A Review of Locomotion Techniques In Quadruped Robot

Anshul Pagariya
Seminar: Learning-Based Control
RWTH Aachen University
anshul.pagariya@rwth-aachen.de

***Abstract*—Legged robots have compelling applications, but legged robot's dynamic and agile maneuvers present a significant challenge. Understanding how such a complex motion can be achieved requires good knowledge of fundamental theoretical knowledge. In this paper, an attempt is made to discuss various aspects and techniques for quadruped robot locomotion by describing and analyzing some critical literature in the field. The paper introduces the necessary theoretical background and general problems related to quadruped locomotion to make it easy for the reader to understand the discussed literature.**

As part of the seminar Learning-Based Control offered by DSME, RWTH Aachen University, this paper aims to provide a comprehensive view on technicalities involved in legged locomotion and introduce state-of-the-art methods of legged locomotion.

I. Introduction

Legged robots can choose footholds within their kinematic limits flexibly. This advantage makes the adoption of legged robots, particularly quadruped robots, very promising. The other alternatives with wheels as the only source of movement do not provide such flexibility to cross difficult terrain. Quadruped robots are already being used for testing radiations in nuclear sites, which if done by humans could be dangerous. In the future, many compelling applications will arise where quadruped robots will be beneficial. A variety of quadruped systems have shown dynamic and agile maneuvers; the well-known are Boston Dynamics, MIT Cheetah, ANYmal by ETH. In this paper, the main focus is on the ANYmal robot. ANYmal (figure 1) has 12 actuators, sensors (both proprioceptive and exteroceptive). It has Mammalian joint arrangement: hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE). ANYmal robot has a Serial Elastic Actuator (SEA) which is very difficult to model accurately.

The review paper is divided into five parts from here on, namely 1.Theoretical background: in which a base knowledge surrounding modeling, equations of motion, and constraints are discussed 2.General problems: in which problems related to legged locomotion are discussed 3.Techniques for developing control policies: in which the essential steps of recent techniques for legged locomotion are explained 4.Experiment performed, and discussion: in which the experiments performed in each technique explained in the previous section are discussed 5.Conclusion: in which conclusion is drawn for this seminar paper.

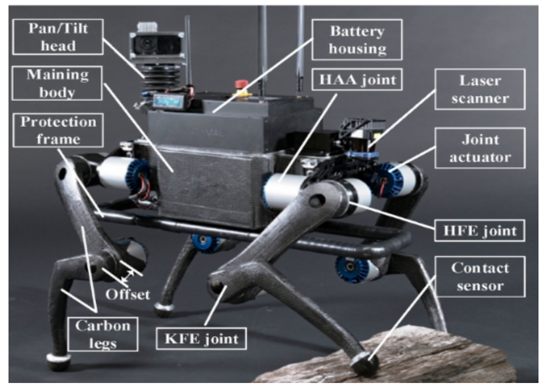

Fig. 1. Different parts of ANYmal robot. Figure from Robotic Systems Lab, ETH Zurich.

II. Theoretical Background

To achieve a feasible motion for the robot a need arises to establish a relationship between forces and the motion generated due to them, which is captured by equation of motion. To generate an equation of motion, a model of the robot is required. Also, the legged motion needs to satisfy many constraints and conditions to achieve a feasible motion. In this section, a discussion is done on ways to model robots, equations of motion, and constraints, conditions for legged locomotion.

A. *Modeling of Robot*

1) ***Linear Inverted Pendulum(LIP)****:* The simplest descriptive model of quadruped robot is Linear Inverted Pendulum (LIP) model (figure 2). In this model, optimization is done on the Center of Mass (CoM) position of the robot using predefined footholds and step timings. The hinged position of the pendulum (given by u in figure 2 known as Zero Moment Point (ZMP) or Center of Pressure (CoP)) must lie on connecting line (i.e., convex hull, the green line in figure 2) between legs in contact. This simple model has also proved to achieve legged locomotion. However, this model can not give complex motions because the optimization of CoM is done only in 2 dimensions. [7] [6].

2) ***Centroidal Dynamics****:* Another standard dynamic model used is the Centroidal dynamics (figure 3) in which the dynamics are approximated by a single rigid-body with mass "m" and inertia "I" located at the CoM of the robot [7].

