Journal of Intelligent & Robotic Systems (2021) 102: 29
https://doi.org/10.1007/s10846-021-01388-0

SHORT PAPER

Behavior-Based Navigation of an Autonomous Hexapod Robot Using a Hybrid Automaton

Mostafa Khazaee[1] · Majid Sadedel[1] · Atoosa Davarpanah[1]

Received: 24 July 2020 / Accepted: 31 March 2021 / Published online: 7 May 2021

Abstract

The hexapod robot is one of the important classes in legged robots due to its great potential to operate in complex settings with high stability and flexibility. However, few researches have investigated the navigation and autonomous locomotion of this type of robot. This paper concerns with the behavior-based control and navigation of an autonomous hexapod robot utilizing a hybrid automaton. Switching between the distinct behaviors is based on the sensory data, and no representation of the environment is included. Since these systems are likely to rise chattering phenomenon, a sliding mode including clockwise and counter-clockwise boundary following behaviors are considered between the goal attraction and obstacle avoidance modes to modify the automaton. The hybrid automaton undertakes the path planning of a reference point instead of the robot. Thus, in order to be able to implement the navigation algorithm, the hexapod robot is converted to a point mass robot within two transformations. A parameter study is also performed to investigate the effects of the controllers' design parameters on the performance of the navigation algorithm and robot. The results show that enhancing the smoothness of the robot's motion would deteriorate the precision in tracking the reference point and the reaction speed and vice versa. Moreover, simulation tests confirm the effectiveness of the navigation algorithm in generating the optimal path by perfectly switching between distinct modes as well as the capability of the robot to follow the reference point with an arbitrary gait. Furthermore, comparing the performance of the presented navigation strategy to that of similar algorithms, such as Bug and Potential field, yields a satisfying result.

Keywords Hybrid system · Navigation algorithm · Autonomous · Hexapod robot · Mobile robot

Nomenclature of the Parameters

a	The length of the robot's platform
b	The width of the robot's platform
d	The lateral distance between the foot tip and the platform when the robot is in its home position
α	The angle of the coxa (hip angle)
β	The angle of the femur (knee angle)
γ	The angle of the tibia (ankle angle)
h	The height between the coxa link and the ground
l_1	The length of the coxa
l_2	The length of the femur
l_3	The length of the tibia
$\sum_G$	The global coordinate
$\sum_B$	The body coordinate
$\sum_L$	The leg coordinate
R_B^L	The rotation matrix between the body to leg coordinates
P_B^L	The translation matrix between the body to leg coordinates
R_G^B	The rotation matrix between the body to global coordinates
$[J]$	The Jacobian matrix
η	The duty factor (gait)
T	The cycle time of the robot
$T_{Support}$	The duration in which the leg is in the support phase
$T_{Transfer}$	The duration in which the leg is in the transfer phase

✉ Majid Sadedel
majid.sadedel@modares.ac.ir

Mostafa Khazaee
mostafa.khazaee@modares.ac.ir

Atoosa Davarpanah
atoosa.davarpanah@modares.ac.ir

[1] Department of Mechanical Engineering, Tarbiat Modares University, Tehran, Iran

$\widetilde{\dot{X}}$	The velocity vector of the reference point (the outcome of the controller)
$\widetilde{X}$	The position of the reference point
X_G	The position of the target point
K_{GA}	The gain of the goal attraction mode controller
e_{GA}	The distance between the reference point and the goal
ξ	The tuning parameter of the goal attraction mode controller
v_0	The critical velocity of the robot
X_O	The position of the point obstacle
e_{OA}	The distance between the reference point and the point obstacle
d_O	The minimum allowed distance between the reference point and the point obstacle
K_{OA}	The gain of the obstacle avoidance mode controller
c	The tuning parameter of the obstacle avoidance mode controller
ε	The tuning parameter of the obstacle avoidance mode controller
λ	The tuning parameter in the sliding mode (boundary following behavior)
R_{OA}^{BF}	The rotation matrix between the direction of the velocity vector in the obstacle avoidance behavior and the direction of that in the boundary following mode
d_τ	The distance between the reference point and the goal when the sliding mode is activated
$\widetilde{X}(\tau)$	The position of the reference point when the sliding mode is activated
R	The curvature radius
v	The velocity of the robot's CoG
ω	The angular velocity of the robot's CoG
θ	The orientation of the robot with respect to the Y_G-axis
L	The distance between the robot's CoG and the reference point

1 Introduction

1.1 State of the Art

A mobile robot is a robot that can move around in its environment and is not fixed to one physical location despite the industrial or manipulating robots. They have great potential for identifying and rescuing applications in surroundings that are unsafe for human beings. Mobile robots can be autonomous, which raises the need for the capability to work in a dynamic and altering setting without the help of physical or electro-mechanical guidance devices. Several methods have been proposed for the navigation and control of autonomous mobile robots in the literature [1–9].

A promising method for constructing a control system in order to achieve the ability to navigate autonomous robots in dynamic and varying surroundings is a behavior-based control strategy [10, 11]. The core idea of this approach is to design separate controllers for predefined behaviors. The preferred robot behaviors are based on sensory feedbacks. A behavior can, for example, be obstacle avoidance where information of the sensors about a nearby obstacle would yield a motion in which the robot escapes from the obstacle. This technique of configuring the control algorithm into different behaviors, devoted to executing particular tasks such as obstacle avoidance or goal attraction, has been appreciated as a promising design approach. One of the most essential advantages of this method is that it creates an integrated system, which makes the design procedure simpler. It also gives us the possibility to add new modes to the system without posing any substantial growth in complexity. Egerstedt [12] experimentally verified the performance of a behavior-based control approach on a Nomad 200 mobile platform. Moreover, he illustrated that separated behaviors could be fused together on the basis of some priorities to form a new behavior. Egerstedt et al. [13] studied the gradient descent method to optimize the mode scheduling in hybrid systems and suggested a bilevel hierarchical algorithm for that.

Mobile robots can be classified into several categories based on their working environment and motion abilities. Mobile robots that operate in contact with the ground are divided into wheeled [14], tracked [15], legged [16], and leg-wheeled robots [17]. Wheeled and tracking robots can quickly transfer in continuous fields, whereas legged robots have an excellent performance over irregular and rugged lands. Accordingly, legged robots have been the subject of interest in recent years [18–21]. Among prevailing legged robots, the hexapod robot, which is inspired from the hexapod animals, has attracted much attention due to its stability and flexibility in comparison with other legged robots. Besides, since the robot can be statically stable even when some of the legs are not in contact with the ground, the free legs can do manipulation. Also, it makes the robot be able to walk when one or two legs are out of commission. Numerous researches have been performed to explore the inverse kinematics, turning pattern, gait pattern, stability, and so on. For instance, Zhong et al. [22] introduced a biomimic method, which was based on Matsuoka's neural oscillator, for controlling the locomotion and gait planning of a hexapod robot. Manglik et al. [23] proposed an adaptive gait generation using a genetic algorithm for hexapod robots to sustain their locomotion when one leg breaks down. Khudher et al. [24] employed a quadratic programming technique to solve the inverse kinematics of a hexapod robot with inequality constraints and verified the algorithm on a leg with three joints.

Gao et al. [25] offered a technique to reduce the impact force and energy consumption of a hexapod robot by providing a scheme of motion planning subject to velocity and acceleration limits. Zhu et al. [26] proposed a random gait pattern generation for the rotation of a bio-inspired hexapod robot. He et al. [27] investigated the mobility features of a wall climbing hexapod robot by analyzing the degrees of freedom, workspace, and singularity of the robot.

1.2 Contribution

To the best of authors' knowledge, few studies have been dedicated to investigating the autonomous navigation of the hexapod robot. The presented work is devoted to the behavior-based control of an autonomous hexapod robot. The navigation algorithm is modeled as a hybrid automaton, where each different node is linked to a specific behavior. The "goal attraction" and "obstacle avoidance" modes are the primary behaviors of the system, and hard switches are used for transitions between the modes. Besides, a sliding mode which is a boundary following behavior is considered on the switching surface to both prevent the so-called chattering phenomena andenable the robot to bypass the obstacles. Although the most related studies proposed control structures which make legged robots walk over obstacles and cross somewhat uneven land [28–32], it seems better to bypass large obstacles in practice. Since the navigation algorithm is based on the optimal trajectory planning for a point mass to avoid dealing with the sophisticated kinematics of the robot in the on-line navigation, the motion of the complex hexapod robot is associated with a reference point using two transformations. Here, each leg of the robot has three joints. Given the trajectories of the robot's center of gravity (CoG) and legs from the outcome of the navigation algorithm, these three joint angles are calculated by solving the inverse kinematics using a geometrical approach. In addition, the effects of the controller's parameters on the robot's behavior are also investigated in this paper. Moreover, different simulation tests are performed to verify the performance of the whole system. It should be noted that all the calculations and simulations are executed in MATLAB/Simulink.

1.3 Outline

The rest of this article is organized as follows. In Section 2, the kinematics model of the hexapod robot is depicted, and after obtaining the forward kinematics, the inverse kinematics is solved using a geometrical approach. Additionally, the popular gait patterns of the hexapod robot are presented. Section 3 is devoted to describing the navigation algorithm and designing the corresponding controllers. Section 4 illustrates the implementation of the navigation algorithm on the hexapod robot. The performance of the robot when guided by the presented navigation algorithm is investigated in Section 5 through simulations. Moreover, Section 6 elaborates on the performance of the robot and the navigation algorithm by studying the system in different conditions. Furthermore, a comparative study is performed in Section 7 to compare the performance of the behavior-based navigation algorithm to that of similar algorithms. At the end, the concluding remarks are encapsulated in Section 8.

2 Robot Description and Kinematics

The kinematics model of the studied hexapod robot is presented in Fig. 1. Each leg of the robot includes three rotational joints, which are equivalent to the hip, knee, and ankle of an insect's leg. As displayed in the figure, the parameters of joints are defined as the angle of the coxa (α), the angle of the femur (β), and the angle of the tibia (γ).

Also, the height between the coxa link of the i^{th} leg and ground is h_{i}. The length of the coxa, femur, and tibia of the i^{th} leg are respectively $l_{1\text{i}}$, $l_{2\text{i}}$, and $l_{3\text{i}}$. Moreover, $\sum_G$ is the global, $\sum_B$ is the body, and $\sum_L$ is the leg coordinates of the robot.

The transformation matrix between the body coordinates $\sum_B$ to leg coordinates $\sum_L$ is achieved as follows:

$$T_B^{L,i} = \begin{bmatrix} R_B^{L,i} & P_B^{L,i} \\ 0 & 1 \end{bmatrix} \in R^{4\times 4} \tag{1}$$

where R_B^L and P_B^L denote the rotation and translation matrices between the body to leg coordinates.

