Inferring Task-Space Central Pattern Generator Parameters for Closed-loop Control of Underactuated Robots

Nathan D. Kent[1], Raunaq M. Bhirangi[2], Matthew Travers[2], and Thomas M. Howard[1]

***Abstract*— The complexity associated with the control of highly-articulated legged robots scales quickly as the number of joints increases. Traditional approaches to the control of these robots are often impractical for many real-time applications. This work thus presents a novel sampling-based planning approach for highly-articulated robots that utilizes a probabilistic graphical model (PGM) to infer in real-time how to optimally modify goal-driven, locomotive behaviors for use in closed-loop control. *Locomotive behaviors* are quantified in terms of the parameters associated with a network of neural oscillators, or rather a central pattern generator (CPG). For the first time, we show that the PGM can be used to optimally modulate different behaviors in real-time (i.e., to select of optimal choice of parameter values across the CPG model) in response to changes both in the local environment and in the desired control signal. The PGM is trained offline using a library of optimal behaviors that are generated using a gradient-free optimization framework.**

I. Introduction

The highly-articulated nature of legged robots presents a challenge as to how to reason, in real-time, over the high-dimensional spaces that underlie their various behaviors. In particular, the coordination of the limbs and their corresponding joints quickly scales in complexity. One of the most popular approaches for addressing this challenge employs randomized sampling-based planning to generate a motion plan that is subsequently executed with the help of online feedback controllers that provide regulation around the desired plan. Unfortunately, conventional techniques often do not scale efficiently as the size of the search space increases and requires the robot to comprise between optimality and reaction time. Dense sampling impedes the robot's ability to respond to abrupt changes in the environment but allows the sampling-based motion planner to search over a diverse space of actions. Coarse sampling increases the cycle rate of the sampling-based planner at the expense of selecting from only a few (and potentially sub-optimal) actions. For the control of a highly-articulated robot, such as the hexapod in Figure 1, it is computationally prohibitive to search over all possible actions for a near-optimal solution using on-board computation. Thus, this work presents an alternative approach for closed-loop control of highly-articulated robots by encoding information about the environment, motion commands, and robot kinematics in a probabilistic graphical model (PGM) to infer parameterized motion primitives for path following control.

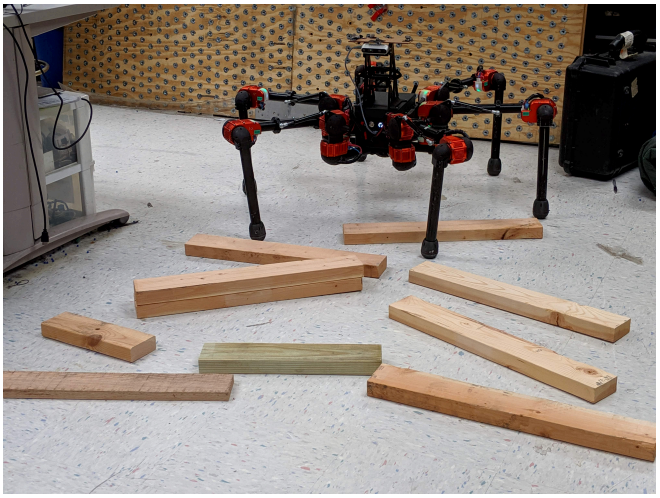


Fig. 1: The hexapod robot on which the simulation-based experiments of the proposed path following control architecture are performed.

The PGM exploits conditional independence assumptions for efficient search in a space of parameterized central pattern generators (CPGs). Factors in this graphical model are conditioned upon estimates of the robot's kinematics, the local environment, and the desired locomotion and exploit the natural hierarchy in the robot's kinematic model. This model adapts search techniques from graphical model-based approaches to grounded language communication [1], [2] where conditional independence assumptions improve the efficiency of natural language symbol grounding. The outputs of the PGM are represented in terms of CPGs parameters that implicitly define cyclic locomotive behaviors and can be modulated in terms of step width, step height, and the locomotive gait pattern that reduce the number of variables over which to reason. This model first infers step width and height parameters for each limb and then infers parameters that coordinate limb behaviors conditioned on these values. Data sets generated from gradient-free optimization performed offline are used to derive a family of near-optimal motion primitives for different environment conditions and robot behaviors. These optimized primitives were used to train factors in a probabilistic graphical model that infers the maximum likelihood set of PGM parameters to control robots in simulated environments.