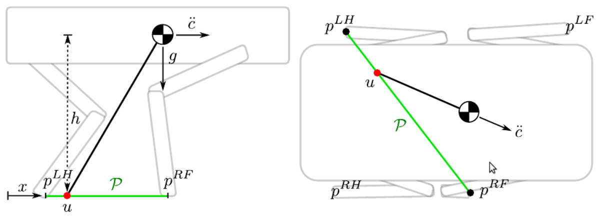

Fig. 2. LIP model of quadruped robot with the right-front $\mathbf{p}^{RF}$ and left-hind $\mathbf{p}^{LH}$ being the legs in contact. Figure from [6]

*3) **Machine Learning Based**:* Machine learning methods, in particular supervised learning, can be used to learn complex dynamics. Equations (1) showing learning of motor dynamics and (2) showing learning transition dynamics are examples of machine learning-based modeling.

$$\tau = f(q_t, q_t') \tag{1}$$

$$x_{t+1} = f\left(x_t, u_t\right) \text{ Where } x_t = [q_t, q_t'] \tag{2}$$

Where x_{t+1} is state of robot at time t+1, u_t is control input, q_t is joint position at time t, q_t' is joint velocity at time t and τ is torque output of motor.

B. *Equation Of Motion*

In literature, many formulations have been discussed, which yields an equivalent description of motion. Here, Newton-Euler and Lagrange equations are briefly presented. The motion equation considers the robot's mass, inertia and establishes a relationship between the robot's state, joint torque, and different forces.

*1) **Newton-Euler Equation**:* The Newton-Euler Equations describes the combined translational and rotational dynamics of the rigid body. The equation is formulated as:

$$M(q)\left(\begin{bmatrix} \ddot{q} \\ \ddot{x}_0 \\ \ddot{\theta}_0 \end{bmatrix} + \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix}\right) + n(q,\dot{q}) = \begin{bmatrix} u \\ 0 \\ 0 \end{bmatrix} + \textstyle\sum_i C_i(q)^\top f_i \tag{3}$$

where $\boldsymbol{M}(\boldsymbol{q})$ is the generalized inertia matrix of the robot, $-\boldsymbol{g}$ is the constant gravity acceleration vector, $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the vector of coriolis and centrifugal effects, $\boldsymbol{u}$ is the vector of joint torques, and for all i, f_i is a force exerted by the environment on the robot and $\boldsymbol{C}_i(\boldsymbol{q})$ is the associated Jacobian matrix [4].

*2) **Lagrange Equation**:* The rigid body system's energy can be described by the lagrangian function L (4) where T is kinetic energy and is V potential energy.

$$\mathcal{L} = T - V \tag{4}$$

Lagrange equations is given by:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0 \tag{5}$$

Where q_i is generalized coordinates and $\dot{q}_i$ is time derivatives [4][5].

Finally, by these equations, an equivalent equation of motion as 3 is generated. In this paper, further explanation on how to get the Lagrangian equation of motion is not done since it will be out of scope.

C. *Constraint and Conditions for Legged Locomotion*

The legged locomotion is constrained with many physical aspects for movement.

*1) **Unilateral Constraint**:* The contact forces are unilateral; they can only push, not pull [5].

*2) **Friction Constraint**:* To have motion without slipping, the robot's contact force must be in friction cone (shown in figure 3 in red color); otherwise, slipping starts.

*3) **Stability Constraint**:* For the robot's dynamic stability, the Zero Moment Point (the point where moment due to forces is zero) must lie within the support polygon. Support polygon (shown in figure 4) is the convex hull generated by footpoint in contact with environment [5][1].

III. General Problems

In this section, discussion on the general problem faced while designing a control method for legged locomotion are highlighted.

*1) **Input Requirements**:* To achieve legged locomotion, generally, a pre-specified input is required. In some methods, the contact schedule,i.e., the predefined lift-off and touch-down timings for each leg and gait pattern, are predefined. Automatic determination of the input parameters, i.e., step timings, foot position, the order of foot sequence, has been done in the literature, and in section IV-A1 the technique is briefly explained.

