Haptic Rendering of Arbitrary Serial Manipulators for Robot Programming

Michael Fennel[1], Antonio Zea[1], Johannes Mangler[2], Arne Roennau[2], Uwe D. Hanebeck[1]

***Abstract*— The programming of manipulators is a common task in robotics, for which numerous solutions exist. In this work, a new programming method related to the common master-slave approach is introduced, in which the master is replaced by a digital twin created through haptic and visual rendering. To achieve this, we present an algorithm that enables the haptic rendering of any programmed robot with a serial manipulator on a general-purpose haptic interface. The results show that the proposed haptic rendering reproduces the kinematic properties of the programmed robot and directly provides the desired joint space trajectories. In addition to a stand-alone usage, we demonstrate that the proposed algorithm can be easily paired with existing visual technology for virtual and augmented reality to facilitate a highly immersive programming experience.**

I. Introduction

The programming of manipulators is a common problem in robotics and appears in many fields, including industrial automation, humanoid robots, and even surgical applications. Despite tremendous efforts to abstract and automate the programming and control of robotic manipulators, no universal approach exists that covers all possible aspects of complex problems like painting or assembly. Furthermore, increased sensing capabilities and environmental information in a machine-readable form are required, which are not always available, especially in cost-sensitive applications, complex or unstructured environments, or in situations with frequent reprogramming. To overcome these issues and to leverage the cognitive capabilities of a human operator or programmer, several programming methods are available [1]–[3].

One option is text-based programming, but this requires special knowledge and a high level of abstraction. Beyond that, it is not guaranteed that a given program with its motions can be executed correctly by the robot due to kinematic and environmental constraints. Another option is programming by demonstration. In its basic forms, the programmer moves the end effector of the programmed robot (PR) either by directly touching it or by moving the end effector of a possibly scaled master robot that is linked with the PR acting as a slave. The disadvantages of both methods are apparent: In the first case, physical interaction with the PR is required causing safety issues. In the latter case, each PR type requires its own, costly master robot with a matching kinematic structure. Another variant of programming by demonstration is the use of a marker (i.e., an object that can be tracked by the programming system) moved by the programmer [1]. Again, physical access to the robot environment is required and the resulting motion might not be compatible with the kinematic constraints of the PR. Additionally, there is no way for the user to feel interaction forces from the PR and its environment.

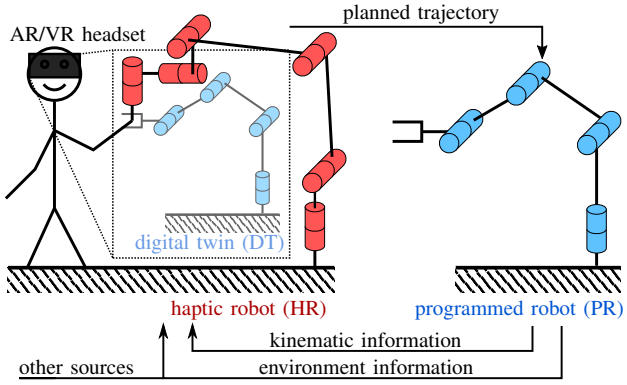


Fig. 1. General structure of the proposed digital twin setup for robot programming using haptic feedback and virtual/augmented reality.

To overcome these issues while maintaining the idea of programming by demonstration, we propose a novel digital twin (DT) system as depicted in Fig. 1. With this system, we leverage existing virtual and augmented reality (VR/AR) techniques in combination with a new method for haptic feedback for the programming of robot motions. To achieve this, the kinematic behavior of the arbitrary PR is rendered by a general-purpose haptic interface, which we will call a haptic robot (HR). With the proposed solution, the programming person will be enabled to feel and explore properties like reach, dexterous workspace, and joint limits of the PR independent of the visual rendering technology. At the same time, undesired properties such as gravity can be removed, while assistive features can be easily added. Due to the potential usage of VR/AR, it is not just possible to display the target environment and the state of the PR, but also to give the programming persons a highly immersive feeling. Ideally, they will get the impression that they are operating the PR in its real environment by guiding the end effector along desired trajectories.

II. Related Work

The application of haptic feedback for path planning or selection tasks is an active topic of research. In [4], an

[1] The authors are with the Intelligent Sensor-Actuator-Systems Laboratory, Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Adenauerring 2, 76131 Karlsruhe, Germany. E-mails: michael.fennel@kit.edu, antonio.zea@kit.edu, uwe.hanebeck@kit.edu

[2] The authors are with the FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany. E-mails: mangler@fzi.de, roennau@fzi.de

This work was supported by the ROBDEKON project of the German Federal Ministry of Education and Research.

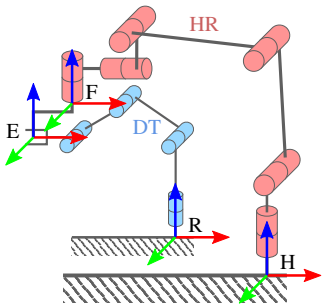


Fig. 2. Definition of the used coordinate frames.

algorithm is presented that supports the user while they choose a path, based on a pre-planned, collision-free path and guiding forces. Kinematic constraints of the PR are not respected. Another concept for haptic guidance during the path selection phase for car-like vehicles is given in [5]. Although this approach respects the kinematic constraints and the limitations of the steering angle, it cannot be applied to other mechanical structures, which disqualifies it for the task of rendering kinematics.

