Design, Development and Experimental Realization of a Quadrupedal Research Platform: Stoch

Dhaivat Dholakiya, Shounak Bhattacharya, Ajay Gunalan, Abhik Singla,
Shalabh Bhatnagar, Bharadwaj Amrutur, Ashitava Ghosal, Shishir Kolathaya

Abstract— In this paper, we present a complete description of the hardware design and control architecture of our custom built quadruped robot, called the *Stoch*. Our goal is to realize a robust, modular, and a reliable quadrupedal platform, using which various locomotion behaviors are explored. This platform enables us to explore different research problems in legged locomotion, which use both traditional and learning based techniques. We discuss the merits and limitations of the platform in terms of exploitation of available behaviours, fast rapid prototyping, reproduction and repair. Towards the end, we will demonstrate trotting, bounding behaviors, and preliminary results in turning. In addition, we will also show various gait transitions i.e., trot-to-turn and trot-to-bound behaviors.

Keywords: Quadrupedal Robot, Legged Locomotion, Robot Design

I. INTRODUCTION

There have been multiple types of quadruped robots in varying sizes over the past ten years, namely, MIT Cheetah [1], Starl ETH [2], Spot mini [3], ANYmal [4], LaikaGo [5], Minitaur [12] etc. Most of these works have mainly focused on answering crucial questions on control and hardware design for walking, and have also demonstrated multiple types of functionalities both in simulation and experiments. Despite their impressive results, it is important to note that these robots cost to the tune of minimum $30,000, are expensive to manufacture, and also require advanced sensing and actuation.

There are also quadrupeds that are less resource intensive such as Tekken [6], Cheetah-cub [7], Onchila [8], Mini-cheetah [9], Aibo [10], Rhex [11], Open cat [13] some of which are open sourced, but not easy to reproduce. In the current state of research in robot locomotion, a beginner usually faces a stiff learning curve to design and fabricate a robust and a reliable quadrupedal walking platform. Crossing this barrier is not only limited by the individual's capacity but also by the requirement of resources. Thus, a robot which is simpler, modular, affordable, easily repairable, and yet facilitating complex behaviors is largely preferred.

Dhaivat Dholakiya, Shounak Bhattacharya, Ajay Gunalan, Abhik Singla are with the Robert Bosch Centre for Cyber-Physical Systems, IISc, Bangalore, India. E-mail: {dhaivatd, shounakb, ajayg, abhiksingla}@iisc.ac.in.

Ashitava Ghosal is with the Faculty of Mechanical Engineering, Bharadwaj Amrutur is with the Faculty of Electrical & Computer Engineering, Shalabh Bhatnagar is with Computer Science & Automation, and Shishir Kolathaya is an INSPIRE Faculty Fellow at the Robert Bosch Center for Cyber Physical Systems, IISc, Bengaluru, India {shalabh,amrutur,asitava,shishirk}@iisc.ac.in

Fig. 1: Figure showing our custom built quadrupedal robot Stoch, and its rendered CAD model in SOLIDWORKS environment.

With a view toward realizing state of the art walking controllers in a modular, and a low-cost walking platform, we developed a quadrupedal robot called the *Stoch*. This manuscript primarily serves as a detailed description of the hardware and software design process and construction of this platform (see Figure 1), which complements the results that were shown in [14]. This hardware features modular central body and a five bar co-axial leg design. Which are fabricated using easily available off the shelf materials like carbon-fiber tubes and custom 3D PLA parts. Carbon fiber tubes, which are strong and light, serve as the structural members and leg links. Similarly, the 3D printed parts, manufactured in-house, act as the joints. The software consists of a real-time trajectory generator, which are then passed as commands to the individual servo drives. The trajectory generator performs reasonably well at control frequencies on many off-the-shelf servos (50 Hz). In addition to realizing robust gaits in the real hardware, the central pattern generator (CPG) based trajectory generator [15], [7], [8] also enables 1. Realization of derived walking gaits like bound, walk, gallop, and turning in hardware with no additional training, and 2. Smooth transitions from one type of gait to another. This is mainly motivated by the various locomotion behaviors realized by [16], [15] in their custom hardware. These various behaviors are then composed together and tele-operated by a remote control device.