*2) **Inaccurate Modeling**:* The limitation of inaccurate modeling of the actuator, complex structure dynamics, physical parameters of the robot (e.g., inertia, mass), and delay in information transfer possesses a great challenge to robust robot locomotion. Recent literature works used Machine learning methods like Supervised learning to learn complex model dynamics like actuator. For robustness, the physical parameters of the robot are randomized (by adding noise). In section IV-B techniques to solve these issues are discussed in more detail.

*3) **Unreliable Control Policy**:* Modeling errors, reliance on the input of robots (e.g., foothold positions, environment map, gait type), unknown terrain pose a significant challenge to create a robust control policy. Data-driven methods, like reinforcement learning (RL), promise to overcome the limitations of prior model-based approaches by learning useful controllers directly from experience. RL's idea is to collect data by trial and error and automatically tune the controller to optimize the given cost (or reward) function representing the task. In section IV-B we will briefly discuss how RL techniques have been implemented to achieve robust policies by training policy in teacher-student and curriculum learning fashion, for the unknown environment and minimum input requirement (i.e., only desired speed and direction).

IV. Techniques for Developing Control Policies

Different researchers have done much work to develop a robust control policy for legged locomotion in recent years. The techniques can be divided into two broad types 1. Optimization formulation with analytical modeling of robots 2. Data-driven (in particular Reinforcement learning) combined with machine learning and analytical modeling based. In this section, a few important pieces of literature for each of these types are discussed.

A. *Optimization Formulation With Analytical Modeling Of Robots*

In this type, generally, various constraints and conditions discussed in II-C along with predefined conditions were formulated as optimization problem i.e Quadratic programming (6), Nonlinear programming (7), to get a feasible trajectory for the robot. The objective in Quadratic programming is to find x which:

$$\text{minimize } \frac{1}{2}\mathbf{x}^{\mathrm{T}}Q\mathrm{x} + \mathrm{c}^{\mathrm{T}}\mathrm{x}, \text{ subject to } A\mathrm{x} \preceq \mathrm{b} \tag{6}$$

Where Q,A are matrix and c and b are vector of appropriate dimensions. The objective of Nonlinear programming is to:

$$\text{minimize } f(x), \text{ subject to } g_i(x) \leq 0 \text{ and } h_j(x) = 0 \tag{7}$$

The robots are modeled analytically, i.e., by Computer Aided Drawing (CAD) in this type. In the following subsection, papers on this type are discussed.

*1) **Paper - Gait and trajectory optimization for legged system through phase-based end-effector parameterization [7] :*** In this paper, the author implemented online optimization of gait sequences and robot motion, contrary to the many other methods where there is a need to specify gait pattern as input in this method inputs were the desired goal position and number of steps only. The motion of feet and contact forces are optimized based on phases (splines were used for optimization). The dynamic model used is simplified centroidal dynamics (figure 3). The Newton-Euler equation gives the rigid body dynamics (i.e., equation of motion).

The trajectory optimization formulation (nonlinear programming formulation (7)) finds CoM linear position, base Euler angles, and discovers appropriate gait pattern, feet position, and contact force by taking into account dynamic model, kinematic model (which enforces constraint on leg movement, shown in figure 3 in blue color) and other physical constraints like friction cone, leg extension limit, terrain height. Once the physically feasible motion is found for an unactuated base, the leg torque is computed using inverse dynamics (algorithm to map the body's kinematics to force and torque). In this method, fourth-order polynomials were used for base optimization, and third-order polynomials were used for foot motion and foot force profile [7].

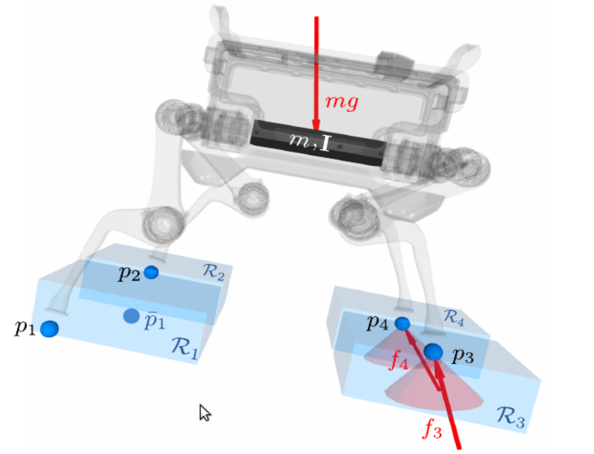

Fig. 3. The robot model used in the original paper. The model is approximated by Centroidal dynamics. The figure shows friction cone in red and kinematic model in blue. Figure taken from [7]

