Optimal Force Allocation for Overconstrained Cable-Driven Parallel Robots: Continuously Differentiable Solutions with Assessment of Computational Efficiency

Einar Ueland*, Thomas Sauder†, Roger Skjetne*,
*Department of Marine Technology; Norwegian University of Science and Technology; Centre for Autonomous Marine Operations and Systems (NTNU AMOS); NO-7491 Trondheim, Norway
†SINTEF Ocean; NO-7465 Trondheim, Norway

***Abstract*—We present a novel method for force allocation for overconstrained CDPR setups that guarantees continuously differentiable cable forces and allows for small penalised errors in the resulting wrench. For the latter, we also provide a bound on the error under some assumptions. We study real-time feasibility by performing numerical simulations on a large set of configurations.**

***Index Terms*—CDPR, force allocation, slack formulation, continuously differentiable solution, Newton's method, real-time feasibility.**

I. Introduction

A cable-driven parallel robot, hereafter referred to as a CDPR, is a mobile platform driven by forces actuated through a set of cables in a parallel topology. Recognised for their large workspace, lightweight structure, fast dynamics, and reconfigurability [1–3], CDPR setups have received significant attention in the last decades [4, 5] with studied cases including diverse applications such as aerial cameras [6], manufacturing [7], and hydrodynamic model testing [8].

This paper considers the *force allocation* problem of distributing a set of lower and upper bounded pull forces in the individual cables on an overconstrained CDPR setup (also referred to as *overactuated* in the control literature), such that the resulting forces and moments match the desired reference wrench. Within the CDPR literature, *force distribution* [5] or *tension distribution* [9] is also used to mean the same. The problem is an important component for the kinetic control of CDPR platforms [10, 11]. For a comprehensive overview of CDPR, including other topics, see [5].

For overconstrained CDPR setups, the force allocation problem is underdetermined and typically solved as an optimisation problem [4]. The cost function is often the 2-norm of the pull forces, which is frequently solved using the pseudoinverse due to its computational simplicity [5]. A drawback of this method is its inability to handle inequality constraints on the forces. In [12], this is handled by minimising the 2-norm about the medium force value, whilst the *redistributive pseudoinverse* method iteratively saturates the actuator forces, with the use of the pseudoinverse in each step [13]. Importantly, however, one generally cannot guarantee continuity of the resulting forces using simple saturated pseudoinverse techniques [14].

Kernel-based methods [15] cast the problem as a combination of a particular and a nullspace solution. In cases with one more actuator than controlled DOFs, the solution minimising the 2-norm is here found by an explicit expression. Although the complexity of the method quickly increases with the number of actuators [9], it is still relevant in practice, since the most common setups appear to have either one (for planar CDPR) or two (for 6-DOF CDPR) more actuators than controlled DOFs [9].

Pseudoinverse- and kernel-based methods can often run in deterministic time with proven worst case computational complexity (see for example [9]). Potential drawbacks are that they either cannot handle a higher number of cables, are restricted to simple cost functions, or reduce the workspace to less than what is maximally feasible [16]. Iterative methods, allowing for more complex cost functions, have generally not been preferred due to lack of real-time guarantees (e.g., as discussed in [3, 5, 16, 17]). Of iterative methods guaranteeing an optimal solution, quadratic programming solvers have been popular; see for example [2]. Although quadratic programming under certain conditions has been reported as real-time feasible [13], limited information, has been presented on the real-time feasibility of iterative methods for CDPR applications [5].

Corresponding author: E. Ueland (email: einar.s.ueland@gmail.com).

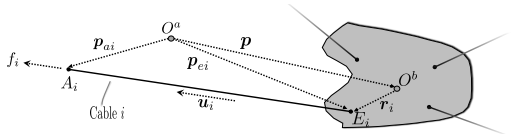

Fig. 1: A CDPR platform, with cable i highlighted.

Beyond CDPR applications, variations of the force allocation problem have been studied extensively both within air vehicle control [18, 19], underwater vehicle control [20], and marine vessel control [21, 22]. Within these fields, continuity of solutions and computational efficiency are important. Several methods similar to those developed for CDPR have also been investigated in other fields [23, 24]. Similar problems also arise in other contexts [14] such as model predictive control [25] and control of multilegged vehicles [26]. A slacked version of the force allocation problem for CDPR, which is a framework where small errors in the resulting wrench are allowed [19], has been considered in [27].

The contributions of this paper are:
1) The optimal force allocation problem for CDPR is analysed, and a new cost function is proposed for the standard version of the problem, ensuring $\mathcal{C}^1$ continuity of actuator forces (*Section III*).
2) A new cost function for the slacked version of the problem is also proposed. It ensures that the slack remains close to zero when needed. We derive an upper bound for the slack error under certain assumptions. This allows for slack to compensate for errors in the force equilibrium when feasibility is not achieved, making the algorithm more robust (*Section III*).
3) We conjecture through extensive numerical simulations that a solver based on the Newton's method is feasible for use in real-time applications. (*Section IV*).
4) The code for all presented results and methods is made accessible (*electronic attachment*).

II. Problem Formulation

A. Optimisation problem

Using Fig. 1 as a reference, we outline the kinematic relationship between the actuator force vector $\boldsymbol{f} \in \mathbb{R}^n$ and the global wrench $\boldsymbol{w} \in \mathbb{R}^m$, where n is the number of connected actuators and m is the number of controlled DOFs of the platform. Using the notation of [28, Ch 2.2.1], with superscript $(\cdot)^{\{a\}}$ and $(\cdot)^{\{b\}}$ denoting earth-fixed (stationary) and body-fixed coordinate frames, respectively, the platforms pose vector is $\boldsymbol{\eta} := (\boldsymbol{p}, \boldsymbol{\Theta}) \in \mathbb{R}^3 \times \mathcal{S}^3_1$, where $\boldsymbol{p} := \boldsymbol{p}^a = (x, y, z)$ and $\boldsymbol{\Theta} := (\phi, \theta, \psi)$ is the platform body pose and orientation, respectively. For each actuator $i \in \{1, \dots, n\}$, let $\boldsymbol{p}^a_{ai}$ be the fixed position of the i^{th} cable exit point A_i. Similarly, let the constant body-fixed lever arm from O^b to the i^{th} cable attachment anchor E_i (on the platform) be denoted $\boldsymbol{r}^b_i$. It follows that the absolute position of E_i is $\boldsymbol{p}^a_{ei} = \boldsymbol{p} + \boldsymbol{R}(\boldsymbol{\Theta})\boldsymbol{r}^b_i$, where $\boldsymbol{R}(\boldsymbol{\Theta})$ is the Euler angle rotation matrix. We assume that from each actuator i, a force f_i directed along the straight line $\boldsymbol{p}_{ai} - \boldsymbol{p}_{ei}$, with direction denoted by the unit vector $\boldsymbol{u}_i := \frac{\boldsymbol{p}_{ai} - \boldsymbol{p}_{ei}}{|\boldsymbol{p}_{ai} - \boldsymbol{p}_{ei}|}$ is actuated on the platform at E_i.

The relationship between the cable forces $\boldsymbol{f} = (f_1, f_2, \cdots, f_n)$ and the resultant wrench $\boldsymbol{w}$ applied by the cables is described by $\boldsymbol{w}(t) = \boldsymbol{W}(t)\boldsymbol{f}(t)$, where

$$\boldsymbol{W}(t)=\begin{bmatrix}\boldsymbol{q}_1(t) & \boldsymbol{q}_2(t) & \cdots & \boldsymbol{q}_n(t)\end{bmatrix}, \text{ with } \boldsymbol{q}_i(t)=\begin{bmatrix}\boldsymbol{u}_i(t)\\ \boldsymbol{r}_i(t)\times\boldsymbol{u}_i(t)\end{bmatrix}, \tag{1}$$

where $\boldsymbol{r}_i(t)=\boldsymbol{r}_i^a(t)=\boldsymbol{R}(\boldsymbol{\Theta}(t))\boldsymbol{r}_i^b$. Since the pose $\boldsymbol{\eta}(t)$ varies with time, it follows that $\boldsymbol{u}_i(t)$ and $\boldsymbol{r}_i(t)$ are time-varying signals, and hence also $\boldsymbol{W}(t)$. Hereafter, we will often denote vectors with their time dependency (t) omitted.

For safety concerns, constraints are imposed on each actuator. A minimum force limit $f_{i,\min}$ is set to prevent the cable from losing tension, while a maximum force limit $f_{i,\max}$ is set due to actuator or cable limitations. This constraint is formulated as;

$$\boldsymbol{h}(\boldsymbol{f})\succcurlyeq 0, \text{ with } \boldsymbol{h}(\boldsymbol{f})=\begin{bmatrix}\boldsymbol{f}-\boldsymbol{f}_{\min}\\ \boldsymbol{f}_{\max}-\boldsymbol{f}\end{bmatrix} \tag{2}$$

where the symbol $\succcurlyeq$ denotes component-wise inequality. For the l^{th}-component in (2), a force-constraint $h_l(\boldsymbol{f})\in\boldsymbol{h}(\boldsymbol{f})$ is said to be *active* if the corresponding force is fixed at the constraint (that is $h_l(\boldsymbol{f})=0$).

