Rapid Computation of Optimally Safe Tension Distributions for Cable-Driven Robots

Per Henrik Borgstrom, Brett L. Jordan, Gaurav S. Sukhatme, Senior Member Maxim A. Batalin, Member and William J. Kaiser, Senior Member

Abstract—In this paper, we present a novel LP formulation that yields “Optimally Safe” tension distributions by means of the introduction of a slack variable. The slack variable also enables explicit computation of a near-optimal, feasible starting point. This, in turn, enables rapid computation of the “Optimally Safe” tension distributions. The formulation also contains a parameter that can be used to steer cable tensions toward desired regions of operation. We present static results from two simulated robotic systems that demonstrate the ability of our formulation to avoid tension limits. Simulated execution of highly dynamic trajectories on both systems demonstrates rapid computation abilities. Furthermore, we present experimental results from a real robotic system which further validate the importance of safe tension distributions.

I. INTRODUCTION

Cable-driven robots consist of computer-driven actuators that enable controlled release of cables. These cables, in turn, may support a wide range of end-effector systems. The actuators can be stationary or mobile and are positioned in the extremities of the robot workspace.

Cable-driven systems possess several advantages over rigid-link parallel manipulators. For one, due to the low mass of cables as compared to rigid arms, they are capable of highly dynamic motion. Furthermore, cable-driven systems can potentially span large workspaces. On the other hand, the inability of cables to exert positive, pushing forces complicates analysis. For example, the feasible workspace of cable-driven platforms is severely impacted. The authors of [1] consider the Wrench Feasible Workspace (WFW) of a cable-driven platform, defining it as the workspace within which a particular set of wrenches can be applied by the end-effector. The authors of [2] and [3] consider the wrench-closure workspace (WCW), which is defined as the space within which the platform can exert any wrench.

The inability of cables to exert pushing forces also significantly impacts motion control of cable-driven robots. The authors of [4] design a controller that guarantees positive tensions and implement it on a four-cable-driven planar platform. Fang et al. [5] use PD compensation to control the 7-cable-driven 6-DOF SEGESTA platform. Furthermore, Kino et al. [6] propose a robust PD control method with adaptive compensation wherein an external sensor (camera) is used to eliminate positioning errors due to errors in actuator placement and cable-length measurement. Other work in design and control of cable-driven robots includes the WARP [7] and FALCON [8] systems.

There exists some work in determining one-norm minimal tension distributions for redundantly-actuated cable-driven robots. The authors of [5] derive analytical expressions for optimal tension distributions in a completely-constrained six degree-of-freedom robot. Their method reduces the computation to a one variable linear program (LP) with the sum of all tensions used as the objective function. The optimization is then solved by exhaustively checking all extreme points. This method does not extend to increased degrees of freedom. The authors of [11] present a highly-efficient LP solver for a four-cable-driven, two-DOF robot. Other previous work [12], [13], [14] has also suggested the use of LP methods to compute one-norm minimal tensions distributions.

There exists some work in determining one-norm minimal tension distributions for redundantly-actuated cable-driven robots. The authors of [5] derive analytical expressions for optimal tension distributions in a completely-constrained six degree-of-freedom robot. Their method reduces the computation to a one variable linear program (LP) with the sum of all tensions used as the objective function. The optimization is then solved by exhaustively checking all extreme points. This method does not extend to increased degrees of actuation redundancy. The authors of [11] present a highly-efficient LP solver for a four-cable-driven, two-DOF robot. Other previous work [12], [13], [14] has also suggested the use of LP methods to compute one-norm minimal tensions distributions.

There are several drawbacks associated with one-norm minimal tension distribution methods. First, iterative LP solvers used to solve such optimizations require feasible starting points. In general, finding these starting points is non-trivial, and a special Phase 1 LP must be solved before the original problem is solved by a Phase 2 solver. While previous
work [10] suggests using the tension distribution computed during the previous servo loop as a starting point, there is no guarantee that this distribution is feasible, and a Phase 1 routine must still be executed. Also, because the optimal point in an LP is always located at an extreme point of the feasible polyhedron, tension distributions found through one-norm minimal formulations are prone to discontinuities, as the optimal operating point may jump from one extreme point to another between successive computations [15],[16]. The resulting step changes in tension may excite high frequency oscillations in cables and otherwise degrade performance.

Other works have suggested the use of bounded quadratic programming (QP) methods in computing two-norm optimal tension distributions for cable-driven systems. The authors of [15] consider a cable array crane and provide efficient solutions for minimum-norm tension distributions, which are shown to yield improved energy efficiency and continuous cable tensions. However, their methods do not extend to redundantly-restrained systems. The authors of [17] use Dykstra’s alternating projection algorithm to find the minimum-norm tension distribution for a pair of simulated robotic systems. This is an iterative convex optimization method requiring a large number of iterations to converge.