*2) **Paper - Dynamic Locomotion Through Online Nonlinear Motion Optimization for Quadrupedal Robots [1] :*** This paper demonstrates highly dynamic robot locomotion (dynamics gaits) and the dynamic transition between gaits. The method requires high-level velocity command, gait pattern, and contact schedule as input. The model of the robot is defined as a base attached with massless limbs. There were two optimizations in this method 1. The foothold optimization 2. Motion optimization. The foothold optimization (quadratic programming formulation (7)) is done by including constraints on leg motion, i.e., kinematic limit, along with computed reference footholds, which are computed using a LIP model (section II-A), which predicts the next foothold as a function of the measured and desired velocity of the torso. Motion optimization relies on a Zero Moment Point (ZMP) (section II-C3) based motion plan with nonlinear constraints to get a motion plan in all dimensions (x,y,z) for Center of mass (CoM). The support polygon (section II-C3) is computed using optimized footholds from the foothold optimization step. The overview of the technique is shown in figure 4. The acceleration and contact force from the motion optimization step goes into the whole body controller (WBC), which computes optimized acceleration and contact forces by solving hierarchical optimization (includes dynamics, friction cone, and other constraints/limits) [1].

B. *Data Driven (Reinforcement Learning) Combined With Machine Learning and Analytical Modeling Based*

In this type, in general, the control policy is learned using Reinforcement Learning (RL), RL in brief; aims to find an optimized policy $\pi(a, s)$ where "a" are actions and "s" are states which maximizes our reward function or sometimes also referred as cost function, by interacting with the environment and collecting rewards. The cost function or reward function embodies the desired behavior we want to achieve. The modeling of the robot is done analytically and also by using machine learning techniques like supervised learning. Equations (1),

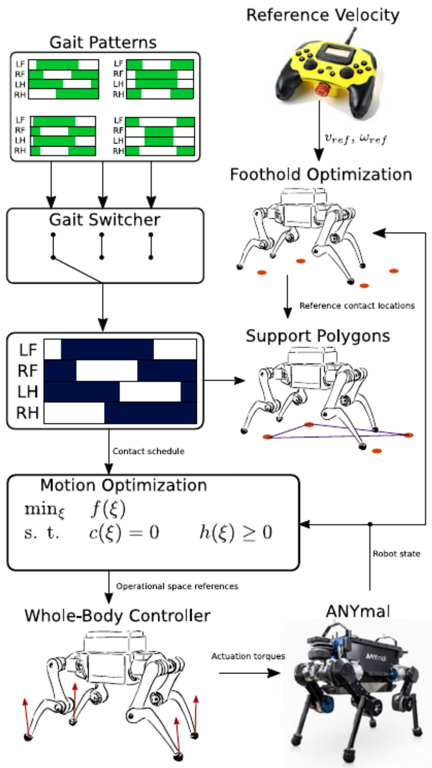

Fig. 4. An overview of the planning and control architecture described in paper. Figure taken from [1]

(2) show how supervised learning can be used to learn motor dynamics and transition dynamics. In the following subsection, papers on this type are discussed.

*1) **Paper - Learning Agile and Dynamic Motor Skills for Legged Robots [2] :*** In this paper, the author showed that using data-driven methods (Reinforcement learning) and supervised learning for learning complex actuator dynamics, a robust control policy could be generated only by training the policy in simulation and, finally, transferring the learned policy to the real system. The existing problem in other approaches was a very inaccurate dynamic model, complex optimization formulation. In this method simulator with the modeled rigid body of the robot was integrated with RL policy, namely TRPO and "actuator net" (actuator net is the learned neural network for motor dynamics similar to (1)). The RL policy generates desired action (joint position) by looking at the robot's state, history of the joint states, and previous actions. The desired joint angle (generated joint angles from RL policy) are subtracted from the current joint angle and, along with velocity history, are fed through a learned Neural network model of the actuator (actuator net), which gives the desired torque for each joint, which then goes into the rigid body simulation. The actuators net were trained outside the simulator by offline data collection. RL policy's cost function (reward) was defined to generate desired motion behavior (like fast locomotion, recovery from fall). Finally, the policy was transferred to the real robot. The policy showed efficient use of hardware capacity. An overview of the training loop and transfer to the real robot can be seen in figure 5 [2].

