Towards bridging the reality gap between tensegrity simulation and robotic hardware

Brian T. Mirletz[1], In-Won Park[2], Roger D. Quinn[1] and Vytas SunSpiral[2]

***Abstract*— Using a new hardware implementation of our designs for tunably compliant spine-like tensegrity robots, we show that the NASA Tensegrity Robotics Toolkit can effectively generate and predict desirable locomotion strategies for these many degree of freedom systems. Tensegrity, which provides structural integrity through a tension network, shows promise as a design strategy for more compliant robots capable of interaction with rugged environments, such as a tensegrity interplanetary probe prototype surviving multi-story drops. Due to the complexity of tensegrity structures, modeling through physics simulation and machine learning improves our ability to design and evaluate new structures and their controllers in a dynamic environment. The kinematics of our simulator, the open source NASA Tensegrity Robotics Toolkit, have been previously validated within 1.3% error on position through motion capture of the six strut robot ReCTeR. This paper provides additional validation of the dynamics through the direct comparison of the simulator to forces experienced by the latest version of the Tetraspine robot. These results give us confidence in our strategy of using tensegrity to impart future robotic systems with properties similar to biological systems such as increased flexibility, power, and mobility in extreme terrains.**

I. INTRODUCTION

Many desirable applications of robotics, such as search and rescue, planetary exploration, and environments dangerous to humans, are still largely inaccessible to current robots. Even though robots operate quite effectively in structured, largely predictable environments such as factories, single floor residences, and missions with detailed guidance from a human operator, most robots have difficulty locomoting through unstructured terrain or with low bandwidth commands. The existing robots most capable of completing these missions tend to use series elastic actuation [1], [28], [31]. Tensegrity, a concept from art and architecture describing systems with continuous tension networks [36], extends this compliant design concept to the entire structure of the robot [27]. One mission concept from the Dynamic Tensegrity Robotics Laboratory at NASA Ames Research Center uses a tensegrity structure to absorb the shock of landing on Saturn's moon Titan, and then re-uses that same structure for the robot's locomotion [37].

This work was supported by a NASA STMD Space Technology Research Fellowship, the NASA Innovative Advanced Concepts Program, and NASA's Game Changing Development: Human Robotic Systems Program

[1]Brian T. Mirletz and Roger D. Quinn are with the Biologically Inspired Robotics Lab, Department of Mechanical and Aerospace Engineering, Case Western Reserve University, Cleveland, OH 44016, bmirletz@case.edu; rdq@case.edu

[2]In-Won Park (Oak Ridge Associated Universities) and Vytas SunSpiral (SGT Inc.) are members of the Intelligent Robotics Group, NASA Ames Research Center, Moffett Field, CA 94035, vytas.sunspiral@nasa.gov

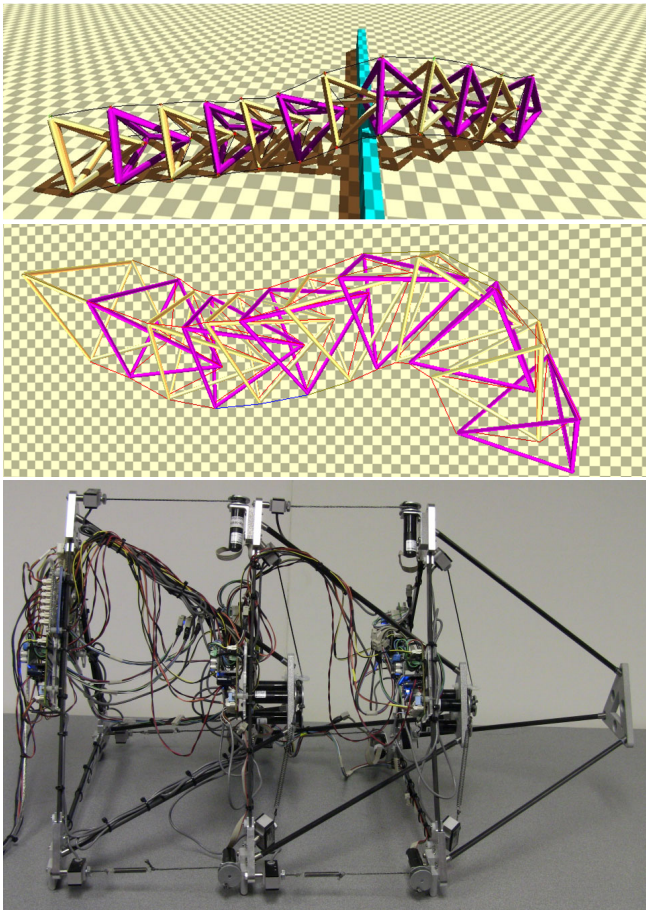

Fig. 1. Top: One frame from the NTRT physics-based simulation showing Tetraspine climbing over a wall. Middle: One frame from the NTRT physics-based simulation showing Tetraspine performing a sidewinding gait. Other gaits were also demonstrated. These highly adaptable gaits give us confidence in tensegrity spines as a locomotion platform. Bottom: Snapshot of TetraSpine3 hardware (the first segment is the farthest right).

