TRAJECTORY GENERATION AND CONTROL OF A CABLE-DRIVEN PARALLEL ROBOT FOR AERIAL VIDEOGRAPHY AND DYNAMIC WORKSPACE OPTIMIZATION

Nathan Raab [1], Andrew Lamperski [2] Ryan J. Caverly [3]
[1]*University of Minnesota Robotics Institute, University of Minnesota, Minneapolis, MN, U.S.A.*
[2]*Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, U.S.A.*
[3]*Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, MN, U.S.A.*
Email: raabx028@umn.edu; alampers@umn.edu; rcaverly@umn.edu

ABSTRACT

This paper investigates nonlinear model-predictive control (MPC) of a suspended cable-driven parallel robot (CDPR) focused on the applications of aerial videography and dynamic trajectory optimization. An MPC approach is used to perform optimal trajectory generation with explicit consideration of trajectory smoothness and position error for aerial videography. A similar MPC algorithm is used to generate payload trajectories outside of the CDPR's static workspace, which can be useful in applications, such as motion simulators, to expand the available workspace. It is shown through numerical simulations that the proposed MPC formulation is flexible in its ability to account for different constraints and cost functions.

Keywords: Cable-Driven Parallel Robots; Model-Predictive Control; Optimal Trajectory Generation.

GÉNÉRATION DE TRAJECTOIRE ET CONTRÔLE D'UN ROBOT PARALÈLLE À CÂBLES POUR LA VIDÉOGRAPHIE AÉRIENNE ET L'OPTIMISATION DES TRAJECTOIRES DYNAMIQUES

RÉSUMÉ

Cet article étudie la commande prédictive nonlinéaire d'un robot parallèle à câbles suspendu concentré sur les applications de la vidéographie aérienne et de l'optimisation de trajectoires dynamiques. Une approche de commande prédictive est utilisée pour la génération d'une trajectoire optimale en tenant compte explicitement de la régularité de la trajectoire et de l'erreur de position pour la vidéographie aérienne. Un algorithme de commande similaire est utilisé pour générer des trajectoires en dehors de l'espace de travail statique du robot. Il est montré par des simulations numériques que la formulation commande prédictive proposée est flexible dans sa capacité à prendre en compte différentes contraintes et fonctions de coût.

Mots-clés : Robot Parallèle à Câbles ; Commande Prédictive ; Génération de Trajectoire Optimale.

1. INTRODUCTION

Cable-driven parallel robots (CDPRs) are a type of robot that uses cables to provide actuation to a central payload [1]. This design yields several advantages over traditional robots composed of rigid body elements. First, the CDPR typically has a larger static workspace than traditional robots, which is limited primarily by the length and strength of its cables [2, 3]. It is also possible for CDPRs to extend this workspace even further with dynamic motion outside of its static workspace [4]. Second, by the use of lightweight cables and a fixed motor platform, the effective payload to robot mass ratio is significantly increased [2, 3]. As a consequence of this, CDPRs are able to generate very high velocities and accelerations due to their relatively small inertia [5]. However, CDPRs also feature practical drawbacks. First and foremost, the cables must be kept under tension to retain control authority [2]. As a consequence, when torques are considered as the inputs to the CDPR system, only torques that yield positive cable tensions are feasible. Furthermore, many CDPRs are over constrained. In the problem of optimal trajectory generation, an over constrained system results in an infinite number of feasible trajectories to reach a desired goal. Finding the trajectory that best optimizes a desired task can therefore be computationally difficult. A notable example of a CDPR is the SkyCam, which is a cable-controlled camera used for aerial videography, typically during sports games [1]. Significant research has also been done on CDPRs for use in motion simulators [4, 6–8] and other applications, such as rehabilitation or large-scale building assembly [1].

The task of designing a control algorithm for a CDPR has traditionally been addressed through kinematics-based (or joint-space) proportional-integral-derivative (PID) controllers [9], where inverse kinematics is used to determine cable lengths that correspond to a desired payload pose. PID controllers are then used on each motor to control the torques and track the cable lengths with motor encoder feedback. However, because feedback control is applied at the motors and not directly on the payload motion, they have no direct feedback on the payload pose. This means any errors in the inverse kinematics will remain as noise in the system. These challenges lend themselves favorably to a nonlinear model-predictive control (MPC) technique that operates directly in task-space and computes either cable tensions or motor torques as control inputs. Nonlinear-MPC can handle constraints on cable tension to avoid cable slack, as well as model the nonlinear dynamics of the CDPR without any linearization of the model [10]. Since MPC is dependent on an accurate model, avoiding the need for linearization is significant in order to prevent modeling errors. This is particularly important when operating far from the linearization point, such as on the boundary of the workspace. Compared to these other techniques, nonlinear-MPC is significantly more computationally expensive, but its versatility in application, hardware design, and implementation is significant, as demonstrated in this paper through the applications of aerial videography and dynamic workspace optimization.

In the literature, CDPRs that utilize MPC have no metric for the smoothness of their trajectory, which is crucial for a camera application [4, 11]. In this paper, the jerk (i.e., the time derivative of acceleration) of the payload is modeled according to first principles and included in the cost function of the MPC algorithm. As such, the camera smoothness can be optimized since smoothness can be interpreted as the inverse of jerk. However, the analytical calculation of jerk is a computationally expensive process. This paper also considers numerical approximations of jerk and simpler cost functions that do not explicitly include jerk, which attempt to speed up the computation time. Results with these approximations are compared to the analytical jerk calculation. To the best of the authors' knowledge, this is the first MPC algorithm applied to CDPRs that models the derivative of acceleration in order to optimize with respect to jerk.