Utilizing the forward kinematics, one can calculate the position of the foot tip X_{Tip} in the body coordinates system as a function of joint angles as below:

$$X_{Tip}^i = \begin{bmatrix} x_{Tip}^i \\ y_{Tip}^i \\ z_{Tip}^i \\ 1 \end{bmatrix} = T_B^{L,i} \begin{bmatrix} (l_{3i}\cos(\gamma_i - \beta_i) + l_{2i}\cos(\beta_i) + l_{1i})\sin(\alpha_i) \\ (l_{3i}\cos(\gamma_i - \beta_i) + l_{2i}\cos(\beta_i) + l_{1i})\cos(\alpha_i) \\ l_{2i}\sin(\beta_i) - l_{3i}\sin(\gamma_i - \beta_i) \\ 1 \end{bmatrix} \tag{2}$$

Assuming that the position of the foot tip is given, one can solve the inverse kinematics problem using a geometrical approach to find the joint angles of the legs [33]. The solution procedure is occurred in four steps that are performed to obtain some vectors and angles required for finding the joint angles. It should be noted that the solution includes minimum possible calculations making it suitable for rapid calculation during motion [33].

(1) The First Loop Closure: Calculation of the Vector $\overrightarrow{l_i}$

Figure 2a shows how the vector $\overrightarrow{l_i}$ which connects the robot's foot tip to the hip is achieved via the first loop closure.

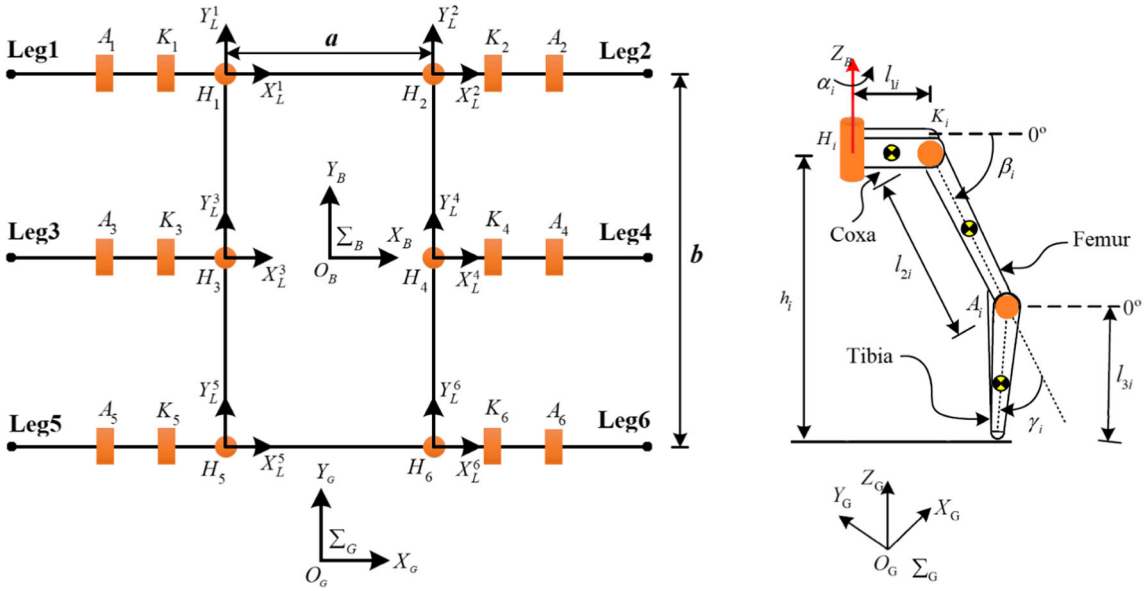

Fig. 1 The model of the robot

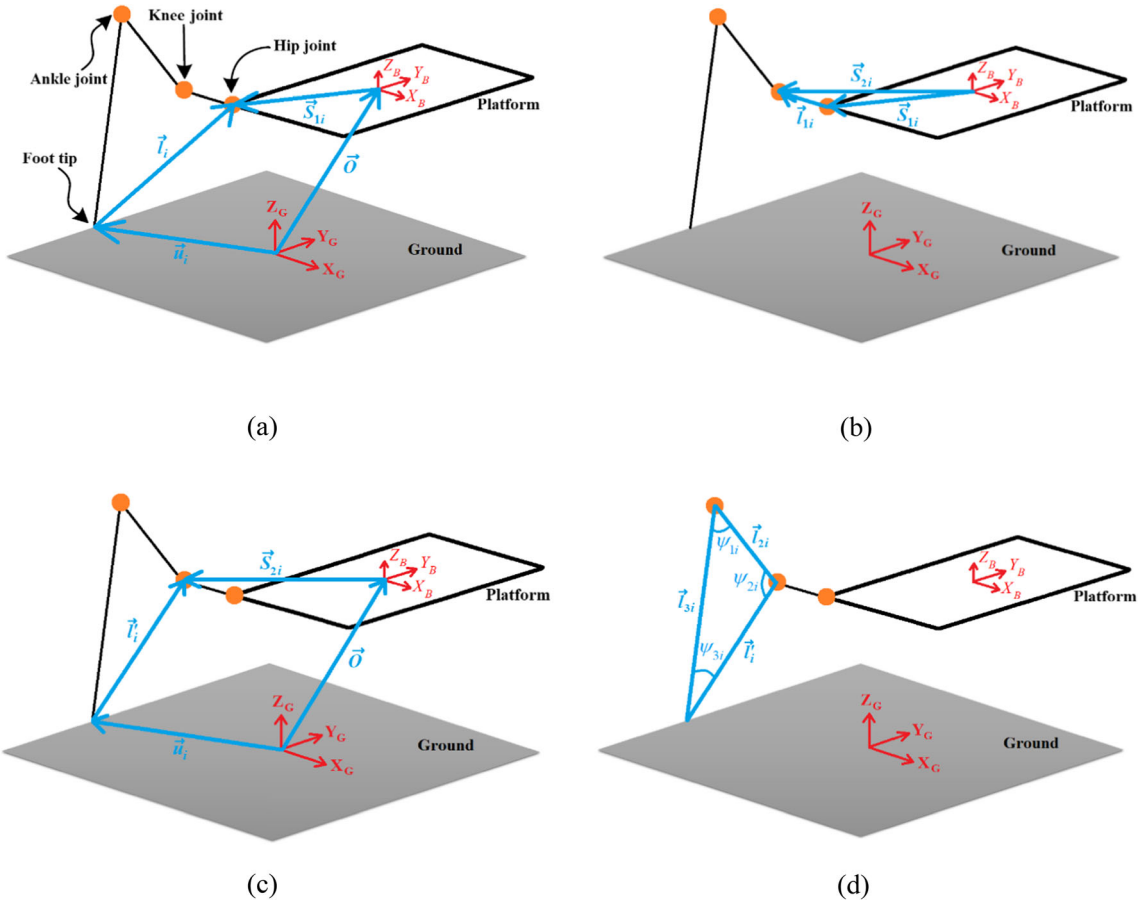

Fig. 2 The loop closures for calculating the required angles and vectors in order to solve the inverse kinematics. **a** The first loop closure which is used for calculating $\overrightarrow{l_i}$: the vector connecting the robot's foot tip to the hip, **b** The second loop closure which is used for calculating $\overrightarrow{S_{2i}}$: the vector connecting the robot's CoG to the knee, **c** The third loop closure which is used for calculating $\overrightarrow{l_i}$: the vector connecting the robot's foot tip to the knee, **d** The fourth loop closure which is used for calculating the internal angles of the triangle obtained from connecting the foot tip, the ankle, and the knee to each other

Here, vectors O, s_{1i}, and u_i are respectively the position of CoG of the robot with respect to the global coordinate, the position of the i^{th} hip joint in the body coordinate, and the position of the i^{th} foot tip with respect to the global coordinate. According to the figure, the vector l_i is obtained from the following equation:

$$\vec{l}_i = \vec{O} + R_B^G \vec{s}_{1i} - \vec{u}_i \tag{3}$$

where R^G_B is the rotation matrix for coordinates $\sum_B$ to $\sum_G$. Now, one can easily calculates the hip joint angle as:

$$\alpha_i = \tan^{-1}\left(\frac{l_{i,Y}}{l_{i,X}}\right) \tag{4}$$

(2) The Second Loop Closure: Calculation of the Vector $\overrightarrow{S_{2i}}$

The second loop closure, which is presented in Fig. 2b, is employed to calculate the i^{th} knee joint vector s_{2i} with respect to the body coordinate as:

$$\vec{s}_{2i} = \begin{bmatrix} \vec{s}_{1i,x} + (-1)^i l_{1i}\cos(\alpha_i) \\ \vec{s}_{1i,y} + (-1)^i l_{1i}\sin(\alpha_i) \\ \vec{s}_{1i,z} \end{bmatrix} \tag{5}$$

(3) The Third Loop Closure: Calculation of the Vector $\vec{l}_i$

Figure 2c demonstrates the third loop closure that is utilized for obtaining the vector l_i' connecting the robot's foot tip to the knee:

$$\vec{l}_i' = \vec{O} + R_B^G \vec{s}_{2i} - \vec{u}_i \tag{6}$$

(4) The Fourth Loop Closure: Calculation of the Internal Angles of the Triangle Obtained from Connecting the Foot Tip, the Ankle, and the Knee to each Other

In the fourth loop closure, which is shown in Fig. 2d, a triangle is appeared by connecting the foot tip, the ankle, and the knee to each other. Accordingly, one can utilize the law of cosines to relate the lengths of the sides of the triangle to the cosine of each of its angles. The mentioned law states:

$$\psi_{1i} = \cos^{-1}\left(\frac{{l_{2i}}^2 + {l_{3i}}^2 - {l_i'}^2}{2 l_{2i} l_{3i}}\right) \tag{7a}$$

$$\psi_{2i} = \cos^{-1}\left(\frac{{l_{2i}}^2 + {l_i'}^2 - {l_{3i}}^2}{2 l_{2i} l_i'}\right) \tag{7b}$$

$$\psi_{3i} = \cos^{-1}\left(\frac{{l_{3i}}^2 + {l_i'}^2 - {l_{2i}}^2}{2 l_i' l_{3i}}\right) \tag{7c}$$

According to the aforementioned loops as well as the relations presented in Fig. 3, one can easily obtain the angle of femur (β) and tibia (γ) as the following equations:

$$\beta_i = \psi_{2i} - (\rho_i + \phi_i) \tag{8}$$

$$\gamma_i = \pi - \psi_{1i} \tag{9}$$

where:

$$\rho_i = \tan^{-1}\left(\frac{h_i'}{\sqrt{{l'}^2_{i,X} + {l'}^2_{i,Y}}}\right), \phi_i = \sin^{-1}\left(\frac{h_i' - h_i}{l_{1i}}\right) \tag{10}$$

In addition, based on the position of the foot tip, the velocity as well as the acceleration of the joints can be presented as:

$$\left[\dot{\alpha}_i \dot{\beta}_i \dot{\gamma}_i\right] = [J]^{-1}\left[\dot{X}^i_{Tip}\right] \tag{11}$$

$$\begin{bmatrix} \ddot{\alpha}_i \\ \ddot{\beta}_i \\ \ddot{\gamma}_i \end{bmatrix} = [J]^{-1}\left[\ddot{X}^i_{Tip}\right] + \left[\dot{J}\right]^{-1}\left[\dot{X}^i_{Tip}\right] \tag{12}$$

where J is the Jacobian matrix, and is presented in Appendix 1.

