2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Energy Efficient Navigation for Running Legged Robots

Mario Y. Harper[1], John V. Nicholson[2], Emmanuel G. Collins, Jr.[2], Jason Pusey[3] and Jonathan E. Clark[2]

***Abstract*— Energy-efficient navigation is an important technology for mobile robots because of its potential to increase the operation time of the robot. In particular, when coupled with a dynamic legged quadruped, the need for energy savings is made more apparent as payloads are limited. Due to the complexity in modeling motion and power models of these robots, a new approach is necessary to effectively motion plan for these complex robots. We accomplish this by using Sampling-Based Model Predictive Optimization (SBMPO) which was extended for use on the LLAMA quadrupedal platform in simulation. SBMPO allows for direct generation of trajectories while using a heuristic-based search to speed up computations. This approach is shown to effectively motion plan while optimizing for energy consumption and maintaining the natural dynamics of the robot in a simulated environment.**

I. INTRODUCTION

Dynamic legged robotics provide numerous potential advantages when navigating through unstructured terrains due to their discrete foot placement and large ground/obstacle clearance [1], [2], [3], however, they also present unique challenges to modeling and motion planning. In particular, the natural dynamics of running, while having self stabilizing properties [4], lack an analytical physics-based model that can be evaluated in a computationally efficient manner.

Current approaches to navigation and motion planning on legged robots generally employ a sampling-based method, usually built on the RRT* algorithm [5], [6] or a heuristic search method [7]. Many robots taking part in the DARPA robotics challenge utilized both methods to ensure success [8], [9] and were able to complete a variety of tasks. Most of these implementations requires an invertible model [10] to control and plan for foot-placements, which when properly selected, provides stability and generates body velocities. However, these methods are limited to slower motions and have ignored the natural dynamics introduced by fast gait-based locomotion. Further, they do not consider kinematic, dynamic or power models that can be used to generate energy optimal trajectories [11].

Sampling-Based Model Predictive Optimization (SBMPO) provides benefits of computation speed provided by heuristics-based search methods [12], [13] combined with sampling-based search and can directly utilize kinodynamic models. The method works well with gait-based locomotion, allowing it to optimally generate a trajectory while maintaining the natural dynamics and self-stabilization properites of high-speed motion. Much like other heuristic search planners, SBMPO requires understanding of robot behavior to formulate an effective cost function that will guide the search towards the goal, something that is difficult due to the complexity of dynamic legged systems.

To address these model complexities, a data driven approach is used to learn a model of the relationship between control inputs to robot body-velocites and power consumption of a high speed quadruped. This learned kinodynamic-power model is then used within SBMPO to optimize energy efficient trajectories to save the robot significant power.

The remainder of the paper is organized as follows: We describe the motion planning algorithm in Section II-A, followed by the description of the quadrupedal platform LLAMA (see Fig 1) II-B. In section III the learned models are described. Finally in Sec. IV and Sec. V, we demonstrate the robot navigating through a set of canonical problems in simulation and conclude with a discussion of our findings and directions for future work.

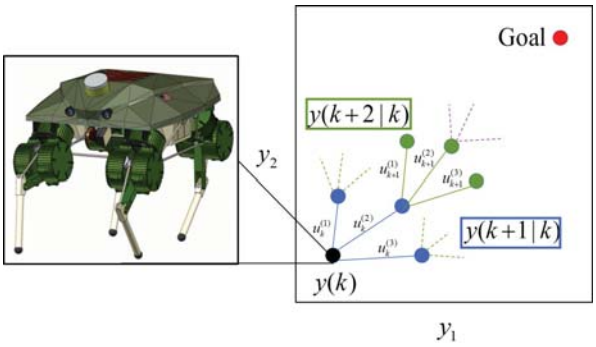


Fig. 1. Illustration of a tree resulting from the SBMPO sampling process. The graph is represented in the output space y. The branching occurs by propagating the sampled inputs u through the model. The result is both spatial discretization (i.e., the individual branches) and time discretization such that each layer corresponds to a given time, i.e., $y(k+i|k)$

II. BACKGROUND

A. SBMPO

Sampling Based Model Predictive Optimization (SBMPO) [14], [15] is a motion planning paradigm that generates optimal trajectories by taking advantage of both heuristic-based and sampling-based search methods while using kinodynamic and power models to constrain planned motions and/or compute motion costs. SBMPO directly generates a trajectory by sampling exclusively in the input space of any model, similar to the input-centric viewpoint of Model Predictive Control [16], returning

[1]M. Harper is with Scientific Computing, Florida State University
[2]J. Nicholson, E. Collins and J. Clark are with Mechanical Engineering, Florida A&M University-Florida State University
[3]J. Pusey is with the Army Research Laboratory

the series of control inputs corresponding to the optimal trajectory once the goal has been achieved. This optimization is accomplished using an heuristic-based search (A*-type) algorithm.

