Novel Bio-inspired Inverse Kinematics for Fault-Tolerant Multilegged Robots

Ashwin Kumar, Mihir Singh Kothari and Santanu Mitra*

Mechanical Engineering Department, Shiv Nadar University, Greater Noida, Uttar Pradesh, India,

Abstract: Nature has the best design, which has evolved over millennia for sustainability. The current research aims to provide a generalized algorithm for multi-legged fault-tolerant walking robots with 3 degrees of freedom per leg. An inverse-kinematics-based model has been implemented for strategic foot placement. Animals walk over rough ground and move each leg separately, often falling in a non-standard gait. The present work allows a provision for the designed robot to move over uneven terrain in a controlled manner, using adaptive gaits while maintaining a constant hip height. One of the key features of the developed algorithm is the ability to adapt locomotion and continue functioning even if some legs are injured. An experimental study was carried out in order to validate the effectiveness and applicability of the current algorithm. As far as the present knowledge of the authors goes, there is no systematic study of developing a generalized algorithm for fault-tolerant multi-legged robots available in the open literature.

Keywords: Hexapod Robot, Design and Control, Legged robots, Bio-inspired Robots, Inverse Kinematics.

1. INTRODUCTION

Walking robots provide greater functionality as compared to wheeled robots, especially in areas where the terrain is unmapped and widely varying. The Legged robots have a higher number of degree of freedom, which makes them relatively difficult to design and control, but they are much more suited for traversal and navigation in areas such as disaster zones. Bio-inspired robotics aims to use these evolved designs to create structures and machines best fit for achieving certain functions. Walking insects have different kinds of gaits for the different kinds of terrains that they may find themselves in. For a robot to perform a walking gait, at least three legs must stay on the ground in order to keep it stable. Taking a hexapod as an example, on smooth terrain it works on a tripod gait, where three legs are actuated simultaneously, which is the fastest possible walking gait in hexapods. In contrast to that, on rough terrain, the hexapod chooses a wave gait, where each leg is actuated one by one, thus making it the most stable walking gait. Along with the tripod and wave gaits, many other gaits, such as the bipod and ripple gaits, are considered to be the standard gaits for legged robots. Other than these, all other gaits are called non-standard. An efficient legged robot design would be one that autonomously and dynamically adapts to the terrain that it is currently present in. It would also be able to walk on standard as well as non-standard gaits in order to get the fastest possible speeds while ensuring stability. An algorithm that allows each leg to work independently at different heights would ensure that the robot has the ability to tackle rough terrain while keeping the hip height constant. It should also not require high computation power. If such an algorithm could be made, it would prove to be a starting point for anyone who wishes to build a legged robot, and hence they can further develop its functionalities to perform any task they want it to.

The development of hexapods began in the 1950s, with manually controlled, rigid motion bots. Different mechanisms, such as electric drives and pneumatics, were experimented with to come up with a sturdy design. In the 1970s, Moscow State University developed a hexapod with proximity sensors and contact sensors that could navigate an obstacle course. The robot had a simple design with tubular axial chassis in the earlier days with articulated legs, possesses three DoFs [1]. In 1983, Carnegie-Mellon University created a hexapod that could use multiple gaits to walk on different kinds of terrain [2]. It was controlled using a combination of hydraulic feedback, computer control, and manual control and was powered by a combustion engine. Subsequently, in 1997 researchers have focussed more on energy efficiency and some of their current disadvantages include greater complexity and the development cost, low energy efficiency [3]. In the mid-90s, the Jet Propulsion Lab came up with a design that used telescopic legs for motion, unlike other hexapod designs that used legs that folded [4]. With the advancement in control systems, electronic components, and computational strength, hexapods have become highly advanced in the recent past. RHex from Boston Dynamics used a novel approach to tackle

*Address correspondence to this author at the Mechanical Engineering Department, Shiv Nadar University, Greater Noida, Uttar Pradesh, India;
E-mail: santanu.mitra@snu.edu.in

E-ISSN: 2409-9694/21

terrain by using only six motors and specially designed wheels, creating a minimal and efficient design [5]. Another design, called the MorpHex MEIII with flexible limbs, is one of the most agile version of a hexapod. It could roll its legs in to form a sphere and perform a rolling motion on flat surfaces, offering very high velocity and it could roll over 2 inch height obstacles [6]. Later researchers came up with a study which gives an overview of the state of the art on hexapod walking robots. In this study the authors referred to the early design solutions and the most recent achievements [7]. An interesting study was carried by Zak *et al*. on basic gaits of hexapod, which can walk even with three legs and adopting different locomotions and the robot was equipped with different sensors and actuators [8]. A study on the advantage of inverse kinematics applied to develop a solution for the animating arbitrary articulated systems and humans [9]. Several fault-tolerant gaits are also studied, and among several algorithms developed in the past, Yang particularly emphasized crab and turning gaits [10]. A very recent study on radially free distributed legs of a hexapod robot can show improvement in their performance in lifting weights or walking on a slope [11]. A computational model and experimental study was carried out by a few researchers in Turkey and used a novel mechanism, and only two actuators were used in the system to propel [12]. Niwa *et al*. studied the control of multi-legged walking robots on even terrain. The applicability of the simulation also assessed by applying the algorithm on 20 legged robot[13]. Another interesting study [14] describes a realistic mechanism for a six-legged spider-like robot. The researchers have also adopted inverse kinematic based approach to analyze the robot. Quingsheng *et al*. [15] used ADAMS platform to study the hexapod mobile robot and the inverse kinematics. Many other different types of hexapods have already been developed, and ongoing research promises to improve the field even further.

In recent times however, focus has shifted towards designing legged robots based on machine learning and genetic algorithms. The present work aims to provide a simple and systematic framework for designing multi-legged robots with 3dof legs, capable of dynamic gait updates, fault tolerance and heading changes.

2. DESIGN AND MODELING

In applied robotics, the prime objectives are dexterity and the low cost. Our novel design focuses on achieving steady trajectory of the hexapod body to capture steady images. Autonomy, a critical component when it comes to failure of one of the component or legs of the robot. Our design of algorithm should be such that it can adapt the gait on its own without explicitly programming. Self-adapting gait is very difficult to achieve and its required strategic algotithm which will get constantly updated as soon as any fault occur. The proposed algorithm aims to provide a simple and precise control architecture for multi-legged robots, which can serve as a starting point for high-level application. Using an Inverse-Kinematics based approach and dividing the flow of control into layers, the algorithm successfully separates various parts of the computational process, lending it a modular nature. This creates a basic system capable of housing various advanced modules on top of it, as demonstrated with the test case. The minimal abilities of a legged robot are designed into the code such that it can be applied directly to designs with completely different body shapes, leg orientations, degrees of freedom and geometries, with just minor calibrations.