By introducing a slack variable $\boldsymbol{s}$, and generally allowing for some penalised errors $\boldsymbol{w}_{\text{err}}=\boldsymbol{Q}\boldsymbol{s}$ in the resulting wrench, the problem of tracking a reference wrench $\boldsymbol{w}_{\text{ref}}$ is formulated as the following optimisation problem;

$$(\boldsymbol{f}_s^*,\boldsymbol{s}^*)=\underset{\boldsymbol{f},\boldsymbol{s}}{\operatorname{argmin}}\ \underbrace{g_f(\boldsymbol{f})+g_s(\boldsymbol{s})}_{g(\boldsymbol{f},\boldsymbol{s})} \tag{3a}$$

$$\text{subject to}\quad \boldsymbol{W}\boldsymbol{f}+\boldsymbol{Q}\boldsymbol{s}=\boldsymbol{w}_{\text{ref}},\quad \boldsymbol{h}(\boldsymbol{f})\succcurlyeq 0 \tag{3b}$$

where $\boldsymbol{f}_s^*$ and $\boldsymbol{s}^*$ is the solution to (3b) that minimise $g(\boldsymbol{f},\boldsymbol{s})$, with $g_f(\boldsymbol{f})$ and $g_s(\boldsymbol{s})$ being the cost-function of the cable-forces and slack variable, respectively. The matrix $\boldsymbol{Q}\in\mathbb{R}^{m\times q}$ scales the penalty on the slack variable $\boldsymbol{s}\in\mathbb{R}^q$. Hence, the resulting wrench is given by $\boldsymbol{w}=\boldsymbol{W}\boldsymbol{f}_s^*=\boldsymbol{w}_{\text{ref}}-\boldsymbol{Q}\boldsymbol{s}^*$. With $q=m$, $\boldsymbol{Q}$ is diagonal. However, one can eliminate the slack and corresponding elements of $\boldsymbol{Q}$ in any DOFs where strict equality is required, such that $q<m$. In particular, $\boldsymbol{Q}\boldsymbol{s}$ and $g_s(\boldsymbol{s})$ can be null, resulting in the *standard formulation* [29]:

$$\boldsymbol{f}^*=\underset{\boldsymbol{f}}{\operatorname{argmin}}\ g(\boldsymbol{f}) \tag{4a}$$

$$\text{subject to}\quad \boldsymbol{W}\boldsymbol{f}=\boldsymbol{w}_{\text{ref}},\quad \boldsymbol{h}(\boldsymbol{f})\succcurlyeq 0. \tag{4b}$$

We say that the CDPR configuration is *wrench feasible* when (4b) is feasible. By using (3), with $q\neq 0$, hereafter referred to as the *slacked formulation*, situations outside the wrench-feasible workspace can be handled. Slack will increase flexibility and robustness. We recognise, however, that it is only relevant in a subset of applications. For this reason, we consider the two formulations separately.

Since, in practice, $\boldsymbol{w}_{\text{ref}}$ and $\boldsymbol{W}$ are provided discretely by a sampled control system, (3) and (4) will be solved in each time-step. For safe and well-behaved performance, we require both that solutions are continuous and that solutions are *real-time feasible*, as recognised by several authors [9, 16].

B. A motivating example

To illustrate the proposed methods, we use a test case of an earlier platform [5, 12] supported by eight symmetrically placed actuators, as shown in Fig. 2, with geometry, parameters and trajectories given in Table I (end of the paper). Of interest here is the resulting actuator force trajectories, as well as the real-time capabilities of the solvers. Other aspects, such as the dynamics of the platform, control laws, and cable-cable interference (collision) are not considered.

Fig. 2(b) presents the resulting actuator forces using both the pseudoinverse method (in blue) and a quadratic solver (in black). Two distinct issues are identified as indicated.

Issue A. Wrench feasibility is lost, that is, (4) has no solution. These cases have not received much attention within CDPR applications, perhaps due to high demands on force accuracy, safety concerns, or because most CDPR setup by design operates well within the wrench-feasible workspace. We argue for the practical relevance of this issue because: 1) in certain applications it is advantageous that the CDPR system can handle wrench-values beyond the wrench-feasible workspace. This is recognised as particularly important for haptic interfaces [27]; 2) allowing for penalised errors in specified low priority DOFs that do not cause any loss of fidelity [30] enhances the robustness of the allocation; and 3) depending on the setup, there may be sections in the operating workspace that are not wrench-feasible due to low controllability (or singularities) [15].

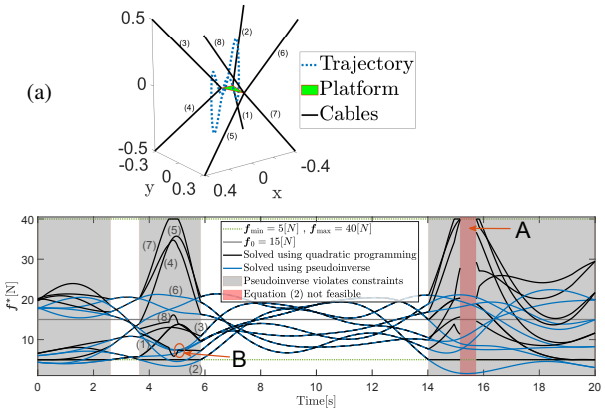

Fig. 2: Motivating example. (a) Trajectory and configuration. (b) Resulting forces. Identifying two distinct issues (A-B).

Issue B. The p-norm cost functions can result in nonsmooth forces, as seen by the quadratic solver ($p=2$) of Fig. 2(b). Whether non-smooth control inputs are an issue depend on factors such as the actuator technology, cable elasticity and tension regulation method.

The issues are particularly relevant to the authors ongoing research project, where CDPR setups are used for marine model testing [8]. Here, a fixed and standardised actuator configuration along the basin walls is planned for use on multiple platforms. The extent of the wrench-feasible workspace will then limit the applicability of the setup. By allowing penalised errors (Issue A) in low prioritised DOFs, flexibility and robustness of the basin specific standardised setup can be increased. Moreover, since accurate force-control is of particular importance [8], the commanded forces are expected to be more precisely tracked by the actuators when the force trajectories are differentiable (Issue B).

C. Problem statement

We will address Issue A by applying the slacked formulation and Issue B by providing theorems and cost functions such that the resulting allocated actuator forces are guaranteed to be $\mathcal{C}^1$. We define *real-time feasibility* as the ability to produce solutions according to specified time limits with sufficiently high probability. We shall design a resolution method for (3) and (4) that is *real-time feasible* within one control cycle of typically a few milliseconds.

III. Cost functions and continuity

A. Continuity conditions

These assumptions are referred to in the remainder of the paper;

A.1 *The CDPR setup is overconstrained ($n>m$) and $\boldsymbol{W}(t)$ has full row-rank for each $t\geq 0$.*

A.2 *The set of equality and inequality constraints in* (4b) *has at least one solution for each $t\geq 0$ (wrench feasibility).*

A.3 *The maps $t\mapsto\boldsymbol{W}(t)$ and $t\mapsto\boldsymbol{w}_{ref}(t)$ are continuously differentiable for each $t>0$.*

A.4 *There exists a $\delta>0$ such that* (4b) *has at least one solution with $\boldsymbol{h}(\boldsymbol{f}(t))\succcurlyeq\delta$ for each $t\geq 0$ (wrench feasibility margin).*

A.5 *The gradients of the equality constraints and of the active inequality constraints in* (4b) *are linearly independent at the solution $\boldsymbol{f}^*(t)$ for each $t\geq 0$.*

Since $\boldsymbol{w}_{\text{ref}}(t)$ is typically generated by a reference model, it is assumed to be sufficiently smooth by construction. Also, $\boldsymbol{W}(t)=\boldsymbol{W}(\boldsymbol{\eta}(t))$ is continuously differentiable by integration of the platform body dynamics $\ddot{\boldsymbol{\eta}}$, so A.3 is in general fulfilled.

For any problem that holds under A.2, one only needs to adjust lower- and upper constraint limits by δ to make it also hold under A.4, hence adding few additional restrictions in practice.

A.5 is known as the linear independence constraint qualification (LICQ) in numerical optimisation. It holds if we assert that the remaining block matrix of $\boldsymbol{W}$ is full row-rank after removing active-constraint columns. In certain situations, active constraints combined with parallel lines could potentially cause A.5 not to hold. However, later in this paper, we will ensure that no constraints are active so that A.5 holds under A.4 and A.1.

The following proofs[1] are for brevity only shown for the standard formulation. However, the reader should note that with small adaptations, the same can be shown for the slacked formulation by using

$$\boldsymbol{x}=\begin{bmatrix}\boldsymbol{f}^\top & \boldsymbol{s}^\top\end{bmatrix}^\top,\quad \boldsymbol{A}=\begin{bmatrix}\boldsymbol{W} & \boldsymbol{Q}\end{bmatrix},\quad g(\boldsymbol{x})=g_f(\boldsymbol{f})+g_s(\boldsymbol{s}), \tag{5}$$

and replacing $\boldsymbol{f}$ with $\boldsymbol{x}$ and $\boldsymbol{W}$ with $\boldsymbol{A}$ where appropriate in the following proofs.

