Solving the Forward Kinematics of Cable-Driven Parallel Robots with Neural Networks and Interval Arithmetic

Valentin Schmidt, Bertram Müller and Andreas Pott

Abstract This paper investigates a new approach for solving the forward kinematics of cable-driven parallel robots. This approach combines an interval algorithm with neural networks to provide a fast but accurate initial guess. The neural networks increase the computation speed by a factor of 200 or more, while the interval algorithm provides guaranteed convergence and a definite solution to any chosen degree of accuracy. Iterative techniques are faster still, but the proposed algorithm is considered real-time feasible.

Keywords Cable-driven robots · Neural networks · Interval analysis · Kinematics

1 Introduction

Cable-driven parallel robots, from now on referred to as cable robots, are a class of parallel robots actuated by flexible cables instead of rigid members.

We will differentiate between two types of cable robots. Completely/redundantly restrained cable robots, where the number of cables m exceeds the degrees of freedom n in order to achieve full control; and a second type of cable robot: the suspended cable robots. Suspended cable robots rely on an external wrench to control the robot position and are most often seen as hanging structures.

The forward kinematics—obtaining platform pose from cable lengths—of parallel machines is a difficult problem. However, for the operation and for advanced

V. Schmidt (✉) · B. Müller · A. Pott
Fraunhofer Institute for Manufacturing Engineering and Automation IPA,
70569 Stuttgart, Germany
e-mail: Valentin.Schmidt@ipa.fraunhofer.de

B. Müller
e-mail: Bertram.Mueller@ipa.fraunhofer.de

A. Pott
e-mail: Andreas.Pott@ipa.fraunhofer.de

control techniques of cable robots, a numerically stable and fast computation of the forward kinematics is needed. In practice, this is often achieved by using optimization algorithms.

In this paper an algorithm for completely restrained cable robots, based on combining neural networks with interval arithmetic, is introduced and tested. This combined approach offers greater computational speed than algorithms purely based on interval arithmetic, whilst maintaining strict confidence in the obtained solution.

2 Literature Review

Various methods to compute the forward kinematics of parallel manipulators already exist. In some specific cases the problem can be simplified due to specific geometrical traits, resulting in a set of algebraic descriptions which can easily be solved symbolically [3, 12]. Algebraic formulations have also been made for the general case, resulting in high-degree polynomials which are very difficult to solve [7].

Merlet introduced a numerically more stable approach for the general parallel machine using Interval Arithmetic [9].

These approaches stand in contrast to the iterative optimization techniques, which evaluate the inverse Kinematics repeatedly, to gain an increasingly accurate pose with each step. The computation time of such techniques is acceptable and has been successfully implemented for real-time execution on robot systems [10].

Suspended cable robots require additional considerations when evaluating the kinematics. Since the external wrench applied by gravity is an integral part to the robots structure, it needs to be taken into account when evaluating the robot's pose. Recent publications discuss this static and dynamic coupling and propose algebraic solutions [2, 4]. Ghasemi has presented a successful implementation of neural networks to find the solution for a suspended cable robot [5].

2.1 Shortcomings of Existing Methods

Despite having a wide range of computational approaches to choose from, the evaluation of forward kinematics remains a challenge, especially when considering real time constraints on modern machines. Simplifying the problem through the selection of special geometries is not always a feasible step depending on the constraints of the robot considered.

Mathematically exact methods, which are capable of finding all possible solutions and using heuristics to select the right one, are not feasible for production machines due to numerical instabilities or excessive computation time. Interval arithmetic on the other hand yields certain guarantees due to its deterministic nature. Unfortunately, interval arithmetic routines are based on exhaustive search algorithms and typically do not perform in a real-time environment.

Faster iterative optimization algorithms, however, demand a good initial guess for the solution and do not guarantee convergence. This can have unexpected consequences and requires additional sub-routines to manage non-convergence, or even divergence from the solution.

Neural networks are computationally fast algorithms but also lack the capability of providing a guaranteed solution or any indication of confidence. Further, these networks need to be trained a-priori, consuming additional time and effort.

