applied sciences

MDPI

Article

Valkyrie—Design and Development of Gaits for Quadruped Robot Using Particle Swarm Optimization

Taarun Srinivas [1], Adithya Krishna Karigiri Madhusudhan [1], Lokeshwaran Manohar [1], Nikhit Mathew Stephen Pushpagiri [1], Kuppan Chetty Ramanathan [1,*], Mukund Janardhanan [2] and Izabela Nielsen [3,*]

[1] Centre for Automation and Robotics, School of Mechanical Sciences, Hindustan Institute of Technology and Science, Chennai 603103, India; 17130012@student.hindustanuniv.ac.in (T.S.); 17130011@student.hindustanuniv.ac.in (A.K.K.M.); 17130001@student.hindustanuniv.ac.in (L.M.); 17130007@student.hindustanuniv.ac.in (N.M.S.P.)
[2] School of Engineering, University of Leicester, Leicester LE1 7RH, UK; mukund.janardhanan@leicester.ac.uk
[3] Department of Materials and Production, Aalborg University, 9220 Aalborg, Denmark
* Correspondence: kuppanc@hindustanuniv.ac.in (K.C.R.); izabela@mp.aau.dk (I.N.)

Abstract: Over the past decades, developments and scientific breakthroughs in the field of robotics have shown the replacement of wheeled robots with legged robots, which are often inspired by the biological characteristics of legged animals. Many industries and urban-based applications promote quadruped robots because of their dexterous ability to efficiently handle multiple tasks in the working environment. Motivated from the recent works in the field of quadruped robots, this research aims to develop and investigate gaits for a 2 DoF mammal-inspired quadruped robot that incorporates 4 hip and 4 knee servo motors as its locomotion element. Forward and inverse kinematic techniques are used to determine the joint angle required for the locomotion and stability calculation are presented to determine the center of mass/center of gravity of the robot. Three types of gaits such as walk, trot, and pace are developed while keeping the center of mass inside the support polygon using a closed-loop control system. To minimize errors and improve the performance of the robot due to its non-linearity, a meta-heuristic algorithm has been developed and addressed in this work. The fitness function is derived based on the Euclidean distance between the target and robot's current position and kinematic equations are used to obtain the relation between joints and coordinates. Based on the literature, particle swarm optimization (PSO) was found to be a promising algorithm for this problem and is developed using Python's 'Pyswarms' package. Experimental studies are carried out quantitatively to determine the convergence characteristics of the control algorithm and to investigate the distance traveled by the robot for different target positions and gaits. Comparison between experimental and theoretical results prove the efficiency of the proposed algorithm and stability of the robot during various gait movements.

Keywords: quadruped robot; particle swarm optimization; kinematics; gait development; support polygon

check for updates

Citation: Srinivas, T.; Madhusudhan, A.K.K.; Manohar, L.; Stephen Pushpagiri, N.M.; Ramanathan, K.C.; Janardhanan, M.; Nielsen, I. Valkyrie—Design and Development of Gaits for Quadruped Robot Using Particle Swarm Optimization. *Appl. Sci.* **2021**, *11*, 7458. https://doi.org/10.3390/app11167458

Academic Editors: Manuel Armada and Roemi Fernandez

Received: 16 July 2021
Accepted: 11 August 2021
Published: 13 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Recent developments in society show the use of robots for different kinds of applications. Robots have automated various sectors of society, reduced manpower, and assisted humans in places where they are restricted, or it is hazardous to work. A robot needs locomotion mechanisms to make it move through its environment, which can be accomplished by wheels, legs, etc. In recent years, the utilization of leg-based walking robots has turned out to be very efficient in the robotics field. These robots possess the ability to deal with irregular terrains, particularly when it's compared with wheeled systems [1–3]. Legged robot locomotion mechanisms, which are often inspired by biological systems, are very successful in moving through a wide area of rough and harsh environments. Legged

robots are classified based on the number of legs used, such as biped, quadruped, hexapod, and so on. Quadruped robots are preferred among the other legged robots because of their less complex structure, giving the fullest requirements for the statically stable gaits during walking [4]. Stability is a very important issue of a robot that is maintained as long as the center of mass (CoM) is within the support polygon which is a polygon with the vertices described by the position of legs in contact with the ground. Stability can be divided into static and dynamic stability criteria. Static stability is given when the CoM is within the support polygon and the polygon's area is greater than zero. Therefore, static stability requires at least three points of ground contact [1]. In general, it is extremely difficult to design gait controllers for legged robots. Various hand-crafted methods have been proposed to construct gait controllers, in which human supervision is needed in most cases [5]. Song and Waldron defined gait as something that is defined by the time and the location of the placing and lifting of each foot, coordinated with the motion of the body in its six degrees of freedom, to move the body from one place to another [6]. The observation, comprehension, and mathematical formulation of bio-inspired gaits were the main focus on legged locomotion, and it was found that many of these motions are periodic in nature. Gaits are generally classified as periodic and non-periodic, and it was found that the motion pattern of the quadruped robots is periodic [7].

Many researchers have developed mathematical formulations of various biological gaits of quadruped robots that ensure maintaining static stability. Estremera et al. [8] designed and implemented a mixture of free and discontinuous gaits to negotiate uneven terrain with a real machine. The fusion of these two main gaits plus the addition of extra constraints to avoid leg-transfer deadlocking produced a new free-crab gait, a free-spinning gait, and a free-turning gait. The developed gaits were tested on a SILO4 walking robot, and their algorithms were proved to be efficient in real-time and adaptive to irregular terrain. Wang et al. [9] adopted a statically stable crawling gait as their main gait and proposed a fuzzy multi-model switching control based on friction compensation to achieve smooth switching of force and position. Their method was found to offer a solution for stable passage in a slope environment by realizing smooth switching of force/position, precise positioning in the swing process, and soft control of force in the supporting phase. Xi W et al. [10] explored the benefits of using multiple gaits in a single robot and analyzed how increasing speed affects the choice of gait and how the choice of gait influences optimal speed. The authors used optimal control as a tool to identify motions that minimize the cost of transport of two detailed models: a planar biped and a planar quadruped. It was found that changing gaits as speed varies leads to the greatly increased energetic economy and concluded that for quadrupeds, the optimal gaits were four-beat walking at low speeds, trotting at intermediate speeds, and galloping at high speeds.

Due to the non-linearity of the system, rather than using a conventional approach, this research focuses on developing a metaheuristic algorithm to achieve various gaits. Metaheuristic algorithms are computational intelligence paradigms used especially for the sophisticated solving of optimization problems [11]. This optimization is achieved by either minimizing or maximizing the solution through computing the fitness function. Winkler et al. [12] presented an algorithm that generates walking motions for quadruped robots by simultaneously optimizing over both the center of mass (CoM) trajectory and the footholds without the use of an explicit footstep planner. Given a desired goal state, the problem was solved using a non-linear programming solver, and their results proved that the algorithm was able to generate walking gaits for multiple steps in milliseconds. Focchi et al. [13] proposed a heuristic-based planning approach that enables a quadruped robot to successfully traverse a significantly rough environment in the absence of visual feedback. Their proposed framework included reflexes triggered in specific situations, and the possibility to estimate online an unknown time-varying disturbance and compensate for it.