This algorithm is computationally fast when the heuristic function is well formulated, and allows the algorithm to be deployed in real-time. The model-centric design of SBMPO also allows it to consider the complex non-holonomic and holonomic constraints that are posed when considering a dynamic legged robot such as LLAMA. While existing algorithms to handle kinodynamic constraints exists [17], [18], these algorithms are generally sampling-based (RRT) and computationally slower than a heuristic-based search method [12], however, heuristic search methods have had difficulty in implemention on legged mobility robots due to the complexity in understanding robot behaivor.

SBMPO has been applied to a number of planning problems on different platforms, including autonomous underwater vehicles [19], skid-steered vehicles [20], and spacecraft [21]. SBMPO uses a model of the robotic system to generate a graph like the one shown in Figure 1. This algorithm also has the ability to optimize for unique and difficult cost functions. These are often based on complex dynamic models as in the case of energy efficient planning.

The general form of SBMPO is shown in Figure 2 and is comprised of an extended heuristics-based search algorithm like A*, in the case of this research, we employed LPA*. The major components of the SBMPO algorithm are as follows:

1) Sampler: Generate a set of inputs, $u(k)$ to the kinematic or dynamic model of the robot.
2) Model: Compute a new state of the robot $x(k)$ based on the sampled inputs $u(k)$ given by the sampler through a prediction model; This module generates a heuristic and cost associated with that new state which is needed by the optimizer.
3) Optimizer: Using a directed graph of the output space, a trajectory is optimized using the cost and heuristic by a LPA* algorithm.

These elements can be seen in Figure 2, where component parts of SBMPO are shown. The core of the optimization, LPA*, sends and receives information from the robot model. The cost (denoted as g) and heuristic (denoted as h) models can be interchanged to solve a wide array of problems. Energy efficient planning minimizes the total traversal energy (denoted as E) of the model, this is expressed through the cost function

$$g_{k+1} = \sum_{k=0}^{N-1} E_{k+i} \tag{1}$$

The heuristic function is defined by the lowest theoretical cost of energy that is needed to attain the goal. This is set as the euclidean distance to the goal measured by stride of the robot multiplied by the energy of the lowest-cost maneuver the robot is capable of, where V_{min} is the minimum possible forward velocity within the sampled domain:

$$h_{k+1} = distance(x_k, x_N)\frac{E_{min}}{V_{min}}. \tag{2}$$

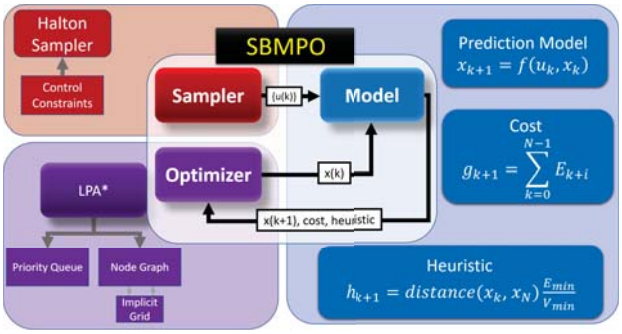


Fig. 2. SBMPO Elements. SBMPO is a motion planning paradigm that uses both sampling and heuristics. Samples from the control input space is generated and converted to robot state (position, orientation, ect.) which is stored in a graph. LPA* optimizes for the graph and returns the trajectory of control inputs upon completion.

B. Robotic Platform

1) Design of Simulation: Simulation of the quadruped robot LLAMA (Figure 1), was conducted in Matlab's Simscape Multibody framework. This simulated robot has a mass of roughly 64 kg and dimensions of 0.7 m x 0.4 m x 0.5 m while standing. Each leg has 3 degrees of freedom and utilizes an asymmetric 5-bar linkage design.

