6

Wire Robots Part I
Kinematics, Analysis & Design

Tobias Bruckmann, Lars Mikelsons, Thorsten Brandt,
Manfred Hiller and Dieter Schramm
University Duisburg-Essen (Chair for Mechatronics)
Germany

1. Introduction

One drawback of classical parallel robots is their limited workspace, mainly due to the limitation of the stroke of linear actuators. Parallel wire robots (also known as Tendon-based Steward platforms or cable robots) face this problem through substitution of the actuators by wires (or tendons, cables, . . .). Tendon-based Steward platforms have been proposed in (Landsberger & Sheridan, 1985). Although these robots share the basic concepts of classical parallel robots, there are some major differences:

Fig. 1(a) Conventional parallel manipulator

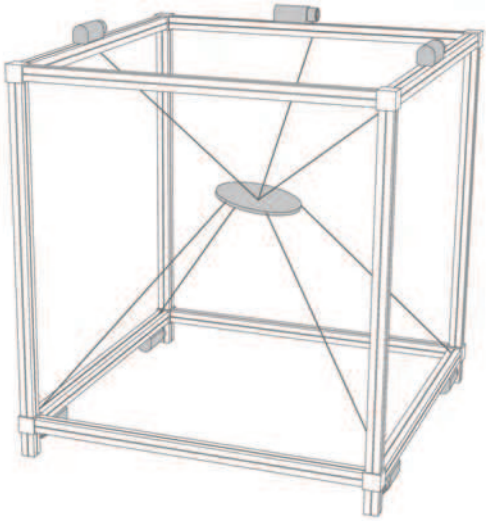

Fig. 1(b) Parallel Wire Robot

- The flexibility of wires allows large changes in the length of the kinematic chain, for example by coiling the tendons onto a drum. This allows to overcome the purely geometric workspace limitation factor of classical robots.

Source: Parallel Manipulators, New Developments, Book edited by: Jee-Hwan Ryu, ISBN 978-3-902613-20-2, pp. 498, April 2008,
I-Tech Education and Publishing, Vienna, Austria

- Wires can be coiled by very fast drums while the moving mass of the robot is extremely low, which allows the robot to reach very high end effector speeds and accelerations.
- Wires are modeled as unilateral constraints, i.e. wires can only transmit pulling forces.
- The number of wires m can be increased to modify the workspace, to carry higher loads or to increase safety due to redundancy. Thus, having an end effector (in the following called platform) with n degrees-of-freedom (d.o.f.), more than n parallel links are used to connect the platform to the base frame.

This contribution is organized as follows: In section 2 the classification of wire robots, based on several approaches is presented. Furthermore, the kinematic calculations for wire robots are described which is followed by the description of the force equilibrium in section 3. Based on the force equilibrium, methods for workspace analysis and robot design are proposed in section 4 and 5, respectively. This contribution is extended in Part 2 (Bruckmann et al., 2008a) by the description of dynamics, control methods and application examples. Within this and the next chapter, the following abbreviations are used:

$B_{\boldsymbol{r}}$	vector $\boldsymbol{r}$ denoted in coordinate system $\mathcal{B}$
$\boldsymbol{r}_i$	i-th component of vector $\boldsymbol{r}$
$\boldsymbol{A}$	matrix $\boldsymbol{A}$
${}^{\mathrm{B}}\boldsymbol{R}_{\mathrm{P}}$	transformation matrix from coordinate system $\mathcal{P}$ to $\mathcal{B}$
$\boldsymbol{A}^{\mathrm{T}}$	shorthand for the transpose of $\boldsymbol{A}$
$\boldsymbol{A}^{-\mathrm{T}}$	shorthand for $(\boldsymbol{A}^{-1})^{\mathrm{T}}$
$\dot{x}$	derivation of x with respect to time, $\dot{x} = \frac{dx}{dt}$

2. Kinematics

2.1 Classification

For wire robots, different classifications based on the difference between the number of wires m and the number d.o.f. n have been proposed. Further on, this difference is called the redundancy $r = m - n$. According to (Ming & Higuchi, 1994) wire robots can be categorized based on the redundancy as follows:

- CRPM (Completely Restrained Parallel Manipulator): The pose of the robot is completely determined by the unilateral kinematic constraints defined by the tensed wires. For a CRPM at least $m = n + 1$ wires are needed.
- IRPM (Incompletely Restrained Parallel Manipulator): In addition to the unilateral constraints induced by the tensed wires at least one dynamical equation is required to describe the pose of the end effector.

In (Verhoeven, 2004) the category of CRPMs is further divided into two categories. The class of the CRPMs is restricted to robots with $m = n+1$ wires. Wire robots with $m > n + 1$ are called RRPMs (Redundantly Restrained Parallel Manipulator). Note that within this definition CRPM and RRPM robots can convert into IRPM robots if they are used at poses where external wrenches (inertia and generalized forces and torques applied onto the platform) are necessary to find completely positive wire forces. Therefore in (Verhoeven, 2004) another classification is proposed based on the number of controlled d.o.f. which is listed below.

- 1T: linear motion of a point
- 2T: planar motion of a point
- 1R2T: planar motion of a body
- 3T: spatial motion of a point
- 2R3T: spatial motion of a beam
- 3R3T: spatial motion of a body

Fig. 2(a) class 1T

Fig. 2(b) class 2T

Fig. 2(c) class 1R2T

Fig. 2(d) class 3T

Fig. 2(e) class 2R3T

Fig. 2(f) class 3R3T

Here T stands for translational and R for rotational d.o.f.. It is notable that this definition is complete and covers all wire robots. The classification of (Fang, 2005) is similar to Verhoeven's approach. Here, three classes are defined as:

- IKRM (Incompletely Kinematic Restrained Manipulators), where $m < n$
- CKRM (Completely Kinematic Restrained Manipulators), where $m = n$
- RAMP (Redundantly Actuated Manipulators), where $m \geq n + 1$

This chapter as well as the next one focuses on CRPM and RRPM robots. For IRPM see e.g. (Maier (2004)).

2.2 Inverse kinematics

Inverse kinematics refers to the problem of calculating the joint variables for a given end-effector pose. For the class of robots under consideration those are the lengths of the wires, comparable to the strokes of linear actuators. Therefore, the kinematical description of a wire robot resembles the kinematic structure of a Stewart-Gough platform, presuming the wires are always tensed and can thus be treated as line segments representing bilateral constraints. Modeling a wire robot as a platform, which is connected to m points on the base

by m bilateral constraints, it is reasonable to denote the platform pose $\boldsymbol{x} = [\ {}^{B}\boldsymbol{r}^{T}\ \ \varphi\ \vartheta\ \psi\ ]$ and the base points ${}^{B}\boldsymbol{b}_i,\ _i = 1 \leq i \leq m$, referenced in the inertial frame $\mathcal{B}$. Besides that, the platform connection points $\boldsymbol{p}^i$ are referenced in the platform-fixed coordinate frame $\mathcal{P}$.The orientation of the platform in the base frame is represented by the rotation matrix ${}^{B}\boldsymbol{R}_P$. Note that throughout this chapter roll-pitch-yaw angles are used. Assuming the wires are led by point-shaped guidances (e.g. small ceramic eyes) from the winches to the platform, the base vectors ${}^{B}\boldsymbol{b}_i$ are constant. Now the vector chain pictured in fig. 3 delivers

$$ {}^{B}\boldsymbol{l}_i = {}^{B}\boldsymbol{b}_i - \underbrace{{}^{B}\boldsymbol{r} - {}^{B}\boldsymbol{R}_P\,{}^{P}\boldsymbol{p}_i}_{{}^{B}\boldsymbol{p}_i}, \quad 1 \leq i \leq m \tag{1} $$

immediately. Hence, the length of the i_{th} wire can be calculated by

$$ l_i = \left\| {}^{B}\boldsymbol{b}_i - {}^{B}\boldsymbol{p}_i \right\|_2, \quad 1 \leq i \leq m \tag{2} $$

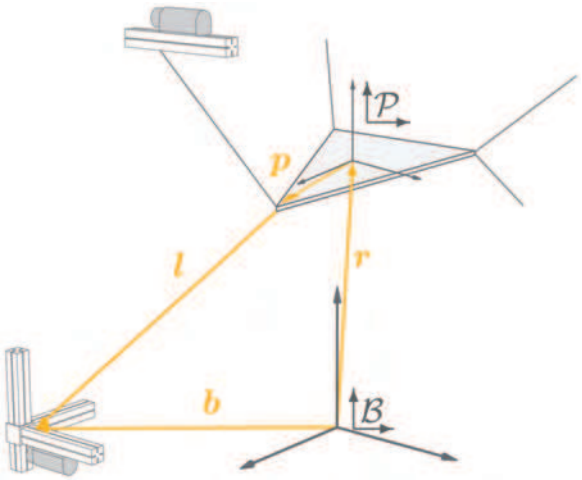

Fig. 3: Kinematics of a wire robot

Based on the relatively simple inverse kinematics, a position control in joint space can be designed for a wire robot which already may deliver satisfying results. Note, this simple calculation only holds for the described simple guidance. While it may be sufficient for simple prototypes, it suffers from a very high wear and abrasion. Thus it is not feasible for practical applications. An alternative concept is the roller-based guidance which is e.g. widely used in theatre and stage technology, see fig. 4. As a drawback, the kinematical description becomes more difficult due to the pose dependent exit points points ${}^{B}\boldsymbol{s}_i$ of the wires. The roller with radius ρ is mounted onto a pivot arm. To calculate the exit points ${}^{B}\boldsymbol{s}_i$, two angles have to be known: the pivoting angle θ_i and the wrap angle α_i(see fig. 4). The pivoting angle can be calculated using a projection onto the plane D whose normal vector is the rotation axis (without loss of generality the z-axis of the inertial frame) of the pivoting angle as:

$$\tan\Theta_i = \frac{{}^B\boldsymbol{p}_{i,y} - {}^B\boldsymbol{b}_{i,y}}{{}^B\boldsymbol{p}_{i,x} - {}^B\boldsymbol{b}_{i,x}}, \quad 1 \le i \le m. \tag{3}$$