A more general approach for realizing constraints on a haptic interface is the so-called virtual fixture as proposed in [6]. While this approach allows constraining the motion on a path or surface, it cannot deal with complex manifolds as they appear in the workspace description of average manipulators. In [7], a method specifically designed for the haptic rendering of surfaces in 3D space is presented. Similar to virtual fixtures, this approach is limited to surfaces (i.e., two degrees of freedom) and cannot be generalized for more degrees of freedom, making it unsuitable for the haptic rendering of arbitrary manipulators.

Independent of haptic feedback, the usage of VR and AR tools in robotics grew in the last decade due to the increasing accessibility of suitable devices [8]. Examples for this are given in [9] and [10], where robot trajectories are either recorded in a fully virtual environment or in the real robot environment that is augmented with a virtual model of the actual robot. The combination of both haptics and VR/AR was demonstrated recently for the application example of welding robots [11]. Although the idea presented therein is independent of a priori environmental knowledge, it does not take into account the kinematic limitations of the robot.

As of this writing, we are not aware of any system for robot programming that combines haptic feedback with VR/AR and that respects all kinematic constraints of the PR.

III. Problem Statement

A. Coordinate Systems and Notation

Throughout this paper, vectors are printed underlined and matrices are printed bold. Positions are denoted with $\underline{p}$ and orientations are denoted using the roll-pitch-yaw angles $\underline{\theta}$. A translation and a rotation can be combined into a pose $\underline{x} = \left[\underline{p}^\mathsf{T}, \underline{\theta}^\mathsf{T}\right]^\mathsf{T}$. Orientations and poses can also be expressed as rotation matrices $\mathbf{R}$ and homogeneous transformation matrices $\mathbf{T}$, respectively. Furthermore, $\underline{f}$ is used for 3D-forces, $\underline{m}$ for 3D-torques, $\underline{q}$ for generalized joint angles, and $\underline{\tau}$ for joint torques. Superscripts denote the reference coordinate frame in which a quantity is expressed, whereas the subscripts indicate a point or a pose. For example, $\underline{p}_B^A$ represents the position of point B given in coordinates of frame A. To reference the i-th element of a vector, $[i]$ is appended to the subscript. All quantities are given in SI units unless otherwise specified.

In the remainder, the coordinate frames as illustrated in Fig. 2 are used. The frame F denotes the flange of the HR and is assumed to be coincident with the position of its force/torque sensor. The frame H denotes the reference frame of the HR. Moreover, the reference frame of the DT is defined as R and the end effector frame of the DT coincident with the programmer's handle is denoted as E.

B. Objective

In the following, the PR is assumed to be a serial kinematic structure with $n \in \mathbb{N}^+$ joints, that are a mixture of prismatic and revolute joints. As the PR might be incompatible with the size of the HR or the user, static up- or downscaling is allowed. The kinematic model of the scaled PR corresponds to the forward kinematics of the DT and is given as

$$\underline{x}_\mathrm{E}^\mathrm{R} = \underline{f}\left(\underline{q}\right) , \tag{1}$$

$$\underline{q}_\mathrm{min} \leq \underline{q} \leq \underline{q}_\mathrm{max} . \tag{2}$$

Additionally, it is assumed that the linkage, i.e., the point where the user touches the HR to move the end effector of the DT, and the position of the DT with respect to the HR are known and fixed. Formally, this can be expressed with the transformations $\mathbf{T}_\mathrm{F}^\mathrm{E}$ and $\mathbf{T}_\mathrm{R}^\mathrm{H}$.

The HR is any haptic interface, whose dexterous workspace is a superset of the DT's workspace and that accepts Cartesian pose setpoints $\underline{x}_\mathrm{F}^{\mathrm{H}*}$ for its flange. Moreover, force and torque measurements at the flange are available, i.e., $\underline{f}_\mathrm{F}^\mathrm{H}$ and $\underline{m}_\mathrm{F}^\mathrm{H}$, are known. Note, that these properties do not require a manipulator specifically designed as a haptic interface.

Given these conditions, the goal is now to develop a system, that lets the handle attached to the HR behave like the user is moving the real kinematic structure of the PR. This requires

1) a haptic rendering algorithm for a kinematic structure defined by (1) and (2), and
2) a visual rendering of the DT in AR or VR.

This paper mainly deals with the former, whereas the visual rendering is covered only briefly. Interested readers are therefore invited to refer to [12] for detailed information about the visual rendering.

IV. Haptic Rendering of Manipulators

In the following, the essential parts of the proposed haptic rendering algorithm are presented.

