On-line force capability evaluation based on efficient polytope vertex search

Antun Skuric, Vincent Padois, David Daney

On-line force capability evaluation based on efficient polytope vertex search

Antun Skuric[1], Vincent Padois[1], David Daney[1]

***Abstract*— Ellipsoid-based manipulability measures are often used to characterize the force/velocity task-space capabilities of robots. While computationally simple, this approach largely approximate and underestimate the true capabilities. Force/velocity polytopes appear to be a more appropriate representation to characterize robot's task-space capabilities. However, due to the computational complexity of the associated vertex search problem, the polytope approach is mostly restricted to offline use, *e.g.* as a tool aiding robot mechanical design, robot placement in work-space and offline trajectory planning. In this paper, a novel on-line polytope vertex search algorithm is proposed. It exploits the parallelotop geometry of actuator constraints. The proposed algorithm significantly reduces the complexity and computation time of the vertex search problem in comparison to commonly used algorithms. In order to highlight the on-line capability of the proposed algorithm and its potential for robot control, a challenging experiment with two collaborating *Franka Emika Panda* robots, carrying a load of 12 kilograms, is proposed. In this experiment, the load distribution is adapted on-line, as a function of the configuration dependant task-space force capability of each robot, in order to avoid, as much as possible, the saturation of their capacity.**

I. Introduction

Robotics manipulators and their environments are traditionally optimised for a set of specific tasks and their efficiency is based on long-term task execution. Recently, with the introduction of collaborative robots in human environments, robots need to be adaptable to unexpected events, flexible in both task definition and execution and provide high degree of safety for all humans involved. Therefore evaluating robot capabilities and optimising their performance in advance, with tools designed for more traditional robots, is no longer a suitable approach. New on-line capable and accurate evaluation techniques are needed to account for constantly changing environments, flexible tasks and interaction requirements of collaborative robots.

There are several types of metrics developed to characterise robot's kinematic, kineto-static and dynamic capabilities. The characteristics of robots are related to their kinematics, mechanical design and actuation/joint limits. Kineto-static capabilities of robots characterize the ranges of achievable twists and wrenches in arbitrary directions while in *static conservative conditions*[1]. Dynamic capabilities characterise ranges of achievable accelerations and link elastic deformations.

Arguably the most widely used metrics are kineto-static capacity metrics based on *dexterity indices*, expressing robot's ability to move and apply forces and torques in arbitrary directions with equal ease [1]. The widely used implementation of this metric are velocity and force manipulability ellipsoids, introduced by Yoshikawa [2]. On the one hand, velocity manipulability ellipsoids are often used as a performance measure to optimise robot trajectories and avoid unwanted configurations. On the other hand, force manipulability ellipsoid characterises the forces the robot can apply or resist based on its kinematics. It is used during the design stage of robotic manipulators [1] as well as for off-line trajectory planning [3].

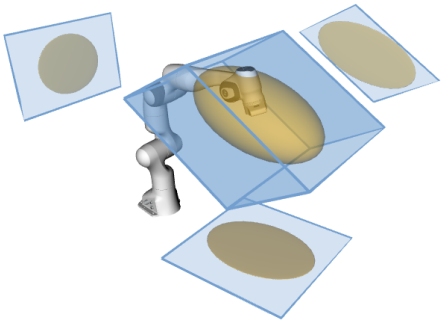

Fig. 1. 2D and 3D force polytopes and their ellipsoid counterparts for a 7 degrees of freedom (DoF) *Franka Emika Panda* robot. Both polytopes and ellipsoids are calculated separately for the 3D and for each of the 2D reduced task-space cases. Both polytopes and ellipsoids take in consideration the true joint torque limits provided by the manufacturer. The underestimation of the true force capabilities of the robot by ellipsoids appears clearly.

Even though manipulability ellipsoids are widely used (mostly because of their calculation efficiency), they are just an approximation and can lead to a drastic underestimation of the real kineto-static capacities of robots [4]. In the context of collaborative robotics, the margin between the actuation capabilities of the robot and the requirement of the task is largely reduced with respect to more classical, oversized, industrial robots. Thus, any over/under-estimation of the true capabilities of the system has strong negative implications either in terms or safety or in terms of efficiency.

Chiacchio et al.[5] demonstrated that the exact wrench and twist capacities have the form of polytopes. Polytopes do not underestimate the robot capacity [6], can easily integrate variable and non-symmetric limits, dynamics and external forces [4]. Additionally, in the context of collaborative robotics, when evaluating capacities of multiple robots, sum and intersection of ellipsoids becomes a complex problem to solve and interpret [7][8][9] whereas these operations are well-defined for polytopes. Figure 1 illustrates the difference of force capability estimation between polytope and ellipsoid characterization. However, evaluating twist and wrench polytopes relies on a *vertex enumeration* problem [10] which is a computationally intensive

*This work is supported by BPI France through the LICHIE project in collaboration with Airbus. The code of the proposed algorithm is publicly available at https://gitlab.inria.fr/askuric/polytope_vertex_search

[1] Auctus, Inria / IMS (Univ. Bordeaux / Bordeaux INP / CNRS UMR 5218), 33405 Talence, France firstname.lastname@inria.fr

task and is the main obstacle for wider uses of polytopes instead of ellipsoids in practice.

In this paper, a brief overview of polytope-based task-space capability characterization is given in section II. Then, a new on-line capable polytope vertex finding algorithm for efficiently calculating twist and wrench polytopes is proposed and described in section III. The new algorithm is capable of calculating polytope vertices for 6DoF and 7DoF robots under 3ms in Python as illustrated in section IV. This opens many doors in terms of on-line usage of such capability indicators. To demonstrate the benefits of our method for on-line robot control, an experimental setup with two collaborating *Franka Emika Panda* robots is proposed. It is shown that by leveraging the real carrying capacity on-line measure of the robots, it is possible to design a control strategy allowing to carry, a voluntarily exaggerated, mass of 12 kilograms. Discussions on the potential applications of the results of this work in collaborative robotics are then proposed in section V.