By providing the desired locomotion from a path planner

[1] Nathan D. Kent and Thomas M. Howard are with the Robotics and Artificial Intelligence Laboratory, University of Rochester, Joseph C Wilson Blvd. Rochester, NY 14627, USA. nkent2@cs.rochester.edu, thoward@ece.rochester.edu

[2] Raunaq M. Bhirangi and Matthew Travers are with the Biorobotics Lab, Carnegie Mellon University, 5000 Forbes ave, Pittsburgh, PA 15213, USA. <rbhirang, mtravers>@andrew.cmu.edu

and the environmental information from the robot's on-board sensors, the PGM is used as a closed-loop path following controller for motion control through complex, unstructured environments. This controller is the first example of this novel framework for CPG parameter inference being applied for path following control in real-time. Experimental results presented later in the paper quantify the performance of learned models for parameterized CPGs inference for following predefined paths across flat and obstacle covered terrains.

II. Background

A. Central Pattern Generators

The foundation of the planning and control framework put forth in this paper is an offline-learned, data set of locomotive motion primitives. The primitives are parameterized in terms of a specific central pattern generator (CPG) model. CPGs are neural circuits that produce cyclical patterns of neural activity that have been observed within a variety of biological organisms. Inspired by biologically-motivated examples, a variety of CPGs models have been deployed as the basis for controlling of articulated locomotive robots [3], [4], [5], [6].

The CPG model developed in this work is inspired by Sartoretti et al. [6] and Yu et al. [7]. More specifically, we draw on these prior works to define a CPG model in terms of a system of coupled oscillators whose joint dynamics are represented as a set of ordinary differential equations. We assume that each oscillator governs the motion of one of the 3DoF legs on the hexapod robot shown in Figure 1.

More specifically, this work assumes that the integral curve associated with the i^{th} oscillator's dynamics physically define the trajectory of the foot on the i^{th} leg in the task-space of the system. Drawing inspiration from Sartoretti et al. [6] we constrain the shape of these trajectories to be super-elliptical, i.e., the trajectories are defined by

$$H_i(x,y) = \left|\frac{x-c_{x,i}}{a_i}\right|^{d_i} + \left|\frac{y-c_{y,i}}{b_i}\right|^{d_i} - 1 = 0 \quad (1)$$

where $(c_{x,i}, c_{y,i})$ is the geometric centre of the ellipse defined in a frame obtained by rotating the robot's body frame (that is fixed to the robot's body and located at $(x,y) = (0,0)$) by 90 degrees about the x-axis; a_i and b_i are the lengths of the semi-major/minor axes with d_i specifying the curvature of the super ellipse.

In addition to the shape of the trajectory traced out by each foot, locomotive motion primitives in this work are quantified in terms of the relative phase differences between the individual limbs on the robot. More specifically, we define the function $f_i : \mathbb{R}^2 - \{0\} \to S^1 (\subset \mathbb{R}^2)$, that maps the output of oscillator i to the unit circle,

$$f_i(x,y) = \begin{bmatrix} \operatorname{sgn}(x-c_{x,i}) \left|\dfrac{x-c_{x,i}}{r a_i}\right|^{d_i/2} \\ \operatorname{sgn}(y-c_{y,i}) \left|\dfrac{y-c_{y,i}}{r b_i}\right|^{d_i/2} \end{bmatrix} \quad (2)$$

where $r = \left(\left|\frac{x-c_{x,i}}{a_i}\right|^{d_i} + \left|\frac{y-c_{y,i}}{b_i}\right|^{d_i}\right)^{1/d_i}$. Every point in $\mathbb{R}^2$ is assigned a phase corresponding to its position on a super-ellipse concentric with and belonging to the same family as $H_i(x,y) = 0$. The absolute phase of the i^{th} oscillator is defined as the four-quadrant tangent inverse of $f_i(x_i, y_i)$. Defining ψ_{ij} to be the desired phase difference between limbs i and j, robust phase coupling between limbs is achieved by including penalty terms that maintain specified phase differences between oscillators such that,

$$\begin{bmatrix}\dot{x}_i \\ \dot{y}_i\end{bmatrix} = \omega_i \hat{t}_i + (1 - H_i(x_i(t), y_i(t)))\hat{n}_i + \sum_j \lambda_{ij}\left[f_i^{-1}(R(\psi_{ij}) f_j(x_j, y_j)) - [x_i, y_i]^T\right], \quad (3)$$

where $f_i^{-1} : S^1 \to \{(x,y) \in \mathbb{R}^2 : H_i(x,y) = 0\}$ is the inverse of f_i restricted to the domain of points that lie on the limit cycle $H_i = 0$, $[\lambda_{ij}]$ quantifies the strength of the influence of oscillator j on oscillator i, and $R(\psi_{ij}) \in SO(2)$ is a rotation matrix corresponding to a counterclockwise rotation by ψ_{ij}. Additionally, $\hat{n}_i$ is the unit vector in the normal direction relative to the super ellipse scaled by a constant ω_i and $\hat{t}_i$ is the unit vector in the tangent direction relative to the super ellipse.