Considering the standard formulation (4), and introducing the notion of convexity [34], the following results hold for the time-continuous case $\boldsymbol{f}^*(t)$:

Theorem 1 *Under A.1-A.3, if $\boldsymbol{f}\mapsto g(\boldsymbol{f})$ is a continuous and strictly convex function, then $t\mapsto\boldsymbol{f}^*(t)$ of* (4) *is a continuous function.*

Proof of Theorem 1: We simply refer to the maximum theorem [33, p. 116]. In short, with $\Omega(t)=\{\boldsymbol{f}\in\mathbb{R}^n\colon \boldsymbol{W}(t)\boldsymbol{f}=\boldsymbol{w}_{\text{ref}}(t),\ \boldsymbol{h}(\boldsymbol{f})\succcurlyeq 0\}$ being the feasible set, $t\mapsto\Omega(t)$ is compact (affine constraints) such that $\Omega(t)\neq\emptyset$ (assumption of feasibility) for each $t\geq 0$. Therefore $\Omega:\mathbb{R}_{\geq 0}\rightrightarrows\mathbb{R}^n$ is a compact valued correspondence. With $\boldsymbol{W}(t)$ full row rank, $\Omega(t)$ is a continuous correspondence (implicit function theorem). Since also $g(\boldsymbol{f})$ is continuous, the conditions of the referenced theorem are satisfied, such that $\boldsymbol{f}^*(t)$ is lower semicontinuous. By strict convexity (with affine constraints) the components of $\boldsymbol{f}^*(t)$ are single-valued, and $\boldsymbol{f}^*(t)$ is a continuous vector-valued function. ■

Theorem 2. *Under A.1-A.4, if $\boldsymbol{f}\mapsto g(\boldsymbol{f})$ is $\mathcal{C}^2$ and strongly convex, then $t\mapsto\boldsymbol{f}^*(t)$ of* (4) *is a piecewise $\mathcal{C}^1$ function with discontinuities in $\frac{d}{dt}\boldsymbol{f}^*(t)$ only appearing where any of the constraints in $\boldsymbol{h}(\boldsymbol{f}^*)$ shifts between being active and inactive.*

Proof of Theorem 2: Since the cost function of (4) is a C^2 convex function subject to affine constraints, the first-order necessary optimality conditions are met [35, Remark 2.2]. This condition implies that if $\boldsymbol{f}=\boldsymbol{f}^*$ is a minimiser, then there at time t exists some $\boldsymbol{\lambda}\in\mathbb{R}^m$ and $\boldsymbol{\mu}\in\mathbb{R}^{2n}$ satisfying the KKT conditions [36]:

$$\boldsymbol{R}(\boldsymbol{z},t)=\begin{bmatrix}\nabla_{\boldsymbol{f}}g(\boldsymbol{f})+\boldsymbol{W}(t)^\top\boldsymbol{\lambda}+\nabla_{\boldsymbol{f}}\boldsymbol{h}^\top(\boldsymbol{f})\boldsymbol{\mu}\\ \boldsymbol{W}(t)\boldsymbol{f}-\boldsymbol{w}_{\text{ref}}(t)\\ \operatorname{diag}(\boldsymbol{\mu})\boldsymbol{h}(\boldsymbol{f})\end{bmatrix}=\boldsymbol{0} \tag{6a}$$

$$\boldsymbol{h}(\boldsymbol{f})\succcurlyeq 0,\qquad \boldsymbol{\mu}\succcurlyeq 0 \tag{6b}$$

where $\boldsymbol{z}=\begin{bmatrix}\boldsymbol{f}^\top,\boldsymbol{\lambda}^\top,\boldsymbol{\mu}^\top\end{bmatrix}^\top\in\mathbb{R}^{3n+m}$, and $\nabla_{\boldsymbol{f}}\boldsymbol{h}^\top(\boldsymbol{f})=\begin{bmatrix}\boldsymbol{I} & -\boldsymbol{I}\end{bmatrix}$.

By strict convexity, the second-order sufficient condition is met [35, Theorem 2.4], which implies that any solution satisfying (6) is a local minimiser. By Assumption 2, with $g(\boldsymbol{f})$ continuous, a well-defined minimiser $\boldsymbol{f}^*$ exists, and by strict convexity, it must be unique. Let $\boldsymbol{h}_{\mathbb{A}}$ be those inequality constraints corresponding to strictly active constraints in (6a), and $\boldsymbol{\mu}_{\mathbb{A}}$ the corresponding elements in $\boldsymbol{\mu}$, that is, $h_i=0,\ \mu_i>0\implies\mu_i\in\boldsymbol{\mu}_{\mathbb{A}},\ h_i\in\boldsymbol{h}_{\mathbb{A}}$. Moreover let $\boldsymbol{z}_{\mathbb{A}}=\operatorname{col}(\boldsymbol{f},\boldsymbol{\lambda},\boldsymbol{\mu}_{\mathbb{A}})\in\mathbb{R}^{n+m+n_a}$ (where n_a is the number of active constraints) and let $\boldsymbol{R}_{\mathbb{A}}(\boldsymbol{z}_{\mathbb{A}},t)$ equals (6a) with rows containing inactive constraints along with the corresponding components in $\nabla_{\boldsymbol{f}}\boldsymbol{h}(\boldsymbol{f})$ removed. For a fixed strictly active set $\boldsymbol{h}_{\mathbb{A}}$, $\boldsymbol{R}_{\mathbb{A}}(\boldsymbol{z}_{\mathbb{A}},t)=0$, defines $\boldsymbol{z}_{\mathbb{A}}$ implicitly by

$$\frac{\partial\boldsymbol{z}_{\mathbb{A}}}{\partial t}=-\left(\frac{\partial\boldsymbol{R}_{\mathbb{A}}}{\partial\boldsymbol{z}_{\mathbb{A}}}\right)^{-1}\frac{\partial\boldsymbol{R}_{\mathbb{A}}}{\partial t}. \tag{7}$$

$\boldsymbol{R}_{\mathbb{A}}(\boldsymbol{z}_{\mathbb{A},t})$ is continuously differentiable w.r.t. $\boldsymbol{z}_{\mathbb{A}}$, and;

$$\frac{\partial\boldsymbol{R}_{\mathbb{A}}(\boldsymbol{z}_{\mathbb{A}},t)}{\partial\boldsymbol{z}_{\mathbb{A}}}=\begin{bmatrix}\nabla^2_{\boldsymbol{f}}\boldsymbol{g}(\boldsymbol{f}) & \boldsymbol{W}^\top(t) & \nabla_{\boldsymbol{f}}\boldsymbol{h}_{\mathbb{A}}(\boldsymbol{f})\\ \boldsymbol{W}(t) & \boldsymbol{0} & \boldsymbol{0}\\ \big(\operatorname{diag}(\boldsymbol{\mu}_{\mathbb{A}})\nabla_{\boldsymbol{f}}\boldsymbol{h}^\top_{\mathbb{A}}(\boldsymbol{f})\big) & \boldsymbol{0} & \boldsymbol{0}\end{bmatrix} \tag{8}$$

is continuously invertible (by strong convexity and LICQ). Moreover, the second term

$$\frac{\partial\boldsymbol{R}(\boldsymbol{z}_{\mathbb{A}},t)}{\partial t}=\begin{bmatrix}\frac{\partial}{\partial t}\big(\boldsymbol{W}(t)^\top\boldsymbol{\lambda}\big)\\ \frac{\partial}{\partial t}(\boldsymbol{W}(t)\boldsymbol{f}-\boldsymbol{w}_{\text{ref}}(t))\\ \boldsymbol{0}\end{bmatrix} \tag{9}$$

is differentiable w.r.t. t, according to A.3. Therefore, by the implicit function theorem, given a fixed strictly active set $\boldsymbol{h}_{\mathbb{A}}$, for every pair $(\bar{\boldsymbol{z}}_{\mathbb{A}},\bar{t})$ satisfying $\boldsymbol{R}_{\mathbb{A}}(\bar{\boldsymbol{z}}_{\mathbb{A}},\bar{t})=0$, there exists a unique differentiable function $\vartheta:\mathbb{R}_{\geq 0}\to\mathbb{R}^{n+m+n_a}$, in the neighbourhood of $\bar{\boldsymbol{z}}_{\mathbb{A}}$, such that $\boldsymbol{R}_{\mathbb{A}}(\vartheta(\bar{t}),\bar{t})=0$. As long as the active set remains strictly active, substituting $\boldsymbol{z}_{\mathbb{A}}(t)$ with $\vartheta(\bar{t})$ implies that $\boldsymbol{z}_{\mathbb{A}}(t)$ and thus $\boldsymbol{f}(t)$ is continuously differentiable w.r.t. t. In points where the active set changes ($\boldsymbol{h}_{\mathbb{A}}$ is not strictly active), $\frac{\partial\boldsymbol{z}_{\mathbb{A}}}{\partial t}$ is not defined. However, the left and right-handed derivatives exist. ■

Practical implications of Theorems 1 and 2:

- Since cost functions using p-norms with $1<p<\infty$ are strictly convex, they yield continuous trajectories.[2]
- Most relevant CDPR studies use p-norms in the cost functions but one may also look at other strongly convex cost functions that have favourable properties.
- Even with strongly convex cost functions, one should expect discontinuities in the derivative $\frac{d}{dt}\boldsymbol{f}^*(t)$ at instances where any individual force f_i locks onto or detaches from either $f_{i,\min}$ or $f_{i,\max}$, which explain Issue B. The method presented in the next section, will ensure that the actuator forces never reach the constraints, resulting in $\mathcal{C}^1$ continuity of actuator forces.

B. Cost functions ensuring $\mathcal{C}^1$

1) Cost function for the standard formulation: The cost function for the standard formulation is often chosen as a norm of the difference to some *preferred* force vector $\boldsymbol{f}_0$, that is $g(\boldsymbol{f})=|\boldsymbol{f}-\boldsymbol{f}_0|_p$, with the 2-norm by far the most used [5]. The use of higher p-norms is sometimes desired [4, 38], as it puts comparatively higher costs on large deviations from the preferred force vector.

