IDETC/CIE 2008
ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
August 3-6, 2008, New York, NY, USA

DETC2008-49480

CONTOUR-CRAFTING-CARTESIAN-CABLE ROBOT SYSTEM: DYNAMICS AND CONTROLLER DESIGN

Robert L. Williams II[1], Ming Xin[1], Paul Bosscher[2]
Department of Mechanical Engineering, Ohio University, Athens, Ohio 45701[1]
Government Communication Systems Division, Harris Corporation, Palm Bay, FL, 32905[2]

Email: williar4@ohio.edu, ming.xin.1@ohio.edu, pbossche@harris.com

ABSTRACT

This paper presents dynamics equations and controller simulation for the Contour-Crafting-Cartesian-Cable (C^4) Robot. The C^4 robot was previously introduced for large-scale contour crafting construction. The pseudostatic and dynamics equations are presented, including how to maintain positive cable tensions. A controller design is also proposed for the C^4 robot, based on the computed-torque method. MATLAB simulation is presented for controller simulation with different trajectories and controller gains.

KEYWORDS

Contour crafting, cable-suspended robot, C^4 robot, translation-only, dynamics, control, computed-torque control positive cable tensions.

1. INTRODUCTION

Khoshnevis ([1], [2]) proposes contour crafting (CC) for construction of single-family dwellings and other buildings. In [3] we proposed the translation-only Contour Crafting Cartesian Cable (C^4) robot to perform the Cartesian motions required in CC construction. In [4] we considered two C^4 robot design alternatives and chose the best design based on wrench-feasible workspace and translational and rotational stiffness. Figure 1 shows the preferred C^4 robot diagram from [4] and Figure 2 shows this C^4 robot in simulated CC construction.

The translation-only C^4 robot consists of a rigid frame and an end-effector suspended from twelve active cables, grouped into eight upper cables and four lower cables. The eight upper cables occur in four pairs of parallel cables. The pulleys for the upper cables are mounted on horizontal crossbars, oriented at 45º with respect to the frame members, where the width of each crossbar is equal to the width of the corresponding side of the end-effector (also oriented at 45º for all motions).

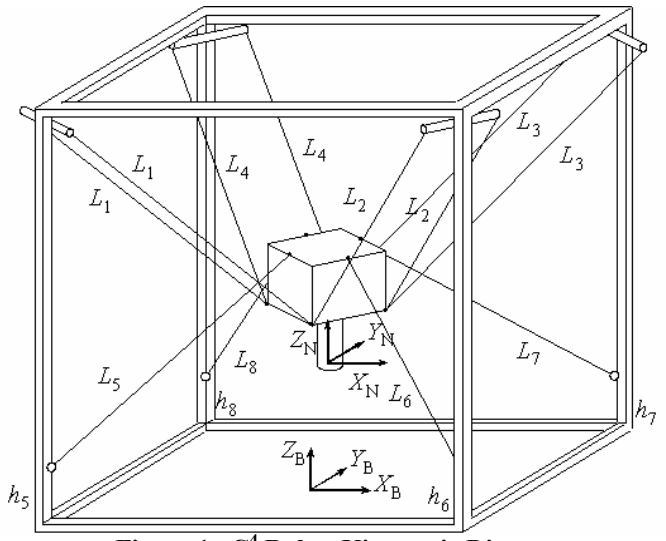

Figure 1. C^4 Robot Kinematic Diagram

The upper cable pairs support the end-effector weight and provide translation-only motion. For each pair of cables, the two cables are controlled to have the same length. A parallelogram is formed by each pair of cables and the corresponding crossbar and end-effector edge. By maintaining this parallelism, translation-only motion can be guaranteed [3]. This simplifies manipulator control and reduces the complexity of the forward kinematics solution.

To prevent cable interference with the building under construction, the four single lower pulleys are actuated vertically (heights h_i, i=5,6,7,8 are independently-controlled variables). The cables have been crossed (the lower cables connect to the end-effector top and the upper cables connect to the end-effector bottom), which leads to superior rotational stiffness [4] without cable interference.

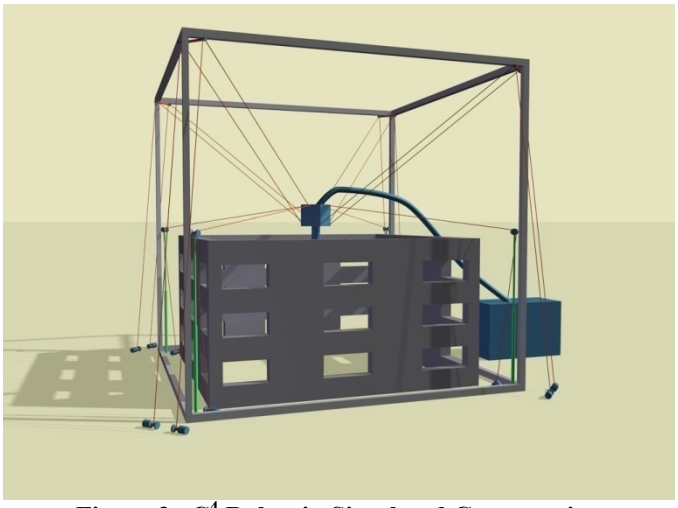

Figure 2. C^4 Robot in Simulated Construction

This paper presents the pseudostatics and dynamics equations for this C^4 robot, followed by controller design and simulation. All cable robot motions must be subject to a constraint of only positive cable tensions, which is presented also.

