Tensegrity Robot Locomotion under Limited Sensory Inputs via Deep Reinforcement Learning

Jianlan Luo[1] Riley Edmunds[2]Franklin Rice[2] Alice M. Agogino[1]

***Abstract*— Tensegrity robots are composed of rigid rods connected by elastic cables, and their unique light-weight yet compliant structure makes them an appealing choice for space exploration. However, locomotion control for these robotic systems remains difficult due to their nonlinear dynamics and high-dimensional state space. We demonstrate that in the domain of tensegrity robotics, it is possible to efficiently learn end-to-end locomotion policies using mirror descent guided policy search (MDGPS) even with limited sensory inputs. We compare learned neural network policies with other locomotion control policies in various testing environments; and results show that neural network policies consistently outperform others. We also shed light to the policy learning process by analyzing different choices of observation inputs to the robot. Moreover these findings motivate exploration of deep reinforcement learning algorithms in the domain of tensegrity robotics. We show preliminary results with one such locomotion example on discontinuous rough terrains.**

I. INTRODUCTION

Tensegrity or tension-integrity is an innovative robotic structure characterized by a network of cables connecting isolated rods [1], [2], [3]. Robots built upon this unique structure are light-weight, low-cost, and capable of withstanding significant impacts by deforming and distributing force across the entire structure [4], [6], [9], [10]. Figure 1 shows the locomotion simulation in this work. Figure 2 shows two actual tensegrity robots and hardware testing built as a joint effort between U.S. National Aeronautics and Space Administration (NASA) and the BEST (Berkeley Emergent Space Tensegrities) Lab at UC Berkeley [26]. As seen in the figure although no rod members touch each other, a tensegrity structure maintains its equilibrium geometry by delicately balancing cable tension and rod compression forces [1]. NASA is researching tensegrity robots for space exploration missions [9], [11], as their unique structure promises significant advantages in locomotion over unpredictable terrains. The tensegrity robot used in this work is the TT-4, a 24-cable, 6-bar, 24-motor spherical tensegrity robot shown in Figure 2.

While agile mobility is desirable in space exploration tasks, locomotion control for soft robotic systems remains a challenging problem due to highly nonlinear, coupled dynamics caused by frequent interaction between connecting members. We generally have three methods to tackle this problem: hand-engineered/hard-coded open-loop controllers

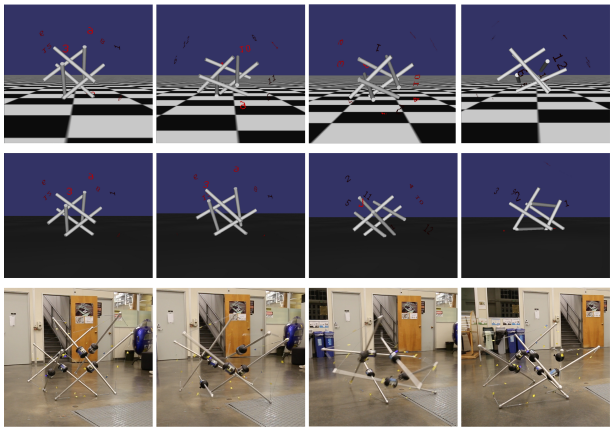

Fig. 1. *Top*: Tensegrity robot running MDGPS in the NTRT simulation environment (flat terrain) with limited sensory input; *Middle*: Tensegrity robot running PIGPS in the NTRT simulation environment (rough terrain); *Bottom*: Hardware tests, transferring the MDGPS policy to the TT-4 Tensegrity Robot.

such as lookup tables; optimal control methods such as linear quadratic regulator (LQR) or model predictive control (MPC); and model-free or model-based reinforcement learning methods.

Hard-coded controllers are typically time-consuming and counter-intuitive. These controllers generally achieve tensegrity locomotion either through crawling or step-by-step punctuated rolling. These approaches boil down to hard-coded look-up tables and tend to follow a common order of commands: 1) contract one cable on the bottom triangle until the robot tips onto the next face, 2) reset the robot by returning all cables to rest length, 3) repeat. A lookup table can then be created from the set of actuations that resulted in successful rolls. The major problem with this approach is that it can be tediously slow and the control can be suboptimal on terrains without hard-coded control policies.

Achieving effective locomotion through traditional optimal control approaches is also difficult due to the highly nonlinear nature of tensegrity tension networks. These optimal control methods typically derive state-space models by linearizing the Lagrangian dynamics of the system, often implemented as second-order Taylor series expansions around a set of operating points, that are only accurate in a small region. Additionally, error accumulation as the optimization horizon increases often makes it difficult to get an accurate global dynamics model, and quickly makes applying traditional optimal control techniques to tensegrity

[1]Department of Mechanical Engineering, University of California, Berkeley, Berkeley, CA, 94720
[2]Department of EECS, University of California, Berkeley, CA, 94720
{jianlanluo, rileyedmunds, frice, agogino} @berkeley.edu

locomotion intractable.

On a similar note, standard model-free reinforcement learning method such as Q-learning and genetic algorithms often encounter the curse of dimensionality as the search space in question grows to be extremely large. Such algorithms often require millions of iterations before converging to a reasonable policy. While these algorithms may capture the expressiveness missing in hand-engineered and optimal control methods, past work in model-free reinforcement learning on tensegrity systems has proven prohibitively inefficient.

