Impact-friendly robust control design with task-space quadratic optimization

Yuquan Wang, Abderrahmane Kheddar

Robotics: Science and Systems 2019
Freiburg im Breisgau, June 22-26, 2019

Impact-friendly robust control design with task-space quadratic optimization

Yuquan Wang and Abderrahmane Kheddar
CNRS-University of Montpellier, LIRMM, Interactive Digital Humans group, Montpellier, France.
Email: {yuquan.wang, kheddar}@lirmm.fr

***Abstract*—Almost all known robots fear impacts. Unlike humans, robots keep guarded motions to near zero-velocity prior to establishing contacts with their surroundings. This significantly slows down robotic tasks involving physical interaction. Two main ingredients are necessary to remedy this limitation: impact-friendly hardware design, and impact-friendly controllers. Our work focuses on the controller aspect. Task-space controllers formulated as quadratic programming (QP) are widely used in robotics to generate modular and reactive motion for a large range of task specifications under various constraints. We explicitly introduce discrete impact dynamics model into the QP-based controllers to generate robot motions that are robust to impact-induced state jumps in the joint velocities and joint torques. Our simulations, validate that our proposed impact-friendly QP controller is robust to contact impacts, shall they be expected or not. Therefore, we can exploit it for establishing contacts with high velocities, and explicitly generate task-purpose impulsive forces.**

I. Introduction

Embedding robots with the capability of generating or handling impulsive forces (i.e. setting or releasing contacts with high speed) will increase their performance for a large variety of tasks or be able to perform new ones. For example, dynamic parkour in legged locomotion (as recently showcased by Boston Dynamics), grasping various objects in the fly or the course of the action, pushing using high impact forces to fight static friction of objects in assembly or motion, etc. For this purpose, one needs to deal with two major aspects that are complementary. First, it is important to have a hardware that can absorb the impact energy by mechanical design, e.g. robots having elastic or variable impedance actuators; and for a given robot, what are the admissible impact ranges. Second, it is similarly important to design a controller that is robust to impact uncertainties. Indeed, it is impossible to know precisely when and where impacts will occur, even if we intentionally planned it. The reason why this is important is because, even for low-velocity, impacts duration is typically less than a millisecond, as suggested in Pashah et al. [1]. Impacts also result in abrupt change in the velocity and torque which are then none-derivable with respect to the control time-step, and hence, non-smooth. Moreover, the dynamics at impact writes differently and is used together with specific state observers. A reset map is generally needed to restart the controller from the current post-impact state.

As presented in Sec. II, there have been several approaches proposed to deal with impacts in robotics. Some, prevent the impact to occur by substantially reducing the speed prior to contacts. We do not consider such approaches as we aim at achieving high impact-purpose tasks. Therefore, we detail and compare approaches that are either robust to impact using dedicated control strategies, e.g. using hybrid control, or designed to specifically generate impacts. The literature in robotic impact is rich in terms of modelling techniques that we borrowed; e.g. the impact-induced state jumps in joint velocities, joint torques and the end-effector impulsive forces can be derived from the algebraic equations given by Zheng and Hemami [2].

Our contribution lies in handling and generating impact tasks using task-space controllers that are formulated as constraint optimization quadratic programming (QP) –e.g. the multi-objective QP controller in Bouyarmane et al. [3], see Sec. III. QP controllers proved to be very efficient in handling concurrently a large number of tasks with various sensors modalities. We asked ourselves the following question: *what changes are to be made to use these controllers to generate impact-tasks?* First, we designed a simulation of a simple robot commanded by a QP controller with few common tasks (set-point reaching) and constraints (torque, joint and velocity limits). Then, we investigated what happens if, in the course of the reaching task, the robot encounter an expected wall (an obstacle) set at a random Cartesian position between the initial robot's terminal point and the set-point to reach. This simulation revealed interesting observations that we report in Sec. III.

In brief, the jump of torques and the absence of contact forces (in the QP model, as we do not expect any contact), will make the QP controller instantly infeasible in many contact situations. This is illustrated in Fig. 1, and the main reason of the controller infeasibility or misbehavior is that the physical limitations can be instantly violated (which is not surprising indeed).

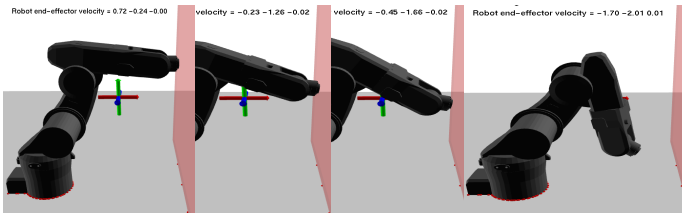

Fig. 1: Snapshots of a manipulator crashed with a wall at 0.72 m/s along the normal direction. After the impact the underlying QP controller becomes infeasible due to the violation of the joint torque limits.

In order to circumvent this, our idea is intuitively simple. We keep the structure of the task-space QP formalism as commonly

written (e.g. [3]) but we rewrite the constraints that are subject to jumps in the state, to embed a worst-case impact-aware strategy so that they remain feasible for a bounded change in states (velocity, torques). This bounded change is set from the hardware admissible impact, i.e. from the range of what a robot can absorb as impact energy by hardware. Our approach is thoroughly detailed in Sec. V. By doing so, we guarantee that our QP controller is robust to impacts shall they occur on purpose with uncertainties on their timing (and even location) or unexpectedly. Moreover, right after the impact, we can easily switch to force control by integrating the contact constraints with admittance task in the QP as in [3].

Using the DART simulator reported in Lee et al. [4], we incrementally validate the following aspects of our proposed impact-friendly QP controller, see Sec. VI:

- stabilization of impacts within the robot admissible limits, in terms of bounded joint velocity and impulsive torques;
- applying impulsive forces higher than the maximal forces provided by the statics.

The simulation results assess our approach in several impact scenarios and show that our formulation keeps the QP controller feasible where a usual QP formulation (that is not impact-aware) would fail. Also the QP controller simulation is connected to a physics engine that provides the feedback that is decoupled from the QP controller dynamical model. This proved to have very high transferability success to real experiments as reported in [3]. To our best knowledge, our work is the first to propose a unified task-space QP controller to generate both kinematic, dynamic and impact tasks.

II. Related work

Impacts amount to sudden transitions from non-contact to contact state with relatively high contact velocities. This leads to state jumps that are not in favor of smooth control approaches. Thus a commonly used strategy to deal with impacts is to exploit small contact velocities to avoid peaks in the robot states and keep use of continuous control laws. Stanisic and Fernández [5] proposed a sliding-mode controller to simultaneously stabilize the impact force and the contact velocity. In the 2D case, Liang et al. [6] and Dupree et al. [7] proposed Lyapunov controllers to stabilize the transition between non-contact to contact modes. Recently Salehian and Billard [8] proposed a local dynamics modulation strategy to ensure a smooth contact status transition. Along the tangential and normal direction of the contact surface, the second order dynamics of the relative position is stabilized. Tarn et al. [9] and Pagilla and Yu [10] apply a contact transition controller that regulates to zero, the velocity normal to the surface. The flying object catching approach proposed by Salehian et al. [11] on the other hand reduces the relative contact velocity by tracking the object trajectory and applies a soft catch. Near-zero contact velocity are not able to touch, push and grasp objects at the human-level speed. Thus we advocate to explicitly analyze the influence of the state jumps on the hardware limitations and re-design the controllers accordingly.