2. PSEUDOSTATICS AND DYNAMICS

We present the pseudostatics and dynamics equations of the C^4 robot of Figures 1 and 2 in this section. Then we discuss how to maintain positive cable tensions for all robot motions.

2.1 Pseudostatics Model

Before we derive the dynamics equation for the C^4 robot, we build a statics model which is required for maintaining positive cable tensions. Figure 3 shows the C^4 robot end-effector free-body diagram. For static equilibrium, the sum of all forces and moments (cable tensions and external wrench including weight) exerted on the end-effector must be zero. Although we use the virtual cables concept (VCC, [3]) to simplify the translational-only kinematics problem motion for the C^4 robot, VCC is not fit for solving the statics or dynamics problems, since pairs of cables in general take different tensions for static and dynamics balance.

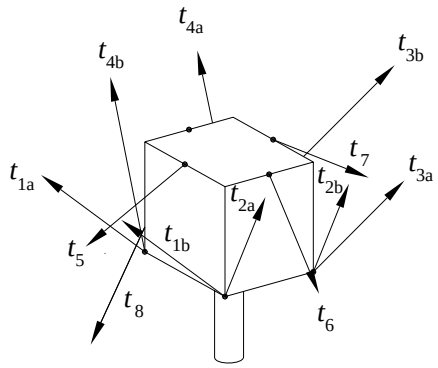

Figure 3. C^4 End-Effector Free-body Diagram

From Figure 3, we can write the force (1) and moment (2) statics equilibrium equations:

$$\sum_{i=1}^{4}(\mathbf{t}_{ia}+\mathbf{t}_{ib})+\sum_{j=5}^{8}\mathbf{t}_j+m\mathbf{g}+\mathbf{F}_R=\mathbf{0} \quad (1)$$

$$\sum_{i=1}^{4}(\mathbf{p}_{ia}\times\mathbf{t}_{ia}+\mathbf{p}_{ib}\times\mathbf{t}_{ib})+\sum_{j=5}^{8}(\mathbf{p}_j\times\mathbf{t}_j)+{}^P\mathbf{P}_{CG}\times m\mathbf{g}+\mathbf{M}_R=\mathbf{0} \quad (2)$$

The end-effector mass is m, and the gravity vector is $\mathbf{g} = [0, 0, -g]^T$. ${}^P\mathbf{P}_{CG}$ is the position vector from the origin of $\{N\}$ to the end-effector center of gravity; this may be a variable due to concrete sloshing and extrusion. In (1) and (2), $\mathbf{t}_k = t_k\hat{\mathbf{L}}_k$ are the cable tension vectors, where t_k is the k^{th} cable tension and $\hat{\mathbf{L}}_k$ is the k^{th} cable direction unit vector, pointing away from the end-effector to indicate positive direction for each cable tension. The k indices are in two groups, ia and ib, i=1,2,3,4 for the upper four pairs of cables, and j=5,6,7,8 for the lower four single cables. Moment arm vectors $\mathbf{p}_k$ are from the origin of $\{N\}$ to the appropriate cable connection point. Although the C^4 robot has twelve cables, there are only eight $\mathbf{p}_k$ vectors because the eight upper cables share four connection points on the bottom plane of the end-effector. $\mathbf{F}_R$ and $\mathbf{M}_R$ are the resultant force and moment vectors of the environment acting on the end-effector. Taken together, this is the external wrench $\mathbf{W}_R$ which is generally zero except in the case of unwanted disturbances. The overall statics equations are:

$$\mathbf{JT}=\begin{Bmatrix}-\mathbf{F}_R-m\mathbf{g}\\ -\mathbf{M}_R-{}^P\mathbf{P}_{CG}\times m\mathbf{g}\end{Bmatrix} \quad (3)$$

where $\mathbf{J}$ is the 6x12 Jacobian matrix (expressed in the base frame $\{B\}$):

$$\mathbf{J}=\begin{bmatrix}\hat{\mathbf{L}}_{1a} & \hat{\mathbf{L}}_{1b} & \cdots & \hat{\mathbf{L}}_{4a} & \hat{\mathbf{L}}_{4b} & \hat{\mathbf{L}}_5 & \cdots & \hat{\mathbf{L}}_8\\ \mathbf{p}_{1a}\times\hat{\mathbf{L}}_{1a} & \mathbf{p}_{1b}\times\hat{\mathbf{L}}_{1b} & \cdots & \mathbf{p}_{4a}\times\hat{\mathbf{L}}_{4a} & \mathbf{p}_{4b}\times\hat{\mathbf{L}}_{4b} & \mathbf{p}_5\times\hat{\mathbf{L}}_5 & \cdots & \mathbf{p}_8\times\hat{\mathbf{L}}_8\end{bmatrix} \quad (4)$$

and $\mathbf{T}=\{t_{1a}\ \ t_{1b}\ \ \cdots\ \ t_{4a}\ \ t_{4b}\ \ t_5\ \ \cdots\ \ t_8\}^T$ is the 12x1 vector of cable tensions.