The paper is organized as follows: Section II details the design and hardware of Stoch. Section III describes the software architecture, on-board compute platform, actuators, communication interfaces, and the sensory inputs. Section IV contains a detailed description of the control algorithm used. Finally Section V describes the experimental results of Stoch demonstrating multiple types of gaits and the associated
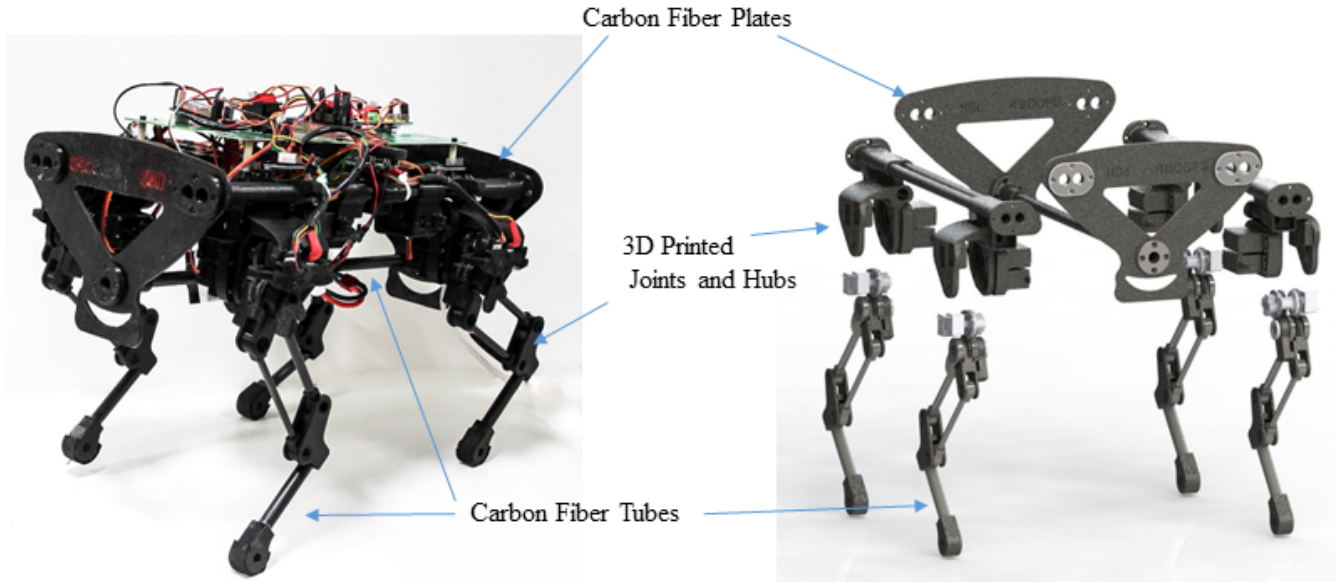
Fig. 2: Left image shows the physical hardware of Stoch, and the right image shows its rendered exploded view.

transitions.

II. MECHANICAL DESIGN

Stoch is a quadrupedal robot (see Figure 1) designed and developed in-house at Indian Institute of Science (IISc), Bangalore, India. The design philosophy behind Stoch is based on modularity, lightweight construction, ease of manufacturing, rapid repair and reproduction.

The robot design can be understood as an assembly of a central body module and four leg modules. Figure 2 shows an exploded view showing all of the modules of Stoch. The leg modules can be used in several types of morphologies as shown in Figure 3. The body consists of carbon fiber (CF) hollow tubes along its length as strong and lightweight structural members. The front and back CF plates along with 3D printed poly lactic acid (PLA) hubs form rigid central frame structure. The body also houses all electronic parts, cable routing and battery power supply. A salient feature of this design is that the leg modules slide and 'sit' on CF tubes placed along the body length. Thus, the geometrical distance between front and hind legs can be modified by changing the CF tube lengths or the assembly positions of the leg modules. Similarly, the distance between the left and right legs, which is constrained by the CF plates, can easily be customized. This feature enables user to rapidly play and experiment with the basic geometry of legs and body module.

In this section, first we discuss the overall geometry of the mechanical structure, then focus on its individual components, i.e., body and leg, design. Finally, we describe the actuator and sensor arrangement on the legs.

A. Geometry of Stoch

The Stoch is designed equivalent to the size of a miniature Pinscher dog. There are four identical legs and eight actuators mounted in the robot, which are distributed equally. Each leg has a hip and a knee joint, and the actuators provide flexion/extension in each joint. As shown in Figure 4, the Stoch is 0.4 m long in total, with 0.25 m between front and hind legs. It has a maximal hip height of 0.24 m with legs at maximum stretch, a standing height of 0.22 m, and an overall height of 0.35 m from ground to the top of electronics and cables. The robot has 0.24 m of lateral spacing between left and right leg planes axes, and a maximum width of 0.27 m. The robot weighs around 3.1 kg and 3.7 kg with the battery-pack. The estimated static centre of mass (COM) of the given system (in standing configuration similar to Figure 4) is situated 35 mm below the hip axis.

Parameter	Value
total leg length	230 mm
leg segment length	120 mm
min./max. hip joint	-45°/ 45°
min./max. knee joint	-70°/ 70°
max hip speed	461°/s
max knee speed	461°/s
max hip torque	29 kg-cm
max knee torque	35 kg-cm
motor power (servo)	16 W

TABLE I: Hardware specifications of leg.

B. Leg Design

Every leg has two degrees of freedom, one each for hip and knee. Detailed view of the leg assembly is shown in Figure 5. The key specifications of leg and actuators are summarized in Table I. The emphasis in the leg design was to keep the inertia of the moving segments minimal. To achieve this, all the actuators and transmissions are mounted to the main body. The leg mechanism is designed as a co-axial five bar linkage[12]. This is beneficial to ensure fast swing motions with minimum leg inertia.