By comparison, a recently developed model-based reinforcement learning algorithm called "guided policy search" (GPS) gives new insights into solving this paradoxical complexity-efficiency trade-off problem [18], [19]. GPS is more sample-efficient than previous model-free reinforcement learning methods because it seeks to find solutions bridging optimal control and deep reinforcement learning, and in contrast to existing policy search algorithms learns local models in the form of linear Gaussian controllers. When provided with rollout data from these linear local models, a global, nonlinear policy can then be learned using an arbitrary parametrization scheme. The method alternates between (local) trajectory optimization and (global) policy search in an iterative fashion [17]. We choose a particular variant of GPS called mirror descent guided policy search (MDGPS) because it is sample-efficient and its use of on-policy sampling makes it a good fit for this periodic locomotion gait task.

In this paper, we demonstrate a method that autonomously learns locomotion policies for our TT-4 tensegrity robot both in simulation (and preliminarily in hardware), even in low dimensional observation spaces. We compare the learned neural network policy to several baselines, including hand-engineered and LQR controllers, with the goal of maximizing distance traveled in a given period of time. Results from these comparisons show that the learned policy outperforms those baselines by a large margin. Upon experimentally examining what the MDGPS policy has learned in these tests, we saw that the learned neural network policy maps directly from low-dimensional observation space to action space rather than estimating system dynamics and generating control laws. These results suggests that end-to-end policy learning may be a better choice than attempting to accurately derive complex dynamics of tensegrity locomotion. Finally, we show that we can use model-free updates to the policy to achieve better performance on contact-rich rough terrains with discontinuous dynamics, further strengthening the argument for end-to-end policy learning.

The main contributions of this work are as follows. In the domain of tensegrity robotics characterized by complex physical dynamics:

1) We found that despite their dynamic complexity, tensegrity robots can learn locomotion in extremely low-dimensional observation spaces, by mapping directly from observation components to actions. We found that neural networks are expressive enough to construct this mapping without resorting to modeling of tensegrity physical dynamics.
2) We evaluated our methods in this domain by showing that learned neural network policies outperform other tensegrity locomotion control policies in limited-sensory-input environments. We demonstrate that even in a three-dimensional observation space, a tensegrity robot whose state space is 96-dimensional can learn efficient locomotion when using a deep neural network policy.
3) Finally, we motivate research into previously unexplored deep reinforcement learning algorithms, and demonstrate preliminary results with one such model on rough terrains characterized by highly discontinuous dynamics.

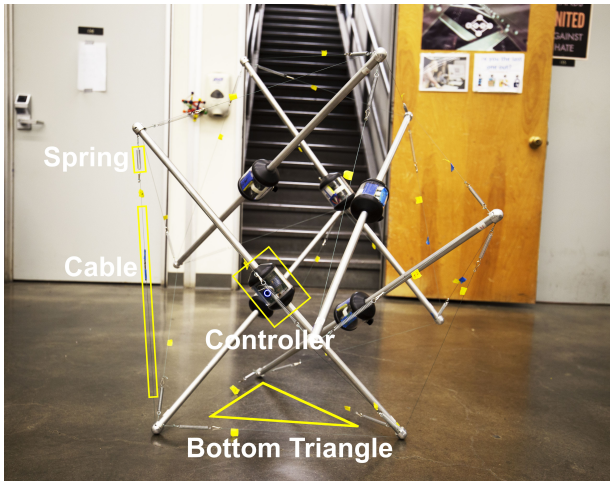

Fig. 2. The "TT-4": a 24-cable, 24-motor tensegrity structure developed by the Berkeley Emergent Space Tensegrities Lab. Notice that the center of mass rests over the bottom triangle while the structure is at rest.

II. RELATED WORK

Both simulations and real robot experiments involving tensegrity robot locomotion have been introduced [1] [2] [7] [8] [12] [13]. Early work focused on quasi-static tensegrity walking via calculation of forward kinematics, assuming the tension networks of the tensegrity robot was in equilibrium for each step[30][31]. NASA's most recent work focuses largely on developing dynamical locomotion controllers in its tensegrity simulation environment, the NASA Tensegrity Robotics Toolkit (NTRT)[21]. By comparison, Skelton et al. [5], [7], [8], [12], [13] formulated tensegrity locomotion as a constrained optimization problem, showing that tensegrity structures can deform along simple trajectories with the bottom triangle fixed in place. This approach, however, is not practical for real world applications because the bottom triangle is fixed, and thus the robot cannot roll. Researchers at NASA have attempted to use model-free learning techniques such as Q-learning and genetic algorithms [3], [4], [9] for these tasks, however, due to their trial-and-error nature,

these methods typically require millions of iterations before reaching the final converged policy.

Additionally, recent work in deep reinforcement learning has enabled the application of deep neural network policies to the problems of bipedal walking[20], and 2-DOF system control[27][28][29].