The contacts of each leg were defined in the Simscape Multibody Contact Forces Library [22]. For this simulation, the Hunt-Crossley model [23] is used for ground contact, with ground stiffness K_{ground} set to $1 * 10^6$ and ground damping B_{ground} at $1 * 10^3$. A Coulomb friction model is used to emulate friction, with static friction F_{ST} set to 1 and Kinetic Friction F_{KE} set to 0.7. These contact models were selected because they capture the basic ground reaction force characteristics while allowing for higher complexity when desired. The torque τ of the DC motor model used in the simulation is given by

$$\tau = T_{Stall} - \omega * T_{Stall}/W_{NL}, \tag{3}$$

where ω is the angular velocity, $T_{Stall} = 26.667Nm$ is the stall torque, and $W_{NL} = 997.5RPM$ is the no load speed. The power consumption P is given by

$$P = \omega * \tau + \left(\frac{\tau}{K_t}\right)^2 * R_m, \tag{4}$$

and includes both the mechanical power (1st term) and the power loss due to heat (2nd term). The winding resistance $R_m = 10.8\Omega$ and the torque constant $K_t = 1.95$. The gear ratio for the motors is $GR = 5.25$. Table I summarizes the parameters in the contact model and motor models used within Simscape for the quadruped.

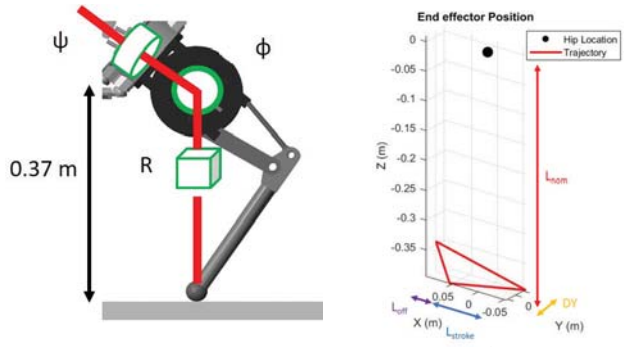
2) Controller: For controlling each leg, a virtual leg was defined as a revolute, revolute, prismatic (RRP) manipulator with generalized coordinates R, ϕ, and ψ as seen in Figure 3(a). Using this virtual manipulator a PD controller is defined on the three generalized coordinates, which allows us to effectively specify stiffness and damping terms in each dimension. The controller used on LLAMA prescribes a 3-point trajectory to each toe with respect to the hip.

TABLE I

SIMSCAPE MODEL PARAMETERS

Parameter	Symbol	Value	Units
Contact Parameters			
Stiffness	K_{ground}	1000000	N/m
Damping	B_{ground}	1000	N*s/m
Static Friction	F_{ST}	1	N/A
Kinetic Friction	F_{KE}	.7	N/A
Motor Model			
Stall Torque	T_{Stall}	26.667	Nm
No Load Speed	W_{NL}	997.5	RPM
Gear Ratio	GR	5.25	N/A
Torque Constant	K_t	1.95	N/A
Winding Resistance	R_m	10.8	Ω

Figure 3(b) shows an example trajectory that could be used by the robot. The shape of the trajectory is defined by the stroke length L_{stroke}, Nominal Leg Length L_{nom}, Flight Offset L_{off}, and Turn Gain DY. L_{stroke} defines the forward distance for each foot to travel, while L_{off} provides an adjusted apex location of the 3-point trajectory to assist with maintaining the forward velocity of the robot. DY is used to specify a lateral foot displacement for the robot to enable turning motions. To coordinate the legs the controller also specifies a trotting pattern, where diagonal pairs of legs are synced together.



(a) (b)

Fig. 3. (a) Legs are controlled using a virtual RRP manipulator defined by the kinematics of the leg. (b) The trajectory defined for each leg is defined with a 3-Point Trajectory.

Using this controller, a Nelder Mead optimization was performed on LLAMA with a cost function of:

$$Cost = \frac{1 + max(yaw)}{(1+V)^3}, \tag{5}$$

with V being the forward velocity of the robot. This function was chosen to attempt to maximize forward velocity while minimizing deviations in the yaw of the robot. Table II shows the 11 parameters used for the optimization and the optimized values. DY was not included in the optimization because the objective was for forward motion. After the optimizer ran for 81 iterations the robot was able to achieve a forward velocity of 1.55 m/s with an average power consumption of 1140 W.