CDPRs can also be applied to motion simulation outside of the static workspace by generating trajectories that keep the payload in constant motion. Some existing literature has investigated the application of CDPRs for trajectory generation in a dynamic workspace for periodic trajectories [4, 6–8]. In this paper, dynamic workspace is defined as the set of all possible poses and accelerations for which all cables have positive tension, as described in [12]. In the literature, one approach is to analytically derive classes of trajectories that

operate outside the static workspace [6, 7]. However, each of these techniques must be solved analytically for the specific class of trajectory and does not generalize to other trajectories. A similar technique in [8] shows that under certain conditions, the CDPR can behave equivalently to a passive linear mass-spring system. However, this method is limited to trajectories for which these linear conditions are met. Due to these limitations, other papers have used MPC for CDPR control [4, 11, 13]. In [11], convex sets within the CDPR workspace are found using offline calculations, which transforms the nonlinear-MPC problem into a linear MPC problem. However, this naturally requires offline calculations and limits the workspace of the CDPR. Alternatively, in [13], it is shown that the dynamics of the CDPR can be modeled as a linear time-varying system, given a desired vertical axis (z-axis) trajectory. Using this method, MPC is implemented for trajectory generation using these linear time-varying dynamics. However, formulating the dynamics with respect to a particular axis causes errors in the state estimation of that coordinate axis to be propagated through the other axes. Further, all of the previously discussed papers implement trajectory generation for periodic trajectories. Some literature has instead investigated point-to-point trajectory generation outside the static workspace [14]. However, these methods cannot be used for periodic trajectory generation. Only one other paper implemented MPC for both periodic and point-to-point trajectory generation [4]. This method uses Chebyshev series polynomials in order to find parameters that produce a trajectory. However, the solution trajectory using Chebyshev series polynomials is only guaranteed to satisfy the constraints at the discretization nodes, but not necessarily at all points along the trajectory determined by the Chebyshev series. Further, this method is limited to polynomial trajectories given by the polynomial order of the Chebyshev series.

The approach proposed in this paper uses Euler integration and nonlinear-MPC in order to generate optimal trajectories with respect to a defined cost function. It is shown that by simply imposing rate constraints on the motor torques and a goal state that is a function of time, the generated trajectory will approach a smooth trajectory, given by a pre-defined goal trajectory. This approach has several advantages over existing literature. Euler integration guarantees that trajectories will be feasible with respect to the constraints, including between the discretization points. The proposed approach can be used to generate both point-to-point trajectories and periodic trajectories, with only a modified cost function and a particular goal trajectory. Further, this approach can be used for trajectory optimization of any functional representation of a desired trajectory, whether feasible or not. In particular, if the desired trajectory is not feasible, the algorithm will achieve the closest optimal trajectory with respect to its design variables. To the best of the authors' knowledge, this is the first demonstration of an MPC algorithm that can be used to generate trajectories outside the static workspace for a CDPR given any functional representation of a desired trajectory.

The novel contributions of this paper are the construction and implementation of nonlinear-MPC to a suspended 3 degree of freedom (DoF) CDPR that is generalizable to several different tasks. It is shown that, by simply changing the cost function and input constraints, the algorithm is tailored to two different applications, each of which improves upon the limitations of control techniques in existing literature. In particular, this paper implements nonlinear-MPC with consideration for the jerk derivative of motion and a CDPR whose trajectory must extend beyond the static workspace, with applications to a CDPR-controlled aerial camera and a CDPR-controlled motion simulator, respectively.

2. CDPR MODEL AND DYNAMICS

The CDPR model considered in this paper is a suspended 4-actuator, 3-DoF robot with a point mass payload, as shown in Fig. 1. Hence, the CDPR is over constrained and gravity is used to keep the cables in tension. The inputs to this system are T_1, T_2, T_3, T_4, the torques applied to each of the motors. The state matrix $\mathbf{z} \in \mathbb{R}^6$ is composed of the position and velocity vectors of the CDPR payload, $\mathbf{x}_p \in \mathbb{R}^3$ and $\dot{\mathbf{x}}_p \in \mathbb{R}^3$, along each of the coordinate axes, x_1, x_2, x_3, and is given by $\mathbf{z} = \begin{bmatrix} \mathbf{x}_p^\top & \dot{\mathbf{x}}_p^\top \end{bmatrix}^\top = \begin{bmatrix} x_{p,1} & x_{p,2} & x_{p,3} & \dot{x}_{p,1} & \dot{x}_{p,2} & \dot{x}_{p,3} \end{bmatrix}^\top$. The torques applied to each winch are also described in vector form as $\mathbf{T} = \begin{bmatrix} T_1 & T_2 & T_3 & T_4 \end{bmatrix}^\top$.

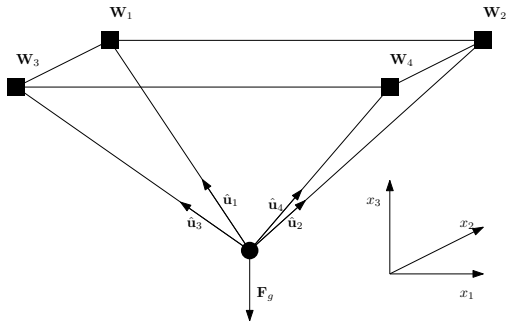


Fig. 1. A schematic of the 4-actuator, 3-DoF CDPR considered in this paper.

It is assumed there is no cable slack and that the tensions in the cables are sufficiently large to ensure the cables can be approximated as straight lines. It is also assumed that the pose of the payload is known to the nonlinear-MPC controller, whether through estimation or forward kinematics. The dynamics of this model are derived from Newton's second law and are given by

$$\frac{T_1}{r}\hat{\mathbf{u}}_1(\mathbf{x}_p)+\frac{T_2}{r}\hat{\mathbf{u}}_2(\mathbf{x}_p)+\frac{T_3}{r}\hat{\mathbf{u}}_3(\mathbf{x}_p)+\frac{T_4}{r}\hat{\mathbf{u}}_4(\mathbf{x}_p)-\mathbf{F}_g=m\ddot{\mathbf{x}}_p, \tag{1}$$

where the unit vectors $\hat{\mathbf{u}}_n(\mathbf{x}_p)$, $n=1,2,3,4$, describe the direction of cables and are defined as

$$\hat{\mathbf{u}}_n(\mathbf{x}_p)=\begin{bmatrix} W_{n,x_1}-x_{p,1} \\ W_{n,x_2}-x_{p,2} \\ W_{n,x_3}-x_{p,3} \end{bmatrix}\frac{1}{\|\mathbf{W}_n-\mathbf{x}_p\|},\ n=1,2,3,4, \tag{2}$$