All approaches suffer under kinematic sensitivity [8]. This provides a measure of how numerically sensitive the poses are to changes in the cable lengths and an indication why some solutions are hard to find. Further discussion is out of scope for this paper.

2.2 Benefits of the Combined Approach

Using neural networks to provide a fast initial estimate for the solution and then applying an interval algorithm to the smaller search space, combines the strengths of these two methods. Unlike optimization algorithms interval analysis can easily differentiate between whether an exact solution can be found in the given bound, to a predetermined level of accuracy, or whether a solution is indeterminable. Even detecting singularities is plausible [9]. Neural Networks, once trained, provide a very fast evaluation with minimal computational effort.

It is this combination approach which enables a robust, but still relatively fast computational method.

3 Description of the Combination Approach

The cable-driven parallel robot IPAnema illustrated in Fig. 1, has eight cables and six degrees of freedom, making it redundantly restrained. The geometry is described using two coordinate systems, one for the platform $\mathcal{K}_\mathcal{P}$ and one global coordinate system for the base $\mathcal{K}_\mathcal{O}$. The vector notation describes the inverse kinematics for cable i as:

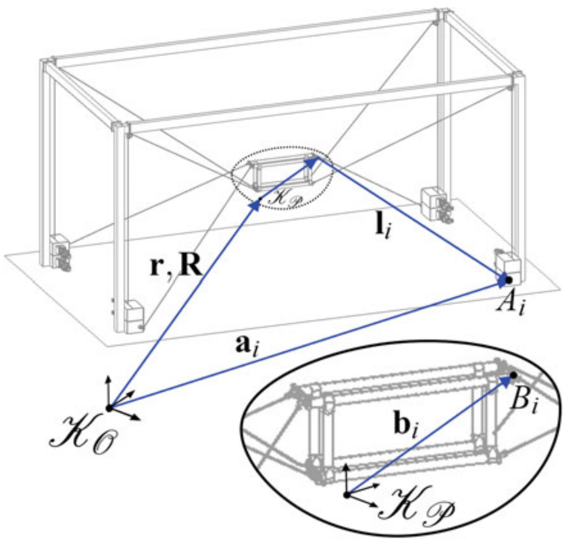
$$\|\mathbf{a}_i - \mathbf{r} - \mathbf{R}\mathbf{b}_i\|_2 = l_i \quad \text{for}\, i = 1, \dots, m \tag{1}$$

where $\mathbf{a}_i, \mathbf{b}_i, \mathbf{r}$ are vectors describing attachment points and platform position, l_i is the length of the cable and $\mathbf{R}$ the rotation matrix at a given pose.

The forward kinematics currently implemented in the IPAnema cable robot [10] uses a Levenberg-Marquardt (LM) algorithm to optimize the function

$$\Psi_i\,(\mathbf{l}, \mathbf{r}, \mathbf{R}) = (\|\mathbf{a}_i - \mathbf{r} - \mathbf{R}\mathbf{b}_i\|_2)^2 - l_i^2 \quad \text{for}\, i = 1, \dots, m \tag{2}$$

to finde the pose $\mathbf{r}, \mathbf{R}$ for a set of cable lengths l_i.



Geometry in [m]

	i	x	y	z
$\mathbf{a}_i$:	1	−4.3	3.0	4.6
	2	4.3	3.0	4.5
	3	4.0	−2.7	4.6
	4	−4.0	−2.8	4.7
	5	−4.3	2.9	0.5
	6	4.3	2.9	0.4
	7	4.0	−2.8	0.4
	8	−4.0	−2.8	0.6
$\mathbf{b}_i$:	1	−0.7	0.2	−0.1
	2	0.6	0.2	−0.1
	3	0.6	−0.1	−0.1
	4	−0.6	−0.1	−0.1
	5	−0.8	0.2	0.4
	6	0.8	0.2	0.4
	7	0.8	−0.0	0.4
	8	−0.8	−0.0	0.4