Biological systems display a similar degree of ruggedness. Therefore, it is not surprising that non-linear properties of the cell's cytoskeleton such as strain hardening can be modeled with tensegrity [14]. These properties can be extended to macroscopic biological structures, as seen in passive tensegrity models of the spine [20], [6], knee [7], and shoulder girdle [19], for review see [32]. As a robotic example Tetraspine (Fig. 1), abstracts the morphology of the tensegrity spine models by using tetrahedrons as segments, connected by six cables, three from the tip vertex to the base of the next segment, and one from each of the vertices

of the base triangle to those of the segment behind it [38]. This design allows for clear actuation directions, and plenty of space for mounting actuators and electronics. The dynamics of Tetraspine and other models were explored both through physics simulation and machine learning [23], and produced locomotion that was highly adaptive to the terrain. Fig. 1 shows frames from simulations of Tetraspine (top) climbing over an obstacle and (bottom) locomoting using a sidewinding gait. Other gaits were also demonstrated.

Much of the analytical work on dynamic tensegrity structures has focused on optimization with well defined trajectories and environmental contact points, [35]. However these assumptions, especially about contact, are rarely valid for robots moving through natural unstructured environments. This suggests that physics-based simulation can provide intuition about the robots' performance in more complex environments. Many of the prior tensegrity simulations were dedicated to modeling only the equations of class 1 tensegrities, defined as bars only connected to cables [9], [10]. Finite element methods such as those used in [21], also assume predefined contacts. Modern physics engines provide advantages such as the fast computation of contact dynamics [3], and new robot morphologies can be easily implemented and tested in complex environments [30], [38], [23]. Engines including ODE [27], [29], [30], Simmechanics [26], and Bullet via the NASA Tensegrity Robotics Toolkit [38], [4], [8], [17] have been used by tensegrity researchers for this purpose. Additionally, the faster run time of these engines means that machine learning algorithms can be used to determine control strategies for structures [27], [15].

To ensure the structures and controllers designed with these simulation tools are realistic, it is important to verify the simulation results in hardware. This is a known problem to those who try to apply controllers learned in a simulator to hardware, and is sometimes called the "reality gap" [18]. Prior hardware implementations of tensegrity robots have qualitatively demonstrated locomotor strategies developed in simulation [27], [38], or tuned their controllers directly on hardware [16], [2]. A quantitative match of kinematics (within 1.3% error) is achieved by Caluwaerts et al. for the NASA Tensegrity Robotics Toolkit, by confirming that a given control signal for a single actuator produces the same rolling motion in a six strut robot [4]. These results could be extended by examining the forces experienced by the robot controlled by multiple actuators and those predicted by the simulator, as motors could be dimensioned using the simulator during design. This paper provides this comparison of forces, resulting in additional confidence in structures and controllers designed in our simulator.

II. CONTROL OF TENSEGRITY SPINES IN SIMULATION

A. The NASA Tensegrity Robotics Toolkit

The NASA Tensegrity Robotics Toolkit (NTRT) is an open source software package for modeling, simulation and control of tensegrity robots[1]. The NTRT simulator uses Bullet Physics version 2.82 as its underlying physics engine [5]. Simulations were run at a fixed timestep (1000 Hz for this paper) at which they can either run in real time with rendering (60 Hz), or faster than real time without rendering. For example, a 60 second simulation without rendering can run in approximately 7 seconds on an Intel Core™ i7 CPU.

Bullet's default softbody strings use a position based animation method, which is not realistic, so we use a Hooke's law based spring and cable model originally implemented in [4]:

$$f = \begin{cases} k(x - \ell_i) + b\dot{x} & : x > \ell_i \\ 0 & : x <= \ell_i \end{cases} \quad (1)$$

$$x = \|\mathbf{p}_{i,0} - \mathbf{p}_{i,1}\| \quad (2)$$

where k is the spring stiffness, b is a linear damping term, x is the Euclidean distance between the attachments of the spring-cable assembly ($\mathbf{p}_{i,0}$ and $\mathbf{p}_{i,1}$), and ℓ_i is the cable length. Actuation is accomplished by changing ℓ_i, which is subject to speed, acceleration, and force constraints comparable to our robot's actuators [23]. These cable-spring systems are not represented as collision objects in Bullet's physics world, and affect the rigid bodies by applying impulses to them. To ensure accurate contact dynamics as we test our structures on rough terrain we are currently implementing a model with both internal forces and contact dynamics, discussed further in Section V.

We construct models in NTRT using a set of builder tools that takes a tagged set of Cartesian coordinates (nodes) and their connectivity (pairs), and generates a tensegrity structure according to physical properties of the rods (radius, density, friction, restitution) and cables (stiffness, damping, motor properties). Previously, models used objects of uniform mass, such as a series of compounded rods. For this paper, we have added additional Bullet collision shapes, which includes the ability to place a sphere at a node as a point mass. This allows concentrations of mass such as motors to be accurately placed in the simulated structure, increasing the accuracy of the inertia matrices used by the simulator.