Raheem et al. [4] used a particle swarm optimization algorithm to find the best value of the stability margin during the robot walking gait. The quadruped robot kinematic

model of the forward and inverse kinematics for each 3-DOF was calculated, which led to finding the minimum stability margins during walking on the vertical geometrical projection of the robot body. Their results demonstrated the best stability margins that guaranteed the preservation of robot COG within the support polygon. Similarly, Golubovic et al. [5] presented an evolutionary algorithm for Sony-legged robots to learn good walking behaviors with little or no interaction with the designers. The algorithm was based on a hybrid approach that changes the probability of genetic operators concerning the performance of the operator's offspring. The resulting gait was proved to be a better solution than the non-interference training for movements over all types of surfaces. Unlike other evolutionary algorithms like genetic algorithm (GA), and ant colony optimization (ACO), PSO achieved global convergence in a stipulated period of time. The former results in overlapping while the latter does not provide guaranteed convergence as the number of iterations increases.

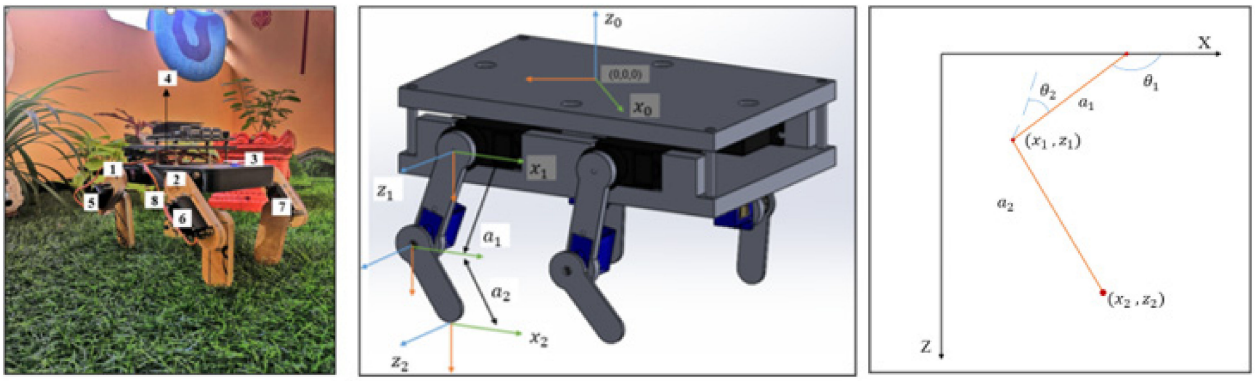
The main goal of this paper is to develop robust gaits for the quadruped robot, and since the inverse kinematic equation can yield many solutions, it is necessary to determine an optimal value of joint angles to achieve efficient locomotion, which can be done by the use of metaheuristic algorithms. Thus, from literature studies and since PSO is simple, computationally efficient, has fewer parameters, and guarantees stable convergence [14,15], we have narrowed down our focus to develop a PSO algorithm to achieve various gaits.

Hence, this work primarily focuses on the development and investigation of gaits for a 2 DoF mammal-inspired quadruped robot incorporating 4 hip and 4 knee motors as its locomotion element. Forward and inverse kinematics were formulated to determine the joint angles and stability analysis was carried out to ensure that the CoM and CoG lay within the support polygon. Walk, trot, and Pace gaits were developed using the particle swarm optimization algorithm to minimize the errors and improve the performance of the robot due to its non-linearity. The fitness function was derived based on the Euclidean distance between the target and robot's current position, and kinematic equations were used to obtain the relation between joints and coordinates. Experimental studies were carried out quantitatively to determine the convergence characteristics of the control algorithm and to investigate the distance traveled by the robot for different target positions and gaits.

The rest of this paper is organized as follows. Section 2 summarizes the mechanical and kinematic modeling of the quadruped robot. Section 3 explains the stability calculations carried out to determine the CoM and CoG. Section 4 describes the development of gaits using the PSO algorithm. Section 5 discusses the experimental results obtained and benchmarks with the theoretical values. Finally, the outcomes of this research are concluded in Section 6.

2. Kinematic Modeling of the Quadruped Robot

A 2 DoF mammal-inspired Quadruped robot was designed and developed as seen in Figure 1a to implement the gaits. Since, the robot was developed to wander in a flat terrain, the legs of the robot are symmetric in nature. Hence, the kinematic modeling of each pair of parallel legs is similar to the kinematic modeling of a bipedal robot where the frame of the robot remains constant [16]. Thus, it is enough to perform the kinematic modeling for the legs of the robot. The parameters of the robot are tabulated in Table 1.



(**a**) (**b**) (**c**)

Figure 1. (**a**) Robot prototype; (**b**) frame assignment; (**c**) kinematic model of a leg.

Table 1. Robot Parameters.

Parameters	Specifications
Robot body dimension	150 mm × 90 mm × 30 mm
Link length (a_1 and a_2)	70 mm
DoF	2/Leg
Weight	1.5 kg
Material	Polyethylene terephthalate (PET) and acrylic
Joint constraints	$0 \leq \theta_1 \leq +20$ $0 \leq \theta_2 \leq +20$

2.1. Forward Kinematics

Forward Kinematics is widely used in robotics to determine the position of the end effector by providing inputs like link lengths and joint angles. The joint variables are θ_1 which is the angle between link 1 and the body of the robot and θ_2 is the angle between link 1 and link 2. a_1 is the length of link 1 and a_2 is the length of link 2. The Z-axis is assigned along the axis of rotation, and the X-axis is placed perpendicular to Z-axis as seen in Figure 1b. The Denavit–Hartenberg method was used to determine the forward kinematics equations. The homogeneous transformation matrix, A_i can be written as

$$A_i = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Using the parameters in the DH Table (Table 2), we could get the transformation matrix as,

$$A_i = A_1 \cdot A_2 = \begin{bmatrix} c_1 & -s_1 & 0 & a_1 c_1 \\ s_1 & c_1 & 0 & a_1 s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_{12} & -s_{12} & 0 & a_1 c_1 + a_2 c_{12} \\ s_{12} & c_{12} & 0 & a_1 s_1 + a_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Table 2. DH parameters Table.

Link	θ	*d*	*a*	*A*
1	θ_1^*	0	a_1	0
2	θ_2^*	0	a_2	0

From the A_i matrix, the position, and the orientation can be determined easily. The position is given as

$$x_2 = a_1 cos_1 + a_2 cos_{12},\ z_2 = a_1 sin_1 + a_2 sin_{12} \tag{3}$$

2.2. Inverse Kinematics

In Inverse Kinematics, the joint angle is determined given the end-effector position and link lengths. Inverse kinematics may not always have a unique solution. Figure 1c represents the kinematic model for one leg of the quadruped robot where x_2 and z_2 represent the end-effector position. a_1 and a_2 are the length of link 1 and link 2, respectively. θ_1 and θ_2 represent the joint angles of link 1 and link 2, respectively. The joint angles θ_1 and θ_2 are determined using the equations.

$$\theta_1 = tan^{-1}\left(\frac{z_2}{x_2}\right) + tan^{-1}\left(\frac{a_2 sin\theta_2}{a_1 + a_2 cos\theta_2}\right) \tag{4}$$

$$\theta_2 = cos^{-1}\left(\frac{{x_2}^2 + {z_2}^2 - a_1^2 - a_2^2}{2a_1 a_2}\right) \tag{5}$$

3. Stability Analysis of the Quadruped Robot

A robot is said to be stable if it regains its original position after external forces are exerted on it. To understand the movements and dynamism of quadruped robots, static stability and dynamic stability analysis are of great importance. Static stability is defined as the stability of a robot at every instant of time with the absence of motion. Static stability is given to a robot when the center of mass is completely within the support polygon (Figure 2), which is a polygon with the vertices described by the position of legs in contact with the ground and whose area must be greater than zero. To determine whether the quadruped robot is statically stable, the following conditions must be achieved.