Fig. 1 IPAnema Robot with Geometrical Parameters: base vector $\mathbf{a}_i$ and platform vector $\mathbf{b}_i$

For the interval algorithm a different parameterization, based on distance equations, is used. This parameterization focuses on finding the position of f linearly independent reference points in the global coordinate frame and hence the pose of the platform. These reference points on the platform were chosen to be cable attachment points and are chosen so that all other attachment points j are linearly dependent on these reference points k. This relation in the coordinate system $\mathcal{K}_\mathcal{P}$ is described by

$$\mathbf{b}_j = \sum_k \mathbf{C}\mathbf{b}_k, \tag{3}$$

where the conversion matrix $\mathbf{C}$ is calculated offline.

The equation sets describing the kinematics, which are solved by the interval algorithm are then as follows. The cable length for each reference point (x_k, y_k, z_k)

$$\left(x_k - A_k^x\right)^2 + \left(y_k - A_k^y\right)^2 + \left(z_k - A_k^z\right)^2 = l_k^2 \quad \text{for } k \in 1, 2, \ldots, f, \tag{4}$$

the remaining cable lengths

$$\left(\sum_{k=1}^{f} \mathbf{C}x_k - A_j^x\right)^2 + \left(\sum_{k=1}^{f} \mathbf{C}x_k - A_j^y\right)^2 + \left(\sum_{k=1}^{f} \mathbf{C}x_k - A_j^z\right)^2 = l_j^2 \quad \text{for } j \in f+1, \ldots, n \tag{5}$$

and the distance between reference point pairs

$$\left(x_p - x_q\right)^2 + \left(y_p - y_q\right)^2 + \left(z_p - z_q\right)^2 = \delta_{pq}^2 \quad \text{for } p, q \in 1, 2, \ldots, f, p \neq q. \quad (6)$$

This parameterization is better suited for interval methods as it avoids overestimation of the interval bounds due to each variable being represented only once in each equation. It was found that this formulation of the forward kinematics problem was not beneficial for the iterative methods over an Euler Angle representation used to generate rotation matrix **R** in Eqs. (1) and (2).

3.1 Neural Network

In the combination approach the pose of the platform is initially estimated by a set of neural networks. This greatly reduces the search space for the more time consuming interval algorithm. The neural network sets return a particular solution for the four reference point positions, whose error/uncertainty bounds define the initial search space of the interval algorithm.

The neural network was designed and implemented using the "Stuttgart Neural Network Simulator" (RSNNS) in the R statistical programming language [1]. The design is based on the multilayer perceptron (MLP), also used by Ghasemi [5]. In this case one neural network was used to determine the position (x, y, z) of a given reference point. Each neural network was built with four hidden layers containing 70 neurons each. Since the IPAnema cable robot has eight cables this results in a $8 \times 70 \times 70 \times 70 \times 70 \times 3$ MLP architecture.

The neural network was trained in a supervised approach with standard back-propagation, using default learning parameters of RSNNS. For a random pose the reference point positions and cable lengths are evaluated through the inverse kinematics (1). Then the artificial neural network calculates the reference point positions. The error residual sum of squares in the computed reference point positions is used to determine the weight adjustments of the individual neurons. One epoch repeats this process for every pose in a training set. A test set evaluates the performance of the neural network whose error is minimized over 800 epochs.

The full set of poses consists of 1,00,000 random poses in a workspace of $6 \times 5 \times 4$ m in x, y, z and rotations in the range of $\pm 20°$ about each axis. 40,000 poses were used as a training set and the remaining 60,000 as a test set. Feature normalization was also applied for the cable lengths in the stated workspace.

Graphs in Fig. 2 show the results of a single network training. The histogram shows the distribution of the final absolute error of the test set. The maximum error determines the bounds of initial search space for the interval algorithm, ± 0.02 m. This speeds up the computation time. While interval analysis is a complex iterative procedure, successive operations necessary for each neuron will only be evaluated once.