Here ${}^B\boldsymbol{b}_i$ denotes the vector to the point, at which the wire enters the roller. With this knowledge the vector ${}^B\boldsymbol{m}_i$ to the midpoint of the $i-$th roller can be constructed

$${}^B\boldsymbol{m}_i = {}^B\boldsymbol{b}_i + \boldsymbol{R}_{z_B,\Theta_i} \cdot \rho \cdot {}^B\boldsymbol{e}_x \tag{4}$$

Where R_{z^B,Θ_i} is a rotation matrix for angle Θ_i around the z-axis of the inertial frame. Note that without loss of generality the projection of ${}^B\boldsymbol{b}_i - {}^B\boldsymbol{m}_i$ onto the $x-z$-plane of $\mathcal{B}$ is parallel to the x-axis in the reference orientation of the roller. Then the wrap angle α_i is according to fig. 4 given by

$$\alpha_i = \pi - (\alpha_{i,1} + \alpha_{i,2}), \tag{5}$$

where

$$\sin\alpha_{i,1} = \frac{{}^Bp_{i,z} - {}^Bm_{i,z}}{\|{}^B\boldsymbol{p}_i - {}^B\boldsymbol{m}_i\|_2}, \quad \cos\alpha_{i,2} = \frac{\rho}{\|{}^B\boldsymbol{p}_i - {}^B\boldsymbol{m}_i\|_2}, \quad 1 \le i \le m. \tag{6}$$

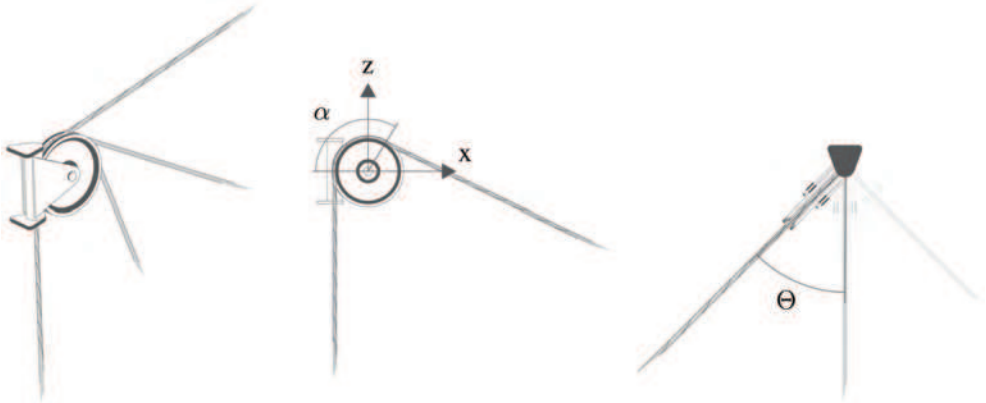

Fig. 4: Roller-based guidance

In a projection onto the plane D, $\alpha_{i,1}$ describes the angle between the x-y-plane of the inertial frame and the vector $\boldsymbol{q}$ from ${}^B\boldsymbol{m}_i$ to the platform connection point ${}^B\boldsymbol{p}_i$. The angle $\alpha_{i,2}$ is the angle between the vector from ${}^B\boldsymbol{m}_i$ to the exit point and vector $\boldsymbol{q}$. Furthermore the exit point ${}^B\boldsymbol{s}_i$ of the i-th wire can be found as

$${}^B\boldsymbol{s}_i = \begin{bmatrix} {}^Bm_{i,x} + \rho\cos\alpha_1 \cdot \cos\Theta_1 \\ {}^Bm_{i,y} + \rho\cos\alpha_1 \cdot \sin\Theta_1 \\ {}^Bm_{i,z} + \rho \cdot \sin\alpha_1 \end{bmatrix}, \quad 1 \le i \le m. \tag{7}$$

Therefore the wire length can be calculated by

$$l_i = \rho \cdot (\pi - \alpha_i) + \left\| {}^B\boldsymbol{s}_i - {}^B\boldsymbol{p}_i \right\|_2. \tag{8}$$

Analog to the Stewart-Gough platform, the forward kinematics is much more complicated, in particular for the case of roller guidances.

2.3 Forward kinematics

In opposite to the inverse kinematics, where the equations are decoupled and therefore straight forward to solve, the forward kinematics problem is more involved. In general the forward kinematics are not analytically solveable. However, in some cases a geometrical approach allows a closed solution. To be more precise, a setup with three base points connected to one platform connection points leads to the task of finding the intersection points of three spheres where the radii of the spheres represent the measured lengths of the wires and the centers of the spheres are the base points $\boldsymbol{b}_i$. Hence, the spheres represent possible positions of the endpoints of the wires. Note, that a point-shaped wire guidance is presumed. More details can be found in (Williams et al., 2004). Nevertheless, in general no analytical solution is at hand. Thus, numerical approaches have to be employed to find the solution, which is disadvantageous in terms of computation time, especially when the computation has to be done in real-time. The forward kinematics problem is generally described by m nonlinear equations in n unknown variables.

$$\rho \cdot (\pi - \alpha_i) + \left\| {}^B\boldsymbol{s}_i - {}^B\boldsymbol{p}_i \right\|_2 - l_i = 0 \tag{9}$$

If point-shaped wire guidances are used, ρ becomes zero. In case of $m = n$, (Fang, 2005) proposes to apply a Newton-Raphson solver while for CRPMs and RRPMs, one has to consider an overdetermined system. A standard approach to this class of problems is the use of a least square method which minimizes the influence of measurement errors. However, the Newton-Raphson approach can also be used for the case of $m \geq n + 1$ as shown in the following, denoting the vector of wire lengths $\boldsymbol{l} = [\, l_1 \ldots l_m]^T$ (Fang, 2005):

$$\dot{\boldsymbol{l}}(t) = \boldsymbol{J}_{inv}(\boldsymbol{l}(t)) \cdot \dot{\boldsymbol{x}}(t) \quad \boldsymbol{J}_{inv}(\boldsymbol{l}(t)) = \frac{\partial \boldsymbol{l}}{\partial \boldsymbol{x}} \in \mathbb{R}^{m \times n}. \tag{10}$$

Since in kinematics positive wire tensions are assumed, the wires are modeled as bilateral constraints, already six constraints fix the platform, i.e. r rows of the inverse Jacobian $\boldsymbol{J}_{inv}$ can be removed, resulting in $\tilde{\boldsymbol{J}}_{inv}$. Assuming $\boldsymbol{J}_{inv}$ having full rank, in case of a CRPM, any arbitrary choice of a row leads to full ranked $\tilde{\boldsymbol{J}}_{inv}$. In case of a RRPM, this does not hold in general. Thus, one has to test for a feasible choice of r rows which allows to calculate the reduced Jacobian of the forward kinematics $\tilde{\boldsymbol{J}}_{forw} = \tilde{\boldsymbol{J}}_{inv}^{-1}$. Without loss of generality, let n wire lengths $l_1, \ldots, l_n$ be chosen. Thus,

$$\dot{\boldsymbol{x}} = \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t)) \underbrace{\begin{bmatrix} l_1 & l_2 & \ldots & l_n \end{bmatrix}^T}_{\boldsymbol{l}_{red}} \tag{11}$$

holds. The position at the time t_1 can be calculated by forward integration in time

$$\boldsymbol{x}(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} \dot{\boldsymbol{x}}(\tau)\, d\tau \;\Leftrightarrow \boldsymbol{x}(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(\tau))\dot{\boldsymbol{l}}_{red}(\tau)\, d\tau. \tag{12}$$

Taylor expansion of the second term around t_0 delivers

$$\int_{t_0}^{t} \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(\tau))\dot{\boldsymbol{l}}_{red}(\tau)\, d\tau = \sum_{k=0}^{\infty} \frac{1}{k!}\left(\int_{t_0}^{t} \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(\tau))\dot{\boldsymbol{l}}_{red}(\tau)\, d\tau\right)^{(k)}_{|t=t_0} \cdot (t - t_0) \tag{13}$$

Neglecting terms of second order and higher leads to

$$\int_{t_0}^{t} \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(\tau))\dot{\boldsymbol{l}}_{red}(\tau)\, d\tau \approx \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t_0))\dot{\boldsymbol{l}}_{red}(t_0) \cdot (t - t_0). \tag{14}$$