A simple 3D simulation was also created using the pygame library on python, which can visualise the results of the algorithm in a virtual environment.

To test the algorithm, a hexapod robot was designed and manufactured as a test case. Throughout the design process, minimization of volume and weight was kept in mind and we had to carry out multiple iteration in order to achieve weight reduction and required strength. The design was created on SolidWorks, keeping in mind the constraint of the size of servo motors. It was an iterative process, by the end of which we had a model based on MG90 servos. A hexapod was chosen as the test case because of the following reasons:

1. Having 6 legs ensures that enough legs stay on the ground, allowing for greater stability, without compromising on mobility.

2. In walking mechanisms, hexapods are faster than quadrupeds.

3. In demanding situations, 2 legs can be repurposed to perform tasks while the remaining 4 continue to perform the task of locomotion.

The base or main body is the primary component of the hexapod. It is meant to house the controller and serves as the hub for all the legs. A regular hexagonal shape was chosen for the base to achieve directional

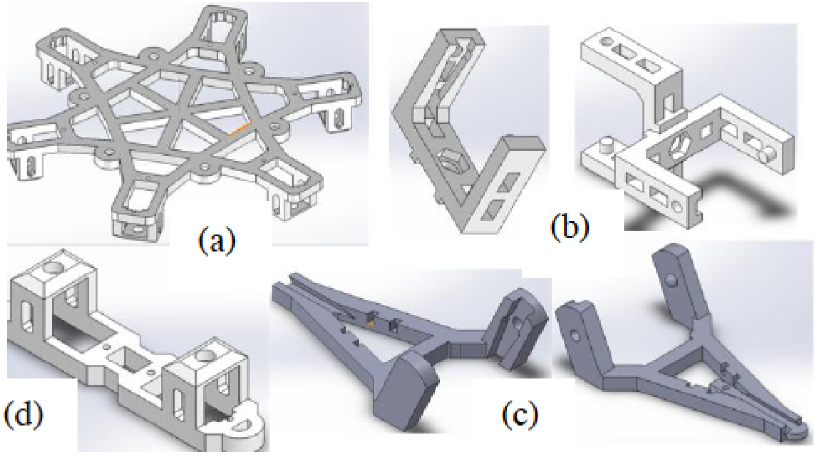

Figure 1: Design of parts for hexapod body: (**a**) Main Body (**b**) Coxa (**c**) Tibia (**d**) Femur.

symmetry and perfect weight balancing at the center. The star shaped geometry minimized the weight while giving us the maximum reach for each leg. The central region provides the structure stability. The leg was designed to have 3 degrees of freedom, for which three parts were designed as it is shown in Figure **1**.

Coxa: Connects the main body to the rest of the leg. Its primary purpose is to provide rotation about the z-axis. Another factor considered was the foldability of the leg, such that the calf tip could fit in the shoulder.

Femur: Main structural component of the leg. It houses two servo motors which control the rotation of the joints about a horizontal axis. Both servos were housed on the femur as it provides a unique geometry which allows the calf to be folded into the femur, converting it into a smaller 2 DoF robot.

Tibia: The tibia is the member of the leg which will directly touch the ground. The inner face of the tibia has been designed to act like a geometric negative to the femur, allowing them to fit together snugly when closed.

Sensor: A slider mechanism was included in the tip of the tibia, which would be triggered whenever the leg touches the ground. With a suitable filter applied, the controller would read the input and make the legs react accordingly.

The weight of the robot was calculated, and after an analysis of torque requirements, the servo motor Tower Pro MG90 were chosen for the joints. For quick prototyping (Figure **2**) and testing, the Arduino Uno was used, along with a couple of servo controller boards PCA9685. A Raspberry Pi was used as a master controller, while relaying a live video feed to the person controlling. A PCB was designed on which the servos, sensors, motor controller and power source were connected.

2.1. Inverse Kinematics Analysis

In this study we have used IK to calculate what angles the servos should be at, in order to reach a specified point that the algorithm for motion generates. This is useful in making the hexapod move all its legs in the direction in which the robot wants to achieve its

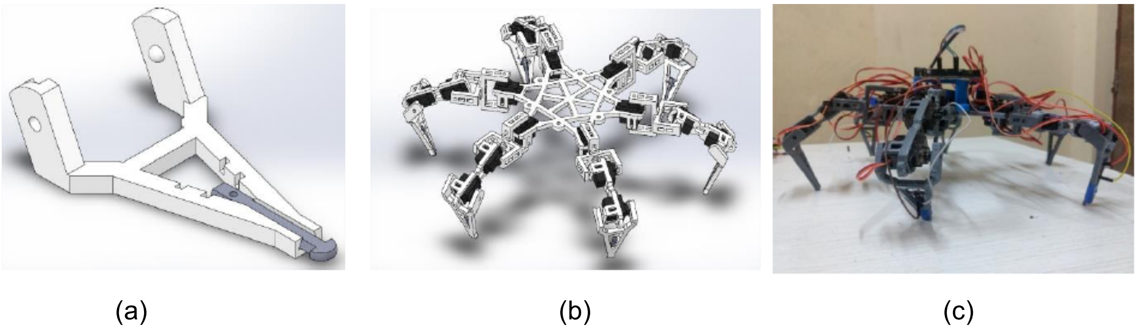

Figure 2: (**a**) Slider Assembly CAD (**b**) Full Assembly, (**c**) Physical prototype, 3D printed using ABS.

motion. Only the equation for the straight line motion of the leg tip is required for the hexapod to move in the direction of the line so that all joint angles can be evaluated from the algorithm and the desired motion can be achieved. This approach also helps in making the algorithm modular. The algorithm for one leg can be directly applied to other legs by changing only one parameter, the orientation. It also ensures that the motion of the hexapod remains dynamic. If an obstacle is encountered on one leg, further iterations of that leg will be based on the position of the leg when it hit the obstacle. Our present consideration is the 3DoF of legs as it is shown in the Figure **3**. The design requires two constraints when modelling for forward motion:

1. Keeping the height of the body from the ground constant

2. Keeping the perpendicular distance of the leg tip from the body constant when the leg is in motion.

Having a low degree of freedom per leg saves space. On the other hand, having a higher degree of freedom provides the legged robot with more functionality.

Perpendicular distance $P = L_3 + L_2 \times sin(T_2) + L_1 \times sin(T_1)$ (1)

Height of bot $H = L_1 \times cos(T_1) + L_2 \times cos(T_2)$ (2)

Where L_1, L_2, L_3 are the length of the coxa, femur and tibia respectively and T_1, T_2, T_3 are the angular positions of the leg modules with respect to reference frame. It is important to know whether the leg can reach the coordinates passed from the main function before solving the inverse kinematics equations and moving the servos. Thus, a mathematical function was created to check whether a given point is within reach or not, using the geometry of the legs as parameters.