To fabricate the mechanism, the linkages are formed with carbon fiber hollow tubes connected by 3D printed poly lactic acid (PLA) connectors. This arrangement enables us to modify the link lengths with minimal effort. To further
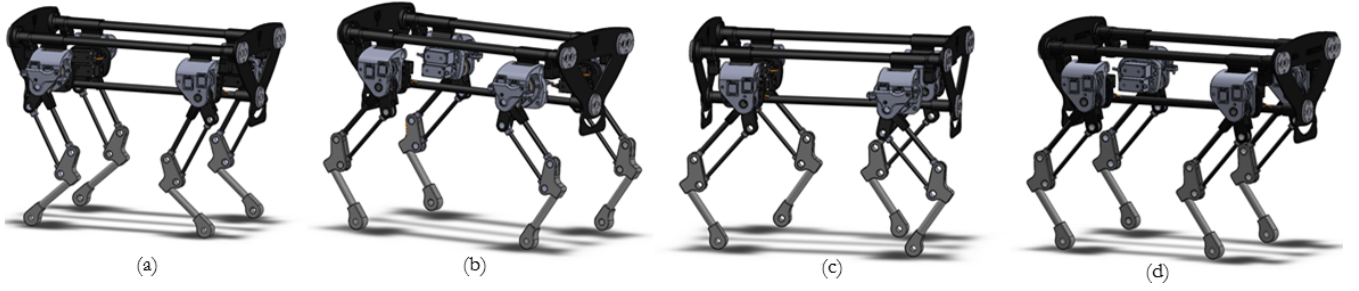
Fig. 3: Modular Configurations of *Stoch*: (a) X configuration, (b) O configuration, (c) XO configuration, (d) Natural Quadruped configuration.
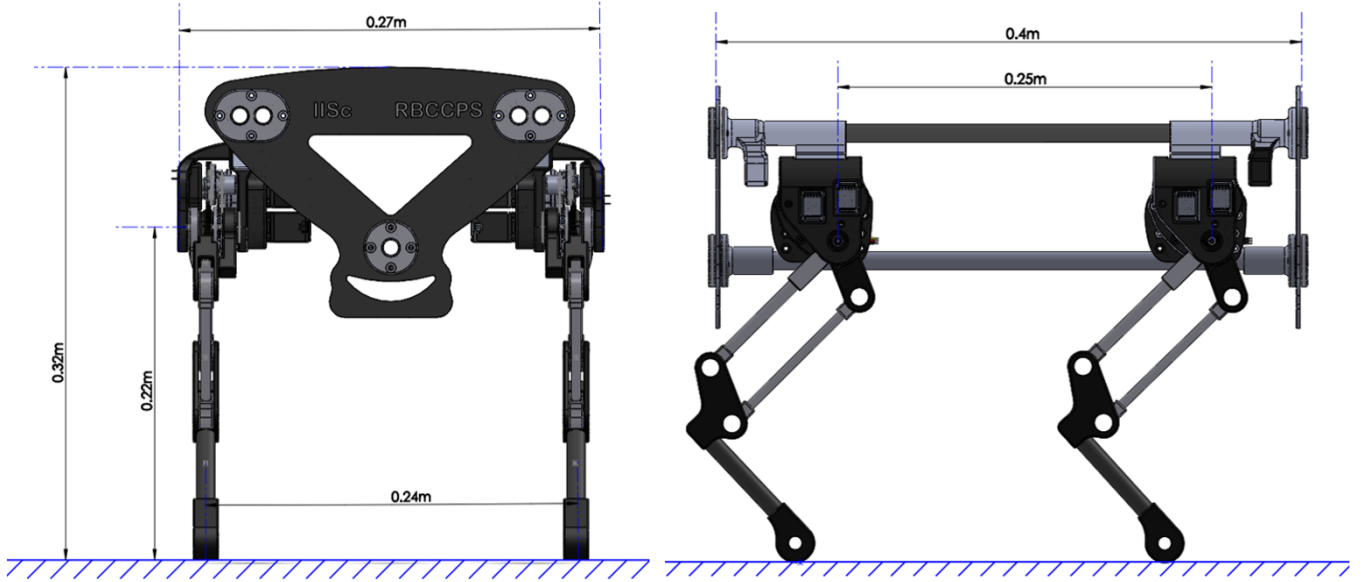

Fig. 4: Physical robot dimensions of Stoch.

reduce the swing leg inertia, we've used CF tubes as journal bearings and shaft at pivot joints instead of regular metal ball bearings. This drastically reduces the weight of the robot.

C. Actuation and Sensing Arrangement in Legs

Figure 5 (i) and (ii) depicts the assembly arrangement of servo actuators to the leg linkages and rotary encoders. For the joint actuation we have used standard servo motors (Knee: Robokits $35\ \mathrm{kg-cm}$ High Torque Servo*, Hip: Kondo 2350HV Servo†). The knee actuators are directly coupled with the 3D printed links, which in turn move the coupler links and eventually the knee joints of the legs. The joint angle sensor (rotary encoder) for knee is connected with the linkage via a laser cut steel gear. Similarly, the hip actuator transfers mechanical power to the hip link via a simple gear train (ratio $1:1$), whereas the hip encoder is directly coupled to monitor the hip servo angle via a 3D printed part.

III. ELECTRONIC SYSTEM ARCHITECTURE

In this section, we describe the electronic system architecture in light of the objectives achieved in *Stoch*. The platform

*https://robokits.co.in/
†https://kondo-robot.com/

achieved three main objectives using this architecture. First, tele-operation of the robot, which is implemented via SSH over wireless communication (Figure 7). Second, the low-level implementation of the walking controller, which passes on the desired joint configurations as commands to the joint actuators. And third, integrating sensory feedback with the walking controller. The last one is achieved by reading and recording different sensors without experiencing significant delay.

The encoders, servos and other parts selected for this work are readily available, off-the-shelf components. This enabled a quick adoption, rapid prototyping while minimizing the total cost. Also, the library associated with the hardware is readily available on GitHub, reducing the overhead associated with firmware development.