Approximating the differential quotient by the difference quotient gives

$$\tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t_0))\dot{\boldsymbol{l}}_{red}(t_0) \cdot (t - t_0) \approx \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t_0))\Delta\boldsymbol{l}_{red}, \tag{15}$$

where

$$\Delta\boldsymbol{l}_{red} = \boldsymbol{l}_{red}(t) - \boldsymbol{l}_{red} \cdot (t_0) \tag{16}$$

Using these simplified expressions, the platform pose $\boldsymbol{x}$ can be approximated by $\boldsymbol{x}_{app}$:

$$\boldsymbol{x}_{app}(t) = \boldsymbol{x}(t_0) + \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t_0))\Delta\boldsymbol{l}_{red} \tag{17}$$

For $\boldsymbol{x}_{app}(t)$, the inverse kinematics and the pose estimation error $\Delta \boldsymbol{x}(t)$ can be calculated, delivering the wire lengths $\boldsymbol{l}_{app}$ for the approximated pose. Now the difference $\Delta \boldsymbol{l}(t)$ between the measured and approximated wire lengths can be calculated, giving a measure for the pose error:

$$\Delta\boldsymbol{l}(t) = \boldsymbol{l}_{app}(t) - \boldsymbol{l}(t) \quad \wedge \quad \Delta\boldsymbol{x}(t) = \boldsymbol{x}_{app}(t) - \boldsymbol{x}(t) \;\Leftrightarrow\; \boldsymbol{x}(t) = \boldsymbol{x}_{app}(t) - \Delta\boldsymbol{x}(t) \tag{18}$$

Once again using the approximations

$$\Delta\boldsymbol{x} = \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}_{app}(t))\Delta\boldsymbol{l}_{red} \quad \text{for} \quad \dot{\boldsymbol{x}} = \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}(t))\dot{\boldsymbol{l}}_{red} \tag{19}$$

it follows

$$\boldsymbol{x}(t) \approx \boldsymbol{x}_{app}(t) - \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}_{app}(t))\Delta\boldsymbol{l}_{red} = \boldsymbol{x}_{app}(t) - \tilde{\boldsymbol{J}}_{forw}(\boldsymbol{x}_{app}(t))(\boldsymbol{l}_{app}(t) - \boldsymbol{l}(t)) \tag{20}$$

where $\boldsymbol{l}_{app}(t)$ is calculated by the inverse kinematics for $\boldsymbol{x}_{app}(t)$. Noteworthy, this approach works only for small pose displacements. When displacements become larger, an iteration can improve the precision of the calculated pose by using $\boldsymbol{x}(t)$ as the estimate $\boldsymbol{x}_{app}(t)$ for the next step (Merlet, 2000). In (Williams et al., 2004), the authors show an iterative algorithm for a roller-based wire guidance neglecting the pivoting angle.

3. Force equilibirum

The end effector of wire robots is guided along desired trajectories by tensed wires. This design is superior to classical parallel kinematic designs in terms of workspace size - due to the practically unlimited actuator stroke creating potentially large workspaces - and mechanical simplicity. On the other hand and caused by the unilateral constraints of the wires, the workspace of wire robots is primarily limited by the forces which may be exerted by the wires. The unilateral constraints necessitate positive forces. Practically, long wires will sag at low tensions which makes kinematical computations more complicated and may lead to vibration problems. Hence, the minimum allowed forces in the wires should never fall below a predefined positive value. Against, high forces lead to increased wear and elastic deformations. Therefore the working load of wires is bounded between predefined values $\boldsymbol{f}_{\min} \in \mathbb{R}^m$ and $\boldsymbol{f}_{\max} \in \mathbb{R}^m$ and wire forces must remain between these limits. Thus, a description of the force distribution in the wires for given end effector poses and wrenches is needed. Here a convenient description of the force distribution will be presented, while in (Bruckmann et al., 2008a) three different methods for the force calculation are shown. The force and torque equilibrium at the end effector gives according to figure 5

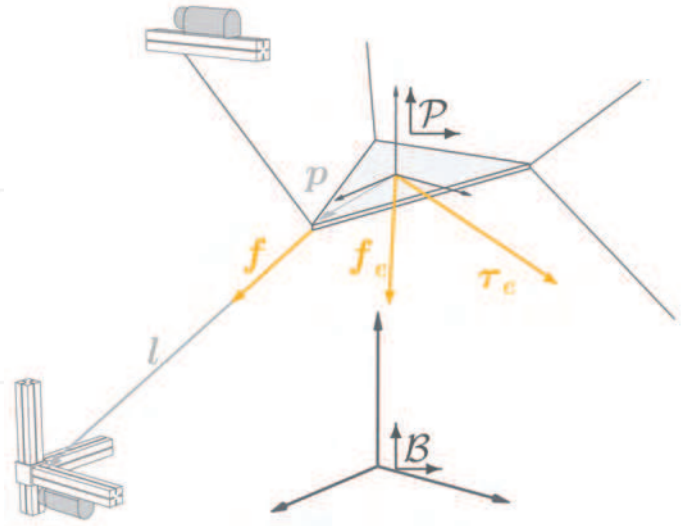

Fig. 5: Forces for a wire robot

$$\sum_{i=1}^{m} \boldsymbol{f}_i + \boldsymbol{f}_p = 0, \text{ and } \sum_{i=1}^{m} \boldsymbol{p}_i \times \boldsymbol{f}_i + \boldsymbol{\tau}_p = 0 \tag{21}$$

The force vectors $\boldsymbol{f}_i$ can be written as

$$\boldsymbol{f}_i = f_i \cdot \frac{\boldsymbol{l}_i}{\|\boldsymbol{l}_i\|_2} = f_i \cdot \boldsymbol{\nu}_i, \quad (1 \leq i \leq m) \tag{22}$$

since the forces act along the wires. Hence, the force and torque equilibrium can be written in matrix form

$$\begin{bmatrix} \boldsymbol{\nu}_1 & \cdots & \boldsymbol{\nu}_m \\ \boldsymbol{p}_1 \times \boldsymbol{\nu}_1 & \cdots & \boldsymbol{p}_m \times \boldsymbol{\nu}_m \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix} + \begin{bmatrix} \boldsymbol{f}_p \\ \boldsymbol{\tau}_p \end{bmatrix} = 0 \tag{23}$$

with

$$\boldsymbol{f}_{\max} \geq \boldsymbol{f} \geq \boldsymbol{f}_{\min} > 0 \tag{24}$$

or in a more compact form as

$$\boldsymbol{A}^{\mathrm{T}} \boldsymbol{f} + \boldsymbol{w} = \boldsymbol{0} \tag{25}$$

$$\boldsymbol{f}_{\max} \geq \boldsymbol{f} \geq \boldsymbol{f}_{\min} > 0. \tag{26}$$

In the following the matrix $\boldsymbol{A}^{\mathrm{T}}$ is called structure matrix. It is noteworthy that the structure matrix can also be derived as the transpose of the Jacobian of the inverse kinematics, but generally, it is easier to construct it based on the force approach (Verhoeven, 2004).

4. Workspace analysis

In practical applications knowledge of the workspace of the robot under consideration is essential. In contrast to conventional parallel manipulators using rigid links, the workspace of a wire robot is not mainly limited by the actuator strokes, since the length of the wires is not the main limiting factor, just restricted by the drum capacity. In fact, the workspace of a wire robot is limited anyway by the wire force limits $\boldsymbol{f}_{\min}$ and $\boldsymbol{f}_{\max}$. A pose $\boldsymbol{r}$ is said to be part of the workspace if a wire force distribution $\boldsymbol{f}$ exists, such that $\boldsymbol{f}_{\min} \leq \boldsymbol{f} \leq \boldsymbol{f}_{\max}$ holds. Additionally further criteria, like stiffness or wire collisions, can be taken into account. Different methods to calculate the workspace of a wire robot are available. Here discrete methods as well as a continuous method using interval analysis are discussed. Further methods exist as for example presented in (Bosscher & Ebert-Uphoff, 2004), where the workspace boundaries are computed.

4.1 Discrete analysis

In order to perform a discrete workspace analysis at first an assumed superset of the workspace is discretized. Mostly an equidistant discretization is desired. This leads to a set of points, which is then tested with respect to the chosen workspace requirements. This is a widely used approach, but nevertheless, some considerations should be taken into account:

- The calculation of the workspace conditions for the grid points generally requires the verification of a valid wire force distribution. Since it is sufficient to identify any valid distribution, fast calculation methods as presented in section (Bruckmann et al., 2008a) can be employed.
- For some parallel kinematic mechanisms, typically symmetrical configurations are singular, leading to uncontrollable d.o.f. of the end effector. Thus, it is recommended to explicitly test at symmetrical poses of the end effector.
- Generally, it is desired to rule out gaps in the workspace. Using a discrete approach, this is intrinsically impossible, but for practical usage, one may try to increase the grid resolution. Clearly this leads to a dramatical increase of the number of points to be checked and thus to extremely long computation times. To come up against this, parallelisation of the calculation by partitioning the workspace and allocation to different processing units is helpful and especially for this problem very efficient due to the independency of the workspace parts. Nevertheless, up from a specific resolution, continuous methods as presented in the next section should be considered.