The first step for this involves getting the distance between the leg tip and the axis of the second servo.

$$Dist = \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z)^2} \tag{3}$$

Here (x, y, z) are the coordinates of the leg tip, and (x1, y1, 0) are the coordinates of the axis of the coxa-femur joint. Then, the result of the following expression determines whether the point is in reach or not.

$$\left| \frac{Dist^2 + L2^2 - L3^2}{2 \times Dist \times L2} \right| \le 1 \tag{4}$$

`L2` is the axis-to-axis length of the femur, and `L3` is the axis-to-tip length of the Tibia. If this condition evaluates to True, then the point is known to be in reach and vice versa.

2.2. Leg Heading

Instead of the using usual methods to move legs, this study attempts to provide a direction vector for the leg tip to move along. This does not mean that the legs are reoriented to move in a particular direction. Instead, a leg heading is defined, which uses the current position of the leg tip. Using the incremental inverse kinematics algorithm, the leg is moved along the ground in that direction. The heading of each leg, thus, defines the angle made by its own direction (the radially outward vector from the robot center) with the global heading.

2.3. Algorithm: Novel Approach

This section gives a detailed explanation of the code when applied to a hexapod design with 3 DoF legs and a radially symmetrical design. For the algorithm, each variable concerning leg motion is considered as an array of 6 elements, one for each leg. The most relevant ones are listed here.

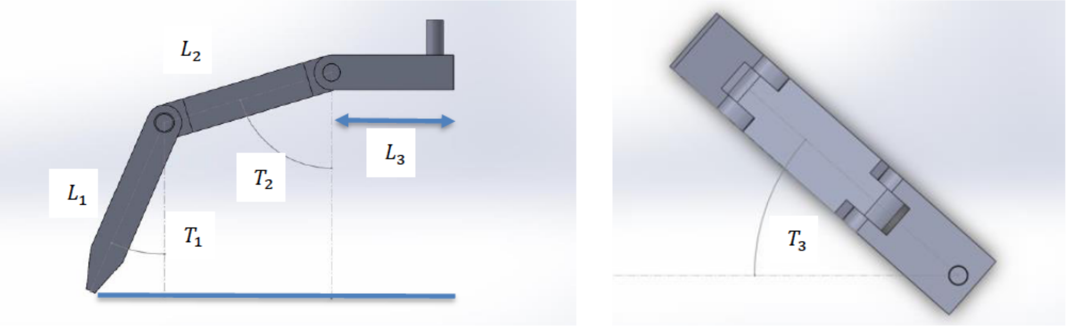

Figure 3: 3-DoF Leg, (a) Side view (b) Overhead view.

T_1, T_2, T_3 : the three angles corresponding to each joint

`x, y, z` : coordinates of the leg tip

`heading` : direction of motion w.r.t. the radially outward vector

`phase` : position in the loop at which leg's duty cycle starts

Using these variables, the algorithm manipulates each leg as a separate entity, where decisions are made only by taking its relative position on the bot as a factor. Having independent frames of reference for each leg also allows for adaptability and modularity in the number of legs. The algorithm provides an adaptable control architecture, allowing for a smooth transition between gaits and a modular bot design. The algorithm is split up into layers as follows

At this first level (root level), the master controller (a microcontroller connected to a camera or a human operator) takes input based on the environment, and an action is decided. This action might be to climb over a rock, to change the orientation of the bot, or to move in a particular direction. Currently the algorithm encompasses moving towards a given objective direction. The root level code accepts directional information as input, and decides what angle the bot has to move towards with respect to its original 0° direction (Figure **4**). The master controller also decides what speed the bot should move at based on this input. This is taken as manual input. The root level is primarily concerned with the operation of the bot as a whole, and acts as a control center for various subsystems.

In the first level, control comes down to the individual leg based coordinate system. At this level, each leg's independent heading is set, based on the heading of the bot. This allows the robot to move in any given direction irrespective of its current orientation, thereby fully utilizing its symmetrical design. For example, if the objective is along the 0° direction, each leg will have a heading as shown in Figure **4**. These are saved in an array accessed globally. Any motion to the leg tip automatically adds these offsets by considering the projection of this vector. This helps by giving a reference to each leg, so that a similar motion along different angles can propel the bot forward.

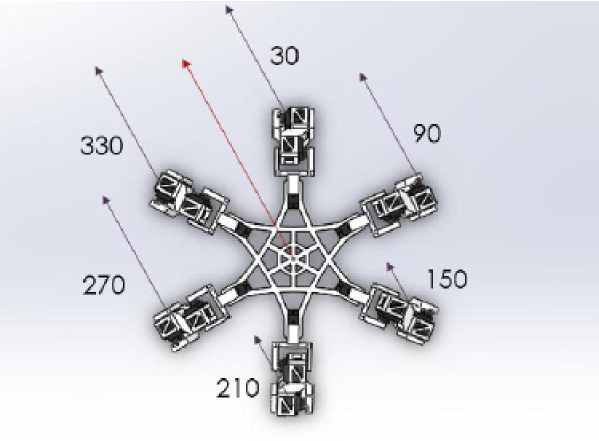

Figure 4: Individual leg headings based on bot heading (Level 1).

In the next level (second level), pre-movement calculations are performed. Certain parameters are calculated for repeated motion in order to move towards the heading direction. At this stage, based on the ground conditions and maximum angular speed of the servos, a velocity of movement is decided. This velocity is used to calculate the duty cycle, which governs which gait the bot will follow. A duty cycle of 50% makes it a tripod gait, 66% makes it bipod gait and 83.3% makes it wave gait. The duty cycle is a variable taken in as an input based on the terrain and the required walking stability. The algorithm has the capability to alter gait in a continuous manner, just by changing the duty cycle. For example, if 3 legs are required on the ground at any instant of time, the duty cycle is calculated as

$$Duty\ Cycle = \frac{Number\ of\ legs\ on\ ground}{Total\ number\ of\ legs} * 100\% \tag{5}$$

Now, to control speed, the time spent by each leg in the air is fixed for all gaits. This is the time spent out-of-phase, and is the corresponding percentage of the duty cycle loop. This dictates the speed of movement, limited by the servo speeds and processing time per iteration. A variable called `speedvar` is used for this study. For a `speedvar` of 25, in a tripod gait, the loop will iterate over 50, implying half the time is spent in the air, and half on the ground. Since speed is primarily controlled by the processor speed and the motors' angular velocity, the duty cycle and the `speedvar` are only approximate controls of speed, instead of providing an exact measure. Doubling the `speedvar` will halve the speed, but the exact value of the said speed can only be ascertained by physical tests or hardware specific calculations. Similarly, a change in

Table 1: Leg Phases for Different Duty Cycles in Percentages

Duty Cycle	Leg Phases (in %)						Gait
50%	0	50	0	50	0	50	Tripod
66%	0	66	33	0	66	33	Bipod
83.3%	0	83.3	66.6	50	33.3	16.6	Wave

the duty cycle will proportionally alter the current speed. In a bipod gait with 66% duty cycle (`globCount`=3*`speedvar`), the speed will be two-thirds of the speed in tripod gait with 50% duty cycle (`globCount`=2*`speedvar`). Speed is continually variable, by either changing the `speedvar` or the duty cycle.