A. Tele-operation

The walking controller operates by changing internal parameters of the Central Pattern Generator (CPG), such as the frequency, phase difference and amplitude. These changes result in generation of different behaviors and motions. In this robot, these parameter changes, are made during the process execution via SSH. We selected the keyboard as the input device for remote operation. This method of control
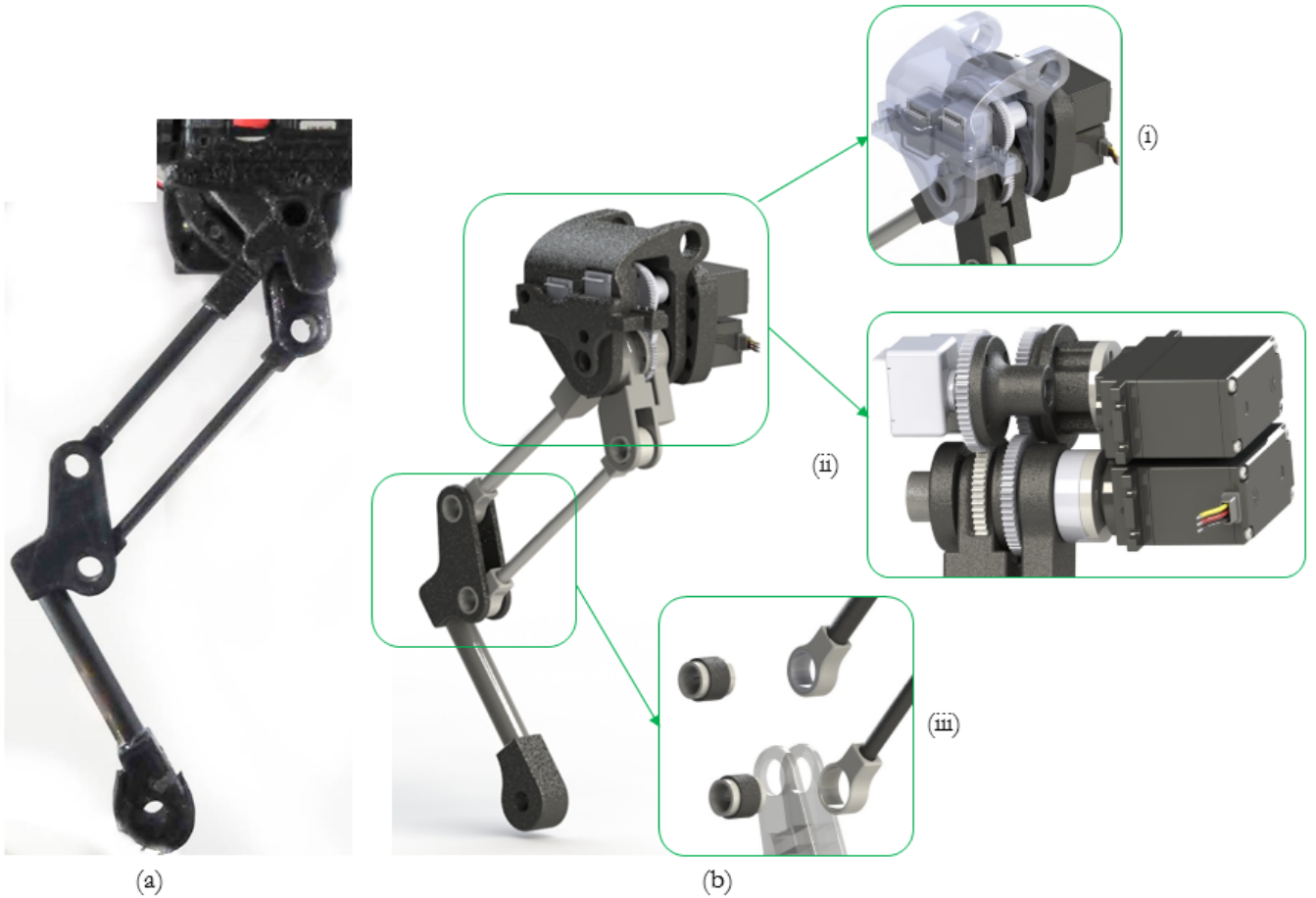
Fig. 5: (a) shows the physical leg assembly, (b) shows the rendered counterpart of the leg assembly. (b) has three parts: (i) sensor mounting and actuation hub, (ii) coupling and transmission of linkages with actuators and rotary encoders, and (iii) exploded view of knee joint and CF journals.

requires continuous process execution, without any blockage. Thus, we have used non-blocking API's from POSIX, namely, "select" to implement on Linux. Similar features can be obtained in Windows by using multi-threading.

B. Motor actuation

The actuators used in this robot are PWM enabled servo motors. To actuate these servos, we have used Adafruit PWM drivers (16-Channel 12-bit: PCA968), connected with the central compute platform (Raspberry Pi3) over I2C Bus. In this robot, there are two sets of servos (see section II-C). These servos operate at 11.1 V and 7.4 V respectively.

Equipment	Details
Joint encoder	Bourns EMS22a
IMU	MPU-9150
ADC	ADS1115
PWM driver	Adafruit PCA968
Servo	Kondo 2350HV
	Robokits RKI-1203
Computer	Raspberry Pi 3b

TABLE II: Electronics hardware specifications.