B. Control Methods

Our controllers also take inspiration from biology, in that they are distributed and hierarchical. At the lowest level we use a distributed form of impedance control, which was originally developed for robotic manipulators [11]. In a form adapted to the individual members of a tensegrity structure, the equation becomes scalar, but provides for control on both length and tension [26], as well as tunable stiffness. The equation is as follows:

$$T = T_0 + K(L - L_0) + B(V - V_0) \quad (3)$$

Where T is the tension setpoint, T_0 is a tension offset, K is a position gain on the difference between the cable's current

[1]Information, source code, and documentation for NTRT can be found at http://irg.arc.nasa.gov/tensegrity/NTRT

actual length L (x in (1)) and desired length L_0. B serves a similar function for V and V_0 (equivalent to $\dot{x}$), where V_0 is a control input from the CPGs or sine waves (in Orki et al.'s implementation V_0 is always zero [26], [38]). All variables except V and V_0 should be greater than or equal to zero for stability, given the convention that lengthening is positive. In our prior work, the impedance controller provided online adaptations for the cable trajectories [38], [24].

To control gait, we use a phase coupled oscillator based on a central pattern generator (CPG) model developed by Ijspeert and Crespi [12], see also [13]. The equations are as follows:

$$\dot{\theta}_i = 2\pi v_i + \sum_j r_j w_{ij} sin(\theta_j - \theta_i - \phi_{ij}) \tag{4}$$

$$\ddot{r_i} = a_i(\frac{a_i}{4}(R_i - r_i) - \dot{r_i}) \tag{5}$$

$$V_i = r_i(cos(\theta_i)) \tag{6}$$

The main variables are the amplitude r_i and phase θ_i which together form the velocity input to the impedance controller V_i via (6). The phase is determined by v_i, a frequency term, the coupling term w_{ij}, and the phase offset ϕ_{ij} according to (4). The amplitude r_i smoothly integrates to the setpoint R_i according to the constant a_i in (5).

When applied to tensegrity spines, we attach one node of the CPG to each cable of the tensegrity. We then couple each node to every other node that shares a rigid body. In the case of a three segment spine like our hardware implementation, this results in all-to-all coupling, but when the spine gets longer the network becomes more efficient as the there are no couplings that span beyond a neighboring segment [23]. Symmetry rules can then be used to reduce the number of phase offsets from 132 to 57 [23]. Thus, the simulation needs to learn 116 parameters, since every coupling requires both an weight and a phase offset, plus two more parameters for globally used amplitude and frequency terms. Given the large number of parameters, we use a Monte Carlo simulation to find an initial estimate of good values, and then simulate hill climbing by iteratively sampling a Gaussian distribution around the best values.

Given the open loop output of (6), a similar gait may be generated using sine waves. This is what we do while hand tuning a gait on the robot directly. However, due to the random nature of the search, a similar number of iterations produces the same quality of output in both CPGs and sine waves (despite the lower number of parameters involved for sine waves for this case), and we anticipate CPGs will ease the inclusion of sensory feedback and gait transitions in the future [13]. While our automated tuning procedure has no guarantee of optimality we have found it to provide effective locomotion on structures within a short period of time, including gaits that would be difficult to design using analytical methods [23]. Future work will incorporate additional optimization methods for these gaits such as the Nelder-Mead simplex method [25], to augment or replace the Gaussian hill climbing.

III. HARDWARE IMPLEMENTATION OF TETRASPINE

Our initial prototype presented in [38] explored the use of knit stretch sensors for tension. This robot was capable of locomotion but the sensors proved too inconsistent and therefore too difficult to calibrate. Our second prototype was made with an aluminum tetrahedron frame, and was only capable of position control, so we recently completed the "Tetraspine3" prototype with all of the sensing and actuation capabilities required to match our simulations. Fig. 1 (bottom) shows a snapshot of the Tetraspine3 hardware, which consists of 12 DC motors, 12 vectran cables, and 12 load cells. The length and diameter of Tetraspine3's rods is equivalent to 38.10 cm and 0.64 cm, respectively. One end of each vectran cable is connected to a DC motor (Faulhaber 1524) with a spool for rotational motion. The other end is fixed to a 222 N (50 lb) tension and compression load cell (Load Cell Central LCC-CTD) for measuring cable tension. The significant improvement compared to the previous two prototypes is the increase in load sensor resolution provided by this load cell and the use of lightweight materials in the robot's construction. For 8 out of 12 cables, a 227 N/m spring is placed before the load cell to increase compliance. The other four cables (top inside and top outside) did not include springs as they required additional stiffness to support the weight of the robot.

All inner and outer compression elements are made of carbon fiber to reduce the mass of each segment. The tip of the tetrahedron, where three motors are located to control the inner cables, and the spacers located between the motor bracket and the base triangle were made using a 3D printer to reduce the overall structural mass. Since the first and the last segment only control the outer and the inner cables, respectively, the mass of the mid-segment is greater than the others (0.73 kg for the front and rear segments, 0.91 kg for the middle segment). This is due to the presence of 6 DC motors to control both the inner and the outer cables.