B. Probabilistic Graphical Models

Working in the context of unstructured terrains naturally requires reasoning over a variety of forms of uncertainty and one of the most common methods for decision making under uncertainty is through the use of probabilistic graphical models (PGMs). As these models can exploit the conditional independence present within the task, they have found great success in the context of natural language processing (NLP). Methods such as Generalized Grounding Graph (G^3) [1] and variations of Distributed Correspondence Graphs (DCGs) [2], [8], [9], [10] infer distributions of symbols that represent objects, spatial relationships, constraints, trajectories, etc. for the individual phrases that form a natural language utterance in the context of the model of the perceived environment. The DCG variations assume conditional independence across language and symbolic constituents in order to efficiently infer an approximation of the probability distribution of expressed correspondences. This work directly draws inspiration from G^3 and DCG frameworks in that the mathematical framework for efficiently inferring distributions of symbols can be utilized to efficiently infer distributions of CPG parameters. Unlike those works, however, this work represents factors using neural networks in place of log-linear models to more efficiently and effectively model real-valued quantities.

This work also represents an expansion on the work presented by Chavali et al. [11]. Whereas the previous framework utilized a joint-space CPG in order to execute an open-loop controller, this work utilizes a new Cartesian-space CPG formulation and utilizes the on-board sensors to close the control loop. Additionally, this work is novel in its use of neural networks to model the conditional probability distributions whereas the work done by Chavali

et al. adapted log-linear models used in natural language symbol grounding.

The remainder of this paper is organized as follows. In Section III a high-level overview of the full closed-loop controller is provided, detailing the proposed modifications to the previous framework. Section IV specifies the experimental setup used to evaluate the proposed system with the results being presented in Section V. Current limitations and future work will be discussed in Section VI.

III. Technical Approach

The framework we propose involves two main components. First, we formulate a model for inferring task-space CPG parameter distributions to control the limbs of highly articulated robots from variables representing the robot's environment, model, and desired behavior. Second, we present an architecture for path following control that uses the task-space CPG parameter inference model for motion control of a hexapod robot.

A. CPG Parameter Inference

To infer a distribution of CPG parameters for executing behaviors in the closed-loop controller, we extend the framework described in [11] to infer the optimal set of eighteen parameters $\mathcal{P}^*$ for the new CPG model from the environment model ($\mathcal{E}$), behavior ($\mathcal{B}$), and robot model ($\mathcal{M}$). This approach deviates from the method outlined in [11] in two ways. First, this approach does not assume that all CPG parameters are conditionally independent. Specifically, this approach makes the assumption that a level of symmetry can be exploited in the model, such that the a_i values of all the right legs can be represented by a single a_1 (with left legs being a_2) and all the b_i values can be modeled as a single uniform step height b. Additionally, the desired phase differences ψ_{ij} are restricted to only the phase difference from the preceding limb and are expressed as $k_1 \dots k_6$. The offset of the body frame from the ground is defined in a single parameter h. Second, the model explored in this paper models the conditional probabilities in the factor graph using neural networks instead of log-linear models that utilized human engineered features. Weights for the neural network are trained from a set of examples generated using the genetic algorithm-based training procedure described in [11] but adapted for the operational space CPG parameters defined in the prior section. We formulate inference now as the search for the most probable CPG parameters given the behavior, environment, and robot model:

$$\mathcal{P}^* = \underset{k_i, a_1, a_2, b, h \in \mathbb{R}}{\arg\max} \prod_{i=1}^{6} p(k_i)\, p(h)\, p(a_2)\, p(a_1)\, p(b) \quad (4)$$

First, we define the probability distributions for each of these parameters as a function of their dependent variables. In this formulation of the model, we assume that probability distribution of CPG parameters a_1 and b depend only on the environment, behavior, and robot model. We apply the theory of total probability to formally define the resulting probability distributions in Equations 5 and 6.