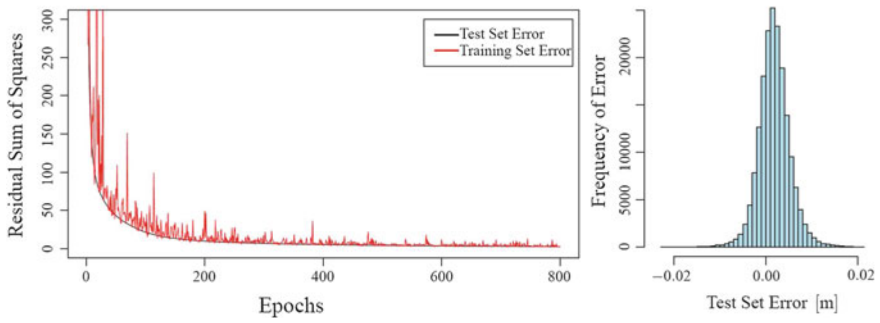


Fig. 2 Single neural network training results

3.2 Interval Algorithm

The interval algorithm based on [9] resembles a bisection method. The workspace is divided into successively smaller boxes which are then evaluated against the parametric Eqs. (4–6) with interval arithmetic. A box is a 12 dimensional hyper-cube of the (x, y, z) values of the four reference points. The starting box was based on the maximum error produced by the neural network test set. Interval arithmetic then determines whether a solution to the parametric equations exists within this box, does not exist, or cannot be determined. This branch and bound algorithm discards boxes containing no solution and continues to divide the rest, until a sufficiently small (specified by the desired accuracy) box without a solution is reached.

This is a deterministic and time consuming method to obtain a solution to the forward kinematics problem. Several techniques to speed up this process by reducing box sizes are implemented. One technique is evaluating for hull (or 2B) consistency as in constraint satisfaction problems as shown in [9]. Here the constraints of a single variable in the parametric equations are shrunk by those defined for the other variables. This can be repeated indefinitely, but tests showed that once the improvement was below 25 % it was more efficient to return to the branch and bound.

Another very effective method to reduce box size was the Interval-Gauss-Seidel method [11], which not only reduces the hyper-volume of a box, but also mitigates effects of solution clustering. Here the variable bounds are also shrunk individually using an iterative technique by applying the formula of a classic linear equation system to this variable.

These methods make the computation more efficient, but not to the same scale as the initial guess by the neural network.

Table 1 Computation time tested on IPAnema geometry

	Interval algorithm	Interval algorithm with neural network	Levenberg-Marquardt optimization [10]
Max. evaluation time [s]	0.764	0.016	N/A
Avg. evaluation time [s]	0.292	0.00121	0.000015
Max. no. of boxes/iterations	173356	303	5
Avg. no. of boxes/iterations	60690	77	4

4 Performance Evaluation

The algorithms for solving the forward kinematics are implemented in C++. This enables the use of SIMD instruction sets on an x64 architecture to provide a significant increase in speed for interval calculations and is discussed in detail by [6]. The processor used for testing was an Intel i7–2600 K with 3.40 GHz.

Table 1 shows the results of a quick comparative benchmark of three algorithms implemented under identical conditions. A random pose list of 1000 poses, with corresponding cable lengths was generated using Eq. (1). Poses were in a $4 \times 4 \times 3$ m box in the workspace and described by a rotation of $-10°$ to $10°$ for each Euler Angle. Each algorithm calculated every pose to an accuracy of 0.00001 m.

The computational speed of the Interval Algorithm is greatly increased through the use of neural networks. The average computation time decreased by more than a factor of 200, the number of boxes even more. This makes the approach more real-time feasible.

Maximum time to solution could not be evaluated for the LM optimization as it was too fast for the timer resolution, so the average was taken from the total time over 1,000 iterations. Keeping in mind that the initial pose estimate is tailored for this cable robot geometry, the optimization algorithm computes extremely fast.

Neural networks do need to be retaught for significant geometry changes, but have no limitations as to the type of geometry to solve. Further, the interval algorithm still can provide certainty of the solution and guarantee convergence, which the optimization cannot.