Tetraspine3 is controlled by six slave boards where each slave board consists of a microcontroller (Arduino Nano), a motor driver (Pololu MC33926), a voltage regulator (LTC3260), and two instrumentation amplifiers (TI INA114) to control two DC motors. As shown in Fig. 1 (bottom), two slave boards are attached to each segment. Since the maximum voltage swing of the load cell is 10.0 mV, the instrumentation amplifier is used to amplify the output voltage of the load cell, which is then directly connected to the 10-bit ADC in the microcontroller. The voltage regulator is used to generate $\pm$5 V from the 12 V supply voltage. Note that the negative output voltage of the amplifier represents the compression mode of the load cell, whereas the positive output voltage corresponds to the tension mode. Fig. 2 shows the results of load cell calibration, where the maximum measurable tension is set to 111.21 N (equivalent to 25 lb). After the load cell calibration process (Fig. 2) the resolution of the load cell is approximately 0.16 N/ADC bit, as shown in the following linear relationship:

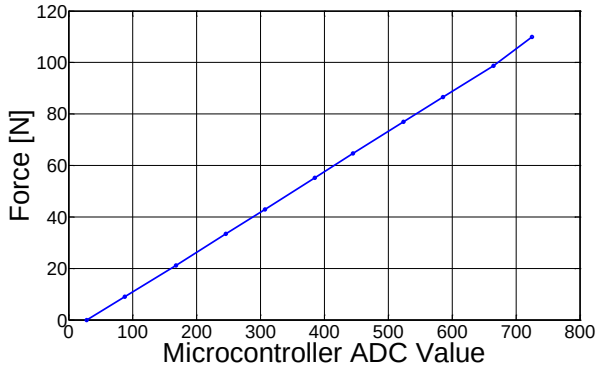

Fig. 2. The load cell calibration.

$$f = 0.1567(ADC) - 4.9363 \tag{7}$$

A PC and the six slave boards communicate using fast-mode I^2C protocol (400 kHz) through Devantech USB-ISS communication module. Every 10 milliseconds, the PC broadcasts a packet to all six slave boards simultaneously, which contains all 12 desired motor commands. Within 20 milliseconds, the PC then connects to each slave board in order to receive the actual and target tension values, the motor encoder value, and the motor PWM output value. In the experiments, the process of broadcasting the packet to all six slave boards and receiving the packet from all six slave boards takes less than 3 milliseconds. Every 200 milliseconds, the PC also stores all feedback values from all six slave boards and shows them in a GUI.

Both position control and impedance control are implemented in the microcontroller, where the control mode can be changed from the PC. The packet contains the desired motor angles in the position control, whereas it represents the desired cable velocity in the impedance controller. Position control is used at startup to set the initial length of each cable and to maintain the balance of all three segments. When impedance control is switched on, the current length and tension values are used as the L_0 and T_0 parameters in (3). In both control modes, the motor is controlled with a PID controller every millisecond (1000 Hz), where the measured loop times of position control and impedance control are 400 microseconds and 800 microseconds, respectively. Implementation of a complementary filter for the load cell ADC value and calculating the actual cable length and velocity for impedance control consume additional computational time compared to position control. Note that the tension offset, the rest length, the stiffness gain, and the velocity gain in (3) can be set differently for each cable from the PC.

IV. RESULTS

With this new robot, we are now able to directly compare the forces experienced by the robot with the simulator. We have chosen to focus on force for two reasons: first, our impedance controllers' output is force, so this provides a good test of our control system. Second, force is a key property in the design of new structures and selection of actuators. Ensuring the simulator's forces are in range with the hardware will allow us to have confidence in future simulated designs.