TABLE II

CONTROL PARAMETERS

Optimization Parameters					
Parameter	Symbol	Upper Bound	Lower Bound	Value	Units
Stroke Length	L_{stroke}	.25	.05	0.156	m
Nominal Leg Length	L_{nom}	.45	.25	0.40	m
Flight Offset	L_{off}	.3	0	0.169	m
Frequency	f	10.0	1.0	4.74	Hz
Duty Cycle	DC	70	30	40	%
Radial Stiffness	K_R	20000	10000	16000	$\frac{N}{m}$
Phi Stiffness ϕ	K_ϕ	1000	100	695	$\frac{Nm}{rad}$
Psi Stiffness ψ	K_ψ	1000	100	500	$\frac{Nm}{rad}$
Radial Damping	B_R	1000	500	500	$\frac{N*s}{m}$
Phi Damping ϕ	B_ϕ	50	1	13.25	$\frac{Nm*s}{rad}$
Psi Damping ψ	B_ψ	50	1	49.8	$\frac{Nm*s}{rad}$

After running the optimizer, the full range of motion of the platform needed to be explored. By adjusting the frequency f and by adjusting the turn gain DY, a large range of various speeds and angular velocities can be reached. Figure 4 gives an example of the range of motion that can be achieved by the robot. These control parameters are used to define $u(k)$ within SBMPO.

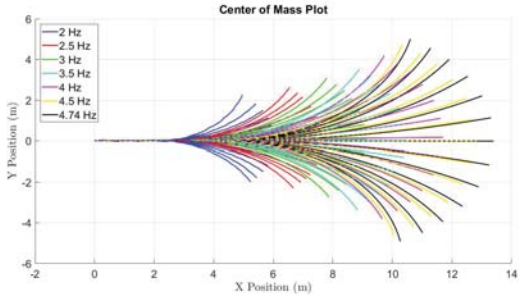
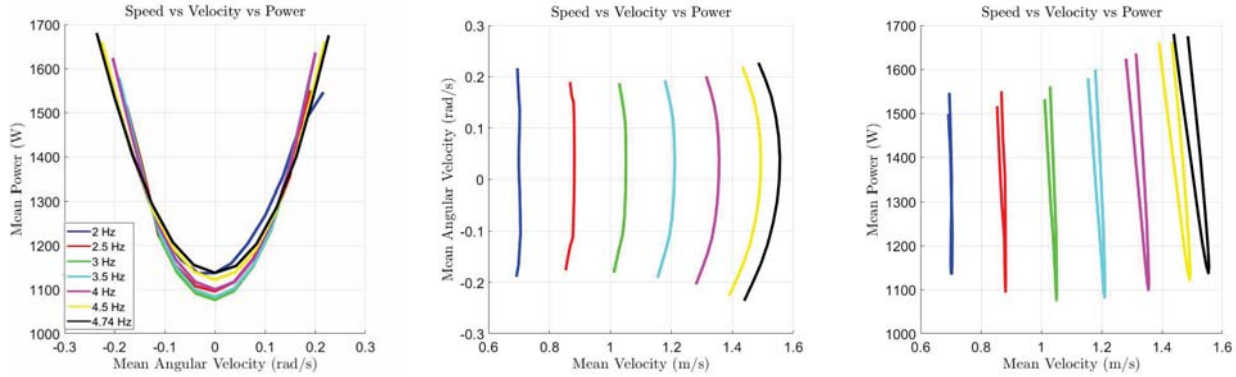


Fig. 4. COM trajectories of the LLAMA platform. This range of motions were achieved by varying the frequency f and turn gain DY. These runs lasted 10 seconds, with 1 second of standing, 3 seconds of trotting forward, and the remaining 6 seconds of turning.

C. SBMPO for Legged Robots

When SBMPO was first employed to motion plan for the legged robot X-RHex Lite [24], the simplified model used as the propagation model ignored some of the complex kinematic and dynamic properties of the robot. In particular, it could not capture ground interactions or the subtle shifts and roll of the vehicle center of mass [25] resulting in a degradation of the fidelity needed for motion planning.

While motion models used in the SBMPO paradigm are often analytically derived using physics, the complexity of legged robots encourages the use of neural networks. In particular, recurrent neural networks (RNNs), convolutional neural networks (CNNs) and hidden Markov models are considered as the state of the art [26] in prediction and forecasting of time series problems. Legged platforms such



(a) (b) (c)

Fig. 5. Graphs depicting power relationships at different velocities and commanded turns: (a) the power cost of fast turns, with power consumption P increasing as the angular velocity ω increases; (b) forward velocity v as a function of angular velocity ω, with the velocity dropping off at higher turning rates; (c) speed vs power plot, with velocity v dropping off as power P increases.

as LLAMA need time-series tools for proper motion prediction. Thus SBMPO has been altered to utilize a single-step unrolled RNN [27], which can be viewed as a deep feed-forward neural network, within the prediction model (as shown in Figure 6).