It should be noted that one can obtain the singularities by equating the determinant of the Jacobian matrix J with zero as bellow:

$$\sin(\gamma_i) = 0 \tag{13a}$$

$$l_{1i} + l_{2i}\cos(\beta_i) + l_{3i}\cos(\gamma_i - \beta_i) = 0 \tag{13b}$$

The proper motion of the robot relies on the coherent action of its legs. The gait configuration of the robot is obtained by the legs that drive the body toward the desired direction on the ground in a support phase, while the other ones move the legs

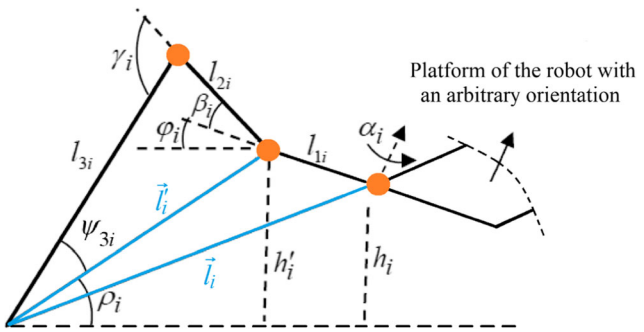

Fig. 3 Knee and ankle angles calculation

of the ground in a transfer phase. One can say that the robot may act as parallel and serial robots in support and transfer phases, respectively. Gait generation is realized by switching between support and transfer phases in different sequences. Duty factor η illustrates the portion of a cycle time in which the leg is in the support phase:

$$\eta = \frac{T_{Support}}{T} = 1 - \frac{T_{Transfer}}{T} \tag{14}$$

where T_{Support}, T_{Transfer}, and T denote the time period of support phase, transfer phase, and one cycle. It is noteworthy to mention that one can find the number of legs that are on the ground (support phase) by multiplying the duty factor to the number of legs.

The most popular gait patterns of the hexapod robot are tripod ($\eta = 1/2$), diagonal ($\eta = 2/3$), and wave gaits ($\eta = 5/6$) that are demonstrated in Fig. 4. Also, it was mentioned in previous studies that the range of duty factor should be ($1/2 < \eta < 5/6$) to guarantee the steady motion of the robot [26].

In Fig. 4a-c, rows correspond to legs, and each column represents 1/12 of the duration of a full cycle. The figures clearly show the phase of legs in any given moment within a cycle for different gait patterns. For instance, it can be interpreted from Fig. 4a that in the interval [0 - T/2], the legs 1, 3, and 5 are in contact with the ground (support phase), while the other legs are in the swing phase. In the second half of the cycle, the opposite scenario would occur. In other words, the legs 1, 3, and 5 are in transfer phase, and other legs are in contact with the ground. Moreover, it is axiomatic that for higher values of η, more legs are in contact with the ground, so the robot is more stable. For instance, when $\eta = 1/2$, $\eta = 2/3$, and $\eta = 5/6$, three, four, and five legs are in contact with the ground in any moment, respectively. Furthermore, one should note that the duration of the transfer phase decreases by increasing η; thus, the robot has a lower maximum velocity in higher values of η compared to the condition in which the robot takes a gait pattern with a lower η. It stems from the fact that since in a gait with a high η, such as the wave gait, each leg has a very low time for the transfer phase compared to that for the support phase, the robot's velocity would be limited. In contrast, in tripod gait, which is the fastest stable gait for the hexapod, having dedicated a similar amount of time to both transfer and support phases, the robot can fulfill its potential and walk at a relatively high pace.

The considered geometric parameters of the robot are indicated in Table 1.

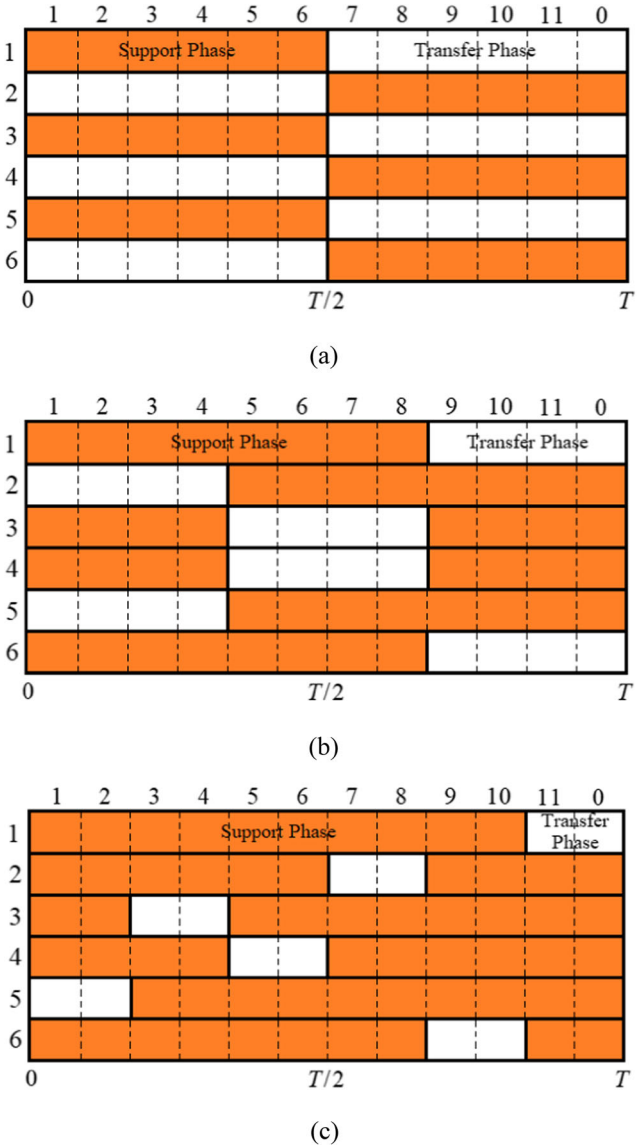

Fig. 4 Gait patterns of the robot. **a** Tripod Gait ($\eta = 1/2$), **b** Diagonal Gait ($\eta = 2/3$), **c** Wave Gait ($\eta = 5/6$)

3 Navigation Algorithm

As already stated, one of the significant methods for organizing the controllers of autonomous robots functioning in partly unknown surroundings is within a behavior-based framework [10, 11]. In this approach, distinctive robot behaviors such as "goal attraction" and "obstacle avoidance" are considered, and switching between the mentioned behaviors is based on the sensory data, and no representation of the environment is included. Also, a sliding mode will be introduced between the goal attraction and the obstacle avoidance modes to eradicate the potential problem occurring when switching between the mentioned primary behaviors. Therefore, we consider three different behaviors for the robot. In this article, the sliding

Table 1 Geometric parameters of the robot

a (m)	b (m)	h (m)	l_1 (m)	l_2 (m)	l_3 (m)	d (m)
0.06	0.15	0.07	0	0.05	0.1	0.05

mode refers to the boundary following behavior by which the robot can perfectly bypass obstacles. Besides, it will be illustrated that this behavior enables the robot to perform well in the deadlock maps or mazes.

3.1 Controller Design

Given the fact that it is not appropriate to deal with the actual kinematics of the robot in an on-line navigation due to its too high computational cost, one would structure the controllers on the basis of the optimal trajectory generation for a point mass (the reference point) [12]. Accordingly, this section concerns with setting the controllers for the two major behaviors of the robot: "goal attraction" and "obstacle avoidance" to control the motion of the reference point. It will be shown that the motion of the robot in the boundary following behavior (the sliding mode) is based on the controller that is designed for the obstacle avoidances mode. In addition, it is noted that the kinematics of the robot will be added when following the generated path.

When it comes to choosing the controllers, undoubtedly, one would have several options. For instance, one can utilize a fuzzy control [34], adaptive control [35], and so on [36]. Nonetheless, owing to the simplicity of the PID controller and the fact that it can properly satisfy the control objectives of the behavior based navigation [12], this type of controller is used in this study.

3.1.1 Goal Attraction Behavior

For the point-to-point motion control of a point mass in the goal attraction behavior, one would introduce the following controller:

$$\tilde{X} = k_{GA} e_{GA} \tag{15}$$

where $\tilde{X} = [\tilde{x}\ \tilde{y}]^T$ and $e_{GA} = X_G - \tilde{X}$. It can be interpreted from Eq. (15) that the presented controller may apply an enormously high input to actuators when the robot is too far from the goal point. To cope with this problem, one may write k_{GA} as a function of error e_{GA} as:

$$k_{GA} = \frac{v_0\left(1 - e^{-\zeta |e_{GA}|^2}\right)}{|e_{GA}|} \tag{16}$$

Here, ζ and v_0 take positive values and are the tuning parameter for the controller and the critical velocity of the robot, respectively. In order to see how the controller works and study the effect of the parameter ζ on the response of the system, the variation of $\tilde{X}$ with respect to the magnitude of e_{GA} is investigated in Fig. 5.