C. Sensory feedback

The sensory inputs contain signals from joint encoders, inertial measurement unit and limit switches. The angular position of each motor shaft is measured by externally coupled absolute magnetic encoders (Bourns EMS22a) at resolution of 1024 counts per revolution. Each encoder is read sequentially by the RPi3 via a multi-duplex, SPI Bus. The low-level drivers (in C) are used to pass the values to the primary process (in Python) in real-time using shared memory. In Comparison with other Inter-Process Communication (IPC) like socket, pipe, etc; the information doesn't go through the kernel, enabling direct access of virtual memory to both processes. This results in a low latency which is vital for controlling the robot. We used the limit switches for self-calibration and failure prevention at the joints. These switches are connected to an analog to digital converter(4 Channel 16 Bit) on the I2C Bus.

The center of mass of the robot, is estimated by the inertial measurement unit (IMU). The states observed by the IMU are three axis acceleration, three axis angular velocity etc. Using sensor fusion, the IMU can also estimate the roll, pitch, yaw of the robot. The IMU, MPU-9150, is placed close to the geometric center of the chassis and is connected to the I2C bus.
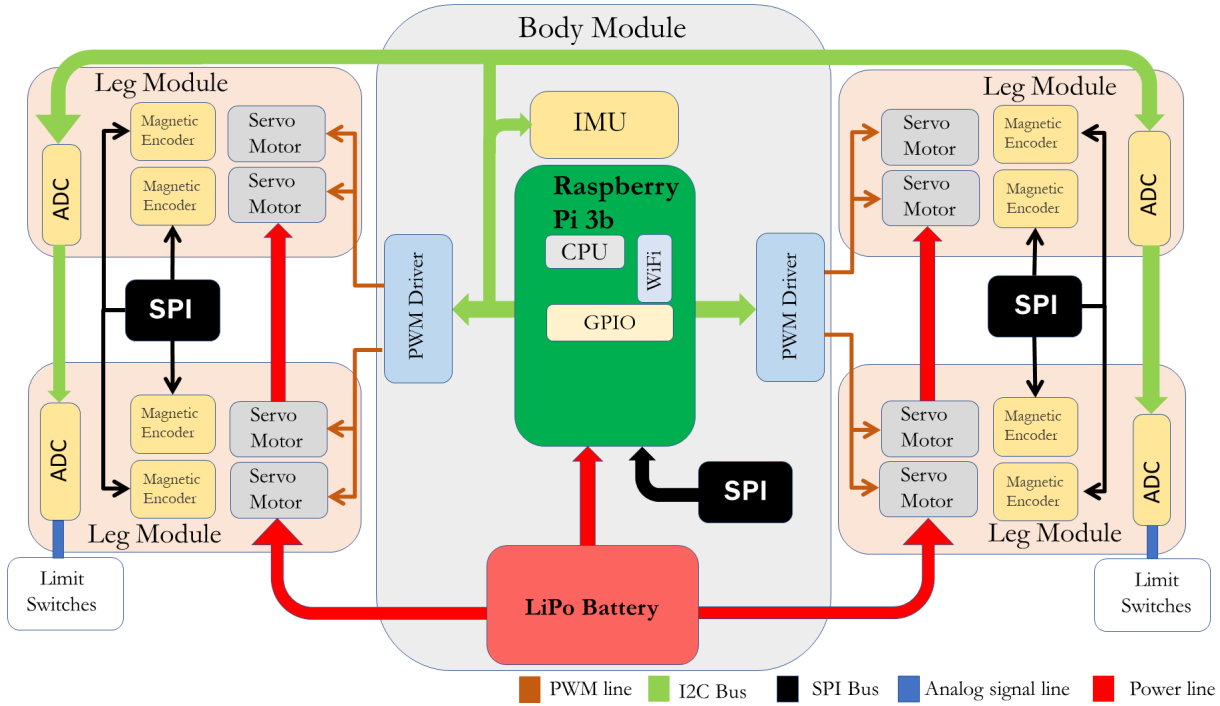
Fig. 6: Schematics of Electronics and communication network

IV. WALKING CONTROLLER

In this section, we describe the design of the control architecture of *Stoch*. The controller is based on the use of nonlinear coupled oscillators, called as *Central Pattern Generator* (CPG) [16], [17], [8]. This approach uses a group of polynomials associated with the coupled oscillators to generate the desired patterns at the end point. We will first describe the methods used to develop the rhythmic patterns using CPG, and then discuss inverse kinematics used to transform these patterns into desired joint angles.

A. Central pattern generator

Central pattern generators (CPGs) are neural networks capable of producing coordinated patterns of rhythmic activity without any rhythmic inputs from sensory feedback or from higher control centers [16]. In this work, the neural network has been replaced with nonlinear coupled differential equations. When a signal is provided with oscillation frequency (ω), the CPG begins to produce rhythmic signals of a predetermined pattern. The oscillation frequency updates the phases (ϕ) and the coupling ensures the phase offsets (Φ) between the oscillators. This difference in phase offset results in the generation of different types of gaits. The equations for the CPG are given as

$$\dot{\omega} = \alpha_\omega(\omega_d - \omega) \tag{1}$$

$$\dot{\phi_i} = \omega + \sum_{j=1}^{N} K_{i,j} \sin\left(\phi_i - \phi_j + \Phi_i - \Phi_j\right) \tag{2}$$

$$\phi_i = mod(\phi_i, 2\pi) \tag{3}$$

$$\dot{\mathbf{X}} = \dot{\phi}\mathbf{X}_d' + \alpha_{\mathbf{X}}(\mathbf{X}_d - \mathbf{X}) + \Delta \tag{4}$$