Marvin et al. [14] demonstrated continuous dynamic locomotion for tensegrity robots, both in simulation and hardware, by adopting a recently developed algorithm called "guided policy search" (GPS)[15], [16], [17], [18], [19]. They were able to directly transfer their learned policy from simulation to a real robot without feeding additional data to compensate for the discrepancies between the simulation environment and the real world. It was unclear what happened during the policy learning process, as this zero-shot policy transfer is not common in the domain of robotics.

III. MIRROR DESCENT GUIDED POLICY SEARCH

Intuitively, guided policy search uses locally optimal controllers to generate samples for training its arbitrarily parameterized global policy. High-capacity models such as deep neural networks are a common choice for this sort of challenge; however rather than using standard optimal control techniques that attempt to achieve success from any initial state, guided policy search seeks optimal solutions only in regions of the state trajectory where linearization is possible. Supervised learning is then used to elicit the underlying pattern in the local data, in order to construct a long-horizon global policy.

Formally, we denote $\pi_\theta(u_t|x_t)$ as the final policy, parameterized by θ over actions u_t, and conditioned on the state x_t. We denote the system dynamics as $p(x_{t+1}|x_t, u_t) = \mathcal{N}(f_{xt}x_t + f_{ut}u_t + f_{ct}, F_t)$, the local linear-Gaussian policies as $P(u_t|x_t) = \mathcal{N}(K_t x_t + k_t, C_t)$, and the cost function to be $l(x_t, u_t)$. We wish to minimize the expected cost along the state trajectory, i.e., $J(\theta) = \sum_{t=1}^{N} E_{\pi_\theta(x_t,u_t)}[l(x_t, u_t)]$. Thus our overall optimization problem looks something like equation below:

$$\min_{\theta, p_1, \cdots, p_N} \sum_{i=1}^{N} \sum_{t=1}^{T} E_{p_i(x_t,u_t)}[l(x_t, u_t)] \quad (1)$$

$$s.t.\ p_i(u_t|x_t) = \pi_\theta(u_t|x_t) \forall x_t, u_t, t, i \quad (2)$$

Instead of performing optimization on the parameter space by directly computing the gradient of $J(\theta)$, mirror descent guided policy search (MDGPS) alternates between local trajectory optimization and global policy optimization. That is to say, we improve the global policy within some trust region in the constraint manifold in policy space, then use supervised learning to project this improved policy back onto the constraint manifold in parameter space. We then choose a simple representation of our global policy by mixing several state trajectory distributions, where convenient trajectory-centric optimization methods can be applied, such as iterative Linear Quadratic Gaussian (iLQG). The full algorithm is detailed in Algorithm 1, where p_i represents the i-th local policy, o_t is the observation at time step t. Note that the equation includes KL-divergence divergence constraints, which are calculated by linearizing the global policy π_θ, and serve to minimize the difference between the global and local policies. In this implementation, we use the same method to linearize the global policy as we used to fit the dynamics, i.e., ask the neural network policy to take an action, record the $\{x_t, u_t, x_{t+1}\}$ tuples, and perform linear regression on them using Gaussian Mixture Models as priors.

Algorithm 1 MDGPS

1: **for** iteration $k \in \{1, ..., K\}$ **do**
2: Generate samples $D_i = \{\tau_{i,j}\}$ by running either p_i or $\pi_{\theta i}$
3: Fit linear-Gaussian dynamics $p_i(u_t|x_t)$ using samples in D_i
4: Fit linearized global policy $\bar{\pi}_\theta(u_t|o_t)$ using samples in D_i
5: $p_i \leftarrow argmin_{p_i} E_{p_i(\tau)}[\sum_{t=1}^{T} l(x_t, u_t)]$ such that $D_{KL}(p_i(\tau)||\bar{\pi}_{\theta i}(\tau)) \leq \epsilon$
6: $\pi_\theta \leftarrow argmin_{\pi_\theta} \sum_{t,i,j} D_{KL}(\pi_\theta(u_t|x_{t,i,j}) ||p_i(u_t|x_{t,i,j}))$ (via supervised learning)
7: Adjust ϵ
8: **end for**

MDGPS is suitable for periodic locomotion tasks because it performs on-policy sampling (line 2 of Algorithm 1). Let T^π represent the horizon of the global policy (we ideally would want $T^\pi = \infty$, however, this is impossible). Since any use of supervised learning introduces some degree of error, there will be discrepancies between local policies and the learned global policies. On-policy sampling directly samples from the global policy, and periodically encounters gait locomotion, while off-policy updates will likely lead the global policy towards failure, as the off-policy distribution quickly gets out of touch with the global policy's data distribution.

Furthermore, we need to have full access to state information only during the trajectory optimization phase, when observations are recorded to train the neural network policy. The fact that our neural network maps directly from observations to actions allows the final trained policy to operate in an environment that is only partially observable. This convenient mechanism allows us to perform training in simulation with fully accessible state information, and then transfer the learned policy directly to hardware for operation using observations recorded during training.

A flowchart outlining the generic MDGPS training scheme can be found in Figure 3.