The LP and QP methods discussed above both suffer from another significant drawback: because they aim to minimize cable tensions in the one-norm or two-norm sense, respectively, resulting tensions frequently reside on the lower tension limit. This results in low robot stiffness and leaves the system prone to slack conditions [16]. To address this issue, the authors of [16] propose a non-iterative method of finding “safe” tension distributions that avoid tension limits. Furthermore, the authors prove that the resulting tension distributions are continuous. However, the proposed method suffers from poor execution time for even moderately complex systems, as the computational intensity increases combinatorially with the number of cables.

In this paper, we present a novel LP formulation that yields “Optimally Safe” (OS) tension distributions by means of the introduction of a slack variable. The introduction of S also enables direct computation of a near-optimal, feasible starting point. This, in turn, enables highly efficient computation of OS tension distributions. The formulation also contains a parameter α that can be used to steer cable tensions toward desired regions of operation.

The remainder of this paper is structured as follows: In Section II, we provide a brief background on Linear Programming. In Section III, we describe previous work in safe tensions and detail our novel LP formulation and the methods used to efficiently compute OS tension distributions. In Section IV, we introduce two cable-driven robotic systems for which we provide static tension distribution results that demonstrate the effect of our formulation on tension distributions. In Section V, we demonstrate the computational efficiency of our solution method by providing results from simulated execution of highly dynamic trajectories. In Section VI, we provide experimental results gathered from a real robotic system, NIMS-PL [11], and, in Section VII, we conclude our paper and describe future research thrusts.

II. LINEAR PROGRAMMING

Before describing our method of computing optimal tension distributions, we begin by providing a brief introduction to LP theory. A more complete review of LP theory is available in, for example, [18]. A linear program is one in which the objective is to minimize a linear function subject to linear equality and inequality constraints. Because an equality constraint can be represented as two inequality constraints, it is sufficient to consider only inequalities. Thus, a typical LP problem might be:

\[
\begin{align*}
\hat{p} &= \min \, \lambda^T e \\
\text{s.t.} \quad A_{ineq} \hat{c} &\leq b_{ineq},
\end{align*}
\]

where \(c \in \mathbb{R}^N\) is the optimization variable, \(\lambda \in \mathbb{R}^N\) is the cost vector, and \(A_{ineq} \in \mathbb{R}^{M \times N}\) and \(b_{ineq} \in \mathbb{R}^M\) represent the inequality constraints. Given such a problem, either (1) the inequality constraints are infeasible, in which case we define \(\hat{p} = \infty\), (2) the problem is unbounded, in which case \(\hat{p} = -\infty\), or (3) the problem is bounded and the feasible region is non-empty, in which case \(\hat{p}\) is finite. The feasible set is an intersection of halfspaces, which is a convex set known as a polyhedron. One property of LPs is that, if the optimal value is finite, an optimal solution, \(\hat{c}\), always occurs at a basic feasible point. A basic feasible point, also known as an extreme point, is one at which the residual, given by \(b_{ineq} - A_{ineq} \hat{c}\), is nonnegative and is zero for at least \(N\) rows, and the matrix \(A_{ineq}\) that contains the rows of \(A_{ineq}\) with zero residuals is of full rank.

Every LP has a dual formulation, given by

\[
\begin{align*}
\hat{d} &= \max \, -z^T b_{ineq} \\
\text{s.t.} \quad (1) \quad A_{ineq}^T \hat{z} + \lambda &= 0 \\
\quad (2) \quad z &\geq 0
\end{align*}
\]

where \(z \in \mathbb{R}^M\) is the dual variable. In bounded feasible LPs, \(d \leq p\). That is, the objective function of the dual problem is a lower bound for the primal problem. Excepting special cases, the bound is, in fact, tight, and equality occurs only for \(\hat{d} = \hat{p}\), corresponding to the optimal solutions of the dual and primal problems, respectively.

Active-set methods such as the well-known simplex method [19] are frequently used to solve LPs. These methods jump from one basic feasible point to another that has a lower objective value. Thus, active-set methods are descent methods, and each iteration has a lower objective value than the previous one. At each iteration, a dual solution is generated by computing \(\hat{z} = -A_{ineq}^T \lambda\), which is a subvector of \(z\). The remaining entries of \(z\) are taken to be zero. If \(\hat{z} \geq 0\), then the current iterate is optimal.

In solving LPs using active-set methods, an initial extreme point must be found. In general, this is non-trivial and requires the solution of a special Phase 1 LP. In the event that a point within the feasible polyhedron is known, an extreme point of the polyhedron can be found in a number of ways.
One is to move in the direction of \(-\lambda\) until a constraint is encountered. There is now at least one active constraint, and we move in the direction of the projection of \(-\lambda\) on the set of active constraints until more constraints are encountered. This is repeated until the set of active constraints has full rank, at which point an extreme point has been reached.