It is not possible for most robots joint motors to react to large impulsive forces within the impact duration. According to the experiments reported in Haddadin et al. [12], the variable stiffness actuator does not help to instantly absorb the impact energy. It takes up to 0.1 second to react to the same amount of impulsive torque that is generated within 1 millisecond. Thus rather than modifying further the mechanical structure of a robot to accommodate the impacts, we choose for now to improve the control systems.

Correct and complete impact laws for multiple contact and impacts are not known for inelastic impacts [13]. Recently in a flying object batting example, Jia et al. [14] validated that we can use the closed-form 2D impact dynamics model to generate the desired impulsive forces. However in 3D cases, Jia and Wang [15] pointed out that unless the initial sliding direction could be controlled along certain directions, a closed-form solution to the impact problem does not exist. Thus we restrict ourselves to the discrete impact model by Zheng and Hemami [2] which consists of a set of algebraic equations.

Introducing multiple impacts modeled by non-smooth mechanics is equivalent to introducing jumps to the robot states (e.g. joint velocities and end-effector forces), which in return requires switching of the constraints or task references. Given the impact location, Pagilla and Yu [10] modifies the reference trajectories such that the reference velocity along the surface normal is zero. Recently, Rijnen et al. [16] proposed to use two joint-space reference trajectories for dealing with pre- and post-impact behaviors, switching once from pre- to post-impact control based on actual impact detection. Using impact dynamics model and nonlinear optimization, Konno et al. [17] uses a three-phase optimization problem to generate reference trajectories as well as the posture at the impact moment to maximize the impulsive force. All of these methods define the reference trajectories in the joint space and rely on off-line computation.

The offline generation of the trajectories compromises reactiveness of a controller, increases the computation and deviates from the multi-objective controllers that uses real-time sensory feedback. More importantly, the switching of the reference [10], [16] or task controllers [17] depends on the impact detection, which is however not easy to obtain and requires precise and fast measurements. In general, due to the unknown exact impact timing, switching controllers that are defined either in task space or joint space, need to use the set points or references that are defined for the pre-impact mode during the post-impact mode for a certain time interval, no matter how small it is. Undesired misbehavior as such would unfortunately impose the risk of breaking the hardware limitations of a robot.

Different optimization-based control frameworks have been introduced to solve the robot redundancy resolution problem, e.g. hierarchical quadratic programming (HQP) solution proposed by Escande et al. [18] and soft task priorities by Salini et al. [19]. The contact transitions has not been explicitly and widely addressed by any of the QP frameworks.

The QP controllers rely on models of the robot itself, i.e.

kinematics and dynamics, and models of the external environment, e.g. the surface geometry and friction coefficient. Inspired by the nonlinear optimization trajectory generation method for impact tasks reported by Konno et al. [17], we introduce the impact dynamics model presented by Zheng and Hemami [2] to the QP controller to formulate constraints with respect to hardware limitations using predictions of state jumps.

Assuming the maximal acceleration/deceleration, Decré et al. [20] proposed to infer the maximal allowable joint velocity such that the joint position bounds will be fulfilled. Recently, Del Prete [21] proposed to use the exact time of constraints saturation compared to the method by Decré et al. [20]. However in case of unforeseen impacts, the modified constraints proposed by Decré et al. [20] or Del Prete [21] are not applicable for treating joint velocity jumps. All of these methods predict the joint state individually as a linear system, whereas the impact induced joint velocity jumps are coupled with each other.

To light of the previous screening, we propose modifications to the task space QP controllers such that they can safely work with and exploit impacts for efficient task execution and generating impulsive forces that could not be achieved otherwise. Compared to the switching schemes and the contact transition controllers, we do not rely on the precise contact location nor the precise impact timing while keeping the reactiveness and modularity of QP controllers.

III. Problem formulation

In Sec. III-A we recall the discrete impact dynamics model. Then, in Sec. III-B we formulate a generic QP controller to explain how it can be jeoperdized at impacts and what needs to be addressed in order to overcome this.

A. Dynamics model

The dynamic equation of an n degrees of freedom robot writes:

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + N(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} + J^\top \boldsymbol{f}, \tag{1}$$

where $M(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ denotes the inertia matrix, the vector $N(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ gathers both the Coriolis and the gravitation forces, $\boldsymbol{q} \in \mathbb{R}^n$ and $\boldsymbol{\tau} \in \mathbb{R}^n$ correspond to the joint position and the joint torque. In case of an established contact or external interaction, the contact force $\boldsymbol{f} \in \mathbb{R}^3$ associated with the Jacobian $J \in \mathbb{R}^{3\times n}$ is added to(1).

1) Impact dynamics: Let t_I be the time when the impact occurs, and δt its the duration. Assuming that δt is infinitely small, the joint configuration $\boldsymbol{q}(t_I)$ keeps constant along $t_I + \delta t$. As a consequence, the gravity and the Coriolis terms are assumed to be constant during δt, see more details in Konno et al. [17] and Zheng and Hemami [2]. The joint torque $\boldsymbol{\tau}$ is also assumed to be constant. As stated by Haddadin et al. [12], the motor needs way more time to change the joint torque than the impact duration that is less than 1 millisecond.

Integrating (1) over the impact duration δt, we can obtain the following:

$$M(\boldsymbol{q})\delta\dot{\boldsymbol{q}} = J^\top \bar{\boldsymbol{f}}\delta t, \tag{2}$$

where the joint velocity jump $\delta\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}(t_I + \delta t) - \dot{\boldsymbol{q}}(t_I)$. The integral of constant $N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{\tau}$ vanishes as δt is infinitely small, however the same does not hold for the impulsive contact force. We denote the average impulsive contact force over δt as $\bar{\boldsymbol{f}}$.

2) Definitions: In view of the discrete impact dynamics model (2) and the invertability of the matrix $M(\boldsymbol{q})$, we can obtain:

$$\delta\dot{\boldsymbol{q}} = M^{-1}J^\top \bar{\boldsymbol{f}}\delta t. \tag{3}$$

Assuming that the planar contact surface normal $\vec{\boldsymbol{n}}$ and the restitution coefficient c_{Res} are known, the post-impact contact point velocity is given by:

$$\dot{\boldsymbol{p}}^+ = -c_{\text{Res}} P_{\vec{\boldsymbol{n}}}\dot{\boldsymbol{p}}^- + N_{\vec{\boldsymbol{n}}}\dot{\boldsymbol{p}}^-,$$

where we used the superscript "$+$" and "$-$" to denote the post- and pre-impact entities, the projector and null-space projector are defined as: $P_{\vec{\boldsymbol{n}}} = \vec{\boldsymbol{n}}\vec{\boldsymbol{n}}^\top$ and $N_{\vec{\boldsymbol{n}}} = I - P_{\vec{\boldsymbol{n}}}$. The sign of c_{Res} depends on the post-impact status: rebounce($-$) or sticking($+$), which is difficult to predict. We conservatively assume that the rebounce will happen anyway and take the negative sign. Thus as long as $\dot{\boldsymbol{p}}^-$ is known, we can estimate to the contact point velocity jump:

$$\delta\dot{\boldsymbol{p}} = \dot{\boldsymbol{p}}^+ - \dot{\boldsymbol{p}}^- = -(1 + c_{\text{Res}})P_{\vec{\boldsymbol{n}}}\dot{\boldsymbol{p}}^-. \tag{4}$$