IV. EXPERIMENTAL DETAILS AND RESULTS

A. Simulation

In our experiments, we define the robot state space as the set of all endpoint position for the six bars, positions of motors and their derivatives, for a total dimension of 96. In our software experiments, we evaluate policies on various

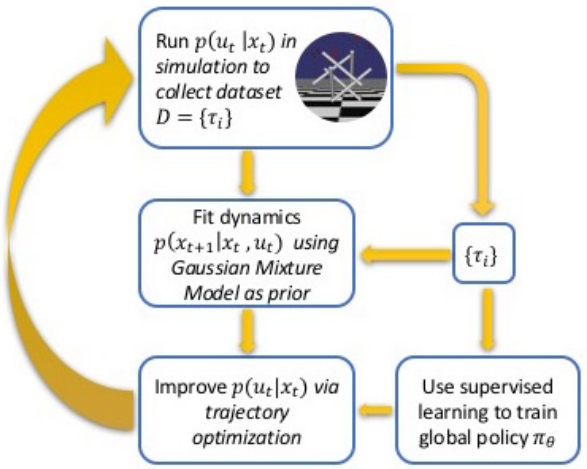

Fig. 3. The MDGPS training scheme used to train the Tensegrity.

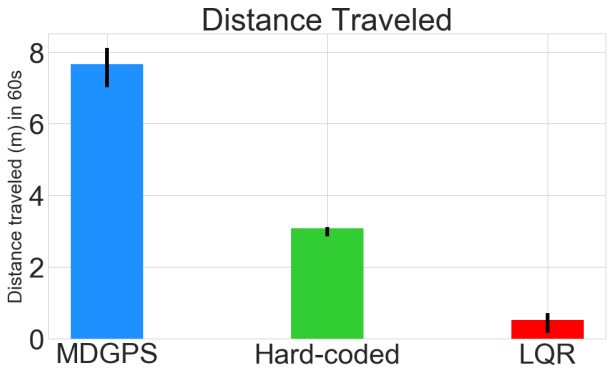

Fig. 4. Average distance traveled by each policy over 60 seconds. Notice that the MDGPS policy can travel over 2.5x faster then the hard-coded locomotion policy, and over 17x faster than that of LQR.

observation spaces (comprised of rod accelerations, rod vectors, and motor positions). Our hardware test's sensory input consists of 1-dimensional acceleration data from each of the six rods. We set the horizon for each local policy p_i to be $T^{p_i} = 50$, with each time step being 0.1 seconds. We choose six local policies starting from the six stable closed triangles of the robot, and let each local policy collect samples by running 15 rollouts in each iteration. Each local policy takes approximately 15 iterations before converging, for a total of 1350 rollouts. The training takes approximately four hours on an Intel I7-7700K machine, with an NVDIA GTX1080 GPU. We set the cost function to be the negative velocity of the center of mass, so as to maximize travel speed. The global policy is represented by a deep neural network with three hidden layers, each composed of 128 neurons, and rectified linear unit (ReLU) activations.

Figure 4 shows a speed comparison between the learned neural network policy, a hand-made controller, and an LQR controller using linearized dynamics. To evaluate our learned policies, we calculate the average distance traveled in 60 seconds by running each policy from its respective triangular base. Our MDGPS policy was able to successfully travel over 2.5x faster than the policy determined by the hard-coded lookup table, and over 17x faster than that of LQR.

For implementation of this algorithm, we used NASA's open source simulator designed for tensegrity robots, NTRT [21], and the guided policy search framework from the Berkeley Robot Learning Lab [19]. Figure 3 shows the training process in our simulator.

B. Hardware

For the hardware implementation, we used the TT-4 tensegrity robot, a six-bar spherical tensegrity robot developed at the Berkeley Emergent Space Technologies (BEST) Lab. The rod length is 1.033m, and the robot has four motors on each of its six rods. Each motor spools a cable through the hole in the end of its hollow rod, contracting the opposite rod towards itself.

To evaluate our learned policy in hardware, we directly transfer the policy trained using MDGPS in a three-dimensional observation space (composed of one-dimensional acceleration readings from three of the six rods). The tensegrity begins at rest on any of its closed triangular bases (visible in Figure 2). We receive the six acceleration readings from the robot's on-board IMUs, and then the neural network policy calculates the corresponding action. Finally, we send the resulting actions back to the TT-4 to be executed as motor commands. All communication protocols run at 10 Hz.

We were able to collect preliminary results in the form of one consecutive roll at a time, demonstrating that the policy is performing reasonably, even in hardware. Unfortunately, we were not able to run the policy sequentially due to communication latency, motor inconsistencies, and asymmetry in the TT-4 hardware design. However, a new generation of more robust TT-5 tensegrity robots is under development, and will open the door for more reliable future hardware experiments.

V. INTERPRETING THE NEURAL NETWORK POLICY MAPPING

MDGPS outperforms traditional tensegrity locomotion policies in limited sensory input environments, and thus TT-4 is able to learn effective locomotion even when presented with a mere three-dimensional observation state. In these reduced-dimensionality environments, we show that the neural network policy is not modeling the system dynamics, but rather maps directly from observations to actions.

Further, we investigate possible salient features commonly found in this observation state (such as gravitational and structural information) by training the neural network policy with a number of diverse observation states.