1. The moment of stability should be a minimum of the sum of the moment of all four legs.
2. The center of gravity (CoG)/center of mass (CoM) should be within the area of support polygon.

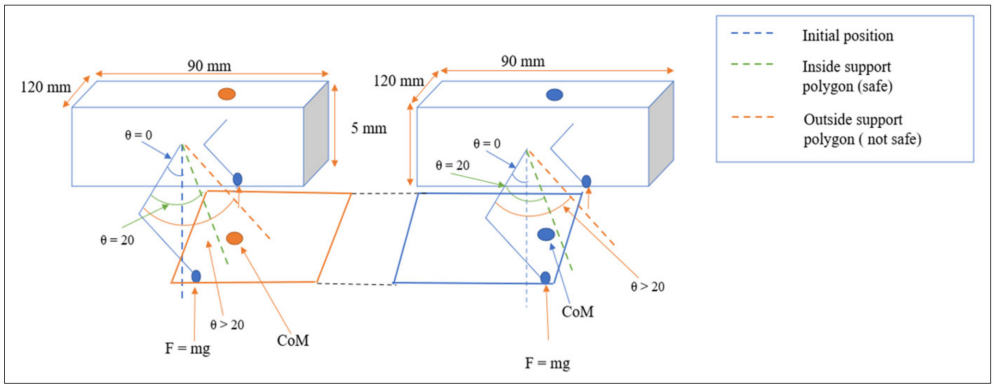


Figure 2. Support polygon of the quadruped robot.

3.1. Moment of Stability (M_{stab})

The moment is defined as the product of force acting on each link or each leg and the distance between the link or leg to the ground and is calculated as,

$$moment = force \times distance \tag{6}$$

The conditions to achieve M_{stab} are,

1. $M_{stab} > 0$
2. $M_{stab} = Min[M_{L1}\ M_{L2}\ M_{L3}\ M_{L4}]$

where, M_{stab} is the moment at which stability is achieved; $M_{L1}, M_{L2,} M_{L3}, M_{L4}$ are the moments of each leg.

$$\text{Moment acting at a leg is calculated as, } M_{L1} = M_{l1} + M_{l2} \tag{7}$$

where, M_{l1} and M_{l2} are the individual moments of the two links that form a single leg. As torque values are known, force is calculated using the below equation;

$$force = \frac{torque}{length + \sin(angle)} = \frac{11}{7 + sin(20)} = 1.497 \text{ kg} \tag{8}$$

The moment for link 1(M_{l1}) = $force\ x\ distance$ = 8.832 kg − cm. In the same manner, the moment for link 2 (M_{l2}) is found to be 14.0718 kg − cm. Thus, the total moment acting on the leg (M_{L1}) is 22.90 kg − cm. Since the legs are symmetric, the moments for the other legs will also be within the range of the moment of the calculated leg. The conditions for M_{stab} are satisfied as to the value of M_{stab} is greater than zero and the same moment at stability is the minimum of the moment of the other legs.

$$M_{stab} = Min[22.90\ M_{L2}\ M_{L3}\ M_{L4}] \text{ kg} - \text{cm} \tag{9}$$

Therefore, the moment at which stability is achieved should have a value that is greater than or equal to 23 kg − cm.

3.2. Center of Gravity (CoG)

Center of gravity (CoG) is defined as the point at which the force of gravity of the overall body acts. The center of gravity must lie within the area of support polygon and is calculated as below.

$$CoG = \frac{\sum Moment}{\sum weight} \tag{10}$$

where, $\sum Moment$ is the sum of the moment acting on all the legs; $\sum weight$ is the sum of all the mass of the motors. As $\sum Moment$ and $\sum weight$ are known, the CoG was found to be 11.5 cm. So, the center of gravity is at a distance of 11.5 cm from the reference point (0,0,0), which was taken concerning the frame of the robot.

3.3. Center of Mass (CoM)

Center of mass is defined as the point at which the resultant mass of the body of the robot acts and is mathematically defined as,

$$CoM = \frac{\sum_{i=1}^{n} M_i x_i}{\sum_{i=1}^{n} M_i}, \frac{\sum_{i=1}^{n} M_i y_i}{\sum_{i=1}^{n} M_i} \tag{11}$$

Determining the distance of the motors from the reference point and knowing the mass of the motors, the CoM along x and y axes was found to be (8.8, 7.64). Thus, CoM is located at a distance of 11.65 cm from the reference point. Since the area of support polygon is in the shape of a parallelogram, the area was found to be 161.5 cm^2. As the CoG and CoM are within the area of support polygon and M_{stab} conditions are satisfied, the robot is concluded to be statically stable.

4. Gait Development Using PSO

Gait development is the selection and formulation of a coordinated sequence of legs and body motions that propels the robot along a desired path [17]. During locomotion, each leg is swung forward and then retracted to contact the ground. Quadrupeds commonly use two types of gaits, i.e., symmetrical gaits (e.g., pace, walk, or trot) where the footfalls of

the pair of are evenly spaced in time and asymmetrical gaits (e.g., gallop, bound) where the actions in at least one pair are unevenly spaced in time [18]. This paper focuses on the development of the three common symmetrical gaits, i.e., walk, trot and pace, for the quadruped robot using particle swarm optimization algorithm and kinematic equations. The gaits are considered according to speed and efficiency since a real-time reasonable selection of walking pattern is necessary to maintain the best walking balance.

A. Walk Gait Implementation

The walk is the slowest and most energy-efficient gait [19]. It is a periodic continuous four-beat gait, in which each leg steps sequentially. The walk step sequence is shown in Figure 3a. In the normal walk gait, two legs alternate with three legs in supporting body weight. Several types of walk gaits are recognized based on variations in leg support, stride length, and step speed.