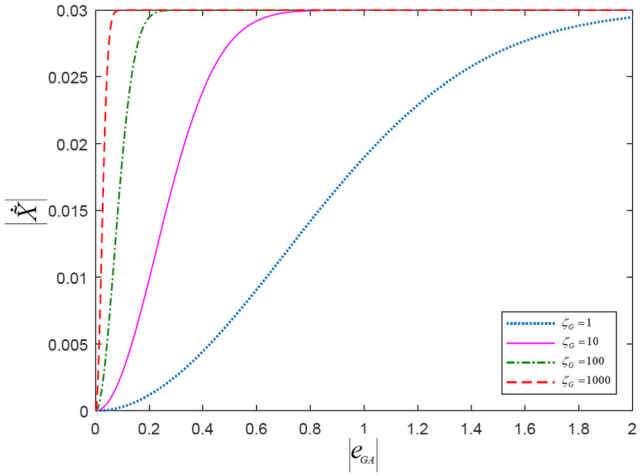

Fig. 5 The velocity magnitude produced by the controller as a function of e_{GA} in the goal attraction behavior for several ζ with $v_0 = 0.03$ m/s

The figure shows that the robot would move with the desired critical velocity v_0 toward the target point, even when it is considerably far from that. As the robot gets closer to the target point, it starts to decrease its velocity to stop gradually. In other words, one can imagine a circular area around the target point that the robot begins to reduce its velocity when it enters this zone. Also, increasing the value of ζ would shrink the radius of this area. Thus, in circumstances that raise a need for smooth velocity changes, ζ should not take a high value.

3.1.2 Obstacle Avoidance Behavior

As soon as the sensors of the robot that are assumed to cover a circular area around the robot's COG recognize a point-obstacle at (x_O, y_O) which is closer to the robot than d_O (safety distance) the obstacle avoidance mode with the following controller would be activated:

$$\tilde{X} = k_{OA} e_{OA} \tag{17}$$

where $\tilde{X} = [\tilde{x}\ \ \tilde{y}]^T$ and $e_{OA} = X_O - \tilde{X}$. The controller produce a velocity vector to move the robot along $\theta_{OA} = \arctan 2(y_O - y, x_O - x)$. Despite the fact that the controller of the goal attraction behavior must be designed in such a way to make the response of the system stable, the presented controller in Eq. (17) need to produce an unstable motion in order to force the robot to escape from the obstacle. However, the robot may show an extremely unstable motion for higher values of e_{OA}, which is not desirable. To avoid this, one can consider k_{OA} as:

$$k_{OA} = -\frac{1}{|e_{OA}|}\left(\frac{c}{|e_{OA}|^2 + \varepsilon}\right) \tag{18}$$

where, c and ε are the tuning parameters of the controller and take positive values. ε prevents the controller from producing extremely high velocities that can damage the system when

the robot is considerably close to the obstacle ($e_{OA} \rightarrow 0$). It is noted that c/ε approximately equates with the velocity of the robot when it is extremely close to the obstacle. Thus, one must choose the value of c and ε in a way that the value of c/ε is not higher than the critical velocity (v_0) of the robot ($c/\varepsilon \leq v_0$). Figure 6 depicts the magnitude of the velocity vector produced by the controller as a function of e_{OA} for several c and ε. According to the figure, one can say that the controller starts to generate a repulsive force as the robot approaches the obstacle to prevent the robot from hitting that. Moreover, it shows that increasing the values of c and ε would enhance the amount of e_{OA}, which can be considered as a threshold below which the controller gets activated, and the consequent smoothness of the robot's reaction when detecting an obstacle. However, this may cause the robot to escape from obstacles that are even behind it and are not interrupting the robot.

It is noteworthy to mention that in the actual application, a real obstacle cannot be considered to be a point. However, the output of the majority of sensors is a point. Therefore, in order to be able to utilize the presented algorithm in practice, one can assign a weight (w) to the corresponding X_O of each sensor on the basis of some criteria and use the vector summation to calculate the equivalent X_O as illustrated in Fig. 7. The position of the detected points with respect to the robot's position as well as the direction of its motion are the primary criteria in the weight assignment procedure. For instance, the points that are either behind the robot posing no problem or points with higher distances with respect to the robot would be associated with low weights and vice versa.

3.2 The Hybrid Automaton

The most straightforward approach for controlling an autonomous hexapod robot toward the desired goal without hitting

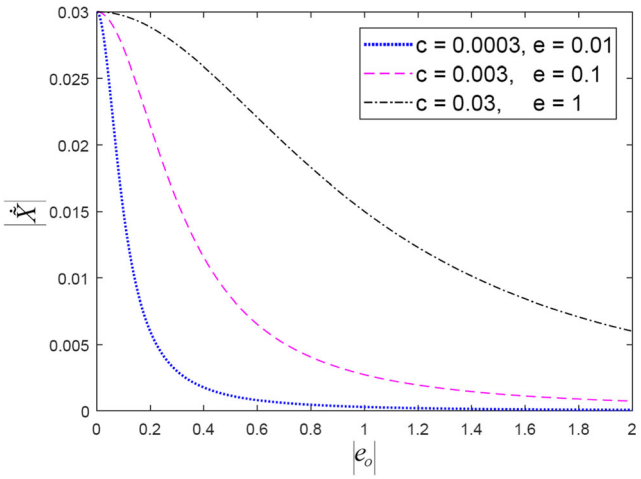

Fig. 6 The velocity magnitude produced by the controller as a function of e_O in the obstacle avoidance behavior for several c and ε when $c/\varepsilon = v_0 = 0.03$ m/s

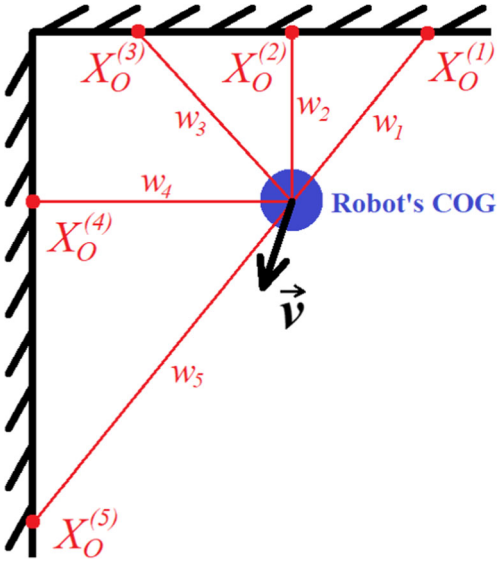

Fig. 7 The sensory data processing

any obstacle is to utilize hard switches between the goal attraction and obstacle avoidance behaviors corresponding to two nodes in an automaton presented in Fig. 8. However, it may enhance the risk of rising chattering phenomena in the system, which is contributed to the so-called *Zeno* properties of the system [37]. This phenomenon refers to a various number of discrete switches in limited time in a hybrid system. This type of systems includes continuous flows in the direction of the switching surface, which yields a different flow on the mentioned surface [12, 38].

To this end, one would modify the automaton by defining a new behavior that corresponds to the flow mentioned above. Figure 9 presents the underlying notion. The added node is the sliding mode that is introduced on the switching surface among the goal attraction and obstacle avoidance behaviors. Here, the sliding mode is the "boundary following" behavior in which the robot would follow the boundary of the safety area surrounding an obstacle when detecting it. It should be noted that in case of a point obstacle, the robot would follow an imaginary boundary around the obstacle.

It can be easily shown that the direction of the produced path by the controller in the obstacle avoidance behavior is

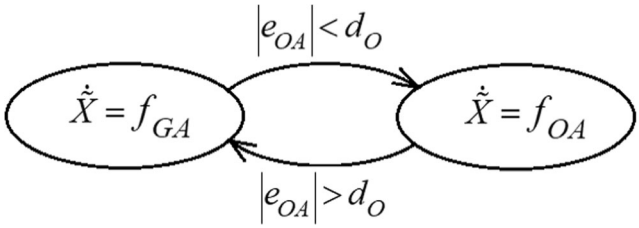

Fig. 8 The hard switches hybrid automaton

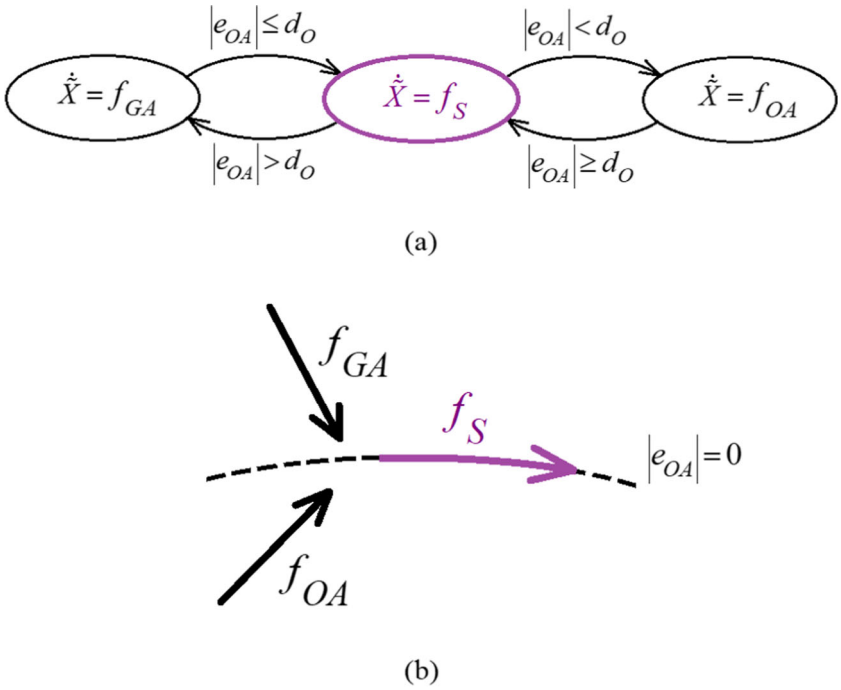

Fig. 9 Modification of the Zeno hybrid automaton

orthogonal to the boundary of the obstacle. Consequently, the velocity vector for the sliding mode can be expressed as:

$$\dot{\tilde{X}} = \lambda R_{OA}^{BF}\left(\pm\frac{\pi}{2}\right)k_{OA}e_{OA} \tag{19}$$

λ is a positive value.R_{OA}^{BF} is the rotation matrix from the direction of the produced path in the obstacle avoidance behavior to the direction of that in the boundary following mode. It is apparent from Eq. (19) that the outcome of the sliding mode (boundary following behavior) equates with the produced velocity vector by the obstacle avoidance mode in the perpendicular direction multiplied into λ. The negative and positive signs of the rotation angle respectively correspond to the clockwise (CW) and counter-clockwise (CCW) rotation of the robot during the sliding mode to follow the boundary around the obstacle. These two different options in the sliding mode will help the robot choose the shortest path to circle the obstacle. The complete navigation system hybrid automaton is depicted in Fig. 10.