While there are many different approaches to solving LPs, in this paper we introduce a novel LP formulation for which an active-set solver is particularly efficient. This is described in Section III-C2.

### III. SAFE TENSION DISTRIBUTIONS

#### A. Definition of the Tension Distribution Problem

Completely restraining a cable-driven robot requires the existence of a strictly positive right null vector of the structure matrix, \(A \in \mathbb{R}^{n \times m}\) [9]. A necessary condition for the existence of such a null vector is that the number of cables exceeds the number of DOFs, i.e. \(m > n\). \(A\) must also have full rank. Furthermore, in order to deliver a desired wrench, \(w \in \mathbb{R}^n\), a set of tensions, \(f \in \mathbb{R}^m = [T_1 \cdots T_m]^T\), must meet the following equality:

\[
Af = w. \tag{3}
\]

As cables are unable to exert pushing forces, all cable tensions must be positive, and, in general, above some positive lower tension limit, \(T_{min}\). Additionally, the finite torque capabilities of system actuators impose an upper limit, \(T_{max}\), on cable tension. Furthermore, the goal of optimization is to minimize some objective function, \(g(f)\), subject to these constraints:

\[
\begin{align*}
\hat{f} &= \text{argmin}_{f} g(f) \\
\text{s.t.} \quad & (1) \quad Af = w \\
& (2) \quad T_{min} \begin{bmatrix} 1 \\ \vdots \\ 1 \\ m \end{bmatrix} \leq f \leq T_{max} \begin{bmatrix} 1 \\ \vdots \\ 1 \\ m \end{bmatrix}
\end{align*} \tag{4}
\]

Previous work has considered both \(g(f) = ||f||_1, [5]\) and \(g(f) = ||f||_2\) [15], [17]. In the former case, the optimization becomes a linear program. A significant shortcoming of tension distributions resulting from this formulation is that they are prone to discontinuities in which an infinitesimally small change in end-effector position or desired wrench can yield a finite change in tension [15], [16]. The resulting step change in desired tensions can excite high frequency modes and degrade robot stability. On the other hand, if the objective function \(||f||_2\) is used, then the optimization becomes a Quadratic Program (QP) with linear constraints. Significantly, the resulting tension distributions are not prone to discontinuous behavior.

Both the LP and QP formulations can be solved by standard convex optimization methods, because the objective functions are convex over the feasible space, which is also convex. As described in Section II, these methods require the solution of a special Phase 1 problem to find a feasible starting point. In a highly dynamic system, rapid computation of tension distributions is critical for robot performance. Thus, the cost associated with solving a separate Phase 1 problem can dramatically decrease the update rate of the tension control loop and thereby reduce the control bandwidth.

#### B. Previous Work in Safe Tension Distributions

A significant issue associated with previous LP and QP formulations is that the resulting tension distributions frequently contain cable tensions that are equal to \(T_{min}\) or \(T_{max}\), as is shown in Section IV. These limits represent extreme operating regions. A system that routinely operates at these points will frequently encounter near slack conditions in the case of operation at lower tension limits and excessively high torque requirements in the case of operation at upper tension limits. To ameliorate this, the authors of [16] recently presented a non-iterative algorithm that computes “safe” and continuous tension distributions for redundant platforms. The distributions are “safe” in that they lie at the Center of Gravity (CoG) of the manifold, \(P\), of feasible tension distributions that exert the desired wrench on the end-effector. Thus, they avoid tension limits, which reduces the likelihood of slack conditions.

In computing the CoG of \(P\), the authors of [16] first find all of its vertices. These vertices are intersection points of at least \(r\) of the \(2m\) tension constraints. All possible combinations of \(r\) out of \(2m\) constraints must be considered, and, for each combination, the corresponding intersection must be computed. If such an intersection complies with all tension inequalities, then it is a vertex of \(P\).

The deterministic, non-iterative nature of this algorithm recommends its use for systems in which the force distribution must be computed during each servo-loop under real-time constraints. However, the combinatorial nature of considering all combinations of \(r\) out of \(2m\) constraints indicates that this method becomes computationally intractable even for moderately complex platforms. In a general case, selecting \(r\) out of \(2m\) constraints results in

\[
\binom{2m}{r}
\]

possible combinations. However, the \(2m\) constraints contain \(m\) pairs of 2 parallel hyperplanes. Because parallel hyperplanes never intersect, each combination can contain at most one constraint from each pair. For example, if \(m=3, r=2\), and the constraints are arranged such that constraints \(2k - 1\) and \(2k, k = 1\ldots m\), are parallel, then the possible combinations are \{1,3\}, \{1,4\}, \{1,5\}, \{1,6\}, \{2,3\}, \{2,4\}, \{2,5\}, \{2,6\}, \{3,5\}, \{3,6\}, \{4,5\}, \{4,6\}\}, and combinations such as \{1,2\}, \{3,4\}, \{5,6\}, which contain parallel constraints, need not be considered. Thus, selecting \(r\) constraints becomes analogous to selecting \(r\) of the \(m\) pairs and considering every possible combination of these \(r\) pairs. There are

\[
\binom{m}{r}
\]

ways of selecting \(r\) pairs, and, for each such selection, there are \(2^r\) combinations. Thus, the total number of combinations
that must be considered in finding all the extreme points \( \mathcal{P} \) is given by:

\[
2^r \binom{m}{r}.
\]