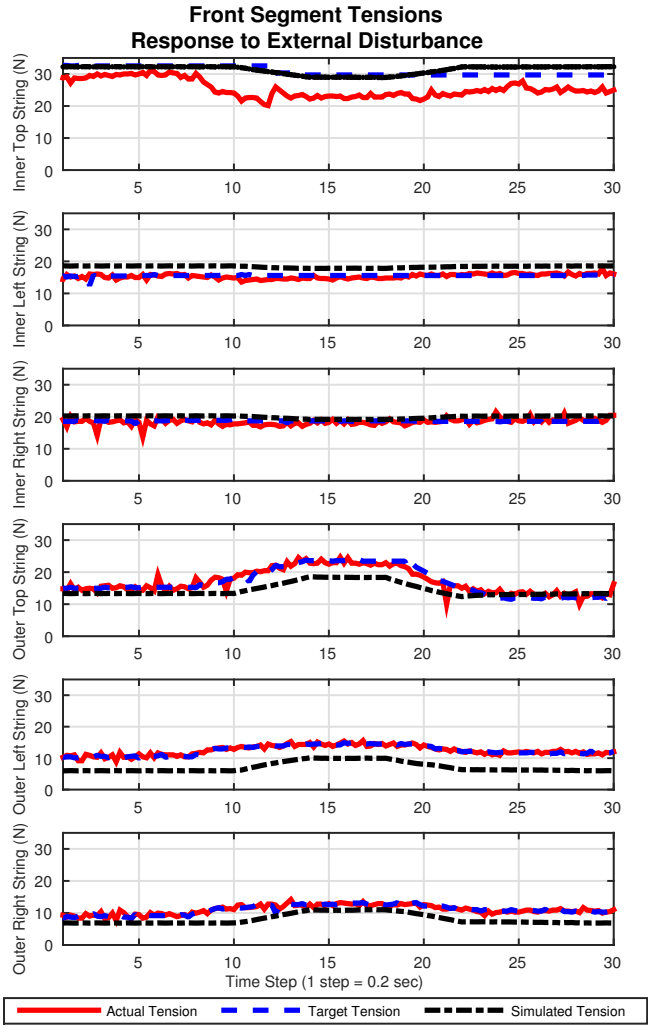

Fig. 3. We tested the steady state responses of the control by applying a steady load of approximately 20 N (measured using a hand-held scale) to the front segment of the robot while holding the other two in place. The resulting displacement and added tension causes the impedance controller to increase the tension setpoint, mostly in the outside cables. The forces experienced in each cable are plotted, with the hardware's setpoint in blue, its actual tension in red, and the simulator's predictions in black. The simulator's error on the system's maximum tension is within 6.1%.

We started by manually updating the simulated version of Tetraspine to match, as closely as possible, the material properties of the robot such as cable stiffness, inertia, and friction. To verify these properties, we used a quasi-static tests of dragging the robot at a steady velocity and applying a 20 N force to the front segment of the robot. Once we were confident in the match between the simulation and hardware, we tested a trajectory that was hand tuned on the robot in the simulator, and then ran a simulator learned trajectory for a different gait on the robot.

A. *Quasi-static*

We started by attempting to calibrate the simulation to the hardware and determine properties such as dynamic friction by measuring the force to drag the robot at a steady velocity. We then applied a steady state disturbance of approximately 20 N (a forward force on the front of the robot, and a restoring force on the rear two segments), and verified that the control response was similar in both cases.

TABLE I

CONTROL PARAMETERS USED IN FIG. 3

Value	Top Cables	Other Cables
K	500 [N/m]	100 [N/m]
B	100 [N/m-s]	100 [N/m-s]

The results of this test are shown in Fig. 3. The overall system dynamics are similar, with the added tension from the disturbance being split between the outside cables. When tuning for this test, we found that even without springs in the top outside and inside cables, it was effective to model them at the same stiffness as the rest of the system (227 N/m), we refer to this as the 'effective stiffness' of these cables. Decreasing the timestep to 250 Hz changes the predicted force on this test by 6% (worst case), increasing the time step to 4000 Hz changes the predicted force by 1.3%.

B. *Hand Tuned Trajectory*

Subsequently, we hand tuned a sine wave as the velocity input to the impedance controller on the hardware. The final tuning resulted in the inside cables having the inverse sign of the outside cables, and the rear cables having a phase offset of π from the front cables.

The velocity input in this case required some additional scaling. Amplitudes given to the simulator were $1/10$ what they were in the hardware (with identical input, the simulated robot's segments would slide together, causing it to topple). The results of this test (Fig. 4) show some differences in the system's pretension levels, but similar sizes of oscillations for each cable. We suspect this is due to the absence of perfect sticking from Bullet's default Coulomb friction model, as the segments tend to slide together under higher pretension. This confirms that some of the actuator properties such as effective stiffness and actuation speed were sufficiently accurate. Another difference is sensor noise, which is present in the tension readings (red line), but not in the simulator data (black line). These experiments show the control is robust to this level of noise.

C. *Simulator Generated Trajectory*

In order to ensure that we have a useful model, as opposed to a fit of existing data, we chose to make a prediction and test it on the hardware. For our prediction, we used our tuning methods in simulation to generate a trajectory through an open loop CPG capable of sideways motion, similar to the side-winding gait for the twelve segment version [23]. We exported these values to a text file and then used them as