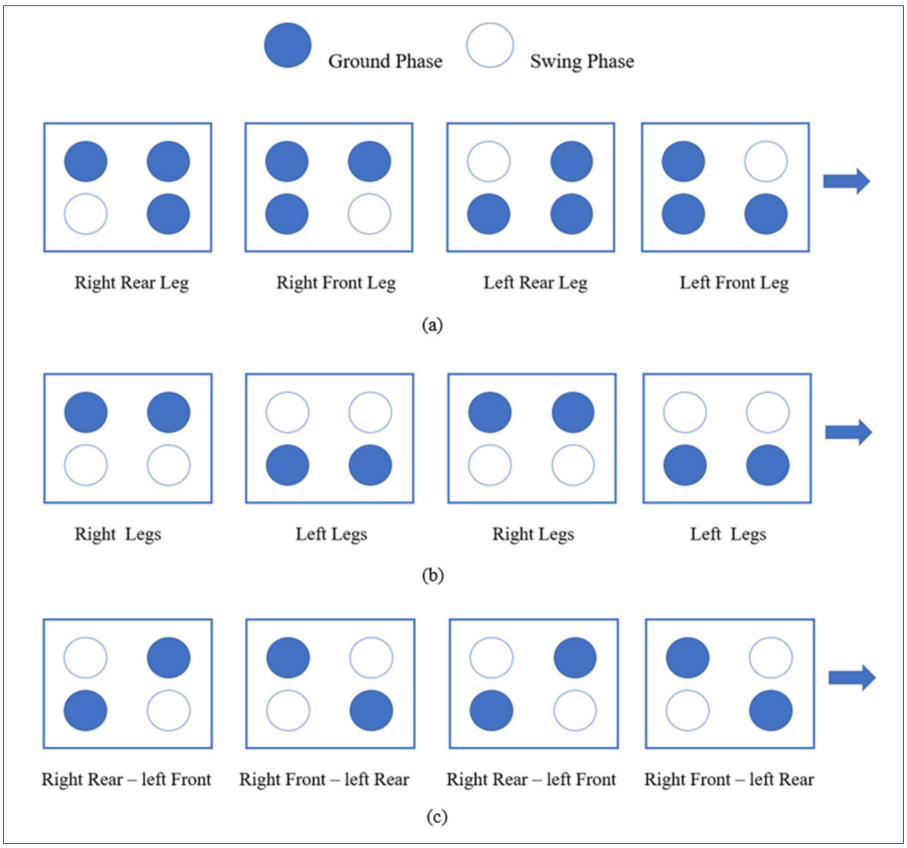


Figure 3. (**a**) Walk gait sequence; (**b**) pace gait sequence; (**c**) trot gait sequence.

B. Pace Gait Implementation

The pace is a two-beat gait. The two lateral limbs are used alternately for weight support, i.e., the left forelimb and left hindlimb move in unison, as do both right limbs. The foreleg may cycle slightly earlier than the ipsilateral hindleg. The step sequence of the trot is shown in Figure 3b. A short period of suspension typically intervenes between the alternating phases of lateral support, particularly at increased speed. The rack gait is a rapid pace in which the two ipsilateral limbs land separately instead of together, producing a four-beat gait. The pace gait can be used with some range of speed, and naturally as the speed increases, the moment of suspension between beats will increase as well [19].

C. Trot Gait Implementation

The trot is a two-beat gait. Right and left diagonals alternate in supporting weight, i.e., the right foreleg and left hind leg move in unison, as do the left foreleg and right

hind leg. The step sequence of the trot is shown in Figure 3c. Twice during each step, the body is ballistic and without help or support from any other body. In trot gait, the pattern of movement is quick compared to other gaits and is one of the most stable gaits but improving its dynamic characteristics is a challenging task. The trot is a common gait in all domestic quadrupeds. It is well-suited for rough, irregular ground and for traveling long distances at a fair rate of speed. Work is spread evenly over all four limbs, and diagonal support makes it easy to maintain equilibrium [19].

4.1. Particle Swarm Optimization (PSO)

In a PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution candidate, called a particle, flies in the problem search space looking for the optimal position to land [20]. Each particle in PSO has two important components, (i) Position $(X(t))$ and (ii) Velocity $(V(t))$.

As time passes, every particle adjusts its position and velocity according to its own experience (personal best or P_{best}) as well as according to the experience of neighbouring particles (global best or G_{best}). The equations for updating the velocity and position of the particle are as follows [21].

$$V(t+1) = \omega \times (V(t) + C_1 \times rand_1 \times (P_{best} - X(t)) + C_2 \times rand_2 \times (G_{best} - X(t))) \quad (12)$$

$$X(t+1) = X(t) + V(t+1) \quad (13)$$

C_1 and C_2 are the acceleration co-efficient that controls the maximum step size the particles can do and 'ω' is the inertial weight. Together they control the impact of previous historical values of particle velocities on its current one and $rand_1$ and $rand_2$ are two random functions with a range [0, 1].

Thus, Equation (12) is used to calculate the particle's new velocity according to its previous velocity and the distances of its current position from both its own best historical position and its neighbor's best position. Then, the particle flies toward a new position according to Equation (13). The performance of each particle is measured according to a pre-defined fitness function, and this process is repeated until the user-defined stopping criterion is satisfied.

4.2. Mathematical Formulation of Fitness Function

The Fitness Function is planned to be formulated in such a way that an optimized inverse kinematic solution will be obtained considering the following constraints.

1. *Center of mass (CoM) and center of gravity (CoG)*—For the robot to stay stable during static and dynamic conditions, the CoM and CoG of the robot must stay within the support polygon.
2. *Maximum angle of links of the robot*—It must be ensured that the legs do not cross the maximum angle so that the CoM and CoG of the robot lie within the polygon.

Assume the initial position to be X_0 (x_0, y_0) and target position to be X_t (x_t, y_t). A PSO particle is given by $Q_i = \{\theta_1, \theta_2\}$, Where Q_i is a set of possible joint angles subjected to the following constraints.

$$\theta_1 \in [\theta_{1min}, \theta_{1max}] \text{ and } \theta_2 \in [\theta_{2min}, \theta_{2max}] \quad (14)$$

After applying the joint angles in the forward kinematic equation, the robot position in a reference frame can be obtained as, X_i $(x_i, y_i) = (X_1, X_2 \ldots\ldots X_n)$ This position is compared with the target position, X_t. Hence, the fitness function, f is given as the Euclidean distance between the target position and the obtained robot position.

$$f = ||X_t - X_i|| = \sqrt{(x_t - x_i)^2 + (y_t - y_i)^2} \quad (15)$$

4.3. PSO Algorithm Implementation and its Significance

Inverse kinematics is a key issue in robotics, especially for problems such as path planning, motion generation, or trajectories optimization. It becomes very tedious to calculate with many degrees of freedom, and also, the results are not optimal. Since the focus of this paper is to develop robust gaits for the quadruped robot and, it is necessary to determine an optimal value of joint angles to achieve efficient locomotion. Due to this non-linearity of the system, rather than using a conventional approach, PSO is a very efficient method. As all the legs are symmetric to each other, it is sufficient to run the algorithm for one leg and utilize the results obtained for the other legs. Thus, through the PSO algorithm, we are accelerating each particle, i.e., joint variables towards the P_{best} and G_{best} through repetitive iterations to obtain the optimal joint variables. The pseudo code of the developed PSO algorithms is given as Algorithm 1 below.