At this level (third level) uses the duty cycle as an input, and assigns a phase to each leg, numbered clockwise from 0 to 5. A full phase cycle over the `globCount` range covers one full step and return to original configuration of all legs. Each leg is defined to be in phase while it is on the ground, and out of phase when it is in the air. A leg is in phase when the current `globCount` lies between the current leg's phase and the next leg's phase. Phase is decided based on the duty cycle. All legs having the same phase move together.

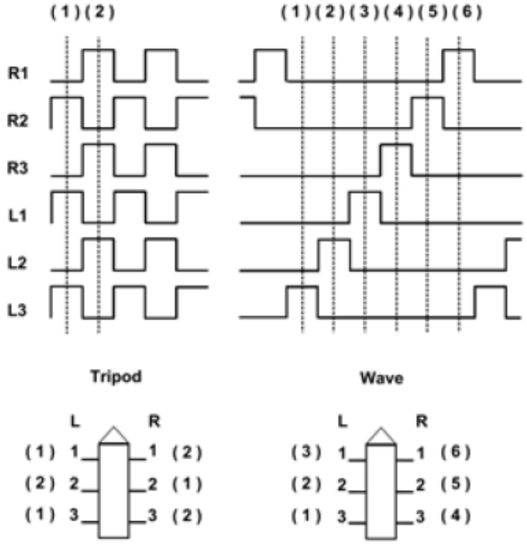

Figure 5: Tripod and wave gait activation patterns (Tedeschi, 2014).

It is not necessary for the duty cycle to take on values that lead to one of the standard gaits. Taking values like 75%, which leads to something between 3 and 4 legs on the ground at any instant, doesn't make sense in the context of a hexapod, but its utility is evident when working with an octopod or a quadruped, where it results in 6 and 3 legs remaining on the ground at any instant respectively. This also gives odd-legged robots the ability to move in a constant manner, taking gaits like 60% for a pentapod, which lends it a tripod like gait Figure **5**.

All earlier level 1, 2 and 3 can be considered the preprocessing stage, and level 4 and 5 are the loop stage. In level 4, phase is cycled over the `globCount` range. Each leg that is in phase is moved backward by an increment equal to the motion vector's projection on the local x and y axis, calculated in the previous step as the in-phase dx and dy. This pushes the bot forward by virtue of friction. Legs that are not in phase are moved up and forward in an arc. The vertical motion is a function of the physical parameters, and should be manually calibrated, while the horizontal motion uses the out-of-phase dx and dy. In this implementation, there is no environmental input, hence it is ideal for motion on a flat plane without obstacles or pits. This step outputs a set of new coordinates x, y and z for each leg, which are sent to the final level.

In the final level, the received x, y and z coordinates are sent to an inverse kinematics function to calculate the angles required by each servo to move the tip to that position. For 3 DoF legs with 180 degree range of motion, the inverse kinematics code is as follows, where L_1, L_2, L_3 are the lengths of the coxa, femur and tibia, measured axis to axis. Any corrections needed to compensate for motor orientations are made in this step. After the angles have been calculated, control is transferred to a function which moves servos. Each servo is instructed to move to the new position through this function, which needs to be defined as per the hardware being used in the physical model. The system iterates over lower levels until a higher level variable's state changes. For example, a change in duty cycle will take the loop back to level 2, and a change in heading will take it back to level 1 as depicted in Figure **6**.

2.4. Algorithm Flowchart

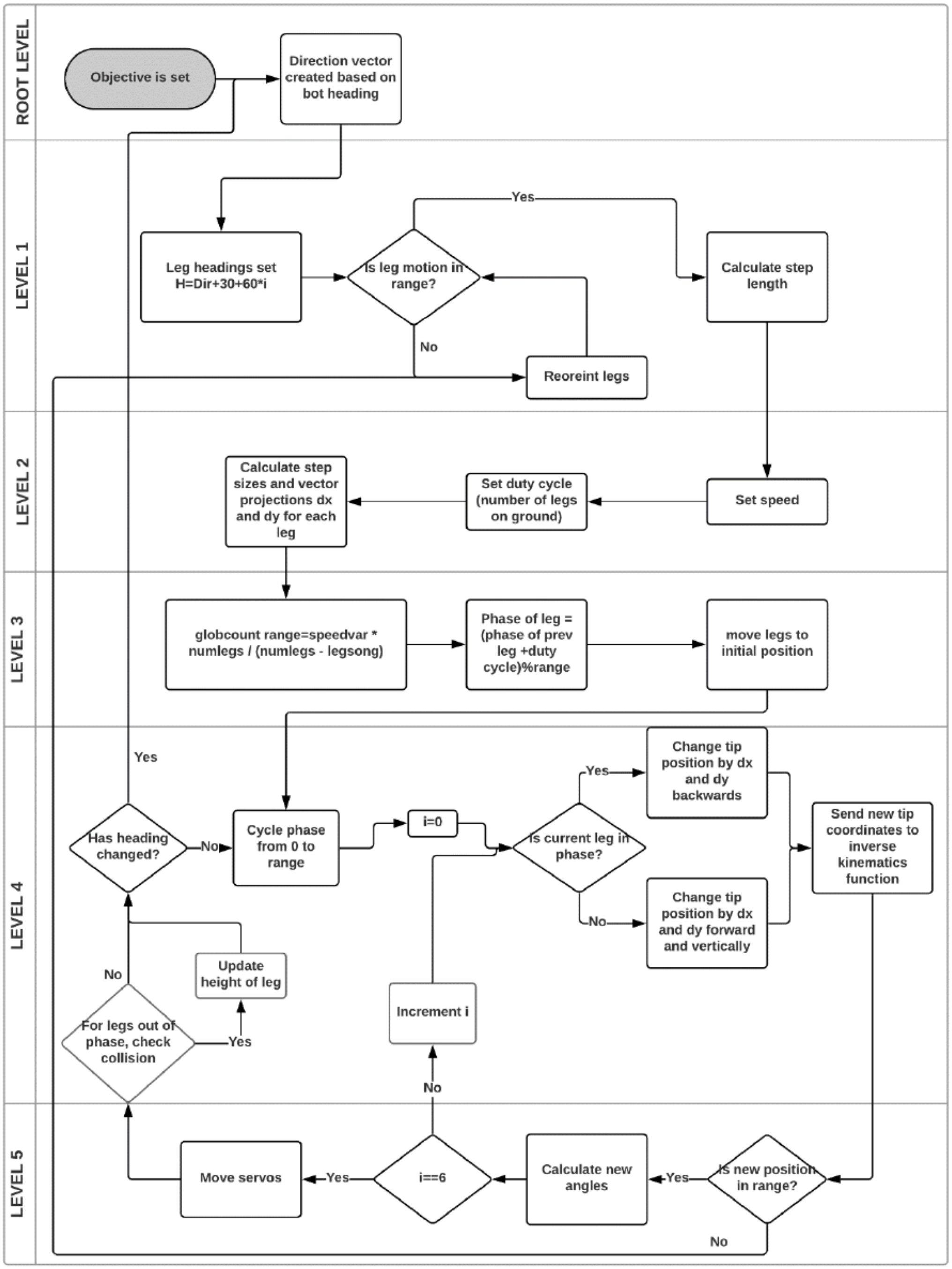
ROOT LEVEL
LEVEL 1
LEVEL 2
LEVEL 3
LEVEL 4
LEVEL 5
Objective is set
Direction vector created based on bot heading
Leg headings set H=Dir+30+60*i
Is leg motion in range?
Yes
No
Calculate step length
Reoreint legs
Calculate step sizes and vector projections dx and dy for each leg
Set duty cycle (number of legs on ground)
Set speed
globcount range=speedvar * numlegs / (numlegs - legsong)
Phase of leg = (phase of prev leg +duty cycle)%range
move legs to initial position
Yes
Has heading changed?
No
Cycle phase from 0 to range
i=0
Is current leg in phase?
Yes
Change tip position by dx and dy backwards
No
Change tip position by dx and dy forward and vertically
Send new tip coordinates to inverse kinematics function
Update height of leg
No
For legs out of phase, check collision
Yes
Increment i
No
Move servos
Yes
i==6
Calculate new angles
Yes
Is new position in range?
No