II. Task-space twist and wrench feasibility polytopes basics

For a serial robotic manipulator with n degrees of freedom, with joint space generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ and with Jacobian matrix $J(\boldsymbol{q}) \in \mathbb{R}^{m\times n}$ (with $m = 6$ in the 3D case and $m = 3$ in the planar one), the task space Cartesian twist $\boldsymbol{v}$ can be calculated from generalized velocities $\dot{\boldsymbol{q}}$ using

$$\boldsymbol{v} = J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{1}$$

The dual relation relaying the Cartesian wrench $\boldsymbol{f} \in \mathbb{R}^m$ with the generalized forces $\boldsymbol{\tau} \in \mathbb{R}^n$ is given by

$$J(\boldsymbol{q})^T\boldsymbol{f} + \boldsymbol{\tau}_0 = \boldsymbol{\tau} \tag{2}$$

$\boldsymbol{\tau}_0$ is a generalized force that does not "produce" any task space wrench, *i.e.* $\boldsymbol{\tau}_0$ belongs[1] to the kernel $\mathcal{K}er(J(\boldsymbol{q}))$ of $J(\boldsymbol{q})$. Conversely, $(\boldsymbol{\tau} - \boldsymbol{\tau}_0)$ lies in the image $\mathcal{I}m(J(\boldsymbol{q})^T)$ of $J(\boldsymbol{q})^T$, *i.e.* "produces" task space wrenches.

While equations (1) and (2) describe the kineto-static behaviour of a robot, generalized forces $\boldsymbol{\tau}$ and velocities $\dot{\boldsymbol{q}}$ are constrained due to the physical limits of the robot construction and actuators

$$\underline{\dot{q}}_i \leq \dot{q}_i \leq \overline{\dot{q}}_i \tag{3a}$$

$$\underline{\tau}_i \leq \tau_i \leq \overline{\tau}_i \tag{3b}$$

Given these constraints, the feasible twist polytope defined by equation (1) can be written

$$\mathcal{P}_v = \left\{\boldsymbol{v} \in \mathbb{R}^m \,\middle|\, \dot{\boldsymbol{q}} \in [\underline{\dot{\boldsymbol{q}}}, \overline{\dot{\boldsymbol{q}}}],\ \boldsymbol{v} = J(\boldsymbol{q})\dot{\boldsymbol{q}}\right\} \tag{4}$$

Similarly, the feasible wrench polytope is defined as[2]

$$\mathcal{P}_f = \left\{\boldsymbol{f} \in \mathbb{R}^m \,\middle|\, \boldsymbol{\tau} \in \left\{[\underline{\boldsymbol{\tau}}, \overline{\boldsymbol{\tau}}] \cap \mathcal{I}m(J^T)\right\}, J^T\boldsymbol{f} = \boldsymbol{\tau}\right\} \tag{5}$$

Equations (4) and (5) provide compact yet implicit representations of the feasible twist and wrench polytope of a robot. The characterization of these capabilities require an explicit definition of these polytopes, *i.e.* a computation of their vertices. In the case of $\mathcal{P}_v$, computing the vertices boils down to determining the convex-hull of the image of the vertices of the constraints polytope defined by equation (3a) through the linear mapping $J(\boldsymbol{q})$. The case of $\mathcal{P}_f$ yields an extra difficulty as one first needs to determine the intersection between the constraints polytope defined by equation (3b) and $\mathcal{I}m(J(\boldsymbol{q})^T)$.

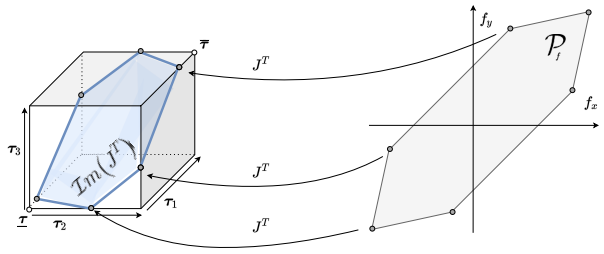

Fig. 2. Example representation of force polytope $\mathcal{P}_f$ vertices in joint and task-space for a redundant 3DoF planar robot: n=3, m=2.

Over the years many vertex search (enumeration) methods have been proposed [11]. Most of the approaches in literature are optimised to solve high dimensional problems and provide an abstraction from the actual system that is being analysed. One of the most commonly used method for vertex enumeration is the double description method [10] which formulates the vertex search problem as transformation of an half-space representation $\mathcal{H} - rep$ to a vertex representation $\mathcal{V} - rep$. This formulation generalizes well in between different high dimensional problems but lacks the flexibility to incorporate information about the physical system being analysed.

Therefore various methods have been introduced in order to leverage specific problem formulation and improve the computational performance of vertex search. The scaling factor method has been developed for planar parallel robots [12] and later adapted to planar serial manipulators [9]. This method reduces the search space with one scaling factor to an exhaustive search in the scalar space. A very efficient algorithm for planar robots has also been proposed by Gouttefarde et al. [13] which navigates the polytope boundaries in search for extremities. Both the scaling factor and boundaries navigation methods are designed for planar vertex search and do not scale to three dimensional world.

Chiacchio et al. in their original paper [5] propose an algorithm for finding the vertices of task-space polytopes by introducing slack variables and performing an exhaustive search through the joint space limits. This algorithm, although significantly improving the evaluation is still too complex for on-line execution. This algorithm has been improved by Sasaki et al. [14] where the computational complexity has been significantly reduced by introducing the geometrical representation of actuator constraints as an n-dimensional parallelotop.