Generally the C4 robot cable tensions cannot be controlled directly; instead the actuator torques are the control variables. In this paper twelve actuator torques are used to independently control the twelve cable tensions. The twelve actuators are divided into two groups: the top eight actuators control four pairs of parallel cables (pairs of cables do not always have the same tensions as their partner cable) and the bottom four actuators control the tensions of four single cables.

We assume that the twelve actuator shaft cable pulley reels all have identical radius r. Then the pseudostatics equations using actuator torques in place of cable tensions are simply:

$$\frac{1}{r}\mathbf{J}\boldsymbol{\tau}=\begin{Bmatrix}-\mathbf{F}_R-m\mathbf{g}\\ -\mathbf{M}_R-{}^P\mathbf{P}_{CG}\times m\mathbf{g}\end{Bmatrix} \quad (5)$$

where $\boldsymbol{\tau}=\{\tau_{1a}\ \ \tau_{1b}\ \ \cdots\ \ \tau_{4a}\ \ \tau_{4b}\ \ \tau_5\ \ \cdots\ \ \tau_8\}^T$ is the 12x1 vector of actuator torques.

2.2 Maintaining Positive Cable Tensions

Because the C^4 robot uses cables to move the end-effector, how to maintain positive cable tensions is important in the C^4 robot operation. In this paper, we propose three solutions to solve this problem: particular & homogeneous solution, `lsqnonneg` solution, and specified-value solution.

To evaluate the effect of all three solutions in this section, we use MATLAB simulation. The specified trajectory is a circle with the center at the point [0, 0, 5], and radius 15 m. The height of the four pulleys is 3 m, shown in Figure 4. The end-effector cube side length is 1m, the cube base frame side is 50m, the horizontal crossbar is b = 1m, the end-effector weight is 1000N, no additional end-effector wrench is considered, and the maximum allowable cable tension is 10kN.

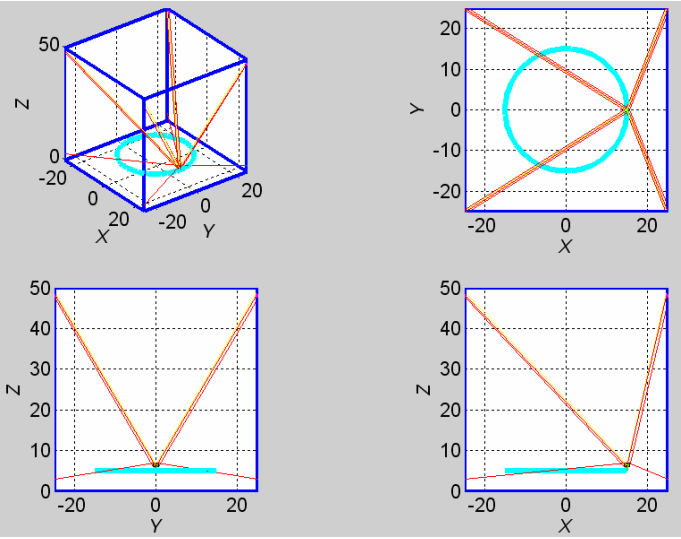

Figure 4. CC Circular Trajectory Simulation

2.2.1 Particular & Homogeneous Solution

The C^4 robot has considerable actuation redundancy to guarantee its manipulator translation-only motion in the workspace. However, because the Jacobian matrix is a 6x12 matrix and the tension (or torque) vector is 12x1, so equations (3) (or (5)) are 6 scalar equations with 12 unknowns, with infinite solutions. To find the solutions (demonstrated for 12 tensions throughout this section), we first adapt the well-known particular and homogeneous solution from resolved rate control of kinematically-redundant serial manipulators:

$$\mathbf{T} = [\mathbf{J}]^{+} \begin{Bmatrix} -\mathbf{F}_R - m\mathbf{g} \\ -\mathbf{M}_R - {}^{P}\mathbf{P}_{CG} \times m\mathbf{g} \end{Bmatrix} + \left([\mathbf{I}_{12}] - [\mathbf{J}]^{+}[\mathbf{J}] \right) \{\mathbf{w}\} \qquad (6)$$

The particular solution is the first term of (6) and the homogeneous solution is the second term of (6). $[\mathbf{I}_{12}]$ is the 12x12 identity matrix and vector $\{\mathbf{w}\}$ is an arbitrary vector which was chosen by trial and error to ensure all twelve cable tensions are positive for all motions. $[\mathbf{J}]^{+}$ is the underconstrained Moore-Penrose pseudoinverse of the 6x12 underconstrained statics Jacobian matrix:

$$[\mathbf{J}]^{+} = [\mathbf{J}]^{T} \left([\mathbf{J}][\mathbf{J}]^{T} \right)^{-1} \qquad (7)$$

MATLAB function `pinv` performs this calculation. First we present the particular-only solution. The results (Figure 5) display some negative tensions, rendering this particular-only solution useless. We note that only lower cables have negative tensions, and the upper cables never do, since they are loaded by gravity. Further, t_5 and t_8, t_6 and t_7, and t_{2a} and t_{2b} are symmetrical to each other about the vertical axis. Also, t_{1a} and t_{1b}, t_{3a} and t_{3b}, and t_{4a} and t_{4b} are identical to each other.