$$\begin{aligned} p(a_1) = \int\int\int & p(a_1|\mathcal{E},\mathcal{B},\mathcal{M})\, p(\mathcal{B}|\mathcal{E},\mathcal{M}) \\ & p(\mathcal{E}|\mathcal{M})\, p(\mathcal{M})\, d\mathcal{B}\, d\mathcal{E}\, d\mathcal{M} \end{aligned} \quad (5)$$

$$\begin{aligned} p(b) = \int\int\int & p(b|\mathcal{E},\mathcal{B},\mathcal{M})\, p(\mathcal{B}|\mathcal{E},\mathcal{M}) \\ & p(\mathcal{E}|\mathcal{M})\, p(\mathcal{M})\, d\mathcal{B}\, d\mathcal{E}\, d\mathcal{M} \end{aligned} \quad (6)$$

We can also define the probability distributions for CPG parameters a_2 and h using the same approach. This model assumes that a_2 and h depend on the environment, behavior, robot model, and the probability distributions of a_1 and b. These expressions are defined in Equations 7 and 8.

$$\begin{aligned} p(a_2) = \int\cdots\int_5 & p(a_2|a_1,b,\mathcal{E},\mathcal{B},\mathcal{M}) \\ & p(a_1|b,\mathcal{B},\mathcal{E},\mathcal{M})\, p(b|\mathcal{B},\mathcal{E},\mathcal{M}) \\ & p(\mathcal{B}|\mathcal{E},\mathcal{M})\, p(\mathcal{E}|\mathcal{M})\, p(\mathcal{M})\, da_1\, db\, d\mathcal{B}\, d\mathcal{E}\, d\mathcal{M} \end{aligned} \quad (7)$$

$$\begin{aligned} p(h) = \int\cdots\int_5 & p(h|a_1,b,\mathcal{E},\mathcal{B},\mathcal{M}) \\ & p(a_1|b,\mathcal{B},\mathcal{E},\mathcal{M})\, p(b|\mathcal{B},\mathcal{E},\mathcal{M}) \\ & p(\mathcal{B}|\mathcal{E},\mathcal{M})\, p(\mathcal{E}|\mathcal{M})\, p(\mathcal{M})\, da_1\, db\, d\mathcal{B}\, d\mathcal{E}\, d\mathcal{M} \end{aligned} \quad (8)$$

Lastly, we assume that the probability distribution of CPG parameters $k_1 \dots k_6$ depend on the the environment, behavior, robot model, and the probability distributions of a_1, a_2, b, and h. This model assumes that each k_i is dependent on all k_j where $j < i$. The probability of k_1 is defined in Equation 9 with the values of all other k_i following a similar pattern but also expressing conditional dependence on the k_j values expressed in the graphical model.

$$\begin{aligned} p(k_1) = \int\cdots\int_7 & p(k_1|a_1,a_2,b,h,\mathcal{E},\mathcal{B},\mathcal{M}) \\ & p(a_1|a_2,b,h,\mathcal{E},\mathcal{B},\mathcal{M})\, p(a_2|b,h,\mathcal{E},\mathcal{B},\mathcal{M}) \\ & p(b|h,\mathcal{E},\mathcal{B},\mathcal{M})\, p(h|\mathcal{E},\mathcal{B},\mathcal{M})\, p(\mathcal{B}|\mathcal{E},\mathcal{M}) \\ & p(\mathcal{E}|\mathcal{M})\, p(\mathcal{M})\, da_1\, da_2\, db\, dh\, d\mathcal{B}\, d\mathcal{E}\, d\mathcal{M} \end{aligned} \quad (9)$$

We simplify these expressions first by assuming that the probability of behavior $\mathcal{B}$ is conditionally independent from the probabilities of the environment model $\mathcal{E}$ and robot model $\mathcal{M}$ and the probability of environment $\mathcal{E}$ is conditionally independent of the robot model $\mathcal{M}$. We approximate the inference procedure defined in Equation 4 by performing beam search in a factor graph that expresses the conditional dependencies of our model. This model is illustrated in Figure 2. Beam search is used to efficiently generate a distribution of effective CPG parameter sets. The resulting distribution is passed onto the path following controller where the parameters would be used to control CPGs with a priority towards parameter distributions that are most probable.