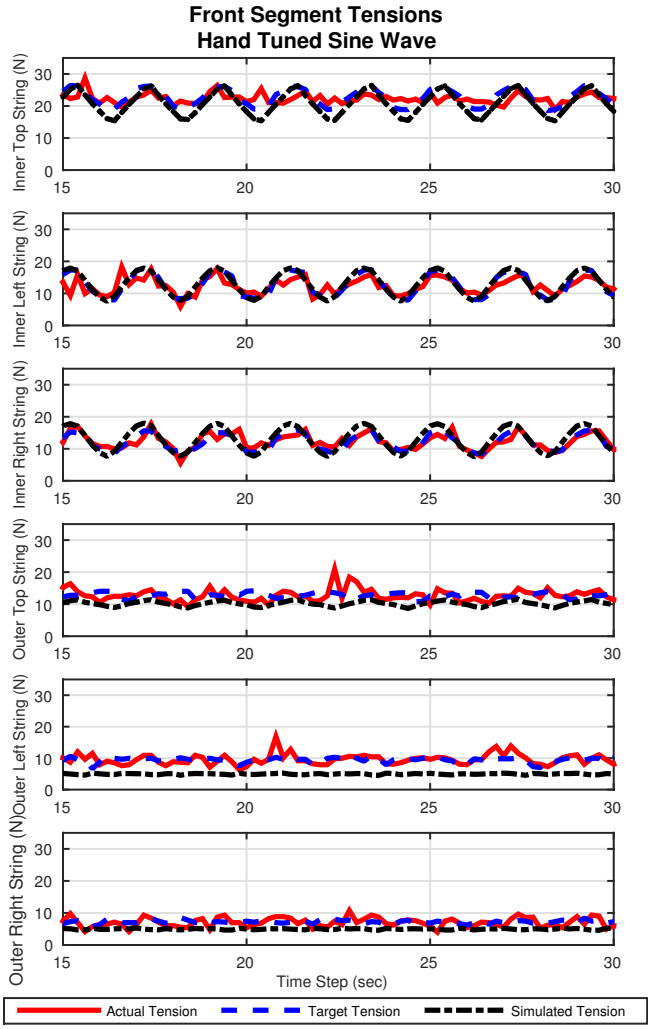

Fig. 4. A comparison of the tensions experienced by the hardware and simulation under a hand tuned, sine wave input on the velocity term of the impedance controller. The simulated maximum system tension in this case is within 7.9% of the hardware.

the velocity input on the hardware. The resulting forces are shown in Fig. 5.

The gait produced successful locomotion on the hardware implementation (see supplementary video). Note how the amplitude of the CPG's velocity trajectory forces the inside cables to a more saturated triangle-wave like oscillation in both cases, while the outside cables are more sinusoidal, verifying the output of the impedance controllers to this input. Again, the pretension is lower in the outside cables, and the speed of locomotion in hardware is slower due to additional static friction and unmodeled cable friction, but the overall direction of locomotion is similar.

V. CONCLUSIONS AND FUTURE WORK

We found that the forces predicted by the NASA Tensegrity Robotics Toolkit were within expected errors for robotic design, and that the accuracy extended from a single motion caused by a single actuator to the continuous coordinated motion of twelve actuators. The errors found here would

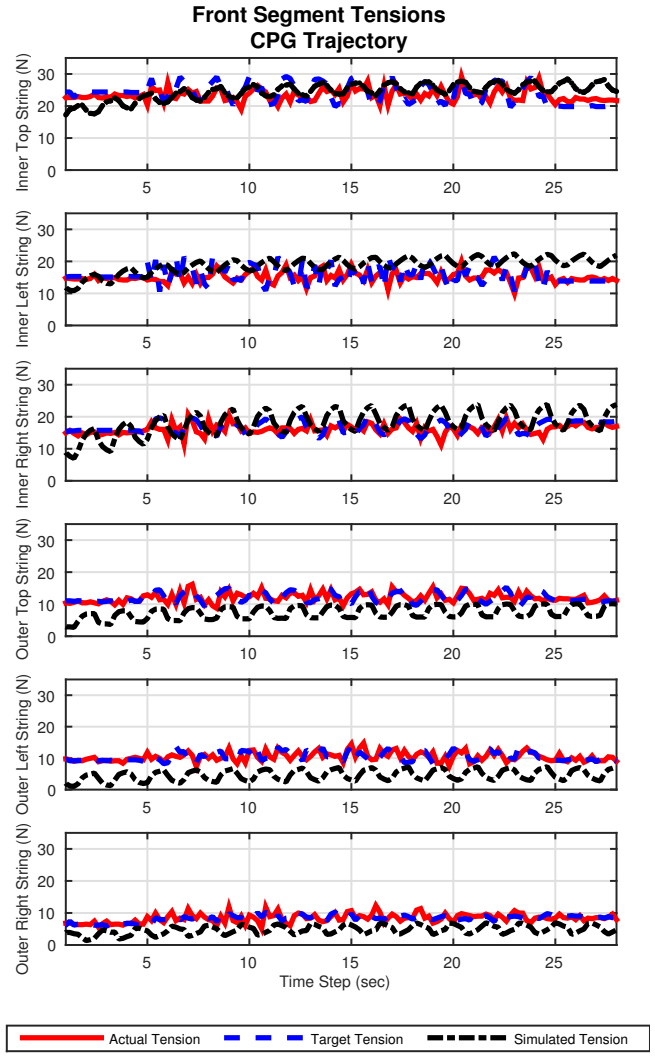

Fig. 5. A comparison of forces experienced under a simulator-tuned CPG based gait. The maximum tension experienced by the robot is within 1.6% of what was predicted by the simulator.

allow for designs with a safety factor of 1.15, whereas the actual safety factor we used in Tetraspine3's design was close to 2. These results confirm that we can continue to use NTRT for the design of new robots, with reasonable safety factors on parameters. Additionally, we generated a new gait for Tetraspine in the simulator and demonstrated the desired locomotion in hardware, though some hand tuning or online optimization is still required for an optimal gait in hardware. This work confirms our approach of using distributed impedance control with oscillatory gaits for controlling tensegrity robots.