Adaptable logarithmic barriers functions are typically used as numerical tools in interior-point nonlinear programming solvers to represent inequality constraints with a high degree of accuracy [39]. Inspired by this (but different in that we use them in the cost function rather than using them as a tool for numerical regularisation), we introduce fixed logarithmic barrier functions. Combining them with the traditional norm-functions, we propose the cost function:

$$g(\boldsymbol{f})=\sum_{i=1}^{n}\left(\frac{|f_i-f_{0,i}|^p}{\alpha_i^p}-c_1\log(f_i-f_{i,\min})-c_2\log(f_{i,\max}-f_i)\right). \tag{10}$$

[1]Considering the existing literature on CDPR, we believe these results to be valuable. The results are, however, less novel from a mathematical perspective. Proofs and results that with some effort can be shown to cover Theorem 1 and Theorem 2 can be found in [31–33]

[2]In the literature on CDPR, more complex and less general proofs, such as in [15], are often referred to for the same result; see [5, 9, 37].

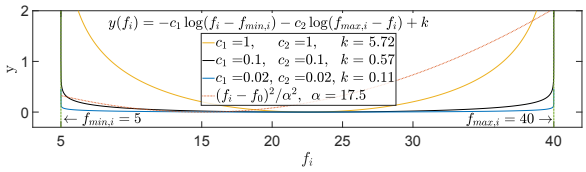

Fig. 3: Logarithmic barrier cost for a cable constrained between $f_{\min}$ and $f_{\max}$ for different values of c_1 and c_2. k is needed only for visualisation purposes (to harmonise the minima). Norm function cost term for $p{=}2$ is included for reference.

The logarithmic terms should keep the resulting actuator forces away from their respective limits. This facilitates faster solver convergence by effectively removing the hard inequality constraints from the formulation. The p-norms are minimised to the power of p, for computational efficiency, and normalised by a factor of α_i to avoid numerical accuracy issues and achieve convergence. In this paper, we use $\alpha_i{=}\frac{f_{i,\max}-f_{0,i,\min}}{2}$. The preferred force vector $\boldsymbol{f}_0$ will typically be dependent on application specific factors such as actuator technology, cable properties, safety concerns, and operating conditions. The constants c_1 and c_2 enable adjustments in how fast the cost function increases as f_i approaches the limit (see Fig. 3).

Proposition 1. *For the standard formulation* (4) *and with cost function given by* (10)*, with* $c_1{>}0$*,* $c_2{>}0$ *and* $1{<}p{<}\infty$*, and under A.1-A.5, the solution* $\boldsymbol{f}^*(t)$ *is a* $\mathcal{C}^1$ *vector-valued function.*

Proof of Proposition 1: If any of the constraints $\boldsymbol{h}(\boldsymbol{f}^*)$ were active, (10) implies that the cost $g(\boldsymbol{f}^*)$ would be infinite. This cannot be since there by A.4 and (10) must always exist a solution $\boldsymbol{f}^*$ with a finite $g(\boldsymbol{f}^*)$. Since $\boldsymbol{h}(\boldsymbol{f}^*)$ is never active, Proposition 1 follows directly from Theorem 2. Note that A.5 is automatically fulfilled under A.1, since constraints are never active and $\boldsymbol{W}$ is full row-rank. ■

2) Cost function for the slacked formulation: We propose a cost function for the slacked formulation (3) that reuses $g_f(\boldsymbol{f}){=}g(\boldsymbol{f})$ from (10), while $g_s(\boldsymbol{s})$ is set as the strongly convex function

$$g_s(\boldsymbol{s}){=}\sum_{j=1}^{q}(b_j\sqrt{\epsilon_j{+}s_j^2}{+}s_j^2), \tag{11}$$

where $\sqrt{\epsilon_j{+}s_j^2}$ approximates the absolute value norm and s_j^2 is added to ensure strong convexity also for large magnitudes of $\boldsymbol{s}$. Here $\epsilon_j{>}0$ adjusts the curvature of the cost function around $s_j{=}0$ while the parameter $b_j{>}0$ steers the gradient of the cost term. Importantly, this function allows for a high gradient even for small values of s_j, which can ensure that, when possible, it is cheaper to adjust the force tensions than to increase the slack (see Proposition 2).

In the remainder of the paper, both for simplicity and space considerations, it is assumed that

$$\{f_{i,\min}{=}f_{\min},\quad f_{i,\max}{=}f_{\max},\quad f_{0,i}{=}f_0\}\quad \forall i{\in}\{1,\cdots,n\}. \tag{12a}$$
$$\{b_j{=}b,\quad \epsilon_j{=}\epsilon\}\quad \forall j{\in}\{1,\cdots,q\}\text{ with } q{=}m \tag{12b}$$
$$c{=}\max(c_1,c_2),\quad \boldsymbol{Q}{=}\boldsymbol{I}{\in}\mathbb{R}^{m\times m} \tag{12c}$$

Moreover, let $\boldsymbol{H}$ be any generalised inverse of $\boldsymbol{W}$ such that $\boldsymbol{W}\boldsymbol{H}\boldsymbol{y}{=}\boldsymbol{y}$ for all $\boldsymbol{y}{\in}\mathbb{R}^m$. Let $\boldsymbol{H}_{\nu 1}$ be the $\boldsymbol{H}$ that has the lowest matrix one-norm [40], and define

$$\sigma{:=}||\boldsymbol{H}_{\nu 1}||_1 \tag{13a}$$
$$\lambda{:=}\max(|f_{\max}{-}f_0|,|f_0{-}f_{\min}|) \tag{13b}$$
$$\delta^*(\boldsymbol{f}_s^*){=}\min\{h_l(\boldsymbol{f}_s^*)\ :\ l{\in}1,\cdots,2n\} \tag{13c}$$
$$\gamma(\boldsymbol{f}_s^*){:=}\frac{p\lambda^{p-1}}{\alpha_p^p}+\frac{c}{\delta^*(\boldsymbol{f}_s^*)}+\frac{c}{f_{\max}{-}f_{\min}{-}\delta^*(\boldsymbol{f}_s^*)}, \tag{13d}$$

where $h_l(\boldsymbol{f}_s^*)$ is the l^{th} row of (2) with $\boldsymbol{f}{=}\boldsymbol{f}_s^*$, and $\gamma(\boldsymbol{f}_s^*)$ represents an upper bound for the cost-function gradient of any individual line (that is; $\gamma(\boldsymbol{f}_s^*){\geq}|\nabla_f g(\boldsymbol{f}_s^*)|_\infty$). We now state the following result on the bound of the slack $\boldsymbol{s}^*$:

Proposition 2. *For the slack formulation* (3)*, using* $g_f(\boldsymbol{f})$ *given by* (10)*,* $g_s(\boldsymbol{s})$ *given by* (11)*, and under* (12) *. If* $b{\geq}\sigma\gamma(\boldsymbol{f}_s^*)$*, then* $(\boldsymbol{s}^*,\boldsymbol{f}_s^*)$ *satisfies the bounding relationship*

$$|\boldsymbol{s}^*|_\infty{\leq}\sqrt{\frac{\epsilon\sigma^2\gamma^2(\boldsymbol{f}_s^*)}{b^2{-}\sigma^2\gamma^2(\boldsymbol{f}_s^*)}}. \tag{14}$$

The qualitative interpretation of Proposition 2 is that given $b{\gg}\sigma\gamma(\boldsymbol{f}_s^*)$ and a small ϵ, then $\boldsymbol{s}^*$ will be close to zero. For a fixed b, $\boldsymbol{W}$, and $\boldsymbol{w}_{\text{ref}}$, there will be a lower limit for δ^* such that $b{>}\sigma\gamma(\boldsymbol{f}_s^*)$ does not hold. This value can be considered as the limit for how close a force can be to a constraint before the slack is allowed to increase significantly.

Proof of Proposition 2: Let $V(\boldsymbol{f},\boldsymbol{s}){=}g_s(\boldsymbol{s}){+}g_f(\boldsymbol{f})$. At $(\boldsymbol{f}_s^*,\boldsymbol{s}^*)$, a directional derivative of V along any vector $\boldsymbol{u}$ satisfying (3b) is zero:

$$\nabla V(\boldsymbol{f}_s^*,\boldsymbol{s}^*)\boldsymbol{u}{=}\begin{bmatrix}\nabla_f g_f(\boldsymbol{f}_s^*)^\top & \nabla_s g_s(\boldsymbol{s}^*)^\top\end{bmatrix}\boldsymbol{u}{=}0. \tag{15}$$