The automaton shows that the motion begins with the goal attraction behavior. When an obstacle is detected by the robot and the required condition for the activation of the sliding mode is met, the boundary following mode is activated, which makes the robot follow the boundary of the obstacle. Once the inner product of the velocity vector generated by the clockwise boundary following mode's and that by the goal attraction behavior takes a positive value, the navigation algorithm would activate the clockwise boundary following mode and vice versa. Also, as soon as the robot realizes an obstacle closer than the safety distance, the obstacle avoidance mode is activated to avoid any prospective collision. In order to switch back to the goal attraction behavior from the sliding mode, the two conditions: "progress" and "clear shot" that are respectively reflected in the hybrid automaton as $e_{GA} < |\tilde{X}(\tau) - X_G| = d_\tau$ and $\left\langle \dot{\tilde{X}}_{OA}, \dot{\tilde{X}}_{GA} \right\rangle > 0$ must be checked.

Where τ is the time when the boundary following mode is initiated. The satisfaction of the first condition shows that the robot is closer to the goal point with respect to the initial moment of the boundary following behavior. In other words, it indicates the progress of the distance error (e_{GA}) reduction, which is the main purpose of the navigation algorithm. The second condition, which is associated with the inner product of the velocity vectors in the goal attraction and the obstacle avoidance behaviors, would be satisfied when the directions of the mentioned vectors are compatible with each other. It shows that the optimal path toward the goal point is not intersecting the safety region around the obstacle.

It should be noted that the condition $|\tilde{X} - X_O| = d_O$ would occur neither in experimental tests nor simulations. To this end, one should use a fat guard, in which the mentioned condition is checked when $d_O - \delta < |\tilde{X} - X_O| < d_O + \delta$ (Fig. 11).

4 Implementation

In the previous section, we illustrated a behavior-based control approach utilizing a hybrid automaton. However, the

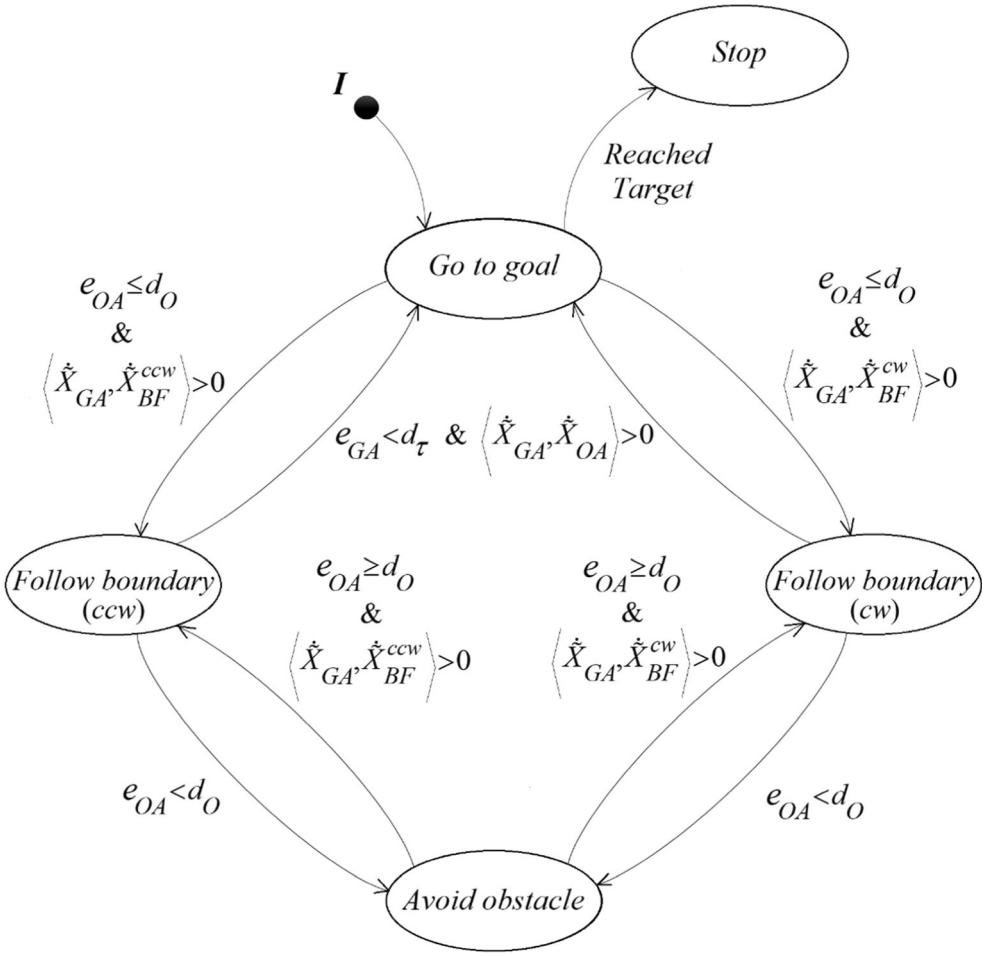

Fig. 10 The complete navigation system hybrid automaton

algorithm was based on a point mass. To come up against this limitation and apply the navigation algorithm to the hexapod robot, one must convert the robot to a point mass robot. For this sake, we present a two-step process for the conversion of the robot as follows:

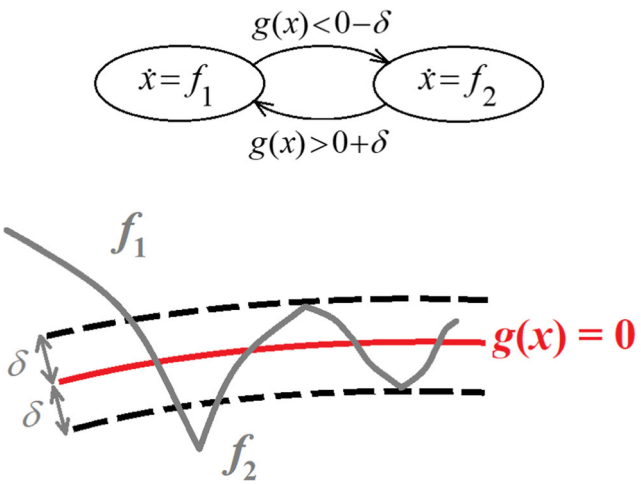

Fig. 11 The idea behind the fat guard. It is noted that δ and $g(x)$ are respectively a small positive value and the switching surface

At first, we transform the sophisticated hexapod robot to a simple unicycle robot with two inputs $[v, \omega]$ and 18 outputs $\left[\dot{\alpha}_i\ , \dot{\beta}_i\ , \dot{\gamma}_i\right]_{6\times 3}$ using the inverse kinematics presented in Section 2. In other words, we are now able to control the whole robot by merely controlling the longitudinal velocity (v) and the angular velocity (ω) of the robot's CoG. In order to make the CoG of the robot move with a given longitudinal velocity v and rotates with a certain angular velocity ω, it needs to travel with the mentioned velocity on a path with a curvature radius $R_{\mathrm{CoG}} = v/\omega$. Accordingly, each foot of the robot must touch the ground on a path with a particular curvature radius, which is elaborated in Fig. 12. In the figure, the black solid and the blue dashed lines are respectively associated with the path of the robot's CoG and the support phase trajectories of legs.

When the robot's CoG travels from position (1) and orientation θ_1 to position (2) and orientation θ_2 with given v and ω,

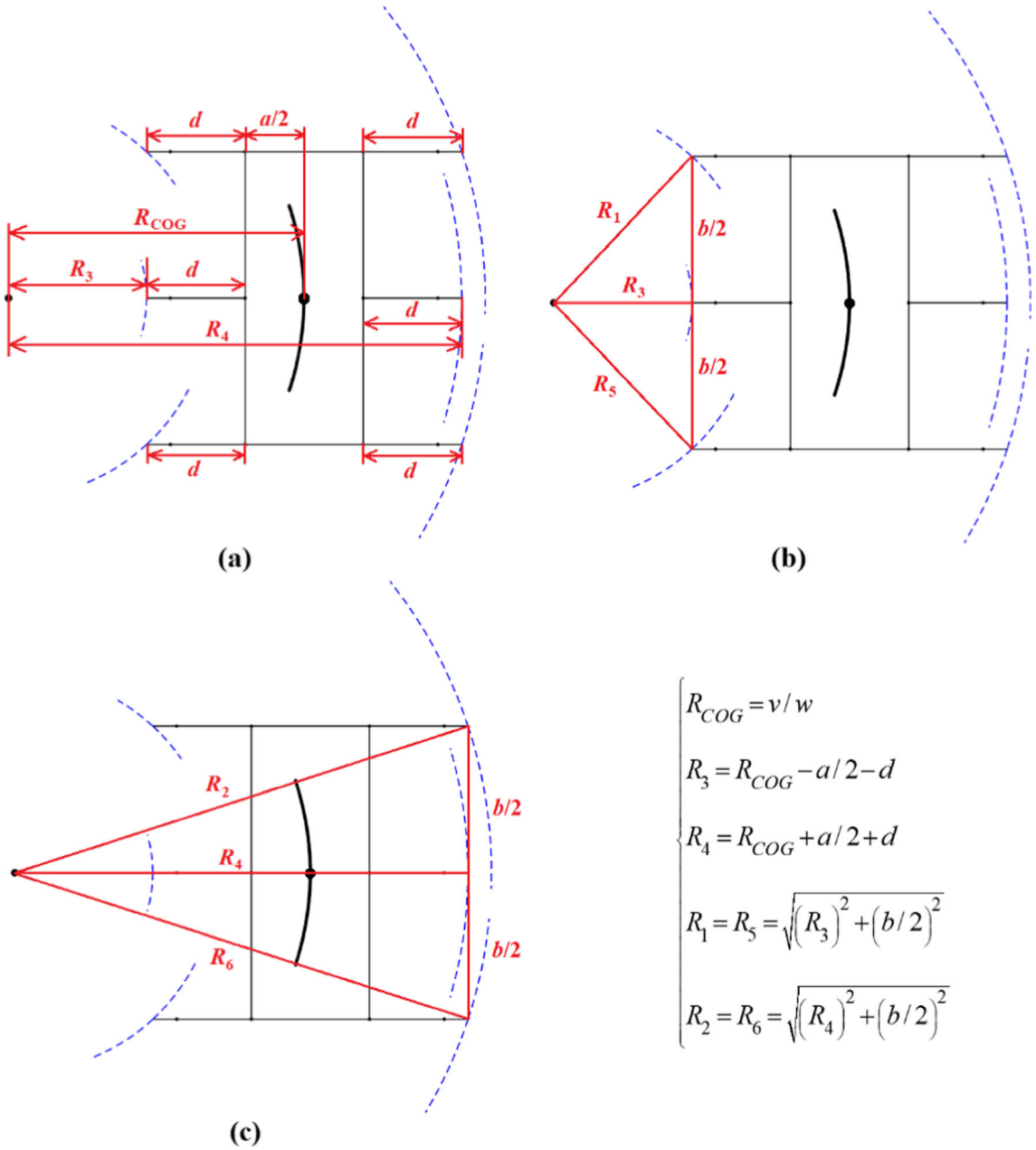

Fig. 12 Curvature radius calculations

the transformation matrix between the body coordinates $\sum_{B_1}$ to $\sum_{B_2}$ is achieved as:

$$T_{B_1}^{B_2} = \begin{bmatrix} \cos(\theta_2-\theta_1) & \sin(\theta_2-\theta_1) & 0 & -R_{CoG}(\cos(\theta_1)-\cos(\theta_2)) \\ -\sin(\theta_2-\theta_1) & \cos(\theta_2-\theta_1) & 0 & R_{CoG}(\sin(\theta_2)-\sin(\theta_1)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{20}$$