Using the kinematics $J\delta\dot{\boldsymbol{q}} = \delta\dot{\boldsymbol{p}}$ and (3), we can express $\bar{\boldsymbol{f}}$ as a function of the contact point velocity jump $\delta\dot{\boldsymbol{p}}$:

$$\bar{\boldsymbol{f}} = \frac{1}{\delta t}(JM^{-1}J^\top)^{-1}\delta\dot{\boldsymbol{p}}. \tag{5}$$

In view of (5) we can define the equivalent mass of the robot at the contact point $\boldsymbol{p}$, which is in essence the ratio between the force $\bar{\boldsymbol{f}}$ and the velocity jump $\delta\dot{\boldsymbol{p}}$:

$$M_{\text{eq}} = (JM^{-1}J^\top)^{-1}. \tag{6}$$

In some literatures, e.g. Ruspini and Khatib [22], M_{eq} is also referred to as the operational space inertia matrix.

B. A generic QP controller

Given multiple control objectives, the QP controller is able to find the local optimal solution if there is a feasible search space that is defined by the equality and inequality constraints. Without loss of generality, we detail the constraints formulation with respect to hardware limitations that are defined on the joint acceleration space.

1) Joint position limits: In view of the constraint $\underline{\boldsymbol{q}} \leq \boldsymbol{q}(t_k) \leq \bar{\boldsymbol{q}}$, we can use the *Euler backward* to approximate the derivative and back step until the joint acceleration:

$$\begin{aligned} \ddot{\boldsymbol{q}}(t_k) &\leq \frac{\bar{\boldsymbol{q}} - \boldsymbol{q}(t_{k-1}) - \dot{\boldsymbol{q}}(t_{k-1})\Delta t}{\Delta t^2} \\ -\ddot{\boldsymbol{q}}(t_k) &\leq -\frac{\underline{\boldsymbol{q}} - \boldsymbol{q}(t_{k-1}) - \dot{\boldsymbol{q}}(t_{k-1})\Delta t}{\Delta t^2}, \end{aligned} \tag{7}$$

where Δt denotes the sampling period.

2) Joint velocity limit: Applying the same procedure for the constraint: $\underline{\dot{\boldsymbol{q}}} \leq \dot{\boldsymbol{q}}(t_k) \leq \bar{\dot{\boldsymbol{q}}}$, we have the following two:

$$\ddot{\boldsymbol{q}}(t_k) \leq \frac{\bar{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_{k-1})}{\Delta t} \text{ and } -\ddot{\boldsymbol{q}}(t_k) \leq -\frac{\underline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_{k-1})}{\Delta t}. \quad (8)$$

3) Joint torque limit: In view of the robot dynamics (1), the constraint takes the following form:

$$\begin{aligned} M(\boldsymbol{q})\ddot{\boldsymbol{q}} &\leq \bar{\boldsymbol{\tau}} + J^\top \boldsymbol{f} - N(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ -M(\boldsymbol{q})\ddot{\boldsymbol{q}} &\leq -(\underline{\boldsymbol{\tau}} + J^\top \boldsymbol{f} - N(\boldsymbol{q}, \dot{\boldsymbol{q}})) \end{aligned}, \quad (9)$$

where we need to leave $M(\boldsymbol{q})$ on the left side to avoid matrix inversion that would otherwise change the polytope on $\ddot{\boldsymbol{q}}$.

4) QP formulation: In line with the state of the art QP controllers, e.g. the task space QP by Bouyarmane et al. [3], we simultaneously optimize multiple objectives by minimizing a weighted sum while fulfilling equality and inequality constraints:

$$\begin{aligned} \min_{\ddot{\boldsymbol{q}}} \quad & \ddot{\boldsymbol{q}}^\top Q \ddot{\boldsymbol{q}} + \sum_{i \in I_o} w_i \|\boldsymbol{e}_{f_i}\|^2 \\ \text{s.t.} \quad & \text{Joint position and velocity constraints: } (7-8), \\ & \text{Joint torque constraints: } (9), \end{aligned} \quad (10)$$

where for each task in the set I_o we expand the error norm as:

$$\|\boldsymbol{e}_{f_i}\|^2 = \ddot{\boldsymbol{q}}^\top Q_{f_i} \ddot{\boldsymbol{q}} + P_{f_i} \ddot{\boldsymbol{q}}.$$

The joint velocities $\dot{\boldsymbol{q}}$ appear in (i) the joint position, (ii) velocity and (iii) motor torque constraints (7-9). The external force appears in the bounds of the polytope defined by $M(\boldsymbol{q})\ddot{\boldsymbol{q}}$ in (9). Due to the instant change of the joint velocities $\delta\dot{\boldsymbol{q}}$ (defined later in (22)) and the impulsive force $\bar{\boldsymbol{f}}$ defined by (5), the feasibility of constraints (7-9) are compromized as the current constraints formulations are not aware of $\delta\dot{\boldsymbol{q}}$ and $\bar{\boldsymbol{f}}$.

If the constraints (7-9) were violated, we would not fulfil the hardware limits of a robot. In such cases, the optimality of the task execution is a none sense. Therefore the generic QP controller (10) needs to be aware of impacts when they are to occur intentionally or not.

IV. Impact robust QP controller

Bearing in mind that the feasibility of the joint position, velocity and motor torque constraints are compromized by the impact induced state jumps, we re-write the kinematics (joint velocity and position) constraints and impulsive joint torque constraints in Sec. IV-A and Sec. IV-B respectively. In Sec. IV-C, we present the post-impact task-space force controller. In Sec. IV-D, we introduce the predicted joint velocity jumps $\delta\dot{\boldsymbol{q}}$ as an additional decision variables to formulate a new QP controller.

A. Joint position and velocity constraints under state jumps

In order to analyze the kinematics constraints with respect to the future impacts, we need the *Euler forward* approximation

$$\dot{\boldsymbol{q}}(t_k) = \frac{\boldsymbol{q}(t_{k+1}) - \boldsymbol{q}(t_k)}{\Delta t}, \quad (11)$$

which allows us to re-write the joint position constraint (7) and joint velocity constraint (8) as:

$$\begin{aligned} \ddot{\boldsymbol{q}}(t_k) &\leq \frac{\bar{\boldsymbol{q}} - \boldsymbol{q}(t_{k+1}) + \dot{\boldsymbol{q}}(t_{k+1})\Delta t}{\Delta t^2} \\ -\ddot{\boldsymbol{q}}(t_k) &\leq -\frac{\underline{\boldsymbol{q}} - \boldsymbol{q}(t_{k+1}) + \dot{\boldsymbol{q}}(t_{k+1})\Delta t}{\Delta t^2} \\ -\ddot{\boldsymbol{q}}(t_k) &\leq \frac{\bar{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_{k+1})}{\Delta t} \\ \ddot{\boldsymbol{q}}(t_k) &\leq -\frac{\underline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_{k+1})}{\Delta t} \end{aligned}. \quad (12)$$