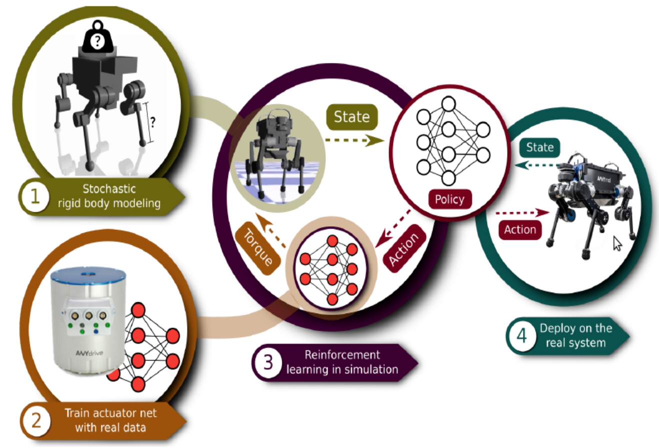

Fig. 5. Control policy architecture in simulation and transfer to real robot. Figure taken from [2]

*2) **Paper - Learning quadrupedal locomotion over challenging terrain [3]**:* In this paper, the author presented a robust control policy trained in simulation and was shown robust even in environments unknown during training and environments with gushing water or thick vegetation. The learned controller uses only proprioceptive measurement (e.g., joint angle values) from joint encoders and IMU as input. In this paper, a teacher policy (which has privileged information) and a separate student policy which essentially imitates the learning derived from teacher policy were used. The teacher and student policy were trained based on curriculum learning fashion i.e., the terrain complexity was increased as the training progressed. The teacher policy (trained in simulation) has privileged information like terrain profile, contact states, and contact forces; due to this information, the teacher policy achieves good performance very early. The teacher policy is based on Multi-layer perceptron (MLP). The teacher policy is distilled in student policy. The loss function for student policy, trained by supervised learning, is shown in equation (8)).

$$\mathcal{L} := \left(\bar{a}_t\left(o_t, x_t\right) - a_t\left(o_t, H\right)\right)^2 + \left(\bar{l}_t\left(o_t, x_t\right) - l_t(H)\right)^2 \quad (8)$$

Where a_t,o_t,x_t,l_t are action, observation, vector of privileged information, latent embedded vector at time t (embedded privileged information) and H is history of proprioceptive measurement. The $(\bar{.})$ represents the target values generated by teacher policy. Student policy only takes sequences of N proprioceptive observation as input. The policy uses a sequence model for training, namely a Temporal convolutional network (TCN), to have temporal knowledge. The action generated by both policies is 16- dimensional vectors consisting of leg frequencies and foot position residuals. Only student policy is deployed in real robots. Finally, the control architecture (figure 6 C) is divided into motion generation and tracking and outputs joint position targets [3].

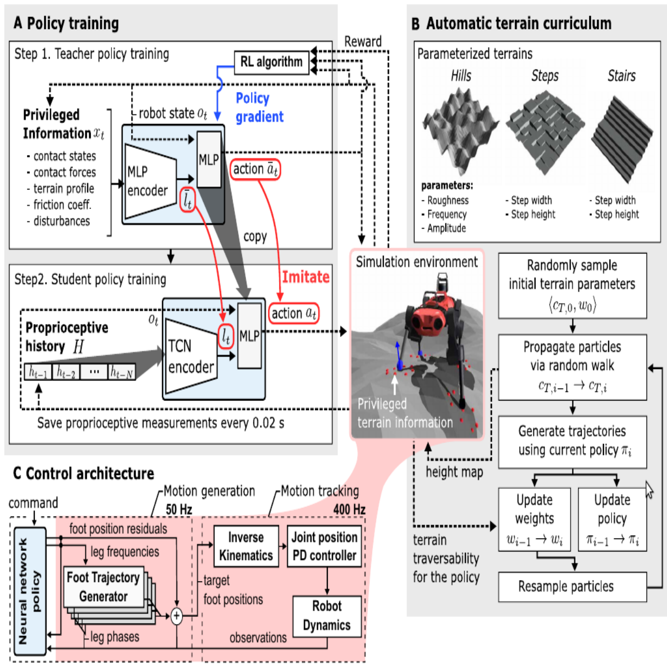

Fig. 6. Locomotion control architecture in the original paper. In A, policy training which is divided in teacher and student learning is shown. In B, the automated terrain curriculum is shown. Figure taken from [3]

