Experimental Validation of Reference Spreading by Robotic Manipulation of Unmodeled Objects

Gijs van den Brandt

Abstract— **Exploiting impacts can speed up robotic object manipulation. However, when using traditional control approaches, impacts are paired with peaks in joint torques and contact forces that may damage the robot or its environment. Reference spreading (RS) is an impact-aware control approach that addresses these peaks without significantly compromising on tracking performance. Numerical studies have validated the benefits of RS controllers for object manipulation tasks, but these controllers rely on knowledge regarding the robot's contact state and accurate object models – information that is not easily available in a real-life setup.**

To facilitate experimental validation of RS for object manipulation, this work describes an impact detection algorithm and a model-free trajectory planner using teleoperation. Furthermore, a soft end effector is designed. These components are implemented on a dual-arm robotic setup to evaluate RS in a usecase that is relevant to industry, namely grabbing a box. Experiments show that RS reduces control effort – and therefore joint torques and contact forces – without inhibiting tracking performance. Consequently, impacts at higher velocities become feasible, meaning that RS can contribute to faster object manipulation.

I. INTRODUCTION

Automation has historically played a crucial role in the logistics industry. Our current way of living depends on autonomous systems for global transportation and warehousing. The growing labor shortage and increasing demand for online retail motivate further developments in the logistics sector [1].

A logistical aspect where machines struggle to fully compete with humans is object manipulation. A practical examples of this is depalletizing. While robots are strong and consistent when manipulating objects, humans are versatile and swift. Robots are held back from faster performance because they must often slow down prior to making contact; establishing contact at a high velocity – an event referred to as an impact – could cause damage to the robot or its environment. On the contrary, humans intrinsically exploit impacts in the form of grabbing, bouncing and hitting.

The field of impact-aware control aims to better equip robots for making contact at high velocities. These impacts are paired with large contact forces that could damage the system. Previous work describes control using the maximum allowable impact velocity that complies with safety constraints such as limits for the contact force [2, 3]. This was combined with a compliant cover for the robot that reduces contact forces at impact, facilitating higher feasible impact velocities. Rather than using a soft cover, low impact forces may also be achieved by designing a robot with low inertia and high backdrivability as was done in [4].

In addition to the large contact forces, a subject of interest is the velocity jump at the time of impact. Time misalignments between velocity jumps in the reference and in the actual system cause the velocity tracking error to peak[5]. This error peak results in undesired control effort and consequently increased joint torques and contact forces, which may damage the robot or its environment. Increased control effort due to error peaking should thus be avoided.

In [6], the robot's velocities are projected into an impact-invariant subspace based on the expected point of impact. As a result, impact-driven peaks in the velocity tracking error are reduced significantly. It is not always possible to describe a point of impact, however. Often times, impacts occur between surfaces rather than just points. Furthermore, corners of the surface may impact at diverging intervals in uncertain order during what is called near-simultaneous impacts.

The impact-aware control scheme called Reference Spreading [7] also adresses error peaking caused by misaligned impacts. It operates on the basis of a tracking error that switches from an ante- to a post impact reference once an impact is detected. Near-simultaneous impacts can be accounted for by an intermediate impact mode as was done in [8]. By addressing the peaking error, reference spreading facilitates faster object manipulation, making it interesting to industry if its effectivity can be proven in practice.

Experimental validations of reference spreading have been limited to interaction with a fixed environment [9, 10]. In numerical studies, it has already been shown that reference spreading can also benefit object manipulation [8, 11]. These works use models of the environment to formulate a trajectory with impact-driven velocity jumps that are coherent with the system's dynamics. A lack of sufficiently accurate object models in the real world means that a different approach to trajectory formulation is required. Furthermore, the controllers used in simulation rely on information, such as contact state or object location, that often is not available on an experimental setup.

The goal of this work is to **evaluate reference spreading for practical object maniulation tasks on an experimental setup.** To this end, a soft end effector is designed to increase friction during object manipulation. This end effector also reduces peaks in contact forces upon impact. Furthermore, since the lack of acurate environment models does not allow for model-based trajectory planning, a motion planner that utilizes teleoperation is formulated. To monitor the contact state using sensor data, a novel impact detector is described

and compared to impact detectors from literature that are prone to false-positives. Finally, these components are implemend with a reference spreading controller on a dual-arm setup. A box-grabbing experiment is then conducted to compare control approaches with and without reference spreading.

II. Background

A. Robots suitable for impacts

goal: Explain why franka emika robot is suitable for impacts, explain torque control, show custom end effectors from literature, motivate design of a new end effector

Traditionally, robots employ low-level controllers which govern a motor torque based on a joint position or velocity reference, encoder measurements, and gain parameters.