Another important consideration when planning for dynamic legged robots is the employed gait, a cyclic pattern of leg motions, used to maneuver and maintain stable locomotion. This type of locomotion requires a fundamental shift in planning methodology from time-based to stride-based. Stride-based node exploration is critically important for a robotic platform that moves in discrete steps [25]. During a single stride, internal body motions (such as roll, pitch, yaw) occur which can add significant volatility to robot state information. By limiting the data to stride, it has been found [25] that data relevant for motion planning is preserved and less experimental data is needed to determine an accurate motion model.

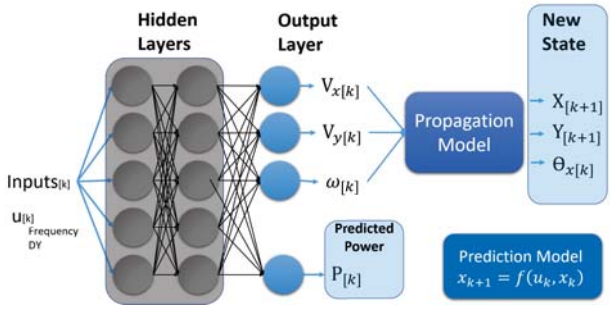


Fig. 6. Prediction model. The prediction model converts control inputs into outputs and is used within SBMPO to generate the predected next state. The central component of this is a neural network which learns the kinematics, this is coupled with a propagation model. To generate energy efficient trajectories, the network is also trained on power consumption data from simulation. The key outputs of the prediction model is robot pose (position, orientation) and power.

The dependence on stride changes the index k in Figure 1 from a general time index into a stride index. This change also modifies the cost function and optimistic heuristic function, the function that represents a lower bound on expected cost between a future node and the goal node. Distance optimal heuristics were adapted to be a simple Euclidean distance measured in attainable stride lengths between a considered node and the goal. The distance-based cost while traveling is calculated by propagating the distance moved by the time taken by each discrete vehicle stride.

In the case of energy optimal planning, the required time to travel the Euclidean distance D between the current node at the minimum-energy fixed stride frequency is computed. This also has a resulting forward velocity V_k, such that $t = \frac{D}{V_k}$. The optimistic heuristic is then given by the energy expended during traversal for this time.

The critical state information at stride k are the angular velocity ω_k, forward velocity V_k and power consumption P_k. The first two are kinematic terms learned from the neural network responsible for robot motion within the propagation model of Figure 2. The power model is computed in a neural network housed within the edge cost evaluation module (found in the same Figure 2) as power is used within the energy efficient cost function.

The propagation model (Fig. 6) takes the neural network outputs of forward velocity V and angular velocity ω and converts them into pose estimates V_x, V_y, and θ by stride through the equations:

$$\theta_{k+1} = \theta_k + \omega_k t_k, \tag{6}$$

$$V_{xk+1} = V_k cos(\theta_k + \omega_k t_k), \tag{7}$$

$$V_{yk+1} = V_k sin(\theta_k + \omega_k t_k), \tag{8}$$

These alterations in the algorithm allow direct planning of the robot trajectories using control parameters directly related to the dynamic motion of the robot. With the addition of neural network tools, the motion model of the robot has increased fidelity and the ability to adapt on-line through sensory feedback.

III. Developing the Learned Models

A. Neural Network Model

The neural networks used in this research are feed-forward networks that have two hidden layers. The network is structured to have 20 hidden neurons in the first layer and 20 in the second layer with identical weights, the output layer having 4 neurons. Inputs to the network are the control parameters f_k (stride frequency) and DY_k (turn gain) with the outputs being angular velocity ω_k, forward velocity vx_k, lateral velocity vy_k, and power consumption P_k.

To generate data for the neural network, a series of simulations were executed with stride frequency ranging from 2.5 to 4.74 in increments of 0.5, ie:($[2.5, 3.0, 3.5, 4.0, 4.5, 4.74]$) and for the turn gain, values were altered ranging from -0.12 to +0.12 in increments of 0.02. These values were set to operate within these limits as they exhibited stable behaviors within the limits of the robot's motion capability. A total of 85 simulations were conducted to create the data necessary for training, testing and validation. These simulations spanned the stable range of control inputs providing the necessary breadth of power and motion data for neural network. In this manner, the neural network was constructed entirely within the stability bounds of the vehicle, implicitly constraining all motion planning samples to be within safe and stable regions while in motion.