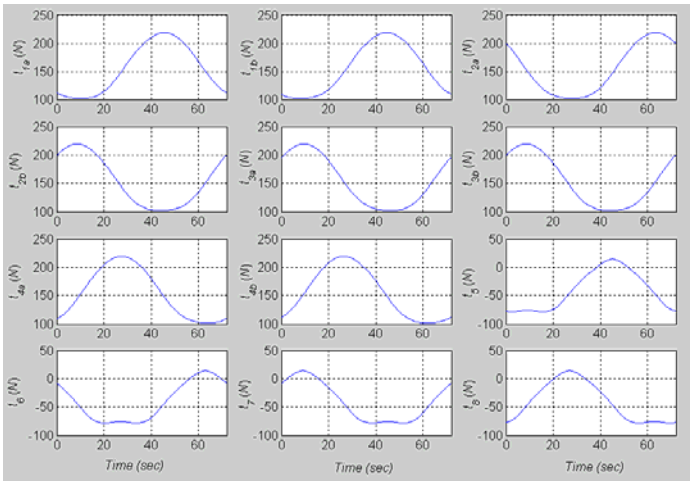

Figure 5. Cable Tensions Particular Solution

Since the particular-only solution fails, we now present an example combining the particular and homogeneous solutions. Vector $\{\mathbf{w}\}$ in (6) is chosen to compensate for the negative particular cable tensions, as shown in the results of Figure 6.

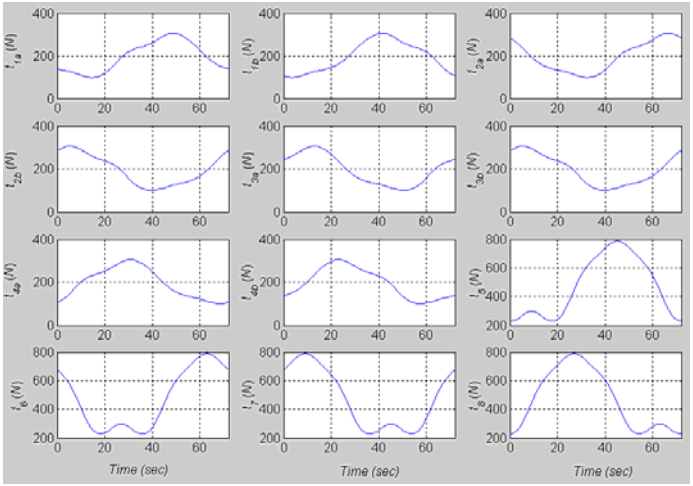

Figure 6. Particular & Homogeneous Solution

All cable tensions are positive in Figure 6. However, there is a height limit: at pulley height h = 40 m (see Figure 1) and z = 46 m the low limit of $\mathbf{w}$ should be 500 N. Above this height

limit, no matter how we increase {**w**}, some cable tensions are always negative, verifying the workspace results in [3].

2.2.2 *lsqnonneg* Solution

Here the MATLAB function `lsqnonneg` (linear least squares with non-negative result) is applied in attempt to maintain positive cable tensions. But the results (Figure 7) still show many negative tensions and the results are not smooth; therefore, this MATLAB function cannot solve the problem.

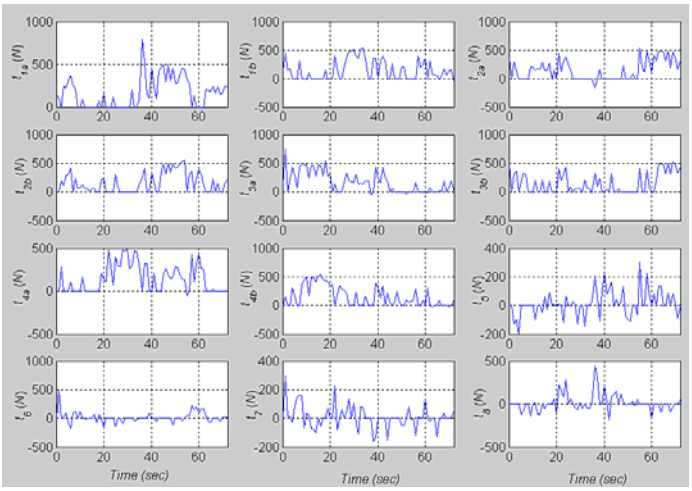

Figure 7. Positive Cable Tensions using *lsqnonneg*

2.2.3 Specified-Values Solution

In Figure 5, the negative cable tensions only appear in the lower C^4 robot cables. To compensate, we can specify positive values for these cables tensions and then use equation (3) to find the values of other tensions [6]. We specify the four lower cable tensions as given positive values for cable tensions t_5 through t_8 and extract their corresponding Jacobian column vectors from **J** to form Jacobian submatrix $[\mathbf{J}_{58}]_{6\times4}$ and separate the rest of columns (for the upper cable pairs) in **J** to form another submatrix $[\mathbf{J}_{14}]_{6\times8}$. The solution for the upper cable tensions t_{1a} through t_{4b} is then:

$$\mathbf{T}_{14} = [\mathbf{J}_{14}]^{+}\left(\begin{Bmatrix} -\mathbf{F}_R - m\mathbf{g} \\ -\mathbf{M}_R - {}^{P}\mathbf{P}_{CG}\times m\mathbf{g} \end{Bmatrix} - [\mathbf{J}_{58}]\begin{Bmatrix} t_5 \\ t_6 \\ t_7 \\ t_8 \end{Bmatrix}\right) \tag{8}$$

where $\mathbf{T}_{14} = \{t_{1a} \quad t_{1b} \quad t_{2a} \quad t_{2b} \quad t_{3a} \quad t_{3b} \quad t_{4a} \quad t_{4b}\}^T$.