Due to the nature of our cable implementation (applying impulses directly) changing Bullet constraint parameters such as the error reduction parameter (by a factor of 4) changes our results by less than 0.3%. We expect improving the friction models, both in actuators and within collisions will be the most effective way to improving the accuracy of this work. The accuracy of the work presented here could also be improved with more formal, automated system identification methods, especially if motion capture is used along with the force results.

Our next major improvement to the simulator will be soft-body cables with contact dynamics. We currently have an experimental implementation of a massless cable based on work by Servin et al. [33], [34]. The above data will help us ensure realistic dynamics, and we will conduct additional tests for the contact dynamics, both on individual cables and on complete robots. This will allow us to examine the dynamics of the robot on rough terrain, more complex cable routings, and perhaps even the details of our actuators. Accurate rough terrain behavior will support our ongoing work on goal directed locomotion for tensegrity spines traversing irregular terrain [22]. Future work on a custom friction model and more detailed actuator dynamics will also improve the transferability of gaits learned in the simulator.

With the new cable model, we will investigate cables that span multiple segments to better exploit the passive dynamics of the structure, and additional optimization of the locomotion. We expect that this will lead to a new tensegrity spine design, which could locomote on its own or incorporate limbs for walking and running, with the long term goal of robots with increased flexibility, power, and mobility on extreme terrain.

ACKNOWLEDGMENT

The authors would like to acknowledge Ryan Adams, Perry Bhandal, Lee Brownston, and Atil Iscen for their contributions to the NASA Tensegrity Robotics Toolkit, Adrian Agogino for discussions on machine learning, and Tom Flemons and Stephen Levin for inspiration on biotensegrity.

REFERENCES

[1] T. J. Allen, R. D. Quinn, R. J. Bachmann, and R. E. Ritzmann. Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1370–1375. IEEE, 2003.

[2] T. Bliss, J. Werly, T. Iwasaki, and H. Bart-Smith. Experimental validation of robust resonance entrainment for cpg-controlled tensegrity structures. *Control Systems Technology, IEEE Transactions on*, 21(3):666–678, 2013.

[3] A. Boeing and T. Bräunl. Evaluation of real-time physics simulation systems. In *Proceedings of the 5th int. conf. on Computer graphics and interactive techniques in Australia and Southeast Asia*, pages 281–288. ACM, 2007.

[4] K. Caluwaerts, J. Despraz, A. Işçen, A. P. Sabelhaus, J. Bruce, B. Schrauwen, and V. SunSpiral. Design and control of compliant tensegrity robots through simulation and hardware validation. *Journal of The Royal Society Interface*, 11(98):20140520, 2014.

[5] E. Coumans. Bullet physics library. `http://bulletphysics.org/wordpress/`, December 2012.

[6] T. Flemons. The geometry of anatomy. `http://www.intensiondesigns.com/geometry_of_anatomy.html`, 2007.

[7] T. Flemons. The bones of tensegrity. *http://www.intensiondesigns.com/bones_of_tensegrity*, 2012.

[8] J. Friesen, A. Pogue, T. Bewley, M. de Oliveira, R. Skelton, and V. SunSpiral. Ductt: a tensegrity robot for exploring duct systems. In *IEEE Int. Conf. on Robotics and Automation (ICRA-2014)*. IEEE, 2014.

[9] A. Graells Rovira and J. M. Mirats Tur. Control and simulation of a tensegrity-based mobile robot. *Robotics and Autonomous Systems*, 57(5):526–535, 2009.

[10] S. Hirai and R. Imuta. Dynamic simulation of six-strut tensegrity robot rolling. In *Robotics and Biomimetics (ROBIO), 2012 IEEE Int. Conf. on*, pages 198–204. IEEE, 2012.

[11] N. Hogan. Impedance control: An approach to manipulation: Part I - Theory. *Transactions of the ASME*, 107(March 1985):1–7, 1985.

[12] A. J. Ijspeert and A. Crespi. Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model. In *Proceedings of the 2007 IEEE int. conf. on robotics and automation (ICRA 2007)*, number BIOROB-CONF-2007-009, pages 262–268, 2007.

[13] A. J. Ijspeert, A. Crespi, D. Ryczko, and J. M. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science (New York, N.Y.)*, 315(5817):1416–1420, March 2007.

[14] D. E. Ingber, N. Wang, and D. Stamenović. Tensegrity, cellular biophysics, and the mechanics of living systems. *Reports on Progress in Physics*, 77(4):046603, 2014.

[15] A. Iscen, A. Agogino, V. SunSpiral, and K. Tumer. Robust distributed control of rolling tensegrity robot. In *The Autonomous Robots and Multirobot Systems (ARMS) workshop at AAMAS 2013*, 2013.

[16] M. Khazanov, B. Humphreys, W. Keat, and J. Rieffel. Exploiting dynamical complexity in a physical tensegrity robot to achieve locomotion. In *Advances in Artificial Life, ECAL*, volume 12, pages 965–972, 2013.