A. Transformation of Forces and Torques

Force and torque are measured with respect to the reference frame of the HR. In order to use these measurements, the reference frame needs to be changed first, using

$$\begin{bmatrix} \underline{f}_\mathrm{F}^\mathrm{R} \\ \underline{m}_\mathrm{F}^\mathrm{R} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_\mathrm{H}^\mathrm{R} \, \underline{f}_\mathrm{F}^\mathrm{H} \\ \mathbf{R}_\mathrm{H}^\mathrm{R} \, \underline{m}_\mathrm{F}^\mathrm{H} \end{bmatrix} . \tag{3}$$

Based on this, the force acting on E,

$$\begin{bmatrix} \underline{f}_{\mathrm{E}}^{\mathrm{R}} \\ \underline{m}_{\mathrm{E}}^{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} \underline{f}_{\mathrm{F}}^{\mathrm{R}} \\ \underline{m}_{\mathrm{F}}^{\mathrm{R}} + (\mathbf{R}_{\mathrm{E}}^{\mathrm{R}}\, \underline{p}_{\mathrm{F}}^{\mathrm{E}}) \times \underline{f}_{\mathrm{F}}^{\mathrm{R}} \end{bmatrix}, \tag{4}$$

must be calculated. This expression assumes that E and F are rigidly linked and that the mass of the handle is negligible or compensated for by the force-torque-sensing method.

B. Simulation of Dynamics

To simulate the behavior of the DT under the influence of the programmer's grasp, a dynamic system model of the DT is used. The key idea here is to calculate the joint movements based on the forces and torques acting on the end effector of the DT. In contrast to standard physic engines, that model each link as a body with 6 degrees of freedom and each joint as a constraint [13], the presented approach performs all calculations directly in the joint space. The simulated joint angles are then forwarded to the HR (see Section IV-C), resulting in a structure similar to an admittance control scheme.

It is well-known from robot dynamics, that

$$\underline{\tau} = \mathbf{H}\left(\underline{q}\right) \ddot{\underline{q}} + \underline{c}\left(\underline{q}, \dot{\underline{q}}\right) + \underline{g}\left(\underline{q}\right) \tag{5}$$

holds, where $\mathbf{H}$ represents the joint space inertia matrix, $\underline{c}$ the torques occurring due to centrifugal and coriolis forces, and $\underline{g}$ the torques induced by gravity. The torque $\underline{\tau}$ can be split into

$$\underline{\tau} = \underline{\tau}_{\mathrm{dri}} + \underline{\tau}_{\mathrm{con}} - \underline{\tau}_{\mathrm{dis}}\,, \tag{6}$$

with the driving torques $\underline{\tau}_{\mathrm{dri}}$, the constraint torques $\underline{\tau}_{\mathrm{con}}$, and the dissipative torques $\underline{\tau}_{\mathrm{dis}}$. To make the DT compliant to the forces and torques applied to its end effector, the driving torques are determined according to [14] by

$$\underline{\tau}_{\mathrm{dri}} = \mathbf{J}^{\mathsf{T}}\left(\underline{q}\right) \begin{bmatrix} \underline{f}_{\mathrm{E}}^{\mathrm{R}} \\ \underline{m}_{\mathrm{E}}^{\mathrm{R}} \end{bmatrix}, \tag{7}$$

where $\mathbf{J}$ is the Jacobi-matrix of the kinematics defined in (1). Other driving torques, e.g., due to joint actuation, are set to zero. For a more pleasant user experience, $\underline{g}$ is set to zero as well. This means that the user does not have to hold the static weight of the DT.

Now, solving (5) for $\ddot{\underline{q}}$ and inserting (6) and (7) yields

$$\begin{bmatrix} \dot{\underline{q}} \\ \ddot{\underline{q}} \end{bmatrix} = \begin{bmatrix} \dot{\underline{q}} \\ \mathbf{H}^{-1}\left(\underline{q}\right) \left(\mathbf{J}^{\mathsf{T}}\left(\underline{q}\right) \begin{bmatrix} \underline{f}_{\mathrm{E}}^{\mathrm{R}} \\ \underline{m}_{\mathrm{E}}^{\mathrm{R}} \end{bmatrix} + \underline{\tau}_{\mathrm{con}} - \underline{\tau}_{\mathrm{dis}} - \underline{c}\left(\underline{q}, \dot{\underline{q}}\right) \right) \end{bmatrix} \tag{8}$$

for the momentary joint speeds and accelerations. In this system of differential equations, $\mathbf{H}$ and $\underline{c}$ are determined using the composite-rigid-body algorithm (CRBA) and the recursive Newton-Euler algorithm (RNEA), respectively. After that, an explicit integration scheme is used to obtain the desired joint positions. Implicit integration schemes are not applicable, as the mentioned algorithms do not provide gradient information with respect to $\underline{q}$ and $\dot{\underline{q}}$.

1) Inertia Matrix: The joint space inertia matrix $\mathbf{H}\left(\underline{q}\right)$ reflects all masses that are present in the DT and its real counterpart. Consequently, all link inertia values must be known for the matrix calculation. Simply taking the physical link inertia values of the PR is a conceivable option for this purpose, but it has two drawbacks: First, precise inertia data is not always available, and second, the inertia of a real robot arm might be higher than that which a user can handle comfortably. For this reason, a new artificial inertia distribution is constructed, which concentrates the majority of the user-configurable inertia (m_{main} and I_{main}) in the end effector E. Additionally, all other links are assigned small inertia values (m_{other} and I_{other}) to avoid ill-conditioned, and hence non-invertible, $\mathbf{H}$-matrices. This way, the felt inertia is concentrated at the user's hand and can be tuned to a comfortable value independent of the real inertia.