(7)

Thus, merely computing the potential vertices of \( \mathcal{P} \) becomes prohibitive burdensome, even for moderately complex systems. For example, in the case of the 9-cable, 6-DOF Wiro 6.3 robot introduced in Section IV, Eqn. (7) yields 672 possible combinations.

C. Optimally Safe Tension Distributions

1) LP Problem Formulation: In this section, we present an LP method that computes “Optimally Safe” tension distributions by introducing a slack variable \( S \). The formulation of this problem is as follows:

\[
\begin{align*}
\hat{S} &= \max_{S} S \quad \text{s.t.} \\
(1) \quad Af &= w \\
(2) \quad (T_{\min} + S) \left[ \begin{array}{c} 1 \\ \vdots \\ 1 \\ 1 \end{array} \right] &\leq f \leq (T_{\max} - \alpha S) \left[ \begin{array}{c} 1 \\ \vdots \\ 1 \end{array} \right]
\end{align*}
\]

for some \( \alpha \geq 0 \), where the optimization variables are the tension vector \( f \) and the slack variable \( S \).

The second constraint in Eqn. (8) represents an \( m \)-dimensional hypercube wherein the upper limits are given by \( T_{\max} - \alpha S \) and the lower limits are given by \( T_{\min} + S \). The feasible set of tensions consists of the solutions \( \hat{f} \) to \( Af = w \) that lie within this hypercube. As \( S \) increases, the hypercube shrinks until only one solution of the equality remains within the hypercube. This point is \( \hat{f} \).

Similarly, we partition the tension vector into two vectors, \( f_1 \in \mathbb{R}^n \) and \( f_2 \in \mathbb{R}^r \), and partition the structure matrix \( A \) into two matrices, \( B \in \mathbb{R}^{n \times 2r} \) and \( H \in \mathbb{R}^{m \times r} \), where \( r = m - n \) is the degree of actuation redundancy presented by the robot:

\[
A = [B | H].
\]

(10)

Similarly, we partition the tension vector into two vectors, \( f_1 \in \mathbb{R}^n \) and \( f_2 \in \mathbb{R}^r \). Thus,

\[
AT = [B | H] \left[ \begin{array}{c} f_1 \\ f_2 \end{array} \right] = w.
\]

(11)

Solving for \( f_1 \) yields

\[
f_1 = B^{-1}(w - Hf_2).
\]

(12)

Depending on robot configuration, care must be exercised to ensure that \( B \) is of full rank. During operation in non-singular regions, the structure matrix, \( A \), is full-rank [9], and it is always possible to select \( n \) linearly independent columns by reordering. The method used to select linearly independent columns of \( A \) may vary depending on platform type. For some robots, a given set of columns will be linearly independent for all end-effector configurations within the workspace. This is true of NIMS-PL, which is introduced in Section IV-A. In this platform, two adjacent cables are never collinear for end-effectors within the workspace, so any such pair of cables can be selected in generating \( B \). If no such special property exists, a QR decomposition of \( A \) can be used to find linearly independent columns.

We see from Eqn. (12) that a tension distribution \( f \) can be completely determined by finding \( f_2 \in \mathbb{R}^r \). Using this property, the optimization in Eqn. (8) can be expressed in the following LP form:

\[
S = 0 \quad 0 < S < S^* \quad S = S^*
\]

Fig. 1. The constraints in Eqn. (8) shown for increasing \( S \). The \( m \)-dimensional hypercube represents the tension limits posed by the second constraint in Eqn. (8). As \( S \) increases, the hypercube shrinks until only one point satisfying \( Af = w \) remains within the hypercube. This point is \( \hat{f} \).

in the event of infeasibility, the Optimally Safe formulation performs gracefully and yields a meaningful result.

Besides providing optimally safe tension distributions, the slack variable, \( S \), enables explicit computation of a starting point that is not only feasible, but also close to the global optimum. The proximity of this starting point to the global optimum results in very fast convergence for active-set methods.

