 \;, \tag{12}$$

where $\mathbb{B}$ is the set of Boolean numbers. Depending on this vector, the resulting hybrid switched-control system always uses the control submodule at the lowest available level for each degree of freedom. If a control submodule is not available, the controller of the next level will be used. At the highest level, a safe backup controller is provided.

B. Termination Condition λ_i

During the execution of an MP, the task parameters $\mathcal{MP}_i$ remain constant. The termination condition λ_i defines a *system state*, after whose attainment the currently executed MP becomes terminated. It is a Boolean expression that is defined as:

$$\lambda_i := \mathcal{S} \;\longrightarrow\; \{true,\, false\} \;. \tag{13}$$

$\mathcal{S}$ is the set of all available exteroceptive and proprioceptive sensor signals and their corresponding filter functions (cf. Fig. 1).

If the termination condition becomes true at an instant T_j with $j \in \mathbb{Z}$, new task parameters $\mathcal{MP}_{j+1}$ are applied in the following control cycle. This sensor-dependent change of parameters (cf. eqns. (2)–(11)) may also induce controller switchings in system depicted in Fig. 3.

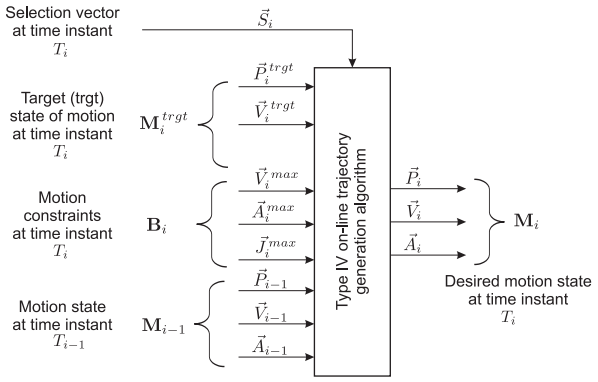

Fig. 4. Input and output values of the Type IV OTG algorithm (cf. [22], [23]).

C. Control Architecture

The realization of a programming interface based on MPs leads to a hybrid switched-system. A sample architecture is shown in Fig. 3. The robotic system at the bottom left corner is equipped with position sensors (encoders or resolvers) and is connected to a set of servo drive controllers, one per degree of freedom, whose control structure is depicted in a simplified manner [33]. The third element of Fig. 3 is a real-time computing platform with a respective interface to servo drive controllers, for example, *Sercos III* [34], *EtherCAT* [35], or even an analog interface with a motion control board. The joint space control concept is also a very classical one. Details on these kinds of control schemes can be found in abundance in the literature [36]–[38].

The top part of the diagram shows controllers in actuator space and in task space. Both are separated by the dashed line. The hybrid switched-system showed at the top of the figure generates new states of motion

$$\begin{aligned} \mathbf{M}_i &= \left(\vec{P}_i,\, \vec{V}_i,\, \vec{A}_i\right) \\ &= \left({}_x\vec{M}_i,\, {}_y\vec{M}_i,\, {}_z\vec{M}_i,\, {}_ⓧ\vec{M}_i,\, {}_ⓨ\vec{M}_i,\, {}_ⓩ\vec{M}_i,\, \right)^T \end{aligned} \tag{14}$$

within each control cycle T_i. Based on the parameters of $\mathcal{TF}_i$ (eqns. (2)–(6)), $\mathbf{M}_i$ is transformed into actuator space and applied there. Which of the control signals of the m control for which degree of freedom contributes to $\mathbf{M}_i$ depends on the adaptive selection matrix (eqns. (7)–(11)), which in turn depends on the current MP parameters.

D. On-Line Trajectory Generation

The used class of OTG algorithms can be considered as a state-feedback pose controller using the current state of motion $\mathbf{M}_{i-1}$ for command variable generation [22], [23]. The input and output values of the Type IV algorithm are shown in Fig. 4. The Type IV algorithm generates jerk-limited motion trajectories and allows to specify a target velocity vector $\vec{V}_i^{\,trgt}$ that is reached in the target pose $\vec{P}_i^{\,trgt}$. Depending on $\mathbf{M}_{i-1}$ (cf. Figs. 3 and 4), an acceleration profile is selected from a finite set of profiles. Based on this profile, a system of nonlinear equations can be set up,