[17] Kyunam Kim, Adrian K. Agogino, Deaho Moon, Laqshya Taneja, Aliakbar Toghyan, Borna Dehghani, Vytas SunSpiral, and Alice M. Agogino. Rapid prototyping design and control of tensegrity soft robot for locomotion. In *International Conference on Robotics and Biometrics*, Dec 2014.

[18] S. Koos, J. B. Mouret, and S. Doncieux. Crossing the reality gap in evolutionary robotics by promoting transferable controllers. In *Proc. of the 12th annual conf. on Genetic and evolutionary computation*, pages 119–126. ACM, 2010.

[19] S. M Levin. Putting the shoulder to the wheel: a new biomechanical model for the shoulder girdle. *Biomedical sciences instrumentation*, 33:412–417, 1997.

[20] S. M. Levin. The tensegrity-truss as a model for spine mechanics: biotensegrity. *Journal of Mechanics in Medicine and Biology*, 2:375–388, 2002.

[21] J. G. McGarry and P. J. Prendergast. A three-dimensional finite element model of an adherent eukaryotic cell. *Eur Cell Mater*, 7:27–33, 2004.

[22] B. T. Mirletz, P. Bhandal, R. D. Adams, A. K. Agogino, R. D. Quinn, and V. SunSpiral. Goal directed cpg based control for high dof tensegrity spines traversing irregular terrain. *Under Review*, 2015.

[23] B. T. Mirletz, I. W. Park, T. E. Flemons, A. K. Agogino, R. D. Quinn, and V. SunSpiral. Design and control of modular spine-like tensegrity structures. In *The 6th World Conf. of the Int. Association for Structural Control and Monitoring (6WCSCM)*, 2014.

[24] B. T. Mirletz, R. D. Quinn, and V. SunSpiral. Cpgs for adaptive control of spine-like tensegrity structures. In *2015 International Conference on Robotics and Automation (ICRA2015) Workshop on Central Pattern Generators for Locomotion Control: Pros, Cons & Alternatives*, 2015.

[25] J. A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[26] O. Orki, A. Ayali, O. Shai, and U. Ben-Hanan. Modeling of caterpillar crawl using novel tensegrity structures. *Bioinspiration & Biomimetics*, 7(4):046006, 2012.

[27] C. Paul, J. W. Roberts, H. Lipson, and F. J. V. Cuevas. Gait production in a tensegrity based robot. In *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th Int. Conf. on*, January 2005.

[28] M. Raibert, K. Blankespoor, G. Nelson, R. Playter, et al. Bigdog, the rough-terrain quadruped robot. In *17th World Congr. The Int. Federation of Automatic Control*, pages 10823–10825, 2008.

[29] J. Rieffel, B. Trimmer, and H. Lipson. Mechanism as Mind : What Tensegrities and Caterpillars Can Teach Us about Soft Robotics The Manduca Sexta Caterpillar : Morphological Communication in Tensegrity Robots. *Artificial Life*, pages 506–512, 2008.

[30] J. A. Rieffel, F. J. Valero-Cuevas, and H. Lipson. Morphological communication: exploiting coupled dynamics in a complex mechanical structure to achieve locomotion. *Journal of the Royal Society, Interface / the Royal Society*, 7(45):613–21, April 2010.

[31] D. Rollinson, S. Ford, B. Brown, and H. Choset. Design and modeling of a series elastic element for snake robots. In *ASME 2013 Dynamic Systems and Control Conf.*, pages V001T08A002–V001T08A002. American Society of Mechanical Engineers, 2013.

[32] G. Scarr. *Biotensegrity: The structural basis of life*. Handspring Publishing, first edition, 2014.

[33] Martin Servin and Claude Lacoursière. Massless cable for real-time simulation. In *Computer Graphics Forum*, volume 26, pages 172–184. Wiley Online Library, 2007.

[34] Martin Servin, Claude Lacoursiere, Fredrik Nordfelth, and Kenneth Bodin. Hybrid, multiresolution wires with massless frictional contacts. *Visualization and Computer Graphics, IEEE Transactions on*, 17(7):970–982, 2011.

[35] R. E. Skelton and M. C. De Oliveira. *Tensegrity Systems*. Springer, 2009 edition, June 2009.

[36] K. Snelson. Continuous tension, discontinuous compression structures. United States patent 3169611, February 1965.

[37] V. SunSpiral, G. Gorospe, J. Bruce, A. Iscen, G. Korbel, S. Milam, A. Agogino, and D. Atkinson. Tensegrity based probes for planetary exploration: Entry, descent and landing (edl) and surface mobility analysis. In *10th International Planetary Probe Workshop*, 2013.

[38] B. R. Tietz, R. W. Carnahan, R. J. Bachmann, R. D. Quinn, and V. Sunspiral. Tetraspine: Robust Terrain Handling on a Tensegrity Robot Using Central Pattern Generators. In *IEEE/ASME Advanced Intelligent Mechatronics*, pages 261–267, 2013.