With $\boldsymbol{H}$ being any generalised inverse of $\boldsymbol{W}$, $\boldsymbol{u}$ can be taken as: $\boldsymbol{u}{=}\begin{bmatrix}-\boldsymbol{H}\boldsymbol{d}\\ \boldsymbol{d}\end{bmatrix}$, for $\boldsymbol{d}{\in}\mathbb{R}^m$ $\left(\text{since }\begin{bmatrix}\boldsymbol{W} & \boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{f}_s^*{-}\boldsymbol{H}\boldsymbol{d}\\ \boldsymbol{s}^*{+}\boldsymbol{d}\end{bmatrix}{=}\boldsymbol{w}_{\text{ref}}\right)$. Let $\boldsymbol{d}{:=}\boldsymbol{d}_k$, where $\boldsymbol{d}_k{\in}\mathbb{R}^m$ is vector that has all zero-entries except the k^{th} entry p_k. Moreover, $p_k{=}\text{sign}(s_k^*)$, where s_k^* is the k^{th} element of $\boldsymbol{s}^*$, for $k{\in}\{1,\cdots,m\}$. Moving the $\boldsymbol{s}^*$ dependent part to the right side of (15) and taking the absolute value of both sides, we get

$$|\nabla_f g_f(\boldsymbol{f}_s^*)^\top\boldsymbol{H}\boldsymbol{d}_k|{=}|\nabla_s g_s(\boldsymbol{s}^*)p_k|, \tag{16}$$

where $\nabla_s g_s(\boldsymbol{s}^*)p_k{=}b\frac{|s_k^*|}{\sqrt{\epsilon{+}s_k^{*2}}}{+}2|s_k^*|{\geq}b\frac{|s_k^*|}{\sqrt{\epsilon{+}s_k^{*2}}}$. The left-hand side of (16) can be bounded by

$$\begin{aligned}|\nabla_f g_f(\boldsymbol{f}_s^*)\boldsymbol{H}\boldsymbol{d}_k|&{\leq}|\nabla_f g_f(\boldsymbol{f}_s^*)|_\infty|\boldsymbol{H}\boldsymbol{d}_k|\\ &{\leq}\underbrace{|\nabla_f g_f(\boldsymbol{f}_s^*)|_\infty}_{\leq\gamma(\boldsymbol{f}_s^*)}\ \underbrace{||\boldsymbol{H}||_1}_{\sigma}{\leq}\sigma\gamma(\boldsymbol{f}_s^*)\end{aligned} \tag{17}$$

Letting[3] $||\boldsymbol{H}||_1{=}||\boldsymbol{H}_{\nu 1}||_1{=}\sigma$, we get $b\frac{|s_k^*|}{\sqrt{\epsilon{+}s_k^{*2}}}{\leq}\sigma\gamma(\boldsymbol{f}_s^*)$, which for $b{>}\sigma\gamma(\boldsymbol{f}_s^*)$ can be rearranged to

$$|s_k^*|{\leq}\sqrt{\frac{\epsilon\sigma^2\gamma^2(\boldsymbol{f}_s^*)}{b^2{-}\sigma^2\gamma^2(\boldsymbol{f}_s^*)}}. \tag{18}$$

Since this must hold for each k, we get (14). ■

This result can be used in the design and tuning phase to provide insight for choosing adequate values for the cost function. It might also be computed during runtime if computational resources allow it.

C. Properties of resulting trajectories

We will now use the motivating example of Section II-B to demonstrate the properties of the presented cost functions. Hereafter, whenever the motivating example is considered, only the first half of the trajectory is used, ensuring that the trajectory is feasible also with the standard formulation.

1) Standard formulation: Fig. 4 shows the resulting trajectory for three sets of c_i values. In accordance with Theorem 2, with $c{>}0$, the force rates become $\mathcal{C}^1$ continuous, as expected.

When deciding c_1 and c_2, we recommend considering the relative magnitude of each term of the gradient of (10) as f_i ranges in the interval $\{f_{i,\min}\ ,f_{i,\max}\}$. We use $c_1{=}c_2{=}0.1$ in later examples.

[3]An iterative method for finding $\boldsymbol{H}_{\nu 1}$ is described in [40]. Alternatively one may use the simpler pseudoinverse (that is, $\boldsymbol{H}{=}\boldsymbol{W}^\dagger{\Longrightarrow}\sigma{=}||\boldsymbol{W}^\dagger||_1$), which will yield a higher bound.

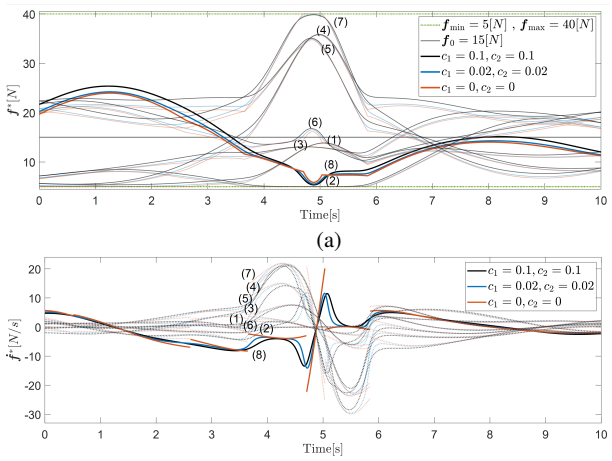

Fig. 4: Force solutions to the standard formulation for the motivating example with varying values of c_1,c_2. One cable (no. 8) is highlighted with solid lines. (a) Actuator forces. (b) Force rates.

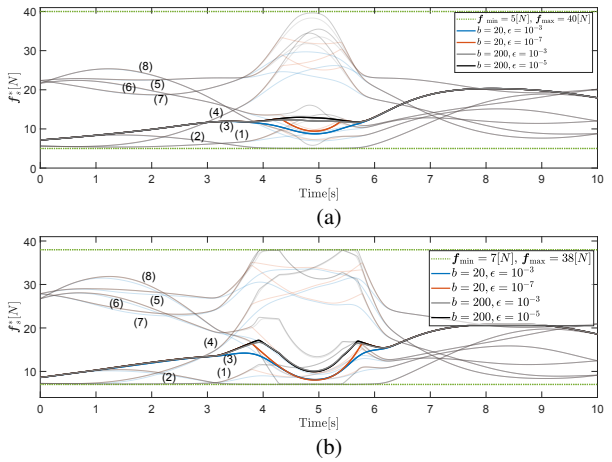

Fig. 5: Force solutions to the slacked formulation for the motivating example with varying values of b and ϵ and two different sets of constraints. One cable (no. 3) is shown with solid lines.

2) Slacked formulation: Fig. 5 shows the resulting trajectory for varying values of b and ϵ, both for the default case and a case where both upper and lower constraint limits have been contracted, making the trajectory infeasible. Fig. 6 shows the resulting absolute norm-error of the slack. The dashed lines in the figure indicate sections where $b \geq \sigma\gamma(\boldsymbol{f}_s^*)$ does not hold.

To emphasise their relative effect, (ϵ,b) is intentionally chosen such the relative difference of $\log_{10}(\frac{\sqrt{\epsilon}}{b})$ remains an integer. The effect is evident in the non-dashed line-sections of Fig. 6, where the logarithmic difference in the slack magnitude remains integers. In general, the error is small whenever no forces are close to their respective constraints. In sections that otherwise would be infeasible, the method robustly produces solutions by increasing the slack.

As observed in Fig. 5, a lower ϵ and a higher b cause sharper shifts in actuator force rates in the transition between the feasible and non-feasible configurations. The result can be that the discretised forces in practice are nonsmooth, even if $t \mapsto \boldsymbol{f}(t)$ remains differentiable. In the following, we use $\epsilon=10^{-3}$ and $b=200$.

IV. Newton's method on the KKT conditions and real-time feasibility

In general, iterative solvers are needed to solve (3) or (4) with (11) and (10). We present a solution to the problem by iteratively

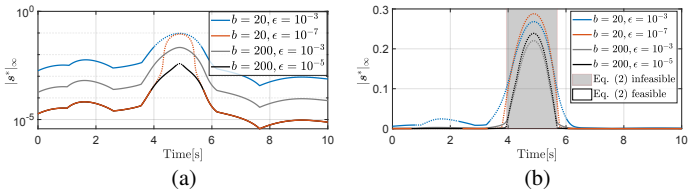

Fig. 6: Slack corresponding to Fig. 5 for the two cases of constraints. (Note that $\boldsymbol{Q}=diag(\boldsymbol{1})$ implies $|\boldsymbol{s}^*|_\infty=|\boldsymbol{w}_{\text{err}}|_\infty$). (a) $f_{i,\min}=5[N], f_{i,\max}=40[N]$. (b) $f_{i,\min}=7[N], f_{i,\max}=38[N]$.

performing Newton steps on the KKT conditions (6), until convergence. Variations of this method are widely used within the numerical optimisation literature [41, p.175]. We are not aware of other papers that go in details on the use of this algorithm for use in CDPR applications (note that [42] considers a different analytical-iterative scheme based on the KKT conditions). Using the notation defined in (5) (with $\boldsymbol{x}=\boldsymbol{f}$, and $\boldsymbol{A}=\boldsymbol{W}$ for the standard formulation) we next briefly describe the key parts of the implemented algorithm;

1) At iteration k, a Newton step is performed on the KKT conditions:

$$\begin{bmatrix} \nabla^2_{\boldsymbol{x}} g(\boldsymbol{x}_k) & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{x}_k \\ \boldsymbol{\lambda}_k+\Delta\boldsymbol{\lambda}_k \end{bmatrix} = - \begin{bmatrix} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k) \\ \boldsymbol{A}\boldsymbol{x}_k-\boldsymbol{w}_{\text{ref}} \end{bmatrix} \tag{19}$$

We denote the Newton direction $\boldsymbol{d}_k=\begin{bmatrix}\Delta\boldsymbol{x}_k & \Delta\boldsymbol{\lambda}_k\end{bmatrix}^\top$.

2) In each iteration, we perform a line-search (backtracing) to determine how large the step in this direction should be:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_k \end{bmatrix} + \kappa \boldsymbol{d}_k \tag{20}$$

3) κ is determined using a line-search strategy where we ensure that solutions remain feasible and that the following merit function is decreasing from step k to $k+1$:

$$\phi(\boldsymbol{x},\boldsymbol{\lambda}) = \left| \begin{matrix} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{\lambda} \\ \boldsymbol{A}\boldsymbol{x}-\boldsymbol{w}_{\text{ref}} \end{matrix} \right|_\infty \tag{21}$$

4) When the merit function is below the predetermined *tolerance* threshold, the solutions are returned, with the number of iterations and line-steps recorded. The use of the algorithm to solve (4) or (3) until convergence at a time instance t_k is hereafter referred to as one evaluation. Further implementation details are provided in the electronic appendix.