The next section introduces additional improvements of the algorithms proposed in [5] and [14]. The resulting algorithm significantly reduces the computation time and complexity of the vertex enumeration problem for task-space polytopes. The proposed description is restricted to the most complex case of

[1]We recall here that $\mathcal{K}er(J(\boldsymbol{q})) = \mathcal{I}m(J(\boldsymbol{q})^T)^\perp$.

[2]The dependence of J to $\boldsymbol{q}$ is dropped here and when further needed for the sake of clarity.

$\mathcal{P}_f$ but the method is also computationally beneficial in the simpler case of computing $\mathcal{P}_v$. Also in order to ease illustrations, the considered task-space wrench is limited to its force component without any loss of generality.

III. Vertex finding algorithm formulation

For an nDoF serial manipulator, the feasible space of joint torques $\boldsymbol{\tau}$ forms an n-dimensional parallelotop with n pairs of parallel faces defined by the joint limits (3b). The image of the Jacobian transpose matrix $\mathcal{I}m(J(\boldsymbol{q})^T)$ is a r-dimensional subspace of the parallelotop (where r is the rank of $J(\boldsymbol{q})^T$ and r=m for non-singular configurations[3]), which corresponds to set of torques yielding non zero task space wrenches. Sasaki et al. [14] have shown that the extreme values of joint torques $\boldsymbol{\tau_{vert}} \in \mathcal{I}m(J(\boldsymbol{q})^T)$ belong to the n-m dimensional faces of the constraint parallelotop. Furthermore, there is a one-to-one correspondence between $\boldsymbol{\tau_{vert}}$ torque vertices and the $\boldsymbol{f}_{vert}$ vertices of polytope $\mathcal{P}_f$ can be computed as

$$\boldsymbol{f}_{vert} = J(\boldsymbol{q})^{T+}\boldsymbol{\tau}_{vert} \tag{6}$$

where $J(\boldsymbol{q})^{T+}$ is the pseudo-inverse of $J(\boldsymbol{q})^T$ which provides the unique and exact solution of the inversion problem in this specific case.

Figure 2 shows the example of a 3DoF planar robot (n=3, m=2) for which the joint torque space is a cube and $\mathcal{I}m(J(\boldsymbol{q})^T)$ is a plane. The extreme values of feasible joint torques are found on its $n-m=1$ dimensional faces, *i.e* its edges. For the example shown in figure 3, n=3 and m=1 so the vertices belong to the $n-m=2$ dimensional faces of the cube, *i.e.* its 2D sides.

Therefore, state-of-the-art algorithms such as [5] and [14] propose an exhaustive search over all the n-m dimensional parallelotop faces to find the extreme values of joint torques $\boldsymbol{\tau_{vert}}$ and consequently the vertices of force polytope $\mathcal{P}_f$. In this paper, a new representation of joint torque vector $\boldsymbol{\tau}$ is proposed enabling an efficient navigation of the $n-m$ dimensional parallelotop faces, reducing the complexity of the exhaustive search.

A. Proposed algorithm

Consider a joint torque vector $\boldsymbol{\tau}$ meeting constraints 3b. It can be defined as

$$\boldsymbol{\tau} = \underline{\boldsymbol{\tau}} + \alpha_1\boldsymbol{\tau}_1 + \alpha_2\boldsymbol{\tau}_2 + ... + \alpha_n\boldsymbol{\tau}_n \tag{7}$$

where $\alpha_i \in [0,1]$ are scalars weights and vectors $\boldsymbol{\tau}_i$ are orthogonal vectors in joint space aligned with i-th axis of the parallelotop, defined as $\boldsymbol{\tau}_i = \begin{bmatrix}0 & \dots & \overline{\tau}_i - \underline{\tau}_i & \dots & 0\end{bmatrix}^T$.

Since the vertices $\boldsymbol{\tau_{vert}} \in \mathcal{I}m(J(\boldsymbol{q})^T)$ belong to the n-m dimensional faces of the joint torque parallelotop, representation (7) provides an elegant way to navigate them, by fixing m out of n scalars α_i either to 0 or to 1.

An n-dimensional parallelotop has $\binom{n}{m}$ sets of 2^m parallel n-m dimensional faces. Each set of parallel faces can be represented with the linear combination of the same set of n-m base vectors $\boldsymbol{\tau}_i$ or in other words with a set of n-m scalars α_i. The remaining m scalars are set either to 0 or 1 and define the origin $\boldsymbol{\tau}_o$ of each one of the 2^m faces from the set

$$\boldsymbol{\tau}_o = \underline{\boldsymbol{\tau}} + \alpha_1\boldsymbol{\tau}_1 + ... + \alpha_m\boldsymbol{\tau}_m \tag{8}$$

Therefore each n-m dimensional face can be reached by partitioning $\boldsymbol{\alpha} = \begin{bmatrix}\boldsymbol{\alpha}_1^T & \boldsymbol{\alpha}_2^T\end{bmatrix}^T$, $\boldsymbol{\alpha}_1$ containing m scalars α_i fixed to 0 or 1 and $\boldsymbol{\alpha}_2$ containing the remaining n-m scalars which define the linear combination of n-m base vectors $\boldsymbol{\tau}_i$.

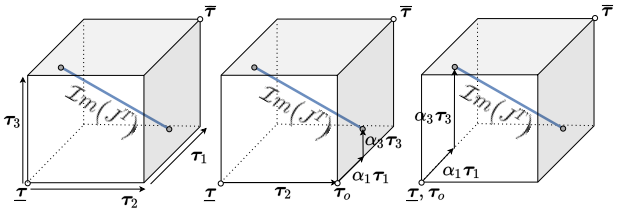

Fig. 3. Example of joint space interpretation of the vertex search algorithm solution for n=3 and m=1.