In a conservative view, suppose that the impact will happen at the next control step (i.e. when sending the current QP output to the robot). One needs to use the post-impact velocities $\dot{\boldsymbol{q}}^+(t_{k+1})$ to re-write the bounds of (12). Using (11) and the definition of the post-impact joint velocity, we can express $\dot{\boldsymbol{q}}^+(t_{k+1})$ as:

$$\begin{aligned} \dot{\boldsymbol{q}}^+(t_{k+1}) &= \dot{\boldsymbol{q}}^-(t_{k+1}) + \delta\dot{\boldsymbol{q}}(t_{k+1}) \\ &= \dot{\boldsymbol{q}}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t + \delta\dot{\boldsymbol{q}}(t_{k+1}), \end{aligned} \quad (13)$$

and:

$$\boldsymbol{q}^+(t_{k+1}) = \boldsymbol{q}(t_k) + \dot{\boldsymbol{q}}(t_k)\Delta t, \quad (14)$$

where we use the fact that the pre- and post-impact joint positions are the same. Then we can use (13) and (14) to rewrite the right hand side of (12) such that the new constraints formulation explicitly depend on the joint velocity jumps $\delta\dot{\boldsymbol{q}}$:

$$\begin{aligned} \ddot{\boldsymbol{q}}(t_k) &\leq \frac{\bar{\boldsymbol{q}} - \boldsymbol{q}^+(t_{k+1}) + \dot{\boldsymbol{q}}^+(t_{k+1})\Delta t}{\Delta t^2} \\ &= \frac{\bar{\boldsymbol{q}} - \boldsymbol{q}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t^2 + \delta\dot{\boldsymbol{q}}(t_{k+1})\Delta t}{\Delta t^2} \\ -\ddot{\boldsymbol{q}}(t_k) &\leq -\frac{\underline{\boldsymbol{q}} - \boldsymbol{q}^+(t_{k+1}) + \dot{\boldsymbol{q}}^+(t_{k+1})\Delta t}{\Delta t^2} \\ &= -\frac{\underline{\boldsymbol{q}} - \boldsymbol{q}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t^2 + \delta\dot{\boldsymbol{q}}(t_{k+1})\Delta t}{\Delta t^2} \\ -\ddot{\boldsymbol{q}}(t_k) &\leq \frac{\bar{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}^+(t_{k+1})}{\Delta t} \\ &= \frac{\bar{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_k) - \ddot{\boldsymbol{q}}(t_k)\Delta t - \delta\dot{\boldsymbol{q}}(t_{k+1})}{\Delta t} \\ \ddot{\boldsymbol{q}}(t_k) &\leq -\frac{\underline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}^+(t_{k+1})}{\Delta t} \\ &= -\frac{\underline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_k) - \ddot{\boldsymbol{q}}(t_k)\Delta t - \delta\dot{\boldsymbol{q}}(t_{k+1})}{\Delta t} \end{aligned} \quad (15)$$

Removing $\ddot{\boldsymbol{q}}(t_k)$ from both sides of (15), we can find a set of constraints that explicitly depends on $\delta\dot{\boldsymbol{q}}(t_{k+1})$ with a straightforward physical meaning in view of joint position and velocity bounds:

$$\begin{aligned} -\delta\dot{\boldsymbol{q}}(t_{k+1}) &\leq \frac{\bar{\boldsymbol{q}} - \boldsymbol{q}(t_k)}{\Delta t} \\ \delta\dot{\boldsymbol{q}}(t_{k+1}) &\leq -\frac{\underline{\boldsymbol{q}} - \boldsymbol{q}(t_k)}{\Delta t} \end{aligned}, \quad (16)$$

$$\begin{aligned} \delta\dot{\boldsymbol{q}}(t_{k+1}) &\leq \bar{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_k) \\ -\delta\dot{\boldsymbol{q}}(t_{k+1}) &\leq -\left(\underline{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}(t_k)\right) \end{aligned}. \quad (17)$$

In view of the reciprocal definition of (3), we can obtain:

$$\delta\boldsymbol{\tau} = J^\top \bar{\boldsymbol{f}} = \frac{M\delta\dot{\boldsymbol{q}}}{\delta t}.$$

Due to the appearance of δt in the denominator, disturbance of $\delta\dot{\boldsymbol{q}}$ lead to much bigger disturbance of the impulsive torque $\delta\boldsymbol{\tau}$. Likewise we can find Δt in the denominator of the joint position constraint (16). Thus we come to an important observation of the constraint violation order:

$$\delta\boldsymbol{\tau}\ (20) \prec\ \dot{\boldsymbol{q}}\ (17) \prec\ \boldsymbol{q}\ (16)\ , \tag{18}$$

where the impulsive joint torque constraint (20) is to be introduced in Sec. IV-B.

B. Impulsive joint torque constraints under state jumps

Using the statics relation $\delta\boldsymbol{\tau} = J^\top \bar{\boldsymbol{f}}$ and the impulsive force $\bar{\boldsymbol{f}}$ defined by (5) , we define the impulsive joint torque as:

$$\delta\boldsymbol{\tau} = \frac{1}{\delta t} \underbrace{J^\top (JM^{-1}J^\top)^{-1}}_{J_M^\dagger} \delta\dot{\boldsymbol{p}},$$

where we defined the weighted pseudo-inverse $J_M^\dagger$. Expanding $\delta\dot{\boldsymbol{p}}$ defined by (4), we can express $\delta\boldsymbol{\tau}$ as a function of the decision variable $\ddot{\boldsymbol{q}}$ of the original QP controller:

$$\begin{aligned}\delta\boldsymbol{\tau} &= \frac{1}{\delta t} J_M^\dagger \delta\dot{\boldsymbol{p}} \\ &= -\frac{c_{\text{Res}}+1}{\delta t} J_M^\dagger P_{\vec{\boldsymbol{n}}} \dot{\boldsymbol{p}}^-(t_{k+1}) \\ &= -\frac{c_{\text{Res}}+1}{\delta t} J_M^\dagger P_{\vec{\boldsymbol{n}}} J(\dot{\boldsymbol{q}}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t).\end{aligned} \tag{19}$$

In view of the impulsive torque bounds $\delta\underline{\tau} \le \delta\boldsymbol{\tau} \le \delta\bar{\tau}$, we can obtain the following constraint defined on the joint acceleration space:

$$\begin{aligned} J_M^\dagger P_{\vec{\boldsymbol{n}}} J\ddot{\boldsymbol{q}}(t_k) &\ge \frac{\delta\underline{\tau}}{c_{\text{Res}}+1} - \frac{J_M^\dagger P_{\vec{\boldsymbol{n}}} J\dot{\boldsymbol{q}}(t_k)}{\delta t} \\ \frac{\delta\bar{\tau}}{c_{\text{Res}}+1} - \frac{J_M^\dagger P_{\vec{\boldsymbol{n}}} J\dot{\boldsymbol{q}}(t_k)}{\delta t} &\ge J_M^\dagger P_{\vec{\boldsymbol{n}}} J\ddot{\boldsymbol{q}}(t_k). \end{aligned} \tag{20}$$

where we need to constrain $J_M^\dagger P_{\vec{\boldsymbol{n}}} J\ddot{\boldsymbol{q}}(t_k)$ as a whole due to the same reason for (9).