However, upon implementation, this method did not work, i.e. some elements of $\mathbf{T}_{14}$ were still negative for some portions of motion. This is due to the crossed cables, which is preferable for rotational stiffness [4], but not for the specified values-solution. Therefore, the particular & homogeneous approach is favored for this robot.

2.3 Dynamics Equations

The C^4 robot dynamics equations with cable tensions are:

$$\mathbf{JT} + \begin{Bmatrix} \mathbf{F}_R + m\mathbf{g} \\ \mathbf{M}_R + {}^{P}\mathbf{P}_{CG}\times m\mathbf{g} \end{Bmatrix} = \begin{bmatrix} \mathbf{m} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}\ddot{\mathbf{X}} \tag{9}$$

Most terms, including Jacobian matrix **J** from (4), were introduced in the pseudostatics equations. The new terms for dynamics are end-effector mass matrix **m**, end-effector principal inertia tensor **I**, and end-effector acceleration $\ddot{\mathbf{X}}$:

$$\mathbf{m} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \qquad \mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{10}$$

$$\ddot{\mathbf{X}} = \{\ddot{x} \quad \ddot{y} \quad \ddot{z} \quad \dot{\omega}_x \quad \dot{\omega}_y \quad \dot{\omega}_z\}^T \tag{11}$$

The C^4 robot dynamics equations with actuator torques are:

$$\frac{1}{r}\mathbf{J}\boldsymbol{\tau} + \begin{Bmatrix} \mathbf{F}_R + m\mathbf{g} \\ \mathbf{M}_R + {}^{P}\mathbf{P}_{CG}\times m\mathbf{g} \end{Bmatrix} = \begin{bmatrix} \mathbf{m} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}\ddot{\mathbf{X}} \tag{12}$$

Equation (12) is used in the Controller Design section.

3 CONTROLLER DESIGN

It is important to design a suitable controller for the C^4 robot to enable its end-effector to follow specified CC trajectories and resist disturbances. In this section, we will focus on the controller design for the C^4 robot system. The C^4 robot is a non-linear system because its system dynamics are functions of the manipulator positions. Thus, the normal linear control cannot meet the control request and we select the classical computed-torque control [7] combined with PD controller to control the C^4 robot.

The computed-torque control method is a nonlinear control method that is popular for robots [7]. Computed-torque is a feedback linearization method applied to a non-linear system. Now we present the computed-torque method equations. The robot dynamics equations are:

$$\mathbf{M(q)}\ddot{\mathbf{q}} + \mathbf{V(q,\dot{q})} + \mathbf{G(q)} + \mathbf{f_{dist}(q,\dot{q})} = \boldsymbol{\tau} \tag{13}$$

Here, the $\mathbf{M(q)}$ is the manipulator inertia matrix, $\mathbf{V(q,\dot{q})}$ is the vector of centripetal and Coriolis torques, $\mathbf{G(q)}$ is the vector of gravitational torques, $\mathbf{f_{dist}(q,\dot{q})}$ is the vector of viscous and Coulomb friction torques, and $\boldsymbol{\tau}$ is the vector of control torques. The desired trajectory is defined as:

$$\mathbf{Y_d} = \mathbf{q_d(t)} \tag{14}$$

The actual path is:

$$\mathbf{Y_r} = \mathbf{q_r(t)} \tag{15}$$

The resulting errors are:

$$\mathbf{e} = \mathbf{q_d} - \mathbf{q_r} \tag{16}$$

$$\dot{\mathbf{e}} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}}_r \tag{17}$$

The computed-torque control law is:

$$\mathbf{M}(\mathbf{q})(\ddot{\mathbf{q}}_d + \mathbf{k}_v\dot{\mathbf{e}} + \mathbf{k}_p\mathbf{e}) + \mathbf{V}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathbf{f}_{dist}(\dot{\mathbf{q}}) = \boldsymbol{\tau} \tag{18}$$

Where, $\mathbf{k}_p, \mathbf{k}_v$ are the diagonal controller gain matrices.

3.1 Proposed Controller Algorithm

Before giving the corresponding algorithm, we need to match the dynamics equations (12) to equations (13):

$$\boldsymbol{\tau} = \mathbf{J}^+\left(r\begin{bmatrix}\mathbf{m} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\ddot{\mathbf{X}} - r\begin{Bmatrix}\mathbf{F}_R + m\mathbf{g}\\ \mathbf{M}_R + {}^P\mathbf{P}_{CG}\times m\mathbf{g}\end{Bmatrix}\right) \tag{19}$$

where again $\mathbf{J}^+$ is the pseudoinverse of the underconstrained 6x12 statics Jacobian matrix. Comparing equations (19) and (13), $\mathbf{V}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{0}$. Also, it is assumed that the system is frictionless, $\mathbf{f}_{dist}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{0}$, and the external wrench of the environment on the end-effector is zero. Therefore, the terms for Cartesian dynamics equations (13) are:

$$\mathbf{M}(\mathbf{q}) = \mathbf{J}^+ r\begin{bmatrix}\mathbf{m} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \tag{20}$$

$$\ddot{\mathbf{q}} = \ddot{\mathbf{X}} = \begin{Bmatrix}\ddot{x} & \ddot{y} & \ddot{z} & \dot{\omega}_x & \dot{\omega}_y & \dot{\omega}_z\end{Bmatrix}^T \tag{21}$$

$$\mathbf{G}(\mathbf{q}) = -\mathbf{J}^+ r\begin{Bmatrix}m\mathbf{g}\\ {}^P\mathbf{P}_{CG}\times m\mathbf{g}\end{Bmatrix} \tag{22}$$

$$\boldsymbol{\tau} = \begin{Bmatrix}\tau_{1a} & \tau_{1b} & \cdots & \tau_{4a} & \tau_{4b} & \tau_5 & \cdots & \tau_8\end{Bmatrix}^T \tag{23}$$

Equation (18) becomes:

$$\mathbf{M}(\mathbf{q})(\ddot{\mathbf{q}}_d + \mathbf{k}_v\dot{\mathbf{e}} + \mathbf{k}_p\mathbf{e}) = \boldsymbol{\tau} - \mathbf{G}(\mathbf{q}) \tag{24}$$

Here, $\mathbf{k}_v$ is the matrix gain of the D controller, and $\mathbf{k}_p$ is the matrix gain of the P controller. Because the manipulator moves translationally-only, the values for $\theta_x, \theta_y, \theta_z$ are always zero and their first order and second order differentials are also zeros. The computed-torque control architecture is shown in Figure 8. First, we specified a desired trajectory (14) in frame $\{B\}$ and its period for the CC process. Then, a time step is set and the period divided into a series of equal time segments. The position, velocity and acceleration at the end of each time segment is computed.

To simulate a trajectory, a starting point *x,y,z* is given. To correct the trajectory, the errors of the end-effector's displacement and velocity in CC process simulation must be solved by equations (16) and (17). Then MATLAB function `ODE45` is used to solve the end-effector displacement and velocity. To compare (x_r, y_r) and $(\dot{x}_r, \dot{y}_r)$ of the simulated trajectory with (x_d, y_d) and $(\dot{x}_d, \dot{y}_d)$ of the desired trajectory, the time step for both trajectories are identical. The errors are solved and applied in the computed-torque control law (19). The new corrected accelerations are the input for another `ODE45` block. After that, the new values of displacement and velocity become the corrected ones. Then, these new values are inputted as the initial value for the next iteration in a `for` loop until the time range ends.

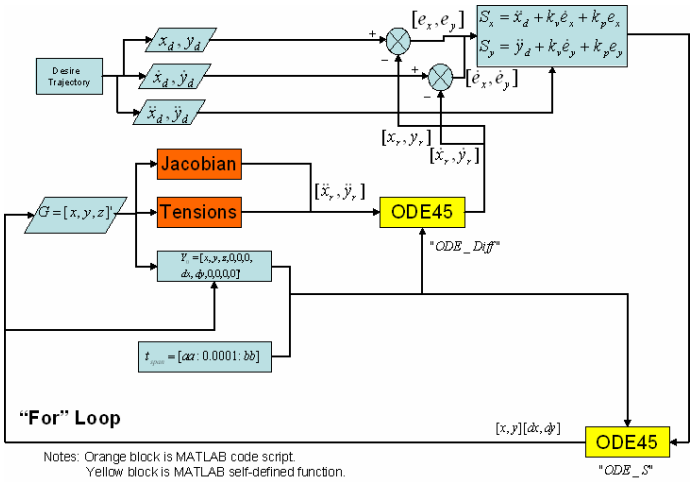

Figure 8. Computed-Torque Algorithm for the C[4] Robot

3.2 Controller Design MATLAB Simulation

The major problem in designing non-linear controllers is: can one set of gains be used for all motions in one plane at all heights? To address this problem, we simulate controller performance at different pulley and end-effector heights. We will follow the discretization method that we did for stiffness [4]. How to choose PD controller gains is another important problem. Based on Sections 2.3 and 3.1, the C[4] robot dynamic equations can be simplified to a second-order dynamic system. The standard form for the second-order characteristic polynomial is, demonstrating for the scalar case:

$$p(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 \tag{25}$$

Where ζ is the dimensionless damping ratio and ω_n is the natural frequency (rad/s). The controller gains are:

$$k_p = \omega_n^2 \tag{26}$$

$$k_v = 2\zeta\omega_n \tag{27}$$

To avoid undesirable collisions, the end-effector should not overshoot. Therefore, choose critical damping ($\zeta = 1$) in the design of the PD controller, leading to the relationship $k_v = 2\sqrt{k_p}$. Setting the natural frequency ω_n = 10 rad/s, the two gains are k_p = 100 and k_v = 20. In circular trajectory simulation, the starting point is (50/3, 0) and the circle center is (0, 0). $\mathbf{k}_p, \mathbf{k}_v$ are the diagonal controller gain matrices with k_p and k_v for the diagonal terms, respectively.

We tested the effectiveness of one set of controller gains in all end-effector heights; we noticed that all trajectory

simulation results are identical [5]. Therefore we will provide only a set of simulation diagrams (h = 0m and z = 3m) out of six sets of pulley and end-effector heights tested.

