2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

The Adaptive Selection Matrix — A Key Component for Sensor-Based Control of Robotic Manipulators

Bernd Finkemeyer, Torsten Kröger, and Friedrich M. Wahl

Abstract— **This contribution introduces a generic framework for sensor-based robot motion control. The key contribution is the introduction of an *adaptive selection matrix* for sensor-based hybrid switched-system control. The overall control system consists of multiple sensors and open- and closed-loop controllers, in-between which the adaptive selection matrix can switch discretely in order to supply command variables for low-level controllers of robotic manipulators. How control signals are chosen, is specified by *Manipulation Primitives*, which constitute the interface to higher-level programming. This programming paradigm is briefly specified in order to be able to define and execute sensor-guided *and* sensor-guarded motion commands simultaneously. The resulting control system is freely adaptable depending on the sensor and control requirements of the desired system and/or application.**

I. Introduction

The integration of sensors belongs to one of the most important future domains for achieving future advancements in robot motion control systems. As even nowadays commercial control units are rarely open for sensor integration, sensor signals are often (even in research institutions) used for sensor-based trajectory and path adaptations. This is a very implicit way of sensor-based control (e.g., force/torque control or visual servo control), and it — of course — already leads to great benefits compared to simple point to point operations, but in such systems, sensors are not part of the feedback control loop, and it is not possible at all to instantaneously react to unforeseen (sensor) events. On the other hand, we can find plenty of approaches using sensors in the feedback loops. Force/torque control [1] has had a dedicated community since the beginning of the 1980s as well as the field of visual servo control [2] has since the beginning of the 1990s; we can find plenty of publications in both areas, but the feature of instantaneous switchings is of fundamental relevance for many real-world problems and applications. This is one of the major reasons, why such approaches (e.g., [1], [2]) can hardly be found in practice.

The basic requirement for such immediate reactions of robotic systems is that their control structure can be abruptly changed and/or adopted from one control cycle to another. Right after some unforeseen event occurred, controllers may be switched from one configuration to another — depending on the current task specification. This paper has two intentions:

1) Giving a brief summary of *Manipulation Primitives* (MP). MPs are used to specify *sensor-guided* and *sensor-guarded* robot motions. This concept is already known and was published in [3].
2) Introducing the concept of the *adaptive selection matrix*. It is described how the parameters of a single MP are mapped to the elements of this particular matrix. Based on the system state given by all available sensor signals, the elements of the adaptive selection matrix can abruptly change the control structure in order to instantaneously react to unforeseen events.

Thus, it becomes possible to react to *uncertainties* within one control cycle (i.e., $\sim 1\,ms$ and less). Such reactions require a switching within the control structure, such that a hybrid switched-system results. After related works are introduced in the next section, Sec. III summarizes the concept of manipulation primitives, and Sec. IV describes how the parameters of MPs are mapped to the elements of the adaptive selection matrix. Furthermore, it is shown, how the elements of this matrix are used in a hybrid switched-control system for robotic manipulators.

II. Related Work

This section addresses related fields: Classic compliant motion control, which is based on force/pose control concepts; sensor integration methods (e.g., vision and distance sensor integration); robot task specification for sensor-based manipulations; control architectures including software technologies.

Mason [4] presented one of the pioneer works in compliant motion control in 1981. Based on his work numerous approaches have been published in this field, and especially the group of De Schutter contributed promising concepts to the community (e.g., [5]–[7]) and coined the phrase *Task Frame Formalism* (TFF), which enables the development of compliant motion solutions on an abstract programming level.

These works understand force/torque control as one basic part for compliant motion concepts [1]. Basically three different approaches are known from literature: 1. Impedance control [8], which uses relationships between acting forces and manipulator poses to adjust the mechanical impedance of the end-effector to external forces. 2. Parallel control [9], which enables to control both, force and pose, along the same task space direction. 3. Force/pose control which controls force and pose in two orthogonal subspaces [10]. The latter reference introduced the term *compliance selection matrix*, which constitutes the basis for the *adaptive selection*

B. Finkemeyer, T. Kröger, and F. M. Wahl are with the Institut für Robotik und Prozessinformatik at the Technische Universität Carolo-Wilhelmina zu Braunschweig, Mühlenpfordtstraße 23, D-38106 Braunschweig, Germany, berndfinkemeyer@kuka-roboter.de, {t.kroeger,f.wahl}@tu-bs.de. B. Finkemeyer is now with the KUKA Roboter GmbH, Zugspitzstraße 140, D-86165 Augsburg, Germany.

matrix. To realize this approach, we have to pay attention to the problem of orthogonality as stated by Duffy [11], who extended the approach, such that it is consistent, independent of units, and independent of any origin coordinate system.

Practical tasks of robot manipulators are usually affected by various uncertainties and inaccuracies. As an obvious result hybrid pose and force/torque control are not always sufficient; the integration of further sensors opens new possibilities and application fields. Especially, the addition of multiple sensors of the same kind, such that the system can situation-dependently benefit from the advantages of single sensors, leads to *robust* system behaviors.

Assuming a robot manipulation system with many different sensors considering a system that enables the execution of sensor-guided motion control commands in any degree of freedom (DOF), it becomes self-evident that we need to switch discretely between several (open- or closed-loop) continuously working discrete controllers at any time. Hence, the analysis of hybrid switched-system control is one fundamental part of the work presented here. Especially, the works of Branicky and Liberzon provide elementary concepts to develop and analyze hybrid switched-system control technologies. The works [12]–[14] are considered as the most relevant ones for this paper as will be pointed out in Sec. IV of this paper.

Systems clearly become more advanced, when the motion control loop is additionally closed by signals from vision systems. A recent work of Gans and Hutchinson describes, how switched-system control concepts can be used for pose- and image-based visual servo control [15]. In [2] and [16], the integration of visual servo control in the TFF approach is discussed.