When discussing *evaluation time*, we generally refer to the computation time used to perform an evaluation at a specific time instant t_k in the trajectory, while the *trajectory evaluation time* refers to the computation time for evaluating all data-points in the trajectory. We only consider computation times using *warm start* initialisation, where (19) is initialised at the solution of the preceding evaluation (good warm start initialisation points should always be available, due to continuity and smoothness of solutions). The internal iterative procedure (20) is not found to constitute a separate computational issue and is only briefly discussed in the electronic appendix. Corresponding results obtained from data with cold start initialisation are also discussed there.

The following examination of computation times uses a standard, relatively powerful workstation computer[4] with *Windows 10*, and with the solver scripts being compiled to *C*-code. Time is measured using MATLAB's *tic* and *toc* functions.[5] As computational resources differ depending on computer configuration, the reader is encouraged to test computation times on their own configurations by use of the

[4]Intel Core i7-7700 (QC/8 MB/8 T/3,6 GHz/65 W)

[5]To overcome issues in estimating very short time-intervals, each function-evaluation is repeated until at-least 0.1 second had elapsed with subsequent averaging. Also, process priority is set to *real-time* in *Windows 10*.

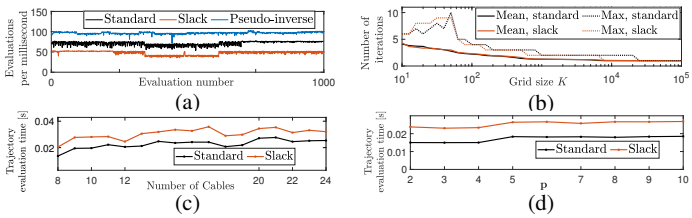

Fig. 7: Solver characteristics and computation time on the motivating example. (a) Computational speed over the trajectory. (b) Number of iterations as a function of changing grid size (K) in the trajectory (c) Trajectory evaluation time as a function of changing the number of cables. (d) Trajectory evaluation time as a function of changing p in the cost function.

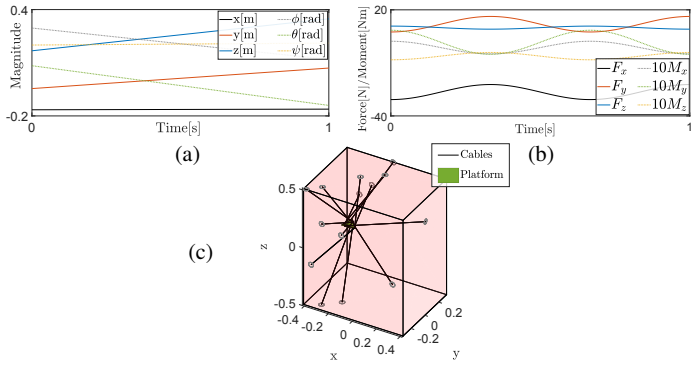

Fig. 8: Sample of randomly generated trajectory and configuration. (a) $\boldsymbol{\eta}(t_k)$. (b) $\boldsymbol{w}_{\text{ref}}(t_k)$. (c) Randomised configuration mode.

electronic appendix.

A. Performance on the motivating example trajectory

We consider computation times for the motivating example with default solver parameters given in Table I(b)-(c). Fig. 7(a) presents the evaluation time along the trajectory, demonstrating how real-time feasibility is achieved well within limits[6]. Fig. 7(b) shows the number of iterations when changing the grid size K, demonstrating how shorter step-sizes, generally results in better initialization and quicker convergence (short cycle times and frequent resampling generally appears to be preferred within relevant literature [43]). Fig. 7(c) demonstrates how the algorithm handles an increased number of cables[7] (some common force allocation methods are not well adapted to handling a large number of cables [13]). Finally, Fig. 7(d) shows how the trajectory evaluation time is nearly independent on the choice of the norm[8] p.

B. Assessing method robustness through randomised cases

A concern in using Newton's method for optimisation in real-time applications is that the solver, for some configurations, could undergo a sequence of iterations with particularly slow linear convergence, requiring numerous iterations to reach the stopping criterion. According to [44] (see also [41, Chapter 12]), it is difficult to assess with generality where relevant local convergence results hold (that is, where quadratic and superlinear convergence can be shown).

[6]For reference Fig. 7(a) also includes computation time using a C-compiled pseudoinverse without equality constraints

[7]In Fig. 7(c), the number of cables were increased until each actuator-base was connected via a cable to each actuator-endpoint). To ensure that the trajectories were feasible for all cable combinations, $\boldsymbol{f}_{\min}$ was reduced from $5[N]$ to $1[N]$.

[8]For high values of p, numerical issues and nonlinearity may degrade the performance of the solver. In that case, one may consider increasing α to avoid large nonlinearities.

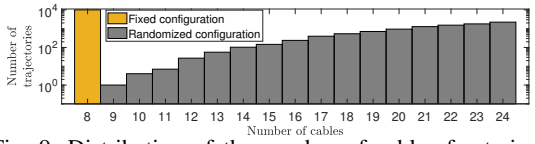

Fig. 9: Distribution of the number of cables for trajectories in Case 1

Therefore, accurate performance must, in practice, be deducted from numerical experiments.

We examine the robustness of real-time feasibility of our method by applying it to a large number of randomised configurations and trajectories which we believe challenge the algorithm for most relevant applications, sampled as follows (see also Table I(d)):
1) In each trajectory, $\boldsymbol{\eta}(t)$ and $\boldsymbol{w}_{\text{ref}}(t)$ are set to vary between randomised lower and upper limits according to Table I(d). See sample in Fig. 8(a,b).
2) Two configurations are used: In the *fixed configuration* mode, the actuators are realistically configured as before (e.g., Table I(a)). In the *randomised configuration* mode, between 8 and 24 actuators are placed randomly around the platform according to Table I(d). See sample in Fig. 8(c).
3) The tests are run using the solver parameters of Table I(c), except for the lower constraint $\boldsymbol{f}_{\min}$, which is reduced from $5[N]$ to $1[N]$.
4) We separate between two cases. In *Case 1* we only consider trajectories that are not feasible. In *Case 2*, we also consider infeasible trajectories.

1) Case 1: Feasible trajectories only: For this case, we sampled trajectories until 10^4 feasible trajectories were sampled for both the randomised and for the fixed configuration (corresponding to $2{\cdot}10^7$ evaluations for each solver-setup). The likelihood that actuator forces are feasible increases with the number of randomly placed actuators. Hence, the resulting trajectories for the randomised configuration tend to feature a higher number of cables as illustrated by Fig. 9. The bar at 8 cables, corresponds to the half of the tests that were performed for the fixed configuration.

The bar charts of Fig. 10(a-c) illustrate the number of evaluation-instances as a function of the number of iterations for different solver setups, showing that no more than twelve iterations were used.

The cutoff criteria $\boldsymbol{h}(\boldsymbol{f}(t)) \succcurlyeq \delta$ for classifying a trajectory as infeasible (relates to A.4) is indicated in Fig. 10(a-b). Increasing the cutoff criteria results in fewer evaluations with many iterations. This illustrates how, for the standard formulation, nonlinear sections near the edges of the feasible workspace reduce the performance of the warm start initialisation. As shown in Fig.10(c), this is not a problem with the slacked formulation, where the highest number of iterations is five.

Fig. 10(d) shows the resulting error ($|\boldsymbol{w}_{err}|_\infty$) of the slacked formulation. It never exceeds 0.026, showing how the slacked version can also be used on feasible trajectories with small to negligible effect on the resulting wrench. Therefore, if one expects trajectories to be close to the boundaries of the feasible workspace, the slack version might be a good compromise between accuracy and robustness.

2) Case 2: Including infeasible trajectories: For this case, which aims to investigate the slacked method, 10^4 trajectories were sampled for the fixed configuration, without rejecting infeasible trajectories. The distribution of iterations is shown in Fig. 11(a), while the corresponding distribution of the slack $\boldsymbol{s}^*$ is given by Fig. 11(b). The number of iterations remains limited and never exceeds 17. Comparing Fig. 10 with Fig. 11, it is evident that introducing infeasible trajectories result in more iterations, which is not so surprising since more nonlinear sections of the cost functions typically lead to a higher number of iterations.