2) Friction: It is desired that a passively moving manipulator reaches a resting state after some time without any driving forces and torques, as is the case with any real manipulator. This can be achieved by introducing friction into the model defined in (8), which also permits an effective limitation of the end effector speeds assuming that the force and torque exerted by the user are limited in magnitude.

To realize the friction, $\underline{\tau}_{\mathrm{dis}}$ is set to the sum of two terms. The first component is the viscous joint friction

$$\underline{\tau}_{\mathrm{dis,jnt}} = d_{\mathrm{jnt}}\, \dot{\underline{q}}, \tag{9}$$

with friction coefficient $\underline{\tau}_{\mathrm{dis,jnt}}$ and the second component is based on a Cartesian, isotropically acting viscous friction defined by

$$\begin{bmatrix} \underline{f}_{\mathrm{E}}^{\mathrm{R}} \\ \underline{m}_{\mathrm{E}}^{\mathrm{R}} \end{bmatrix}_{\mathrm{dis,cart}} = \begin{bmatrix} d_{\mathrm{cart},p}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & d_{\mathrm{cart},\theta}\mathbf{I} \end{bmatrix} \dot{\underline{x}}_{\mathrm{E}}^{\mathrm{R}}\,, \tag{10}$$

where $\dot{\underline{x}}_{\mathrm{E}}^{\mathrm{R}} = \mathbf{J}(q)\, \dot{\underline{q}}$. The parameters $d_{\mathrm{cart},p}$ and $d_{\mathrm{cart},\theta}$ characterize the translational and rotational friction, respectively. Analogous to (7), this yields the dissipative joint torques

$$\underline{\tau}_{\mathrm{dis,cart}} = \mathbf{J}^{\mathsf{T}}\left(\underline{q}\right) \begin{bmatrix} d_{\mathrm{cart},p}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & d_{\mathrm{cart},\theta}\mathbf{I} \end{bmatrix} \mathbf{J}(q)\, \dot{\underline{q}}\,. \tag{11}$$

Both friction components are required for a safe and comfortable user experience. While the former is mainly meant for damping high joint speeds near singularities, the latter one ensures that the damping force felt by the user is not dependent on the link lengths of the PR as long as the kinematic properties are respected.

3) Joint Limits: Robotic manipulators usually have a limited range of joint motion, which restricts their working range and introduces properties of a non-trivially solvable hybrid system. To incorporate this into the presented haptic rendering algorithm, joint limits must be mimicked as well. A simple way to achieve this is the modeling of each active joint limit as a virtual spring. However, this method results in indistinct joint limits and stiff differential equations that facilitate oscillations.

For this reason, joint limits are interpreted as contacts between rigid bodies, which allows the application of the contact modeling described in [15]. The idea behind this

approach is to use the yet unknown constraint torques $\underline{\tau}_{\text{con}}$ introduced in (6) and to summarize all known quantities in (5) as

$$\hat{\underline{\tau}} = \underline{\tau}_{\text{dri}} - \underline{\tau}_{\text{dis}} - \underline{c}\left(\underline{q}, \dot{\underline{q}}\right) \tag{12}$$

yielding

$$\mathbf{H}\ddot{\underline{q}} = \hat{\underline{\tau}} + \underline{\tau}_{\text{con}}. \tag{13}$$

For the modeling of the contacts, let m be the number of joint limits that are currently active and $l\,(i) \in \{1, \ldots, n\}$ the joint number of the i-th joint at its limit. Furthermore, the sign of the joint limit is defined as

$$s\,(i) = \begin{cases} -1 & \text{if } \underline{q}_{[l(i)]} \geq \underline{q}_{\max[i]} \\ +1 & \text{if } \underline{q}_{[l(i)]} \leq \underline{q}_{\min[i]} \end{cases}. \tag{14}$$

Now, the constraint torques can be substituted with

$$\underline{\tau}_{\text{con}} = \mathbf{F}\underline{\alpha}\,, \tag{15}$$

where $\mathbf{F} \in \mathbb{R}^{n \times m}$ is a directional matrix given as

$$\mathbf{F} = \begin{bmatrix} s\,(1)\,\underline{e}_{l(1)} & \cdots & s\,(m)\,\underline{e}_{l(m)} \end{bmatrix}. \tag{16}$$