$\mathbf{F}_g=\begin{bmatrix}0 & 0 & mg\end{bmatrix}^\top$ is the force of gravity on the payload, T_i is the torque applied by motor i, r is the radius of the winch of each of the motors. Further, $W_{i,j}$ represents the x_j coordinate position of motor winch i, and $x_{p,j}$ represents the x_j coordinate position of the payload. The jerk of the payload can be found by taking the time derivative of Eq. (1). This is solved as

$$\dddot{\mathbf{x}}_p=\frac{\mathrm{d}}{\mathrm{d}t}\ddot{\mathbf{x}}_p=\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{mr}\left(T_1\hat{\mathbf{u}}_1(\mathbf{x}_p)+T_2\hat{\mathbf{u}}_2(\mathbf{x}_p)+T_3\hat{\mathbf{u}}_3(\mathbf{x}_p)+T_4\hat{\mathbf{u}}_4(\mathbf{x}_p)-r\mathbf{F}_g\right)\right)=\mathbf{J}_x\dot{\mathbf{x}}_p+\mathbf{J}_T\dot{\mathbf{T}}, \tag{3}$$

where $\mathbf{J}_x=\frac{\partial\ddot{\mathbf{x}}_p}{\partial\mathbf{x}_p}$ and $\mathbf{J}_T=\frac{\partial\ddot{\mathbf{x}}_p}{\partial\mathbf{T}}$ are Jacobians whose entries are computed as $\frac{\partial\ddot{x}_{p,i}}{\partial T_j}=\frac{1}{mr}\frac{(W_{j,i}-x_{p,i})}{\|\mathbf{W}_j-\mathbf{x}_p\|}$ and

$$\frac{\partial\ddot{x}_{p,i}}{\partial x_{p,j}}=\begin{cases}\frac{1}{mr}\sum_{n=1}^{4}\left(\frac{T_n(W_{n,i}-x_{p,i})^2}{\|\mathbf{W}_n-\mathbf{x}_p\|^3}-\frac{T_n}{\|\mathbf{W}_n-\mathbf{x}_p\|}\right) & i=j \\ \frac{1}{mr}\sum_{n=1}^{4}\frac{T_n(W_{n,i}-x_{p,i})(W_{n,j}-x_{p,j})}{\|\mathbf{W}_n-\mathbf{x}_p\|^3} & i\neq j\end{cases}. \tag{4}$$

The MPC algorithm presented in Section 3 requires a discrete-time formulation of the system dynamics. This is described by

$$\mathbf{z}^{k+1}=\mathbf{f}(\mathbf{z}^k,\mathbf{T}^k)=\begin{bmatrix}\mathbf{x}_p^k+h\dot{\mathbf{x}}_p^k \\ \dot{\mathbf{x}}_p^k+h\left(\frac{1}{mr}\left(T_1^k\hat{\mathbf{u}}_1(\mathbf{x}_p^k)+T_2^k\hat{\mathbf{u}}_2(\mathbf{x}_p^k)+T_3^k\hat{\mathbf{u}}_3(\mathbf{x}_p^k)+T_4^k\hat{\mathbf{u}}_4(\mathbf{x}_p^k)-r\mathbf{F}_g\right)\right)\end{bmatrix}, \tag{5}$$

where the superscript k denotes a variable evaluated at time step k, h is the length of the time step, and Euler integration is used to perform the discrete-time propagation of the system dynamics in Eq. (1). Euler integration is chosen since it guarantees that trajectories will be feasible with respect to the constraints, including between the discretization points. Although Euler integration is chosen for this implementation, other numerical integration schemes can be employed, as long as they result in a discrete-time update equation of the form $\mathbf{z}^{k+1}=\mathbf{f}(\mathbf{z}^k,\mathbf{T}^k)$.

3. MODEL-PREDICTIVE CONTROL METHODOLOGY AND IMPLEMENTATION

The model derived in the previous section is used to formulate the nonlinear-MPC algorithm in this section. The dynamic model of the CDPR is the same in both the aerial videography and the dynamic workspace application. Since each application has different operating criteria, the cost function and the system constraints are tailored appropriately in the following subsections.

3.1. Model-Predictive Control for Aerial Videography

The operating constraints and desired operation of an aerial videography CDPR depend on several factors. The main goal is to have a smooth trajectory while still maintaining minimal error with respect to a goal state. The operating parameters of the motors and other hardware must not be exceeded. The camera payload must also avoid contact with the ground. Considering all these factors, the finite time horizon optimal MPC algorithm for an aerial camera is formulated, given by

$$\min_{\mathbf{T}^0,\ldots,\mathbf{T}^{N-1}} \sum_{k=0}^{N-1} C(\mathbf{z}_k, \mathbf{T}^k) \tag{6}$$

$$\text{subject to} \quad \mathbf{z}^{k+1} = \mathbf{f}(\mathbf{z}^k, \mathbf{T}^k) \quad k = 0,\ldots,N-1, \tag{7}$$

$$\dot{\mathbf{T}}^{k+1} = \frac{\mathbf{T}^{k+1} - \mathbf{T}^k}{h} \quad k = 0,\ldots,N-1, \tag{8}$$

$$0 < T_n^k < T^{max} \quad n = 1,2,3,4, \tag{9}$$

$$|\dot{T}_n^k| < \dot{T}^{max} \quad n = 1,2,3,4, \tag{10}$$

$$x_{p,3}^{min} < x_{p,3}^k < x_{p,3}^{max}, \tag{11}$$

$$\ddot{\mathbf{x}}_p^k = \mathbf{J}_x \dot{\mathbf{x}}_p^k + \mathbf{J}_T \dot{\mathbf{T}}_p^k, \tag{12}$$

$$|\ddot{x}_{p,i}^k| < \ddot{x}_p^{max} \quad \forall i \in 1,2,3, \tag{13}$$

where $T_{end} = Nh$ is the finite time horizon of the optimal control problem, N is the number of discrete time steps in the prediction horizon, and the state is initialized with $\mathbf{z}^0$ matching the measured state at the time step coinciding with the beginning of the prediction horizon. In the version of the MPC algorithm that calculates jerk analytically, the cost function is defined as

$$C(\mathbf{z}_k, \mathbf{T}^k) = \alpha \sum_{i=1}^{3} (x_{p,i}^k - x_{p,i}^{goal})^2 + \beta \sum_{n=1}^{4} (T_n^k)^2 + \gamma \sum_{n=1}^{4} (\dot{T}_n^k)^2 + \delta \sum_{i=1}^{3} (\ddot{x}_{p,i}^k)^2, \tag{14}$$

where $x_{p,i}^{goal}$ is a pre-defined function that simulates the desired position of the payload with respect to some moving object on the ground. Although the function is defined in advance for simplicity, only the current value of this function is known to the algorithm. This simulates the camera following an object that has self-determined motion which cannot be predicted, such as a person or an animal. The tunable parameters α, β, γ, and δ are chosen based on experimental observation. Intuitively, the α and δ parameters are the most important for the application of aerial videography, since these correspond to the relative weights of the error in position and the jerk, respectively. The β and γ variables can be set fairly small, but must be included in the cost function in order to avoid an indefinite solution since the CDPR is over constrained. Further details on these tuning parameters are provided in the numerical examples in Section 4. The derivative of torque is found using a linear approximation with respect to the discrete step size, as shown in Eq. (8). The rate limits imposed on $\ddot{x}_{p,i}$ are used to avoid a bang-bang trajectory or other similar trajectories with very large instantaneous jerk values, shown in Eq. (13). The constraints imposed on T_i are either given by or can be calculated from the parameters of particular motor hardware, and are given by Eq. (9). The constraints on