Figure 3 illustrates the case where $\mathcal{I}m(J^T)$ is a line in joint space. In this case the intersection space is n-m=2 dimensional. In this case, two scalars α_1 and α_3 are enough to characterise both vertices of interest with α_2 fixed to 0 or 1.

In order to find the vertices of the force polytope $\mathcal{P}_f$, equations (2) and (7) are combined to restrict the search to the set of torques $\boldsymbol{\tau} \in \left\{[\underline{\boldsymbol{\tau}}, \overline{\boldsymbol{\tau}}] \cap \mathcal{I}m(J^T)\right\}$

$$J(\boldsymbol{q})^T\boldsymbol{f}_{vert} = \underline{\boldsymbol{\tau}} + \alpha_1\boldsymbol{\tau}_1 + \alpha_2\boldsymbol{\tau}_2 + ... + \alpha_n\boldsymbol{\tau}_n \tag{9}$$

For each $\binom{n}{m}$ combination of the m scalars, 2^m values of $\boldsymbol{\tau}_o$ can be calculated. For each possible value of $\boldsymbol{\tau}_o$, a linear system deriving from equation (9) can be solved to find the n-m scalars $\boldsymbol{\alpha}_2$ and the corresponding force polytope vertex $\boldsymbol{f}_{vert}$

$$\underbrace{\begin{bmatrix}J(\boldsymbol{q})^T & -\boldsymbol{\tau}_{m+1} & \dots & -\boldsymbol{\tau}_n\end{bmatrix}}_{Z_{n\times n}} \begin{bmatrix}\boldsymbol{f}_{vert} \\ \alpha_{m+1} \\ \vdots \\ \alpha_n\end{bmatrix} = \boldsymbol{\tau}_o \tag{10}$$

If Z is invertible and if $\boldsymbol{\alpha}_2 \in [\boldsymbol{0}, \boldsymbol{1}]$, $\boldsymbol{f}_{vert}$ is a vertex of the polytope $\mathcal{P}_f$

$$\begin{bmatrix}\boldsymbol{f}_{vert} \\ \boldsymbol{\alpha}_2\end{bmatrix} = Z^{-1}\boldsymbol{\tau}_L, \quad \boldsymbol{\alpha}_2 \in [\boldsymbol{0}, \boldsymbol{1}] \tag{11}$$

Overall, in order to navigate all possible combinations of the n-m dimensional faces which may contain vertices of the force polytope $\mathcal{P}_f$, $\binom{n}{m}=\frac{n!}{m!(n-m)!}$ inversions of Z and $2^m\frac{n!}{m!(n-m)!}$ checks that $\boldsymbol{\alpha}_2 \in [\boldsymbol{0}, \boldsymbol{1}]$ have to be performed.

In the case depicted in figure 3 (n=3, m=1), Z is inverted only $\binom{3}{1}$=3 times and conditions are evaluated 6 times, which exactly corresponds to the number of faces of the parallelotop. Taking the example of the 3DoF planar robot (n=3, m=2) the number of Z inversions is 3 and the number of condition evaluation is 12, which exactly corresponds to the number of edges of the parallelotop. One of the nice features of this approach is that it does not require an explicit inversion of $J(\boldsymbol{q})^T$ as in equation (6) to actually obtain the set of vertices composing $\mathcal{P}_f$.

[3]In the remainder of this paper, the robot is, without loss of generality, assumed to be in a non singular configuration.

B. Improvement using the SVD

In order to further improve the efficiency of the proposed algorithm the Singular Value Decomposition (SVD) [15] of $J(\boldsymbol{q})$ is used

$$J(\boldsymbol{q}) = U \underbrace{\begin{bmatrix} S & O_{m\times(n-m)} \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}}_{V^T} \tag{12}$$

where $S = diag(\sigma_1 \dots \sigma_m)$ is a diagonal matrix containing the m singular values of $J(\boldsymbol{q})$. $V_1^T \in \mathbb{R}^{m\times n}$ projects the space of generalized velocities onto the preimage of $\mathcal{I}m(J(\boldsymbol{q}))$ and $V_2^T \in \mathbb{R}^{n-m\times n}$ is a basis of $\mathcal{K}er(J(\boldsymbol{q}))$. V_1^T and V_2^T are orthogonal vector subspaces yielding $V_2^T V_1 = O$ and thus

$$V_2^T J(\boldsymbol{q})^T \boldsymbol{f} = \boldsymbol{0} \tag{13}$$

This allows to reduce (11) to

$$\underbrace{V_2^T \begin{bmatrix} -\boldsymbol{\tau}_{m+1} \cdots -\boldsymbol{\tau}_n \end{bmatrix}}_{T_{(n-m)\times(n-m)}} \boldsymbol{\alpha}_2 = V_2^T \boldsymbol{\tau}_o \tag{14}$$

If T is invertible, $\boldsymbol{\alpha}_2$ can be computed by

$$\boldsymbol{\alpha}_2 = T^{-1} V_2^T \boldsymbol{\tau}_o \tag{15}$$

When the computed $\boldsymbol{\alpha}_2 \in [\boldsymbol{0}, \boldsymbol{1}]$, the corresponding force polytope vertex can be calculated as

$$\boldsymbol{f}_{vert} = J(\boldsymbol{q})^{T+} \big(\underline{\boldsymbol{\tau}} + \alpha_1 \boldsymbol{\tau}_1 + \cdots + \alpha_n \boldsymbol{\tau}_n \big) \tag{16}$$

This new approach requires the calculation of the SVD and the Jacobian matrix pseudo-inverse $J(\boldsymbol{q})^{T+}$. Nevertheless, since the Jacobian transpose pseudo-inverse can be efficiently calculated from the SVD as $J(\boldsymbol{q})^{T+} = U\Sigma^{T+}V^T$ and since both the SVD and pseudo-inverse are calculated once and of all per algorithm run, the computation efficiency is greatly improved due to the matrix dimension reduction from n to $(n-m)$ when inverting T instead of Z.