Here, $\underline{e}_k$ denotes the k-th unit vector. The vector $\underline{\alpha} \in \mathbb{R}^m$ contains the yet unknown constraint torque coefficients, which can be interpreted as the sought constraint torque magnitudes, since all columns of $\mathbf{F}$ are orthonormal. For each contact i,

$$\left(\underline{\alpha}_{[i]} = 0 \wedge s(i)\underline{e}_{l(i)}^{\mathsf{T}}\ddot{\underline{q}} > 0\right) \vee \left(\underline{\alpha}_{[i]} \geq 0 \wedge s(i)\underline{e}_{l(i)}^{\mathsf{T}}\ddot{\underline{q}} = 0\right) \tag{17}$$

must hold. This boolean expression states, that the currently active joint limit either becomes inactive (i.e., the constraint torque is zero and the joint is accelerating away from the limit) or remains active (i.e., no acceleration, but a constraint torque). If these conditions are reformulated and merged for all contacts, the vector notation

$$\left(\text{diag}\left(\underline{\alpha}\right)\mathbf{F}^{\mathsf{T}}\ddot{\underline{q}} = 0\right) \wedge \left(\underline{\alpha} + \mathbf{F}^{\mathsf{T}}\ddot{\underline{q}} \geq 0\right) \tag{18}$$

is obtained. Inserting (13) and (15) into the previous expression yields the quadratic equality/inequality system

$$\begin{aligned} &\left(\text{diag}\left(\underline{\alpha}\right)\mathbf{F}^{\mathsf{T}}\mathbf{H}^{-1}\left(\hat{\underline{\tau}} + \mathbf{F}\underline{\alpha}\right) = 0\right) \quad \wedge \\ &\left(\underline{\alpha} + \mathbf{F}^{\mathsf{T}}\mathbf{H}^{-1}\left(\hat{\underline{\tau}} + \mathbf{F}\underline{\alpha}\right) \geq 0\right), \end{aligned} \tag{19}$$

which can be solved for $\underline{\alpha}$ using the root finding algorithm given in [15]. The eventually calculated value for $\underline{\tau}_{\text{con}}$ is then fed back into (8) before integration.

Although the above-mentioned steps ensure that joint limits are always respected, it does not guarantee that the mechanical impulse of the DT is preserved, when physically possible. To avoid this uncomfortable effect for the user, the impact model between rigid bodies during the compression phase [15] is applied before each integration step. In this, the sought change in joint velocity $\Delta\dot{\underline{q}}$ is physically described through

$$\Delta\dot{\underline{q}} = \mathbf{H}^{-1}\mathbf{F}\underline{\gamma}\,, \tag{20}$$

where $\underline{\gamma} \in \mathbb{R}^m$ can be interpreted as a yet unknown impulse strength. Similar to (17),

$$\begin{aligned} &\left(\underline{\gamma}_{[i]} = 0 \;\wedge\; s(i)\underline{e}_{l(i)}^{\mathsf{T}}\left(\dot{\underline{q}} + \Delta\dot{\underline{q}}\right) > 0\right) \quad \vee \\ &\left(\underline{\gamma}_{[i]} \geq 0 \;\wedge\; s(i)\underline{e}_{l(i)}^{\mathsf{T}}\left(\dot{\underline{q}} + \Delta\dot{\underline{q}}\right) = 0\right) \end{aligned} \tag{21}$$

Algorithm 1 Simulation of the DT's dynamic behavior for the haptic rendering.

procedure UPDATE($\Delta t, \underline{f}_{\text{E}}^{\text{R}}, \underline{m}_{\text{E}}^{\text{R}}$)
  $\mathbf{H} \leftarrow \text{getInertiaMatrix}\left(\text{model}, \underline{q}\right)$ ▷ CRBA
  $\Delta\dot{\underline{q}} \leftarrow \text{calculateVelocityDelta}\left(\mathbf{F}, \mathbf{H}, \dot{\underline{q}}\right)$ ▷ (20), (22)
  $\dot{\underline{q}} \leftarrow \dot{\underline{q}} + \Delta\dot{\underline{q}}$
  $\mathbf{J} \leftarrow \text{getJacobian}\left(\text{model}, \underline{q}\right)$
  $\underline{c} \leftarrow \text{getCoriolisTerm}\left(\text{model}, \underline{q}, \dot{\underline{q}}\right)$ ▷ RNEA
  $\underline{\tau}_{\text{dri}} \leftarrow \mathbf{J}^{\mathsf{T}}\left[\underline{f}_{\text{E}}^{\text{R}\,\mathsf{T}}, \underline{m}_{\text{E}}^{\text{R}\,\mathsf{T}}\right]^{\mathsf{T}}$
  $\underline{\tau}_{\text{dis}} \leftarrow \underline{\tau}_{\text{dis,jnt}} + \underline{\tau}_{\text{dis,cart}}$ ▷ (9), (11)
  $\hat{\underline{\tau}} \leftarrow \underline{\tau}_{\text{dri}} - \underline{\tau}_{\text{dis}} - \underline{c}$
  $\underline{\tau}_{\text{con}} \leftarrow \text{calculateConstrTorques}(\mathbf{F}, \mathbf{H}, \hat{\underline{\tau}})$ ▷ (15), (19)
  $\ddot{\underline{q}} \leftarrow \mathbf{H}^{-1}\left(\hat{\underline{\tau}} + \underline{\tau}_{\text{con}}\right)$
  $\left(\underline{q}, \dot{\underline{q}}\right) \leftarrow \text{integrate}\left(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}, \Delta t\right)$
  $\mathbf{F} \leftarrow \text{calculateFMatrix}\left(\text{model}, \underline{q}\right)$ ▷ (14), (16)
  $\underline{q} \leftarrow \max\left(\min\left(\underline{q}, \text{model}.\underline{q}_{\max}\right), \text{model}.\underline{q}_{\min}\right)$
end procedure

