Cable-Driven Parallel Robot for Rapid Manufacuring

Wendy Lam
Robotics Engineering Student
Worcester Polytechnic Institute
Worcester, Massachusetts
wlam@wpi.edu

Joseph Lombardi
Robotics Engineering Student
Worcester Polytechnic Institute
Worcester, Massachusetts
jplombardi@wpi.edu

***Abstract*—Cable-driven parallel robots are a potential solution to automated manufacturing of buildings. This paper presents the kinematic equations and algorithms necessary for an arbitrary 8-wire robot to follow motion commands.**

Keywords—cable driven parallel robot, rapid manufacturing, 3d printing, concrete, kinematics, calibration.

I. INTRODUCTION

Cable Driven Parallel Robots (CDPR) are lightweight, versatile, and have benefits that let them compete with gantry-style and serial-arm manipulators in the field of autonomous building construction. A CDPR is a mechanism with an end-effector that is oriented and translated through tensioning flexible cables, rather than rotating solid links of a classic industrial robotic arm. Because CDPRs do not have bulky links, they can move the end effector rapidly to positions within the workspace. Moreover, they are relatively simple to manufacture, have a wide workspace, and may be taken down and reconfigured for different locations [2]. These properties all lend themselves well towards a machine that can be sent to remote areas of undeveloped land to construct rigid, permanent structures for human habitation. The ratio of weight-to-workspace makes CDPRs a possible contender for autonomous construction on other planets.

Prior research has been performed in 3d-printing buildings using clay and concrete [1][2][3]. Additionally, CDPRs have been developed that can lay bricks in a defined pattern using a grasper. Team Cable Bot proposed combining the above capabilities by laying bricks and then spreading mortar to construct buildings in a fashion similar to traditional construction. The team expected to reduce weight by reducing the manipulator size, and time to construct a building by reducing the end-effector weight, thus increasing the speed at which the manipulator can move.

Team Cable Bot aimed to develop a miniature model that showcased a possible opportunity for future CDPR development. The team investigated a specific model of CDPR, analyzed the kinematics of the robot, and provided models for simulation with the goal of following a motion profile. Given bricks and mortar, the CDPR that Team Cable Bot designed and simulated was able to follow a trajectory to lay the foundation of an arbitrary building plan up to the second layer of brick.

II. METHODOLOGY

For reference throughout the paper, the methods the project team utilized are detailed herein.

A. Model Geometry

The size of the CDPR structure was determined to be the size of two Ultimaker3 side by side. The 3D printer was determined due to prior experience of using the 3D printers and knowing that they give ample room. The structure would be 660 x 480 x 300 mm. The end effector was modeled after the 3D printer extruder, which came to be 60 x 46 x 41 mm. In addition, for any realistic construction of the machine, an aluminum extrusion with a size of 20 x 20 x 300 mm was modeled to hold up the pulleys. The pulleys used in the 3D model do not have a set size.

In the initial modeling for the simulation, shown in Figure 1, a pulley is placed at the top and bottom of each beam, represented as a teal color circle, and at each corner of the end effector. Due to the pulley multibody block being used, there is an orange cable that could be seen wrapping around the pulley on to the corners of the end effector. The initial modeling did not have the ability to process inputs and feedbacks. Thus, the pulley multibody block was later changed to be a belt-cable spool multibody block.

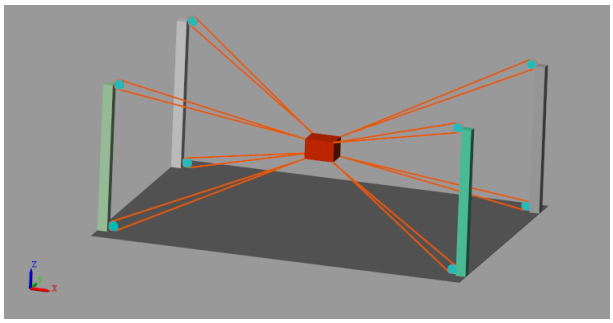

Figure 1 – Initial iteration of the simulation in MATLAB Simscape

In the final modeling for the simulation, shown in Figure 2, there are no size changes to the over all structure. The major changes were finalizing the joints at each pulley as well as the corners of the end effector.