Figure 6: Algorithm flowchart based on hexapod design.

3. SIMULATION AND EXPERIMENTAL RESULTS

A simple 3D GUI was created to help visualize the output of the algorithm. A series of study on different gaits for legged robots were carried out to assess the applicability and versatility of the algorithm. Following Figures **7**-**15** are images of the hexapod for the Tripod, Bipod Wave gaits. Each set of images consists of screenshots of leg configurations at equidistant intervals over a single `globCount` loop. If we number the legs clockwise from from the heading direction then in one phase leg no. 1, 4 and 5 are on air and other legs are on the ground and in the next cycle leg no. 2, 3 and 6 are on the air and moving forward to achieve tripod gait. This nomenclature of legs is different than Tedeschi [7].

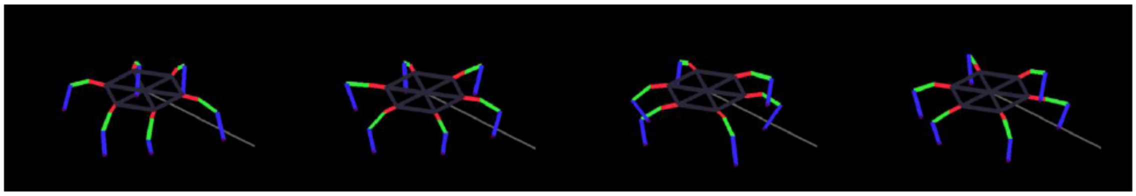

Figure 7: Tripod gait simulation, isometric.

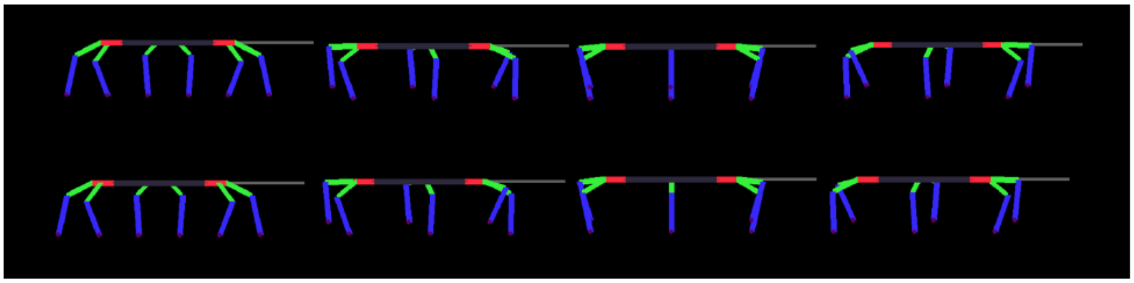

Figure 8: Tripod gait simulation, side view.

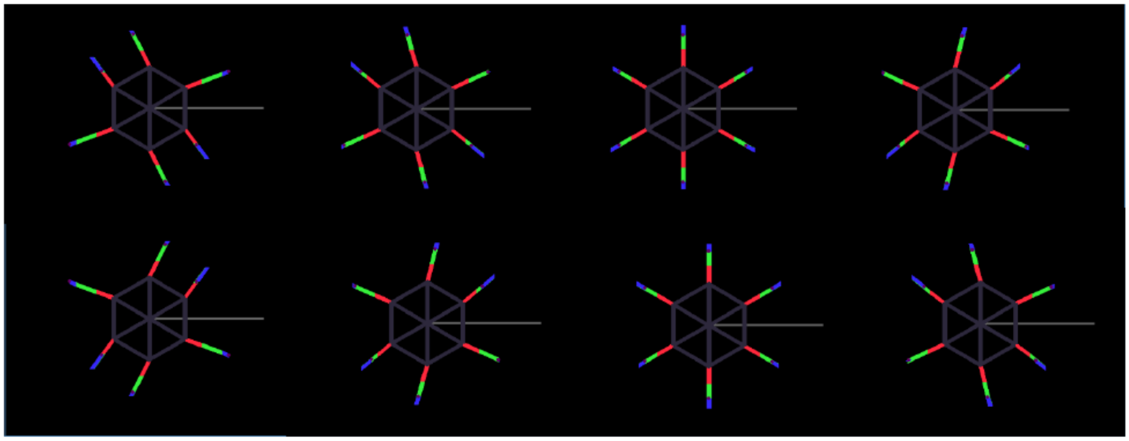

Figure 9: Tripod Gait simulation, Top view.