V. Experiment Performed and Discussion

In this section, the experiments performed and discussion on the four papers explained in section IV are presented in the same sequence.

A. *Paper - Gait and trajectory optimization for legged system through phase-based end-effector parameterization [7]*

The techniques demonstrated 6-dimensional motion for the quadruped robot, and directly planned contact forces were executed on the real system. Detailed results for the real quadruped robot were not discussed in the paper. However, results for the biped robot were given in detail. Some critical points from the result are discussed hereon. Figure 7, shows that foot height is mostly zero, meaning the same as terrain showing foot contact constraint. Sometimes the curve goes below 0; it is when the bipedal robot enters a gap region. The technique demonstrated an automatic change in gait sequence and timings depending on terrain. The method has shown that in 4.1 s, motion for biped robots can be generated, including finding a 6-D motion plan, foothold position and forces, and enforcing all constraints. However, When the robot had heavy limbs, the centroidal dynamics assumption could not give accurate results, and a more accurate dynamic model is needed. Moreover, since the terrain constraint (terrain height, friction constraint) is enforced at some time intervals, the undesirable obstacle between this time interval could cause unexpected behavior.

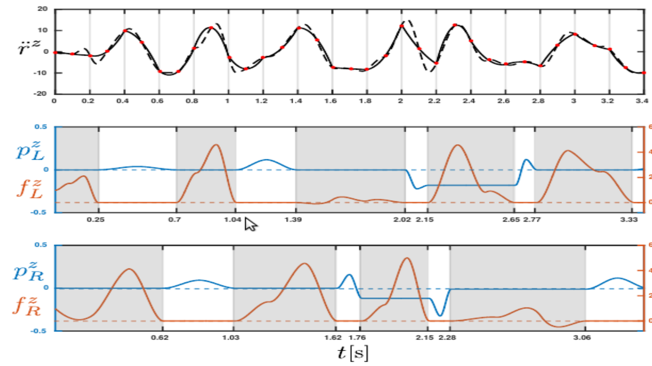

Fig. 7. Experiments for 20 kg-bipedal robots crossing a 1 m wide gap. In the top figure, planned base vertical acceleration is compared with acceleration computed by the Newton-Euler equation of motion evaluated with values from contact force and current footholds. The red dot spaced 0.1 s apart shows that the dynamic constraints are applied at these time points. The second and third plots show the vertical position and vertical forces for left(L) and right(R) legs. The unshaded region shows the swing phase. [7].

B. *Paper - Dynamic Locomotion Through Online Nonlinear Motion Optimization for Quadrupedal Robots [1]*

The technique demonstrated optimization for the position, velocity, and acceleration of COM of the robot in all directions resulting in motion plans for jumping, a broader range of gaits, dynamic lateral walk, pronking. The method is capable of online optimization. The computational time compared with a previous method that used linear ZMP based motion plans (that is, motion plans in only x, y-direction) was high. The method also included a gait switching module that enables automatic gait switching; this module is not explained in the seminar paper but can be read in the original paper [1]. The method is not robust in a harsh environment because of no knowledge of terrain type. In future work, terrain knowledge can be also be included manually through stereo cameras.