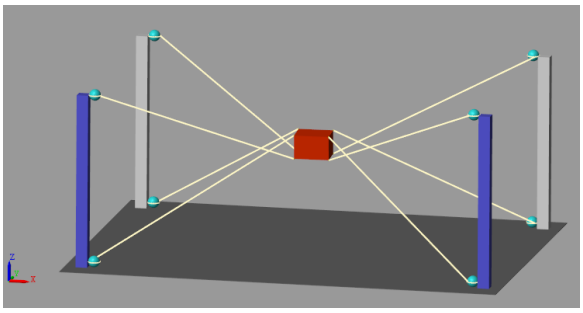

Figure 2 - Final iteration of the simulation in MATLAB Simscape

The overall model-based system of the simulation is shown in Figure 3. There are 4 major subsystems for the model: MATLAB Code (shown in Figure 4), Joint Connections (shown in Figure 5), Cable Controller (shown in Figure 6), and Ground (shown in Figure 7). MATLAB Code subsystem contains all the necessary inputs into the MATLAB function cdprlK, which would calculate the inverse kinematics of the robot, and the output l, which contains the calculated cable length for the robot. The Joint Connection subsystem contains all the necessary blocks to control each pulley on each beam, which would either pull or release the cable according to the MATLAB code. The Cable Controller subsystem takes the MATLAB code and makes the necessary changes to the system using a PID controller. Additionally, the Cable Controller also houses the joints for each pulley on the beam as well as the end effector corners. The joints for the pulleys on the beam have a spherical joint, prismatic joint, and a revolute joint. The spherical and prismatic joints give the pulleys the flexibility to reel in the spool and turn in the direction of where the spool is being pulled to. The revolute joint is required for the spool because it requires the specified shape to reel back the cable. The corners of the end effector are spherical joints because it needs to have the flexibility to turn to wherever it is being pulled or released. The ground subsystem includes the World Frame, Mechanism Configuration, and Solver Configuration which is necessary to simulate a system. The ground solid shape is also housed in this subsystem for organization.