2) Efficient Computation Methods: We begin with Eqn. (3), and partition the structure matrix \( A \) into two matrices, \( B \in \mathbb{R}^{n \times 2r} \) and \( H \in \mathbb{R}^{m \times r} \), where \( r = m - n \) is the degree of actuation redundancy presented by the robot:

\[
A = [B | H].
\]

(10)
\[ \hat{y} = \arg\min_y d^T y \quad \text{s.t.} \quad A_{LP} y \leq b_{LP} \]  

where \[ y = \begin{bmatrix} f_2 \\ S \end{bmatrix}, \quad d = [0_{M_1}^T, -1]^T \]

\[ A_{LP} = \begin{bmatrix} -I_r & 1_r \\ B^{-1}H & 1_n \\ -B^{-1}H & \alpha I_r \\ -B^{-1}H & \alpha I_n \end{bmatrix} = [A_{LP_1} | b_{LP_2}], \text{ and} \]

\[ b_{LP} = \begin{bmatrix} -T_{\min} + B^{-1}w \\ T_{\max} - B^{-1}w \end{bmatrix}. \]

Because Eqn. (13) is solved during each iteration of the servo-loop, we expect the optimal solution found during the \( M - 1^{th} \) loop, \( y_{M-1} = [f_{2,M-1}^T, S_{M-1}]^T \), to be close to \( \hat{y}_M \), as only one servo-cycle has elapsed. We begin by checking if the corresponding set of active constraints, \( A_{LP_{M-1}} \), still yields an optimal result. As described in Section II, this is accomplished by computing a set of active constraints, \( A_{LP_{M-1}} \), and checking whether \( z \geq 0 \). If this is the case, then the optimal solution to Eqn. (13) has been found. If \( z \) contains negative elements, but the point \( y_{\text{new}} = A_{LP_{M-1}}^T b_{LP_{M-1}} \) still represents an extreme point for Eqn. (13), we begin the simplex method with this point as the starting iterate.

If \( y_{\text{new}} \) is not feasible, a feasible and near-optimal starting point can be explicitly found by relaxing the slack variable, \( S \), until feasibility is achieved. In doing so, we reduce \( S \) just enough to make the starting point feasible. The largest \( S \) for which \( f_{2,M-1} \) is feasible for the \( M^{th} \) servo-loop can be found explicitly as follows:

\[ S_{\text{max,M}} = \min ((b_{LP} - A_{LP} f_{2,M-1}) / a_{LP_2}), \]  

where we use the "/" notation to indicate component-wise division. In other words, \( S_{\text{max,M}} \) is the largest value of \( S \) for which

\[ b_{LP} - [A_{LP} | b_{LP_2}] \begin{bmatrix} f_{2,M-1} \\ S \end{bmatrix} \geq 0. \]

It should be noted that \( S_{\text{max,M}} \) can be negative. However, this still corresponds to a feasible starting point in Eqn. (8), which places no explicit constraint on the sign of \( S \). However, if the subsequent optimization does not generate a non-negative value of \( S \), then the problem is infeasible.

The resulting vector, \( y_{M} = [f_{2,M-1} | S_{\text{max,M}}]^T \), is now a feasible starting point from which an extreme point can be found as described in Section II. Once an extreme point has been found, the well-known simplex method is used to find the global optimum.

We expect agreement between the starting iterate and the optimal point to improve as the sampling rate increases. As shown in Section V, the proximity of the starting point to the optimal value results in fast convergence to the global optimum.

3) On the Continuity of Optimally Safe Tension Distributions: A simple example shows that Optimally Safe tensions can undergo step discontinuities in response to infinitesimally small changes in the shape of the set of feasible tensions. This is shown in Fig. 2. These discontinuities occur when the set of active constraints in the solution to Eqn. (13) becomes singular. However, we have not witnessed such behavior in any of our simulations or experiments on several different cable-driven platforms. Continuity for other platforms can be ensured by offline pre-checking or online supervision to stop the system in case of discontinuous tension distributions.

IV. STATIC TENSION DISTRIBUTION SIMULATIONS

In this section we present results showing the distribution of \( OS \) tensions for two simulated robotic systems. The first system considered is a four-cable-driven, 2-DOF system known as NIMS-PL [11]. The second is a nine-cable-driven, 6-DOF robot known as WiRo-6.3 [20].

A. NIMS-PL

NIMS-PL is a four-cable-driven, 2-DOF robot intended for actuated-sensing applications in aquatic environments[11]. A schematic diagram is shown in Fig. 3, and a kinematic representation is shown in Fig. 4. For this system, the \( i^{th} \) column of the structure matrix \( A_{NIMS} \in \mathbb{R}^{2 \times 4} \) is given by \( a_i = \frac{p_i - x}{|p_i - x|} \), the unit vector directed from the end-effector, \( x \), toward the \( i^{th} \) cable origin, \( p_i \), i.e.:

\[ A_{NIMS} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}. \]  

For our simulations, we consider a square horizontal workspace with dimension \( 1 \text{ m} \times 1 \text{ m} \) in which a 5 kg point-mass supported by a frictionless surface is actuated. The tension limits are given by \( T_{\min} = 10 \text{ N} \), and \( T_{\max} = 100 \text{ N} \).