C. Matrix inverse condition

Due to the constrained nature of $\boldsymbol{\alpha}_2$, it is possible to efficiently calculate some bounds $\boldsymbol{t}_{ub}$ and $\boldsymbol{t}_{lb}$ such that $T\boldsymbol{\alpha} \in [\boldsymbol{t}_{lb}, \boldsymbol{t}_{ub}]$. These bounds are found by row-wise summing only positive and only negative elements t_{ij} of T

$$t_{i,ul} = \sum_j \max(t_{ij}, 0), \qquad t_{i,lb} = \sum_j \min(t_{ij}, 0) \tag{17}$$

This creates a bounding box around the space defined with the vector $T\boldsymbol{\alpha}_2$. Therefore, one can conclude that if all the 2^m combinations of $\boldsymbol{\tau}_o \notin [\boldsymbol{t}_{lb}, \boldsymbol{t}_{ub}]$, the system (14) cannot have a solution which satisfies $\boldsymbol{\alpha}_2 \in [\boldsymbol{0}, \boldsymbol{1}]$, and there is no need to invert T to figure it out.

D. Extension to residual polytopes

When evaluating the capability of a robot, it is a common practice to take into consideration the joint torques necessary to compensate for gravity $\boldsymbol{\tau}_g = \boldsymbol{g}(\boldsymbol{q})$ [16]

$$J(\boldsymbol{q})^T \boldsymbol{f} = \boldsymbol{\tau} - \boldsymbol{\tau}_g \tag{18}$$

Algorithm 1 New vertex search algorithm pseudo-code

Require: J, $\overline{\boldsymbol{\tau}}$, $\underline{\boldsymbol{\tau}}$ (Eq. 3b or Eq. 22)
  $U, \Sigma, V^T \leftarrow svd(J)$
  $J^{T+} = U\Sigma^{T+}V^T$
  V_1, $V_2 \leftarrow V$
  calculate n base vectors $\boldsymbol{\tau}_1 \dots \boldsymbol{\tau}_n$ (Eq. 7)
  for all $\binom{n}{m}$ combinations of m fixed α_i **do**
    construct T matrix (Eq. 14)
    find bounds $\boldsymbol{t}_{lb}, \boldsymbol{t}_{ub}$ (Eq. 17)
    for all 2^m vectors $\boldsymbol{\tau}_o$ **do**
      if $V_2^T \boldsymbol{\tau}_o \in [\boldsymbol{t}_{lb}, \boldsymbol{t}_{ub}]$ **then**
        $\boldsymbol{\alpha}_2 = T^{-1} V_2^T \boldsymbol{\tau}_o$
        if $\boldsymbol{\alpha}_2 \in [0, 1]$ **then**
          $\boldsymbol{\tau}_{vert} = \underline{\boldsymbol{\tau}} + \alpha_1 \boldsymbol{\tau}_1 + \cdots + \alpha_n \boldsymbol{\tau}_n$
          $\boldsymbol{f}_{vert} = J^{T+} \boldsymbol{\tau}_{vert}$
        end if
      end if
    end for
  end for

Furthermore, the same can be done to include the effects of the robot dynamics. For a fixed-base, serial manipulator the dynamics can be written as

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} - J(\boldsymbol{q})^T \boldsymbol{f} \tag{19}$$

where M and C are respectively the mass and Coriolis-Centrifugal matrices. The residual joint torque vector $\boldsymbol{\tau}$ can then be expressed as

$$J(\boldsymbol{q})^T \boldsymbol{f} = \boldsymbol{\tau} - \boldsymbol{\tau}_g - \big(\underbrace{M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}}}_{\boldsymbol{\tau}_d} \big) \tag{20}$$

Finally, in many cases it useful to evaluate the residual force ability of a robot already applying a certain nominal wrenches $\boldsymbol{f}_n$ which imply to generate joint torques $\boldsymbol{\tau}_n$

$$\boldsymbol{\tau}_n = J(\boldsymbol{q})^T \boldsymbol{f}_n \tag{21}$$

The *residual force polytope* [17] corresponding to the feasible space of Cartesian wrenches $\boldsymbol{f}$ that can be applied and rejected by the robot end-effector, taking into consideration the torque necessary to overcome gravity $\boldsymbol{\tau}_g$, robot dynamics $\boldsymbol{\tau}_d$ and any nominal joint torque $\boldsymbol{\tau}_n$, can be computed using algorithm 1 and the updated joint constraints

$$\begin{aligned} \underline{\boldsymbol{\tau}}' &= \underline{\boldsymbol{\tau}} - \boldsymbol{\tau}_g - \boldsymbol{\tau}_d - \boldsymbol{\tau}_n \\ \overline{\boldsymbol{\tau}}' &= \overline{\boldsymbol{\tau}} - \boldsymbol{\tau}_g - \boldsymbol{\tau}_d - \boldsymbol{\tau}_n \end{aligned} \tag{22}$$

IV. Experiments and results

In this section, the complexity of the proposed algorithm is evaluated for different robots and compared to state-of-the-art algorithms. Furthermore, a real-time control strategy for dual arm collaborative object handling using the proposed on-line polytope estimation is introduced and experimentally validated in section IV-B.