$\dot{T}_i$ are necessary to ensure feasible motor inputs, given by Eq. (10). Finally, the constraint on $x_{p,3}$ denotes the maximum vertical workspace, which is necessary to avoid colliding with the ground or exceeding the constraints of the winches, where the CDPR would no longer be suspended, and is given by Eq. (11).

Although it is desired for the MPC algorithm to generate a trajectory that is optimal with respect to jerk, calculating the jerk analytically at each step k is computationally expensive. Therefore, this MPC algorithm is modified in three different ways. These modifications attempt to simplify the MPC algorithm while still approximating the desired operation of the aerial videography CDPR. The first calculates the jerk function using a linear approximation, which is given by

$$\dddot{\mathbf{x}}_p^{k+1} = \frac{\ddot{\mathbf{x}}_p^{k+1} - \ddot{\mathbf{x}}_p^{k}}{h}. \tag{15}$$

Here, the jerk derivative is assumed to be a linear function of time between time steps. This approximation uses the cost function defined by Eq. (14).

The remaining two simplified algorithms do not directly optimize over $\ddot{\mathbf{x}}_p$ in the cost function. Each of these has a modified cost function that is given by

$$C(\mathbf{z}_k, \mathbf{T}^k) = \alpha \sum_{i=1}^{3} (x_{p,i}^k - x_{p,i}^{goal})^2 + \beta \sum_{n=1}^{4} (T_n^k)^2 + \gamma \sum_{n=1}^{4} (\dot{T}_n^k)^2. \tag{16}$$

The first of these simplified algorithms uses the cost function given by Eq. (16), but retains the constraints on minimum and maximum jerk in Eq. (13). The second of these algorithms instead removes all jerk calculations and is used as a baseline comparison, including removing Eq. (13) from the MPC algorithm.

3.2. Model-Predictive Control for Trajectories That Extend Beyond the Static Workspace

Similar to the application of aerial videography, the MPC algorithm for an expanded dynamic workspace uses a finite time horizon optimal nonlinear-MPC algorithm. Compared to the aerial videography application, the model of the CDPR is the same but the operation criteria is very different. Unlike the aerial videography application, the pre-selected ideal trajectory is known to the CDPR, though is not necessarily a feasible trajectory given particular input constraints. If the goal trajectory is feasible, the trajectory of the CDPR will closely follow this ideal trajectory, minimizing the error with respect to the value of the ideal trajectory at each time step. As a consequence, this algorithm can be used to confirm feasibility of a trajectory with respect to particular system constraints; however, an inability to find a trajectory using nonlinear-MPC does not necessarily imply a desired trajectory is infeasible. As before, the CDPR must also not exceed the hardware operating parameters. Considering these factors, the nonlinear-MPC algorithm used to achieve dynamic equilibrium is given by Eqs. (6)-(11), where the cost function is described as

$$C(\mathbf{z}^k, \mathbf{T}^k) = \alpha_i \sum_{i=1}^{3} (x_{p,i}^k - x_{p,i}^{goal,k})^2 + \beta \sum_{n=1}^{4} (T_n^k)^2 + \gamma \sum_{n=1}^{4} (\dot{T}_n^k)^2 + \delta_j \sum_{j=1}^{3} (\dot{x}_{p,j}^k)^2. \tag{17}$$

Here, the dependence on $\ddot{\mathbf{x}}_p$ has been removed from the cost function since the payload undergoes constant motion, and therefore acceleration, in order to sustain a trajectory outside of the static workspace. Although the bounds of $x_{p,3}^{max}$ in (11) are not strictly necessary for a CDPR operating in its dynamic workspace, it is retained for this simulation since exceeding the height of the winches causes the CDPR to be prone to cable tangle. The tunable parameters α_i, β, γ, and δ_j are chosen based on experimental observation. As before, the α_i parameter is the most important in order to minimize the position error with respect to the desired trajectory. As before, β and γ, the coefficients of the torque and torque rates, must be included in the cost function in order to avoid solutions with large control inputs. In this cost function, δ_j is used to damp

oscillations on the $x_{p,j}$ pose. In particular, the δ_j term is useful for certain trajectories where one or several of the desired $x_{p,j}$ values is not defined. If no particular $x_{p,j}$ value is desired, then α_j is set to zero, and δ_j is set to a nonzero value in order to damp oscillations on that axis. This makes the algorithm extremely versatile in its construction. It is noted in [6] that, in general, the frequency of oscillation of the CDPR payload is constrained. For example, for simple harmonic motion of three-DoF CDPRs, the natural frequency of the oscillations and the $x_{p,3}$ axis is coupled according to the expression $\omega_n = \sqrt{\frac{g}{c_{x_{p,3}}}}$, where $c_{x_{p,3}}$ is the center of oscillation in the $x_{p,3}$ axis. Rather than solving a feasible trajectory analytically, this nonlinear-MPC algorithm can be utilized to find the optimal trajectory given the particular design parameters for any desired trajectory function. Finally, the goal state is defined as a function of time rather than as a static point. This is denoted in Eq. (17) by the k subscript in $x_{p,i}^{goal,k}$. Thus, the cost at each step k compares the position of the payload at time k with the dynamic goal state at step k. This allows the payload to maintain a smooth trajectory in dynamic equilibrium.