4.2 Continuous analysis

In this section a method to compute the workspace of a wire robot, formulating this task as a constraint satisfaction problem (CSP), is shown. The CSP can be solved using interval analysis. However, other solving algorithms are also conceivable. The presented formulation can also be used for design just by interchanging the roles of the variables (Bruckmann et al., 2007), (Bruckmann et al., 2008b). This fact simplifies the generally complicated and complex task of robot design. For details see section 5. In (Gouttefarde et al., 2007) also interval analysis is used to determine the workspace of a wire robot. A criteria for the solvability of the interval formulation of eqn. 24 is given. In particular, the interval formulation is reduced to 2^n $n \times m$ systems of linear inequalities in the form of eqn. 24. The solvability of those 2^n systems of linear inequalities guarantees the existance of at least one valid wire force distribution. Based on this criteria a bisection algorithm is presented. This approach is beneficial in terms of the number of variables on which bisections are performed since no verification or existance variables are required. Here, however the CSP approach is presented due to its straight forward transferability to robot design.

4.2.1 Constraint satisfaction problems (CSP)

A constraint satisfaction problem (CSP) is the problem of determining all $\boldsymbol{c} \in \mathcal{X}_c$ such that

$$\boldsymbol{\Phi}(\boldsymbol{c}, \boldsymbol{v}) > 0, \forall \boldsymbol{v} \in \mathcal{X}_v, \tag{27}$$

where Φ is a system of real functions defined on a real domain representing the constraints. It will be shown later that for a description of the workspace, this problem can to be extended to

$$\Phi(\boldsymbol{c}, \boldsymbol{v}, \boldsymbol{e}) > 0, \forall \boldsymbol{v} \in \mathcal{X}_v, \exists\, \boldsymbol{e} \in \mathcal{X}_e. \tag{28}$$

Within this definition

- $\boldsymbol{c}$ is the vector of the calculation variables,
- $\boldsymbol{v}$ is the vector of the verification and,
- $\boldsymbol{e}$ is the vector of the existance variables.

The solution set for calculaton variables of a CSP is called $\mathcal{X}_S$ i.e.

$$\Phi(\boldsymbol{c}, \boldsymbol{v}, \boldsymbol{e}) > 0, \forall \boldsymbol{c} \in \mathcal{X}_S \subset \mathcal{X}_c, \forall \boldsymbol{v} \in \mathcal{X}_v, \exists \boldsymbol{e} \in \mathcal{X}_e, \tag{29}$$

where $\mathcal{X}_c$ is the so-called search domain, i.e. the range of the calculation variables wherein for solutions is searched.

4.2.2 Workspace analysis as CSP

Examining eqn. 25, the structure matrix $\boldsymbol{A}^{\mathrm{T}}$ needs to be inverted to calculate the wire forces $\boldsymbol{f}$ from a given platform pose and given external forces $\boldsymbol{w}$. Since $\boldsymbol{A}^{\mathrm{T}}$ has a non-squared shape, this is usually done using the Moore-Penrose pseudo inverse. Thus, the calculated forces will be a least squares solution. In fact, not a least squares result but a force distribution within predefined tensions is demanded. To overcome this problem, the structure matrix is divided into a squared $n \times n$ matrix $\boldsymbol{A}^{T}_{pri}$ and a second matrix $\boldsymbol{A}^{T}_{\mathrm{sec}}$ with $r = m - n$ columns. Now, the resulting force distribution can be calculated as

$$\boldsymbol{f}_{pri} = -\boldsymbol{A}^{T}_{pri}{}^{-1}(\boldsymbol{w} + \boldsymbol{A}^{T}_{sec}\boldsymbol{f}_{sec}). \tag{30}$$

In this equation, $\boldsymbol{f}_{\mathrm{sec}}$ is unknown. Every point and wrench satisfying

$$\boldsymbol{f}_{\min} \leq \boldsymbol{f}_{sec} \leq \boldsymbol{f}_{\max} \tag{31}$$

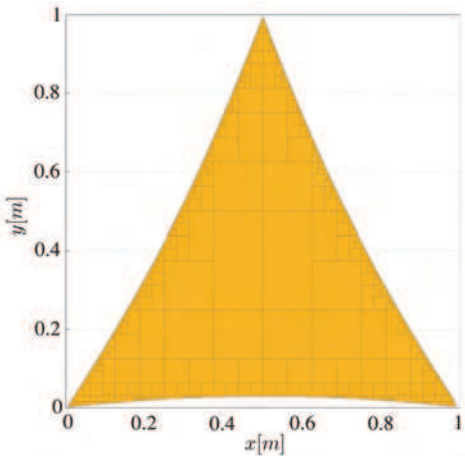

Fig. 6: Force equilibrium workspace of plain manipulator, 2 translational d.o.f., $\boldsymbol{w}^{\mathrm{T}} = (0, 0)N$, $\boldsymbol{f}_{\min} = 10N$, $\boldsymbol{f}_{\max} = 90N$

and leading to primary wire forces

$$\boldsymbol{f}_{\min} \leq \boldsymbol{f}_{pri} \leq \boldsymbol{f}_{\max} \tag{32}$$

belongs to the workspace. Hence eqns. 31 and 32 represent a CSP of the form of eqn. 28 with $\boldsymbol{f}_{sec}$ as existence an variable. To calculate a workspace for a specific robot, the following variable set for the CSP is used:

- The platform coordinates are the *calculation variables*.
- The wire forces $\boldsymbol{f}_{sec}$ are the *existence variables*.
- Optionally, the exerted external wrench w and desired platform orientations can be set as *verification variables*. The workspace for a fix orientation of the platform is called *constant orientation workspace* according to (Merlet, 2000). On the other hand, sometimes free orientation of the platform within given ranges must be possible within the whole workspace. The resulting workspace is called the *total orientation workspace*.

In fig. 6, the workspace of a simple plain manipulator is shown, based on the force equilibrium condition. In fig. 7, the workspace under a possible external load range is shown. Fig. 8(b) shows an example of the workspace of a spatial CRPM robot prototype while fig. 9(b) is the same protoype in a RRPM configuration with 8 wires. Additionally, the RRPM workspace was calculated with a verification range of ±3° for φ and θ, i.e. $\varphi = \theta = [-3, 3]$ °.

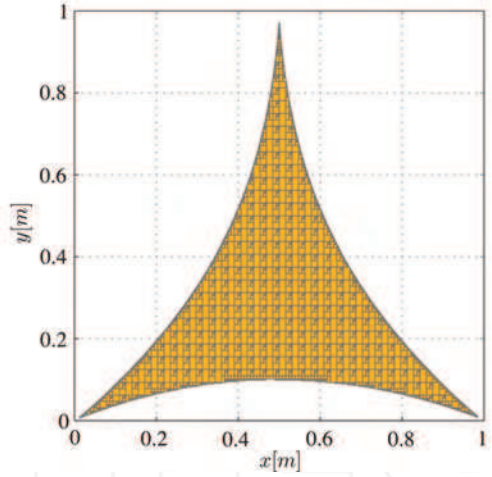

Fig. 7: Force equilibrium workspace of plain manipulator, 2 translational d.o.f., $\boldsymbol{w}^{\mathrm{T}} = ([-20, 20]N, [-20, 20]N)$, $\boldsymbol{f}_{\min} = 10N$, $\boldsymbol{f}_{\max} = 90N$

4.2.3 Interval analysis

Interval Analysis is a powerful tool to solve CSPs. Therefore a short introduction is given in the following section. For two real numbers a, b an interval $\boldsymbol{I} = [a, b]$ is defined as follows

$$[a, b] := \{r \mid a \leq r \leq b\}, \tag{33}$$

where

$$a \leq b \tag{34}$$

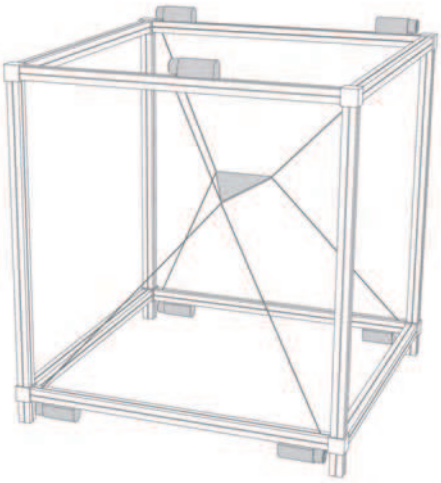

Fig. 8(a) SEGESTA prototype with 7 wires prototype

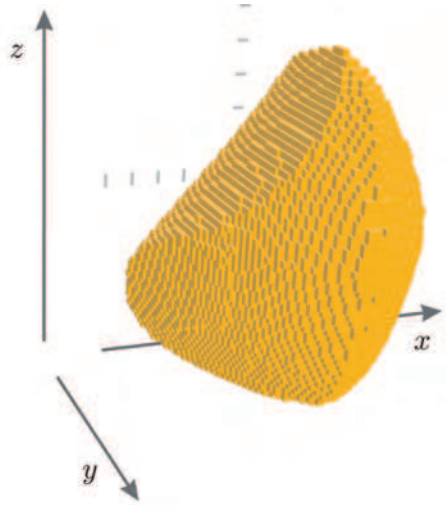

Fig. 8(b)Workspace of the SEGESTA with 7 wires

Then b is called the supremum and a the infimum of $\boldsymbol{I}$. A n-tupel of intervals is called box or interval vector. It is possible to define every operation $\circ$ on $\mathbb{R}$ on the set of intervals $\boldsymbol{I} = \{[a, b] \mid a, b \in \mathbb{R}, a \leq b\}$, such that the following holds:
Let $\boldsymbol{I}_0, \boldsymbol{I}_1 \in \boldsymbol{I}$ be two intervals. Then

$$\forall u \in I_0,\ \forall v \in I_1\ \exists z \in I_0 \circ I_1, \tag{35}$$

where

$$z = u \circ v. \tag{36}$$

Hence

$$\max_{u \in I_0, v \in I_1} u \circ v \leq \mathbf{Sup}(I_0 \circ I_1), \tag{37}$$

where < occurs if one variable appears more than once. This phenomenon is called overestimation and causes additional numerical effort to get sharp boundaries. For sure the same holds for min and Inf. Thus for input intervals $\boldsymbol{I}_0, \ldots, \boldsymbol{I}_n$ interval analysis delivers evaluations for the domain $\boldsymbol{I}_0 \times \boldsymbol{I}_1 \times \ldots \times \boldsymbol{I}_n$. This evaluation is guaranteed to include all possible solutions, e.g.