C. Contact transition

By fulfilling the modified joint position, velocity and impulsive torque constraints (16), (17) and (20), the new QP controller allows the robot to apply safely the maximal admissible contact velocity. Upon the detection of an impact, the QP controller is enhanced with an admittance control task without specific modifications to stabilize the contact force; and eventually regulates the post-impact force to a desired value, see details in Bouyarmane et al. [3]. We do not need to limit the desired contact force $\boldsymbol{f}^*$ or clamp the contact velocities.

Moreover, the contact force is subscribed within the friction cone to avoid non-desired slippage. We can approximate the friction cone with a generation matrix K and predict the contact force by

$$\boldsymbol{f}_{\text{QP}} = K\boldsymbol{\lambda},$$

where $\boldsymbol{f}_{\text{QP}} \in \mathbb{R}^3$, $K \in \mathbb{R}^{3\times N_c}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{N_c}$ denotes the weight of the generation matrix K. Each column $\boldsymbol{k}_i$ of K corresponds to a generating vector of the friction cone, which is visualized in Fig. 2. Empirically we can choose $N_c = 4$ for each contact point if the contact surfaces are accurately modeled. We use

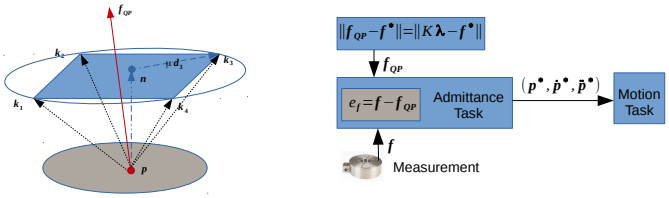

Fig. 2: The left sub-figure shows that each supporting vector $\boldsymbol{k}_i = \boldsymbol{n} - \mu\boldsymbol{d}_i$, where $\boldsymbol{n}$ and μ denote the surface normal direction and the friction coefficient. There is a schematic view of the task space force control in the right sub-figure.

the weights $\boldsymbol{\lambda}$ as part of the decision variables. We associate the search space of $\boldsymbol{\lambda}$ and $\ddot{\boldsymbol{q}}$ through the modified torque limit constraint (9):

$$\begin{aligned} M(\boldsymbol{q})\ddot{\boldsymbol{q}} - J^\top K\boldsymbol{\lambda} &\le \bar{\boldsymbol{\tau}} - N(\boldsymbol{q},\dot{\boldsymbol{q}}) \\ -M(\boldsymbol{q})\ddot{\boldsymbol{q}} + J^\top K\boldsymbol{\lambda} &\le -(\underline{\boldsymbol{\tau}} - N(\boldsymbol{q},\dot{\boldsymbol{q}})) \end{aligned}. \tag{21}$$

Given the desired force $\boldsymbol{f}^*$, we minimize the following error

$$\boldsymbol{e}_{\boldsymbol{\lambda}} = \boldsymbol{f}_{\text{QP}} - \boldsymbol{f}^* = K\boldsymbol{\lambda} - \boldsymbol{f}^*,$$

by adding $\|\boldsymbol{e}_{\boldsymbol{\lambda}}\|^2$ to the QP objectives such that the QP controller can infer the optimal $\boldsymbol{f}_{\text{QP}}$ with respect to the robot dynamics, constraints and objectives. Then simultaneously we use $\boldsymbol{f}_{\text{QP}}$ as a set-point to define the error for an admittance task which regulates the contact point motion based on the following error:

$$\boldsymbol{e}_{\boldsymbol{f}} = \boldsymbol{f} - \boldsymbol{f}_{\text{QP}},$$

where $\boldsymbol{f}$ denotes the measured force which is visualized in Fig. 2. At time step k, the QP controller infers the $\boldsymbol{f}_{\text{QP}}$ to be used at time step $k+1$ for the admittance task. In this way, the admittance task would track the force $\boldsymbol{f}_{\text{QP}}$ which is compatible to the rest of the control objectives, e.g. contact maintenance, rather than $\boldsymbol{f}^*$. Likewise the admittance task generates the set point $(\boldsymbol{p}^*, \dot{\boldsymbol{p}}^*, \ddot{\boldsymbol{p}}^*)$ for the motion tracking task which enables two separate set of stiffness parameters for force tracking and motion tracking separately.

D. Robust QP controller formulation

We propose to introduce $\delta\dot{\boldsymbol{q}}(t_{k+1})$ as additional decision variables for a QP controller. In view of (3), (4) and (5), we can predict $\delta\dot{\boldsymbol{q}}(t_{k+1})$ as:

$$\begin{aligned}\delta\dot{\boldsymbol{q}}(t_{k+1}) =& M^{-1} \underbrace{J^\top (JM^{-1}J^\top)^{-1}}_{J_M^\dagger} \delta\dot{\boldsymbol{p}}(t_{k+1}) \\ =& -(\bar{c}_{\text{Res}}+1) M^{-1} J_M^\dagger P_{\vec{\boldsymbol{n}}} J\dot{\boldsymbol{q}}^-(t_{k+1}) \\ =& -(\bar{c}_{\text{Res}}+1) M^{-1} J_M^\dagger P_{\vec{\boldsymbol{n}}} J(\dot{\boldsymbol{q}}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t),\end{aligned} \tag{22}$$

where we apply the upper bound of $c_{\text{Res}} \in [\underline{c}_{\text{Res}}, \bar{c}_{\text{Res}}]$.

Using (21) and (22), we can relate $\boldsymbol{\lambda}$ and $\delta\dot{\boldsymbol{q}}$ to $\ddot{\boldsymbol{q}}$ respectively. Using the decision variables:

$$\boldsymbol{x} = [\ddot{\boldsymbol{q}}(t_k),\ \delta\dot{\boldsymbol{q}}(t_{k+1})\ ,\boldsymbol{\lambda}(t_k)]^\top$$

we can conclude the following impact-robust QP controller:

$$\begin{aligned}
\min_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top Q\boldsymbol{x} + \sum_{i\in I_o} w_i \|\boldsymbol{e}_{f_i}\|^2 \\
\text{s.t.} \quad & \text{Joint position and velocity constraints: } (7-8), \\
& \text{Contact force within the friction cone: } \boldsymbol{\lambda} \geq \mathbf{0}, \\
& \text{Joint velocity jump prediction } \delta\dot{\boldsymbol{q}}(t_{k+1}) : (22)\ ,\cdot \\
& \text{Joint limits on } \delta\dot{\boldsymbol{q}}(t_{k+1}) : (16-17)\ , \\
& \text{Impulsive torque constraints:(20)}\ , \\
& \text{Joint torque constraints: (21),}
\end{aligned} \tag{23}$$

where the gray high-lighted parts are added compared to the original QP controller (10). The combination of (20) and (21) allows us to bound the torque and the impulsive torque.

In practice it is difficult to detect the exact moment t_I when the impact happens. The QP controller (23) optimizes the robot configurations such that the modified constraints (16-17) and (20) are always fulfilled, shall an impact happens or not.