B. WiRo-6.3

The second cable-driven robotic system considered in our simulations is WiRo-6.3 [20], a 6-DOF, nine-cable robot. A
schematic diagram is shown in Fig. 5. The authors of [20] describe several advantages of WiRo6.3, including a large feasible workspace and the existence of closed form solutions to inverse and forward kinematic routines. The structure matrix $A_{WiRO} \in \mathbb{R}^{6 \times 9}$ can be computed as in, for example, [16]. Following the definitions in Fig. 5, in our simulations, we take $r_A = r_B = h = 1.00$ m, and the end-effector is a thin 5 kg disc with radius .2 m. The tension limits are given by $T_{min} = 10$ N and $T_{max} = 100$ N.

### C. Static Tension Results

In order to demonstrate the effect of Optimally Safe tension distributions, we consider $\alpha = .5, 1, 2$.  

1) Static NIMS-PL Results: For NIMS-PL, we consider 1000 randomly selected static configurations, with 4000 corresponding tensions. The results are shown in histogram form in Fig. 6 compared with the $L^1$-minimal solution. It is readily apparent that the $L^1$ tension distributions frequently result in operating points on the lower tension limit. In fact, for static NIMS-PL configurations, two of the four cables are always on the lower tension limit. For the Optimally Safe distributions, it is evident that the tension limits are successfully avoided. Furthermore, varying $\alpha$ has the intended effect on the tension distributions, as the $\alpha = .5$ plot shows a higher concentration of cable tensions in the upper tension region, whereas the $\alpha = 2$ plot shows more cables operating under lower tensions. This is further evident in the average cable tensions, which are given in Table I.

D. Static WiRo-6.3 Results

For WiRo-6.3, we again consider 1000 randomly selected static configurations, with 9000 corresponding tensions. The results are shown in histogram form in Fig. 8 compared varying $\alpha$ on the resulting tension distributions, we consider $\alpha = .5, 1, 2$.

<table>
<thead>
<tr>
<th>Table I</th>
<th>Mean Cable Tensions for Static NIMS-PL Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha = .5$</td>
<td>$\alpha = 1$</td>
</tr>
<tr>
<td>Avg T (N)</td>
<td>57.4</td>
</tr>
</tbody>
</table>
with the $L^1$-minimal solution. Once again, the $L^1$ tension distributions frequently result in operating points on the lower tension limit. Of the 9000 cable tensions considered, 2996 lie on the lower tension boundary, while 4 lie on the upper limit. For Optimally Safe distributions, boundary conditions are once again successfully avoided, although the effect of varying $\alpha$ is more subtle than in the NIMS-PL case. While increasing $\alpha$ does noticeably shift tensions into lower regions, the effect is much less dramatic for this system. The average cable tensions, given in Table II, also indicate a smaller shift in mean tensions as compared to $L^1$-minimal solutions.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Avg T (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>.5</td>
<td>43.5</td>
</tr>
<tr>
<td>1</td>
<td>44.0</td>
</tr>
<tr>
<td>2</td>
<td>41.8</td>
</tr>
</tbody>
</table>

TABLE II
MEAN CABLE TENSIONS FOR STATIC WiRo-6.3 CONFIGURATIONS.

It should be noted that these simulations consider end-effector dynamics but ignore other dynamic effects, such as those introduced by cables and actuator systems. Including these effects would influence the desired output wrench, $w$, and may slightly increase the computational burden required in the optimization. However, the presented results indicate that the computational burden associated with computing Optimally Safe tension distributions is very small and that

V. DYNAMIC TENSION DISTRIBUTION SIMULATIONS

While the results presented in Section IV show the effectiveness of $OS$ tension distributions in avoiding tension limits, the configurations considered are static, and therefore nothing can be said about computational efficiency. In order to enable such analysis, we simulated the execution of highly dynamic trajectories for both NIMS-PL and WiRo-6.3 for $\alpha = 1$ and for servo frequencies ranging from 5 Hz to 50 Hz. We show that, even for the complex WiRo-6.3 system with a servo-rate of 50 Hz, which exceeds the bandwidth of typical tension control systems, the worst-case computational burden of computing optimally safe tension distributions consumes only a small fraction of the processing power of commercially available embedded processors.
the optimization would still be manageable despite moderate increases in complexity.

### A. NIMS-PL Dynamic Results

In order to evaluate the computational efficiency of our LP solver for NIMS-PL, we simulated the execution of a highly dynamic trajectory on the same NIMS-PL configuration considered in the previous section. The trajectory considered is a triangle with start and end-points given in Table III. In this simulation, the maximum velocity is 2 m/s and the maximum acceleration is 10 m/s². Furthermore, an s-curve trajectory is used, wherein the acceleration profile is continuous and rises to its maximum value in .02 s. The duration of the entire triangular trajectory, shown in Fig. 10, is 1.52 s.