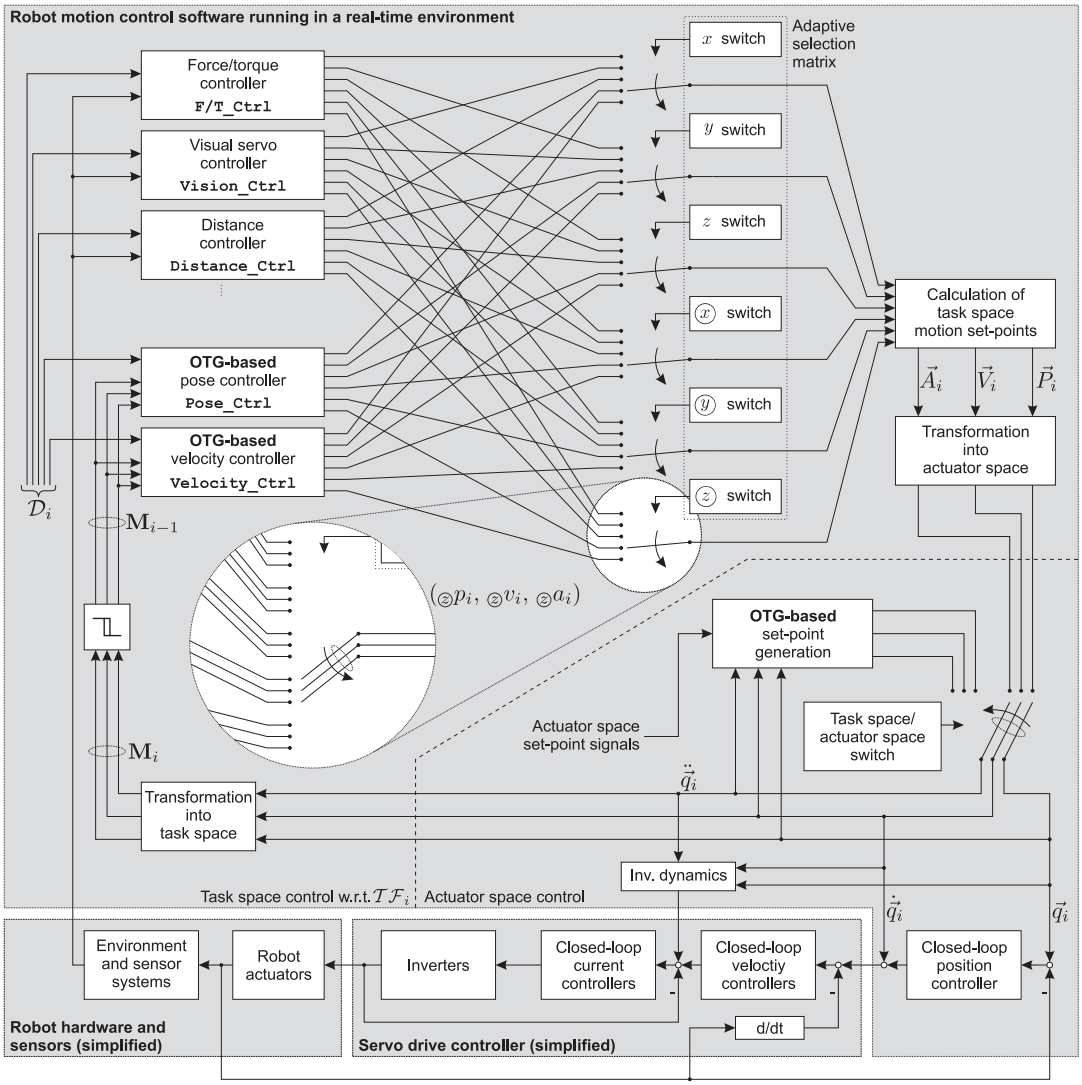

Fig. 3. Schematic robot motion control architecture for Manipulation Primitives based on a hybrid switched-system (cf. [23]).

whose solution contains all trajectory parameters to transfer the system from its current state of motion to a desired target state of motion while considering the given kinematic set motion constraints $\mathbf{B}_i$ in the shortest possible time.