$$[1, 3] + [1, 3] \cdot [-2, 1] = [-5, 6] \tag{38}$$

while

$$[1, 3] \cdot (1 + [-2, 1]) = [-3, 6] . \quad (39)$$

As shown in detail in (Pott, 2007), a CSP can be solved using interval analysis which guarantees reliable solutions (Hansen, 1992),(Merlet, 2004b),(Merlet, 2001). Solving the CSP with interval analysis delivers a list of boxes $\mathcal{L}_S$ representing an inner approximation of $\mathcal{X}_S$. According to eqn. 29, the solutions in $\mathcal{L}_S$ hold for total $\mathcal{X}_v$ and a subset of $\mathcal{X}_e$. Additionally, available implementations for interval analysis computations are robust against rounding effects. The following CSP solving algorithms have been proposed in (Pott, 2007) and (Bruckmann et al., 2008b). To use it for the special problem of analyzing wire robots, they have been extended. Details are described in the next sections.

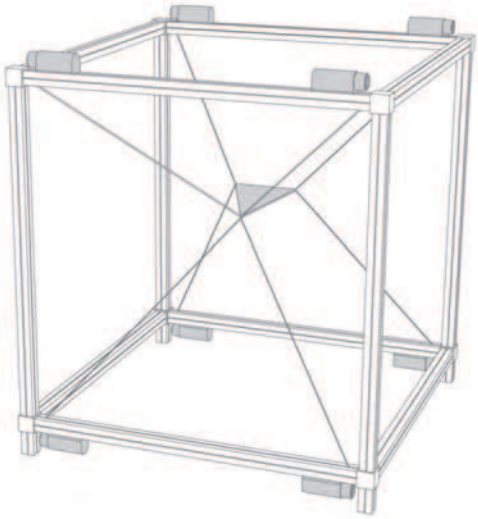

Fig. 9(a) SEGESTA prototype with 8 wires

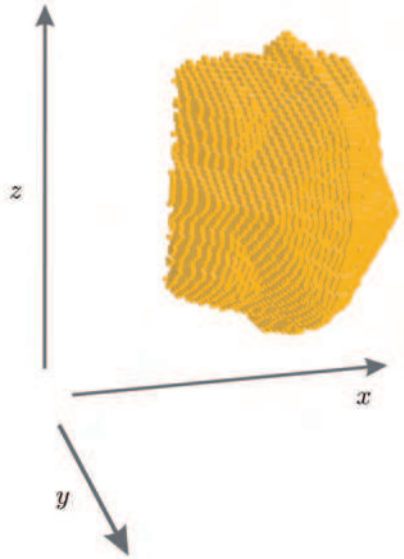

Fig. 9(b)Workspace of the SEGESTA prototype with 8 wires

Algorithm Verify

Verify is called with a box $\hat{\boldsymbol{c}}$ and checks whether

$$\begin{aligned} &\Phi(\hat{c}, v, e) > 0 \\ &\forall\, v \in \mathcal{X}_v \\ &\exists\, e \in \mathcal{X}_e \end{aligned} \quad (40)$$

is valid for the given box $\hat{\boldsymbol{c}}$. Here the domain $\mathcal{X}v$ is represented by the list of boxes $\mathcal{L}_T^v$. Thus, the result can be *valid*, *invalid*, *undefined* or *finite*. If at least one box is invalid, the whole search domain does not fulfill the required properties and is therefore invalid. Algorithm *Verify*

1. Define a search domain in the list $\mathcal{L}_T^v$. In the simplest case, $\mathcal{L}_T^v$ contains one search box.

2. If $\mathcal{L}_T^v$ is empty, the algorithm is finished with *valid*.
3. Take the next box $\hat{\boldsymbol{v}}$ from the list $\mathcal{L}_T^v$.
4. If the diameter of the box $\hat{\boldsymbol{v}}$ is smaller than a predefined value ϵ_v return with *finite*.
5. If existence variables are present, call *Existence* with $\hat{\boldsymbol{c}}$ and $\hat{\boldsymbol{v}}$. If the result is *valid*, goto (2). If the box is invalid, return with *invalid*. If the box is *finite*, goto (10).
6. Evaluate $\hat{\boldsymbol{h}} = \Phi(\hat{\boldsymbol{c}}, \hat{\boldsymbol{v}})$.
7. If Inf $\hat{\boldsymbol{h}} > 0$, the infimum of $\hat{\boldsymbol{h}}$ is greater than 0 in all its components. Thus, the box is valid. Goto (2).
8. If Sup $\hat{\boldsymbol{h}} < 0$, the supremum of $\hat{\boldsymbol{h}}$ is smaller than 0 in at least one component. Thus, the box is invalid. Return with *invalid*.
9. If Inf $\hat{\boldsymbol{h}} < 0 <$ Sup $\hat{\boldsymbol{h}}$, $\hat{\boldsymbol{h}}$ is rated as *undefined*.
10. Divide the box on a verification variable and add the parts to $\mathcal{L}_T^v$. Goto (2).

Algorithm Existence

Existence is a modification of *Verify*. It is called with the boxes $\hat{\boldsymbol{c}}$, $\hat{\boldsymbol{v}}$ and checks whether

$$\begin{aligned} &\Phi(\hat{c}, \hat{v}, e) > 0 \\ &\exists\, e \in \boldsymbol{X}_e \end{aligned} \tag{41}$$

is valid. Here the domain $\mathcal{X}_e$ is represented by the list of boxes $\mathcal{L}_T^e$ The result can be *valid*, *invalid* or *finite*. If at least one box is valid, the whole search domain fulfills the required properties and is therefore valid. Algorithm *Existence*

1. Define a search domain in the list $\mathcal{L}_T^e$. In the simplest case, $\mathcal{L}_T^e$ contains one search box.
2. If $\mathcal{L}_T^e$ is empty, the algorithm is finished with *invalid*.
3. Take the next box $\hat{\boldsymbol{e}}$ from the list $\mathcal{L}_T^e$.
4. If the diameter of the box $\hat{\boldsymbol{e}}$ is smaller than a predefined value ϵ_e, return with *finite*.
5. *Evaluate* $\hat{\boldsymbol{h}} = \Phi(\hat{\boldsymbol{c}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{e}})$.
6. If Inf $\hat{\boldsymbol{h}} > 0$, the infimum of $\hat{\boldsymbol{h}}$ greater than 0 in all its components. Thus, the box is valid. Return with *valid*.
7. If Sup $\hat{\boldsymbol{h}} < 0$, the supremum of $\hat{\boldsymbol{h}}$ smaller than 0 in at least one component. Goto (2).
8. If Inf $\hat{\boldsymbol{h}} < 0 <$ Sup $\hat{\boldsymbol{h}}$, $\hat{\boldsymbol{h}}$ is rated as *undefined*. Divide the box on an existence variable and add the parts to $\mathcal{L}_T^e$. Goto (2).

Algorithm Calculate

Calculate is called with a search domain for $\boldsymbol{c}$ represented by a list of boxes $\mathcal{L}_T^c$. It uses *Existence* or *Verify* to identify valid boxes within the search domain. Thus, the result is a list $\mathcal{L}_S$ of valid boxes (and optionally the lists $\mathcal{L}_I$ for invalid boxes and $\mathcal{L}_F$ for finite boxes, respectively). Algorithm *Calculate*

1. Define a search domain in the list $\mathcal{L}_T^c$. In the simplest case, $\mathcal{L}_T^c$ contains one search box.

2. Create the lists
   (a) $\mathcal{L}_S$ for solution boxes,
   (b) $\mathcal{L}_I$ for invalid boxes,
   (c) $\mathcal{L}_F$ for finite boxes.
3. If $\mathcal{L}_T^c$ is empty, the algorithm is finished.
4. Take the next box $\hat{\boldsymbol{c}}$ from the list $\mathcal{L}_T^c$.
5. If the diameter of the box $\hat{\boldsymbol{c}}$ is smaller than a predefined value ϵ_c the box is treated as *finite* and thus moved to the list $\mathcal{L}_F$. Goto (3).
6. If verification variables are present, call *Verify* with $\hat{\boldsymbol{c}}$. Otherwise call *Existence* with $\hat{\boldsymbol{c}}$ and an empty box for $\hat{\boldsymbol{v}}$.
7. If the result of *Verify* is *valid*, move the box to the solution list $\mathcal{L}_S$. Goto (3).
8. If the result of *Verify* is *invalid*, move the box to the invalid list $\mathcal{L}_I$. Goto (3).
9. If the result of *Verify* is *finite*, move the box to the finite list $\mathcal{L}_F$. Goto (3).