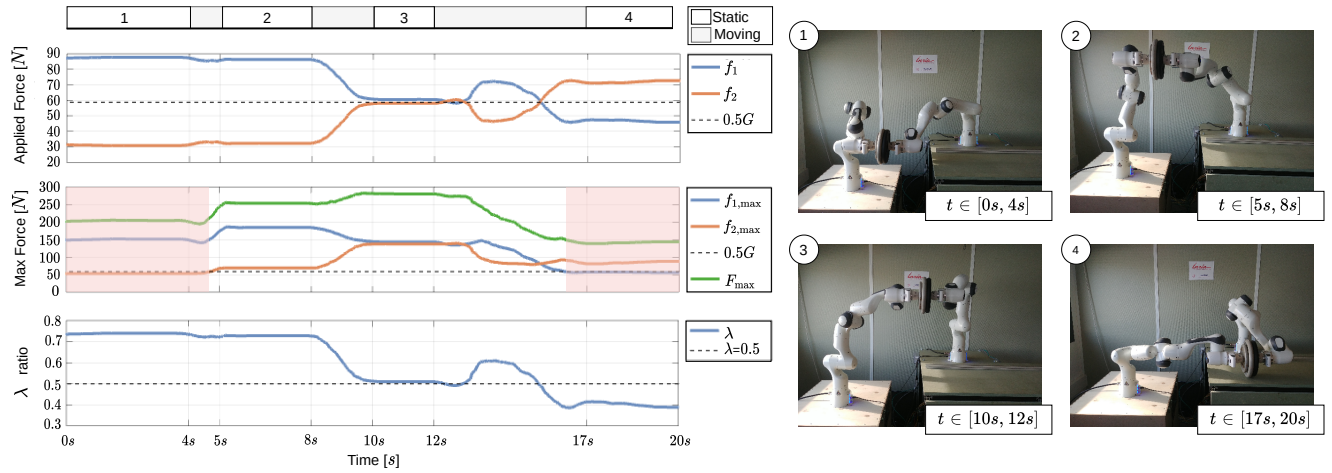

Fig. 4. Images 1 to 4 on the right show the sequence of poses the manipulators is placed by a human operator during the course of the experiment. On left side, on the top is the graph showing the time evolution of applied manipulator forces f_1, f_2. The middle graph provides the evolution of the maximal applicable forces $f_{i,max}$ of each manipulator separately and their joint capacity F_{max}, based on the proposed polytope evaluation algorithm. In this graph, the time periods when the constant strategy λ=0.5 (dashed line on all graphs) would not be feasible for at least one of the two robot are indicated with red background. The bottom graph shows the evolution in time of the control parameter λ for the proposed adaptive strategy.

A. Complexity analysis

To demonstrate the efficiency of the proposed polytope vertex search algorithm, it is compared to the polytope vertex search algorithm introduced by Chiacchio et al. [5]. Furthermore the comparison is extended to the algorithm proposed by Sasaki et al. [14] which is, to our knowledge, the only algorithm exploiting the geometric structure of the problem.

The three algorithms have been tested to find vertices of the force polytope $\mathcal{P}_f$ for three different robots: a 4R planar robot (n=4,m=2), the *Universal Robots* UR5 6DoF robot (n=6,m=3) and the *Franka Emika Panda* 7DoF robot (n=7,m=3). The results are averaged over 1000 randomly selected robot configurations. All algorithms have been implemented in the programming language Python and tested on a laptop equipped with a 1.90GHz Intel i7-8650U processor.

Table I shows the results of this complexity evaluation. The proposed vertex search algorithm drastically reduces the number of matrix inversions and thus reduces the processing time considerably: 4-10$\times$ faster execution than Chiacchio's algorithm and on average 20% faster than Sasaki's. Results show that even for the cases of 6DoF and 7DoF industrial robots our approach is capable of evaluating the polytope vertices under $3ms$. Such a low processing time opens numerous opportunities for the online use of polytope based capacity evaluation especially in the area of robot control.

B. Dual arm collaborative object carrying

In this section, the real-time force capability evaluation is employed in the context of robot control. An experiment is designed using two *Franka Emika Panda* robots involved in a collaborative load carrying task for an object of mass M. Each robot is contributing to the compensation of the overall gravity force $G = Mg$ applied at the end-effector by the carried object, with forces f_1 and f_2 respectively such that

$$f_1 + f_2 = G \tag{23}$$

ICRA2021 Additionally, a human operator is moving the object freely through the common work-space, implicitly changing both robots positions and configurations in real-time. As a consequence, the task space trajectory and the evolution of the robot configurations are not known in advance.

The experiment has been implemented using the ROS programming environment. Both robots are torque controlled at $1kHz$. The update of the force capability estimation are performed at $40Hz$. All the control software is run from one computer with the 1.90GHz Intel i7-8650U processor. This experiment is illustrated in the accompanying video.

Panda robots are rated to a maximal carrying capacity of 3kg which corresponds to the absolute minimal carrying capacity of the robot evaluated in one of its near-singular configurations. It is, by definition, an underestimation of the real robot task space wrench capacity and relying on it, as it is commonly done, limits the scope of the possible tasks and applications.

The goal of this experiment, it is to demonstrate the fact that, taking into account the true force capabilities of each robot, it is possible to considerably go beyond the robots conservative rated capacity without comprising safety nor exceeding any of the actuation limits. The weight of the object chosen for the experiments is $M = 12kg$, voluntarily far above the recom-

TABLE I

COMPLEXITY AND EXECUTION TIME COMPARISON FOR THREE DIFFERENT VERTEX SEARCH ALGORITHMS.