B. Feedback Control

Following arbitrary paths in non-trivial environments with underactuated robots is difficult because of the challenges

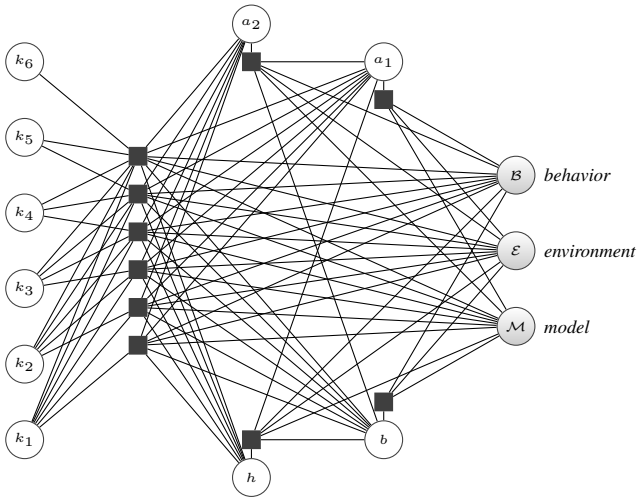


Fig. 2: An illustration of the factor graph used to infer a distribution of parameters for this CPG model. The model exploits conditional independence assumptions to efficiently infer the most likely set of CPG parameter values $k_1 \dots k_6, a_1, a_2, b, h$ from models of the environment ($\mathcal{E}$), behavior ($\mathcal{B}$), robot ($\mathcal{M}$).

of balancing safety and stability and minimizing cross-track error. We propose the block diagram illustrated in Figure 3 for path following control with underactuated robots using a learned model of CPG parameters that depend on estimates of the environment, robot, and behavior. In this architecture the robot provides an estimate of the current pose (x,y,ψ) and a pointcloud from LiDAR and/or RGB-D cameras. The terrain estimator compresses this information into a parametric representation of terrain roughness encapsulated in the random variable $\mathcal{E}$. The path following controller uses the robot's current pose and a path that is either defined a priori or continuously updated by a path planning module to estimate a probability distribution for the robot's desired behavior $\mathcal{B}$. The robot may also provide an estimate of it's internal configuration $\mathcal{M}$ in order to model degraded mobility or intrinsic deformations. The CPG parameter inference, which is constructed from neural networks trained offline from data sets synthesized from genetic algorithm-based search, estimates the most likely set of CPG parameters $\mathcal{P}^*$ from this information using the beam search method described in Section III-A and transmits those parameters to the central pattern generator module. That process then converts the parameters into the Cartesian position of the n limbs ($x_{1:n}$,$y_{1:n}$,$z_{1:n}$). The inverse kinematics module then converts these values into the m joint angles ($\theta_{1:m}$) defined by the robot's kinematic model. The robot subscribes to these angles and drives each actuator to follow the desired joint angle profiles.

IV. Experimental Design

To experimentally validate the performance of the proposed model, we designed, implemented, and experimented

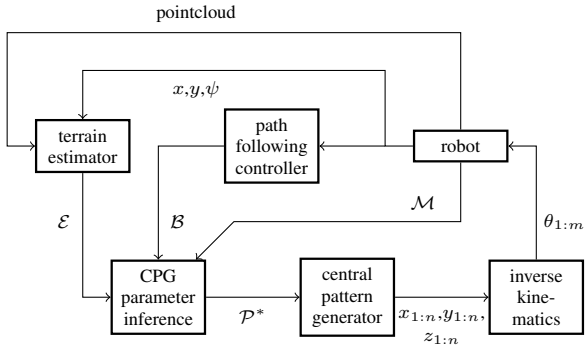


Fig. 3: The block diagram for path following control using the CPG parameter inference.

with the graphical model-based approach to CPG parameter inference on a simulated model of the hexapod robot illustrated in Figure 1. In order to learn pairs of CPG parameters with environment, robot, and behaviors, we adapted the genetic algorithm procedure described in [11] for task-space CPG parameters and new environments, behaviors, and kinematic model. The obstacle heights were one of $0.0m$ (flat ground), $0.01m$, $0.025m$, $0.035m$, $0.05m$, and $0.1m$ and were scattered through the terrain using a Poisson Random Process. Illustrations of the non-zero obstacle height environment models are shown in Figure 4. Commands were also provided to have the robot locomote with curvatures of $0m^{-1}$, $0.2m^{-1}$, $0.286m^{-1}$, $0.4m^{-1}$, and $1.0m^{-1}$. Individuals were scored via the line integral of their path through the vector field resulting from normalizing F:

$$H(x,y) = c^2x^2 + (cy-1)^2 \tag{10}$$

$$W_x(x,y) = (1 - H(x,y))\frac{\partial H}{\partial x}(x,y) \tag{11}$$

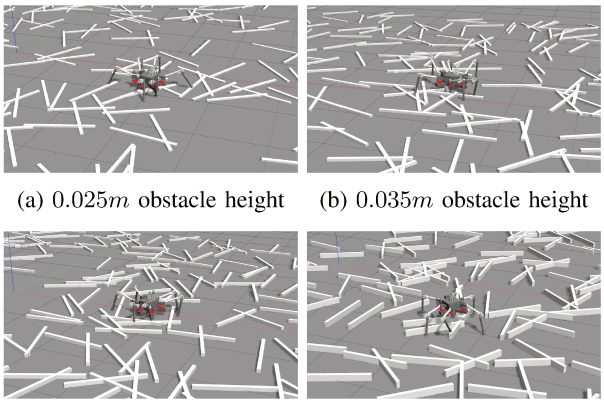
$$W_y(x,y) = (1 - H(x,y))\frac{\partial H}{\partial y}(x,y) \tag{12}$$

$$F(x,y) = \left\langle \frac{\partial H}{\partial y}(x,y) + \frac{W_x(x,y)}{2c}, -\frac{\partial H}{\partial x}(x,y) + \frac{W_y(x,y)}{2c} \right\rangle \tag{13}$$

The parameter c in these expressions is the desired curvature of the robot's motion. Several example vector fields generated using this equation are visualized in Figure 5.

Using 100 individuals evolved over 200 generations on 10 random obstacle fields, 268 data points were collected to train the PGM. From this data we extracted pairs of environments, behaviors, robot models, and probability distributions of CPG parameters that were used to train neural networks that represent factors in the probabilistic graphical model. In contrast to the log-linear model approach described in [11] that learns a single model for all factors, individual networks were trained for each factor representing CPG parameters ($k_1 - k_6$, a_1, a_2, h, and b) in the graphical model. The neural network used in the simulation experiments consisted of 3 layers of 30 hidden units each.

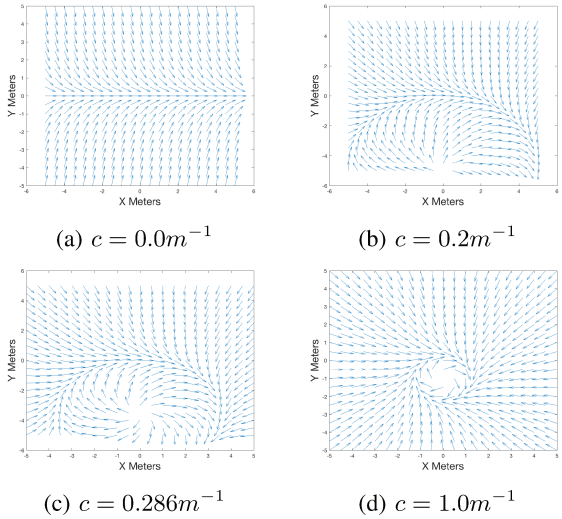
To evaluate the learned models, the trained CPG parameter



(a) $0.025m$ obstacle height (b) $0.035m$ obstacle height

(c) $0.05m$ obstacle height (d) $0.1m$ obstacle height

Fig. 4: Example terrains used in simulations of CPG parameters within the genetic algorithm.



(a) $c = 0.0m^{-1}$ (b) $c = 0.2m^{-1}$

(c) $c = 0.286m^{-1}$ (d) $c = 1.0m^{-1}$

Fig. 5: Example vector fields used for evaluating individuals within the genetic algorithm.

inference model was evaluated on flat terrain and non-flat terrain across three predefined paths. Terrain information was fed to the controller via a 2.5D height map generated using the on-board depth camera. The non-flat terrain consists of increasing obstacle heights placed using a Poisson Random Process and is illustrated in Figure 6. Terrain roughness was estimated by averaging the height of the non-empty cells within $0.5m$ of the robot's current position.

Similar to other studies that measure relative path following controller performance [12], we quantify path tracking performance as a function of the average distance between the desired and executed paths. The nearest distance to the desired path is calculated at a rate of 1 Hz during each experiment. We adapted the Pure Pursuit algorithm [13] for path following control with a fixed lookahead distance of $1m$ from the center of the robot. The neural network model is evaluated 10 times on 6 combinations of paths and terrains (3 paths $\times$ 2 terrains) for a total of 60 simulated experiments.