The target state of motion in task space at instant T_i w.r.t. $\mathcal{TF}_i$ is represented by the matrix

$$\mathbf{M}_i^{trgt} = \left(\vec{P}_i^{\,trgt}, \vec{V}_i^{\,trgt}, \vec{0}\right) \tag{15}$$

Furthermore, the kinematic motion constraints are analogously defined as

$$\mathbf{B}_i = \left(\vec{V}_i^{\,max}, \vec{A}_i^{\,max}, \vec{J}_i^{\,max}\right), \tag{16}$$

Whether a degree of freedom is selected to be guided by the algorithm depends on the selection vector $\vec{S}_i$, which directly corresponds to the adaptive selection matrix. If a single Manipulation Primitive $\mathcal{MP}$ specifies the pose-based state feedback controller `Pose_Ctrl` at a certain level l, and if this level is the lowest available one (cf. availability flag vector $\vec{f}_i^{\texttt{Pose\_Ctrl}}$, eqn. (12)), the OTG algorithm will compute desired states of motion $\vec{M}_i$ for all selected degrees of freedom $k \in \left\{x, y, z, \textcircled{x}, \textcircled{y}, \textcircled{z}\right\}$. $\mathbf{B}_i$ and $\mathbf{M}_i^{trgt}$ are part of the set-point subset ${}^{l}D_i^{\texttt{Pose\_Ctrl}}$ (cf. Figs. 3 and 4). Internally, a time-synchronized trajectory to transfer the current state of motion $\mathbf{M}_{i-1}$ to the target state of motion $\mathbf{M}_i^{trgt}$ under consideration of the constraints $\mathbf{B}_i$ in the shortest possible time is generated.

How this framework can be applied in a hybrid switched-system was presented in [39]. With respect to the Manipulation Primitive framework, the OTG algorithm always runs as a safe backup controller at the highest controller level ($l = m$), because it can run independently of exteroceptive sensor signals and only relies on the current state of motion.

IV. Continuous Motions During Abrupt Switchings

The previous section summarized the Manipulation Primitive framework and suggested a highly reactive control architecture based on a hybrid switched-system. Such ar-

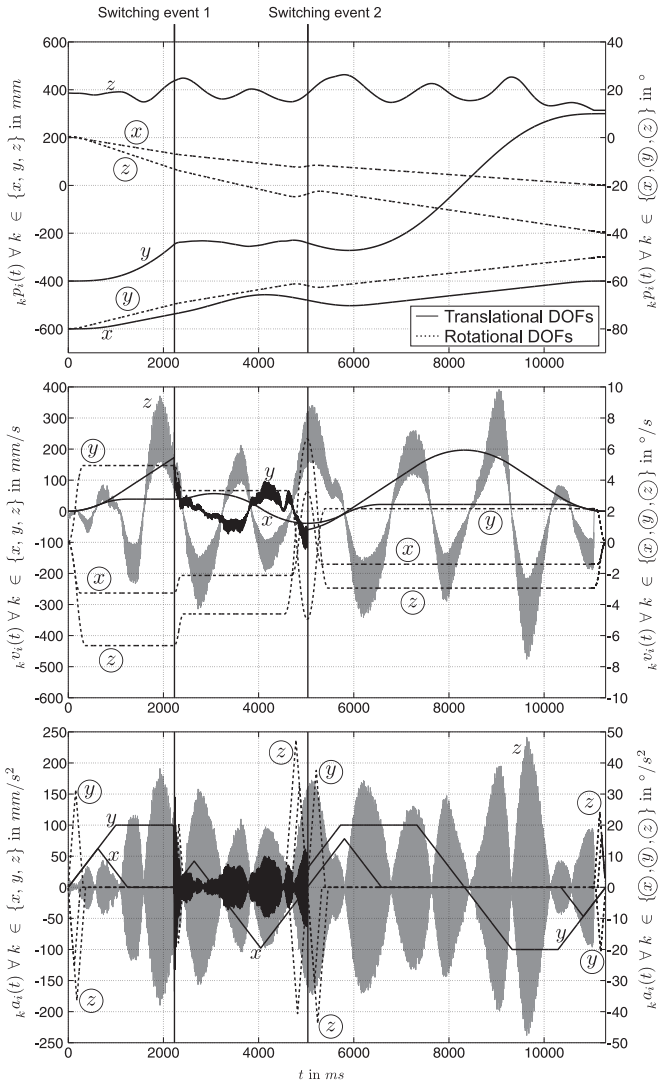

Fig. 5. Position, velocity, and acceleration progressions during one single MP illustrating switchings between control submodules due to the adaptive selection matrix. DOF y is first guided by the OTG-based pose control submodule; at $t = 2.192\,ms$, ${}_yf_{2192}^{\mathtt{F/T\_Ctrl}}$ becomes `true`, and DOF y is compliantly controlled by the force controller; at $t = 5.024\,ms$, ${}_yf_{5024}^{\mathtt{F/T\_Ctrl}}$ becomes `false` again, and the OTG-based pose control submodule instantaneously takes over control from the current state of motion.