Up to now, many research groups published approaches in the field of open control systems allowing the integration of multiple sensors (e.g., [17]–[19]). The work by Cortesão et al. [20] presents promising experimental results of force, vision, and pose data fusion.

Control of sensor-based manipulation systems is one major part of this paper, but — in addition — there is still a need for suitable robot programming paradigms allowing this kind of integration and programming when using multiple sensors. One recent and closely related work was published by De Schutter et al. [21], where a unified constraint-based framework of task specification was presented. For practical implementations, a generic interface is required, which remains unchanged, even when further sensors and/or corresponding controllers are integrated. Besides the parametrization of set-points, such a programming interface must be able to adapt the control system to the current work step and to the required sensor(s).

In order to extend the addressed works, we consider a generic and universal system with *Manipulation Primitives* as the top-level interface mapping sensor signals consistently to stable, unambiguous, and deterministic manipulator motions. This mapping is done by the adaptive selection matrix, which is the core part of this paper.

III. Manipulation Primitives

This section introduces *Manipulation Primitives* and gives a formal definition, which will be important for the next sections, where the concept of unambiguous mapping of MP parameters to low-level control is explained.

An MP is formally defined as the three-tuple

$$\mathcal{MP} := \{\mathcal{HM},\, \tau,\, \lambda\} \tag{1}$$

where $\mathcal{HM}$ defines a hybrid motion, τ contains tool commands, and the stop condition λ determines the end of execution of a single MP. These three quantities will be described and discussed in the following.

A. Hybrid Move $\mathcal{HM}$

$\mathcal{HM}$ defines a hybrid move in the sense of the TFF [5], which becomes extended here in order to take multiple sensors into consideration. In the classical case, a motion command is given w.r.t. the Task Frame and is determined by a six-dimensional vector representing Mason's *Compliance Frame* [4]. This simple matrix is not a sufficient and in particular not a practical way to parameterize sensor-guided robot motion commands when considering more than one sensor.

The question of how to specify motion commands unambiguously and universally, such that any sensor can be addressed, must be answered in this context. For this purpose, the hybrid motion command $\mathcal{HM}$ is defined as

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

with

$$\mathcal{TF} := \{\vec{\theta},\, RF,\, ANC,\, FFC\} \tag{3}$$

$$\vec{\theta} = \left(\theta_x,\, \theta_y,\, \theta_z,\, \theta_{\textcircled{x}},\, \theta_{\textcircled{y}},\, \theta_{\textcircled{z}}\right)^T \in \mathbb{R}^6 \tag{4}$$

$$RF,\, ANC \in \{HF,\, WF,\, BF,\, EF\} \tag{5}$$

$$FFC \in \{WF,\, BF,\, EF\} \tag{6}$$

and

$$\begin{aligned} \mathcal{D} \;:=\; & \{D_i | D_i = \iota_i \cup \xi_i;\; \iota_i \in \zeta,\, \forall\, i,\, j\!:\; i \neq j \\ & \Longrightarrow\; \iota_i \neq \iota_j;\; i,\, j \in \{0,\, \ldots,\, |\zeta|\}\} \end{aligned} \tag{7}$$

with

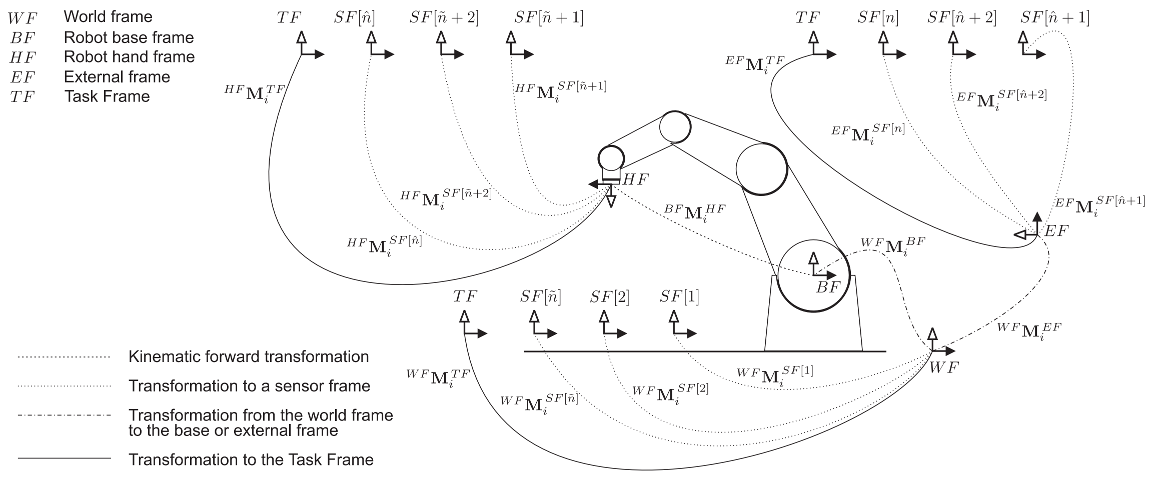
$$\zeta \;:=\; \{x,\, y,\, z, \textcircled{x}, \textcircled{y}, \textcircled{z}\} \times \{0,\, \ldots,\, (m-1)\} \tag{8}$$

$$\iota_i \;:=\; \{dof\_id_i, level\_id_i\} \in \zeta \tag{9}$$

$$\xi_i \;:=\; \{value_i, device_i\} \tag{10}$$

Let us discuss eqns. (2) – (6) and eqns. (7) – (10) in the following.