Rather than following periodic motion, the CDPR can instead be defined to follow a point-to-point trajectory with points described outside the static workspace. This is achieved by the goal-state formulation

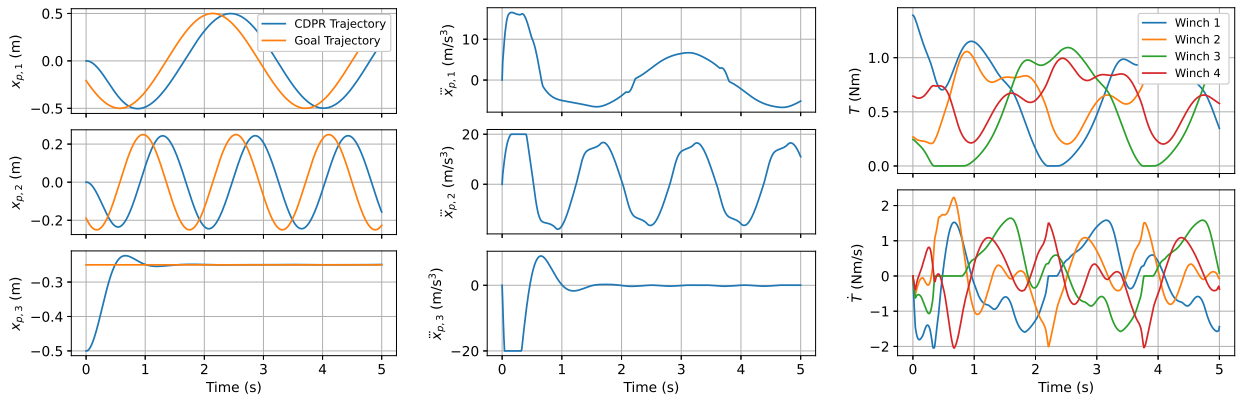
$$\mathbf{x}_{p,i}^{goal,k} = x_{p,i}^{0} + \sum_{d=1}^{D} (x_{p,i}^{d} - x_{p,i}^{0}) e^{-\varepsilon(t-T^d)^2}, \tag{18}$$

where the index d iterates over the desired points to be approached during the trajectory, ε is a tuning parameter to describe the time-scale of the trajectory, and T^d is the time when the goal position should occur in the trajectory. Finally, $x_{p,i}^{0}$ represents the initial position of the payload in the x_i coordinate axis, and $x_{p,i}^{d}$ represents the coordinate value of the desired point in the x_i coordinate axis. This function is derived from the Gaussian function and is analogous to a soft constraint on the desired position. The Gaussian function is chosen since it is a simple method of defining a point-to-point trajectory using a function. This point-to-point trajectory utilizes the same cost function as described in Eq. (17), but with $\delta_j = 0$.

4. RESULTS

The MPC algorithms proposed are validated through numerical simulations of a 4-cable, 3-DoF suspended CDPR in this section. The geometry of the CDPR is defined by the model derived in Section 2. The simulator chosen for these nonlinear optimization algorithms is CasADi [15]. The winch locations are defined at positions $\mathbf{W}_1 = \begin{bmatrix}-0.5 & -0.5 & 0\end{bmatrix}^\top$ m, $\mathbf{W}_2 = \begin{bmatrix}-0.5 & 0.5 & 0\end{bmatrix}^\top$ m, $\mathbf{W}_3 = \begin{bmatrix}0.5 & -0.5 & 0\end{bmatrix}^\top$ m, $\mathbf{W}_4 = \begin{bmatrix}0.5 & 0.5 & 0\end{bmatrix}^\top$ m. The payload is initialized to be at position $\mathbf{x}_p^0 = \begin{bmatrix}0 & 0 & -0.5\end{bmatrix}^\top$ m.

The payload is selected to be 2 kg and the winch radius on each of the motors is selected to be 0.05 m. The constraint imposed on motor torque, denoted by T^{max} in Eq. (9), is chosen as 10 N·m for all simulations based on common hobby motor specifications. In Eq. (10), the value of $\dot{T}^{max}$ was selected conservatively to be 20 N·m/s. This value can be adjusted based on the evaluated slew rate for the particular system. In Eq. (13) the value of $\ddot{x}_p^{max}$ is selected conservatively to be 20 m/s^3 for all cases except where the jerk parameter is not considered. This parameter represents hard constraints on the payload jerk, and can be adjusted based on operating requirements. In Eq. (11), $x_{p,3}^{min}$ is selected to be -1 m, which denotes a workspace constraint at -1 m in the vertical direction. This is an arbitrary selection and can be adjusted based on the workspace constraints of the particular system. Further, $x_{p,3}^{max}$ is selected to be 0 m which corresponds to the height of the winches. In the aerial videography simulation, the number of discrete points in the time horizon is chosen to be $N = 100$. This can be increased to improve the accuracy of the solver at the expense of computation time. In the dynamic workspace simulation, the number of discrete points is instead chosen to be $N = 200$. In both simulations, the time horizon T_{end} is chosen as 1 second so that the algorithm has some knowledge of future trajectories to inform its current input while still converging in a reasonable amount of



(a) (b) (c)

Fig. 2. Trajectory generated with the MPC formulation and analytical jerk calculation in Eq. (6), where α = 1000, β = 0.01, γ = 0.01, δ = 0.01. The plots include (a) position, (b) jerk, and (c) torque and torque derivative versus time.

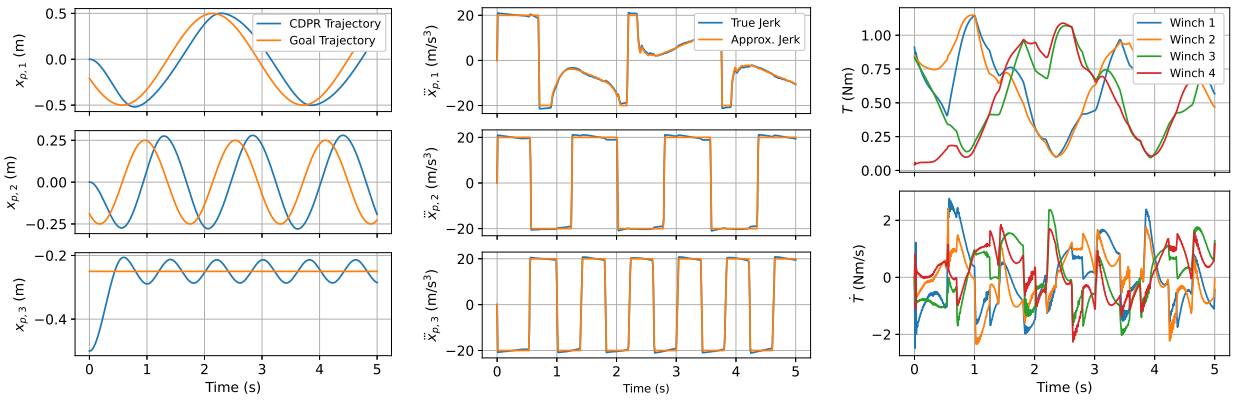
time. As a consequence of $T_{end} = Nh$, the step size h is 0.01 s and 0.005 s for the aerial videography and the dynamic workspace simulations, respectively. Setting this step size higher for the dynamic workspace simulation resulted in system instability, caused by the loss of accuracy in the integration.