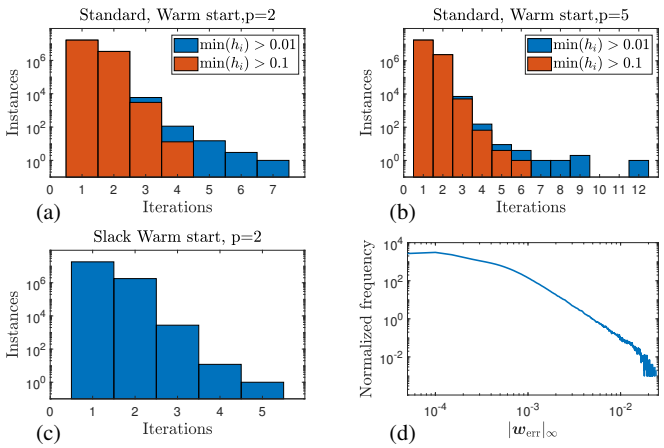

Fig. 10: Case 1. (a-c) Number of iterations per evaluation for approximately $2{\cdot}10^7$ different evaluations (in each subfigure) (d) Distribution of $|\boldsymbol{w}_{\text{err}}|_\infty$ using the slacked solver setup. (Note that $\boldsymbol{Q}{=}\boldsymbol{I}$ implies $|\boldsymbol{s}^*|_\infty{=}|\boldsymbol{w}_{\text{err}}|_\infty$)

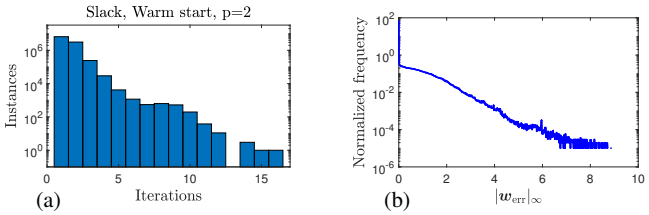

Fig. 11: Case 2: Number of iterations per evaluation ($2{\times}10^7$ different evaluations) (b) Distribution of $|\boldsymbol{w}_{\text{err}}|_\infty$. ($|\boldsymbol{s}^*|_\infty{=}|\boldsymbol{w}_{\text{err}}|_\infty$)

3) Relating the number of iterations to the evaluation times: To assess how the number of iterations and evaluation times relates to each other, data from all separate evaluation instances is plotted in Fig. 12. We see that when initialised in warm start, the evaluation times are always shorter than $0.25ms$ It is believed that in practice, the upper outliers that are far away from the mean evaluation times are likely to be due to noise (jitter, task-scheduling, inaccurate timing, etc.) associated with estimating short intervals, which a real-time dedicated computer should overcome (see next section).

Due to randomisations, the upper number of iterations experienced should be seen as strong indicators of what to expect in worst-cases rather than as theoretical maximums. Therefore, some safety margin, accounting for a higher number of iterations, is advised. In the electronic appendix, we show corresponding data when cold start initialisation is used. These provide more samples for higher number of iterations, and show that the trends seen in Fig. 12 continues, also for a higher number of iterations, indicating that even if accounting for a substantially higher number of iterations, real-time feasibility is achieved.

C. Test on a practical application

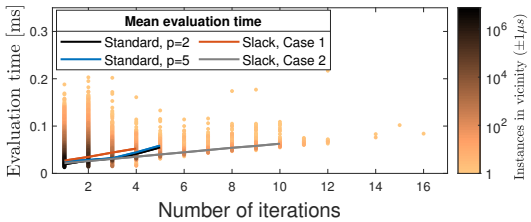

Fig. 12: Scatterplots of evaluation-time as a function of iterations. Time data calculated over by averaging computation times over ten repeated evaluations.

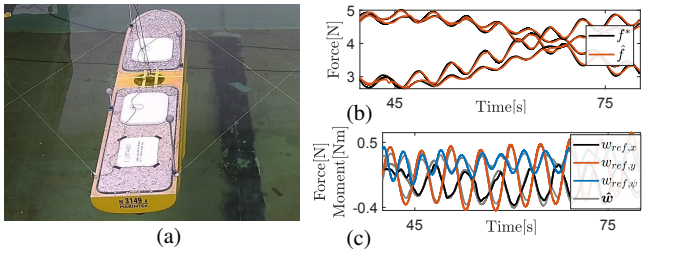

Fig. 13: Experimental data from one of the setups that have been used by the authors and collaborators [8]. Measured forces has been lowpass filtered at 1 Hz. (a) The planar, 4-cabled marine CDPR. (b) Cable force tracking. (c) Target wrench tracking

The presented methods have been applied successfully in experiments at the Marine Technology Center at NTNU; see Fig. 13. The code was executed on real-time hardware (NI cRIO-9034) and always provided solutions within each cycle time of 5 ms (using 1-4 Newton iterations). Fig. 13(b-c) illustrates the force tracking performance for a sample interval from one of the experimental tests, where the algorithm was used to compute target cable forces $\boldsymbol{f}^*{\in}\mathbb{R}^4$ from target wrench $\boldsymbol{w}_{\text{ref}}{=}\{w_{\text{ref},x}, w_{\text{ref},y}, w_{\text{ref},\psi}\}{\in}\mathbb{R}^3$. The resulting measured forces $\hat{\boldsymbol{f}}{\in}\mathbb{R}^4$, is mapped back to the corresponding load vector $\hat{\boldsymbol{w}}{\in}\mathbb{R}^3$ to assess the wrench tracking performance. Force tracking represents a separate control problem, which depends on actuators technology, actuator control methods, force sensors, communication and more. This is pursued in other studies and not considered here.

The conducted experiments have demonstrated the applicability of the methods presented in this paper. However, the rather few actuators and relative slow motions of the marine platform do not sufficiently challenge the algorithm to verify the limiting performance. For this purpose, the simulation studies were performed to more significantly challenge the performance of the presented methods.

V. Multimedia Attachment

To promote further developments, adaptation and replication, an electronic appendix containing additional material, implementation details, animations and code for producing all figures in the paper is attached and available at http://ieeexplore.ieee.org. A more extensive version can also be found at *Github* [45]-*the CDPR force allocation toolbox*, as a tool for engineers who wish to explore the force allocation algorithms further.

VI. Conclusions

By using a new cost function for the optimal force allocation problem, continuous differentiability of the actuator forces in a CDPR are guaranteed, which avoids undesired discontinuous accelerations of actuators. We further show that a Newton's method implementation for solving the KKT conditions, specialised to the problem at hand, can be used in practical real-time applications. The presented methods are flexible in handling different problem configurations (varying number of cables, p- norms and actuator configurations) and allow for an intuitive tuning of the cost-function, thus overcoming some of the challenges of existing methods. For use in cases where it is suitable, we have also introduced and analysed a slacked version of the optimal force allocation problem, and provided an upper bound of the wrench-error.

A delimiting note: Although we argue that the majority of the practical cases is covered by the presented study on real-time feasibility, it is not difficult to design setups where the number of iterations exceeds the numbers presented in this paper (for example with a near singular $\boldsymbol{W}$, or with abrupt changes in reference wrench

TABLE I: Configurations, parameters and trajectories

(a) Actuator placements and cable connections

	Cable exit points ($\boldsymbol{p}_a^a$)								Cable attachment anchors ($\boldsymbol{r}_i^b$)		
	1	2	3	4	5	6	7	8	1	2	3
x	-0.415	-0.415	0.415	0.415	0.415	-0.415	-0.415	0.415	-0.0525	0.0525	0
y	-0.315	-0.315	-0.315	-0.315	0.315	0.315	0.315	0.315	-0.076	-0.076	0.124
z	-0.500	0.500	0.500	-0.500	-0.500	0.500	-0.500	0	0	0	0

Cable:	1	2	3	4	5	6	7	8
$\boldsymbol{r}_i^b$:	1	1	2	2	3	3	3	3
$\boldsymbol{p}_a^a$:	1	2	3	4	5	6	7	8

(b) Motivating example; parameters and trajectories

Parameter	Value
$\boldsymbol{f}_0$	$\begin{bmatrix}15 & 15 & 15 & 15 & 15 & 15 & 15 & 15\end{bmatrix}^\top$
$\boldsymbol{f}_{\max}$	$\begin{bmatrix}40 & 40 & 40 & 40 & 40 & 40 & 40 & 40\end{bmatrix}^\top$
$\boldsymbol{f}_{\min}$	$\begin{bmatrix}5 & 5 & 5 & 5 & 5 & 5 & 5 & 5\end{bmatrix}^\top$
Evaluations K^*	2000 before Section III. 1000 after Section III.
t_K^{**}	20 [s] before Section III. 10 [s] after Section III.
Wrench $\boldsymbol{w}_{\text{ref}}(t_k)$:	$(F_x,F_y,F_z,M_x,M_y,M_z)$ $(0,0,5,0,0,0)$
Tractory:$\boldsymbol{\eta}(t_k)$ $\begin{pmatrix}t_k=\Delta_t k\\ \Delta_t=\frac{t_K}{K}\\ \omega=0.1\pi\end{pmatrix}$	(x,y,z,ϕ,θ,ψ) $\begin{pmatrix}0.1\sin(\omega t_k),0.1\cos(\omega t_k),0.34sin(\omega t_k)cos(2\omega t_k)-0.052,\\ -0.02t_k cos(\omega t_k),0,-0.17sin(\omega t_k)cos(2\omega t_k)\end{pmatrix}$

$^*K$ *is the number of evenly spaced evaluations points* t_k *along the trajectory.*
$^{**}t_K$*is the time at the final evaluation point (that is, the total trajectory time).*

(c) Cost function and solver parameters*

Parameter	Value	Parameter	Value
c_1,c_2	(0.1,0.1)	b_j	200, for j=1,2,$\cdots m$
p (norm):	2	α_i	$(f_{\max}-f_{\min})/2$, for i=1,2,$\cdots n$
ϵ_j	10^{-3}, for j=1,2,$\cdots m$	Tolerance	$5e{-}5$

*With different force-levels one may consider scaling some solver parameters.