The hybrid move command $\mathcal{HM}$ of eqn. (2) is specified w.r.t. the Task Frame $\mathcal{TF}$, in which a set of set-points $\mathcal{D}$ is applied. All control values are measured in some Sensor Frame $SF^{[j]}$ with $j \in \{1,\, \ldots,\, n\}$, which may be located anywhere and must be transformable into the currently applied Task Frame (cf. Fig. 1). In contrast to the classical TFF, we admit coupling of the Task Frame w.r.t. any



WF World frame
BF Robot base frame
HF Robot hand frame
EF External frame
TF Task Frame
TF
$SF[\tilde{n}]$
$SF[\tilde{n}+2]$
$SF[\tilde{n}+1]$
TF
$SF[n]$
$SF[\hat{n}+2]$
$SF[\hat{n}+1]$
${}^{HF}\mathbf{M}_i^{TF}$
${}^{HF}\mathbf{M}_i^{SF[\tilde{n}+1]}$
${}^{EF}\mathbf{M}_i^{TF}$
${}^{EF}\mathbf{M}_i^{SF[\hat{n}+2]}$
${}^{EF}\mathbf{M}_i^{SF[n]}$
HF
${}^{HF}\mathbf{M}_i^{SF[\tilde{n}+2]}$
${}^{BF}\mathbf{M}_i^{HF}$
${}^{EF}\mathbf{M}_i^{SF[\hat{n}+1]}$
EF
${}^{HF}\mathbf{M}_i^{SF[\tilde{n}]}$
${}^{WF}\mathbf{M}_i^{BF}$
BF
TF
$SF[\tilde{n}]$
$SF[2]$
$SF[1]$
${}^{WF}\mathbf{M}_i^{EF}$
WF
Kinematic forward transformation
Transformation to a sensor frame
Transformation from the world frame to the base or external frame
Transformation to the Task Frame
${}^{WF}\mathbf{M}_i^{TF}$
${}^{WF}\mathbf{M}_i^{SF[\tilde{n}]}$
${}^{WF}\mathbf{M}_i^{SF[2]}$
${}^{WF}\mathbf{M}_i^{SF[1]}$

Fig. 1. Frame assignments according to [3] for the proposed TFF specification.

frame in the work cell. $\vec{\theta}$ determines the pose of the Task Frame w.r.t. the Reference Frame RF at the beginning of a single MP execution. The translational DOFs are denoted by x, y, and z, the rotational ones by $\textcircled{x}$, $\textcircled{y}$, and $\textcircled{z}$. RF can be selected from a set of frames known by the system (e.g., World Frame WF, Base Frame BF, or Hand Frame HF). Pose, velocity, and acceleration of each frame is updated every control cycle, typically every millisecond. The frame ANC serves as anchor and connects the Task Frame rigidly to another frame of the work cell. As known from [3] and shown in Fig. 1, the Task Frame can be either anchored to

- the Hand Frame of the manipulator ($ANC = HF$),
- a fixed frame (e.g., $ANC = WF$ or $ANC = BF$), or
- an External Frame known by the control system ($ANC = EF$).

The motion state (pose, velocity, and acceleration) of the Task Frame w.r.t. the Anchor Frame ${}^{ANC}\mathbf{M}^{TF}$ is constant during the complete MP execution. Hence, the ANC frame has a fundamental influence on the motion to be executed. The frame FFC (eqns. (3) and (6), feedforward compensation) usually equals the World Frame or the robot Base Frame if a mobile manipulation system is considered. If (compliant) motions are to be executed w.r.t. an external moving coordinate system, the FFC frame (i.e., its pose, velocity, and acceleration) enables the internal computation of a feedforward compensation (FFC) signal, such that a sensor-based motion command can be executed in dynamic systems in the same way as in in static ones [22]. A detailed description on these parameters as well as examples can be found in [3].

The meaning of the set-point set $\mathcal{D}$ of eqns. (7)–(10) will be explained in this paragraph. Assuming, we only apply trajectory-following and force/torque control to a robotic manipulation system with six Cartesian DOFs; the consideration of all possible combinations of pose and force/torque set-points would lead to $2^6 = 64$ hybrid move commands. If controller switchings should be realized with three controllers (e.g., by adding a distance or visual servo controller), the number of motion commands would increase to $3^6 = 729$. Of course, this is not a practical solution. Thus, it has to be possible to determine an alternative set-point individually for each DOF within every control cycle (i.e., on-line). For example, if force/torque control in one direction is not possible, the system should be able to automatically switch to the next alternative (open- or closed-loop) controller for the respective DOF. If the prerequisites to apply this controller are not fulfilled, a further alternative may be chosen etc. If no further alternative can be determined, a save backup controller has to be used. Commonly, this is an only trajectory generation module [23], [24], which is able to generation command variables without sensor feedback and from any arbitrary state of motion. This way, a deterministic and stable robot behavior is guaranteed in every situation. The set $\mathcal{D}$ contains up to $6m$ set-points D_i, where m is the number of available controllers; each set-point D_i consists of two attributes:

- ξ_i, a tuple containing $value_i$, which is assigned to a particular controller $device_i$, and
- ι_i defining the DOF dof_id_i, to which the set-point is applied as well as the level $level\_id_i$.

ζ is a set of possible DOF and level combinations and generally contains $|\zeta| = 6\,m$ elements. Hence, we cannot only specify one set-point per DOF but also a set of alternative set-points. Of course, these alternatives are optional, and its number depends on the current task. They are indicated unambiguously by the level ID value ($level\_id_i \in \mathbb{N}$). A set-point with $level\_id = 0$ represents a desired robot state of first choice. $level\_id = 1$ denotes the first alternative set-point and so on. For example, in y-direction force control with a desired value of $(-15\,N)$ has been chosen. If force control is not possible (e.g., because there is no contact), the corresponding DOF will automatically be controlled by a distance controller with a set-point of $0\ mm$. If this controller is also not able to service the DOF (e.g., in case the robot is outside of the sensor's measurement range),