Using Eq. (20) and relations presented in Fig. 8, one can find the trajectory of the support phase as follows:

$$X_{Tip,2}^i = \begin{bmatrix} X_{Tip,2x}^i \\ X_{Tip,2y}^i \\ X_{Tip,2z}^i \end{bmatrix} = \begin{bmatrix} \cos(\theta_2^i-\theta_1^i)X_{Tip,1x}^i + \sin(\theta_2^i-\theta_1^i)X_{Tip,1y}^i - R_i(\cos(\theta_1^i)-\cos(\theta_2^i)) \\ -\sin(\theta_2^i-\theta_1^i)X_{Tip,1x}^i + \cos(\theta_2^i-\theta_1^i)X_{Tip,1y}^i + R_i(\sin(\theta_2^i)-\sin(\theta_1^i)) \\ X_{Tip,1z}^i \end{bmatrix} \tag{21}$$

In addition, the velocity terms are achieved by computing the first derivative of Eq. (21) as below:

$$V^i_{Tip,2} = \begin{bmatrix} V^i_{Tip,2x} \\ V^i_{Tip,2y} \\ V^i_{Tip,z} \end{bmatrix} = \begin{bmatrix} -(\omega^i_2-\omega^i_1)\sin(\theta^i_2-\theta^i_1)X^i_{Tip,1x} + (\omega^i_2-\omega^i_1)\cos(\theta^i_2-\theta^i_1)X^i_{Tip,1y} - R_i(\omega^i_2\sin(\theta^i_2)-\omega^i_1\sin(\theta^i_1)) \\ -(\omega^i_2-\omega^i_1)\cos(\theta^i_2-\theta^i_1)X^i_{Tip,1x} - (\omega^i_2-\omega^i_1)\sin(\theta^i_2-\theta^i_1)X^i_{Tip,1y} + R_i(\omega^i_2\cos(\theta^i_2)-\omega^i_1\cos(\theta^i_1)) \\ 0 \end{bmatrix} \tag{22}$$

Once, the support phase trajectory is obtained, the transfer phase trajectory is utilized to link the start and end points of the support phase trajectory. A quadratic polynomial is considered for the trajectory of the transfer phase to ensure a smooth trajectory in the air with no impact among the foot and ground [26]:

$$\begin{cases} x_{Transfer} = a_{0x} + a_{1x}t + a_{2x}t^2 + a_{3x}t^3 + a_{4x}t^4 \\ y_{Transfer} = a_{0y} + a_{1y}t + a_{2y}t^2 + a_{3y}t^3 + a_{4y}t^4 \\ z_{Transfer} = a_{0z} + a_{1z}t + a_{2z}t^2 + a_{3z}t^3 + a_{4z}t^4 \end{cases} \tag{23}$$

The coefficients of Eq. (23) are obtained by substituting the components of the predefined points that are the position and time of the start point, the position and time of the end point, the position and time of the middle point, the velocity of the start point, and the velocity of the end point.

Second, the simplified robot is converted to a point mass robot using the following trick [39]. We begin with introducing an imaginary point near the CoG of the robot as (Fig. 13):

$$\begin{aligned} \tilde{x} &= x - L\sin(\theta) \\ \tilde{y} &= y + L\cos(\theta) \end{aligned} \tag{24}$$

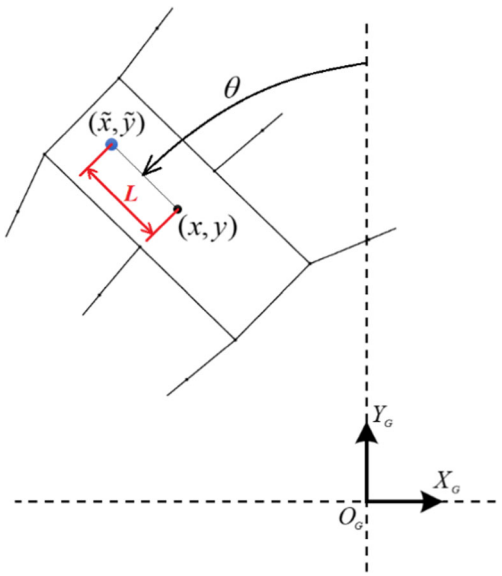

Fig. 13 The position of the tracking point

where L is the distance between the robot CoG and the imaginary point. Considering the velocities of the CoG of the robot in x and y directions respectively equates with $-v\sin(\theta)$ and $v\cos(\theta)$, the first derivative of Eq. (24) yields:

$$\begin{aligned} \dot{\tilde{x}} &= \dot{x} - L\dot{\theta}\cos(\theta) = -v\sin(\theta) - L\omega\cos(\theta) \\ \dot{\tilde{y}} &= \dot{y} - L\dot{\theta}\sin(\theta) = v\cos(\theta) - L\omega\sin(\theta) \end{aligned} \tag{25}$$

By further simplification of Eq. (25) we have:

$$\begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{y}} \end{bmatrix} = \begin{bmatrix} -\sin(\theta) & -\cos(\theta) \\ \cos(\theta) & -\sin(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & L \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{26}$$

We can also write v and ω as a function of the velocities of the introduced imaginary point as:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} -\sin(\theta) & \cos(\theta) \\ -\cos(\theta) & -\sin(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1/L \end{bmatrix} \begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{y}} \end{bmatrix} \tag{27}$$

Eq. (27) clearly illustrates the fact that the motion of the robot, which has been characterized by the longitudinal velocity v and angular velocity ω, can be easily controlled by controlling the motion of $(\tilde{x}, \tilde{y})$ point. In other words, the point might be considered as a tracking point for the robot or a virtual vehicle that the robot would follow.

Besides, Eq. (27) indicates that the angular velocity of the robot is inversely proportional to the value of the parameter L. For instance, choosing a very low value for L would lead to an enormous amount of ω which can be undesirable or even impossible for the actuators to perform. Also, very large values of L lead to an inaccurate tracking of the desired path. So, the value of L should be neither too large nor too small in comparison with the geometry of the robot. One can say that there is an optimal range for this variable, within which the motion of the robot is smoother for higher values of L; whereas, the robot would follow the generated path more precisely when L is small. Thus, the fact that how one defines the best value for L in the optimal interval depends on priorities in different conditions. It can be interpreted from the mentioned statements that one can easily control the

Table 2 Hybrid automaton parameters

	ζ	c	ε	λ	v_0 (m/s)	L (m)	d_O (m)	Goal
Sim. (I)	10,000	0.5	10	1	0.05	0.1	0.2	(0, 1.4)
Sim. (II)	10,000	0.4	10	1	0.04	0.1	0.25	(1, − 1.8)

smoothness of the motion of the robot or the path following precision by merely adjusting a single parameter (L).

It should be pointed out that one can utilize the aforementioned procedure to apply this navigation algorithm or any other planning methods to a complicated mobile robot. In fact, one should only solve the inverse kinematics of a robot, then this method can be applied. Plus, the other advantage of this method is that utilizing this approach, there is no need to employ a controller for controlling the orientation of the robot. In fact, according to Eq. (27), one can easily guide a mobile robot by merely controlling a reference point.