(d) Data for drawing of trajectories and configurations

Case	Value
Sampling, random config.	p_{ai}^a: Drawn from surface area of cuboid with center at (0,0,0) and side lengths (0.83,0.63,1) p_{ei}^a Drawn from the line segments constituting the hull from the motivating example (i.e., Triangle with vertices at a_{e1},a_{e2} and ,a_{e3})
Fixed config.	As in Table I(a)
Sampling of trajectories, Case 1	$\boldsymbol{\eta}_0$=rnd$(m,1)\boldsymbol{\eta}_{lim}$, $\boldsymbol{\eta}_K$=rnd$(m,1)\boldsymbol{\eta}_{lim}$, $\boldsymbol{\eta}(t_k)=\boldsymbol{\eta}_0+\frac{k}{K}(\boldsymbol{\eta}_K-\boldsymbol{\eta}_0)$ $\boldsymbol{w}_0$=rnd$(m,1)\boldsymbol{w}_{lim}$, $\boldsymbol{w}_K$=rnd$(m,1)\boldsymbol{w}_{lim}$, $\boldsymbol{w}_{\text{ref}}(t_k)=\boldsymbol{w}_0+0.5(\boldsymbol{w}_K-\boldsymbol{w}_0)(1-cos(3\pi\frac{k}{K}))$ $\boldsymbol{w}_{\text{ref},lim}$ = $\begin{bmatrix}50 & 50 & 50 & 1 & 1 & 1\end{bmatrix}^\top$, $\boldsymbol{\eta}_{lim}$ = $\begin{bmatrix}0.3 & 0.15 & 0.4 & 0.3 & 0.3 & 0.3\end{bmatrix}^\top$
Sampling of trajectories, Case 2	$\boldsymbol{\eta}_0$=rnd$(m,1)\boldsymbol{\eta}_{lim}$, $\boldsymbol{\eta}_K$=rnd$(m,1)\boldsymbol{\eta}_{lim}$, $\boldsymbol{\eta}(t_k)=\boldsymbol{\eta}_0+\frac{k}{K}(\boldsymbol{\eta}_K-\boldsymbol{\eta}_0)$ $\boldsymbol{w}_0$=rnd$(m,1)\boldsymbol{w}_{lim}$, $\boldsymbol{w}_K$=rnd$(m,1)\boldsymbol{w}_{lim}$, $\boldsymbol{w}_{\text{ref}}(t_k)=\boldsymbol{w}_0+0.5(\boldsymbol{w}_K-\boldsymbol{w}_0)(1-cos(1\pi\frac{k}{K}))$ $\boldsymbol{w}_{\text{ref},lim}$ = $\begin{bmatrix}20 & 20 & 20 & 2 & 2 & 2\end{bmatrix}^\top$, $\boldsymbol{\eta}_{lim}$ = $\begin{bmatrix}0.24 & 0.12 & 0.24 & 0.3 & 0.3 & 0.3\end{bmatrix}^\top$

*rnd(m,1)$\in\mathbb{R}^m$ is a vector with m elements, uniformly sampled from the interval between -1 and 1.

between two subsequent iterations). When testing a new setup, we therefore recommend to first use the provided methods to verify that the number of iterations remains acceptable using suitable offline scenarios.

ACKNOWLEDGMENT

This work was supported by grant No. 254845 *Real-Time Hybrid Model Testing for Extreme Marine Environments* and the Research Council of Norway through the Centre of Excellence NTNU AMOS, project no. 223254.

REFERENCES

[1] W. Kraus, V. Schmidt, P. Rajendra, and A. Pott, "System identification and cable force control for a cable-driven parallel robot with industrial servo drives," in *ICRA*, pp. 5921–5926, IEEE, 2014.

[2] S.-R. Oh and S. K. Agrawal, "Cable suspended planar robots with redundant cables: Controllers with positive tensions," *IEEE T-RO*, 2005.

[3] J. Lamaury and M. Gouttefarde, "A tension distribution method with improved computational efficiency," in *CDPRs*, Springer, 2013.

[4] C. Gosselin and M. Grenier, "On the determination of the force distribution in overconstrained cable-driven parallel mechanisms," *M*, 2011.

[5] A. Pott, *Cable-Driven Parallel Robots: Theory and Application*, vol. 120. Springer, 2018.

[6] J. Rodnunsky and T. Bayliss, "Aerial cableway and method for filming subjects in motion," July 6 1993. US Patent 5,224,426.

[7] A. Pott, H. Mütherich, W. Kraus, V. Schmidt, P. Miermeister, and A. Verl, "Ipanema: a family of cable-driven parallel robots for industrial applications," in *CDPRs*, pp. 119–134, Springer, 2013.

[8] E. S. Ueland, R. Skjetne, and S. A. Vilsen, "Force actuated real-time hybrid model testing of a moored vessel: A case study investigating force errors," *IFAC-CAMS*, 2018.

[9] 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 T-RO*, 2015.

[10] R. L. Williams, P. Gallina, and J. Vadia, "Planar translational cable-direct-driven robots," *J Robotic Syst*, 2003.

[11] M. A. Khosravi and H. D. Taghirad, "Robust PID control of fully-constrained cable driven parallel robots," *IFAC-Mech*, 2014.

[12] A. Pott, T. Bruckmann, and L. Mikelsons, "Closed-form force distribution for parallel wire robots," in *CK*, Springer, 2009.

[13] A. Pott, "An improved force distribution algorithm for over-constrained cable-driven parallel robots," in *CK*, Springer, 2014.

[14] T. A. Johansen and T. I. Fossen, "Control allocation—a survey," *Auto*, 2013.

[15] R. Verhoeven, *Analysis of the workspace of tendon-based Stewart platforms*. PhD thesis, UDE, Fiw, 2004.

[16] K. Müller, C. Reichert, and T. Bruckmann, "Analysis of a real-time capable cable force computation method," in *CDPRs*, Springer, 2015.

[17] L. Mikelsons, T. Bruckmann, M. Hiller, and D. Schramm, "A real-time capable force calculation algorithm for redundant tendon-based parallel manipulators," in *ICRA*, IEEE, 2008.

[18] W. C. Durham, "Constrained control allocation," *JGCD*, 1993.

[19] M. W. Oppenheimer, D. B. Doman, and M. A. Bolender, "Control allocation for over-actuated systems," in *MED*, IEEE, 2006.

[20] T. I. Fossen, T. A. Johansen, and T. Perez, "A survey of control allocation methods for underwater vehicles," in *UV*, IO, 2009.

[21] T. A. Johansen, T. I. Fossen, and S. P. Berge, "Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming," *IEEE T-CST*, 2004.

[22] R. Skjetne and Ø. K. Kjerstad, "Recursive nullspace-based control allocation with strict prioritization for marine craft," *IFAC P-V*, 2013.

[23] K. A. Bordignon, *Constrained control allocation for systems with redundant control effectors*. PhD thesis, Virginia Tech, 1996.

[24] J. Virnig and D. Bodden, "Multivariable control allocation and control law conditioning when control effectors limit," in *GNCC*, 1994.

[25] L. Borrelli, T. Baotic, A. Bemporad, and T. Morari, "Efficient on-line computation of constrained optimal control," in *C-DaC*, IEEE, 2001.

[26] C. Klein and T.-S. Chung, "Force interaction and allocation for the legs of a walking vehicle," *J-RA*, 1987.

[27] A. F. Côté, P. Cardou, and C. Gosselin, "A tension distribution algorithm for cable-driven parallel robots operating beyond their wrench-feasible workspace," in *ICCAS*, IEEE, 2016.

[28] T. I. Fossen, *Handbook of marine craft hydrodynamics and motion control*. jws, 2011.

[29] P. H. Borgstrom, B. L. Jordan, G. S. Sukhatme, M. A. Batalin, and W. J. Kaiser, "Rapid computation of optimally safe tension distributions for parallel cable-driven robots," *IEEE T-RO*, 2009.

[30] T. Sauder, S. Marelli, and A. J. Sørensen, "Probabilistic robust design of control systems for high-fidelity cyber–physical testing," *Auto*, 2019.

[31] K. Jittorntrum, "Solution point differentiability without strict complementarity in nonlinear programming," in *SSPA*, Springer, 1984.

[32] Y. Terazono and A. Matani, "Continuity of optimal solution functions and their conditions on objective functions," *SIAM J-O*, 2015.

[33] C. Berge, *Topological Spaces: including a treatment of multi-valued functions, vector spaces, and convexity*. CC, 1997.

[34] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer, 2013.

[35] G. Still, "Lectures on parametric optimization: An introduction," *UT*, 2006.

[36] J. Nocedal and S. J. Wright, *Numerical optimization 2nd*. Springer, 2006.

[37] T. Bruckmann, A. Pott, M. Hiller, and D. Franitza, "A modular controller for redundantly actuated tendon-based stewart platforms," *ECMS*, 2006.

[38] V. Chabaud, L. Eliassen, M. Thys, and T. Sauder, "Multiple-degree-of-freedom actuation of rotor loads in model testing of floating wind turbines using cable-driven parallel robots," in *JoF:C*, IOP P, 2018.

[39] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *MP*, 2000.

[40] I. Dokmanić and R. Gribonval, "Beyond moore-penrose part i: Generalized inverses that minimize matrix norms," *arXiv*, 2017.

[41] J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, *Numerical optimization: theoretical and practical aspects*. Spring, 2006.

[42] Y. B. Bedoustani and H. D. Taghirad, "Iterative-analytic redundancy resolution scheme for a cable-driven redundant parallel manipulator," in *AIM*, IEEE.

[43] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear mpc: bridging the gap via the real-time iteration," *IJC*, 2016.

[44] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[45] U. Einar, "CDPR force allocation toolbox." https://github.com/EinarUeland/CDPR-force-allocation-toolbox, 2020.