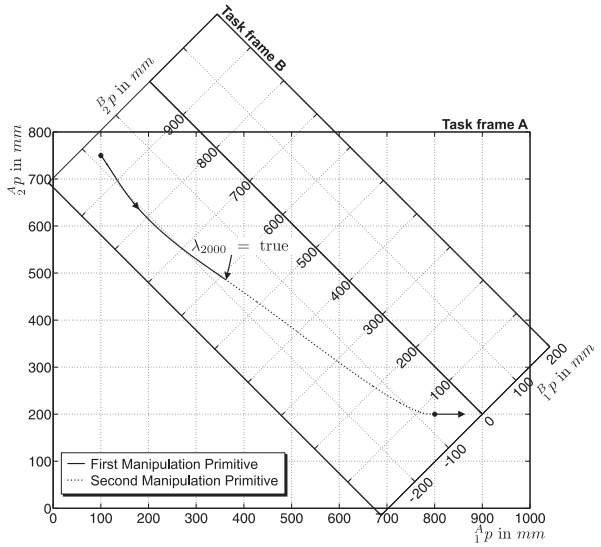

Fig. 6. XY-plot of the three-DOF *path* that corresponds to the trajectory of Fig. 7. The path of the first MP is shown by the solid line, and after the termination condition becomes true at $t = 2000\,ms$, the path of the second MP is shown by the dashed line (cf. [23]).

chitectures are very well-suited for real-world manipulation tasks. Switchings may happen at several levels, and a mechanism to generate smooth and continuous robot motions despite these abrupt switching procedures is required. This section discusses all kinds of different switchings in this context and suggests the use of an OTG algorithm that can instantaneously generate a robot motion trajectory from any arbitrary initial state of motion in response to unforeseen switching events.

A. Switchings During One Manipulation Primitive

During the execution of one single MP, the input values for the hybrid switched-system, that is, the set-point set $\mathcal{D}_i$ and the Task Frame parameters $\mathcal{TF}_i$, remain constant. Which of the m control submodules is activated for a degree of freedom $k \in \left\{x,\, y,\, z, \textcircled{x}, \textcircled{y}, \textcircled{z}\right\}$ at an instant T_i depends on the availability flag vectors $\vec{f}_i^{\,c}$ with $c \in \mathcal{C}$.

An example for such a switching procedure is shown in Fig. 5, which shows the trajectory of a KUKA Light-Weight Robot IV [40] during one single MP. DOF z is continuously controlled by a force controller, such that this DOF is compliant all the time. The DOFs x,$\textcircled{x}$,$\textcircled{y}$, and$\textcircled{z}$ are continuously controlled by the OTG-based pose control submodule. The switching procedure is shown by means of DOF y: first the OTG-based pose controller is active, after ${}_yf_{2192}^{\mathtt{F/T\_Ctrl}}$ becomes `true` at $t = 2.192\,ms$, this DOF is also controlled by the force control submodule; at $t = 5.024\,ms$, the availability flag ${}_yf_{5024}^{\mathtt{F/T\_Ctrl}}$ turns to `false` again, and the OTG-based pose control submodule instantaneously takes over control from the current state of motion.

B. Switchings Between Two Manipulation Primitives

After the termination condition λ_{i-1} becomes true in the control cycle at instant T_{i-1}, new values for $\mathcal{D}_i$ and $\mathcal{TF}_i$ are available in the control cycle at instant T_i. If the parameters of the set-point set $\mathcal{D}_i$ change, the elements of the adaptive selection matrix switch instantaneously, such that the same behavior as described in the previous subsection IV-A is obtained.

Changes in the Task Frame parameters $\mathcal{TF}_i$, however, require a transformation of all m control submodules, that is, the states of all controllers must be transformed in order to sustain continuous control output values. For the case that filters for sensor signals are applied — before and/or subsequent to a controller — the filter states, also have to be transformed into the new coordinate frame in order to sustain continuous control output values. Compared to filters and closed-loop controllers, which always possess a *state*, the