B. Quadratic programing control

goal: motivate QP control, formulate carthesian impedance task, posture task, and list dynamics of motion and safety constraints

Controlling a robot for object manipulation involves, e.g., adressing redundancy and acounting for limits of the robot, while simultaneously applying a contact force and moving to a position. These objectives often clash with each other, meaning that they cannot always be perfectly satisfied. It is possible to distinguish between constraints – control objectives that must always be satisfied – and tasks – objectives that should be performed as good as possible but not necesarily perfectly.

Quadratic Programming (QP) relies on this philosophy. In a QP scheme, constraints and task errors are defined as a linear function of optimization parameters. Then it becomes possible to efficiently find the parameters that minmize the task errors while adhering to the constraints. Furthermore, priority between the tasks can be achieved by assigning weights. In the context of a robot where the optimization parameters are, for example, joint accelerations $\ddot{q}$, the optimization problem can be formulated as

$$\min_{\ddot{q}\in\chi} \sum_i w_i \|e_i(\ddot{q})\|^2 \tag{1}$$

where w_i and e_i are the respective weight and error of the task with index i, and χ is the feasible space where all constraints are met. The remainder of this section describes constraints and tasks that are relevant for object manipulation.

Torque constraint: To protect the robot, we can define a constraint that limits the control torque τ within a safe range, i.e.,

$$\tau_{min} \leq \tau \leq \tau_{max} \tag{2}$$

This constraint must be formulated as a function of optimization parameter $\ddot{q}$ to be compatible with the QP scheme. The relation between $\ddot{q}$ and τ is given by the equations of motion

$$M\ddot{q} + h = \tau \tag{3}$$

with inertia matrix M and vector of gravity, centrifugal, and Coriolis terms h. Subsequently, the torque constraint formulated as linear function of the optimization parameter becomes

$$\tau_{min} \leq M\ddot{q} + h \leq \tau_{max}. \tag{4}$$

Impedance task: When manipulating objects, both the position and the contact force of the robot should be controlled. Impedance relates displacement to force, meaning that enforcing a desired impedance is a suitable control approach for object manipulation.

Rather than enforcing a virtual impedance at each joint, it is preferable to apply a virtual impedance in Carthesian space, which is more intuitive when tuning task weights or when performing teleoperation. The impedance should then act on the carthesian pose of a body, i.e., its translational coordinates p and rotation R. Velocity and accelerations are represented by ω and α for rotations, and by v and a for translations.

An impedance controller aims to achieve the closed loop behaviour

$$\Lambda \begin{bmatrix} \alpha \\ a \end{bmatrix} + D \begin{bmatrix} \omega - \omega_d \\ v - v_d \end{bmatrix} + K \begin{bmatrix} -e_{rot} \\ p - p_d \end{bmatrix} = f_d \tag{5}$$

with Carthesian-space inertia, damping and stiffness matrices Λ, D and K, and rotation tracking error e_{rot} as defined in Appendix xxx. Subscript d denotes the desired value, and f represents the external wrench.

Stiffness K is typically chosen as a diagonal matrix. The damping matrix is determined following $D = 2(\Lambda K)^{\frac{1}{2}}$ which guarantees stable behavior when K and Λ are symmetric [12]. Furthermore, for the inertia matrix, two options are considered. Choosing a diagonal matrix Λ decouples the accelerations w.r.t. to the position error, resulting in better position tracking in free motion. This approach was used in [13]. In this work, however, the task-space inertia is set to match the joint-space inertia following $\Lambda^{-1} = JM^{-1}J^T$, decoupling the contact force w.r.t. the position error for better performance during contact.

We can define the target task-space accelerations α_t and a_t based on (5):

$$\begin{bmatrix} \alpha_t \\ a_t \end{bmatrix} = \Lambda^{-1} \left(D \begin{bmatrix} \omega_d - \omega \\ v_d - v \end{bmatrix} + K \begin{bmatrix} e_{rot} \\ p_d - p \end{bmatrix} + f_d \right) \tag{6}$$

The desired behaviour of (5) is achieved as the robot's acceleration approaches the target acceleration, i.e., we want to minimize the impedance task error e_{imp} given by

$$e_{imp} = \begin{bmatrix} \alpha_t \\ a_t \end{bmatrix} - \begin{bmatrix} \alpha \\ a \end{bmatrix} \tag{7}$$