has to be true. Here, the former condition represents a contact that is in the process of breaking (i.e., the new velocity causes the joint to move away from its active limit) and the latter condition represents a contact that is still active (i.e., new velocity must be zero). Combining (20) and (21) for all contacts yields the system

$$\begin{aligned} &\left(\text{diag}\left(\underline{\gamma}\right)\mathbf{F}^{\mathsf{T}}\left(\dot{\underline{q}} + \mathbf{H}^{-1}\mathbf{F}\underline{\gamma}\right) = 0\right) \quad \wedge \\ &\left(\underline{\gamma} + \mathbf{F}^{\mathsf{T}}\left(\dot{\underline{q}} + \mathbf{H}^{-1}\mathbf{F}\underline{\gamma}\right) \geq 0\right), \end{aligned} \tag{22}$$

whose solution scheme for $\underline{\gamma}$ is the same as that for (19). Following this, $\Delta\dot{\underline{q}}$ is obtained using (20) and added to $\dot{\underline{q}}$.

With this, all steps involved in the dynamic simulation of the DT and thus for the haptic rendering are known. If the operations are now executed as outlined in Algorithm 1, a full cycle of the dynamic simulation is performed.

C. Output Preparation

After each simulation cycle (i.e., integration step of differential equation (8)), $\underline{q}$ is passed to the forward kinematics (1) of the DT. The resulting pose $\underline{x}_{\text{E}}^{\text{R}}$ is then transformed to the desired flange pose of the HR with respect to its reference frame using

$$\mathbf{T}_{\text{F}}^{\text{H}*} = \mathbf{T}_{\text{R}}^{\text{H}}\,\mathbf{T}_{\text{E}}^{\text{R}}\,\mathbf{T}_{\text{F}}^{\text{E}}\,, \tag{23}$$

which is then sent to the HR for execution.

V. Implementation

A block diagram of the proposed haptic rendering method is depicted in Fig. 3. The loop closure is achieved through the user's hand-arm-system acting as a variable mechanical impedance.

For validation, the whole system was implemented as a *ROS*-node in C++. To ensure reusability, the code is kept robot-agnostic with respect to the PR and the HR. Therefore, the HR is commanded via Cartesian pose-setpoints and the entire geometry and joint information of the PR is provided through an *URDF*-based robot description, that can be downscaled if necessary. The implementations of the CRBA and the RNEA as well as the kinematic operations are taken from the *Orocos*

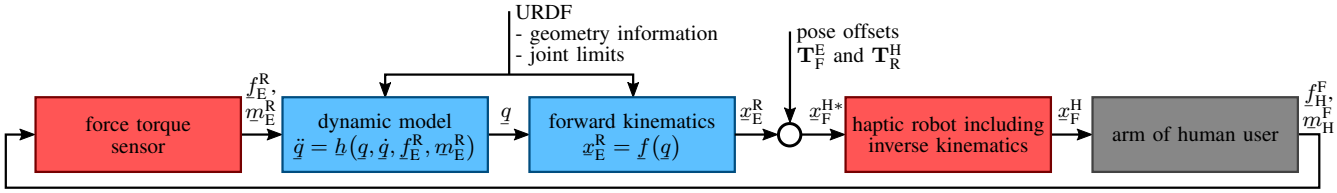
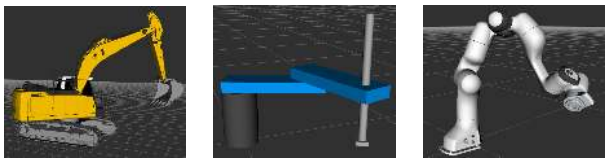


Fig. 3. Block diagram of the proposed haptic rendering method for serial manipulators. Blocks drawn in red correspond to the HR, while blocks drawn in blue correspond to the DT.



(a) Liebherr excavator (4 joints).

(b) SCARA (4 joints) with prismatic joint.

(c) Franka Emika Panda (7 joints).

Fig. 4. Robot models used as PR during the evaluation. The joints are numbered ascending from the base to the tip for each robot.

Kinematics Dynamics Library (*KDL*) [16]. For the integration of (8), the fourth-order Runge-Kutta method was used.

For the visual rendering of the DT and other information such as forces and environmental information, the iviz visualization platform [12] is deployed. By leveraging the Unity Game Engine and a high-performance ROS interface, iviz enables effortless robotic visualization on a variety of AR/VR devices, including systems running Apple iOS, Microsoft Windows, and Google Android. Due to the full compatibility with ROS and the URDF-format, iviz requires zero porting effort for existing ROS applications and has a function range that is comparable with the ROS-tool rviz.