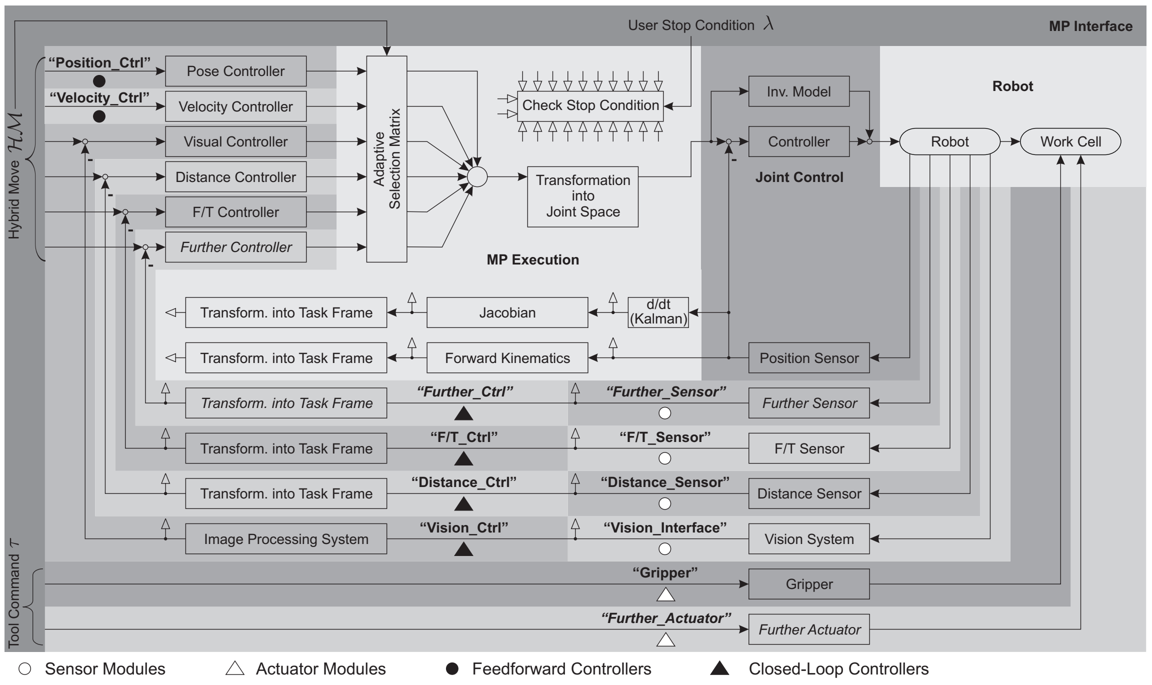


Fig. 2. Overview of the hybrid switched-system control scheme to execute Manipulation Primitives.

the determined open-loop velocity controller is activated. If no further alternative is available, a save default backup controller will be activated. In general, a pose controller will keep the robot in a stable state in this situation and the current MP would be terminated with an error state. Of course, any combination of controllers and set-points is possible as a matter of principle. By means of the attribute $device_i$, it is possible to choose the optimal controller for each task, i.e. the best appropriate force or visual servo control concept etc. comes to execution. The necessary switching processes are handled by the *adaptive selection matrix* (cf. Sec. IV).

B. Tool Command τ

The second part of an MP, the *tool command* τ, allows the integration and the access to any tool (gripper, drilling machine, welding apparatus, etc.) mounted to the manipulator's end-effector or to any device of the work cell. Formally, it is defined as

$$\tau := \{\tau_i | \tau_i = \{tool\_name, command\}\} \tag{11}$$

The structure of the $command$ attribute is kept very open and very general in order to permit the integration of very complex tools.

C. Stop Condition λ

The *stop condition* defines the termination of an MP. A single MP runs until the sensor states and the robot state, which are defined by the user in the stop condition λ, are reached. This leads to a higher flexibility regarding the simultaneous execution of sensor-guided *and* sensor-guarded motions, motions, which even may exhibit fault tolerances due to kinematic inaccuracies and to uncertainties in the environment.

The user stop condition is a Boolean expression defined as

$$\lambda := \mathcal{S} \longrightarrow \{true, false\} \tag{12}$$

where $\mathcal{S}$ is the set of available sensors and their corresponding filter functions. The control cycle, in which λ becomes true, lets the current MP terminate and the next MP becomes executed until its stop condition becomes also true, etc.

IV. The Adaptive Selection Matrix

This section is the core part of this paper and describes the control architecture for the execution of MP and introduces adaptive selection matrix, which is responsible for the mapping of MP parameters to lower-level controllers.

A. Control Architecture

The overall control architecture can be considered a hybrid switched-system [13], [14]. Fig. 2 depicts a rough overview of such a control scheme. Here, hybrid switched-system control of pose, velocity, force/torque (F/T), distance, and vision is indicated. Of course, this set can be expanded by any other physical variable and/or controller, respectively. At first glance, the structure seems similar to common hybrid switched-system control structures. But in contrast to common hybrid switched-system controllers, the selection matrix is not static during one robot command: It considers the current robot and environment state. Thus, it is an *adaptive selection matrix.*

B. The Adaptive Selection Matrix

As described in the previous section, the MP approach allows the definition of several alternative set-points (eqns.

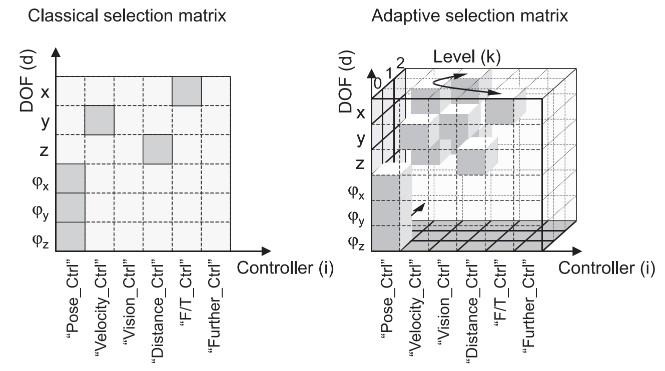


Fig. 3. Classical two-dimensional assignment (left) and adaptive three-dimensional assignment (right) of controllers and DOFs.

(7) – (10)), such that a stable and efficient robot control is feasible for each individual DOF.