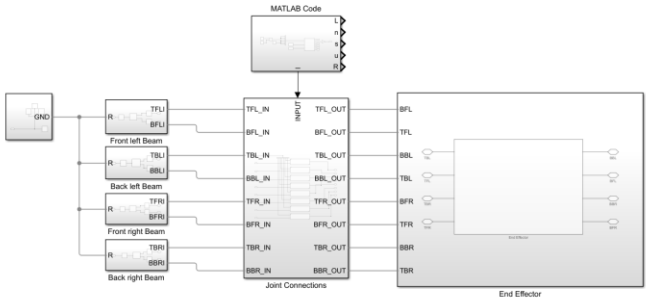

Figure 3 – Model Based System for CDPR

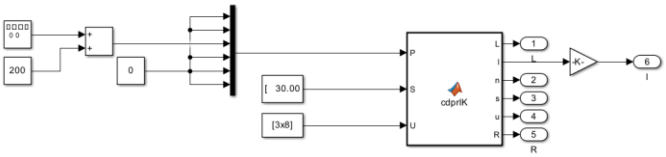

Figure 4 - MATLAB Code Subsystem for CDPR

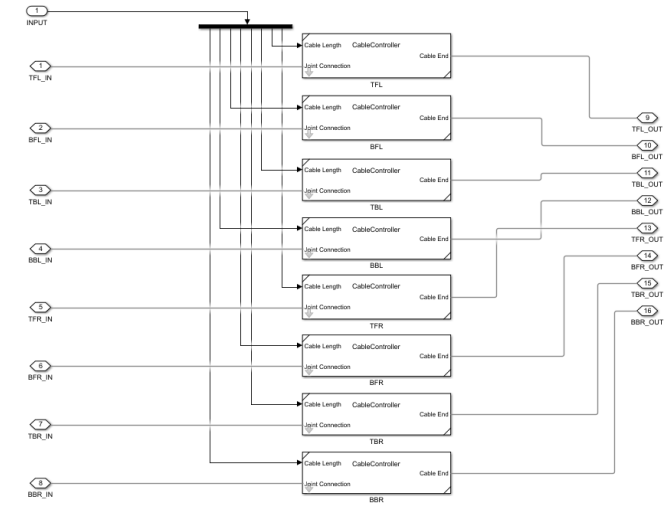

Figure 5 – Joint Connections Subsystem for CDPR

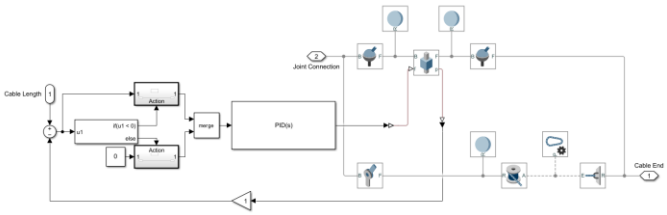

Figure 6 – CableController Subsystem for CDPR

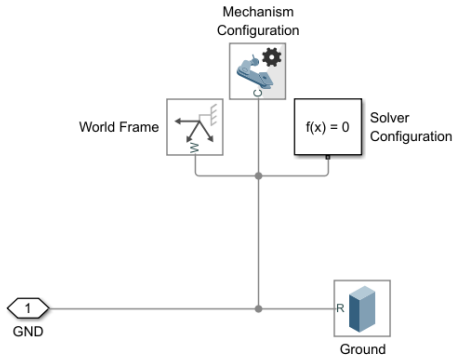

Figure 7 – Ground Subsystem for CDPR

B. *Inverse Kinematics*

The inverse kinematics for a CDPR take the form of most other parallel manipulators. Using a pose, P, composed of a vector O representing the location of the end-effector (EE) and 3 Euler angles, one can use geometric data from the CDPR configuration to directly calculate the cable vectors [4]. The geometric data takes the form of the cable start points on the frame, and the cable end points on the end effector, notated U and S respectively. The points are represented as vectors originating from the base coordinate frame for the U points, and the end-effector frame for the S points.

The inverse kinematics can now be computed as a vector loop closure, with the i^{th} cable of the CDPR noted as l_i, according to the equation

$$U_i - O - RS_i - l_i = 0 \quad (1)$$

where O is the position vector of the pose, R is the rotation matrix of the pose, and U_i and S_i are the connection points of the i^{th} cable, also represented in Figure 8 below.

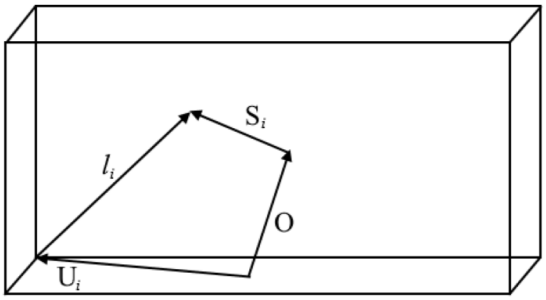

Figure 8 – Graphical representation of inverse kinematics

C. Velocity Kinematics

The velocity Jacobian, J_v, is necessary for calculating the relationship between the velocity of a pose, both linear and angular, and the rate of change of the cable lengths. The forward Jacobian is calculated according to formula (2)

$$J_{vi} = [n_i^T [RS_i \times n_i]^T] \quad (2)$$

with n_i representing the unit vector of each cable, R the rotation matrix, x the cross product, and J_{vi} as the velocity Jacobian of the i^{th} leg. The full representation of J_v is composed of (2) with each row calculated with i = [1,8] representing each cable. Using the information obtained during the inverse kinematics calculation is especially helpful for the velocity Jacobian.

Since the CDPR is this study has 8 cables, the J_v matrix is 8x6, not inherently invertible, and thus the inverse velocity kinematics from cable length velocity to pose velocity is not readily achievable. Other methods are available, such as the right pseudoinverse, but these methods were not necessary for the study at hand.

D. Forward Kinematics

Forward kinematics for many parallel mechanisms require work above and beyond that required for serial mechanisms. Since CDPRs are over determined, having more cables than degrees of freedom, CDPRs are no different. The forward kinematics for CDPRs have been discussed in works such as [5], and methods commonly used are non-linear least squares algorithms such as Levenberg-Marquardt. The team implemented a forward kinematics schema in MATLAB that uses the configuration of the CDPR and the current cable lengths, and a cost function defined by equation (3)

$$\Phi(l, O, R) = \Sigma[||U_i - O - RS_i||^2 - l_i^2] \quad (3)$$

to calculate the pose, with the summation ranging i = [1,8].

There are various optimizations that the team could implement, such as that in [5], with which a real-time algorithm can be implemented. Real-time algorithms utilize the same cost function as in (3), but the initial pose fed to the nonlinear least-squares algorithm is within a bounding box defined by the cable lengths. The real-time forward kinematics are valuable for realizations of CDPRs, as the computation time must be discrete for real-time controllers.