A. Evidence

We first verified empirically in simulation that spherical tensegrity robots can still effectively learn a locomotion policy given only a low-dimensional observation space (See Figures 5 and 6). We reduced the observation space from 12-dimensional acceleration data, distributed across six distinct

points (two directions of bar acceleration for each rod in a six-bar spherical tensegrity robot) to three-dimensional acceleration, distributed from three distinct points (one measure of acceleration along the direction of each of three of the six rods, so that each pair of parallel bars is represented once). In simulation we found that this 75% reduction in the dimensionality of the input data caused only slight detriment in the effectiveness of the learned policy (Figures 5 and 6), indicating that the higher-dimension data was carrying redundant or noisy information. This experiment validates the effectiveness of neural networks in the domain of spherical tensegrity locomotion to learn effective actions given limited sensory input.

It is notable that the tensegrity can roll effectively even when presented with merely three-dimensional input (Figures 5 and 6). When working in a three-dimensional observation space, compared with its 96-dimensional state space, the tensegrity only has access to a minuscule proportion of the total available sensory data. We claim that the tensegrity cannot learn its highly-complex physical dynamics model from a merely three-dimensional observation input, wherein 96-dimensional system state could not be estimated accurately from 3-dimensional observations, thus making estimating physical dynamics intractable.

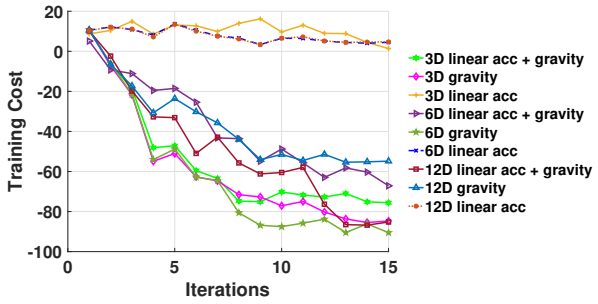

Fig. 5. Training cost over time for various observation spaces. More negative cost is better (cost is proportional to negative speed).

Fortunately, besides trying to learn actions, the model has an alternative: learn a function from the observation space to the action space. In this manner, the policy is able to circumvent modeling the system dynamics, and instead process the observation data in a highly nonlinear fashion in order to output the action which is most likely to maximize the robots expected velocity.

In conclusion, the TT-4 is able to perform efficient locomotion in limited sensory environments, without need to (or likely even ability to, under such limited sensory information) model the complex system dynamics. We expect that the inherent expressiveness of the neural network allows for expressive transformations from limited observation features into the space of actions.

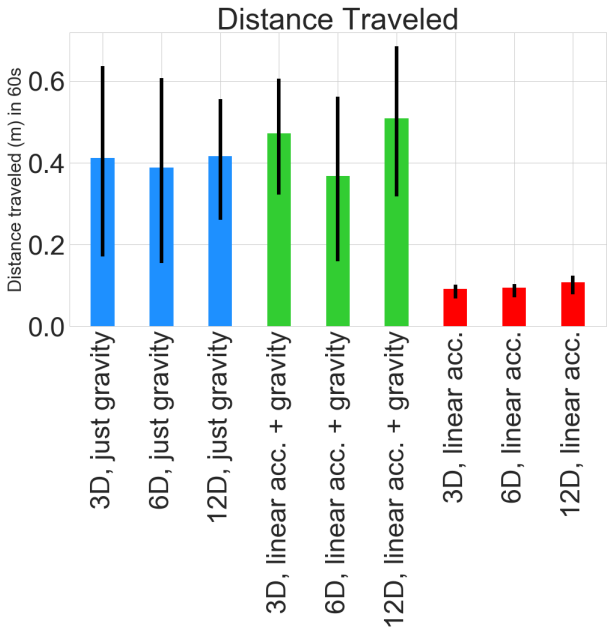

Fig. 6. Average distance traveled by each policy in 5 seconds. The observation space is defined by either measurements of gravity, linear acceleration, or both, in varying dimensions.

B. Exploration

We analyze the feature set existing in the low-dimensional observation space by training MDGPS neural network policies on that observation space, and testing their effectiveness.

We found that the most efficient manner of dissecting this low-dimensional observation space is by inspection of the gravity vector, as gravity measurements can be used to infer the relative orientation of each bar. With this in mind, we validated our hypothesis by training policies on simulated acceleration observations with gravity vectors removed (See Figure 6). Without this information, the resulting policy is unable to learn effective locomotion, indicating that gravity is indeed a fundamental feature of an effective lower-dimensional observation space.

We hypothesize that the geometric information needed by a spherical tensegrity robot to effectively plan locomotion comes from a small set of fundamental positional information. We surmised that this information would differ depending on whether the robot is at rest (has all three points of a single triangle on the ground) or in motion (no clear base triangle; between rest states) as follows: When the structure is at rest, the controller must simply discern which of the triangles is currently the base; When the structure is is motion, the controller must determine the basic structural geometry of the robot, from which it can deduce the degree of transition between rest states.