In common hybrid switched-system control approaches, selection matrices $\mathbf{S}_i$ are generated from the compliance frame for each available controller type. The corresponding controller is addressed by the index i. The resulting six-dimensional control variable vector $\vec{o}$ is computed out of the n available control variable vectors $\vec{o_i}$ of the involved controllers in the following way:

$$\vec{o} \;=\; \sum_{i=0}^{n-1} \mathbf{S}_i\; \vec{o_i} \tag{13}$$

where the following condition with the identity matrix $\mathbf{I}$ must be fulfilled

$$\sum_{i=0}^{n-1} \mathbf{S}_i \;=\; \mathbf{I} \tag{14}$$

This results in the required unique assignment of controllers to DOFs. Fig. 3 (left) depicts a sample assignment.

For the execution of MPs, which allow the usage of any kind and any number of sensors, this simple two-dimensional view of selection matrices does not suffice. The definition of several alternative control loops requires the extension by a third dimension, representing the *control level*. This three-dimensional perception is illustrated in the right part of Fig. 3. The control loops of a DOF d are shifted along the double-headed arrow. Thus, the active control variable can no longer be determined by constant selection matrices $\mathbf{S}_i$ of eqn. (13). The selection works dynamically and depends on two factors:

1) The currently available sensors and controllers as well as the current system state.
2) The assignment of controllers and control levels per DOF.

Formally, a controller i delivers three matrices: the assignment matrix $\mathbf{Z}_i$, the control matrix $\mathbf{O}_i$, and the availability matrix $\mathbf{F}_i$. They are the basis for the calculation of the control variable assignment matrix $\mathbf{E}_o$ and flag assignment matrix $\mathbf{E}_f$, which are used to determine the selection matrix $\mathbf{L}$, from which the resulting control value vector $\vec{o}$ can be derived. All these matrices and all calculation steps will be explained in the following.

Assignment Matrices $\mathbf{Z}_i$ As described in Sec. III, it is possible to determine alternatives for each set-point. Thus, each DOF d contains several control levels k. The assignment of the i-th controller to a control level k is formally represented by an assignment matrix $\mathbf{Z}_i$:

$$\mathbf{Z}_i \;=\; \begin{pmatrix} z_{0,\,x} & z_{0,\,y} & \cdots & z_{0,\,\textcircled{z}} \\ z_{1,\,x} & z_{1,\,y} & \cdots & z_{1,\,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ z_{k,\,x} & z_{k,\,y} & \cdots & z_{k,\,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ z_{(m-1),\,x} & z_{(m-1),\,y} & \cdots & z_{(m-1),\,\textcircled{z}} \end{pmatrix} \;\in\; \mathbb{B}^{m\times 6}\,, \tag{15}$$

$$\text{where } \mathbb{B} \;=\; \{0,\,1\}\;.$$

Each column corresponds to a DOF and each row represents a control level, where m determines the maximum number of control levels. An entry of '1' assigns the DOF of the i-th controller to the control level k. The matrix corresponds to a vertical slice of the three-dimensional adaptive selection matrix representation of Fig. 3 (right).

The following assignment matrix $\mathbf{Z}_{\texttt{Velocity\_Ctrl}}$ has been defined for velocity control:

$$\mathbf{Z}_{\texttt{Velocity\_Ctrl}} \;=\; \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{16}$$

Here, the controller is selected to manage the y-direction on level 0 and to manage the x- and z-direction on level 1. All remaining DOFs are controlled by some other controller.

The assignment matrices $\mathbf{Z}_i$ are implicitly defined by the set-point set $\mathcal{D}$ and are automatically computed by the control system.

Control Matrices $\mathbf{O}_i$ The control variables of one single controller i are summarized as vector $\vec{o}_i$. The dimension of the vector corresponds to the number of available DOFs of the robotic system. For computing the adaptive selection matrix, the control variables are represented by the diagonal elements of a control matrix $\mathbf{O}_i$. For a robot with six DOFs, the matrix $\mathbf{O}_i$ for one single device i is defined as

$$\mathbf{O}_i \;=\; \begin{pmatrix} o_x & 0 & 0 & 0 & 0 & 0 \\ 0 & o_y & 0 & 0 & 0 & 0 \\ 0 & 0 & o_z & 0 & 0 & 0 \\ 0 & 0 & 0 & o_{\textcircled{x}} & 0 & 0 \\ 0 & 0 & 0 & 0 & o_{\textcircled{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & o_{\textcircled{z}} \end{pmatrix} \;\in\; \mathbb{R}^{6\times 6} \tag{17}$$

Thus, the common control variable vector $\vec{o}_i$ of the i-th controller can be derived from $\mathbf{O}_i$ as follows:

$$\vec{o}_i \;=\; \operatorname{diag}(\mathbf{O}_i) \tag{18}$$

Availability Matrix $\mathbf{F}_i$ To avoid the usage of control variables of inoperative control loops (e.g., force/torque control in free space), the validity is indicated by a flag. A value of '1' denotes a usable control variable and '0' denotes an invalid control variable. The elements of the availability matrix $\mathbf{F}_i$ contain the mentioned flags of the i-th controller. For six DOFs, $\mathbf{F}_i$ is given by

$$\mathbf{F}_i = \begin{pmatrix} f_x & 0 & 0 & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 & 0 & 0 \\ 0 & 0 & f_z & 0 & 0 & 0 \\ 0 & 0 & 0 & f_{\textcircled{x}} & 0 & 0 \\ 0 & 0 & 0 & 0 & f_{\textcircled{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & f_{\textcircled{z}} \end{pmatrix} \in \mathbb{B}^{6\times 6} \tag{19}$$