E. Calibration

Automatic calibration routines for CDPRs have been researched by Xue Jun Jin et al [6]. One method uses a laser to measure the distance between the end effector and a known plane, such as the base X-Y plane (considering Z to represent vertical distance). The difference between the expected measurement (d_{exp}) and the actual measurement (d_{mes}) is utilized in an error function according to the equation below.

$$\phi = d_{exp}^2 - d_{mes}^2 \quad (4)$$

Applying the non-linear least squares method over a range of poses with high observability provides best-fit kinematic parameters for the U and S points of a given CDPR. The calibrated results will not be exact, as the calibration process is a numerical method, but the calibrated kinematic parameters will reduce error across the CDPR's workspace. Refer to Figure 9 below.

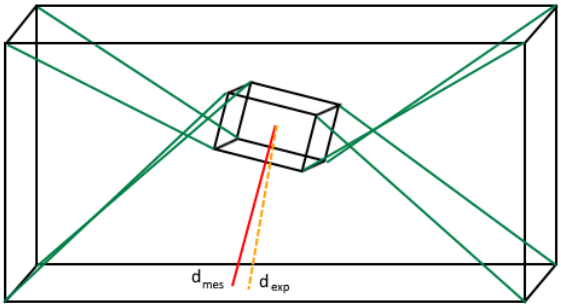

Figure 9 - Illustration of difference between d_{exp} and d_{mes}

F. Trajectory Generation

Trajectory generation is performed from point-to-point using cubic trajectory profiles. Given points P_0 and P_1, the algorithm utilizes the previously mentioned Inverse Kinematics of the CDPR to find the 8 cable lengths L_0 and L_1 corresponding to the two points P_0 and P_1 respectively, as in (5). Given the time allowed for the transition between the two points, a cubic motion profile for each of the cables l_i with i = [1,8] according to (6). Coefficients a0, a1, a2, and a3 for (6) were found using (7) with t_0 as the initial time, t_f as the final time, q_0 as the initial cable length, q_f as the final cable length, v_0 as the initial velocity, and v_f as the final velocity. Both v_0 and v_f were equal to zero for this project's purposes.

$$P_i \rightarrow IK \rightarrow l_i \quad (5)$$

$$L_i(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (6)$$

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 \\ 0 & 1 & 2t_0 & 3t_0 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix}^{-1} \begin{bmatrix} q_0 \\ v_0 \\ q_f \\ v_f \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (7)$$

An example of the trajectory generation algorithm is displayed in Figure 10, note the positions of the 8 cable lengths over time.

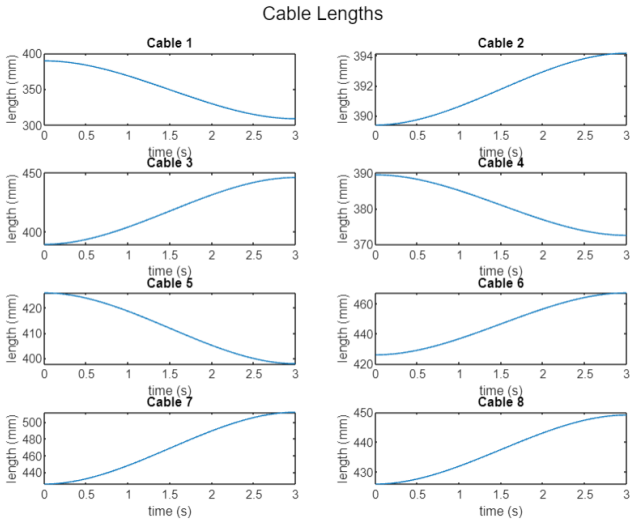

Figure 10– Cable lengths over time for cubic trajectory

The corresponding end-effector path is shown to follow the predicted line closely, as shown in Figure 11. Future iterations of the algorithm shall have the ability to create a smooth spline between points, and the option to carry velocity through points or stop at each point.

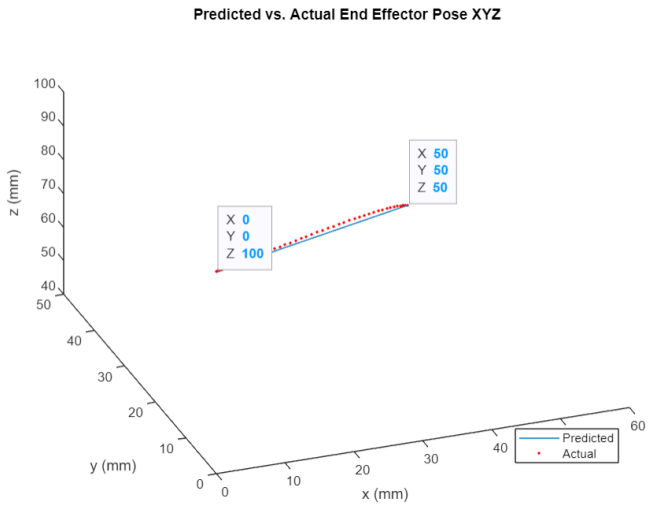

Figure 11– End-effector motion: predicted vs actual

III. Discussion and Results

The results for the project are detailed below are composed of the simulation and calibration.