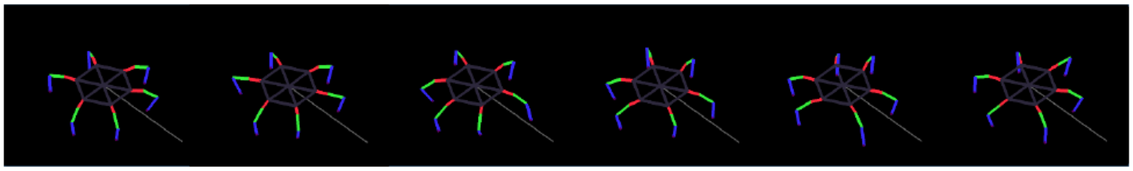

Figure 10: Bipod Gait simulation, isometric view.

3.1. Tripod Gait

From the various screenshots of the tripod gaits, our developed simulation tools' applicability has been assessed. Three views, namely isometric, side, and top views, have been represented. In isometric view Figure **7** we have shown locomotion in four screenshots for the tripod gait. Similarly, in Figure **8** and Figure **9** screenshots, the tripod gait was captured very well.

Next bipod and wave gait also have been simulated and shown in Figure **10** to Figure **15**.

3.2. Bipod Gait

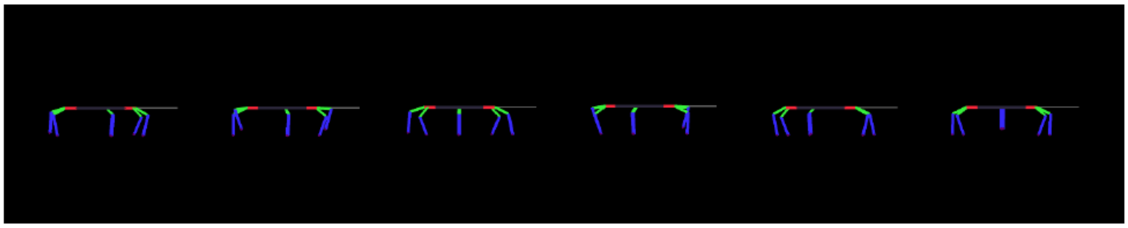

Figure 11: Bipod gait simulation, side view.

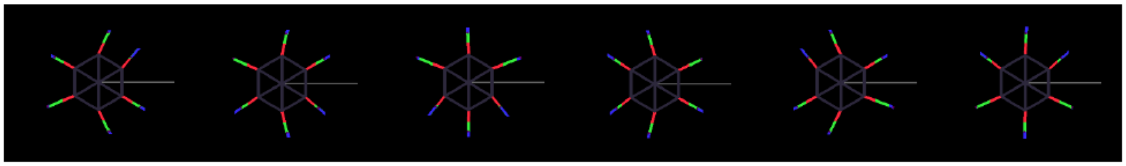

Figure 12: Bipod Gait simulation, top view.

3.3. Wave Gait

3.4. Modifications and Modularity

Results also were obtained for multi-legged bot The algorithm can also work on any symmetrical or non-symmetrical bot with more than 4 legs. The inclusion of the duty cycle as a key parameter ensures modularity and adaptability. For example, for walking on 5 legs (Figure **16**), the headings will be modified as in the algorithm and subsequently in the program. The duty cycle now will get modified accordingly. For tripod gait (three legs on the ground at any time), the duty cycle calculated similarly. Next in our simulation, the speed in phase, and speed out of phase are calculated, and the inverse kinematics code is executed after updating the leg tip positions exactly as mentioned in the previous section. This capability comes from the fact that the code treats each leg as an independent entity, and the motion of each leg is independent of the others. This allows the code to run for any number of legs. Changing only the design parameters in the code like joint lengths and step length allow it to be modified to any size of robot. Further, it allows the robot to have a

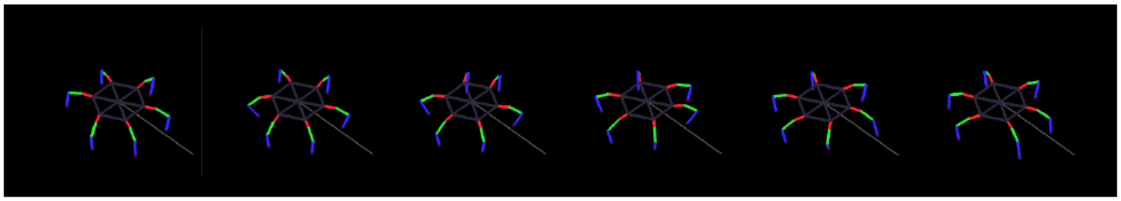

Figure 13: Wave gait simulation, isometric view.

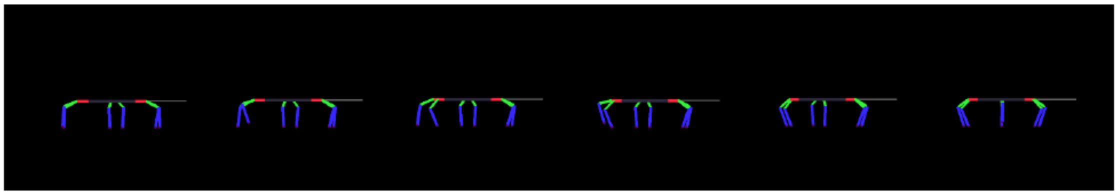

Figure 14: Wave Gait simulation, side view (1).

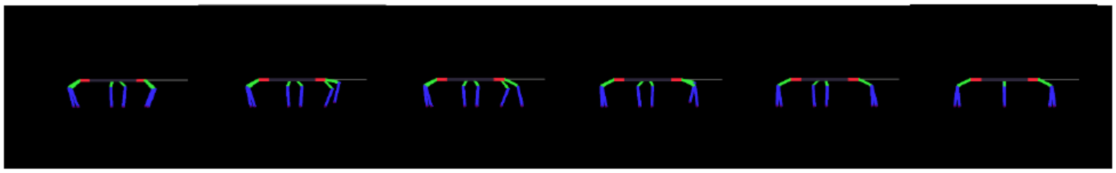

Figure 15: Wave Gait simulation, side view (2).