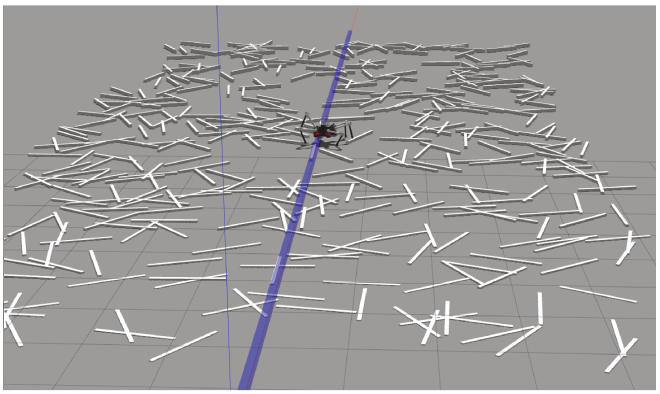


Fig. 6: A illustration of the robot navigating the non-flat terrain for path 1 of the path following control experiments.

V. Experimental Results

Following the experiment design outlined in the previous section, we recorded the average cross-track error for following the three predefined paths in flat and non-flat terrains illustrated in Figures 7 and 8 in Tables I–III.

	path 1	path 2	path 3
Flat Terrain	0.35	0.45	0.49
Non-flat Terrain	0.81	0.66	0.58

TABLE I: A comparison of average cross-track error on flat and non-flat terrain using all evaluations, showing that the proposed framework kept the robot within $1m$ on average for all combinations of paths and terrains.

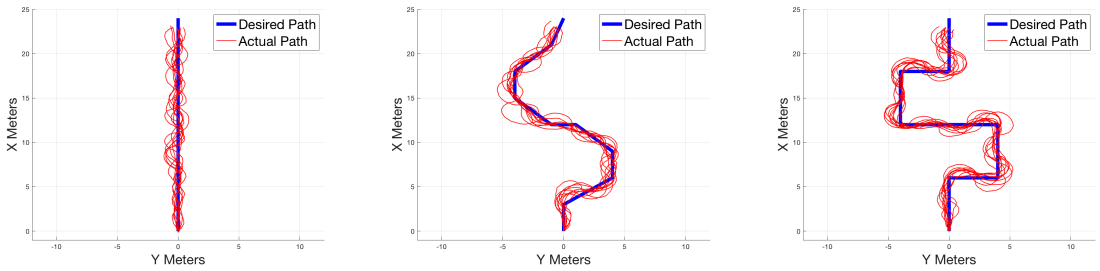
	path 1	path 2	path 3
Flat Terrain	0.34	0.44	0.47
Non-flat Terrain	0.61	0.59	0.52

TABLE II: A comparison of the average cross-track error on flat and non-flat terrain with the worst performing evaluation removed.

	path 1	path 2	path 3
Flat Terrain	0.35 (10)	0.45 (10)	0.49 (10)
Non-flat Terrain	0.59 (5)	0.37 (1)	N/A (0)

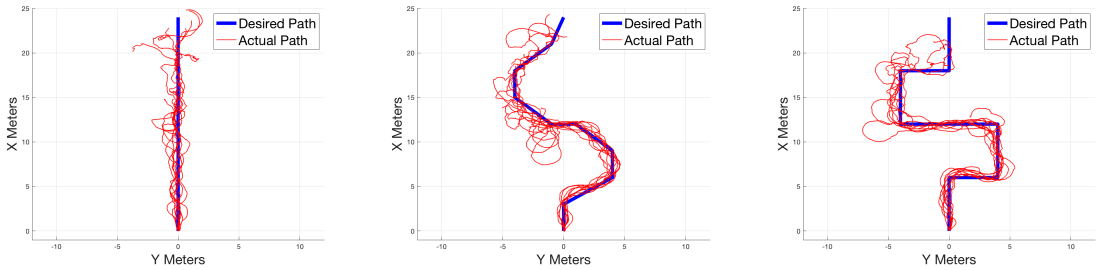
TABLE III: A comparison of the average cross-track error on flat and non-flat terrain using only the evaluations which reach the end goal.

As expected, we observe that the path following performance of the learned CPG parameters is better in the flat terrain than the non-flat terrain. The robot was able to successfully navigate to the end of all predefined paths in flat terrain but only a subset of the non-flat terrains. We hypothesize that the robot had difficulty navigating across the tall obstacles at the end of the non-flat course because the terrain roughness estimate provided to the CPG parameter inference model was too coarse and the restriction that only



(a) path 1 (b) path 2 (c) path 3

Fig. 7: Path following performance of inferred CPG parameters across 3 path shapes on flat terrain.