Despite that the proposed QP controller enables dynamic contact transition at a contact velocity significantly higher than the motion generated by an impedance or admittance controllers, the modified constraints set would restrict the robot motion as if the impact was about to happen in the next step. Therefore if there does not exist a planned contact or absolutely zero possibility of an impact, we can use the original QP controller (10). In order to make the motion less conservative, we can remove these additional constraints when we are confident enough to be far from any contact.

V. Impact exploitation

Exploiting impacts enables higher robot performances for a large variety of explosive tasks. One can maximize the contact velocity $\boldsymbol{v}_n$ along the desired direction by adding $-\dot{v}_{\boldsymbol{v}_n}$ to the objective of (23), where $\_\boldsymbol{v}_n$ denotes the magnitude of $\boldsymbol{v}_n$:

$$v_{\boldsymbol{v}_n} = \frac{1}{2}\boldsymbol{v}_n^\top \boldsymbol{v}_n.$$

When we are in need of impulsive forces, we can maximize the momentum: $P = \int_0^{\delta t} \boldsymbol{f} dt$, or simply as $P = \bar{\boldsymbol{f}}\delta t$ using the average force $\bar{\boldsymbol{f}}$ defined by (5). Maximizing the the average impulsive force is equivalent to maximizing the momentum P. Using the definition of the equivalent mass (6), we define the equivalent momentum along the contact surface normal as:

$$P = M_{\text{eq}}\boldsymbol{v}_n = (J_{\boldsymbol{v}_n} M^{-1} J_{\boldsymbol{v}_n}^\top)^{-1} J_{\boldsymbol{v}_n}\dot{\boldsymbol{q}}.$$

We can measure the magnitude of P with the quadratic term

$$f_P = \frac{1}{2}P^\top P$$

whose gradient is available through straightforward differentiation:

$$\begin{aligned}
\dot{f}_P &= P^\top \dot{P} = \boldsymbol{v}_n^\top M_{\text{eq}}^\top (\dot{M}_{\text{eq}}\boldsymbol{v}_n + M_{\text{eq}}\ddot{\boldsymbol{v}}_n) \\
&= \boldsymbol{v}_n^\top M_{\text{eq}}^\top (\dot{M}_{\text{eq}} J_{\boldsymbol{v}_n}\dot{\boldsymbol{q}} + M_{\text{eq}}(\dot{J}_{\boldsymbol{v}_n}\dot{\boldsymbol{q}} + J_{\boldsymbol{v}_n}\ddot{\boldsymbol{q}})) \\
&= \underbrace{\boldsymbol{v}_n^\top M_{\text{eq}}^\top M_{\text{eq}} J_{\boldsymbol{v}_n}}_{P_P}\ddot{\boldsymbol{q}} + \underbrace{\boldsymbol{v}_n^\top M_{\text{eq}}^\top (\dot{M}_{\text{eq}} J_{\boldsymbol{v}_n} + M_{\text{eq}}\dot{J}_{\boldsymbol{v}_n})\dot{\boldsymbol{q}}}_{C_P}.
\end{aligned} \tag{24}$$

Note that the constant C_P is independent of the optimization variables. Therefore we only need to add $-\dot{f}_P$ to the objective of the QP controller (23) without explicitly expanding C_P.

VI. Simulation

We use simulations powered by the DART[1] physics engine to validate the proposed QP controller (23). DART simulates impulse with the numerical implementation proposed by Stewart and Trinkle [23] except for the circumscribed friction cone.

Due to space limitation, we present the results of contacts with a coefficient of restitution equals to 0.8 and a friction coefficient equals to 0.7, despite the proposed QP is able to work with different combinations. All the simulation code are written in Python and publicly available[2]. The interested readers are welcome to reproduce the results or modify the proposed QP controller.

In order to analyze the QP controller's constraints violation in a tractable way, and regardless of the complexity of the robot kinematics and dynamics, we choose a 6 degrees of freedom KR5 robot shown in Fig. 1. The QP controller generates joint acceleration commands at 200 Hz using close-loop feedback including joint configurations and forces. The impact duration δt is set to 0.005 s.

We compare the original QP controller (10) and our proposed impact-aware QP controller (23) with the same task: crashing a wall without knowing its exact location (and hence, the time of contact). The contact surfaces of both the palm and the wall are planar.

The results presented in Sec. VI-A, validate that despite the lack of knowledge by our QP controller on the impact, the modified constraints always meet the hardware limits. The figures from Sec. VI-B prove that we can stabilize the impact while regulating the post-impact contact force to a desired value. The simulation results presented in Sec. VI-C reveal that if we need impulsive forces beyond the limit inferred from the statics relation, we can maximize the impulsive force with task (24).

A. Feasibility comparison

In order to compare the original QP controller (10) and our proposed QP controller (23), we generate energetic impacts by maximizing the robot contact velocity to collide with a wall. The normal direction of the wall is known to the robot, i.e. the contact surface, but not the exact location of the wall. Determining the exact time of impact is far from being an easy

[1] https://dartsim.github.io/

[2] https://github.com/wyqsnddd/pyQpController

task in practice. Thus we consider that the time of impact is not known to the robot.

On top of different constraint specifications, both of the QP controllers have the same set of tasks: a (3DoF) orientation task that fixes the orientation of the terminal point to fit the wall's surface normal, a (2DoF) translation task that constrains the robot end-effector to slide on a planar surface and a (1DoF) velocity task that maximizes the velocity along the normal direction of the contact surface.

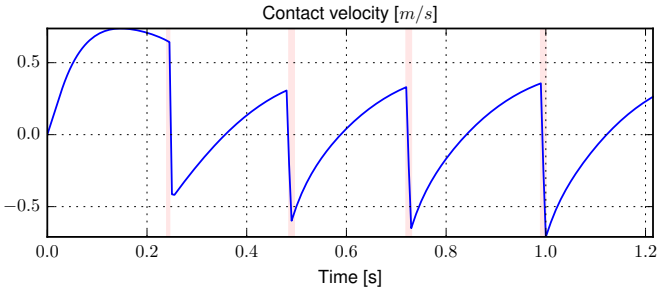

Fig. 3: Contact velocity generated by the original QP controller (10). The impact moments are highlighted with light red color.

Starting from the original (i.e. generic) QP controller (10), we can observe four impacts in Fig. 3 from the jumps of end-effector velocities. Due to the high contact velocities, each of them produces a rebounce. In Fig. 4, the joint torques (solid lines) violate the impulsive torque limits at each impact. However in all of these impact events, we can use the conservative predictions of the impulsive torques (dashed lines) to predict the constraints violation. Due to the constraints violation order (18), the impulsive torque constraints (20) would be violated prior to the violation of (16-17). Therefore we skip the figures for the modified joint velocity and position constraints (16-17) as the QP controller is already not feasible.

We apply the same colliding task with our proposed QP controller (23) and plot the contact velocities in Fig. 5. Thanks to the modified constraints the proposed QP controller computes the maximal feasible contact velocity 0.155 m/s.

We can find the corresponding joint torques in Fig. 6, where the impulsive torques are within the pre-defined bounds. At the same time, the QP solver exploited the hardware potential as we can tell that the prediction of $\delta\boldsymbol{\tau}_3$ is sliding along the lower bound during the task execution between $[0, 0.81]$ s. However there is indeed a gap between the predicted impulsive torque and its prediction in Fig. 6 due to the conservative computation.