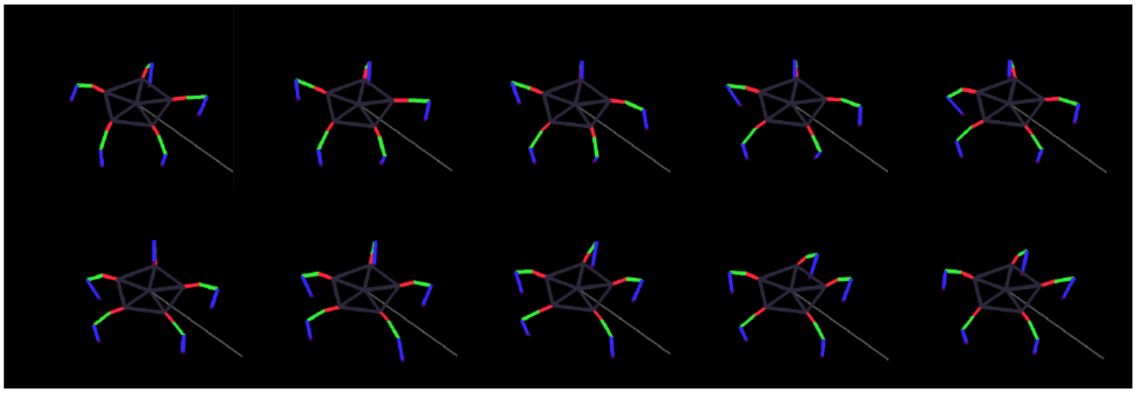

Figure 16: Motion with 5 legs.

degree of adaptability. In case of a leg failure in a hexapod, the remaining legs can be reoriented to new headings, compensating for the gap created by the missing leg. This can be done entirely within the code without any physical modification or correction, by creating angle offsets for the coxa motors. Now, running the same code with 5 legs, and the adjusted positions of the legs as their home position, a six-legged robot can be run on 5 legs as it is shown in Figure **16**. Similarly, the code can work for other leg orientations apart from radial. For a sagittal motion the headings will be changed to [-90, -90, -90, 90, 90, 90]. No change is needed to other levels in terms of code.

3.5. Validation of Enhanced Capabilities

The basic algorithm is sufficient to get a legged robot set up for walking with minimal calibration. This allows for a steep learning curve and lower set-up times when using this control structure as a basis for design. The second key feature distinguishing it is the ability to add control code over the base. As a proof of concept, we created a feedback system to detect foot collision with the ground, adding it over the base algorithm. Legs not in phase are moved up and forward in an arc, while a passive sensor waits for collisions between foot tip and ground. If a collision signal is detected, the system rewrites the height values for the particular leg, so that now, the leg is able to move at the new height. In the algorithm, the vertical motion during the out-of-phase movement was exaggerated. The return to the ground was changed from deterministic to incremental and as soon as collision is detected, motion in the vertical axis stops. A limit was defined beyond which the leg wouldn’t descend further, and would back off. Based on the use case, an exception handling function can be created to react to such input in a different manner. The leg contact sensor was demonstrated to work, having a reaction tolerance of 5mm in the vertical direction. Legs were made to work at different heights using feedback from this sensor to capture steady images, shown in Figure **17**.

This simple addition to the code now enables the bot to move over obstacles while maintaining a constant body height. In addition to maintaining constant body height, one more feature was tested as a possible addition: a camera module which took a picture once every 5 seconds, and ran the image

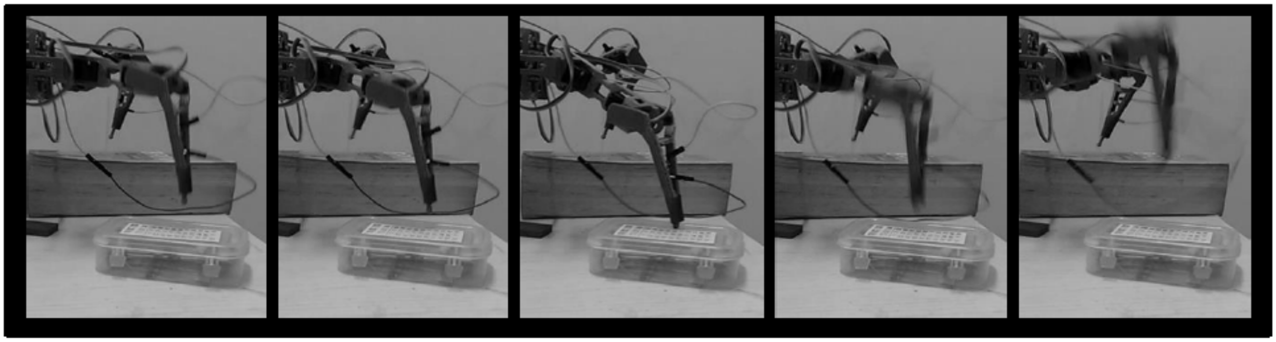

Figure 17: Contact sensor reaction test.

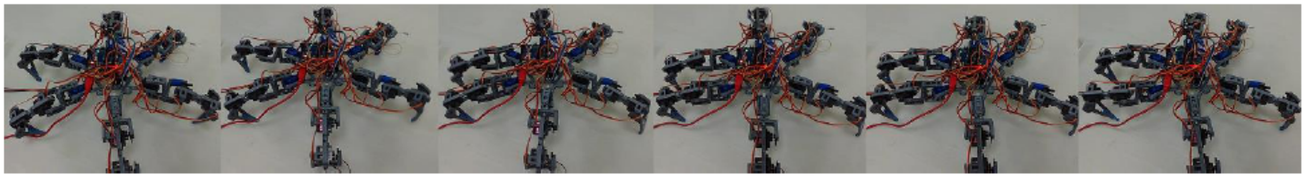

Figure 18: Bipod gait on prototype.

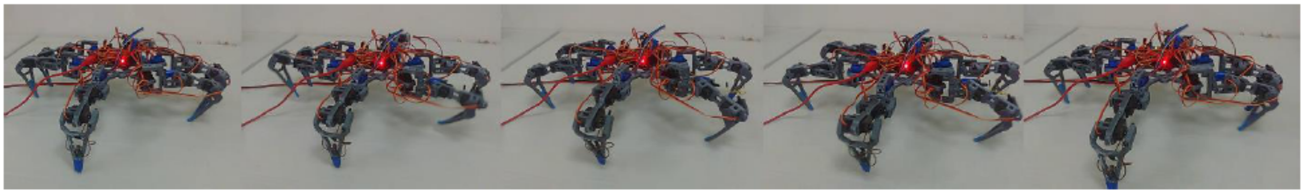

Figure 19: Tripod gait on prototype.

through a pre-trained CNN to identify the type of terrain. Based on the classification, the bot could change its gait to meet stability criteria, adding a layer of autonomy to the design.

Our developed prototype was able to adopt strategic gait according to the command and feedback. Figure **18** and **19** show the bot moving with the standard biopod and tripod gaits as expected from our simulation based on the variation of the duty cycles. These are some representative snapshots to showcase the different locomotion and working model of our hexapod. All the simulated gaits were validated experimentally with our simulated results which shows accrurancy and applicability of our methodology. The algorithm was tested on the prototype and was found to work at various speeds and gaits. The relation between the variable `speedvar` and ground speed was plotted for different standard gaits in Figure **20**. This shows a clear linear trend for speed at each particular gait and a distinct proportional speed difference with a change in gait, as expected. An average maximum ground speed of 3.52 cm/s was recorded over a distance of 60cm for tripod gait at 20 `speedvar`.