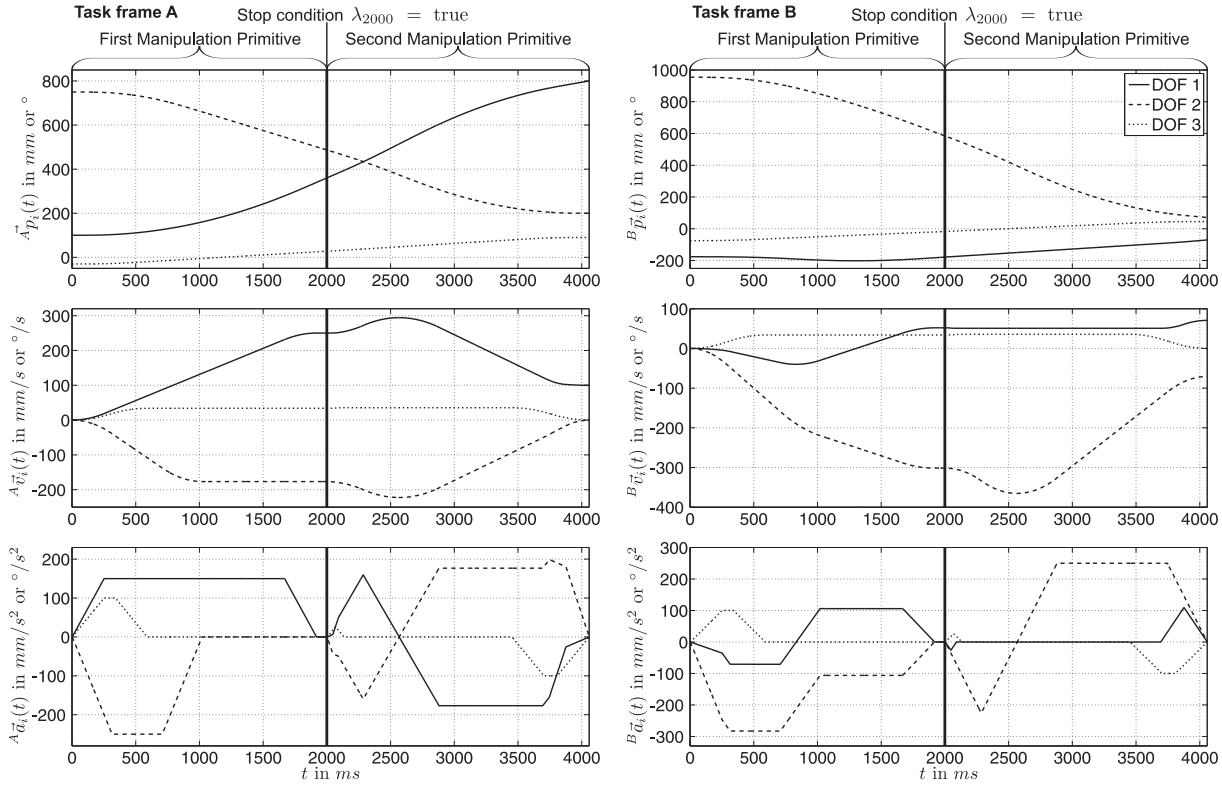

Fig. 7. Position, velocity, and acceleration progressions of a three-DOF trajectory. The left diagrams show the trajectory of the first MP (task frame A), and the right shows the trajectory of the second MP (task frame B). The termination condition becomes true at $t = 2000\,ms$. The corresponding path is displayed again in Fig. 6 (cf. [23]).

OTG algorithm is state-/memoryless. This module does not need to be transformed; it only receives input values given w.r.t. the new Task Frame.

Figures 7 and 6 show the trajectory and corresponding path of a switching procedure between two MPs.

C. Switchings Between Task and Actuator Space

As indicated in the actuator space control scheme of Fig. 3, it may also happen that we instantaneously switch from task space control to actuator space control (or vice versa) during an arbitrary motion of the robot and at unforeseen and sensor-dependant instants. Achieved with a Stäubli RX60 industrial robot [41], Fig. 8 shows results of this kind of switching behavior. Based on the high noise level, one can clearly recognize that the translational DOFs are controlled by a sensor feedback module, and the rotational ones are pose controlled to keep the orientation during the currently executed MP. From the moment of switching on, the hybrid controller is deactivated, a trajectory is generated from the current state of motion on, and the actuator space controller takes over control.

V. Summary

After a general review of the Manipulation Primitive framework and the On-Line Trajectory Generation framework, three different kinds of instantaneous switching procedures have been regarded, and experimental results have been presented: (1) switching between control submodules within one single Manipulation Primitive, (2) switching from one Manipulation Primitive to another, and (3) switching between task and actuator space control. During all switchings, continuous and jerk-limited robot motions are guaranteed. The proposed concept is based on a class of On-Line Trajectory Generation algorithms that are able to provide robot motion trajectories from any arbitrary state of motion within one control cycle. In the unforeseen moment of switching, these algorithms instantaneously provide a motion trajectory that continues the current motion without any additional jerk.