where, ω, ω_d are the current and target oscillation frequency of the legs, in radian per second. ϕ_i is the phase of the ith internal pattern generator. ϕ is the vector containing the ϕ_i. Φ_i is the phase difference between the ith leg and the first leg. $K_{i,j}$ are coupling constants. Equations 1 and (4) represent low pass filter in time domain [16]. These filters ensure there are no discontinuous changes in the variables. The low pass filter in equation 4 [17] is modified such that, periodic signal of any desired frequency can pass through but undesired signal of the high frequency can not. In eqaution 4, $\mathbf{X}, \mathbf{X}_d$ are the current and the desired state values and $\mathbf{X}_d' = \partial \mathbf{X}_d / \partial \phi$. However, eqation 4 can only be used if sensory feedback is enabled. When sensory feedback is disabled, we directly feed the desired values to the inverse kinematics solver. The constants $\alpha_\omega, \alpha_{\mathbf{R}}$ are used to determine the corner frequency of these filters. Δ is the discrete or event based input provided by the user or the sensors on-board. In this work, the filters are used in places where a discrete signal is added by the user or by some feedback in response to an event.

After the phase (ϕ_i) is determined, any function described in form of ϕ_i can be used to create the endpoint behaviour of the i^{th}leg. We have

$$\mathbf{X}_{d,i} = \mathbf{X}_d(\phi_i) = [x(\phi_i) \quad y(\phi_i)] \tag{5}$$

$$x(\phi_i) = \sum_{j=1}^{n} W_{x,j,i} f_{x,j}(\phi_i) \tag{6}$$

$$y(\phi_i) = \sum_{j=1}^{n} W_{y,j,i} f_{y,j}(\phi_i), \tag{7}$$

and their derivatives are given by

$$\dot{\mathbf{X}}_{d,i} = \dot{\phi}_i \mathbf{X}'_d(\phi_i) = \dot{\phi}_i \left[x'(\phi_i) \quad y'(\phi_i) \right] \tag{8}$$

$$\dot{x}(\phi_i) = \dot{\phi}_i \sum_{j=1}^{n} W_{x,j,i} f'_{x,j}(\phi_i) \tag{9}$$

$$\dot{y}(\phi_i) = \dot{\phi}_i \sum_{j=1}^{n} W_{y,j,i} f'_{y,j}(\phi_i) \tag{10}$$

In the above equations, $x(\phi_i)$, $y(\phi_i)$ represent the desired end point position values of the leg from the hip joint of the ith leg. This scheme can also be used to generate $r(\phi_i)$, $\theta(\phi_i)$ in polar co-ordinates which are used in the reinforcement learning framework [14]. The constant vectors, $W_{x,i,j}$, $W_{y,i,j}$, are the weights associated with the basis functions, $f_{x,j}$, $f_{y,j}$. The basis functions can range from kMPs [18], [14], trigonometric functions, Bezier polynomials [19], time responses of any dynamical system, to even data points. For example, a circular end point trajectory with trigonometric functions $f_{x,1} = f_{y,1} = \sin \phi_i$ and $f_{x,2} = f_{y,2} = \cos \phi_i$ would need a weight $W = [W_x; W_y] = [1\ 0;\ 0\ 1]$. The patterns generated in Cartesian or in polar co-ordinates are then transformed into joint co-ordinates via inverse kinematics.

B. Inverse kinematics

Having obtained the coordinates (either in the form of polar coordinates (r, θ) or Cartesian coordinates (x, y) of end points), we generate the joint angles via inverse kinematics. Figure 8 describes the leg configuration and the variables. If the end points of the robot are described in Cartesian coordinates, then 11-14 are used to determine the joint angle. In case the description is in polar co-ordinate, equation11 can be avoided, since l_3 and θ are directly provided. The inverse kinematics are given by

$$l_3 = \sqrt{x^2 + y^2}, \quad \theta = \arctan\left(\frac{y}{x}\right) \tag{11}$$

$$\Phi = \arccos\left(\frac{l_3^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \tag{12}$$

$$\zeta = \arctan\left(\frac{l_2 \sin \Phi}{l_1 + l_2 \cos \Phi}\right) \tag{13}$$

$$q_{hip} = \theta + \zeta, \quad q_{knee} = \Phi + q_{hip}, \tag{14}$$

where, l_1, l_2 are the hip and knee lengths of the robot. Since

Fig. 7: Controller and teleoperation schematic

each robot has a different reference frame associated with legs, the joint angles are transformed to obtain motor angles by using

$$M_{hip} = \pi - q_{hip}, \quad M_{Knee} = -q_{knee},$$

where M_{hip} and M_{Knee} are the motor signals generated before transformation to a pulse width modulation (PWM) signal required for motor drivers.
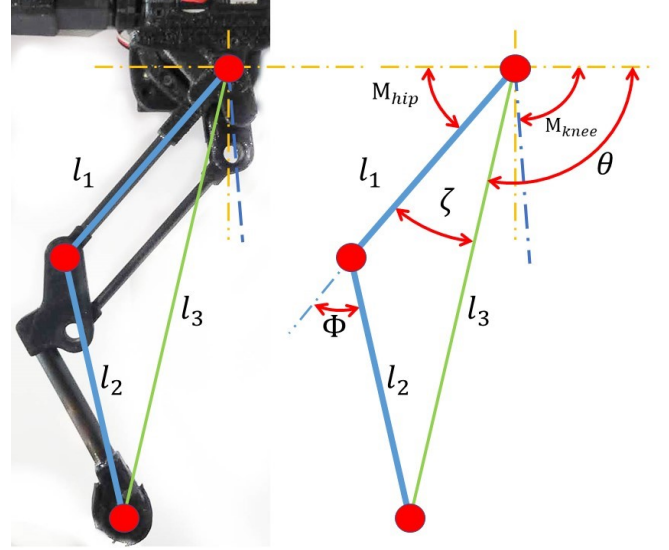
Fig. 8: Inverse kinematics of robot leg

V. EXPERIMENTAL RESULTS

In this section, we will describe various strategies undertaken to generate the different types of robot behaviors. First, we discuss the various types of flat ground walking gaits. Next, we describe the abduction free turning, and then finally describe the transitions between these gaits.