A. Simulation

The simulation of the model following a trajectory generated was successful with some inconsistencies. The simulation followed a path set forth by the trajectory, however, it was very shaky and showed a lot of differences between the actual and the prediction. One of the possible reasons for the difference between the prediction and actual is due to how the joints are set up for the robot in the simulation. There are multiple joints to choose from in Simulink, however the team learned that other complex joints, such as 6 DOF joint, caused singularity in the model which prevented a simulation from happening. In order to avoid the singularity, the team had to breakdown the complexity of the joint to a spherical joint and a prismatic joint, which also simplified the inputs and outputs for the system. By doing that, there are a lot of inconsistencies from joint to joint. Another possible reason for inconsistencies could be found in where the pulleys were located on the beam. To simplfy the model simulated, the team decided to attach the pulley directly onto one side of the beam and have the cable connect to the location to the corner of the corresponding end effector. This would also cause some differences, which adds to the inconsitencies caused by the joints. In order to make the model more appropriate, the team believes that the two types of joints used should be combined to a 6 DOF joint and more time to explore possible limitations of the spool on the pulley system and how to utlize properly.

B. Calibration

The calibration algorithm as-is yields unsatisfactory results. The team was not able to replicate the success in paper [6]. The team believes that part of this is due to user error, but that another issue is that a single-dimensional calibration measure for a 6DOF robot is not ideal. One issue that arises is that rotation of the end-effector throws off results dramatically. Rotation of the end-effector about its Z axis does not afect the laser measurement, but greatly affects the length of the 8 cables. Alternatively, if the CDPR is utilized as a cartesian mechanism with only 3DOF corresponding to XYZ movement, then the calibration algorithm yields kinematic parameters that satisfy the goal of the simulation. The team believes that adding one or more lasers to the end effector for calibration would yield more closely correlated kinematic parameters.

IV. Conclusion

Overall, the team considers that they attempted too much in too little time. While the schedule leading up to the team's progress review was conservative and met without issue, the team was unable to finish the objectives of the latter parts of the project, namely the manipulator and physical realization. The team believes that this is due to a skill and time gap that the members were unable to bridge.

Despite not realizing all of the desired functionality, the team is very pleased with the work accomplished. Had the team more time, the team is confident that all functionality would have been implemented. The team is satisfied with the inverse, forward and velocity kinematics, as well as the basic functionality of the calibration algorithm. The calibration algorithm in future iterations would be adjusted so as to provide more predictable sensitivity to end-effector orientation, by including more lasers.

The team considers that the project was a successful foray into CDPRs, and that the mechanism requires more research into simulation suites, and improved calibration procedures.

References

[1] Tho TP, Thinh NT. Using a Cable-Driven Parallel Robot with Applications in 3D Concrete Printing. Applied Sciences. 2021; 11(2):563. https://doi.org/10.3390/app11020563J.

[2] Izard, JB., Dubor, A., Hervé, PE. et al. Large-scale 3D printing with cable-driven parallel robots. Constr Robot 1, 69–76 (2017). https://doi.org/10.1007/s41693-017-0008-0K.

[3] D. Lau, J. Eden, Y. Tan and D. Oetomo, "CASPR: A comprehensive cable-robot analysis and simulation platform for the research of cable-driven parallel robots," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South), 2016, pp. 3004-3011, doi: 10.1109/IROS.2016.7759465.

[4] Pott, Andreas. (2010). An Algorithm for Real-Time Forward Kinematics of Cable-Driven Parallel Robots. ARK. 10.1007/978-90-481-9262-5_57.

[5] Pott, Andreas & Schmidt, Valentin. (2015). On the Forward Kinematics of Cable-Driven Parallel Robots. 10.1109/IROS.2015.7353818.

[6] Jin X, Jung J, Ko SY, Choi E, Park JO, Kim CS. Geometric Parameter Calibration for a Cable-Driven Parallel Robot Based on a Single One-Dimensional Laser Distance Sensor Measurement and Experimental Modeling. Sensors (Basel). 2018 Jul 23;18(7):2392. doi: 10.3390/s18072392. PMID: 30041466; PMCID: PMC60687