Algorithm 1 PSO Algorithm Implementation

Inputs: t_{max} (Maximum Iterations), S (Swarm Size), D (Dimension of particle), C_1 and C_2 (Learning rates), ω (Inertial Weight), $[\theta_{1min},\ \theta_{2min}]$ (Lower bounds), $[\theta_{1max},\ \theta_{2max}]$ (Upper bounds)
Outputs: G_{best} (Global / Best Cost)

```
Randomly generate the Initial Swarm S within the range of 0 to 20 for each particle of dimension D
Initialize the Swarm Parameters
Check whether the CoM/CoG lies within the support polygon for the generated particles
Calculate the current position using Equation (3)
Determine the Fitness value using Equation (15)
Evaluate the fitness of the swarm and obtain the P_best and G_best
Update the velocity and position of particles using Equations (12) and (13)
while t ≤ t_max do
    Evaluate the fitness of new particles generated.
    for each particle in the swarm
        if current fitness < P_best, assign P_best = current fitness
        end if
    end for
    Repeat steps 2, 3, 4 and 5
    Increase the iteration number t by 1
end while
```

5. Experimental Results and Discussions

Determining the joint angles in an efficient way during dynamic locomotion is a challenging task. As the robot traverses through the terrains, the joint angle limits are not going to be the same. Each leg will require a desired joint angle depending on various constraints such as the frame of the robot, stability, and the compensation of the angle on each leg due to angle requirement on other legs of the robot. The novelty of this paper lies in the development of a PSO algorithm considering kinematic solutions to achieve an optimal joint angle in an efficient manner. The algorithm is implemented in Python programming language using Visual Studio Code/Python Editor through Linux OS (Ubuntu Version 18.04).

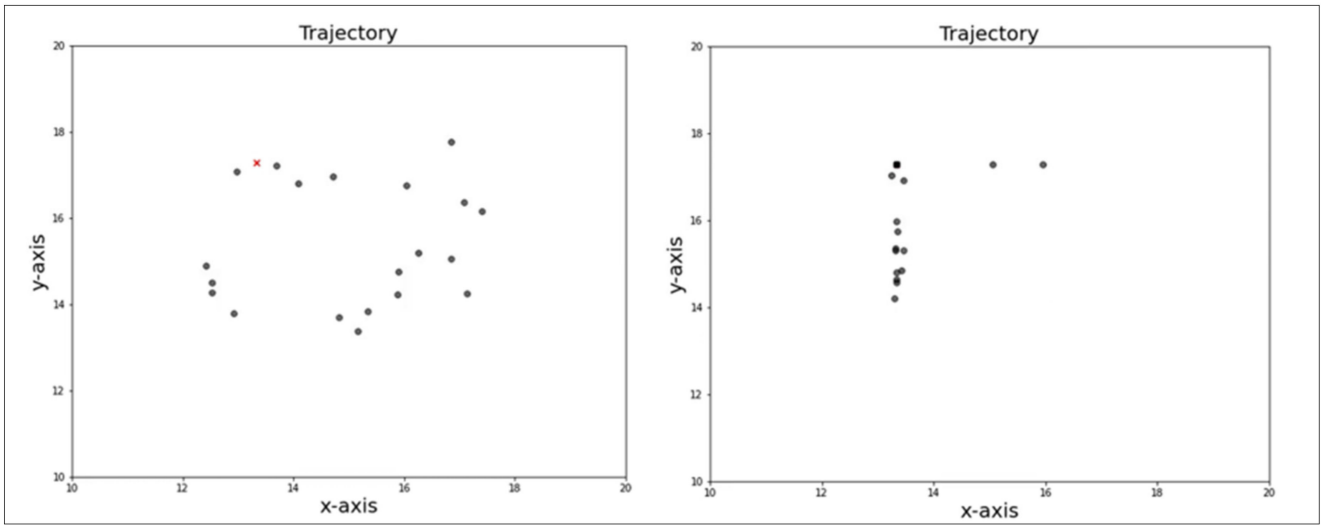
Table 3 gives the proposed PSO algorithms parameters. Since each leg consists of 2 DoF, the dimension of each particle is 2. The acceleration coefficients (C_1, C_2) define the magnitude of the influence on the particles velocity in the directions of the local and the global optima. To have a better balance of search space between local and global exploration, it is good to assign a value between 1.0 and 2.0. Similarly, inertia weight

(ω), plays a major role in determining the proportional relationship between local and global convergence. Since a larger inertia weight tends to facilitate global exploration, whereas a smaller inertia weight facilitates local exploration, the value is set to be 0.5. Using Pyswarms's built-in optimizer function, the objective function (Equation (15)) is optimized, and the constraints are passed as its arguments.

Table 3. PSO parameters.

Parameter	Value
No. of particles (or) swarm size	20
Dimension of a particle	2 ($[\theta_1, \theta_2]$)
Max iterations	1000
Learning rates (C_1, C_2)	1.5
Inertia weight ω	0.5

The algorithm is made to run for 1000 iterations and the best cost and the optimal joint variables to reach the target position of (10,0,0) are obtained as 1.14×10^{-14} and [13.34, 17.29], respectively. Figure 4a,b shows the movement of the swarm towards the global best (marked in the red cross) where the x and y-axis correspond to the joint variables θ_1 and θ_2 in degrees, respectively, and Figure 5 shows the experimental setup. In addition, a graph between the number of iterations and cost is shown in Figure 6, where the best cost was found to decrease.



(**a**) (**b**)

Figure 4. (**a**) Initial location of the swarm; (**b**) Final location of swarm after convergence (https://drive.google.com/file/d/1c50Om2ocqiqzBCHhs6DjpV4Vy83q07HK/view, accessed on 10 August 2021).

To understand the convergence rate of the algorithm, the distance the robot travels, and computational time, the code is made to run five times for different target positions and be implemented in the robot. The results are tabulated in Table 4, where the target is given as (10,0,0) and (20,0,0), which describes the movement of the robot on the x-axis for a distance of 10 mm and 20 mm. Thus, it is understood that the algorithm was able to achieve the best/global cost of 1.14×10^{-14} for the target value of (10,0,0) and 1.00 for (20,0,0), respectively. This ensures that the algorithm was able to achieve global convergence.