The QP scheme requires the task to be formulated as a linear function of the optimization parameter $\ddot{q}$. The relation between the task-space and joint-space acceleration can be found using the Jacobian J, which adheres to

$$\begin{bmatrix} \omega \\ v \end{bmatrix} = J\dot{q}. \tag{8}$$

Deriviating with respect to time gives

$$\begin{bmatrix} \alpha \\ a \end{bmatrix} = J\ddot{q} + \dot{J}\dot{q}. \tag{9}$$

Finally, combining (7) and (9) results in the impedance task error as a function of the optimization variable:

$$e_{imp}(\ddot{q}) = \begin{bmatrix} \alpha_t \\ a_t \end{bmatrix} - J\ddot{q} - \dot{J}\dot{q}. \tag{10}$$

Posture task: In object manipulation, the 6-DoF end effector pose is of particular interest. Robot arms that mimic human anatomy often posses 7 DoF's, however. This results in a DoF redundancy, which should be addressed for a well-posed QP scheme. The additional DoF can be imposed by a so-called posture task, as was done in [13]. This task attaches a virtual, critically-damped spring and damper with stiffness k to the joint angles, i.e.,

$$\ddot{q}_t = k(q_d - q) - 2\sqrt{k}\dot{q}. \tag{11}$$

The posture task error is then given by

$$e_{pos}(\ddot{q}) = k(q_d - q) - 2\sqrt{k}\dot{q} - \ddot{q} \tag{12}$$

C. Reference spreading

goal: explain error peaking, explain reference spreading, formulize reference extending, explain intermediate mode

Error peaking: Many controllers, such as the impedance-based controller described in the previous section, rely on a velocity reference for tracking. When a trajectory contains impacts, both the measured and reference velocity signals contain jumps. Due to, e.g., imperfect tracking or uncertainty regarding the position of the contact surface, the measured and reference jumps can be misaligned in time. This causes the tracking error to peak: a phenomenon that will be refered to as error peaking. This is shown in Figure xxx. The peak is paired with high control efforts that could damage the robot and its environment, while not necesarily contributing to better tracking. To prevent large peaks in the tracking error, robots often avoid high-velocity impacts.

Switching reference: Reference spreading (RS) enables high-velocity impacts by adressing error peaking. RS distinguishes between a reference before and after the impact. While neither of references contains velocity jumps, switching from the ante- to the post-impact reference upon impact results in a velocity jump that is aligned with the the velocity jump of the actual system. Figure xxx shows that this reduces the peak in the tracking error, meaning that faster impact become feasible.

Near-simultaneous impacts: Sometimes, impacts are paired with multiple velocity jumps. Corners of a contact surface may impact at diverging intervals in uncertain order during what is called near-simultaneous impacts. This is visualised in Figure xxx. Between the moment of impact and contact completion – an interval refered to as the intermediate (impact) phase – both the ante- and post impact reference would result in error peaking. Literature describes intermediate phase controller to deal with this: during the intermediate phase, velocity feedback may be disabled [8], or the velocity reference may be set to zero [10].

Reference extending: RS relies on the formulation of ante- and post impact references. These can be generated by splitting a reference at the nominal impact time, and then extending beyond the nominal impact time. One extending approach is maintaining constant

D. Impact detection

goal: explain error peaking (including near-simultaneous impacts), explain reference spreading, formulize reference extending, explain intermediate mode

E. Trajectory planning

III. Methods

A. Soft end effector

B. Trajectory planning via teleoperation

Due to the absence of an environment model, we cannot replicate the trajectory planning methods used in simulation as described in Section xxx. Alternatively, a human could demonstrate tasks through teleoperation. The trajectory tracked by the robot during demonstration could then be used as reference as it is guaranteed to be coherent with the robot's dynamics. This section frormulates a teleoperation controller and how it can be used to plan trajectories.

The teleoperation controller is based on the HTV Vive VR handheld devices. These VR devices provide rotation R_{VR}, angular velocity ω_{VR}, position p_{VR}, and velocity v_{VR}. These signals will be used with the QP framework described in Section II-B will be used for the teleoperation controller. Substituing the desired values into 10

C. Constraining control effort

Section xxx explained how velocity jumps result in control effort that may damage the robot or the environment. Even though there are no impact-driven velocity jumps in a reference extracted from a non-haptic VR device, jumps in velocity measurements may still cause excessive control effort.

D. Reference spreading formulation

IV. Experimental validation

A. Implementation

B. End effector evaluation

C. Reference spreading

D. Intermediate modes

V. Conclusion

APPENDIX

A. Rotation mathematics

B. End effector design

C. Supplementary experimental results