Robot	**Chiacchio**[5]	**Sasaki** [14]	**Proposed**
(n, m)	inversions: $\frac{2n!}{m!(2n-m)!}$ mat. size: $2n \times 2n$ time[ms]: t $\pm$ sd (max)	$\frac{n!}{m!(n-m)!}$ $m \times m$	$\leq \frac{n!}{m!(n-m)!}$ n-$m \times n$-m
$(4, 2)$	24 8×8 2.7±0.1 (3.6)	6 2×2 0.93±0.1 (1.1)	4.2±1.4 (6) 2×2 0.64±0.1 (1.5)
$(6, 3)$	220 12×12 14.2±0.9 (20)	20 3×3 2.1±0.2 (3.4)	2.8±2.5 (10) 3×3 1.5±0.2 (2.7)
$(7, 3)$	364 14×14 25±0.5 (27)	35 3×3 3.5±0.1 (5.4)	7.3±4.2 (20) 4×4 2.6±0.2 (4.3)

mended joint carrying capacity $M \leq 6kg$.

Evaluating the maximal applicable force in the vertical direction can be viewed as a special case of the force polytope $\mathcal{P}_f$ vertex search problem where the task-space force of interest f is a scalar and $m = 1$. The proposed algorithm described in section III is used to efficiently find $f_{i,max}$, the limits (vertices) of the maximal applicable force in z-axis direction, for each robot separately. Their overall carrying capacity can be calculated as the sum of the two $F_{max} = f_{1,max} + f_{2,max}$.

To efficiently distribute and adapt the carried load in between robots, each robot compensate for a part of the object's weight G which is proportional to its force capacity $f_{i,max}$

$$f_1 = \lambda G \quad f_2 = (1-\lambda)G, \quad \lambda = \frac{f_{1,max}}{f_{1,max} + f_{2,max}} \tag{24}$$

A straightforward control strategy, requiring each robot to compensate for half of object mass $f_1 = f_2 = 0.5G$ is taken as baseline to evaluate the performance of the new method.

Figure 4 (right) shows one possible manipulation trajectory, where a human operator displaces the carried object in the common workspace in four discrete locations over a 20 seconds period of time. On the left side of the figure 4 the evaluated maximal forces and force applied by each robot during the experiment are shown. Additionally, the $0.5G$ strategy line is shown for comparison.

The proposed control strategy is successful in ensuring compensation of the object weight during the full length of the experiment. In the starting pose $t = 0s$, robot 2 (robot on the left) is close to it's singular configuration and its load carrying capacity is close to 5kg. Controlling it to compensate for half of the weight of the object would result in a security exception and could damage the robot hardware. The same is true for pose 4 at $t = 17s$. Robot 1 (robot on the right) is not be able to compensate for half of the object weight but thanks to the proposed adaptive control law, the task can successfully be achieved. This may not be the case over the entire common workspace of the two robots but this example is a good illustration of the interest of accounting for the true capabilities of the system at the control level. An additional interesting result from the experiment is the transition between pose 3 and 4 ($t \in [12s, 16s]$). Even though the object is much closer to robot 2 (right), robot 1 has a much higher carrying capacity. This illustrates the fact that force capabilities are highly nonlinear functions which are very hard to predict. This provides an additional argument in favor of on-line force capacity evaluation.

V. Discussions

In this section, the potential of applying the on-line capacity evaluation for collaborative robots and the human-robot interaction is discussed.

A. Polytopes for multi-robot systems

Up to recently [18], the estimation of task-space capabilities of multiple robotic manipulators was largely depending on manipulability ellipsoids [8][19][7].

However as the number of robots grows, the ellipsoid approach, which rely on a Euclidean norm rather than an infinity norm [4], provides a less accurate approximation and the questions of what is the best practice to sum [8][9] or intersect [20][7] multiple ellipsoids becomes problematic.

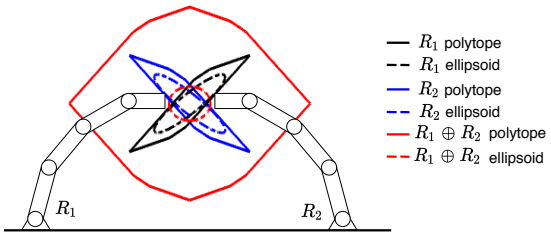

Fig. 5. 4DoF robots collaboration: polytopes and ellipsoids comparison

Figure 5 shows one example of two collaborating 4DoF planar robots joint force capacity based on ellipsoids and polytopes. The ellipsoids are calculated using the approach proposed by Chiacchio [8] and the joint polytope is calculated as the Minkowski sum of the individual force polytope of each robot.

$$\mathcal{P}_{f_\oplus} = \mathcal{P}_{f_{R_1}} \oplus \mathcal{P}_{f_{R_2}} \tag{25}$$

This example clearly demonstrates that even though ellipsoids can reasonably well provide an approximation of the force capacity of each robot separately, the same is not the case for the capacity of the dyad.

A similar conclusion can be drawn for use cases where the joint capacity of multiple robots can be represented by the intersection of their polytopes [18], *i.e* when the robots are working in opposition, both applying a similar force $\boldsymbol{f}$ in opposite directions[4]

$$\mathcal{P}_f = \mathcal{P}_{f_{R_1}} \cap \mathcal{P}_{f_{R_2}} \tag{26}$$

In that case, the polytope intersection problem can be elegantly transformed into the vertex enumeration problem for the extended system

$$J_\cap = [J_{R_1}\, J_{R_2}], \quad \boldsymbol{\tau} = \left[\boldsymbol{\tau}_{R_1}^T\, \boldsymbol{\tau}_{R_2}^T\right]^T, \quad J_\cap^T \boldsymbol{f} = \boldsymbol{\tau} \tag{27}$$

B. Polytopes for human-robot collaboration

Several authors have studied the use of robotics performance measures such as manipulability ellipsoids and polytopes for human arms using 7DoF manipulator models [21][22][23][24] or full musculoskeletal model [25]. A promising approach for leveraging the efficient on-line polytope evaluation algorithm is to estimate the joint capacity of the robot and human and adapt the robotic assistance as a function of the evolution of human capabilities. One of the associated challenges is to extend the proposed vertex search algorithms to actuation constraints which result from more complex actuation mechanisms such as muscles and can thus no longer be represented as parallelotops.