Per the qualification in the prior paragraph, it is important to note that while the three-dimensional gravity-vector observation space works well when the system is at rest, it is not sufficient to know solely the direction of gravity when the system is in motion. To account for varying degrees of cable contraction and varying relative rod positioning, it is

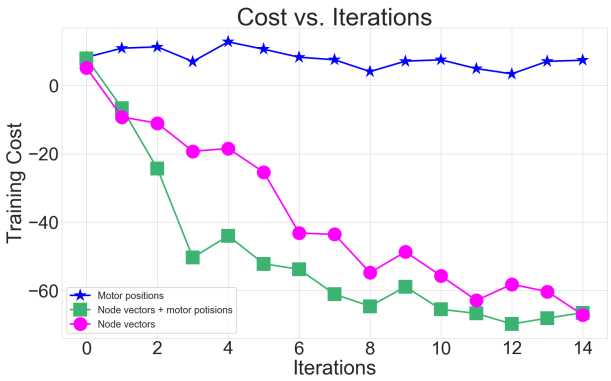

Fig. 7. Training costs plot for motor positions, rod vectors and combined observation spaces. Notice that the motor position policy does not converge, while policies with node vector information tend to converge.

useful to know the geometry of the systems rods themselves. We hypothesize that observations which define both the position as well as the orientation of the rods contribute to understanding the structural geometry of the system more successfully, and thus are more effective observations for learning dynamic tensegrity locomotion. To validate this hypothesis, we trained policies using observation spaces from which we can deduce structural geometry, but not gravity, for example vectors tracking the rods or motor encoder counts. We further validated the necessity of structural geometry input by training policies using observation spaces that contain no gravitational or structural-geometric information. We can see in Figure 7, that the policy learns effective locomotion only when presented with information from which it can deduce structural geometry. In order to explicitly encode structural geometry information, we define "rod vectors: a directional vector representing the position and orientation of each tensegrity rod. For any given rod with endpoint positions (a,b), this is the three-dimensional difference between them: $(x_b - x_a, y_b - y_a, z_b - z_a)$. Not surprisingly, our most successful policy is that which has access to this explicit encoding of structural geometry information: the policy whose observation space is defined to be the rod vectors (see Figure 7).

As we have shown that effective locomotion policies can be learned by spherical tensegrity robots with only structural geometry and orientation data, it can be understood that tensegrities are especially well-suited for learning approaches using deep reinforcement learning. While governed by complex dynamics, the robots tension network is inherently compliant, and endowed with a natural symmetry that renders it an ideal testbed for dynamics-free deep reinforcement learning research.

VI. PATH INTEGRAL GUIDED POLICY SEARCH ON UNEVEN TERRAINS

Motivated by the success of MDGPS, we decided to explore deep reinforcement learning algorithms that have never been applied to the tensegrity locomotion problem. We realized that MDGPS performed poorly on rough, uneven

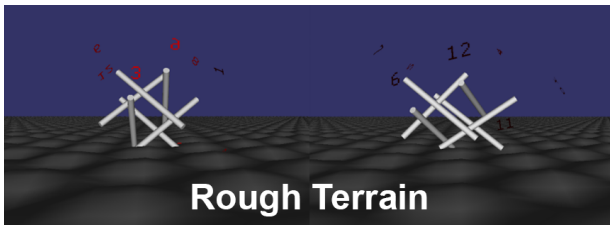

Fig. 8. Illustrations of the simulation environment for rough terrains with discontinuous dynamics. On this terrain, PIGPS outperforms MDGPS.

terrains. Figure 8 illustrates such terrains, where dynamics is discontinuous due to the contact-rich nature. The LQR-based methods used in MDGPS's trajectory optimization phase were unable to handle the terrains discontinuous dynamics. In order to address this issue, we adopted a recently developed algorithm [32], called path integral guided policy search (PIGPS). It modifies the trajectory optimization method in shown in Algorithm 1, line 5, in order to encourage more exploration of the local policy via "path integral optimization"(PI^2), when dynamics are highly discontinuous. The full algorithm is detailed in Algorithm 2.

Algorithm 2 MDGPS with PI^2 and Global Policy Sampling

1: **for** iteration $k \in \{1, ..., K\}$ **do**
2: Generate samples $\mathcal{D} = \{\tau_i\}$ by running noisy π_θ on each randomly sampled instance
3: Perform one step of optimization with PI^2 independently on each instance:
$min_p E_p[l(\tau)]$ s.t. $D_{KL}(p(u_t|x_t)||\pi_{\theta(u_t|x_t)}) \leq \epsilon$
4: Train global policy with optimized controls using supervised learning:
$\pi_\theta \leftarrow \text{argmin}_\theta \sum_{i,t} D_{KL}(\pi_\theta(u_t|x_{i,t})||p(u_t|x_{i,t}))$
5: **end for**

The linear Gaussian controllers are given by $p(u_t|x_t) = \mathcal{N}(K_t x_t + k_t, C_t)$ and are parameterized by K_t, k_t, and a covariance matrix C_t. In this implementation, we instantiate K_t with the simple linear feedback control law calculated by LQR, and in subsequent iterations, we hold it constant. The feedforward term k_t and the covariance term C_t are randomly sampled from a Gaussian distribution. Each iteration, we generate N training samples in simulation, and then calculate the cost-to-go $S_{i,t}$ and softmax probabilities $P_{i,t}$ with temperature $\frac{1}{n}$ for each sample $i \in 1...N$:

$$S_{i,t} = S(\tau_{i,t}) = \sum_{j=t}^{T} l(x_{i,j}, u_{i,j}), P_{i,t} = \frac{e^{-\frac{1}{n}S_{i,t}}}{\sum_{i=1}^{N} e^{-\frac{1}{n}S_{i,t}}} \quad (3)$$