1) Flat ground walking: On the flat plane, we have introduced 6 different types of gaits, trot, gallop, bound, walk, modified trot 1 and modified trot 2. We have used the basis function based approach [14] to store the information about the trajectories in a few constant values. These constant values are stored in the weight matrix (W_x, W_y). Using the weights, we can obtain the desired end point trajectories and the velocities. The target end point trajectory for the trot gait is shown in Fig.9. We get

$$x_i = \sum_{j=0}^{5} f_{kMP,j}(\phi_i) W_{x,i,j} \tag{15}$$

$$y_i = \sum_{j=0}^{5} f_{kMP,j}(\phi_i) W_{y,i,j}, \tag{16}$$

where, x_i, y_i are the target x, y coordinates of the ith leg. $f_{kMP,j}(\phi_i)$ is the jth kinematic motion primitive [14], which essentially is a polynomial of ϕ_i.

Table III presents the comparison of speeds obtained for various gaits. A more detailed description of the basis functions are given in [14].
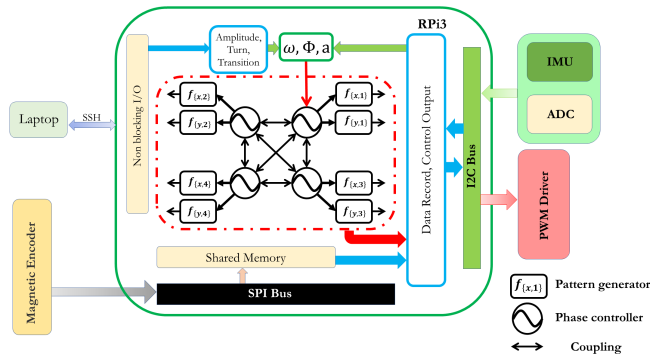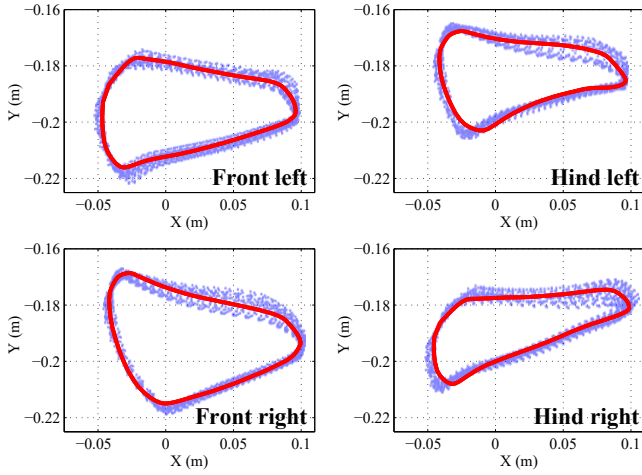
Fig. 9: Endpoint trajectories learned from the reinforcement learning based training[14]

Gait type	Trot	Walk	Gallop	Bound	Modified Trot 1	Modified Trot 2
Speed (m/s)	0.60	0.51	0.51	0.55	0.62	0.59

TABLE III: Walking speed of Stoch obtained during different gait experiments. The last two columns are the result of reshaping of the end point trajectories of trot. All the gaits are shown in the video.

2) Turning: The current model of Stoch, does not contain any abduction. This limits us to realization of abduction free/locked turning. We set the X axis (forward-backward) movement of the desired leg to zero, and use

$$x_{turn,i} = \alpha_{t,i} x_i \qquad (17)$$

$$\alpha_t = \begin{cases} [1, 0, 0, 1] & \text{left turn} \\ [0, 1, 1, 0] & \text{right turn} \\ [1, 1, 1, 1] & \text{no turn} \end{cases} \qquad (18)$$

where α_t is the vector containing the turning coefficients and $x_{turn,i}$ is the target x axis position of the ith leg. To avoid discontinuous changes in the motion, we pass the α_t through a low pass filter, equation 1. This results in a smooth trot-turn transition.

3) Gait transition: In this section, we will provide a general methodology for transitioning between the flat-ground walking gaits. In section IV-A, we described the phase offset, Φ, as a constant set of values which can be changed to obtain various gaits. This change can be achieved, on the fly, by passing the desired phase offset through a low pass filter:

$$\dot{\Phi}_i = \alpha_\Phi (\Phi_{d,i} - \Phi) \qquad (19)$$

where $\Phi_{d,i}$ is the desired phase offset of the ith leg, and Φ_i is the current phase offset of the ith leg. This allows for a smooth change in the phase values.