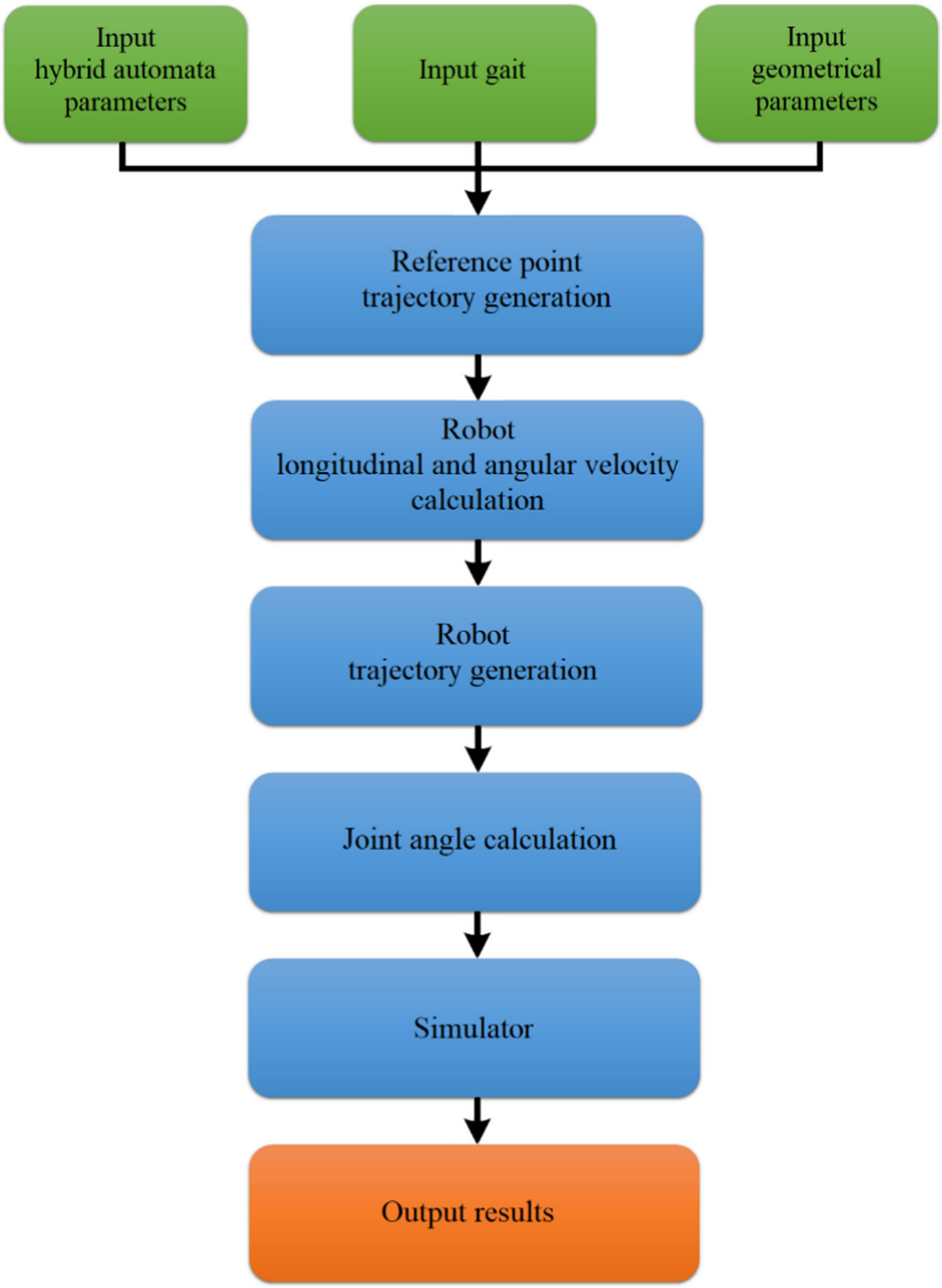

Fig. 14 Simulation flow chart

5 Simulations

In order to validate the effectiveness of the presented hybrid automaton in navigating and control of the studied hexapod robot, two different simulation tests are executed. The simulation parameters are comprised of the geometric parameters, hybrid automaton parameters, and the gait of the robot. The hybrid automaton parameters are considered as the tuning variables of the controller associated with each mode, the safety distance, the goal point, and the position of the reference point in the robot body coordinate system, which are displayed in Table 2.

The simulation procedure is a sequence of actions that take place one after the other as follows:

(1) Setting simulation parameters based on Tables 1 and 2 and the preferred gait.
(2) Imputing the mentioned parameters to the navigation system to generate the trajectory of the reference point.
(3) Calculating the longitudinal and angular velocities of the robot.
(4) Generating the trajectory of the robot's COG and legs.
(5) Obtaining joint angles by solving the inverse kinematics of the robot and supply the simulator with these values.
(6) Driving the robot to the goal point and output result. The simulation flow chart is depicted in Fig. 14.

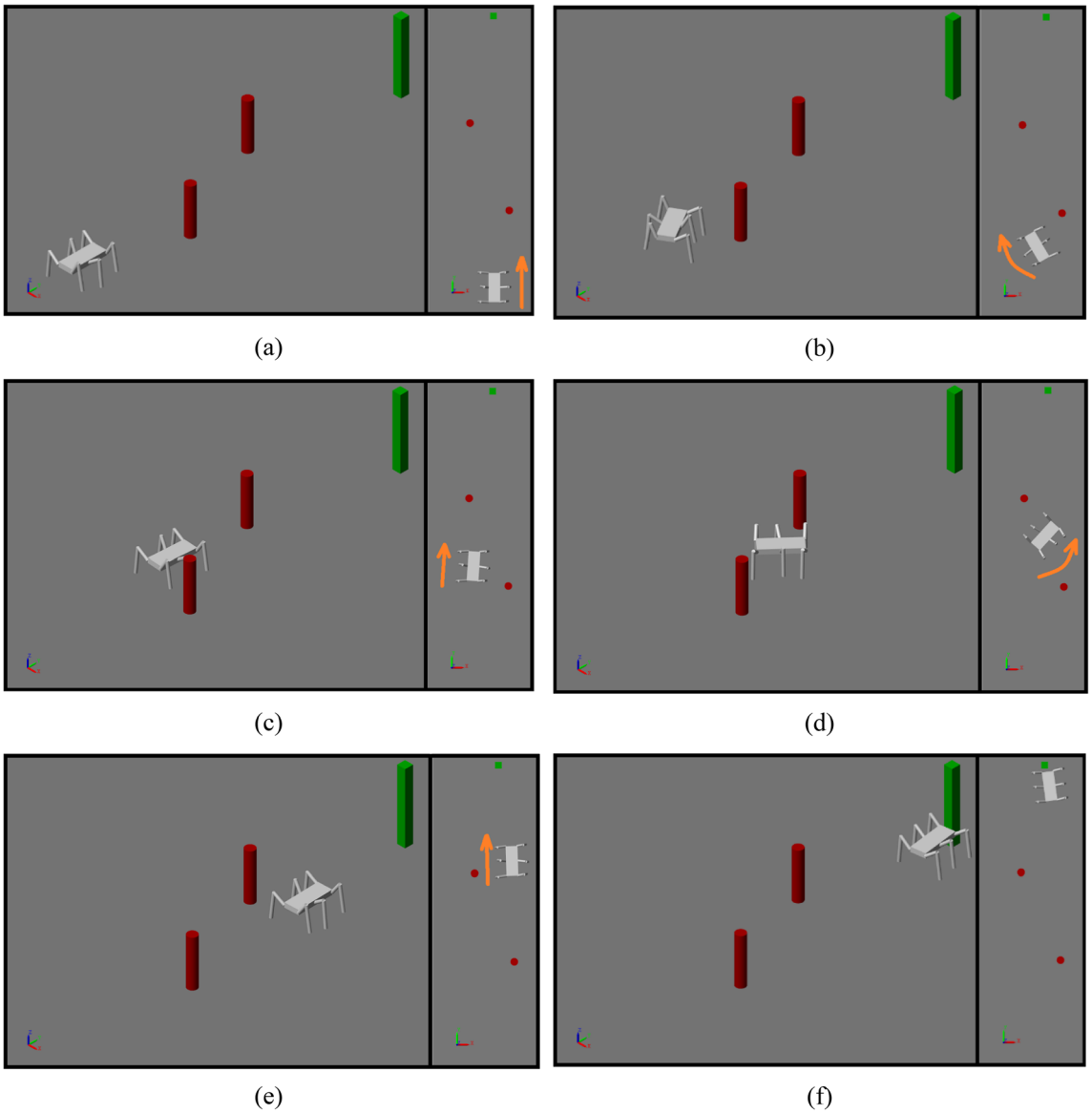

Fig. 15 The isometric and top view of the robot's motion in simulation (I), **a** Goal attraction, **b** Boundary following, **c** Goal attraction, **d** Boundary following, **e** Goal attraction, **f** Stop

As mentioned above, two simulation tests are performed in this section. In both cases, two point-obstacles are considered between the initial position of the robot and the target point, and the presented hybrid automaton undertakes the task of navigation. Also, the goal point and obstacles are respectively represented with green rectangular and red cylindrical shapes in the simulation environment.

It should be stated that the efficiency of the navigation algorithm in producing the optimum path and the performance of the robot in following the generated path does not rely on the gait pattern of the robot. In order to validate it, the simulations are based on different gait patterns. In the first simulation, the tripod gait ($\eta = 1/2$) is considered for the robot, and the movement of the robot is demonstrated in Fig. 15, which consists of the results of several positions during the motion.

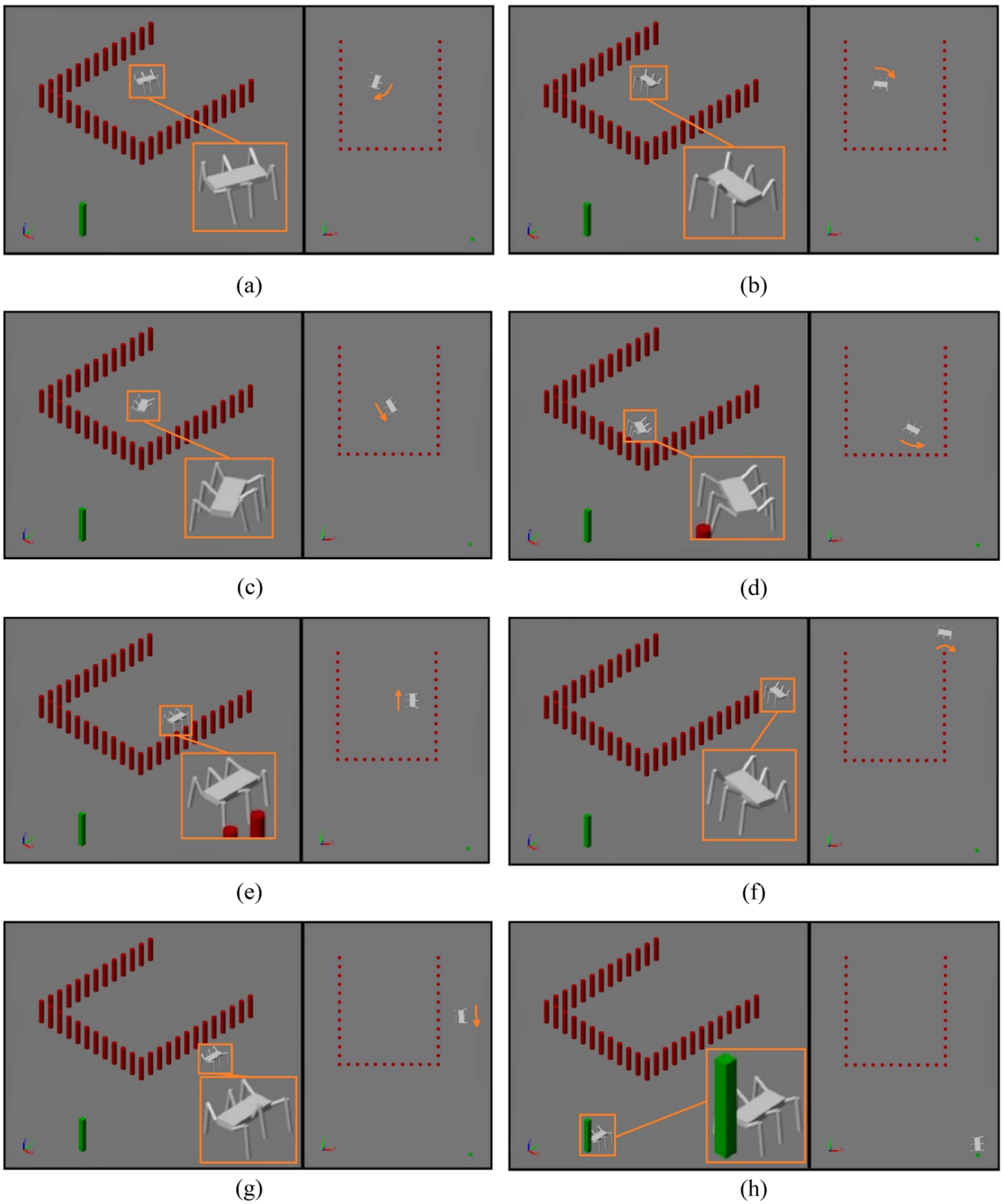

Fig. 16 The isometric and top view of the robot's motion in simulation (II), **a** Goal attraction, **b** Goal attraction, **c** Goal attraction, **d** Boundary following, **e** Boundary following, **f** Boundary following, **g** Goal attraction, **h** Stop