Calling Sequence

Let $\mathcal{X}_c$, $\mathcal{X}_v$, $\mathcal{X}_e \neq 0$ be given and represented as lists of boxes $\mathcal{L}_T^c, \mathcal{L}_T^v, \mathcal{L}_T^e$. In order to determine $\mathcal{L}_S$, *Calculate* is called with the search domain $\mathcal{L}_T^c$. Within *Calculate*, *Verify* is called. Since existence variables are present, *Existence* is called in order to validate the current calculation box (Otherwise in *Verify* the CSP would be directly evaluated). In the *Existence* algorithm the CSP is evaluated and the result is rated. In case that the result is *undefined*, the current box is divided on an existence variable. In case that the *Existence* algorithm returns with *finite*, the calling algorithm divides on its own variables and calls *Existence* again. If the result is *valid* or *invalid*, the result is directly returned to the calling algorithm. If *valid* is returned, the result is valid for all values within $\hat{\boldsymbol{c}}$ and $\hat{\boldsymbol{v}}$. The same calling sequence and return behaviour is used in *Calculate* calling *Verify*. For an effective CSP solver the return scheme should be more advanced in the way that not one variable is bisected until the box under consideration is finite, but a more sophisticated bisection distribution is used. It is noteworthy that the calculation time increases considerably with the number of variables and decreasing ϵ_i, $i \in \{c, v, e\}$.

Preliminary Checks

Since solving the force equilibrium is a computationally expensive task, favorable prechecks are demanded to reduce computation time. An effective check is to examine the interval evaluation of $\tau_{check} := \boldsymbol{A}^T \boldsymbol{f}_{check} + \boldsymbol{w}$ for $\boldsymbol{f}_{check}$ being the box with infimum $\boldsymbol{f}_{min}$ and supremum $\boldsymbol{f}_{max}$. If

$$\exists i \in 1,\ldots, m \; 0 \notin \tau_{check,i}\,, \tag{42}$$

one can conclude that the poses under consideration do not belong to the workspace under the given load $\boldsymbol{w}$ due to the non-existance of valid wire force distributions. The resulting preliminary workspace is an outer estimate and excludes poses which are not treated furthermore. Another possibility to reduce the computation time is to take symmetries into account. If symmetry axes as well as a symmectrical load range are present it is sufficient to compute only one part of the workspace and to complete the workspace by proper mirroring.

4.3 Further criteria

4.3.1 Stiffness

Besides the force equilibrium, additional workspace conditions can be applied. Due to the high elasticity of the wires (using plastic material, e.g. polyethylene), the stiffness may be low in parts of the workspace. Thus, for practical applications, especially if a predefined precision is required, it may be necessary to guarantee a given stiffness for the whole workspace. Otherwise, the compensation of elasticity effects by control may be required. Generally, this should be avoided as far as possible by an appropriate design. As shown in (Verhoeven, 2004), the so-called passive stiffness can be described as the reaction of a mechanical system onto a small pertubation, described by a linear equation:

$$\delta \boldsymbol{w} = \boldsymbol{K}(\boldsymbol{x})\delta \boldsymbol{x} \tag{43}$$

where

$$\boldsymbol{K}(\boldsymbol{x}) = k' \boldsymbol{A}^T(\boldsymbol{x}) \boldsymbol{L}^{-1}(\boldsymbol{x}) \boldsymbol{A}(\boldsymbol{x}). \tag{44}$$

Here, $\boldsymbol{L}$ is the diagonal matrix of the wire lengths and k' is the proportionality factor (force per relative elongation), treating the wires as linear springs. For the calculation, the inverse problem

$$\delta \boldsymbol{x} = \boldsymbol{K}(\boldsymbol{x})^{-1} \delta \boldsymbol{w} \tag{45}$$

is solved and evaluated where only domains having a position pertubation within the predefined limits $\delta \boldsymbol{x}_{\min}$ and $\delta \boldsymbol{x}_{\max}$ under predefined loads between $\delta \boldsymbol{w}_{\min}$ and $\delta \boldsymbol{w}_{\max}$ are considered as workspace. This equation can again be treated as a CSP. However, stiffness can also be checked performing a discrete workspace analysis. The stiffness workspace for a simple plain manipulator with 2 translational d.o.f. is shown in fig. 10(a). The parameters k' = 1000**N**, $\boldsymbol{f}_{\min}$ = 10**N** and $\boldsymbol{f}_{\max}$ = 90**N** were set. For a given load of $\delta \boldsymbol{w}$ = ([−20, 20]**N**, [−20, 20]**N**) the platform was allowed to sag elastically in the ranges $\delta \boldsymbol{x}$ = ([−0.015, 0.015]**N**, [−0.015, 0.015]**N**).

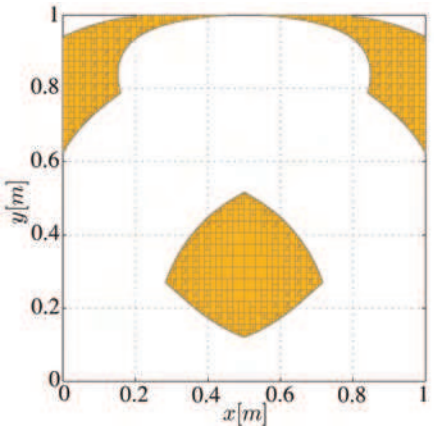

Fig. 10(a) Stiffness workspace of plain manipulator

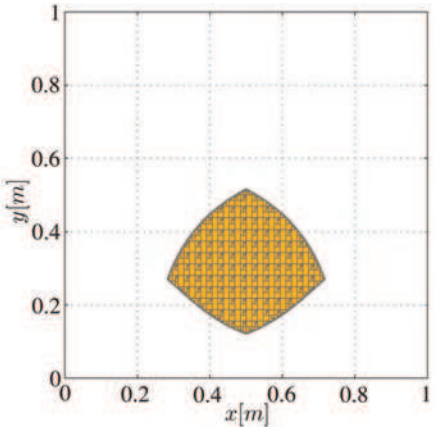

Fig. 10(b) Combined force equilibrium and stiffness workspace of plain manipulator

4.3.2 Singularities

A pose of a wire robot is said to be singular if and only if

$$rank\ \boldsymbol{A}^T < n. \tag{46}$$

Therefore all wire robots with pure translational d.o.f. are singularity free except those, which are always singular (Verhoeven, 2004). For a wire robot with rotational and translational d.o.f. the workspace certainly has be to checked for singularities. Since within the workspace analysis (discrete or continuous) typically a system of linear equations is solved, the singularity criteria eqn.46 can be checked implicitly. Mechanically, at singular poses certain d.o.f. become uncontrollable (overmobility). Often this happens in symmetrical configurations.

4.3.3 Wire collision

In analogy to the problem of link collisions for conventional parallel manipulators, wire collisions have to be avoided. Due to their normally small diameter one possibility is to consider the wires as lines. In (Merlet, 2004a) an algorithm is proposed to determine the regions in which collisions between wires as well as the collisions between wires and the end-effector occur. Practically, wires have certain diameter and thus, a predefined minimum distance (at least the wire diameter) should be always ensured. Therefore, the well-known problem of determining the smallest distance between two lines arises. Since the lines are known after solving the inverse kinematics this is a very basic task but may be computational expensive. Clearly, the distance condition has to be formulated as a inequality. Hence, this criteria can be easily included in the CSP formulation.

5. Robot design

While workspace analysis examines the properties of already parametrized manipulators which allows to determine the applicable use cases, robot design describes the opposite task of finding the optimal robot for a given task. Generally, the task is abstracted e.g. as a desired workspace or a desired path or trajectory. To identify the optimal robot, usually different designs have to be compared with respect to the desired properties which makes the design process generally a computationally expensive task. Finally, one or more designs turn out as most favourable. In parallel to the analysis methods, again both discrete as well as continuous methods are available and show differences in the analysis quality and the calculation effort. For the continuous approach the CSP formulation can be used again which is amongst others advantageous in terms of implementation effort. The interchanging of the roles of the variables turns the workspace analysis just into a design task. According to (Merlet, 2005), the design (or synthesis) task can be divided into two separated subtasks:

- structure synthesis: This step includes the determination of the topology of the mechanical structure. In particular, the number and type of d.o.f. of the joints and their interconnection is identified.
- dimensional synthesis: Here position and orientation of the joints as well as the length of the links is specified.

For the special case of a wire robot, the structure synthesis covers different aspects: While the link topology itself is fixed, one has to choose the number of wires wisely.

Additionally, the concurrence of at least two (in the planar case) or three (in the spatial case) platform connection points may be prudential:

- Forward kinematic calculations become much easier (see section 2.3).
- The number of design parameters is reduced, which is beneficial in terms of computation time.
- The occurence of wire collisions is reduced since wires can intersect in at most one point.
- The workspace is comparably large (Fang, 2005).