(a) path 1 (b) path 2 (c) path 3

Fig. 8: Path following performance of inferred CPG parameters across 3 path shapes on non-flat terrain.

two parameters (a_1 and a_2) define the step width of all six legs. In circumstances where small perturbations to the initial state or CPG parameters cause significant deviations in the projected motion of the robot, trajectory optimization techniques that more carefully plan motions through complex terrain may be needed for robust path following control.

VI. Conclusion

This paper describes a novel approach to control of highly-articulated robots that exploits learning in two ways. First, genetic algorithms are used to find task-space CPG parameters that provide effective motion performance across different terrain shapes, kinematic configurations, and locomotion behaviors. Second, weights for neural networks are learned from these CPG parameters to represent a conditional probability in the context of these variables. These neural networks represent factors in a probabilistic graphical model that is used to efficiently infer distributions of task-space CPG parameters. In contrast to the model presented in [11], the factor graphs implemented and experimentally validated in this paper do not assume that all parameters are conditionally independent. In simulation experiments the most likely CPG parameters inferred by the PGM are guided by a path following controller that adjusts the desired curvature of the robot's current motion to follow predefined paths over flat and non-flat terrains.

We recognize several limitations of the current approach. First, we experimentally observed that path tracking performance was dependent on hyperparameters of the neural networks that model the conditional probabilities of CPG parameters in the factor graph. While these models did not require human engineered features to train log-linear models of the natural language symbol grounding models that influenced this formulation, we hypothesize that larger training sets that better represent the conditional dependence of CPG parameters on environment, model, and behavior would result in more robust training performance. The current investigation also does not survey variations of the conditional dependence assumptions of CPG parameters and evaluate the relative performance of those graphical models. Although the choice of model to represent the factors does not impact the technical contribution of this paper, this model provides a useful data point for future work as other alternative models to learn conditional distributions of continuous variables are explored. Additionally, we intend to expand the training data to include more variation of learned parameters and utilize distributions of terrain properties, behaviors, and kinematic model parameters during training and inference. Although the formulation enables inference of CPG parameters in environments where these random variables are represented by Gaussian distributions or particles, the experimental results presented here assume the most likely value for each of these model inputs and only a single kinematic model.

Acknowledgments

This work was supported in part by the National Science Foundation under grants IIS-1724000 and IIS-1723972.

References

[1] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, "Understanding natural language commands for

robotic navigation and mobile manipulation," in *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[2] R. Paul, J. Arkin, D. Aksaray, N. Roy, and T. M. Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot platforms," *International Journal of Robotics Research*, June 2018.

[3] A. Crespi and A. J. Ijspeert, "Online optimization of swimming and crawling in an amphibious snake robot," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 75–87, 2008.

[4] J. Conradt and P. Varshavskaya, "Distributed central pattern generator control for a serpentine robot," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 2003, pp. 338–341.

[5] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 819–824.

[6] G. Sartoretti, S. Shaw, K. Lam, N. Fan, M. Travers, and H. Choset, "Central pattern generator with inertial feedback for stable locomotion and climbing in unstructured terrain," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–5.

[7] H. Yu, H. Gao, L. Ding, M. Li, Z. Deng, and G. Liu, "Gait generation with smooth transition using cpg-based locomotion control for hexapod walking robot," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 9, pp. 5488–5500, Sep. 2016.

[8] T. M. Howard, S. Tellex, and N. Roy, "A natural language planner interface for mobile manipulators," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6652–6659.

[9] I. Chung, O. Propp, M. R. Walter, and T. M. Howard, "On the performance of hierarchical distributed correspondence graphs for efficient symbol grounding of robot instructions," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 5247–5252.

[10] R. Paul, J. Arkin, N. Roy, and T. M. Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators," in *2016 Robotics: Science and Systems Conference*, June 2016.

[11] R. A. K. Chavali, N. D. Kent, M. E. Napoli, T. M. Howard, and M. Travers, "Inferring distributions of parameterized controllers for efficient sampling-based locomotion of underactuated robots," in *American Controls Conference*, 2019.

[12] T. M. Howard, R. A. Knepper, and A. Kelly, "Constrained optimization path following of wheeled robots in natural terrain," in *Experimental Robotics. Springer Tracts in Advanced Robotics*, vol. 39. Springer, Jan. 2008, pp. 343–352.

[13] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-92-01, January 1992.