Table 2: Ground Speed at Different Speedvar and Gaits

	Speed (cm/s)		
SpeedVar	**Tripod**	**Bipod**	**Wave**
50	1.304348	0.983607	0.47619
30	2.5	1.764706	0.833333
20	3.529412	2.307692	1.111111

The average speed over many tests at bipod gait was observed to nearly double of the speed in wave gait, and the speed in tripod gait triple of the same as shown in Table **2**. This agrees with theoretical predictions.

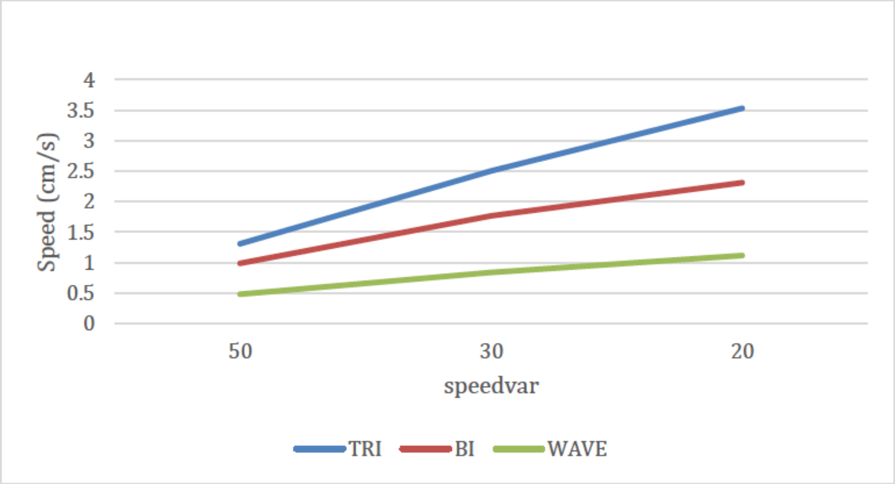

Figure 20: Variation of ground speed with speedvar for different gaits.

4. CONCLUSION

A versatile multi-legged framework has been developed with adaptive gait and fault tolerant capabilities. The control algorithm developed in this study provides a potential baseline towards developing intelligent systems in legged robotics domain. This novel framework is useful for working with high-level applications in robotics that rely on accuracy and stability. The simulation provides a way to quickly test design changes, reach, and orientations. Also, when the developed hexapod was tested with an onboard camera and tactile sensors on each foot, this algorithm maintained constant hip height in any type of terrain, making the robot most suitable for applications where a stable image output is required. The validation of the algorithm has been carried out by assessing its capability to incorporate changes in design and modular systems, giving great potential for any robot designed strategically through prototype development. The algorithm provides the much-needed flexibility and modular capability required in designing legged robots in terms of a learning tool or a logical base. It is expected that future implementations of this system will allow for innovation and ingenuity, backed by logical robustness, which will give practical value useful to the robotics community.

REFERENCES

[1] Gurfinkel V, Gurfinkel E Devjanin E, Efremov E, Zhicharev D, Lensky A, Schneider A, Shtilman L. Investigation of robotics. In Six-legged Walking Model of Vehicle with Supervisory Control; Nauka Press: Moscow, Russia, 1982; pp. 98-147.

[2] Raibert M. Legged Robots that Balance; MIT Press: Cambridge, London, 1986; pp. 180-201.

[3] Gregorio P, Ahmadi M, Buehler M. Design, control, and energetics of an electrically actuated legged robot. Syst. Man Cybern. B IEEE Trans 1997; 27: 626-634. https://doi.org/10.1109/3477.604106

[4] Pfeiffer F, Eltze J and Weidemann HJ. Six-legged technical walking considering biological principles. Robotics and Autonomous Systems 1995; 14(2-3): 223-232. https://doi.org/10.1016/0921-8890(94)00031-V

[5] Saranli U, Buehler M and Koditschek DE, RHex: A Simple and Highly Mobile Hexapod Robot. The International Journal of Robotics Research 2001; 20(7): 616-631. https://doi.org/10.1177/02783640122067570

[6] Halvorsen K (2014) Zenta Robotic Creations. Technical report published 30th June 2014, Wordpress 2015.

[7] Tedeschi F and Carbone G. Design Issues for Hexapod Walking Robots. Robotics 2014; 3(2): 181-206. https://doi.org/10.3390/robotics3020181

[8] Zak M and Rozman J. Design, construction and control of hexapod walking robot. IEEE 13th International Scientific Conference on Informatics 2015. https://doi.org/10.1109/Informatics.2015.7377851

[9] Chin KW, Konsky BV and Marriott A (n.d.) 2002, Closed-form and generalized inverse kinematics solutions for the analysis of human motion. Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. Magnificent Milestones and Emerging Opportunities in Medical Engineering (Cat. No.97CH36136).

[10] Yang J. Omnidirectional walking of legged robots with a failed leg. Mathematical and Computer Modelling 2008; 47(11-12): 1372-1388. https://doi.org/10.1016/j.mcm.2007.08.006

[11] Guoliang Z, Long C, Hua D. A Performance Oriented Novel Design of Hexapod Robots. IEEE/ASME Transaction on Mechatronics, 2017; 22(3): 1435-1443. https://doi.org/10.1109/TMECH.2017.2681722

[12] Soyguder S. and Alli H. Design and Prototype of a Six-Legged walking insect robot. Industrial Robot: An International Journal 2007; 34(5): 412-422 https://doi.org/10.1108/01439910710774412

[13] Niwa T, Inagaki S and Suzuki T 2009, Locomotion control of multi-legged robot based on Follow-the-Point-Contact gait. In: ICROS-SICE International Joint Conference, Fukuoka International Congress Centre, Japan, 18-21 August 2009, pp. 2247-2253.

[14] Mostafa G, Amir Z. Inverse Kinematic Analysis of a Hexapod Spider-like Mobile Robot, Advanced Marterials Research, 2012; 403-408, pp 5061-5067 https://doi.org/10.4028/www.scientific.net/AMR.403-408.5061

[15] Quingsheng L, Hui Z, Baoling H, Xiaochuan Z. "research on Biologically inspired Hexapod Robot's Gait and Path Planning," IEEE International Conference on Information and Automation 2009; pp. 1546-1550.

Received on 29-09-2021 Accepted on 25-10-2021 Published on 30-12-2021

DOI: https://doi.org/10.31875/2409-9694.2021.08.5