C. *Paper - Learning Agile and Dynamic Motor Skills for Legged Robots [2]*

In the paper, the experiments were done with objectives 1. Command-conditioned locomotion 2.High-speed locomotion 3. Recovery from a fall. In Command-conditioned locomotion, the robot followed desired direction and speed. When the same method was applied to the robot model with the ideal actuator model, the robot failed to walk even a step. This policy used lesser torque for the same speed compared to the robot's previous best controller. In the high-speed locomotion test, the robot used full hardware capacity i.e., using maximum torque and joint velocity. In recovery from a fall experiment, the robot was tested with many initially fallen poses and in all cases, recovered in a standing position. The method is not robust for the outdoor environment because of a lack of knowledge of ground truth. This problem is overcome by the method demonstrated in paper [3]. Another major limitation of this work was that the policy could only be trained on single-faceted tasks to overcome this author has hinted to use hierarchical structure in the policy network. The method uses joint state history as observation in RL problem which as the paper hypothesis helps in contact detection.

D. *Paper - Learning quadrupedal locomotion over challenging terrain [3]*

The controller retains its robustness under conditions that were never encountered during training: deformable terrains such as mud and snow, dynamic footholds such as rubble, and overground impediments such as thick vegetation and gushing water. In figure 8, the left plot shows the error between baseline velocity (4 m/s) and one achieved by the robot with and without a payload on the robot. The right plot shows an error in the heading angle. The shaded area in both curves is 95% confidence interval (CI). The method overcame many problems encountered in [2] as it was robust in outdoor environments due to its ground truth knowledge, the policy trained was multifaceted. The method used proprioceptive history, which enables the detection of contact and slippage events. Even though the policy was robust in vegetation and gushing water, such an environment's modeling was not done in simulation because of simulation's limitations. The method did not exhibit all gait patterns demonstrated by quadruped in nature. The method relies on proprioceptive sensors, which force the robot to take conservative action, which reduces speed and does not give a global view, for e.g. when there is a cliff in front of the robot, it will fall in the cliff since it is not aware of it. The author suggests using a combination of proprioceptive and exteroceptive sensors (hybrid sensors) to overcome these limitations.

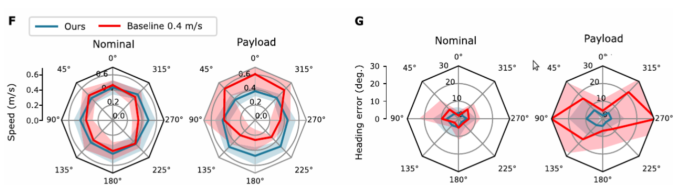

Fig. 8. Plot showing velocity and heading angle error. Figure taken from [3].

VI. Conclusion

At this moment, a robust policy can be achieved after training the policy in simulation for real-world deployment. The techniques presented in section IV highlighted two broad types 1. Analytical modeling of robots combined with optimization techniques and 2. Data-driven (Reinforcement learning) combined with machine learning and analytical modeling based. The latter type has recently been practical and has many advantages compared to the former like ease of training, robust policy, less hyperparameter tuning (particularly in case of recovery from fall wherein other methods trajectory have to be predefined to recover from fall), accurate modeling, maximum hardware capacity utilization. In the former type, the trajectory optimization formulation requires a solid understanding of mathematics, control policies were also not very robust, the modeling of parts was inaccurate. However, the latter type still requires a lot of domain expertise and in no way a trivial method, choosing the correct reward function, considering various scenarios in training and their simulation models, and many other technical aspects remain challenging. Further, since model-free reinforcement learning was used in the later type, which is sample inefficient, in the future use of more data-efficient techniques to learn control policies can be investigated.

References

[1] C Dario Bellicoso et al. "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2261–2268.

[2] Jemin Hwangbo et al. "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26 (2019).

[3] Joonho Lee et al. "Learning quadrupedal locomotion over challenging terrain". In: *Science robotics* 5.47 (2020).

[4] A René Geist and Sebastian Trimpe. "Structured learning of rigid-body dynamics: A survey and unified view". In: *arXiv e-prints* (2020), arXiv–2012.

[5] Pierre-Brice Wieber, Russ Tedrake, and Scott Kuindersma. "Modeling and control of legged robots". In: *Springer handbook of robotics*. Springer, 2016, pp. 1203–1234.

[6] A. W. Winkler et al. "Fast Trajectory Optimization for Legged Robots Using Vertex-Based ZMP Constraints". In: *IEEE Robotics and Automation Letters* 2.4 (2017), pp. 2201–2208. DOI: 10.1109/LRA.2017.2723931.

[7] Alexander W Winkler et al. "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1560–1567.