The availability flag vector $\vec{f_i}$ of the i-th controller can be written as

$$\vec{f_i} = \mathrm{diag}\,(\mathbf{F}_i) \tag{20}$$

Control Variable Assignment Matrix $\mathbf{E}_o$ The calculation of the resulting control value vector $\vec{o}$ requires the mapping of the control values of all controllers to the corresponding DOF d *and* control level k. This is possible, as the definition of eqn. (7) ensures, that one and only one controller can be chosen per DOF and level. The mapping is represented by the control variable assignment matrix $\mathbf{E}_o$. The calculation of $\mathbf{E}_o$ with n involved controllers leads to

$$\begin{aligned} \mathbf{E}_o &= \textstyle\sum_{i=0}^{n-1} \mathbf{Z}_i\ \mathbf{O}_i \\ &= \begin{pmatrix} o'_{0,x} & o'_{0,y} & \cdots & o'_{0,\textcircled{z}} \\ o'_{1,x} & o'_{1,y} & \cdots & o'_{1,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ o'_{k,x} & o'_{k,y} & \cdots & o'_{k,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ o'_{(m-1),x} & o'_{(m-1),y} & \cdots & o'_{(m-1),\textcircled{z}} \end{pmatrix} \in \mathbb{R}^{m\times 6} \end{aligned} \tag{21}$$

Each column corresponds to a DOF d and each row to one particular control level k.

Selection Matrix L To calculate the control value vector $\vec{o}$, which contains the input values for the *joint controller* (cf. Fig. 2), the correct row of $\mathbf{E}_o$ must be chosen for each single DOF. This is realized by the selection matrix $\mathbf{L}$. It is defined as

$$\mathbf{L} = \begin{pmatrix} l_{0,x} & l_{0,y} & \cdots & l_{0,\textcircled{z}} \\ l_{1,x} & l_{1,y} & \cdots & l_{1,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ l_{k,x} & l_{k,y} & \cdots & l_{k,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ l_{(m-1),x} & l_{(m-1),y} & \cdots & l_{(m-1),\textcircled{z}} \end{pmatrix} \in \mathbb{B}^{m\times 6} \tag{22}$$

The columns are assigned to the system DOFs, and the rows are assigned to control levels. The entries contain '1' or '0', where '1' selects the corresponding value. Of course, it exists exactly one '1' per column. Thus, we obtain

$$\sum_{k=0}^{m-1} l_{k,d} \overset{!}{=} 1 \quad \text{with } d \in \left\{x,\, y,\, z, \textcircled{x}, \textcircled{y}, \textcircled{z}\right\} \tag{23}$$

Transposing $\mathbf{L}$ and multiplying it with $\mathbf{E}_o$ results in a symmetric matrix, whose diagonal elements contain the resulting control value vector $\vec{o}$ (cf. eqn. (13)). It can be written as

$$\vec{o} = \mathrm{diag}\left(\mathbf{L}^T\ \mathbf{E}_o\right) \tag{24}$$

Please note that the elements of $\mathbf{L}$ are still undetermined. For its calculation the flag assignment matrix $\mathbf{E}_f$ is responsible.

Flag Assignment Matrix $\mathbf{E}_f$ The matrix $\mathbf{L}$ depends on the n available $\mathbf{F}_i$ matrices of the n involved controllers (with $i \in \{0, \ldots, n-1\}$ and $n \leq 6\,m$) and on the assignment matrixes $\mathbf{Z}_i \ \forall\ i \in \{0, \ldots, n-1\}$. This results in the flag assignment matrix

$$\begin{aligned} \mathbf{E}_f &= \textstyle\sum_{i=0}^{n-1} \mathbf{Z}_i\ \mathbf{F}_i \\ &= \begin{pmatrix} f'_{0,x} & f'_{0,y} & \cdots & f'_{0,\textcircled{z}} \\ f'_{1,x} & f'_{1,y} & \cdots & f'_{1,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ f'_{k,x} & f'_{k,y} & \cdots & f'_{k,\textcircled{z}} \\ \vdots & \vdots & \vdots & \vdots \\ f'_{(m-1),x} & f'_{(m-1),y} & \cdots & f'_{(m-1),\textcircled{z}} \end{pmatrix} \in \mathbb{B}^{m\times 6} \end{aligned} \tag{25}$$

1) Determination of $\mathbf{L}$ *and* $\vec{o}$*:* The columns (DOFs) of $\mathbf{E}_f$ can be mapped to the corresponding columns of $\mathbf{L}$. The mapping law for each single DOF d ($d \in \{x,\, y,\, z, \textcircled{x}, \textcircled{y}, \textcircled{z}\}$) is given in the following table. A '$\times$' entry means *'don't care'*.

$f'_{0,d}$	$f'_{1,d}$	...	$f'_{k,d}$	...	$f'_{(m-1),d}$	$l_{0,d}$	$l_{1,d}$	...	$l_{k,d}$	...	$l_{(m-1),d}$
0	0	...	0	...	0	0	0	...	0	...	0
1	$\times$	...	$\times$	...	$\times$	1	0	...	0	...	0
0	1	...	$\times$	...	$\times$	0	1	...	0	...	0
0	0	...	1	...	$\times$	0	0	...	1	...	0
0	0	...	0	...	1	0	0	...	0	...	1

As can be seen from the table, one and only one controller is active per DOF, such that it is in accordance with eqn. (23). The available controller with the lowest $level\_id$ will be activated. The first line of the table indicates an error state: No controller of the current MP is able to service the DOF. In such a case, the MP will be terminated, and the corresponding DOF will be controlled by a backup controller. The table can be rewritten as

$$\begin{aligned} l_{0,d} &= f'_{0,d} \\ l_{1,d} &= \overline{f'_{0,d}} \wedge f'_{1,d} \\ &\ \vdots \\ l_{k,d} &= \overline{f'_{0,d}} \wedge \overline{f'_{1,d}} \wedge \cdots \wedge f'_{k,d} \\ &\ \vdots \\ l_{(m-1),d} &= \overline{f'_{0,d}} \wedge \overline{f'_{1,d}} \wedge \cdots \wedge \overline{f'_{k,d}} \wedge \cdots \wedge f'_{(m-1),d} \end{aligned} \tag{26}$$