The full trajectory of each simulation is plotted over a 5 second time horizon. Since the MPC time horizon is limited to 1 second, this means the MPC does not have full information about the future goal states or positional states. For each iteration of MPC, the initial guess of the next trajectory is set to be the current position for all steps k, and the initial guess of the optimal inputs is instantiated to be the optimal trajectory found during the last iteration. The remaining step which was not computed during the previous MPC iteration has its inputs instantiated to the last computed optimal input. At any iteration, if a solution is not found, then the previous iteration's solution is used for the following step until a valid solution can be found.

4.1. Model-Predictive Control for Aerial Videography Results

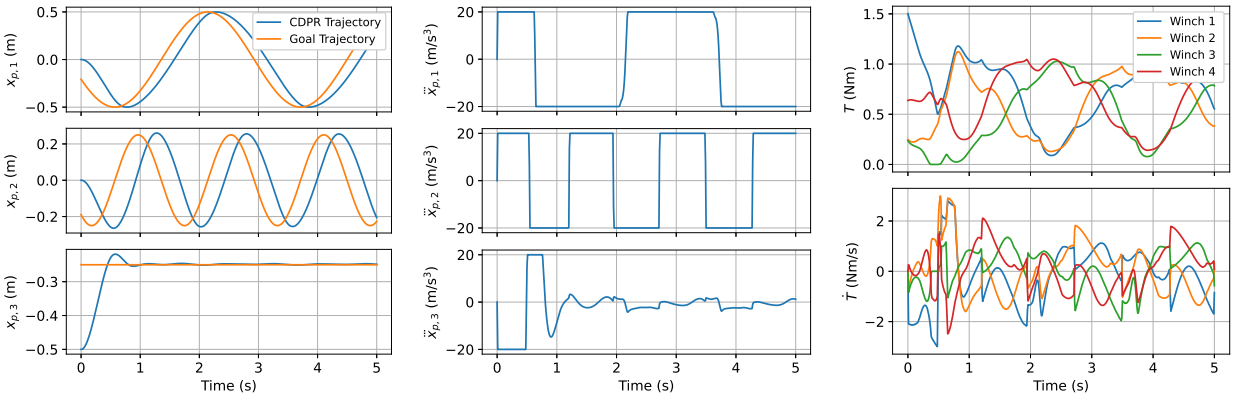
An example goal trajectory is defined whose output is a figure-eight. The functional representation of this is given in discrete time as $\mathbf{x}_p^{goal,k} = \begin{bmatrix} 0.5\cos(2(hk+t_0)) & 0.5\sin(2(hk+t_0))\cos(2(hk+t_0)) & -0.25 \end{bmatrix}^\top$ m. This simulates a camera following directly overhead with respect to a desired point on the ground. This goal state trajectory operates on the boundary of the static workspace of the CDPR, further emphasizing the worst-case scenario performance of the CDPR.

The results of the MPC algorithm with analytical jerk derivative calculation are shown in Fig. 2. These results show that, while the $x_{p,1}$ trajectory closely matches the desired trajectory, the $x_{p,2}$ trajectory is somewhat out of phase. Yet, the torque and torque derivative values remain within the constraints, and the jerk remains nominal. This illustrates the trade-off between smoothness and position tracking. The parameters of each of the jerk approximation methods in Fig. 3, Fig. 4, and Fig. 5 are selected based on the values of the full analytical jerk evaluation, shown in Fig. 2, in order to illustrate the effectiveness of each MPC algorithm. Compared to Fig. 2, Fig. 3 has slightly more divergence of the payload positions from the desired positions while also saturating the $\ddot{x}_{p,2}$ trajectory. This illustrates the limitations of the jerk approximation when compared to the full analytical jerk calculation. As expected, the plots which do not optimize over jerk in Fig. 4 and Fig. 5 have better positional tracking when compared to any of the algorithms which optimize with respect to jerk. However, this comes with particular drawbacks. In the case of Fig. 4, this means the jerk is always at saturation of its constraint limits, while in Fig. 5 this comes at the cost of very large instantaneous jerk values. This indicates that the jerk bounds are indeed providing a smoother trajectory compared



(a) (b) (c)

Fig. 3. Trajectory generated with the MPC cost function and linear approximation of jerk in Eq. (15), where $\alpha = 1000$, $\beta = 0.01$, $\gamma = 0.01$, $\delta = 0.01$. The plots include (a) position, (b) jerk, and (c) torque and torque derivative versus time.

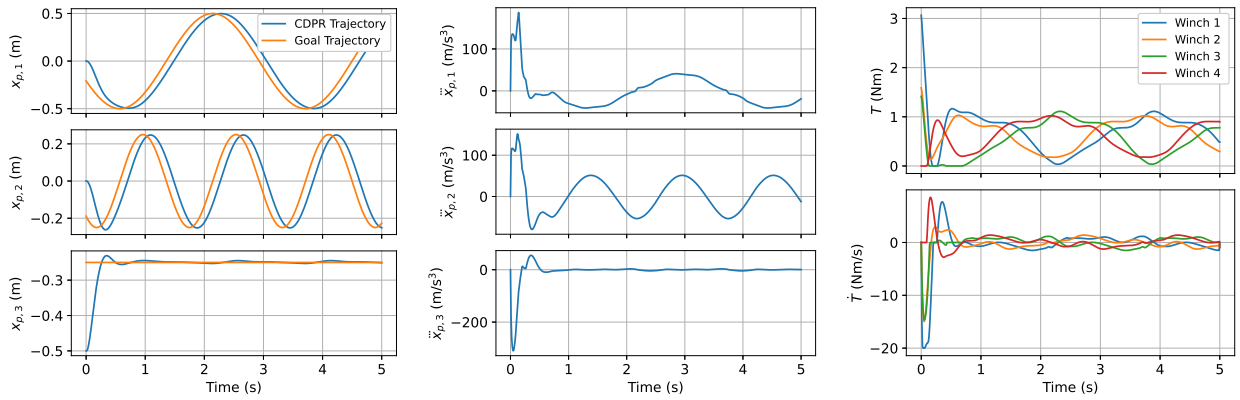


(a) (b) (c)

Fig. 4. Trajectory generated with the MPC cost function in Eq. (16) and no optimization over jerk, but retaining the jerk constraints, where $\alpha = 1000$, $\beta = 0.01$, $\gamma = 0.01$. The plots include (a) position, (b) jerk, and (c) torque and torque derivative versus time.

to the optimization without considering jerk. Based on these results, it appears that the trajectory generated by the nonlinear-MPC algorithm using the jerk approximation is stuck in a local minimum compared to the analytical jerk calculation. This is noted by the saturation of the $\ddot{x}_{p,3}$ axis and the lack of saturation of $\ddot{x}_{p,1}$ axis, which is different than the results in Fig. 2, Fig. 3, and Fig. 4. With additional tuning of the cost function parameters, this could be adjusted, but these were chosen to match the other trials for consistency.