VI. Experimental Results

For the evaluation, a *Universal Robots UR16e* manipulator, which can be considered as a low-end haptic device, was used as an HR. This choice was made to demonstrate that the presented algorithms do not rely on the availability of special haptic interfaces. The Cartesian pose control of the UR16e was realized using the inverse kinematics algorithm presented in [17]. The control loops were set to a frequency of 500 Hz running on a laptop with an *Intel Core i7-9570H* CPU. To demonstrate the capabilities of the proposed haptic rendering method, a set of different PRs as depicted in Fig. 4 was selected. The chosen manipulators include a variety of kinematic structures reaching from classical industrial to more exotic manipulators. In addition, prismatic and revolute joints are included as well as a redundant kinematic structure.

For all experiments, the friction parameters were chosen empirically as $d_{\text{jnt}} = 0.8\,\text{N m s}$, $d_{\text{cart},p} = 17.0\,\text{N s m}^{-1}$, and $d_{\text{cart},\theta} = 1.5\,\text{N m s}$. The inertia values were set to $m_{\text{main}} = 30.0\,\text{kg}$, $I_{\text{main}} = 0.03\,\text{kg m}^2$, $m_{\text{other}} = 6.0\,\text{kg}$, and $I_{\text{other}} = 6.0 \times 10^{-3}\,\text{kg m}^2$. These values result in a user experience with reasonable required user forces without being susceptible to force/torque sensor noise or exceeding the driving capabilities of the actuators.

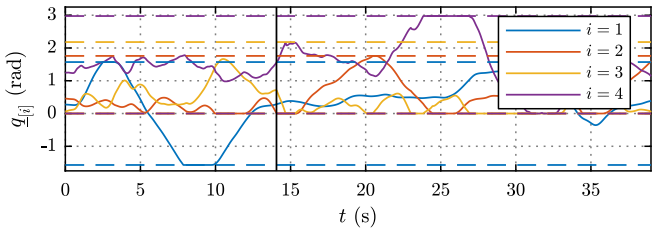


Fig. 5. Recorded joint space trajectories of the excavator's DT. The upper and lower joint limits are drawn as dashed lines. The black line marks the point in time when a velocity impulse is applied to joint 3.

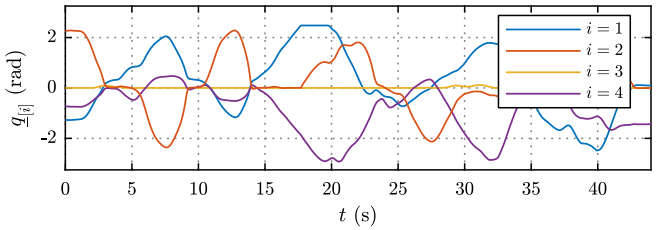


Fig. 6. Joint space trajectories of the SCARA manipulator's DT, when singularities ($\underline{q}_{[2]} = 0$) are passed.

A. Joint Limits

In order to test the behavior of the DT's joint limits, the manipulator of an excavator as shown in Fig. 4(a) was used as the PR. Therefore, the end effector of the DT, i.e., the shovel, was moved around by hand to hit the joint limits on purpose. The resulting joint space trajectories are depicted in Fig. 5. Clearly, it can be seen that all joint limits are respected without oscillations at the boundaries, independent of the number of joints at the limit. At $t = 14\,\text{s}$, joint 2 (boom) reaches its lower limit causing a sudden change in the velocity of joint 3. This demonstrates that the impulse preservation described in Section IV-B.3 works as expected. In general, the obtained joint trajectories are ready to be processed for validation or to be directly sent to the executing robot.

B. Singularities

The SCARA arm from Fig. 4(b) features a singularity if it is stretched to its full extent ($\underline{q}_{[2]} = 0$). As illustrated in Fig. 6, this singularity neither restricts the movements of the remaining joints nor the proposed rendering algorithm in general. The manipulator can be moved into and out of singularities at any time if suitable forces are applied as one might expect from the real mechanical structure.

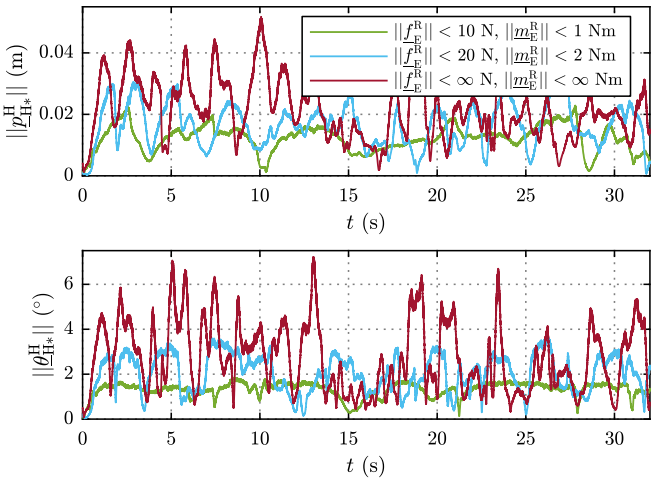


Fig. 7. Translational and rotational tracking error between HR and DT for different maximal forces and torques. The ∞-symbol in the legend indicates, when the maximal forces and torques are not bounded.