Figure 5 shows the angular velocity, forward velocity and power consumption of the robot, respectively, for distinct operating frequencies. Figure 5(a) illustrates the increased power consumption as angular velocity of the platform increases. Figure 5(b) highlights the inverse relationship between angular velocity and forward velocity. Figure 5(c) adds insight into the inverse relationship between speed and power during increased angular velocity, with asymetric behaviors appearing at higher frequencies as seen by the u-shaped pattern. At higher frequencies ($f = 4.74, f = 3.0$), the power to turn radius relationship is similar to those of a skid-steered robot [28].

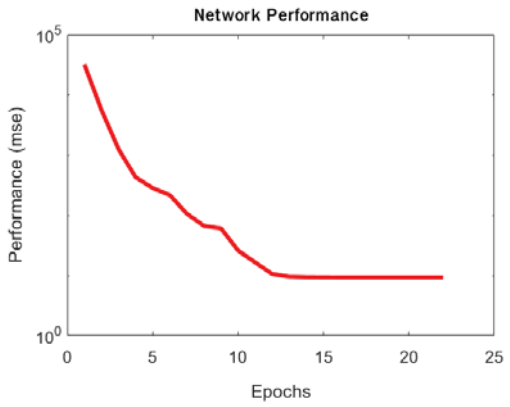


Fig. 7. Network Performance. The mean squared error (mse) of the network is shown. Errors converged to 9.32 at Epoch 16 and the learning was terminated at Epoch 23.

A total of 7931 data points were collected in simulation. Each data point corresponds to a stride taken with different f and DY. The network was trained using sigmoidal activation functions and optimized with the levenberg-marquardt algorithm. The trained network was able to fit the data with a R^2 value of 0.9942 in training and 0.981 in validation.

B. Motion Planning Simulations

Testing and validation of the motion planning was carried out in the MATLAB Simscape Multibody software. A scenario map of obstacles was loaded into SBMPO which returns a series of stride-by-stride control instructions f and DY. Execution of these instructions were completed on the simulated robot.

The Simscape environment accepts SBMPO generated trajectory control inputs to execute planned motion. The simulation environment constrains the vehicle to only accept new commanded controls once the stride time has completed. When testing SBMPO trajectories in this environment, it was discovered that transient effects from sudden or short turns failed to be properly captured by the neural network used by SBMPO as the propagation model, causing failure to track the original trajectory.

To correct for the resulting simulation pose error, a tracking controller was implemented. This controller relies on the same learned motion and power models developed for SBMPO in expanding and selecting optimal trajectory merging, subject to sampling. This controller is a Sampling-Based Model Predictive Control (SBMPC) and is effectively a receding-horizon SBMPO algorithm [29]. The controller attempts to track onto the originally computed SBMPO node, a single stride ahead. If there is no need to correct the trajectory, i.e., there is no pose error, no additional action is required and the original plan is executed.

IV. Results

Results were generated by the planner (SBMPO) and verified in Simscape. Some differences occured on execution of the trajectory in simulation due to unmodeled transient effects, but are corrected for by the model predictive controller described in section III-B.

Two cases are displayed in this section, the first is a variant of a cannonical planning problem, where a narrow gap must be navigated. The second is a slightly more realistic situation for outdoor settings with small obstacles scattered in a field. Both distance-optimized and energy-optimized plans are compared in each scenario and computation time, distance cost and energy cost are measured. Computation time is recorded after the algorithm has been initialized (ie: given goal, current pose) and is stopped once the optimal trajectory has been found.

A. Gap Configuration

Figure 8 shows the resulting trajectory from navigating through a wall gap problem. The energy optimal case elects to take a path that minimizes the amount of turns the platform needs to take, as seen in cyan. The distance optimal case planned for the platform to adjust its heading more, as seen

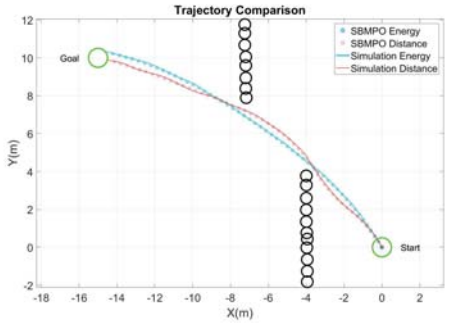


Fig. 8. Distance-based (red) and Energy-based (cyan) trajectories. This is a variant of a typical obstacle configuration that is used to test motion planning. Original trajectories planned by SBMPO are shown as small dots in their respective colors. The actual trajectory taken after tracking corrections are implemented is shown as a solid line.