The mean number of active-set iterations required for convergence is shown against servo frequency in Fig. 11. It is apparent that the number of iterations per servo-loop decreases significantly as the servo update rate increases. This is in accordance with expectations, as the starting point for each iteration, which is given by the optimal value from the previous iteration, becomes closer to the new optimal value as the sample rate increases.

Because robotic systems are subject to real-time constraints, it is critical that the tension optimization be entirely completed during each servo-loop. Thus, worst-case performance must not be exceedingly poor. In this simulation, the largest number of iterations required for convergence was 5. The greatest computational burden required of the CPU was approximately 580 floating point operations (FLOPs) at an update rate of 48

<table>
<thead>
<tr>
<th>Path Number</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Point [x, y(m)]</td>
<td>[.5, .2]</td>
<td>[.5, .3]</td>
<td>[.8, .7]</td>
</tr>
<tr>
<td>End Point [x, y(m)]</td>
<td>[.3, .5]</td>
<td>[.8, .7]</td>
<td>[.5, .2]</td>
</tr>
</tbody>
</table>

### TABLE III

<table>
<thead>
<tr>
<th>Path Number</th>
<th>Start Point [x, y(m)]</th>
<th>End Point [x, y(m)]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[.5, .2]</td>
<td>[.3, .5]</td>
</tr>
<tr>
<td>2</td>
<td>[.5, .3]</td>
<td>[.8, .7]</td>
</tr>
<tr>
<td>3</td>
<td>[.8, .7]</td>
<td>[.5, .2]</td>
</tr>
</tbody>
</table>
Active-set Iterations per Servo Loop for WiRo6.3

Fig. 12. Active-set iterations per servo-loop shown against servo frequency.

Hz, which corresponds to roughly \(27 \frac{kFLOPs}{s}\). This represents a very small fraction of the computational power of modern embedded processors. The average computational burden at 50 Hz was 9.7 \(kFLOPs\).

B. WiRo-6.3 Dynamic Results

In order to evaluate the computational efficiency of computing OS solutions for a more complex system, we simulated the execution of a highly dynamic trajectory on the same WiRo-6.3 configuration considered in the previous section. We define the position vector \(x = [x y z \theta_x \theta_y \theta_z]^T\), where the units of the translational DOFs are meters and the units of the rotational DOFs are radians. The trajectory starts and ends at \(x = [0.0 0.2 0.0 .35]^T\) and consists of the following 5 parts:

1) Move in a straight line to \(x = [0.0 .8 0.0 −.35]^T\).
2) Move in a straight line to \(x = [0.0 .6 0.0 0]^T\).
3) Move in a straight line to \(x = [.35 0.6 0.0 0]^T\).
4) Execute a circle with radius .35 m about the z-axis.
5) Return to the starting point.

For all trajectory segments, the maximum velocity is 1.5 m/s and the maximum acceleration is 10 m/s². Again, an s-curve trajectory is used, wherein the acceleration profile is continuous and rises to its maximum value in \(t_a = .05\) s. The duration of the entire trajectory is 4.02 s.

The mean number of active-set iterations required for convergence is shown against servo frequency in Fig. 12. Again, the number of iterations per servo-loop decreases significantly as the servo update rate increases.

In this simulation, the highest computational burden during a single servo-loop was 8 active-set iterations with a servo rate of 48 Hz, corresponding to approximately 110 \(kFLOPs\). The average computational burden at 50 Hz was 24 \(kFLOPs\). Again, this represents a small fraction of the computational power of commercially available embedded processors. Thus, we see that our method of computing Optimally Safe tensions is sufficiently fast for a 6-DOF, 9-cable robot executing a highly dynamic trajectory, even with an update rate that far exceeds the bandwidth of typical tension control systems.

In order to compare our algorithm to that of the non-iterative method presented in [16], we consider again that, for a 6-DOF, 9-cable robot, Eqn. (7) yields that 672 intersections of hyperplanes must be computed. This requires solving 672 sets of linear equations in \(\mathbb{R}^3\). If Gaussian Elimination is used, computing one of these solutions requires 28 FLOPs, and evaluating whether it is feasible requires an additional 90 FLOPs. Thus, approximately 80 kFLOPs are required to find the vertices of the feasible polyhedron \(\mathcal{P}\). If the servo update rate is 50 Hz, then \(4\frac{kFLOPs}{s}\) are required merely to compute the possible extreme points. Thus, we see that the computational burden presented in finding the extreme points of \(\mathcal{P}\) in the non-iterative method of [16] is at least 36 times greater than the worst-case complexity of our algorithm and at least 165 times greater than its average complexity.