C. Tracking Error

For the goal of accurate robot programming, it is crucial that the HR closely follows the DT and vice versa. As explained in Section IV-C, the pose of the DT is used as an input for the Cartesian controller of the HR. This implies that the achievable tracking error depends on the controller properties of the HR. To examine the accuracy, the tracking errors for the translation $\underline{p}_{\mathrm{H}*}^{\mathrm{H}}$ and the rotation $\underline{\theta}_{\mathrm{H}*}^{\mathrm{H}}$ were evaluated for different maximal forces and torques exerted by the user, when the robot in Fig. 4(c) was moved around randomly. The results depicted in Fig. 7 indicate that the expected errors are bound. From this, it also becomes apparent that lower errors can be achieved by limiting the force/torque artificially without changing the HR or its control.

D. Integration with VR/AR

Due to the implementation as a ROS node, seamless interoperability with iviz is enabled. This was successfully tested through the haptic rendering and visualization of the manipulators shown in Fig. 4. An example for the visualization, that was performed with the HTC Vive (VR), the Apple iPad (AR) as well as the Microsoft Hololens (AR) can be found in Fig. 8. We refer the reader to the supplementary material of this paper for more examples.[1]

VII. Conclusions

In this paper, a haptic rendering algorithm suitable for the programming of serial manipulators was presented. Practical tests demonstrated that the presented rendering method provides a feasible solution for the joint space trajectories of the PR, independent of the presence of singularities or the kinematic structure. Furthermore, it was shown that the achievable tracking accuracy mainly depends on the controller of the HR and the magnitude of input force and torque. The overall system is well integrated with VR/AR methods.

Future research will involve the integration of an HR with a faster Cartesian controller, the application in a real-world programming scenario, and the design of assistive features such as the display of forces due to collisions between the DT and its environment. Additionally, the presented programming approach will be evaluated with a group of test subjects. At the theoretical level, the stability of the system during human-robot interaction needs to be studied. Especially the observation, that a limited force input yields a bounded tracking error, which would imply stability, needs to be proven mathematically.

[1] https://youtu.be/5t37cfeE7E0

Fig. 8. Iviz visualization of the programming process for an excavator. The robot model is rendered in real-time onto the camera image of an Apple iPad.

References

[1] G. Biggs and B. MacDonald, "A survey of robot programming systems," in *Proceedings of the Australasian conference on robotics and automation*, 2003.

[2] G. F. Rossano, C. Martinez, M. Hedelind, S. Murphy, and T. A. Fuhlbrigge, "Easy robot programming concepts: An industrial perspective," in *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, 2013, pp. 1119–1126.

[3] V. Villani, F. Pini, F. Leali, C. Secchi, and C. Fantuzzi, "Survey on human-robot interaction for robot programming in industrial applications," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 66–71, 2018.

[4] N. Ladeveze, J. Y. Fourquet, B. Puel, and M. Taïx, "Haptic assembly and disassembly task assistance using interactive path planning," in *2009 IEEE Virtual Reality Conference*, 2009, pp. 19–25.

[5] M. Fennel, A. Zea, and U. D. Hanebeck, "Haptic-guided path generation for remote car-like vehicles," *IEEE Robotics and Automation Letters*, 2021, to appear.

[6] P. Marayong, M. Li, A. M. Okamura, and G. D. Hager, "Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures," in *2003 IEEE International Conference on Robotics and Automation*, vol. 2, 2003, pp. 1954–1959.

[7] N. Zafer, "Constraint-based haptic rendering of a parametric surface," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 221, no. 3, pp. 507–517, 2007.

[8] Z. Makhataeva and H. A. Varol, "Augmented reality for robotics: A review," *Robotics*, vol. 9, no. 2, 2020.

[9] H. Fang, S. K. Ong, and A. Y. Nee, "Robot programming using augmented reality," in *2009 International Conference on CyberWorlds*, 2009, pp. 13–20.

[10] A. Burghardt, D. Szybicki, P. Gierlak, K. Kurc, P. Pietruś, and R. Cygan, "Programming of industrial robots using virtual reality and digital twins," *Applied Sciences*, vol. 10, no. 2, 2020.

[11] D. Ni, A. W. W. Yew, S. K. Ong, and A. Y. C. Nee, "Haptic and visual augmented reality interface for programming welding robots," *Advances in Manufacturing*, vol. 5, no. 3, pp. 191–198, 2017.

[12] A. Zea and U. D. Hanebeck, "iviz: A ROS visualization app for mobile devices," *Software Impacts*, vol. 8, 2021.

[13] D. Baraff, "Physically based modeling," in *SIGGRAPH 99 course notes*, 1999.

[14] R. Featherstone and D. E. Orin, "Dynamics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2008, pp. 35–65.

[15] R. Featherstone, *Robot dynamics algorithms*. Boston, Dordrecht, Lancester: Kluwer Academic Publishers, 1987.
[16] P. Soetens, T. Issaris, H. Bruyninckx, S. Joyeux, R. Smits *et al.* (2021, Feb.) KDL overview – Orocos documentation. [Online]. Available: https://docs.orocos.org/kdl/overview.html
[17] S. Scherzinger, A. Roennau, and R. Dillmann, "Inverse kinematics with forward dynamics solvers for sampled motion tracking," in *2019 19th International Conference on Advanced Robotics (ICAR)*, 2019, pp. 681–687.