Since these algorithms do not have information about the future position of the goal state when determining the optimal trajectory, none of these trajectories are able to converge to zero position error. This information was hidden in order to simulate camera tracking of a moving object, and to demonstrate the worst-case performance of the algorithm. The position error could be further reduced by increasing the relative cost of position error, but this has a tradeoff of high jerk and torque values. Alternatively, as shown in the following section, providing the MPC algorithm with full information of the underlying goal trajectory would allow the position error to converge to zero.



(a) (b) (c)

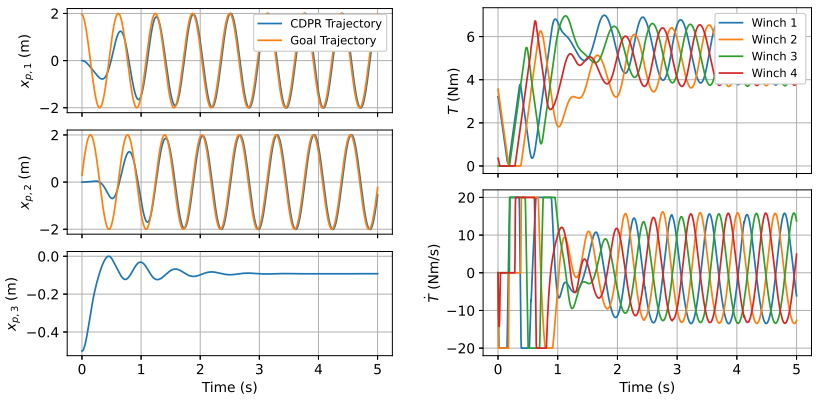
Fig. 5. Trajectory generated with the MPC cost function in Eq. (16) and no optimization over jerk or jerk constraints, where $\alpha = 1000$, $\beta = 0.01$, $\gamma = 0.01$. The plots include (a) position, (b) jerk, and (c) torque and torque derivative versus time.

It should also be noted that the algorithms were not solved in real time. There are likely several reasons for this, including the use of a relatively computationally inefficient solver (MUMPS) and coding language (Python). The nonlinear jerk function in Eq. (3) also likely contributes to the computational complexity.

4.2. Model-Predictive Control for Trajectories That Extend Beyond the Static Workspace

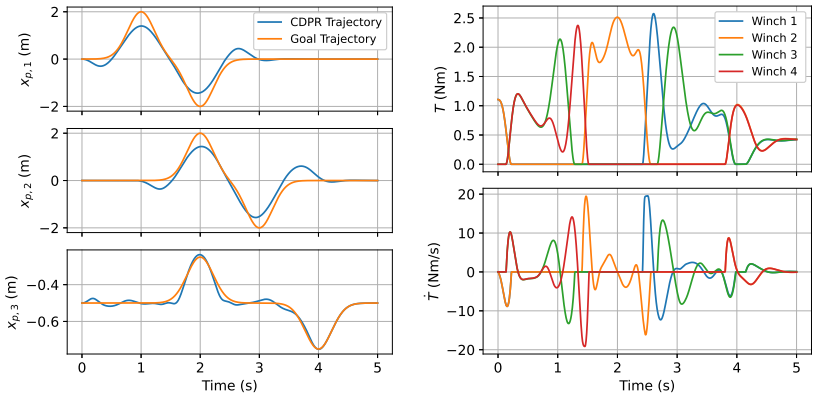
Example goal trajectories are defined for the application of periodic and point-to-point motion beyond the static workspace. The periodic trajectory is chosen to be a circle, and is described in discrete time as $\mathbf{x}_p^{goal,k} = \begin{bmatrix} 2\cos(10(hk+t_0)) & 2\sin(10(hk+t_0)) \end{bmatrix}^\top$ m. As discussed in [6], since the $x_{p,3}$ axis and the natural frequency are coupled, the desired $x_{p,3}$ parameter does not need to be defined. Thus, in Eq (17), α_3 is set to zero. However, the oscillations on $x_{p,3}$ need to be damped, so δ_3 is set to 10, and $\delta_j = 0$ for $j = 1,2$. In contrast, the points along the point-to-point trajectory are given by $\mathbf{x}_p^1 = \begin{bmatrix} 2 & 0 & -0.5 \end{bmatrix}^\top$ m, $\mathbf{x}_p^2 = \begin{bmatrix} -2 & 2 & -0.25 \end{bmatrix}^\top$ m, $\mathbf{x}_p^3 = \begin{bmatrix} 0 & -2 & -0.5 \end{bmatrix}^\top$ m, and $\mathbf{x}_p^4 = \begin{bmatrix} 0 & 0 & -0.75 \end{bmatrix}^\top$ m, where $\mathbf{x}_p^1$, $\mathbf{x}_p^2$, $\mathbf{x}_p^3$, $\mathbf{x}_p^4$ are desired to occur at 1 second, 2 seconds, 3 seconds, and 4 seconds, respectively. These are formulated into a goal function according to Eq. (18) with $\varepsilon = 10$ based on experimental observation. Each of these trajectories operates outside of the static workspace. Results with the proposed MPC algorithm implemented with these goal trajectories are shown in Fig. 6 and Fig. 7.