Although the control law and PD controller are proved to be effective in correcting the end-effector trajectory errors in the simulated CC process [5], the deflection of the path is still large in the first half period and the settling time for path correcting is still too large because of low gains [5]. Therefore, the gains of the PD controller were changed to improve both aspects. We applied trial-and-error to find new gains: the natural frequency was increased to ω_n = 50 rad/s so the new gains are k_p = 2500 and k_v = 100. The MATLAB simulation results are shown in Figure 9.

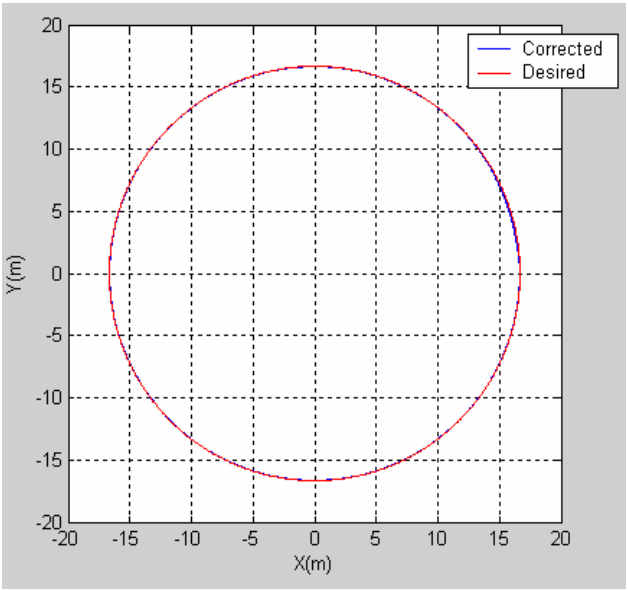

Figure 9. Controller Simulation at h = 0 m and z = 3m

In Figure 9, the controlled (corrected) path is almost following the desired path. It is obvious that the path shows the desired performance under the high gains control. The lower gains yielded unacceptable error performance and so these results are not shown.

Figure 10 shows the controller errors for the high gains. Though the initial error is large (~0.7 m), this is almost 10 times less error than the controller with low gains. The settling time also decreases significantly for the higher gains. The size of steady-state error is on the centimeter level (-0.026 m < error < 0.026 m), in a 50 m high building simulation, this is acceptable [1,2].

However, the initial error in Y is unacceptable, approaching 0.7 m for a brief time. Therefore, we increase the natural frequency to 250 Hz which yields new gains k_p = 62500 and k_v = 500. The new simulation is shown in Figure 11 (the desired and controlled (corrected) appear identical at this scale) and Figure 12 shows that the new errors are much smaller than those of Figure 10; the worst Y error is now about 2.5 cm.

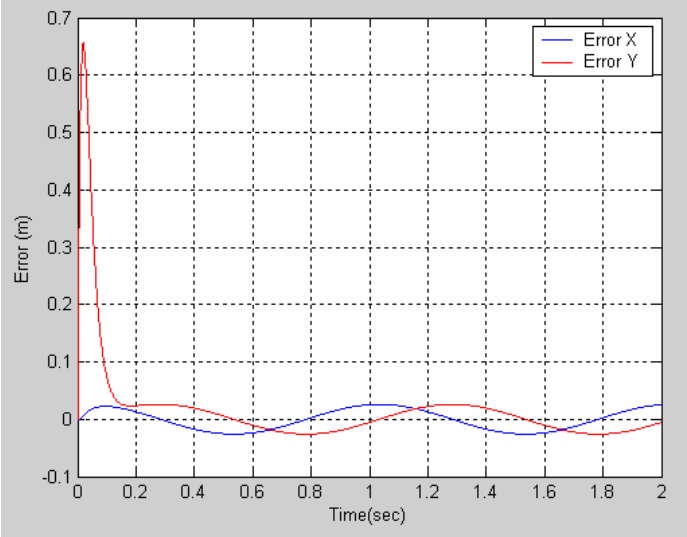

Figure 10. Position Errors for Figure 9

We ask if our controller can be used for all layers at all heights (pulley heights h = 0m, 15m and 30m; end-effector heights z = 3m, 18m and 33m). Six MATLAB controller simulation results were compared [5] and their performance was essentially identical for the higher gains controller simulation. Thus, we just show two of them (Figures 11 and 13) to demonstrate this point.