TABLE III
RESULTS FROM SBMPO AND THE SIMULATION

Gap Scenario				
	Energy		Distance	
Cost Function	SBMPO	Simscape	SBMPO	Simscape
Distance (m)	17.86	18.38	17.72	18.19
Energy (kJ)	19.1	21.7	20.7	21.6
Computation Time (s)	0.0021		0.558	

with the solid red line. Both cases planned to a similar result with influences from their respective cost functions.

Table III details the distance traveled and energy consumed. Predicted numbers based off from SBMPO are compared to that of validation on Simscape. Since the behaviors of the energy and distance optimal cases were similar when tested in Simscape. Transient behaviors during strides caused deviations from the SMBPO trajectory and increases in the distance traveled and energy consumed in each gait. It is also interesting to note that the energy-efficient planner required a shorter time to compute.

B. Scattered Configuration

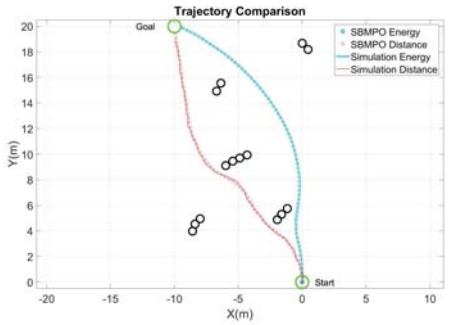


Fig. 9. Distance-based (red) and Energy-based (cyan) trajectories. This is a typical obstacle configuration seen in vegetated outdoor environments. Original trajectories planned by SBMPO are shown as small dots in their respective colors. The actual trajectory taken after tracking corrections are implemented is shown as a solid line.

TABLE IV
RESULTS FROM SBMPO AND THE SIMULATION

Scattered Scenario				
	Energy		Distance	
Cost Function	SBMPO	Simscape	SBMPO	Simscape
Distance (m)	23.68	24.01	22.89	23.01
Energy (kJ)	27.0	30.5	41.5	44.8

More pronounced differences between the energy-efficient and distance cases are found in the scattered obstacles scenario (Figure 9). The distance-optimized trajectory tends to skirt very closely to obstacles changing directions as needed, causing small perturbations to result while tracking. The smoother trajectories generated by the energy-optimized trajectory has fewer perturbation and is easier for the controller to correct.

Resulting values (Table IV) show when looking in terms of distance, the distance optimal case finds a shorter path to the goal by a meter within the Simscape model while navigating the complex terrain. However when comparing the energy consumption of the platform the distance optimal case take roughly 50% more energy. The energy optimal case has an overall better performance when using a legged platform in a more complex environment.

V. CONCLUSIONS AND FUTURE WORK

This paper shows that real-time motion planning for complex legged robots is possible through the use of the SBMPO planning paradigm. Neural networks were used as the motion and power models and were learned to sufficient accuracy to enable effective motion planning. Further, this proposed method is capable of optimizing a trajectory that is energy efficient, directly increasing operation life of the legged robot and results in less control effort expended for correction. This planning paradigm is also computationally fast, allowing use of it in on-line motion planning.

Further work will transition the developed technologies onto the physical version of the quadrupedal platform. New gaits will be developed and implemented to the SBMPO framework allowing a greater range of motions, such as strafing and jumping.

Utilizing the energetic costs of motion a robot can learn to distinguish between different terrains. This will allow further development in allowing legged robots to motion plan in multi-terrain environments. Neural networks of increased complexity, such as a LSTM network will also be used to increase accuracy of SBMPO's propagation model and incorporate the transient effects of sudden motions.

ACKNOWLEDGMENT

This work was supported by the collaborative participation in the Robotics Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD 19-01-2-0012. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes not withstanding any copyright notation thereon.

REFERENCES

[1] U. Saranli, M. Buehler, and D. E. Koditschek, "Rhex: A simple and highly mobile hexapod robot," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.

[2] H.-W. Park, P. M. Wensing, and S. Kim, "High-speed bounding with the mit cheetah 2: Control design and experiments," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 167–192, 2017.

[3] D. J. Blackman, J. V. Nicholson, J. L. Pusey, M. P. Austin, C. Young, J. M. Brown, and J. E. Clark, "Leg design for running and jumping dynamics," in *Robotics and Biomimetics (ROBIO), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2617–2623.

[4] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.

[5] M. Hutter, C. Gehring, A. Lauber, F. Gunther, C. D. Bellicoso, V. Tsounis, P. Fankhauser, R. Diethelm, S. Bachmann, M. Blösch *et al.*, "Anymal-toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, 2017.

[6] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 1184–1189.