Acknowledgements

The research described in this paper was mainly conducted at the *Institut für Robotik und Prozessinformatik* at the Technische Universitaet Carolo-Wilhelmina zu Braunschweig, Braunschweig, Germany, headed by *Professor Friedrich M. Wahl*, to whom we would like to express our sincere gratitude. Furthermore, the first author would like to express his appreciation to *Professor Oussama Khatib*, who is currently hosting him at the *Stanford Artificial Intelligence Laboratory*. The works of our former diploma students, *Michaela Hanisch*, *Christian Hurnaus*, and *Adam Tomiczek* are highly appreciated. Moreover, the first author is deeply indebted to the *Deutsche Forschungsgemeinschaft* (DFG, German Research Foundation) that is currently funding him. Finally, we would like to thank *QNX Software Systems* and *Wind River Systems* for providing free software licenses.

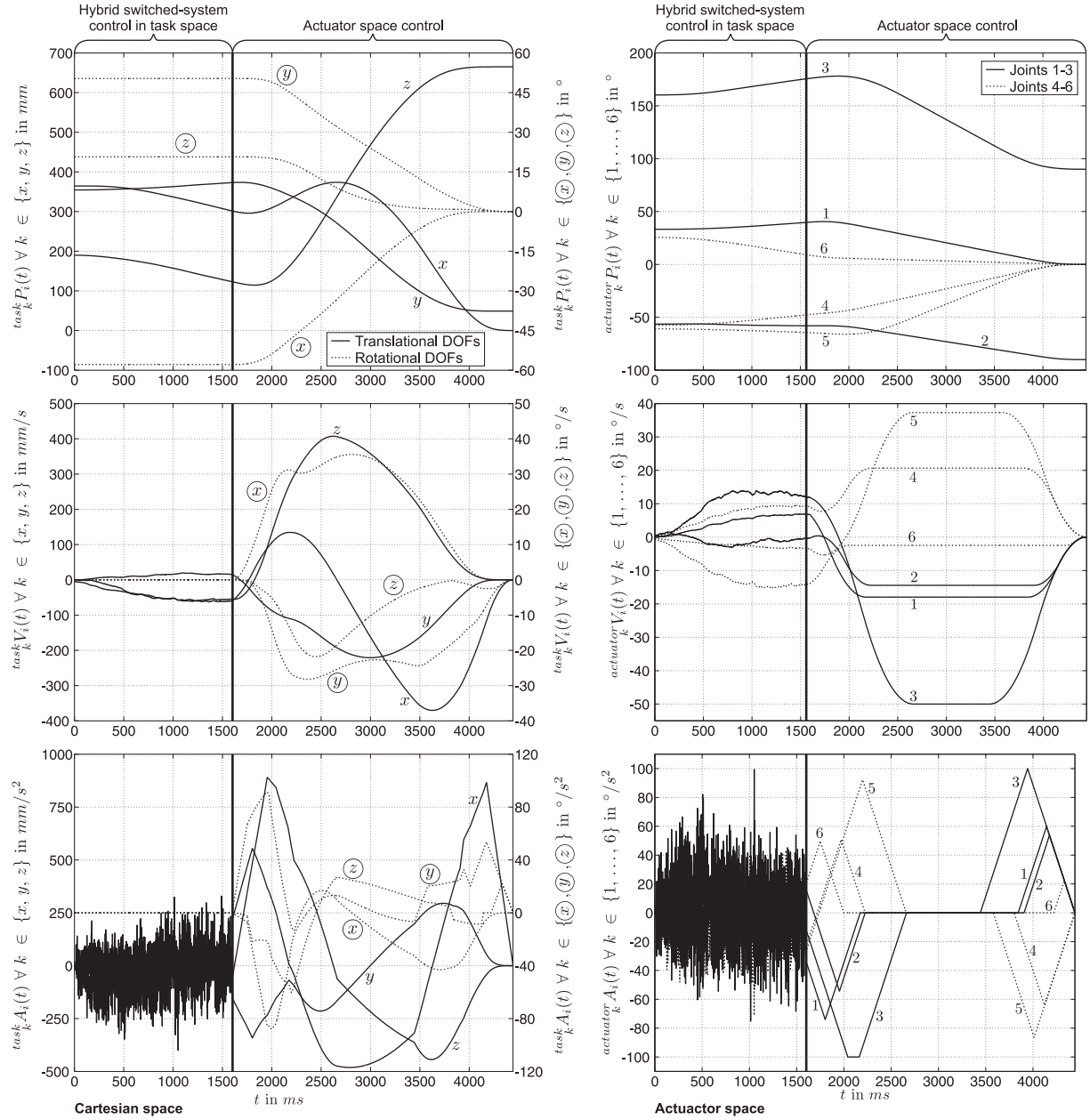

Fig. 8. Position, velocity, and acceleration progressions in Cartesian space (left) and in actuator space (right) of a sample motion of Stäubli RX60 industrial manipulator [41]. At $t = 1599\,ms$, a sensor event happens, and the switching form Cartesian space control to joint space control is performed instantaneously within the same control cycle, in which the event was detected (cf. [23]).