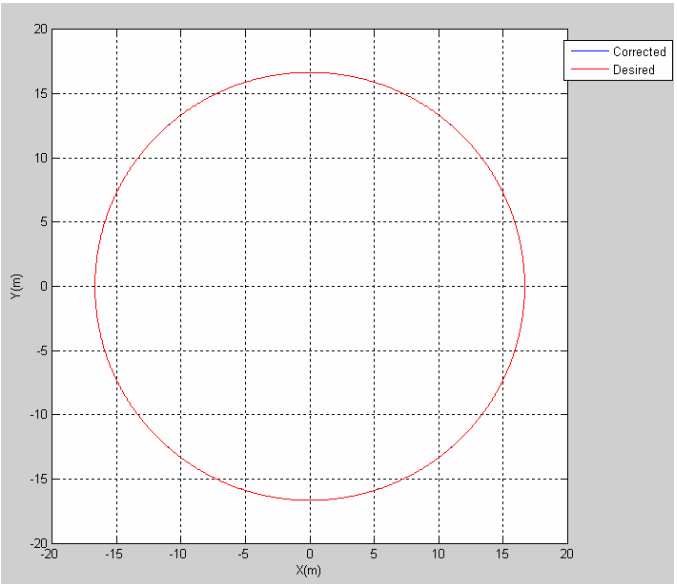

Figure 11. Higher Gains Simulation at h = 0 m and z = 3m

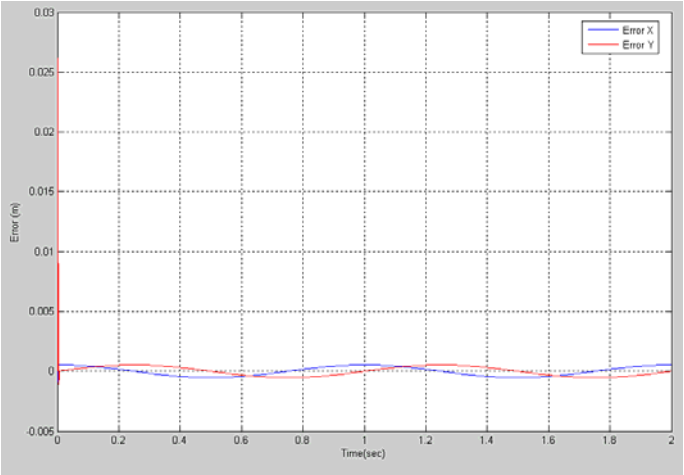

Figure 12. Position Errors for Figure 11 (higher gains)

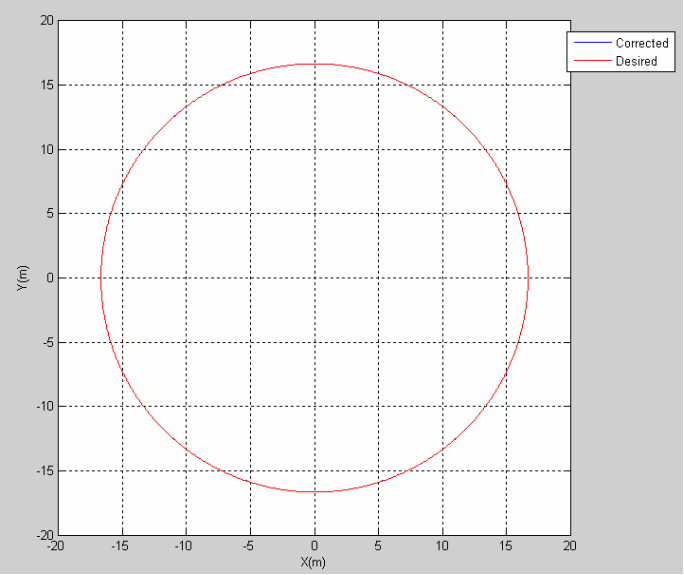

Figure 13. Higher Gains Simulation at h = 15m and z = 18m

4. CONCLUSION AND FUTURE WORK

This paper presents pseudostatics, dynamics, and control for the C^4 robot in simulated contour-crafting construction. Under pseudostatics, methods were presented in attempt to maintain positive cable tensions for all motion.

In the controller design, we chose the classic computed-torque control method, with a PD controller for the outer-loop control. A single set of PD controller gains was sufficient to control the robot at different pulley and end-effector heights.

Future work includes building and controlling scale experimental C^4 robot hardware. The controller design still requires further work to ensure generality and robustness in the nonlinear system control. We can also optimize the performance of our controller for practical applications.

REFERENCES

[1] Khoshnevis, B., 2004. "Automated Construction by Contour Crafting – Related Robotics and Information Technologies". Journal of Automation in Construction – Special Issue: The best of ISARC 2002, 13(1): 5–19.

[2] Khoshnevis, B., Russel, R., Kwon, H, and Bukkapatnam, S., "Crafting Large Prototypes," IEEE Robotics & Automation Magazine, pp. 33-42, September 2001.

[3] Bosscher, P., Williams II, R. L., Bryson, S and Castro-Lacouture, D., "Cable-suspended Robotic Contour Crafting System," in Proceedings of the 2006 ASME DETC/CIE conferences, Philadelphia, PA, DETC2006-99016, September 2006.

[4] Ming, X., Williams II, R. L., and Bosscher, P., 2008, "Contour-Crafting-Cartesian-Cable Robot System Concepts: Workspace and Stiffness Comparisons", submitted, ASME DETC/CIE conferences, New York, NY, August 2008.

[5] Ming, X., 2007, "Kinematics, Dynamics and Controller Design for Contour Crafting Cartesian Cable Robot" M. S. Thesis, Ohio University, R.L. Williams II, advisor.

[6] Williams II, R. L., 2007, "Cable-suspended Vehicle Simulation System Concept" in Proceedings of the 2007 ASME DETC/CIE conferences, Las Vegas, NV, DETC2007-34595, September 2007.

[7] Lewis, F.L., Control of Robot Manipulators, Abdallah, C.T., and Dawson D.M., Macmillan Publishing Company, New York, NY, 1993.