A. Video Result

Video results of walking experiments on the *Stoch* are available on `https://youtu.be/Wxx9pwwTIL4`. Specifically, we show trot (Fig. 10), bound, gallop, walk gaits on the robot. Additionally turning and gait transitions are also shown. Further, the animation and physical assembly is also shown in the video for easy understating of the *Stoch* design.

VI. CONCLUSION

This work presents a custom built quadruped robot along with its design, software, control framework and experimental validation on the hardware. Compared to other existing quadruped robots, our platform requires less resource and costs under USD 1000. Several gaits were realized such as walk, trot, gallop, and bound. Additionally, the robot can turn with a small radius via CPG based gait transitions, without the use of abduction. The robot reached a maximum forward speed of 0.6 m/s. An open-loop controller was used to set the speed and direction in all these experiments. Future work involves increasing the payload, and also incorporate external sensors for closed-loop control of the quadruped.

REFERENCES

[1] H.-W. Park, S. Park, and S. Kim, "Variable-speed quadrupedal bounding using impulse planning: Untethered high-speed 3d running of mit cheetah 2," in *Robotics and automation (ICRA), 2015 IEEE international conference on*. IEEE, 2015, pp. 5163–5170.

[2] M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, C. D. Remy, and R. Siegwart, "StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion," in *Adaptive Mobile Robotics*. World Scientific, 2012, pp. 483–490.

[3] "SpotMini robot," https://www.bostondynamics.com/spot-mini/.

[4] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 38–44.

[5] "Laikago robot," http://www.unitree.cc/.

[6] H. Kimura and Y. Fukuoka, "Biologically inspired adaptive dynamic walking in outdoor environment using a self-contained quadruped robot:'Tekken2'," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1. IEEE, 2004, pp. 986–991.

[7] A. Spröwitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert, "Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 932–950, 2013.

[8] A. T. Sprowitz, A. Tuleu, M. Ajaoolleian, M. Vespignani, R. Moeckel, P. Eckert, M. D'Haene, J. Degrave, A. Nordmann, B. Schrauwen *et al.*, "Oncilla robot: a versatile open-source quadruped research robot with compliant pantograph legs," *Frontiers in Robotics and AI*, vol. 5, p. 67, 2018.

[9] W. Bosworth, S. Kim, and N. Hogan, "The mit super mini cheetah: A small, low-cost quadrupedal robot for dynamic locomotion," in *Safety, Security, and Rescue Robotics (SSRR), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 1–8.

[10] "AIBO robot," https://us.aibo.com/.

[11] Rhex, "RHex robot," https://www.bostondynamics.com/rhex/.

[12] A. De and D. E. Koditschek, "Vertical hopper compositions for preflexive and feedback-stabilized quadrupedal bounding, pacing, pronking, and trotting," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 743–778, 2018.

[13] "OpenCat robot," https://www.hackster.io/petoi/opencat-845129.

[14] A. Singla, S. Bhattacharya, D. Dholakiya, S. Bhatnagar, A. Ghosal, B. Amrutur, and S. Kolathaya, "Realizing learned quadruped locomotion behaviors through kinematic motion primitives," *arXiv preprint arXiv:1810.03842*, 2018.
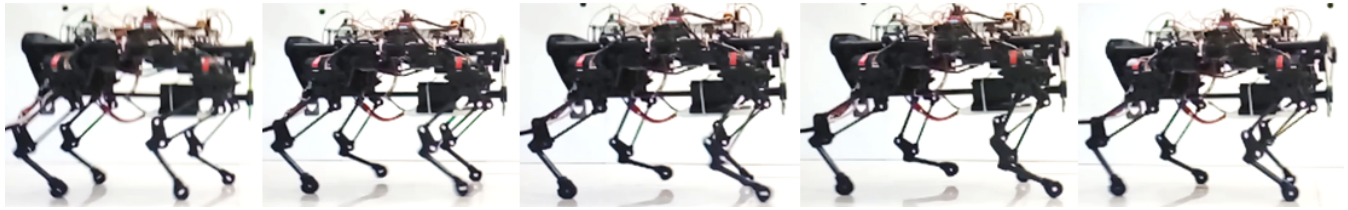
Fig. 10: Trot sequence of Stoch on flat terrain

[15] A. T. Sprowitz, A. Tuleu, A. J. Ijspeert *et al.*, "Kinematic primitives for walking and trotting gaits of a quadruped robot with compliant legs," *Frontiers in computational neuroscience*, vol. 8, p. 27, 2014.

[16] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: a review," *Neural networks*, vol. 21, no. 4, pp. 642–653, 2008.

[17] M. Ajallooeian, J. van den Kieboom, A. Mukovskiy, M. A. Giese, and A. J. Ijspeert, "A general family of morphed nonlinear phase oscillators with arbitrary limit cycle shape," *Physica D: Nonlinear Phenomena*, vol. 263, pp. 41–56, 2013.

[18] F. L. Moro, A. Spröwitz, A. Tuleu, M. Vespignani, N. G. Tsagarakis, A. J. Ijspeert, and D. G. Caldwell, "Horse-like walking, trotting, and galloping derived from kinematic motion primitives (kmps) and their application to walk/trot transitions in a compliant quadruped robot," *Biological cybernetics*, vol. 107, no. 3, pp. 309–320, 2013.

[19] G. A. Castillo, B. Weng, A. Hereid, and W. Zhang, "Reinforcement learning meets hybrid zero dynamics: A case study for RABBIT," *arXiv preprint arXiv:1810.01977*, 2018.