The trajectory in Fig. 7 is tracked with only moderate position error, while the trajectory in Fig. 6 is tracked with minimal error. Despite the high cost parameter on the position error, the torques never reach their upper constraint bounds. Although the torque derivative is observed to saturate, this is feasible since this only represents a transient effect. Since the torque is not operating at its bounds of operation and has feasible rate constraints, the positional trajectory is able to effectively track the goal position. Further, since the goal state is a function of time, the optimal trajectory is such that the position over time approaches the trajectory given by the goal function. Thus, with this formulation of the goal state and when the torques are operating within their bounds, the optimal solution is shown to approach the goal state and approximate a smooth trajectory. It is noted that the dynamic workspace simulation, particularly the point-to-point trajectory simulation, is extremely sensitive to the finite time horizon window. This is because the CDPR has to gain momentum in order to reach desired points outside of the static workspace, which it cannot do if a point is outside its time horizon window. Thus, the best results are achieved with a large time horizon. Similar to the aerial videography MPC algorithm, the dynamic workspace MPC algorithm could not perform in real time, which



(a) (b)

Fig. 6. Trajectory generated with the dynamic workspace MPC formulation for circular periodic motion trajectory, where $\alpha = 100$, $\beta = 0.01$, $\gamma = 0.01$, $\delta = 10$. The plots include (a) position and (b) torque and torque derivative versus time.



(a) (b)

Fig. 7. Trajectory generated with the dynamic workspace MPC formulation for point-to-point goal trajectory, where $\alpha = 100$, $\beta = 0.01$, $\gamma = 0.01$, $\delta = 10000$. The plots include (a) position and (b) torque and torque derivative versus time.

will be addressed in future work. One method to increase computation speed would be to take multiple steps along the generated trajectory before re-optimizing. However, this can still lead to problems, as an increased step size can lead to tracking error if the system encounters noise or disturbances.

5. CONCLUSIONS

This paper investigated two different applications of a novel nonlinear-MPC algorithm to a 4-cable 3-DoF suspended CDPR. This paper showed that introducing a calculation of the payload jerk could effectively produce an optimally smooth trajectory. It was also shown that several jerk approximations were also able to fundamentally capture this same trajectory. In particular, simply adding rate constraints on the torque and position is shown to have only slightly worse performance when compared to the analytical jerk calculation. Further work would be useful to determine whether these approximations still effectively capture the dynamics of the analytical jerk calculation for a variety of different configurations and trajectories. This paper

further showed that nonlinear-MPC could effectively find a smooth trajectory outside of the static workspace by using Euler integration and imposing rate constraints on the torque, given a goal function that depends on time. However, this also represents a drawback since this algorithm requires the formulation of a goal trajectory, which must be a function with respect to time. Furthermore, although a point-to-point trajectory was generated by using soft positional constraints, there is no guarantee of the magnitude of the positional error with respect to the desired points of the trajectory. An alternative formulation may be constructed such that the positional error must be sufficiently small with respect to certain goal states at given time intervals, i.e., a hard constrained point-to-point trajectory. While this nonlinear-MPC algorithm is flexible to allow for this point-to-point trajectory to be formulated in such a way, it is unclear whether this algorithm would still return a feasible solution for this application. Another possible extension may be to introduce a more complex model of the CDPR, such as a non-point mass payload or a fully-constrained CDPR.

REFERENCES

1. Pott, A. "Cable-driven parallel robots - theory and application." In "Springer Tracts in Advanced Robotics," , 2018.
2. Tang, X. "An overview of the development for cable-driven parallel manipulator." *Advances in Mechanical Engineering*, Vol. 6, p. 823028, 2014.
3. Khoshkam, S., Khosravi, M.A. and FesharakiFard, R. "Model predictive control for a 3-dof suspended cable robot based on laguerre functions." In "International Conference on Electrical Engineering," pp. 827–832, 2022.
4. Xiang, S., Gao, H., Liu, Z. and Gosselin, C. "Trajectory optimization for a six-dof cable-suspended parallel robot with dynamic motions beyond the static workspace." In "IEEE International Conference on Robotics and Automation," pp. 3903–3908, 2020.
5. Phuoc Tho, T. and Truong Thinh, N. "Using a cable-driven parallel robot with applications in 3d concrete printing." *Applied Sciences*, Vol. 11, p. 563, January 2021.
6. Mottola, G., Gosselin, C. and Carricato, M. "Dynamically feasible periodic trajectories for generic spatial three-degree-of-freedom cable-suspended parallel robots." *Journal of Mechanisms and Robotics*, Vol. 10, No. 3. 031004, 2018.
7. Gosselin, C. *Global Planning of Dynamically Feasible Trajectories for Three-DOF Spatial Cable-Suspended Parallel Robots*, pp. 3–22. Springer, Berlin, Heidelberg, 2013.
8. Jiang, X., Barnett, E. and Gosselin, C. "Periodic trajectory planning beyond the static workspace for 6-dof cable-suspended parallel robots." *IEEE Transactions on Robotics*, Vol. 34, No. 4, pp. 1128–1140, 2018.
9. Zhang, Z., Shao, Z., You, Z., Tang, X., Zi, B., Yang, G., Gosselin, C. and Caro, S. "State-of-the-art on theories and applications of cable-driven parallel robots." *Frontiers of Mechanical Engineering*, Vol. 17, No. 3, p. 37, 2022.
10. Santos, J.C., Gouttefarde, M. and Chemori, A. "A nonlinear model predictive control for the position tracking of cable-driven parallel robots." *IEEE Transactions on Robotics*, Vol. 38, No. 4, pp. 2597–2616, 2022.
11. Song, C. and Lau, D. "Workspace-based model predictive control for cable-driven robots." *IEEE Transactions on Robotics*, Vol. 38, No. 4, pp. 2577–2596, 2022.
12. Barrette, G. and Gosselin, C.M. "Determination of the dynamic workspace of cable-driven planar parallel mechanisms." *Journal of Mechanical Design*, Vol. 127, No. 2, pp. 242–248, 2005.
13. Xiang, S., Gao, H., Liu, Z. and Gosselin, C. "Dynamic transition trajectory planning of three-DOF cable-suspended parallel robots via linear time-varying MPC." *Mechanism and Machine Theory*, Vol. 146, p. 103715, 2020.
14. Jiang, X., Barnett, E. and Gosselin, C. "Dynamic point-to-point trajectory planning beyond the static workspace for six-DOF cable-suspended parallel robots." *IEEE Transactions on Robotics*, Vol. 34, No. 3, pp. 781–793, 2018.
15. Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B. and Diehl, M. "CasADi – A software framework for nonlinear optimization and optimal control." *Mathematical Programming Computation*, Vol. 11, No. 1, pp. 1–36, 2019.