B. Impact Stabilization

From Fig. 7 we can tell that the proposed QP controller established a contact with contact velocity $v_n = 0.156$ m/s and regulated the post-impact impulsive force to the desired contact force $\boldsymbol{f}^* = 57$ N using simulated force feedback.

Compared to Sec. VI-A, the QP controller replaced the maximal contact velocity task with the admittance control task upon the detection of an impact. However the impact timing is

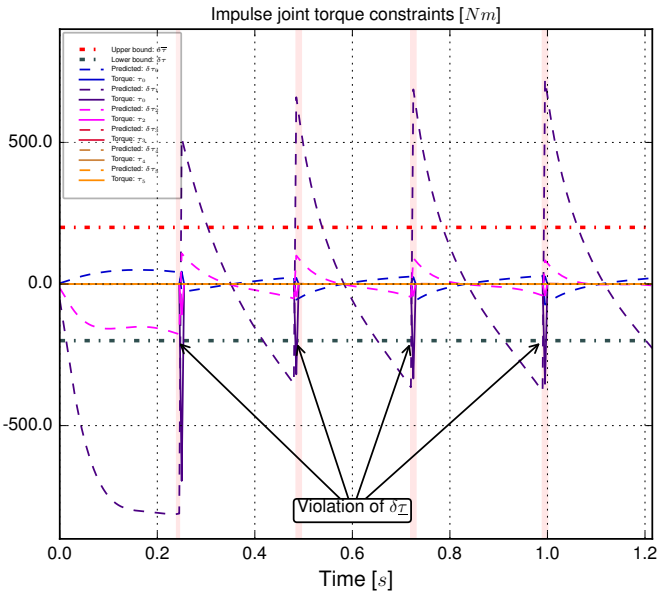

Fig. 4: The impulsive torque bounds are violated due to impacts. However we can predict the violation with the predictions that are plotted in dashed lines.

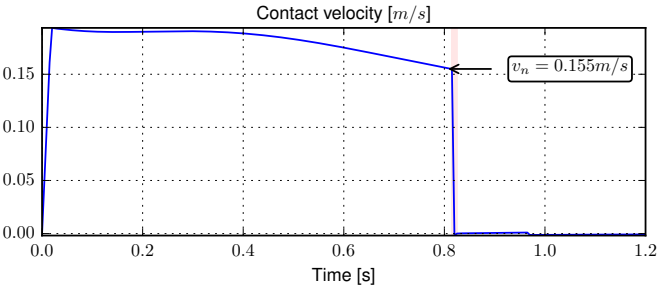

Fig. 5: Contact velocity generated by our proposed QP controller (23). Compared to Fig. 3, the controller (23) uses the same parameters for all the tasks and constraints. However the QP solver slows down the contact velocity based on the admissibility constraints (20).

not important thanks to the fact that the feasibility is always guaranteed by the modified joint constraints. For instance, during the simulation corresponding to Fig. 5, the QP controller was not aware of the impact during the entire process. We can tell that the joint torques shown in Fig. 6 increased until the maximal joint torque (25 Nm) after the impact. Indeed, the robot is trying to reach the desired end-effector velocity while actually sticking on a fixed wall.

C. Maximal Impulsive Force

In the vicinity of the contact surface, an admittance or impedance control law generates motions with near-zero contact velocity. The proposed QP controller on the other hand is able to generate motions that maximize the impulsive force via task (24). In Fig. 8, the impulsive force $\bar{\boldsymbol{f}}$ reached 599.04 N

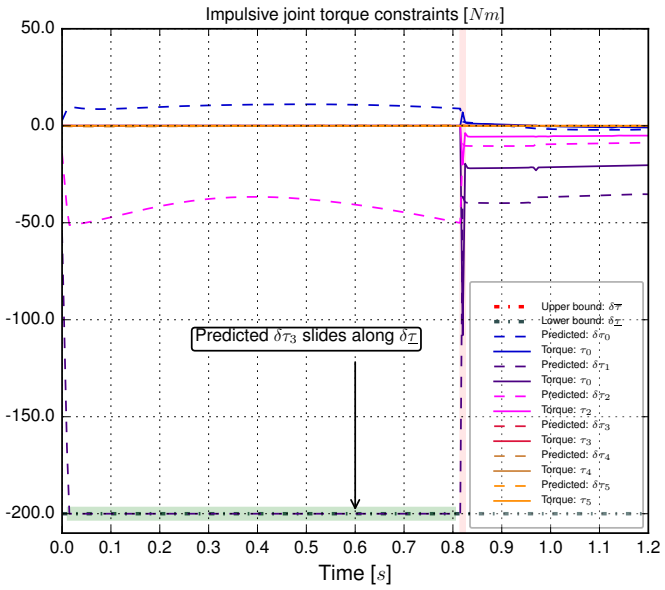

Fig. 6: The impulsive torque bounds are fulfilled while our proposed QP controller (23) is using the maximal feasible contact velocity as the predicted impulsive torque $\delta\boldsymbol{\tau}_3$ slides along the lower bound $\boldsymbol{\tau}_3$.

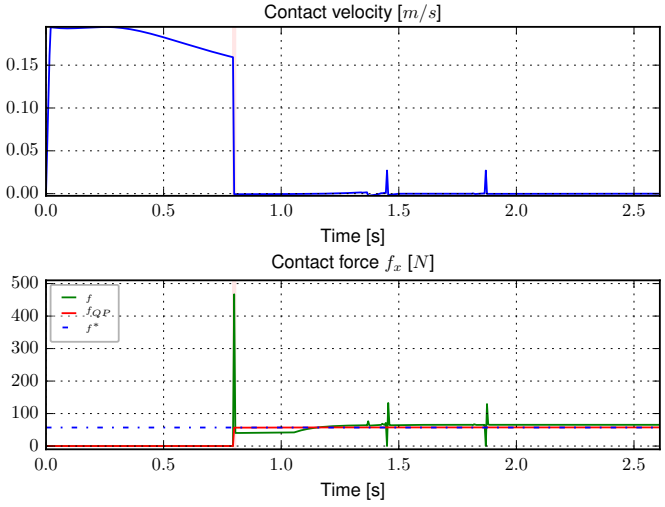

Fig. 7: The contact force $\boldsymbol{f}$ (green line) is stabilized to the desired value $\boldsymbol{f}^* = 57$ N (blue line). After the contact, the red line indicates that the QP controller assigned $\boldsymbol{f}_{\text{QP}} = \boldsymbol{f}^*$ for the admittance task depending on feasibilities of the constraints and the optimality of each of the other tasks.

which is not achievable by the control law based on statics.