VI. Conclusion

The proposed vertex search algorithm provides a computationally efficient tool for the polytope-based, online evaluation of task-space robot capabilities. As illustrated in this work, the ability to accurately evaluate on-line the capabilities of robots leverages several possibilities in modern robotic applications where oversizing robots is no longer an option.

[4]A typical example of this situation is the case of the frictional grasp of an object.

REFERENCES

[1] J. Angeles and F. C. Park, "Design and Performance Evaluation," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), Springer Handbooks, pp. 399–418, Cham: Springer International Publishing, 2016.

[2] T. Yoshikawa, "Manipulability of Robotic Mechanisms," *The International Journal of Robotics Research*, vol. 4, pp. 3–9, June 1985.

[3] J. Kuffner and J. Xiao, "Motion for Manipulation Tasks," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), Springer Handbooks, pp. 897–930, Cham: Springer International Publishing, 2016.

[4] J. P. Merlet, "Jacobian, Manipulability, Condition Number, and Accuracy of Parallel Robots," *Journal of Mechanical Design*, vol. 128, pp. 199–206, Jan. 2006.

[5] P. Chiacchio, Y. Bouffard-Vercelli, and F. Pierrot, "Evaluation of force capabilities for redundant manipulators," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, (Minneapolis, MN, USA), pp. 3520–3525, 1996.

[6] R. Finotello, T. Grasso, G. Rossi, and A. Terribile, "Computation of kinetostatic performances of robot manipulators with polytopes," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3241–3246, May 1998.

[7] S. Lee, "Dual redundant arm configuration optimization with task-oriented dual arm manipulability," *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 78–97, Feb. 1989.

[8] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Task Space Dynamic Analysis of Multiarm System Configurations," *International Journal of Robotic Research*, vol. 10, pp. 708–715, Dec. 1991.

[9] J. C. Frantz, L. Mejia, H. Simas, and D. Martins, "Analysis of wrench capability for cooperative robotic systems," in *Proceeding in 23rd ABCM International Congress of Mechanical Engineering*, 2015.

[10] D. Avis and K. Fukuda, "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra," *Discrete & Computational Geometry*, vol. 8, no. 3, pp. 295–313, 1992.

[11] D. Avis and C. Jordan, "Comparative computational results for some vertex and facet enumeration codes," *arXiv preprint arXiv:1510.02545*, 2015.

[12] S. B. Nokleby, R. Fisher, R. P. Podhorodeski, and F. Firmani, "Force capabilities of redundantly-actuated parallel manipulators," *Mechanism and Machine Theory*, vol. 40, pp. 578–599, May 2005.

[13] M. Gouttefarde, J. Lamaury, C. Reichert, and T. Bruckmann, "A Versatile Tension Distribution Algorithm for n-DOF Parallel Robots Driven by $n + 2$ Cables," *IEEE Transactions on Robotics*, vol. 31, pp. 1444–1457, Dec. 2015.

[14] M. Sasaki, T. Iwami, K. Miyawaki, I. Sato, G. Obinata, and A. Dutta, "Vertex search algorithm of convex polyhedron representing upper limb manipulation ability," *Search Algorithms and Applications*, 2011.

[15] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Transactions on Automatic Control*, vol. 25, pp. 164–176, Apr. 1980.

[16] B. Wei and F. Gao, "Output force capacity polytope approach for actuator forces selection of three degrees of freedom excavating manipulator," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 228, no. 11, pp. 2007–2017, 2014.

[17] H. Ferrolho, W. Merkt, C. Tiseo, and S. Vijayakumar, "Comparing metrics for robustness against external perturbations in dynamic trajectory optimization," *arXiv preprint arXiv:1908.05380*, 2019.

[18] P. Long and T. Padir, "Constrained manipulability for humanoid robots using velocity polytopes," *International Journal of Humanoid Robotics*, vol. 17, no. 1, 2020.

[19] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Global task space manipulability ellipsoids for multiple-arm systems," *IEEE Transactions on Robotics and Automation*, vol. 7, pp. 678–685, Oct. 1991.

[20] H. Sekiguchi and K. Ohnishi, "Force capability evaluation methods for bilateral controlled manipulators," in *2017 IEEE International Conference on Mechatronics (ICM)*, pp. 111–116, IEEE, 2017.

[21] N. Rezzoug, J. Jacquier-Bret, V. Hernandez, and P. Gorce, "Application of robotic indices to evaluate human upper-limb force capacities," in *IECON 38th Annual Conference on IEEE Industrial Electronics Society*, pp. 5568–5573, 2012.

[22] M. Sasaki, T. Iwami, K. Miyawaki, I. Sato, G. Obinata, and A. Dutta, "Higher dimensional spatial expression of upper limb manipulation ability based on human joint torque characteristics," in *Robot Manipulators New Achievements* (A. Lazinica and H. Kawai, eds.), ch. 36, Rijeka: IntechOpen, 2010.

[23] M. G. Carmichael and D. Liu, "Estimating physical assistance need using a musculoskeletal model," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 7, pp. 1912–1919, 2013.

[24] M. G. Carmichael and D. Liu, "Towards using musculoskeletal models for intelligent control of physically assistive robots," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 8162–8165, 2011.

[25] V. Hernandez, N. Rezzoug, and P. Gorce, "Toward isometric force capabilities evaluation by using a musculoskeletal model: Comparison with direct force measurement," *Journal of Biomechanics*, vol. 48, pp. 3178–3184, Sept. 2015.