As these terms become very complex for multiple levels, we introduce the Boolean variable $\rho_{k,d}$

$$\begin{aligned} \rho_{k,d} \ :=\ & \overline{f'_{k,d}} \wedge \rho_{(k-1),d}, \text{ where} \\ & k \in \{0, \ldots, (m-1)\} \text{ and } \rho_{(-1),d} := 1\ . \end{aligned} \tag{27}$$

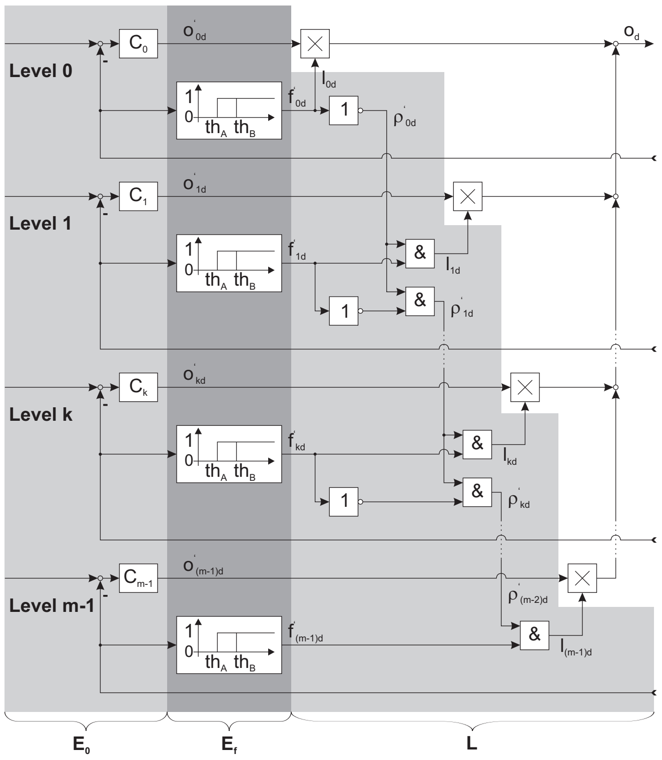


Fig. 4. Technical representation for one DOF of the adaptive selection matrix.

Now eqn. (26) can be rewritten as

$$
\begin{aligned}
l_{0,\,d} &= f'_{0,\,d} \\
l_{1,\,d} &= \rho_{0,\,d} \wedge f'_{1,\,d} \\
&\vdots \\
l_{k,\,d} &= \rho_{(k-1),\,d} \wedge f'_{k,\,d} \\
&\vdots \\
l_{(m-1),\,d} &= \rho_{(m-2),\,d} \wedge f'_{(m-1),\,d}
\end{aligned}
\tag{28}
$$

By means of this recursive mapping rule and by applying eqn. (24), $\vec{o}$ can be computed as

$$
\vec{o} = \begin{pmatrix}
\sum_{k=0}^{m-1} & o'_{k,\,x} & \big(\rho_{(k-1),\,x} & \wedge & f'_{k,\,x}\big) \\
\sum_{k=0}^{m-1} & o'_{k,\,y} & \big(\rho_{(k-1),\,y} & \wedge & f'_{k,\,y}\big) \\
\sum_{k=0}^{m-1} & o'_{k,\,z} & \big(\rho_{(k-1),\,z} & \wedge & f'_{k,\,z}\big) \\
\sum_{k=0}^{m-1} & o'_{k,\,\textcircled{x}} & \big(\rho_{(k-1),\,\textcircled{x}} & \wedge & f'_{k,\,\textcircled{x}}\big) \\
\sum_{k=0}^{m-1} & o'_{k,\,\textcircled{y}} & \big(\rho_{(k-1),\,\textcircled{y}} & \wedge & f'_{k,\,\textcircled{y}}\big) \\
\sum_{k=0}^{m-1} & o'_{k,\,\textcircled{z}} & \big(\rho_{(k-1),\,\textcircled{z}} & \wedge & f'_{k,\,\textcircled{z}}\big)
\end{pmatrix}
\tag{29}
$$

Fig. 4 presents the resulting block diagram of eqn. (29) for one DOF. It summarizes the mathematical representation of the *adaptive selection matrix* from a technical point of view.

C. Final Remarks on the Availability Flag Vector $\vec{f_i}$

The availability flag vector decides if a control loop is currently able to cope with the current system state. Thus, each element of $\vec{f_i}$ is determined by a function that maps state variables, sensor signals, or any other events to a logical value. One element d of $\vec{f_i}$ of the i-th controller can be written as

$$
f_{i,d} := \mathcal{S} \longrightarrow \mathbb{B} \tag{30}
$$

where $\mathcal{S}$ is the set of available sensor signals (cf. eqn. (12)). The mapping function depends on the corresponding controller. It can be a simple constant ($f_{i,d} = 1$), the result of a comparison with a threshold (th_A and th_B, cf. Fig. 4), or any complex function. By means of this flag function stability of each continuously working discrete controller in the hybrid switched-system must be guaranteed.

In this way many critical situations can be handled. Some examples are outlined in the following:

- Managing the transition from free space motions to force/torque controlled or any sensor-guided compliant motions, respectively.
- Any internal controller errors, for example, a missing sensor signal, can set the flag to zero.
- If switching to a controller yields heavy jerk values in the robot's end-effector or drives, it can be avoided by this flag.
- It becomes possible to implement monitoring and safety functions with the adaptive selection matrix.

As already stated in Sec. II, the problem of orthogonality is very relevant. Compared to classic approaches [4], [6], [7], orthogonality is always guaranteed by the adaptive selection matrix in principle.

This section introduced the adaptive selection matrix — the main contribution of this paper. One further important issue concerns the switching behavior of the adaptive selection matrix. Predefined trajectories as they are common in the states of research and technology are unsuited. All non-sensor-guided motions must be generated by an on-line trajectory generator, which is able to take over control for one or more DOFs from any arbitrary system state [23], [24].