In the next iteration, we update k_t and the covariance matrix C_t:

$$k_t^{new} = \sum_{i=1}^{N} P_{i,t} k_{i,t}$$

$$C_t^{new} = \sum_{i=1}^{N} P_{i,t}(k_{i,t} - k_t^{new})(k_{i,t} - k_t^{new})^\top$$

Preliminary results on rough terrain demonstrate that this method outperforms MDGPS as shown in Figure 9.

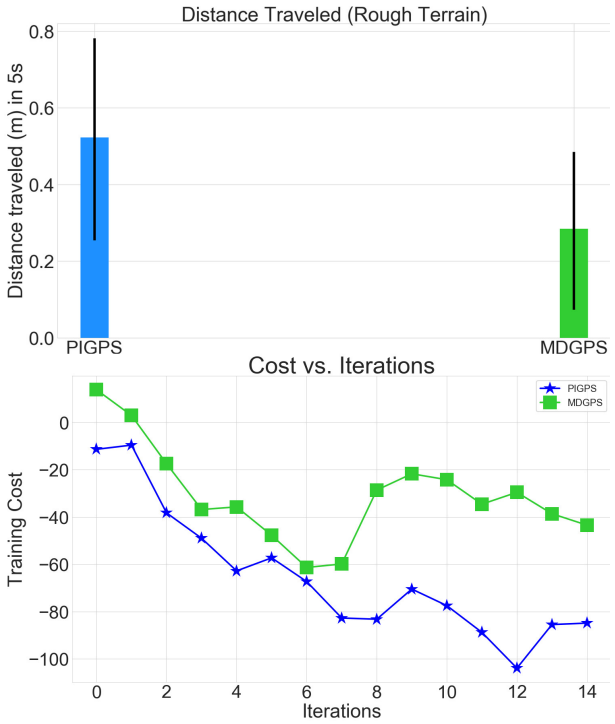

Fig. 9. PIGPS vs MDGPS locomotion on uneven terrain. *Top*: Distance traveled at test-time during one 5s roll-out. Notice that PIGPS travels approximately 2x as far as MDGPS. *Bottom*: Training costs by iterations. Notice that PIGPS converges faster than MDGPS on uneven terrains characterized by discontinuous dynamics.

VII. CONCLUSIONS AND FUTURE WORK

We present evidence that in the domain of tensegrity robotics, as characterized by complex, highly-nonlinear dynamics, we can effectively learn locomotion gaits even in low-dimensional observation spaces. Further, we analyze neural networks as expressive models able to map directly from these low-dimensional observation spaces to favorable actions. We present evidence, contrasting performance of deep neural network policies working in various sizes and types of observation spaces. The most restrictive observation spaces are so small that it would be infeasible to use them to learn the system dynamics. However, despite the observation's limited informational content, the network is still able to learn an effective mapping.

Furthermore, the implications of our findings motivate research of novel dynamics-free and observation-to-action-based deep reinforcement learning algorithms in the tensegrity domain. Finally, we showed initial promise towards this end by demonstrating preliminary success of PIGPS in unpredictable locomotion environments.

The new generation of tensegrity robot TT-5 is now under development at the BEST Lab. This new prototype will allow us to reliably conduct hardware tests. We hope to deploy these learned policies with limited sensory inputs into TT-5 in the near future, thus further validating our results.

ACKNOWLEDGMENT

The authors gratefully acknowledge funding for this research through NASA Early State Innovation grant NNX15AD74G, and support from NASA Ames Intelligent Robotics Group. We also thank other graduate or undergraduate researchers at BEST lab for setting up hardware tests and valuable feedback on this manuscript.

REFERENCES

[1] K. Kim, A. K. Agogino, A. Toghyan, D. Moon, L. Taneja and A. M. Agogino, Robust learning of tensegrity robot control for locomotion through form-finding, in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, sep 2015.

[2] K. Kim, L. Chen, B. Cera, M. Daly, E. Zhu, J. Despois, A. K. Agogino, V. SunSpiral, and A. M. Agogino, Hopping and rolling locomotion with spherical tensegrity robots, in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, oct 2016.

[3] A. Iscen, A. Agogino, V. SunSpiral, and K. Tumer, Flop and roll: Learning robust goal-directed locomotion for a tensegrity robot, in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, sep 2014.

[4] K. Caluwaerts, J. Bruce, J. M. Friesen and V. SunSpiral, State estimation for tensegrity robots, in 2016 IEEE International Conference on Robotics and Automation. IEEE, may 2016.

[5] J. Pinaud, M. Masic and R. E. Skelton, Path planning for the deployment of tensegrity structures, in 2003 Smart Structures and Materials: Modeling, Signal Processing, and Control. SPIE, 2003.