VII. Conclusion

For a large variety of explosive tasks such as fast grasping, catching, pushing vividly, hitting strongly... the robots must be designed to not fear impact. In order to make possible a robot to generate task-purpose impacts, we need first to design a robust hardware that allows high energy transfer, but this is not sufficient. Indeed, we show that task-space closed-loop controller that write as quadratic programs to embed both tasks

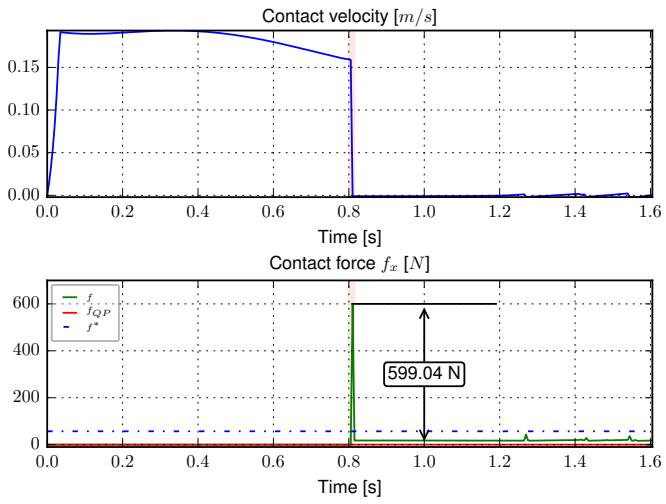

Fig. 8: The QP controller achieved 599.04 N impulsive force $\bar{\boldsymbol{f}}$ via task (24). The post-impact admittance force control presented in Sec. IV-C is not applied since $\boldsymbol{f}_{\text{QP}}$ determined by the QP solver is zero during the entire process.

objectives and constraints, will fail when subsequent jumps occur in the robot state (namely joint velocities and torques). It is easy to incorporate admissible impact ranges from the hardware into the QP controllers. Yet, this is not sufficient. We propose to enhance them with a formulation that allow the QP control framework as they write nowadays to deal in a stable and robust way to jumps in the state inherent to impacts.

Rather than slowing down to near-zero contact velocity or tracking pre-defined joint space trajectories, our proposed QP formulation is able to find the maximal feasible contact velocity based on predictions of the state jumps in joint velocities and torques using the discrete impact dynamics model without any reset map and by keeping closed feedback loop. The idea is to incorporate possible changes in the states jumps as decision variable and modulate the speed at a the worst case assuming the contact to occur in the next iteration. Although conservative by nature, this impact-aware QP control proved to be simple to implement. Through a series of simulations we assessed that it indeed enables impact stabilization and is able to maximize the impulsive force that would not be achieved otherwise. To the best of our knowledge, this is the first unified task-space QP controller that generates kinematic, dynamic and impact tasks.

As future work, we intent to achieve real experiments. The first problem we face is the unavailability of the admissible impact range data for all the robots we have in hands. This already shows that almost all robots were not designed for such a perspective.

Acknowledgments

The first author is partially supported by the National Natural Science Foundation of China (U1613216) and Shenzhen Fundamental Research Grant (JCYJ20180508162406177) through the Chinese Univ. of Hong-Kong, Shenzhen. He gratefully acknowledge the support.

References

[1] S Pashah, M Massenzio, and E Jacquelin. Prediction of structural response for low velocity impact. *International Journal of Impact Engineering*, 35(2):119–132, 2008.

[2] Yuan-Fang Zheng and Hooshang Hemami. Mathematical modeling of a robot collision with its environment. *Journal of Field Robotics*, 2(3):289–307, 1985.

[3] Karim Bouyarmane, Kevin Chappellet, Joris Vaillant, and Abderrahmane Kheddar. Quadratic programming for multirobot and task-space force control. *IEEE Transactions on Robotics*, 2019. doi: 10.1109/TRO.2018.2876782.

[4] Jeongseok Lee, Michael X Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S Srinivasa, Mike Stilman, and C Karen Liu. Dart: Dynamic animation and robotics toolkit. *The Journal of Open Source Software*, 3 (22):500, 2018.

[5] Ranko Zotovic Stanisic and Ángel Valera Fernández. Simultaneous velocity, impact and force control. *Robotica*, 27(7):1039–1048, 2009.

[6] Chien-Hao Liang, Shubhendu Bhasin, Keith Dupree, and Warren E Dixon. A force limiting adaptive controller for a robotic system undergoing a noncontact-to-contact transition. *IEEE Transactions on Control Systems Technology*, 17(6):1330–1341, 2009.

[7] Keith Dupree, Chien-Hao Liang, Guoqiang Hu, and Warren E Dixon. Adaptive lyapunov-based control of a robot and mass–spring system undergoing an impact collision. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):1050–1061, 2008.

[8] Seyed Sina Mirrazavi Salehian and Aude Billard. A dynamical system based approach for controlling robotic manipulators during non-contact/contact transitions. *IEEE Robotics and Automation Letters*, 2018.

[9] Tzyh-Jong Tarn, Yunying Wu, Ning Xi, and Alberto Isidori. Force regulation and contact transition control. *IEEE Control Systems*, 16(1):32–40, 1996.

[10] Prabhakar R Pagilla and Biao Yu. A stable transition controller for constrained robots. *IEEE/ASME transactions on mechatronics*, 6(1):65–74, 2001.

[11] Seyed Sina Mirrazavi Salehian, Mahdi Khoramshahi, and Aude Billard. A dynamical system approach for softly catching a flying object: Theory and experiment. *IEEE Transactions on Robotics*, 32(2):462–471, 2016.

[12] Sami Haddadin, Alin Albu-Schäffer, and Gerd Hirzinger. Requirements for safe robots: Measurements, analysis and new insights. *The International Journal of Robotics Research*, 28(11-12):1507–1527, 2009.

[13] David E Stewart. Rigid-body dynamics with friction and impact. *SIAM review*, 42(1):3–39, 2000.

[14] Yan-Bin Jia, Matthew Gardner, and Xiaoqian Mu. Batting an in-flight object to the target. *The International Journal of Robotics Research*, 0(0):0278364918817116, 0. doi: 10.1177/0278364918817116. URL https://doi.org/10.1177/0278364918817116.

[15] Yan-Bin Jia and Feifei Wang. Analysis and computation of two body impact in three dimensions. *Journal of Computational and Nonlinear Dynamics*, 12(4):041012, 2017.

[16] Mark Rijnen, Eric de Mooij, Silvio Traversaro, Francesco Nori, Nathan van de Wouw, Alessandro Saccon, and Henk Nijmeijer. Control of humanoid robot motions with impacts: Numerical experiments with reference spreading control. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4102–4107. IEEE, 2017.

[17] Atsushi Konno, Tomoya Myojin, Takaaki Matsumoto, Teppei Tsujita, and Masaru Uchiyama. An impact dynamics model and sequential optimization to generate impact motions for a humanoid robot. *The International Journal of Robotics Research*, 30(13):1596–1608, 2011.

[18] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, 33(7):1006–1028, 2014.

[19] Joseph Salini, Vincent Padois, and Philippe Bidaud. Synthesis of complex humanoid whole-body behavior: a focus on sequencing and tasks transitions. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1283–1290. IEEE, 2011.

[20] Wilm Decré, Ruben Smits, Herman Bruyninckx, and Joris De Schutter. Extending itasc to support inequality constraints and non-instantaneous task specification. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 964–971. IEEE, 2009.

[21] Andrea Del Prete. Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators. *IEEE Robotics and Automation Letters*, 3(1):281–288, 2018.

[22] Diego Ruspini and Oussama Khatib. Collision/contact models for the dynamic simulation of complex environments. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 82. Citeseer, 1997.

[23] David E Stewart and Jeffrey C Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering*, 39(15):2673–2691, 1996.