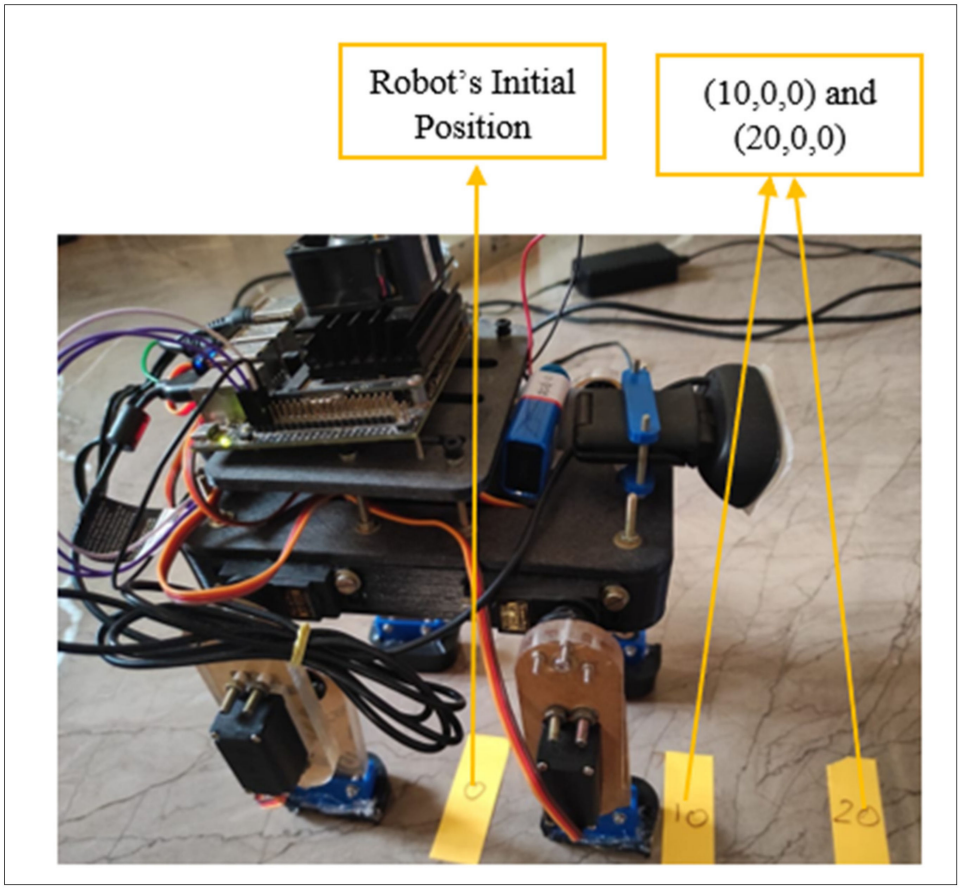


Figure 5. Experimental setup.

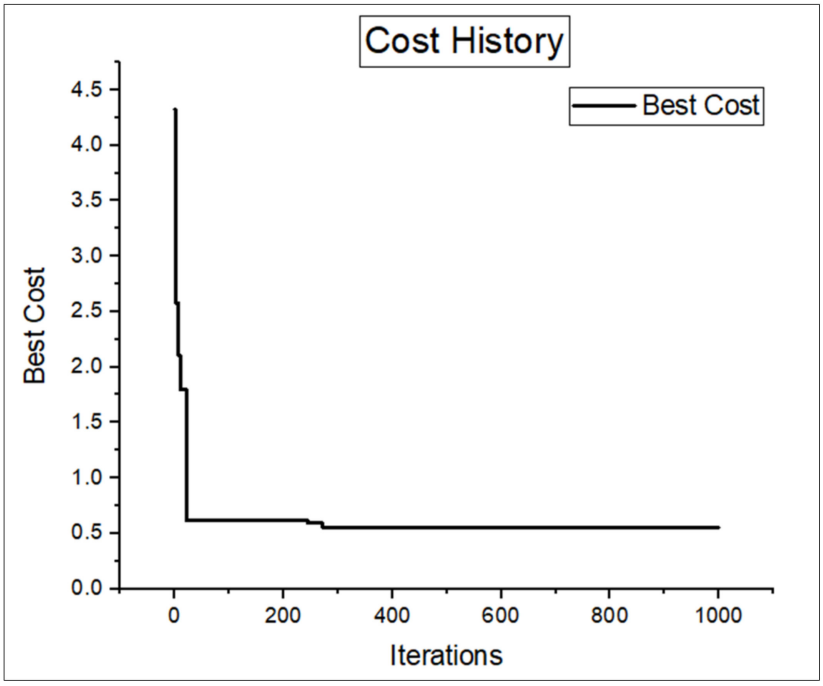


Figure 6. Iteration vs. cost.

Table 4. Convergence investigation.

Target (mm)	Best Cost	Joint Variables (Degrees)	Distance (mm)	Runtime (Seconds)
(10,0,0)	1.14×10^{-14}	[13.34156399 17.29916917]	10	1.81
	1.14×10^{-14}	[13.34156399 17.29916917]	10	1.85
	1.14×10^{-14}	[13.34156399 17.29916917]	10	1.79
	1×10^{-14}	[19.62354122 17.29984651]	15	1.84
	1.14×10^{-14}	[13.34156399 17.29916917]	10	1.75
(20,0,0)	1.00	[18.84955593 18.84955591]	20	1.80
	1.00	[18.84955593 18.84955591]	19.5	1.81
	1.00	[18.84955594 18.84955589]	19	1.79
	1.00	[18.84955593 18.84955591]	20	1.79
	1.00	[18.84955594 18.84955589]	20	1.81

It is seen that the run time of the algorithm is around 1.80 s which proves the algorithm to be computationally efficient. In addition, it is understood that for a target value of 10, the robot moves 10 mm forward in a single step, and similarly, for a target value of 20, the robot moves 20 mm in a single step. Hence, an investigation between the theoretical and experimental values has been carried out and presented in Table 5 for the three gaits considered. The error has been calculated as given in Equation (16).

$$\text{Error } \% = \frac{d_T - d_E}{d_T} \times 100 \tag{16}$$

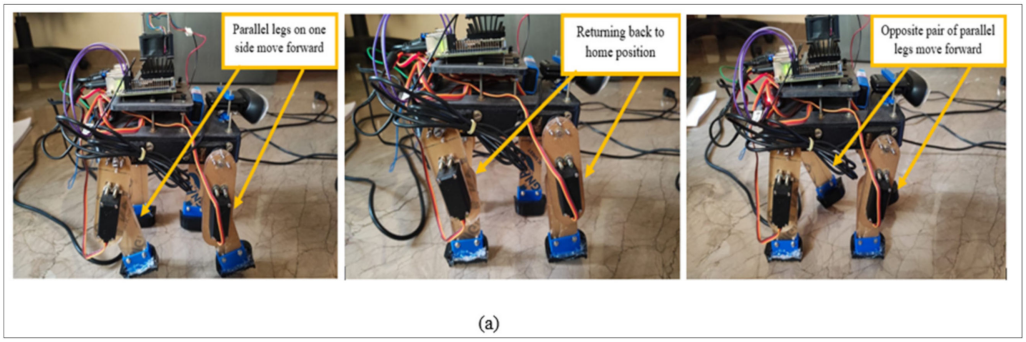
where d_T and d_E are the distance traveled by the robot estimated in theoretical and experimental, respectively. For this investigation, the robot was made to travel a distance of 10 cm, 20 cm, 100 cm, and 200 cm as tabulated in Table 5, and the actual distance traveled by the robot is noted. It can be found that the robot was not able to exactly reach the target position. This error can be due to the slippage or loose ground contact during the motion. In particular, it can be seen that a maximum error of 5.2 % on average was produced by the walk gait, while the trot and pace gait was found to produce a minimum error of 1.31 and 3.31%, respectively. These results support the statement that trot gait is the most stable and improves the dynamic characteristics of the robot. Since walk is a very slow gait, the robot might be prone to errors reducing the efficiency. Overall, the robot was able to achieve the gait patterns with a maximum error of only 10%, which can be further reduced, improving the stability and foot-ground contacts. Furthermore, the time taken to reach the target was found to determine the velocity of the robot. It can be seen that the velocity of the robot is not constant during all the gaits. On average, the walk gait had a velocity of 0.246 cm/s, the trot gait had 0.42 cm/s velocity, and the pace gait was 0.307 cm/s. Thus, it is understood that since walk gait is the slowest of the three, it has the lowest velocity, while trot has the highest velocity. But, however, it should be taken into consideration that the time taken for the robot to reach the targets in all three gaits is high, which in turn results in low speed. The sequence of the robot performing the three gaits is presented below in Figure 7.