After completion of the structure synthesis a dimensional synthesis can be performed. For a wire robot this is nothing but the identification of feasible base points. This section is addressed to dimensional synthesis mainly.

5.1 Discrete synthesis

Discrete methods are widely used for wire robot design. In (Fattah & Agrawal, 2005) and (Pusey et al., 2004) both the parameter set and an assumed superset of the workspace are discretized. Then for every point on the resulting parameter grid the discretized workspace is computed and its volume is determined by counting the points on the grid fulfilling all workspace conditions. The approaches share the same concept:

1. Build up an equidistant Grid of the design variables and loop through all parameter sets.
2. For every parameter set, specify a superset of the workspace and discretize it by an equidistant grid.
3. Loop through all grid points of step 2. For every point, determine if a valid wire force distribution according to eqn. 25 and 26 exists.
4. Count all points belonging to the workspace and store the number for every parameter set.
5. Obtain the maximum volume workspace, i.e., the maximum of all workspace volumes that are counted in step 4, and the associated optimized design variables.

Instead of the volume of the workspace a different optimization criterion can be employed. To increase the practical usability and the robustness of the design, a dexterity criterion is proposed, which uses the condition number of the structure matrix $\boldsymbol{A}^{\mathrm{T}}$. These approaches have two drawbacks. Since the design variables are discretized, every combination of parameters is checked. Hence, this method is computationally intensive. Furthermore, no desired workspace can be guaranteed by the obtained design. Hay and Snyman use a special optimizer instead of a grid of the design variables (Hay & Snyman, 2004), (Hay & Snyman, 2005). Again, in this approach a desired workspace is not guaranteed by the obtained optimal design.

5.2 Continuous synthesis (Design-To-Workspace)

Examining eqn.28, eqn.31 and eqn.32, the roles of the variables can arbitrary be assigned. An imaginable choice is

- The winch poses and platform fixation points are the *calculation variables*. Thus, the calculation delivers robot designs solving the CSP.
- The platform coordinates are *verification variables*. Hence, the workspaces of all resulting robot designs will cover the set given in $\mathcal{X}_v$ for the platform coordinates for sure.

- Optionally, the exerted external wrench $\boldsymbol{w}$ and desired platform orientations can be set as *verification variables* to extend the applicability of the emerged designs for certain process wrenches and tasks.
- The wire forces $\boldsymbol{f}_{\text{sec}}$ are the *existence variables*.

The suggested choice of variables leads to a CSP, whose solutions are robot designs. Furthermore, each obtained robot can reach every point given in $\mathcal{X}_v$ for the platform coordinates with every orientation and wrench given in $\mathcal{X}_v$. Generally, the design task is deemed to be more complicated than the analysis. Here, the methods and formulations are inherited and just adapted to the design problem. Nevertheless, robot design is a computationally intensive task. The use of parallel computations is strongly advised. Solving the CSP is advantageous due to the following reasons:

- The workspaces of the resulting designs are guaranteed to have no holes or singularities.
- The design process can be extended by a global optimization step.
- The interval CSP solver can be effectively parallelized.

5.3 Continuous optimization

Optimization is always performed with respect to a cost function. In industrial application usually the term optimal is used with respect to economic aspects, i.e. costs. In the case of wire robots, the most cost-driving factor are the wire winch units. However, optimizing the number of winches is part of the structure synthesis. Thus, here another cost function has to be chosen. This choice is generally arbitrary. Nevertheless, a reasonable choice is the volume expansion. On one hand, reducing the expansion of the robot saves space within a production facility which reduces costs, on the other hand, the required wire lengths are minimized. In literature, usually the optimization is performed with respect to the size (or volume) of the workspace or the integral of workspace indices over the workspace. This gives finally the robot with optimal (e.g. largest) workspace with respect to some criterion, but it says nothing about its shape and its usability for applications. Thus, here another approach is used (Pott, 2007): Not a maximum size of the workspace is demanded, but the guaranteed enclosure of a predefined domain is desired. The optimization is performed using interval analysis. Let a list $\mathcal{L}$ of n boxes of robot designs, e.g. a solution of the according CSP be given. The following algorithm performs the required steps for a minimization (maximization is performed analogously):

1. Set $i = 0$ and $\boldsymbol{F}_{opt} = [\infty,\infty]$.
2. Set $i = i + 1$. If $i > n$ the algorithm finishes.
3. Take the i-th element l_i of $\mathcal{L}$ and compute its cost function $\boldsymbol{F}(l_i)$.
4. If $\text{Sup}(\boldsymbol{F}(l_i)) < \text{Sup}(\boldsymbol{F}_{opt})$, set $\boldsymbol{F}_{opt} = F(l_i)$.
   - If $\text{Sup}(\boldsymbol{F}(l_i)) < \text{Inf}(\boldsymbol{F}_{opt})$ delete all elements of the solution list and initialize it with l_i. Goto 2.
   - Store l_i in the solution list. Goto 2.
5. If $\text{Inf}(\boldsymbol{F}(l_i)) < \text{Sup}(\boldsymbol{F}_{opt})$ store l_i in the solution list.
6. Discard l_i and goto 2

For performance reasons the optimization can be included in the CSP Solver. This will reduce computation time drastically since non-optimal designs are discarded at an early

stage. An example for the optimization of an 1R2T robot is shown in fig. 11(b). For the upper winches, y-positions are free, for the lower ones, the x-positions are the free optimization parameters.

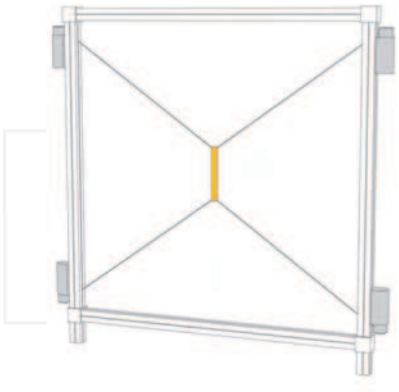

Fig. 11(a) 1R2T example

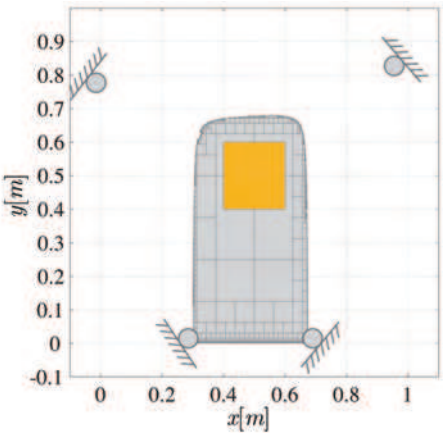

Fig. 11(b) 1R2T robot optimized for shown desired quadratic workspace.

5.4 Design-To-Task

The Design-to-Workspace method results in manipulators, guaranteed to have a desired workspace. Thus, the manipulator is able to perform every task within this workspace. Nevertheless, from the economic point of view, there is a need for manipulators which perform a specific task in minimum time, with minimum energy consumption or with lowest possible power. A typical industrial application is e.g. the pick-and-place task, moving a load from one point to another. Usually, this task is performed within series production, i.e. it is repeated many times. In such an application the optimal manipulator for sure finishes the job in minimal time with respect to the technical constraints (here, the term optimal is used with respect to minimal time without loss of generality). Thus, the set-up of a specialized (i.e. taskoptimized) manipulator can be profitable. When using classical industrial robots, the freedom to modify the mechanical setup of the robot is very limited. Thus, only the trajectories can be modified and optimized with respect to the task. Due to the modular design of a wire robot, the task-specific optimization can be seperated into two tasks:

- Optimization of the robot: within all suitable designs, the robot which performs the task in shortest time is chosen.
- Optimization of the trajectory: within all possible trajectories, the trajectory which connects the points in shortest time is chosen. The concepts needed for this step are partly explained in (Bianco & Piazzi, 2001b),(Bianco & Piazzi, 2001a) and (Merlet, 1994).

By treating this task as a CSP, both claims can be optimized at the same time. In particular, the final result contains the robot which is able to perform the task quickest *and* the corresponding trajectory description. To perform an optimization of the wire robot and the trajectory simultaneously, the latter is planned first. Afterwards it is checked whether the complete trajectory belongs to the workspace. The robot designer may provide a predefined

trajectory or leave this up to the optimizer. The parameters of the trajectory are therefore either fixed or *calculation variables*. Hence, the CSP looks the same as in eqn.31 and eqn.32 except the previous trajectory generation. For integrated optimization, the variables are assigned as follows. Note, that also a separate optimization of robot and trajectory is possible:

- Robot optimization
  - The robot base is described by the positions of the winches. To optimize the robot, the winches can be moved. Therefore, $\boldsymbol{b}_i$ are *calculation variables*
  - The end effector is described by the positions of the platform anchor points $\boldsymbol{p}_i$. To optimize the robot, these points can be moved on the platform. Therefore, $\boldsymbol{p}_i$ are *calculation variables*
- Trajectory optimization
  - The path is described by a polynomial of fourth order without loss of generality. Besides the start and end poses, also the velocities are predefined. This leaves one free parameter, e.g. the start acceleration for translational d.o.f. or the orientation at half travel time for rotational d.o.f.. These can be set as *calculation variables*.
  - To describe the trajectory, additionally the travel time T has to be defined. To calculate the minimum time, T is a *calculation variable*.
  - For the whole trajectory, a path parameter t is assigned. Usually, it is normalized between zero and one. Since the whole trajectory shall betraced for validity, t is a *verification variable*

Optionally, the exerted external wrenches $\boldsymbol{w}$ can be set as *verification variables*. Note, that within the trajectory verification the dynamics of the robot are taken into account by adding the inertia loads resulting from the calculated accelerations to the platform loads $\boldsymbol{w}$. The example in fig. 12(b) shows the result of an optimization for a point-to-point (PTP) movement. A $n = 3$ d.o.f. wire robot with $m = 4$ wires is considered (see 12(a)). It consists of a bar-shaped platform of $0.1m$ length, connected by four winches to the base frame. Free optimization parameters were the y-position of the upper right winch, the travel time and the intermediate acceleration of the rotation angle at $T = 0.5$ s.