[6] A. Sabelhaus, J. Bruce, K. Caluwaerts, P. Manovi, R. Firoozi, S. Dobi, A. M. Agogino, and V. SunSpiral, System design and locomotion of SUPERball, an untethered tensegrity robot, in 2015 IEEE International Conference on Robotics and Automation. IEEE, may 2015.

[7] R. Skelton, R. Adhikari, J.-P. Pinaud, Waileung Chan, and J. Helton, An introduction to the mechanics of tensegrity structures, in 40th IEEE Conference on Decision and Control. IEEE, 2001.

[8] R. E. Skelton and M. C. de Oliveira, Tensegrity systems. Springer, 2009.

[9] A. K. Agogino, V. SunSpiral, and D. Atkinson, "Super Ball Bot structures for planetary landing and exploration", NASA Innovative Advanced Concepts (NIAC) Program, Final Report, Jul. 2013.

[10] J. Bruce, K. Caluwaerts, A. Iscen, A. P. Sabelhaus, and V. SunSpiral, Design and evolution of a modular tensegrity robot platform, in 2014 IEEE International Conference on Robotics and Automation. IEEE, may 2014.

[11] K. Kim, A. K. Agogino, D. Moon, L. Taneja, A. Toghyan, B. Dehghani, V. SunSpiral, and A. M. Agogino, Rapid prototyping design and control of tensegrity soft robot for locomotion, in 2014 IEEE International Conference on Robotics and Biomimetics. IEEE, 2014.

[12] R. E. Skelton, Dynamics and control of tensegrity systems, in IUTAM Symposium on Vibration Control of Nonlinear Mechanisms and Structures. Springer, 2005.

[13] R. E. Skelton and C. Sultan, Controllable tensegrity: A new class of smartstructures, in Smart Structures and Materials 97. International Society for Optics and Photonics, 1997.

[14] M. Zhang, X. Geng, J. Bruce, K. Caluwaerts, M. Vespignani, V. SunSpiral, P. Abbeel and S. Levine, Deep Reinforcement Learning for Tensegrity Robot Locomotion, in 2017 IEEE International Conference on Robotics and Automation. IEEE, may 2017.

[15] S. Levine and P. Abbeel, Learning contact-rich manipulation skills with guided policy search, in 2015 IEEE International Conference on Robotics and Automation. IEEE, 2015.

[16] W. Montgomery and S. Levine, Guided Policy Search as Approximate Mirror Descent, in Advances in Neural Information Processing Systems. NIPS, 2016.

[17] S. Levine and P. Abbeel, Learning Neural Network Policies with Guided Policy Search under Unknown Dynamics, in Advances in Neural Information Processing Systems. NIPS, 2014.
[18] S. Levine and V. Koltun, Guided Policy Search International Conference on Machine Learning, ICML, 2013.
[19] C. Finn, M. Zhang, J. Fu, W. Montgomery, X. Tan, Z. McCarthy, B. Stadie, E. Scharff, S. Levine, "Guided Policy Search Code Implementation". rll.berkeley.edu/gps 2016.
[20] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, Trust region policy optimization, in International Conference on Machine Learning. ICML, 2015.
[21] NASA Tensegrity Robotics Toolbox. https://github.com/NASA-Tensegrity-Robotics-Toolkit/NTRTsim 2015.
[22] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver,and D. Wierstra, Continuous control with deep reinforcement learning, in International Conference on Learning Representations. ICLR, 2016.
[23] M. Cutler, T. J. Walsh, and J. P. How, Real-world reinforcement learning via multifidelity simulators, IEEE Transactions on Robotics, vol. 31, 2016.
[24] S. Levine, C. Finn, T. Darrell, and P. Abbeel, End-to-end training of deep visuomotor policies, Journal of Machine Learning Research. JMLR, 2016.
[25] UC Berkeley Deep Reinforcement Learning Course Notes (CS294-112), http://rll.berkeley.edu/deeprlcourse/ Spring 2017.
[26] L. Chen, K. Kim and A. Agogino Soft Spherical Tensegrity Robot Design Using Rod-Centered Actuation and Control, in ASME International Design Engineering Technical Conference, IDETC, 2016.
[27] I. Mordatch, K. Lowrey, and E. Todorov, Ensemble-CIO: Full-body dynamic motion planning that transfers to physical humanoids, in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, oct 2015.
[28] J. Peters and S. Schaal, Reinforccement learning of motor skills with policy gradients, in Neural Networks, vol. 21, 2008.
[29] J. Peters, K. Mulling, and Y. Altun, Relative entropy policy search, in AAAI Conference on Artificial Intelligence, IJCAI, 2010.
[30] R. Moto, Tensegrity: Structural Systems of the Future. Kogan, 2003.
[31] M. Arsenault and C. M. Gosselin, "Kinematic and Static Analysis of a Three-degree-of-freedom Spatial Modular Tensegrity Mechanism," IJRR, vol. 27, no. 8, pp. 951-966, 2008.
[32] Y. Chebotar, M. Kalakrishnan, A. Yahya, S. Schaal, S. Levine, Path Integral Guided Policy Search, in 2017 IEEE International Conference on Robotics and Automation. IEEE, 2017.