Figure 7a represents the pace gait pattern where the parallel legs on either side of the robot move forward together. Figure 7b represents the walk gait where the sequence was as follows; right hind leg, left front leg, right front leg, and left hind leg. Finally, Figure 7c represents the trot gait where the diagonally opposite pair of legs move forward achieving in forward motion of the robot.

The developed algorithm can be applied for any type of legged robots, not restricted to only quadruped robots, but is limited to only flat terrain. With the establishment of the locomotion pattern, the algorithm will be able to determine the optimal joint angle and minimize the errors produced in the system by compensating for the joint angle required to make the robot reach the target destination. All the abbreviations and symbols used in this work are tabulated in Table 6 for the ease of readability.

Table 5. Theoretical and experimental comparison of gaits.

Gait	Target (cm)	Distance Travelled		Error (%)	Time Taken (s)	Velocity (cm/s)
		Theoretical (cm)	Experimental (cm)			
Walk	10	10	9	10	32	0.312
	20	20	19	5	71	0.281
	100	100	94.4	5.6	515	0.183
	200	200	198.9	0.55	943	0.210
Trot	10	10	9.8	2	20	0.500
	20	20	19.5	2.5	54	0.370
	100	100	99.6	0.4	298	0.334
	200	200	199.3	0.35	414	0.481
Pace	10	10	9	10	27	0.370
	20	20	19.7	1.5	63	0.317
	100	100	98.9	1.1	437	0.226
	200	200	198.7	0.65	623	0.318



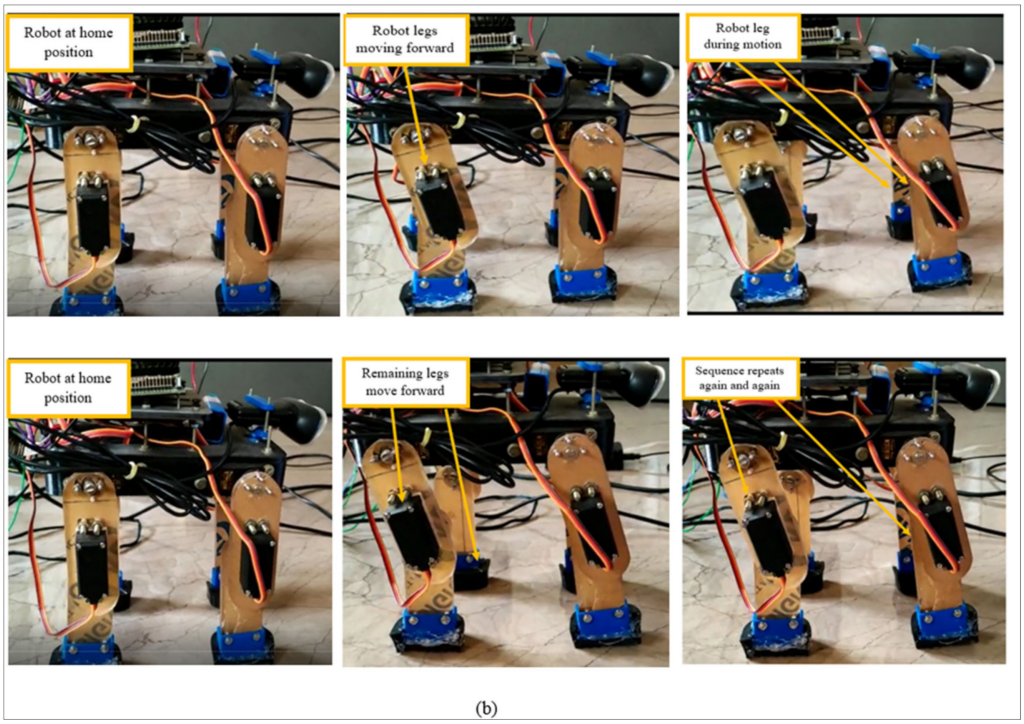


Figure 7. *Cont.*

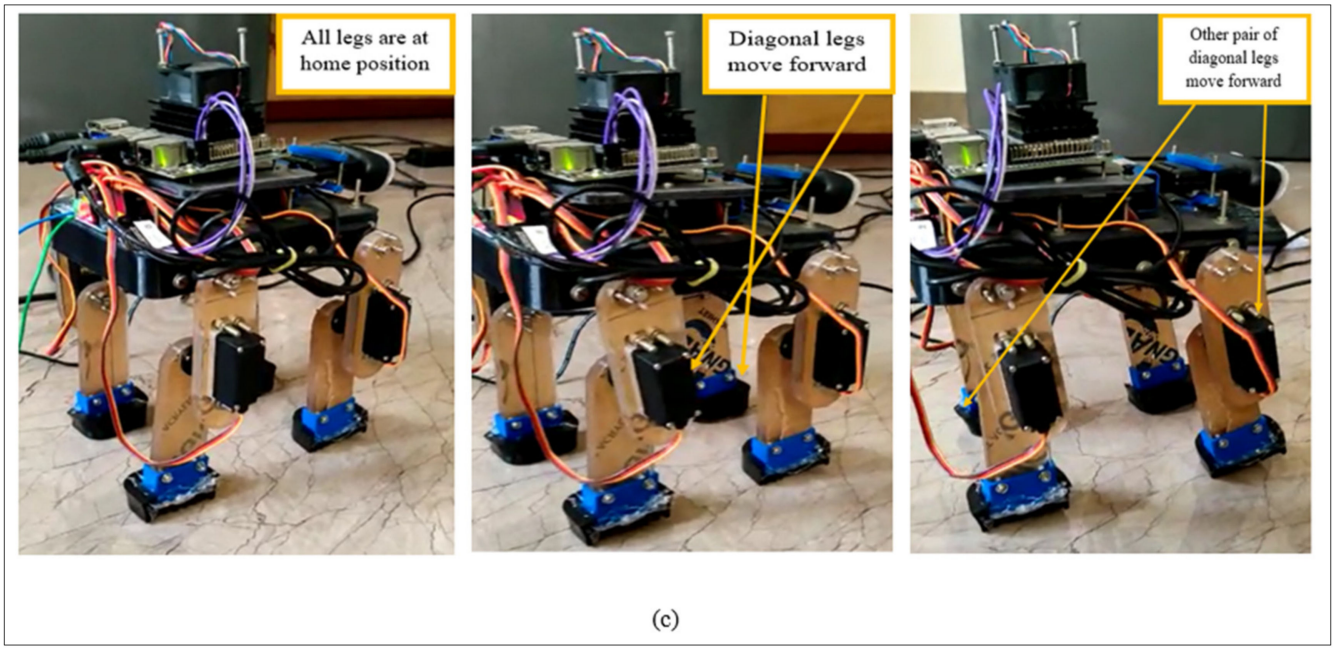


Figure 7. (**a**) Pace gait movement; (**b**) walk gait movement; (**c**) trot gait movement.

Table 6. List of symbols.