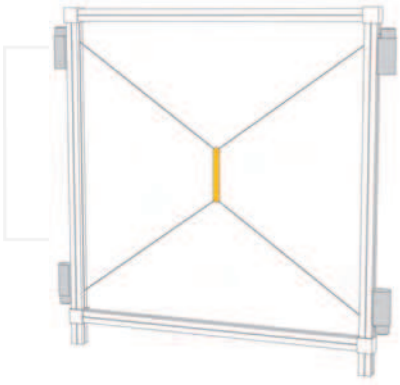

Fig. 12(a) 1R2T example

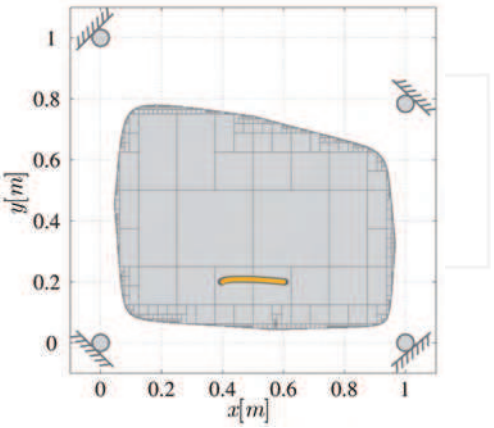

Fig. 12(b) 1R2T robot optimized for shown desired PTP trajectory

6. Conclusion

In this chapter, the analysis and design of wire robots was discussed. The required basics like kinematics and the force equilibrium - which is the one of the main workspace criteria - were introduced as well as serveral classification approaches. The analysis of wire robots was described as a CSP task which can be solved by interval analysis. Besides reliable results, the same CSP can be used for robot design by a variable exchange, which is generally a challenging problem. In addition to this continuous approach, also the more straightforward discrete methods are shortly introduced. The next chapter is dedicated to the application and control of wire robots. Therefore, the dynamical description as well as different methods to calculate a force distribution for a given pose and platform wrench are presented. Based on this, some control concepts are described. The use of wire robots for several fields of application is demonstrated by a number of examples.

7. Acknowledgements

This work is supported by the German Research Council (Deutsche Forschungsgemeinschaft) under HI370/24-1, HI370/19-3 and SCHR1176/1-2. The authors would like to thank Martin Langhammer for contributing the figure design.

8. References

Bianco, C. G. L. and Piazzi, A. (2001a). A hybrid algorithm for infinitely constrained optimization. *International Journal of Systems Science*, 32(1):91–102.

Bianco, C. G. L. and Piazzi, A. (2001b). A semi-infinite optimization approach to optimal spline trajectory planning of mechanical manipulators. In Goberna, M. A. and Lopez, M. A., editors, *Semi-Infinite Programming: Recent Advances*, chapter 13, pages 271–297. Kluwer Academic Publisher.

Bosscher, P. and Ebert-Uphoff, I. (2004). Wrench-based analysis of cable-driven robots. *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, pages 4950–4955.

Bruckmann, T., Mikelsons, L., Brandt, T., Hiller, M., and Schramm, D. (2008a). Wire robots part II - dynamics, control & application. In Lazinica, A., editor, *Parallel Manipulators*, ARS Robotic Books. I-Tech Education and Publishing, Vienna, Austria. ISBN 978-3-902613-20-2.

Bruckmann, T., Mikelsons, L., and Hiller, M. (January 9-11, 2008b). A design-to-task approach for wire robots. In Kecskeméthy, A., editor, *Conference on Interdisciplinary Applications of Kinematics 2008*, Lima, Peru.

Bruckmann, T., Mikelsons, L., Schramm, D., and Hiller, M. (2007). Continuous workspace analysis for parallel cable-driven stewart-gough platforms. *to appear in Proceedings in Applied Mathematics and Mechanics*.

Fang, S. (2005). *Design, Modeling and Motion Control of Tendon-based Parallel Manipulators*. Ph. D. dissertation, Gerhard-Mercator-University, Duisburg, Germany. Fortschritt-Berichte VDI, Reihe 8, Nr. 1076, Düsseldorf.

Fattah, A. and Agrawal, S. K. (2005). On the design of cable-suspended planar parallel robots. *ASME Transactions, Journal of Mechanical Design*, 127(5):1021– 1028.

Gouttefarde, M., Merlet, J.-P., and Daney, D. (2007). Wrench-feasible workspace of parallel cable-driven mechanisms. *2007 IEEE International Conference on Robotics and Automation, ICRA 2007, 10-14 April 2007, Roma, Italy*, pages 1492–1497.

Hansen, E. *(1992). Global Optimization using Interval Analysis.* Marcal Dekker, Inc.

Hay, A. and Snyman, J. (2004). Analysis and optimization of a planar tendon-driven parallel manipulator. In Lenarcic, J. and Galetti, C., editors, *Advances in Robot Kinematics,* pages 303–312, Sestri Levante.

Hay, A. and Snyman, J. (2005). Optimization of a planar tendon-driven parallel manipulator for a maximal dextrous workspace. In *Engineering Optimization,* volume 37 of 20, pages 217–236.

Landsberger, S. and Sheridan, T. (1985). A new design for parallel link manipulator. In *International Conference on Cybernetics and Society*, pages 812–814,, Tucson, Arizona.

Maier, T. (2004). *Bahnsteuerung eines seilgeführten Handhabungssystems - Modellbildung, Simulation und Experiment.* PhD thesis, Universität Rostock, Brandenburg. Fortschritt-Berichte VDI, Reihe 8, Nr. 1047, Düsseldorf.

Merlet, J.-P. (1994). Trajectory verification in the workspace for parallel manipulators. The *International Journal of Robotics Research,* 13(4):326–333.

Merlet, J.-P. (2000). *Parallel Robots.* Kluwer Academic Publishers, Norwell, MA, USA.

Merlet, J.-P. (2001). A generic trajectory verifier for the motion planning of parallel robots. *Journal of Mechanical Design,* 123:510–515.

Merlet, J.-P. (2004a). Analysis of the influence of wires interference on the workspace of wire robots. *On Advances in Robot Kinematics*, pages 211–218.

Merlet, J.-P. (2004b). Solving the forward kinematics of a gough-type parallel manipulator with interval analysis. *Int. J. of Robotics Research,* 23(3):221–236.

Merlet, J.-P. (2005). Optimal design of robots. In *Robotics: Science and Systems,* Boston.

Ming, A. and Higuchi, T. (1994). Study on multiple degree of freedom positioning mechanisms using wires, part 1 - concept, design and control. *International Journal of the Japan Society for Precision Engineering,* 28:131–138.

Pott, A. (2007). *Analyse und Synthese von Parallelkinematik-Werkzeugmaschinen.* Ph. D. dissertation, Gerhard-Mercator-University, Duisburg, Germany. Fortschritt-Berichte VDI, Reihe 20, Nr. 409, Düsseldorf.

Pusey, J., Fattah, A., Agrawal, S. K., and Messina, E. (2004). Design and workspace analysis of a 6-6 cable-suspended parallel robot. *Mechanism and Machine Theory.* Vol. 39, No.7, pp.761-778.

Verhoeven, R. (2004). Analysis *of the Workspace of Tendon-based Stewart Platforms.* PhD thesis, University of Duisburg-Essen.

Williams, R. L., Albus, J. S., and Bostelman, R. V. (2004). 3d cable-based cartesian metrology system. *Journal of Robotic Systems*, 21(5):237–257.

Parallel Manipulators, New Developments

Edited by Jee-Hwan Ryu

ISBN 978-3-902613-20-2
Hard cover, 498 pages
Publisher I-Tech Education and Publishing
Published online 01, April, 2008
Published in print edition April, 2008

Parallel manipulators are characterized as having closed-loop kinematic chains. Compared to serial manipulators, which have open-ended structure, parallel manipulators have many advantages in terms of accuracy, rigidity and ability to manipulate heavy loads. Therefore, they have been getting many attentions in astronomy to flight simulators and especially in machine-tool industries.The aim of this book is to provide an overview of the state-of-art, to present new ideas, original results and practical experiences in parallel manipulators. This book mainly introduces advanced kinematic and dynamic analysis methods and cutting edge control technologies for parallel manipulators. Even though this book only contains several samples of research activities on parallel manipulators, I believe this book can give an idea to the reader about what has been done in the field recently, and what kind of open problems are in this area.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821