[7] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1325–1349, 2008. [Online]. Available: https://doi.org/10.1177/0278364908098447

[8] H. Wang, Y. F. Zheng, Y. Jun, and P. Oh, "Drc-hubo walking on rough terrains," in *Technologies for Practical Robot Applications (TePRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–6.

[9] C. G. Atkeson, B. P. W. Babu, N. Banerjee, D. Berenson, C. P. Bove, X. Cui, M. DeDonato, R. Du, S. Feng, P. Franklin *et al.*, "No falls, no resets: Reliable humanoid behavior in the darpa robotics challenge," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, pp. 623–630.

[10] N. A. Radford, P. Strawser, K. Hambuchen, J. S. Mehling, W. K. Verdeyen, A. S. Donnan, J. Holley, J. Sanchez, V. Nguyen, L. Bridgwater *et al.*, "Valkyrie: Nasa's first bipedal humanoid robot," *Journal of Field Robotics*, vol. 32, no. 3, pp. 397–419, 2015.

[11] G. C. Haynes, J. Pusey, R. Knopf, A. M. Johnson, and D. E. Koditschek, "Laboratory on legs: an architecture for adjustable morphology with legged robots," in *Unmanned Systems Technology XIV*, R. E. Karlsen, D. W. Gage, C. M. Shoemaker, and G. R. Gerhart, Eds., vol. 8387. SPIE, 2012, p. 83870W. [Online]. Available: http://link.aip.org/link/?PSI/8387/83870W/1

[12] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic a," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 224–243, 2016.

[13] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A*," *Artificial Intelligence*, vol. 155, no. 1, pp. 93–146, 2004.

[14] O. Chuy, E. Collins, D. Dunlap, and A. Sharma, "Sampling-based direct trajectory generation using the minimum time cost function," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, J. P. Desai, G. Dudek, O. Khatib, and V. Kumar, Eds. Springer International Publishing, 2013, vol. 88, pp. 651–666.

[15] D. Dunlap, C. Caldwell, and E. Collins, "Nonlinear model predictive control using sampling and goal-directed optimization," in *Proceedings of the Multi-Conference on Systems and Control*, Yokohama, Japan, September 8-9 2010.

[16] J. Maciejowsky, *Predictive Control with Constraints*. Prentice Hall, 2002.

[17] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[18] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[19] C. V. Caldwell, D. D. Dunlap, and E. G. C. Jr., "Application of Sampling Based Model Predictive Control to an autonomous underwater vehicle," *Ship Science and Technology*, vol. 4, no. 7, pp. 55–63, July 2010.

[20] N. Gupta, C. Ordonez, and J. E. G. Collins, "Dynamically feasible, energy efficient motion planning for skid-steered vehicles," *Autonomous Robots*, 2016.

[21] G. Francis, E. Collins, O. Chuy, and A. Sharma, "Sampling-based trajectory generation for autonomous spacecraft rendezvous and docking," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013, p. 4549.

[22] S. Miller. (2017) Simscape multibody contact forces library. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/47417-simscape-multibody-contact-forces-library

[23] L. Ding, H. Gao, Z. Deng, J. Song, Y. Liu, G. Liu, and K. Iagnemma, "FootâĂŞterrain interaction mechanics for legged robots: Modeling and experimental validation," *The International Journal of Robotics Research*, vol. 32, no. 13, pp. 1585–1606, 2013. [Online]. Available: https://doi.org/10.1177/0278364913498122

[24] C. Ordonez, N. Gupta, E. G. Collins, J. Clark, and A. M. Johnson, "Power modeling of the XRL hexapedal robot and its application to energy efficient motion planning," in *Proceedings of the International Conference on Climbing and Walking Robots*, Baltimore, Maryland, USA, July 2012, pp. 689–696.

[25] M. Harper, J. Pace, N. Gupta, C. Ordonez, and E. G. Collins, "Kinematic modeling of a rhex-type robot using a neural network," in *Unmanned Systems Technology XIX*, vol. 10195. International Society for Optics and Photonics, 2017, p. 1019507.

[26] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi, "Classification and segmentation of satellite orthoimagery using convolutional neural networks," *Remote Sensing*, vol. 8, no. 4, p. 329, 2016.

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[28] N. Gupta, C. Ordonez, and E. G. Collins, "Dynamically feasible, energy efficient motion planning for skid-steered vehicles," *Autonomous Robots*, pp. 1–19, 2016. [Online]. Available: http://dx.doi.org/10.1007/s10514-016-9550-8

[29] B. M. Reese and E. G. Collins Jr, "A graph search and neural network approach to adaptive nonlinear model predictive control," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 250–268, 2016.