5 Conclusions

It was shown that combining neural networks with the standard implementation of interval arithmetic to solve the forward kinematics of cable robots provides significant decrease in computation time. This enables a more real-time feasible implementation, while retaining strengths of the interval methods. While the speed does not surpass that of highly optimized LM iterative solutions, it provides a method with guaranteed convergence, and the possibility of finding numerous poses. It is a viable alternative for actual real-time controllers.

Improvements can still be made. The neural network training algorithm could be optimized to provide a more accurate initial guess. Several steps in the interval algorithm could be taken to optimize the switching between the branch and bound, consistency checks, and the Interval-Gauss-Seidel method. This is difficult to optimize for the general case.

The teaching of neural networks is still an issue, as it diminishes the ease of changing geometric configuration. However, slight configuration changes still enable the neural network to converge enough for the interval algorithm to run at acceptable speeds. Further, the only requirement for teaching the networks is a working inverse kinematic implementation. This process can be heavily automated.

Problems of kinematic sensitivity are not addressed by either of these algorithms, but are still subject of ongoing research.

Acknowledgments This work was partially supported by the Fraunhofer-Gesellschaft Internal Programs under Grant No. WISA 823 244. Furthermore, the research leading to these results received founding for the European Community's Seventh Framework Program under Grant agreement number NMP2-SL-2011-285404-CableBot.

References

1. Bergmeir, C., Benítez, J.M.: Neural networks in R using the Stuttgart neural network simulator: RSNNS. J. Stat. Softw. **46**(7), 1–26 (2012). http://www.jstatsoft.org/v46/i07/
2. Berti, A., Merlet, J.P., Carricato, M.: Solving the direct geometrico-static problem of 3–3 cable-driven parallel robots by interval analysis: preliminary results. In: Bruckmann, T., Pott, A. (eds.) Cable-Driven Parallel Robots. vol. 12, pp. 251–268. Springer, Heidelberg (2013)
3. Bosscher, P., Williams II, R.L.: Cable-suspended robotic contour crafting system. Autom. Constr. **17**(1), 45–55 (2007)
4. Carricato, M., Abbasnejad, G.: Direct geometrico-static analysis of under-constrained cable-driven parallel robots with 4 cables. In: Bruckmann, T., Pott, A. (eds.) Cable-Driven Parallel Robots. vol.12, pp. 269–285 (2013)
5. Ghasemi, A., Eghtesad, M., Farid, M.: Neural network solution for forward kinematics problem of cable robots. J. Intell. Rob. Syst. **60**(2), 201–215 (2010)
6. Goualard, F.: Fast and correct SIMD algorithms for interval arithmetic. In: Proceedings of PARA '08, Lecture Notes in Computer Science. Springer, Trondheim (2010)
7. Husty, M.L.: An algorithm for solving the direct kinematic of stewart-gough-type platforms. Mech. Mach. Theory **31**(4), 365–380 (1996)
8. Khalilpour, S., Loloei, A., Taghirad, H., Masouleh, M.: Feasible kinematic sensitivity in cable robots based on interval analysis. In: Bruckmann, T., Pott, A. (eds.) Cable-Driven Parallel Robots, vol. 12, pp. 233–249. Springer, Heidelberg (2013)
9. Merlet, J.P.: Solving the forward kinematics of a gough-type parallel manipulator with interval analysis. Int. J. Robot. Res. **23**(3), 221–235 (2004)
10. Pott, A.: An algorithm for real-time forward kinematics of cable-driven parallel robots. International symposium on advances in robot kinematics, Springer (2010)
11. Shary, S.: Interval Gauss-Seidel method for generalized solution sets to interval linear systems. Reliable Comput. **7**, 141–155 (2001)
12. Won Jeong, J., Hyun Kim, S., Keun Kwak, Y.: Kinematics and workspace analysis of a parallel wire mechanism for measuring a robot pose. Mech. Mach. Theory **34**(6), 825–841 (1999)