REFERENCES

[1] M. T. Mason. Compliance and force control for computer controlled manipulators. *IEEE Trans. on Systems, Man, and Cybernetics*, 11:418–432, June 1981.

[2] J. De Schutter and J. van Brussel. Compliant robot motion I. A formalism for specifying compliant motion tasks. *The International Journal of Robotics Research*, 7(5):3–17, August 1988.

[3] J. De Schutter and J. van Brussel. Compliant robot motion II. A control approach based on external control loops. *The International Journal of Robotics Research*, 7(4):18–33, August 1988.

[4] H. Bruyninckx. *Kinematic Models for Robot Compliant Motion with Identification of Uncertainities*. PhD thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium, 1995.

[5] H. Bruyninckx and J. De Schutter. Specification of force-controlled actions in the task frame formalism — A synthesis. *IEEE Trans. on Robotics and Automation*, 12(4):581–589, August 1996.

[6] H. Mosemann. *Beiträge zur Planung, Dekomposition und Ausführung von automatisch generierten Roboteraufgaben (in German)*. Shaker Verlag, Aachen, Germany, 2000.

[7] H. Mosemann and F. M. Wahl. Automatic decomposition of planned assembly sequences into skill primitives. *IEEE Trans. on Robotics and Automation*, 17(5):709–718, October 2001.

[8] B. Finkemeyer. *Robotersteuerungsarchitektur auf der Basis von Aktionsprimitiven (in German)*. Shaker Verlag, Aachen, Germany, 2004.

[9] B. Finkemeyer, T. Kröger, and F. M. Wahl. Executing assembly tasks specified by manipulation primitive nets. *Advanced Robotics*, 19(5):591–611, June 2005.

[10] B. Finkemeyer, T. Kröger, and F. M. Wahl. The adaptive selection matrix — A key component for sensor-based control of robotic manipulators. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3855–3862, Anchorage, AK, USA, May 2010.

[11] T. Kröger, B. Finkemeyer, and F. M. Wahl. Manipulation primitives — A universal interface between sensor-based motion control and robot programming. In D. Schütz and F. M. Wahl, editors, *Robot Systems for Handling and Assembly*, volume 67 of *Springer Tracts in Advanced Robotics*, pages 293–313. Springer, Berlin, Heidelberg, Germany, first edition, 2010.

[12] G. Milighetti and H.-B. Kuntze. On a primitive skill-based supervisory robot control architecture. In *Proc. of the IEEE International Conference on Advanced Robotics*, pages 141–147, Seattle, WA, USA, July 2005.

[13] G. Milighetti and H.-B. Kuntze. On the discrete-continuous control of basic skills for humanoid robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3474–3479, Beijing, China, October 2006.

[14] G. Milighetti and H.-B. Kuntze. Fuzzy based decision making for the discrete-continuous control of humanoid robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3580–3585, San Diego, CA, USA, October 2007.

[15] U. Thomas. *Automatisierte Programmierung von Robotern für Montageaufgaben (in German)*. Shaker Verlag, Aachen, Germany, 2008.

[16] U. Thomas, F. M. Wahl, J. Maaß, and J. Hesselbach. Towards a new concept of robot programming in high speed assembly applications. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3827–3833, Edmonton, Canada, August 2005.

[17] J. Maaß, S. Molkenstruck, U. Thomas, J. Hesselbach, F. M. Wahl, and A. Raatz. Definition and execution of a generic assembly programming paradigm. *Assembly Automation*, 28(1):61–68, 2008.

[18] J. Maaß, J. Steiner, A. Raatz, J. Hesselbach, U. Goltz, and A. Amado. Self-management in a control architecture for parallel kinematic robots. In *Proc. of the 27th ASME Computers and Information in Engineering Conference*, New York, NY, USA, August 2008.

[19] F. Dietrich, J. Maaß, A. Raatz, and J. Hesselbach. RCA562: Control architecture for parallel kinematic robots. In D. Schütz and F. M. Wahl, editors, *Robot Systems for Handling and Assembly*, volume 67, pages 315–331. Springer, Berlin, Heidelberg, Germany, first edition, 2010.