V. Practical Results

In order to save space, we refer to other publications showing experimental as well as practical results achieved with this concept. The manipulator of [25] plays the parlor game Jenga [26], and applies force/torque, acceleration, vision, and distance distance sensors in one exhibit. [27] and [28] deliver further details of this concept. How the concept of MPs can be used as an interface to higher-level motion planning systems, which consider inaccuracies in the kinematics as well as in the environment (e.g., [21]), can be found in [3] and [24].

VI. Conclusion

We introduced a very basic but nevertheless powerful and generic interface represented by *Manipulation Primitives* to simultaneously specify sensor-guided and sensor-guarded robot motion commands, which are executed by a hybrid switched-system controller. As a consequence (multi-)sensor integration in robotic manipulation control systems becomes strongly simplified. The major contribution of this paper is the introduction of the *adaptive selection matrix*, which maps the parameters of the manipulation primitive paradigm to low-level control layers including hybrid switched-system control structures.

We would like to encourage developers to reason on practical implementations of the Task Frame Formalism in the sense proposed in this paper. As a result, the potential of this formalism becomes clearer and thus enables the research communities to provide further advancements in robotics and automation systems.

Acknowledgements

The authors would like to thank *QNX Software Systems* for providing free software licenses.

References

[1] L. Villani, , and J. De Schutter. Force control. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 7, pages 161–185. Springer, Berlin, Heidelberg, Germany, first edition, 2008.

[2] F. Chaumentte and S. A. Hutchinson. Visual servoing and visual tracking. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 24, pages 563–583. Springer, Berlin, Heidelberg, Germany, first edition, 2008.

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

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

[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] 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.

[7] 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.

[8] N. Hogan. Impedance control: An approach to manipulation. Part I: Theory. Part II: Implementation. Part III: Applications. *ASME Journal of Dynamic Systems, Measurment, and Control*, 107:1–24, March 1985.

[9] S. Chiaverini and L. Sciavicco. The parallel approach to force/position control of robotic manipulators. *IEEE Trans. on Robotics and Automation*, 9(4):361–373, August 1993.

[10] M. H. Raibert and J. J. Craig. Hybrid position/force control of manipulators. *ASME Journal of Dynamic Systems, Measurement and Control*, 102:126–133, June 1981.

[11] J. Duffy. The fallacy of modern hybrid control theory that is based on "orthogonal complements" of twist and wrench spaces. *Journal of Robotic Systems*, 7(2):139–144, April 1990.

[12] M. S. Branicky. *Studies in Hybrid Systems: Modeling, Analysis, and Control*. PhD thesis, Electrical Engineering and Computer Science Dept., Massachusetts Institute of Technology, `http://dora.cwru.edu/msb/pubs.html` (accessed: Dec. 15, 2008), 1995.

[13] M. S. Branicky. Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Trans. on Automatic Control*, 43(4):475–482, April 1998.

[14] D. Liberzon. *Switching in Systems and Control*. Systems and Control: Foundations and Applications. Birkhäuser, Boston, MA, USA, 2003.

[15] N. R. Gans and S. A. Hutchinson. Stable visual servoing through hybrid switched-system control. *IEEE Trans. on Robotics*, 23(3):530–540, June 2007.

[16] J. Baeten and J. De Schutter. *Integrated Visual Servoing and Force Control*, volume 8 of *Springer Tracts in Advanced Robotics*. Springer, 2004.

[17] J.-J. Borelly, E. Coste-Maniere, B. Espiau, K. Kapellos, R. Pissard-Gibollet, D. Simon, and N. Turro. The orccad architecture. *The International Journal of Robotics Research*, 17(4):338–359, April 1998.

[18] H. Bruyninckx, P. Soetens, and B. Koninckx. The real-time motion core of the orocos project. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2766–2771, Taipei, Taiwan, September 2003.

[19] S. A. Scheider, V. W. Chen, G. Pardo-Castellote, and H. H. Wang. Controlshell: A software architecture for complex electromechanical systems. *The International Journal of Robotics Research*, 17(4):360–380, April 1998.

[20] R. Cortesão, R. Koeppe, and G. Hirzinger. Data fusion for robotic assembly tasks based on human skills. *IEEE Trans. on Robotics*, 20(6):941–952, December 2004.

[21] J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx. Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *The International Journal of Robotics Research*, 26(5):433–454, May 2007.

[22] 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.

[23] T. Kröger and F. M. Wahl. On-line trajectory generation: Basic concepts for instantaneous reactions to unforeseen events. *Accepted for publishing in: IEEE Trans. on Robotics*, 2009.

[24] T. Kröger. *On-Line Trajectory Generation in Robotic Systems*, volume 58 of *Springer Tracts in Advanced Robotics*. Springer, Berlin, Heidelberg, Germany, first edition, 2009.

[25] T. Kröger, B. Finkemeyer, S. Winkelbach, S. Molkenstruck, L.-O. Eble, and F. M. Wahl. A manipulator plays Jenga. *IEEE Robotics and Automation Magazine*, 15(3):79–84, September 2008.

[26] Hasbro Inc., 1027 Newport Avenue, Mailstop A906, Pawtucket, RI 02861, USA. Jenga homepage. `http://www.jenga.com` (accessed: Dec. 15, 2008). Internet, 2008.

[27] T. Kröger, B. Finkemeyer, S. Winkelbach, S. Molkenstruck, L.-O. Eble, and F. M. Wahl. Demonstration of multi-sensor integration in industrial manipulation (poster). In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4282–4284, Orlando, FL, USA, May 2006.

[28] T. Kröger, B. Finkemeyer, S. Winkelbach, S. Molkenstruck, L.-O. Eble, and F. M. Wahl. Demonstration of multi-sensor integration in industrial manipulation (video). In *Proc. of the IEEE International Conference on Robotics and Automation*, Orlando, FL, USA, May 2006.