VI. EXPERIMENTAL RESULTS

In order to evaluate the performance of OS tension distributions on an actual physical system, we implemented an OS tension solver on an actual NIMS-PL system, which was configured in a square with side length 1.264 m. For these trials, \(T_{min} = 20\) N, and \(T_{max} = 100\) N. Cable tensions were monitored while a circular trajectory centered at .632 m with radius .35 m was executed at a speed of .55 m/s. We consider \(\alpha = .5, 1, 2\) as well as the \(L^1\)-minimal tension distribution. For these trials, tension distributions were computed at 17.75 Hz on an 80 MHz DSP56300 embedded chip. It should be noted that this CPU can operate at frequencies up to 240 MHz, thereby increasing computational bandwidth and the rate at which tension distributions are computed. Furthermore, modern DSPs are available that vastly outperform our current CPU, so very high update rates are readily achievable.

No tension discontinuities were observed during the execution of the trajectories. Histograms of the resulting tensions are shown in Fig. 13. We do not expect exact agreement with Fig. 6 because we now consider only points along the experimental trajectory, whereas, in the previous case, we considered a large number of randomly selected points. We observe that increasing \(\alpha\) does noticeably decrease tensions. This is further evident in the mean cable tensions given in Table IV.

It is apparent from Fig. 13 that all tension distributions experience occasional violations of constraints caused by dynamic effects and imperfect tension control. This underscores the importance of maximizing the distance between cable tensions and their imposed limits. For the OS distributions, violations of tension limits are rare, whereas, for the \(L^1\)-minimal distributions, they occur quite frequently. The CDF of the distance of cable tensions from the closest boundary condition is shown in Fig. 14. We see that the use of OS cable tensions dramatically reduces the occurrence of near-slab operating conditions and violation of tension limits.

<table>
<thead>
<tr>
<th>(\alpha)</th>
<th>Avg T (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>55.2</td>
</tr>
<tr>
<td>2</td>
<td>48.2</td>
</tr>
<tr>
<td>3</td>
<td>27.4</td>
</tr>
</tbody>
</table>

TABLE IV
MEAN CABLE TENSIONS FOR EXPERIMENTAL EXECUTION OF A CIRCULAR TRAJECTORY ON NIMS-PL.
In this paper, we have presented a novel LP formulation that generates “Optimally Safe” tension distributions for cable-driven robotic systems. We have shown that introduction of a slack variable enables rapid generation of a feasible starting point from the solution of the previous servo-loop. Furthermore, the proximity of this starting point to the global optimum results in rapid convergence to the optimal operating point and, thereby, a highly efficient computation process. In order to demonstrate the extensibility of our algorithm to platforms with varying degrees of freedom and actuation redundancy, we have considered both a four-cable, 2-DOF system, and a nine-cable, 6-DOF system. For both of these platforms, we have presented static tension distribution simulations that illustrate the ability of our algorithm to avoid near-slack operating conditions. Furthermore, these simulations demonstrate the ability of our LP formulation to favor high or low tension configurations, depending on user preference. We have also simulated the execution of highly dynamic trajectories on both robotic platforms to demonstrate the high efficiency of computing Optimally Safe tension distributions. We have shown that, for a complex, nine-cable, 6-DOF robot executing a highly dynamic trajectory, our algorithm is sufficiently fast for operation at servo-update rates far beyond the control bandwidth of typical tension control systems. In order to evaluate our algorithms on a real robotic system, we implemented them on NIMS-PL, a 4-cable 2-DOF robot, and executed a circular trajectory. The resulting tension distributions avoided near-slack operating conditions and demonstrated continuous behavior.

VII. CONCLUSION

In this paper, we have presented a novel LP formulation that generates “Optimally Safe” tension distributions for cable-driven robotic systems. We have shown that introduction of a slack variable enables rapid generation of a feasible starting point from the solution of the previous servo-loop. Furthermore, the proximity of this starting point to the global optimum results in rapid convergence to the optimal operating point and, thereby, a highly efficient computation process. In order to demonstrate the extensibility of our algorithm to platforms with varying degrees of freedom and actuation redundancy, we have considered both a four-cable, 2-DOF system, and a nine-cable, 6-DOF system. For both of these platforms, we have presented static tension distribution simulations that illustrate the ability of our algorithm to avoid near-slack operating conditions. Furthermore, these simulations demonstrate the ability of our LP formulation to favor high or low tension configurations, depending on user preference. We have also simulated the execution of highly dynamic trajectories on both robotic platforms to demonstrate the high efficiency of computing Optimally Safe tension distributions. We have shown that, for a complex, nine-cable, 6-DOF robot executing a highly dynamic trajectory, our algorithm is sufficiently fast for operation at servo-update rates far beyond the control bandwidth of typical tension control systems. In order to evaluate our algorithms on a real robotic system, we implemented them on NIMS-PL, a 4-cable 2-DOF robot, and executed a circular trajectory. The resulting tension distributions avoided near-slack operating conditions and demonstrated continuous behavior.

REFERENCES