[20] T. Reisinger. *Kontaktregelung von Parallelrobotern auf der Basis von Aktionsprimitiven (Interaction Control of Parallel Robots Based on Skill Primitives)*. PhD thesis, Institut für Regelungstechnik, Technische Universität Carolo-Wilhelmina zu Braunschweig, `http://www.digibib.tu-bs.de/?docid=00022368` (accessed: Dec. 15, 2008), 2008.

[21] C. Zieliński and T. Winiarski. Motion generation in the mrroc++ robot programming framework. *The International Journal of Robotics Research*, September 2009.

[22] T. Kröger and F. M. Wahl. On-line trajectory generation: Basic concepts for instantaneous reactions to unforeseen events. *IEEE Trans. on Robotics*, 26(1):94–111, February 2010.

[23] T. Kröger. *On-Line Trajectory Generation in Robotic Systems*, volume 58 of *Springer Tracts in Advanced Robotics*. Springer, Berlin, Heidelberg, Germany, first edition, January 2010.

[24] S. Macfarlane and E. A. Croft. Jerk-bounded manipulator trajectory planning: Design for real-time applications. *IEEE Trans. on Robotics and Automation*, 19(1):42–52, February 2003.

[25] B. Cao, G. I. Dodds, and G. W. Irwin. A practical approach to near time-optimal inspection-task-sequence planning for two cooperative industrial robot arms. *The International Journal of Robotics Research*, 17(8):858–867, August 1998.

[26] X. Broquère, D. Sidobre, and I. Herrera-Aguilar. Soft motion trajectory planner for service manipulator robot. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2808–2813, Nice, France, September 2008.

[27] X. Broquère, D. Sidobre, and K. Nguyen. From motion planning to trajectory control with bounded jerk for service manipulator robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4505–4510, Anchorage, AK, USA, May 2010.

[28] S. Liu. An on-line reference-trajectory generator for smooth motion of impulse-controlled industrial manipulators. In *Proc. of the seventh International Workshop on Advanced Motion Control*, pages 365–370, Maribor, Slovenia, July 2002.

[29] R. Haschke, E. Weitnauer, and H. Ritter. On-line planning of time-optimal, jerk-limited trajectories. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3248–3253, Nice, France, September 2008.

[30] K. Ahn, W. K. Chung, and Y. Yourn. Arbitrary states polynomial-like trajectory (ASPOT) generation. In *Proc. of the 30th Annual Conference of IEEE Industrial Electronics Society*, volume 1, pages 123–128, Busan, South Korea, November 2004.

[31] R. H. Castain and R. P. Paul. An on-line dynamic trajectory generator. *The International Journal of Robotics Research*, 3(1):68–72, March 1984.

[32] X. Broquère. Homepage — Movies, `http://homepages.laas.fr/xbroquer/media2.php` (accessed: Mar. 6, 2011). Internet, 2011.

[33] W. Leonhard. *Control of Electrical Drives*. Springer, Berlin, Germany, third edition, 2001.

[34] SERCOS International e.V., Küblerstrasse 1, D-73079 Süssen, Germany. Homepage. `http://www.sercos.de` (accessed: Mar. 18, 2011). Internet, 2011.

[35] EtherCAT Technology Group, Ostendstraße 196, D-90482 Nuremberg, Germany. Homepage. `http://www.ethercat.org` (accessed: Mar. 18, 2011). Internet, 2011.

[36] W. Chung, L.-C. Fu, and S.-H. Hsu. Motion control. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 6, pages 133–159. Springer, Berlin, Heidelberg, Germany, first edition, 2008.

[37] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*, chapter 14, Motion Control, pages 347–376. Hermes Penton, Ltd., London, UK, first edition, 2002.

[38] M. W. Spong, S. A. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. John Wiley and Sons, 2006.

[39] T. Kröger and F. M. Wahl. Stabilizing hybrid switched motion control systems with an on-line trajectory generator. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4009–4015, Anchorage, AK, USA, May 2010.

[40] KUKA Roboter GmbH, Zugspitzstraße 140, D-86165 Augsburg, Germany. Homepage. `http://www.kuka.com/en/company/group` (accessed: Mar. 20, 2011). Internet, 2011.

[41] Stäubli Faverges SCA, Place Robert Stäubli BP 70, 74210 Faverges (Annecy), France. Homepage. `http://www.staubli.com/en/robotics` (accessed: Jan. 9, 2011). Internet, 2011.