Symbols	Full Form
CoM	Center of mass
DoF	degrees of freedom
CoG	center of gravity
GA	genetic algorithm
ACO	ant colony optimization
PSO	particle swarm optimization
PET	polyethylene terephthalate
a_1 and a_2	length of links 1 and 2
θ_1 and θ_2	joint variables
$V(t)$	velocity of a particle at a given time
$X(t)$	position of a particle at a given time
P_{best}	best position reached by a particle in a swarm
G_{best}	best position reached by the swarm
C_1 and C_2	acceleration co-efficients
$rand_1$ and $rand_2$	random numbers from 0 to 1
ω	inertial weight

6. Conclusions

The use of legged robots over wheeled robots has demonstrated convincible advantages to society. These types of robots can replace humans working in military areas as well as assists humans in industries and other urban-related applications. In appraising the advantages of legged robots, this paper presented the development and investigations of various gaits for a 2 DoF mammal-inspired quadruped robot that incorporated four hip and four knee joints actuated by electronic servo motors as its locomotion element. Several analyses such as forward and inverse kinematics and stability analysis have been developed and discussed to achieve an efficient gait pattern of the robot. Due to the non-linearity of the system, particle swarm optimization (PSO) has been considered for the development of the control system. This paper focused on the development of three common symmetric gaits, i.e., walk, trot and pace. The robot was studied and investigated in three ways based on the number of steps taken to achieve the target. The robot took 10, 100, and 200 steps to reach a distance of 10, 100, and 200 cm by all the three types of

gaits. Parameters like velocity, time taken, and actual distance traveled were considered to evaluate the performance of the robot. The proposed algorithm was able to achieve global convergence of 90%, and based upon the investigations carried out it was found that the maximum error was demonstrated by walk gait, and minimum error was found during trot gait implementation. On average, the maximum error produced by the robot during all the gaits was found to be only 10%. The robot will be further studied at different speeds of motion and has to be tested in climbing at different angles to test out the off-road ability of the robot. Future works of this research include improving the performance of the current robot, developing dynamic control strategies by implementing deep learning-based neural networks to ensure the stability of the robot during the transient state, and comparing the performance of the developed algorithm with classical algorithms such as central pattern generator (CPG) and other optimization algorithms.

Author Contributions: Conceptualization, K.C.R.; methodology, L.M.; software, A.K.K.M.; validation, N.M.S.P.; investigation, T.S.; writing—original draft preparation, T.S., A.K.K.M., L.M., and N.M.S.P.; writing—review and editing, M.J., K.C.R., and I.N.; supervision, K.C.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Böttcher, S. Principles of robot locomotion. In *Proceedings of Human Robot Interaction Seminar*; 2006. Available online: http://users.dimi.uniud.it/~{}antonio.dangelo/Robotica/2012/helper/leggedRobot/RobotLocomotion.pdf (accessed on 12 August 2021).
2. Zhai, K.; Li, C.A.; Rosendo, A. Scaffolded. Gait Learning of a Quadruped Robot with Bayesian Optimisation. *arXiv* **2021**, arXiv:2101.09961.
3. Xin, G.; Xin, S.; Cebe, O.; Pollayil, M.J.; Angelini, F.; Garabini, M.; Vijayakumar, S.; Mistry, M. Robust Footstep Planning and LQR Control for Dynamic Quadrupedal Locomotion. *IEEE Robot. Autom. Lett.* **2021**, *3*, 4488–4495. [CrossRef]
4. Raheem, F.A.; Flayyih, M.K. Quadruped robot creeping gait stability analysis and optimisation using PSO. In Proceedings of the 2nd Al-Sadiq International Conference on Multidisciplinary in IT and Communication Science and Applications (AIC-MITCSA), Baghdad, Iraq, 30–31 December 2017; pp. 79–84.
5. Golubovic, D.; Hu, H. Evolving locomotion gaits for quadruped walking robots. *Ind. Robot. Int. J.* **2005**, *32*, 259–267. [CrossRef]
6. Song, S.M.; Waldron, K.J. *Machines that Walk: The Adaptive Suspension Vehicle*; MIT Press: Cambridge, MA, USA, 1989.
7. Mishra, A.K. *Design, Simulation, Fabrication and Planning of Bio-Inspired Quadruped Robot*; Requirement of the Degree of Master of Technology; Indian Institute of Technology Patna: Bihar, India, 2014.
8. Estremera, J.; Gonzalez de Santos, P. Free gaits for quadruped robots over irregular terrain. *Int. J. Robot. Res.* **2002**, *21*, 115–130. [CrossRef]
9. Wang, P.; Song, C.; Li, X.; Luo, P. Gait planning and control of quadruped crawling robot on a slope. *Ind. Robot. Int. J. Robot. Res. Appl.* **2019**, *47*, 12–22. [CrossRef]
10. Xi, W.; Yesilevskiy, Y.; Remy, C.D. Selecting gaits for economical locomotion of legged robots. *Int. J. Robot. Res.* **2016**, *35*, 1140–1154. [CrossRef]
11. Yang, X.S. *Nature-Inspired Metaheuristic Algorithms*; Luniver Press: Bristol, UK, 2010.
12. Winkler, A.W.; Farshidian, F.; Neunert, M.; Pardo, D.; Buchli, J. Online walking motion and foothold optimisation for quadruped locomotion. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 5308–5313.
13. Focchi, M.; Orsolino, R.; Camurri, M.; Barasuol, V.; Mastalli, C.; Caldwell, D.G.; Semini, C. Heuristic planning for rough terrain locomotion in presence of external disturbances and variable perception quality. In *Advances in Robotics Research: From Lab to Market*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 165–209.
14. Shen, W.; Li, M.; Wang, J.; Chai, C.; Zheng, W. An Algorithm for Quadruped Robot's Inverse Kinematic Problems Based on PSO. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 5152–5157.
15. Rong, C.; Wang, Q.; Huang, Y.; Xie, G.; Wang, L. Autonomous evolution of high-speed quadruped gaits using particle swarm optimisation. In *Robot Soccer World Cup*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 259–270.
16. Sen, M.A.; Bakircioglu, V.; Kalyoncu, M. Inverse Kinematic Analysis of a Quadruped Robot. *Int. J. Sci. Technol. Res.* **2017**, *6*, 285–289.

17. Wettergreen, D.; Thorpe, C. Gait generation for legged robots. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Raleigh, NC, USA, 7–10 July 1992.
18. Abourachid, A. A new way of analyzing symmetrical and asymmetrical gaits in quadrupeds. *Comptes Rendus Biol.* **2003**, *326*, 625–630. [CrossRef]
19. A Guide to Quadruped Robot's Gaits. Available online: https://www.animatornotebook.com/learn/quadrupeds-gaits (accessed on 19 May 2021).
20. Laskari, E.C.; Parsopoulous, K.E.; Vrahatis, M.N. Particle Swarm Optimisation for Integer Programming. In Proceedings of the 2002 Congress on Evolutionary Computation, Honolulu, HI, USA, 12–17 May 2002; Volume 2, pp. 1582–1587.
21. Kennedy, J.; Eberhart, R. Particle Swarm Optimisation. In Proceedings of the ICNN'95–International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.