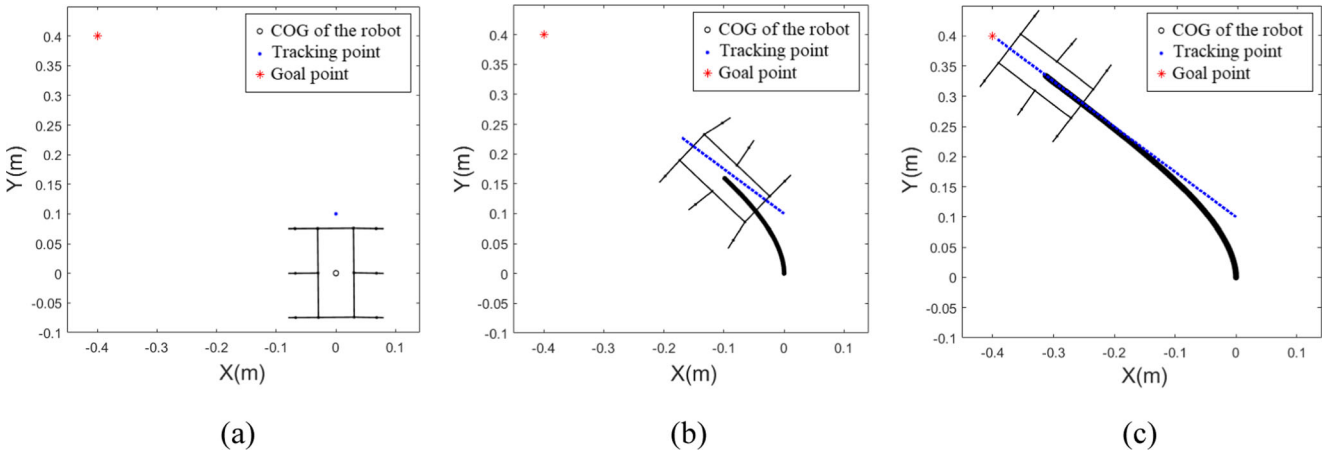

Fig. 17 The motion of the robot toward the goal point with $\zeta = 10{,}000$, $L = 0.1$ m, $v_0 = 0.03$ m/s, $\eta = 2/3$, and $T = 1$ s

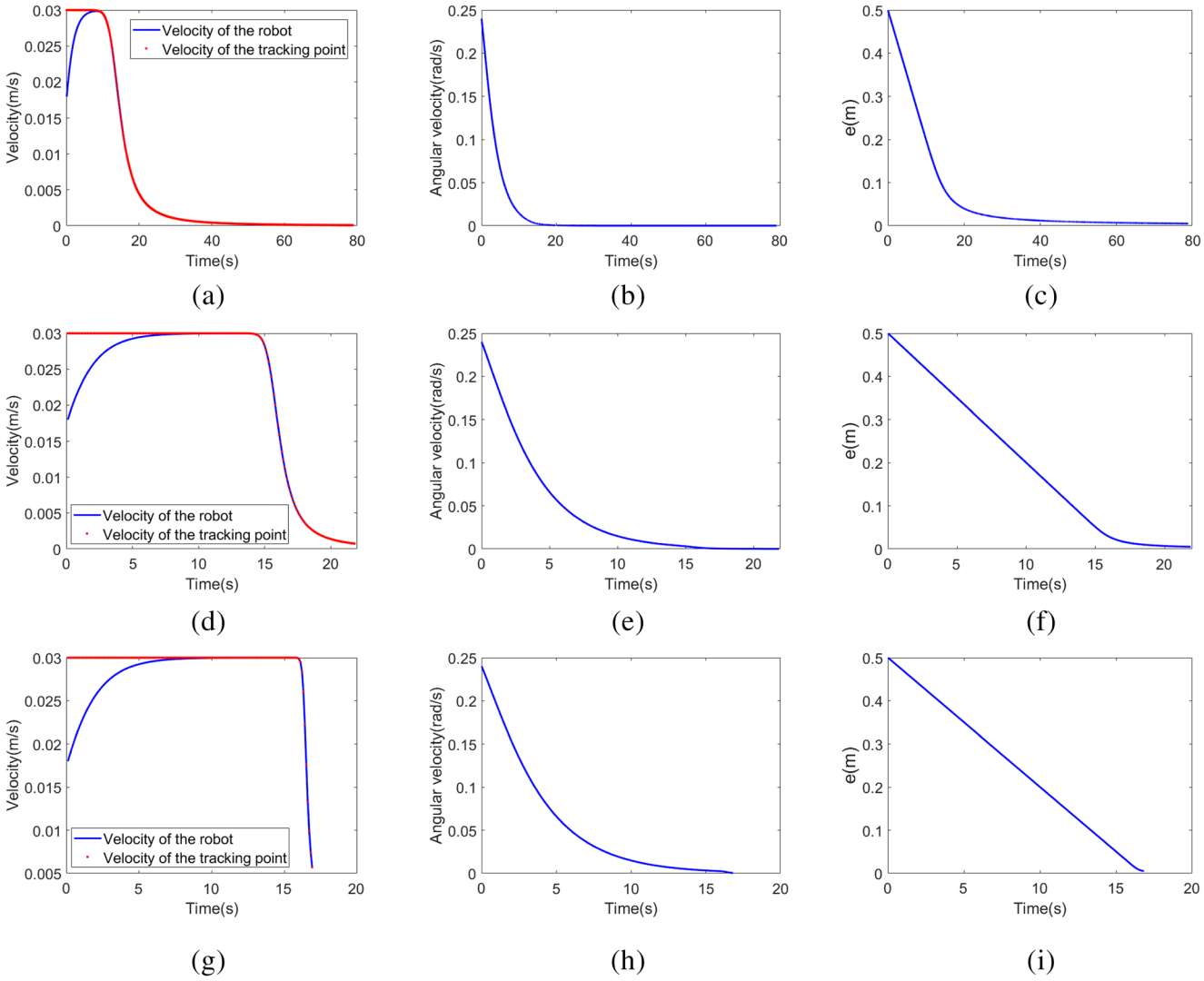

Fig. 18 The variation of the velocities (*left*), angular velocity (*middle*), and distance error (*right*) with respect to time

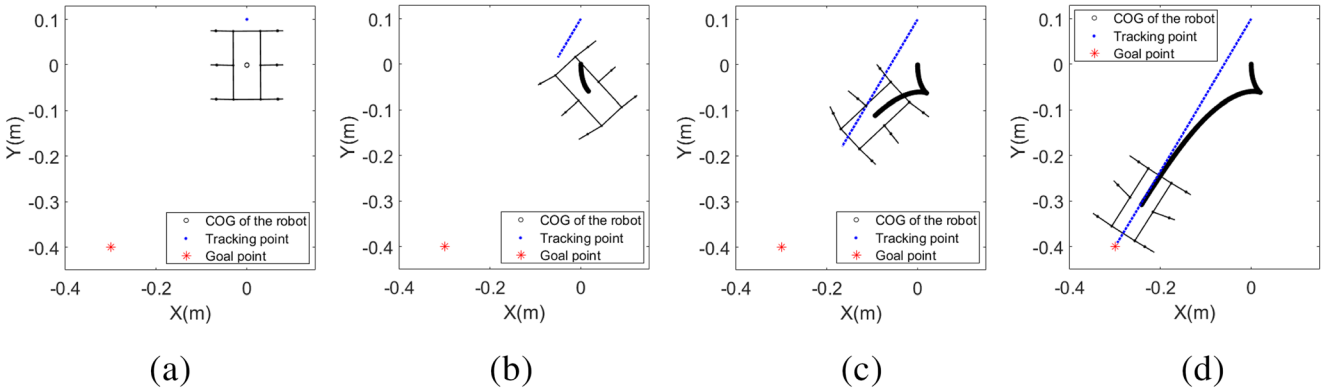

Fig. 19 The motion of the robot toward the goal point with $\zeta = 10{,}000$, $L = 0.1$ m, $v_0 = 0.03$ m/s, $\eta = 2/3$, and $T = 1$ s

The progressions of the simulation are as follows. At first, since the robot does not perceive any obstacle closer than the safety distance, the goal attraction behavior is activated, and the robot travels toward the target point (Fig. 15a). Then the first obstacle is detected, and the clockwise boundary following mode is activated according to the existing situation. Thus, the robot follows the boundary of the obstacle until the "progress" and "clear shot" conditions are checked (Fig. 15b). In this situation, the goal attraction mode gets activated again, and the robot drives to the goal until it perceives the second obstacle (Fig. 15c). After that, the counter-clockwise boundary following behavior is the strategy that the navigation algorithm comes up with to bypass the obstacle (Fig. 15d). Finally, the motion mode switches back to the goal attraction behavior, and the robot approaches the target point and eventually stops (Fig. 15e, f).

In the second simulation, the robot is placed inside a deadlock map with U-shaped obstacles and travels with the diagonal gait ($\eta = 2/3$). Contrary to the previous case, the goal is placed behind the robot (Fig. 16). As expected, the progress begins with a backward motion of the robot accompanied by a rotation to make the reference point stays in its given position in the body coordinate system. Then the robot go forward and track the path generated by the hybrid automaton in each circumstances to reach the target, similar to the preceding simulation.

The simulations indicate that the hybrid automaton can correctly switch between the different behaviors to generate the optimal path for the reference point in any circumstances, and the robot would track the reference point whatever the gait pattern is (noted that the gait pattern should be stable).

6 Results and Discussion

In the preceding sections, we presented a navigation algorithm for a point mass and illustrated how the algorithm can be implemented for the hexapod robot. The mentioned navigation strategy is applied on the studied robot to investigate the performance of the whole system. At first, the behavior of the robot in the goal attraction and obstacle avoidance modes are studied separately. Figure 17 illustrates how the robot would move toward the target point in goal attraction behavior. It is evident from the figure that the controller produces the shortest path, which is a straight line from the tracking point to the goal. For further investigation of the robot's behavior,

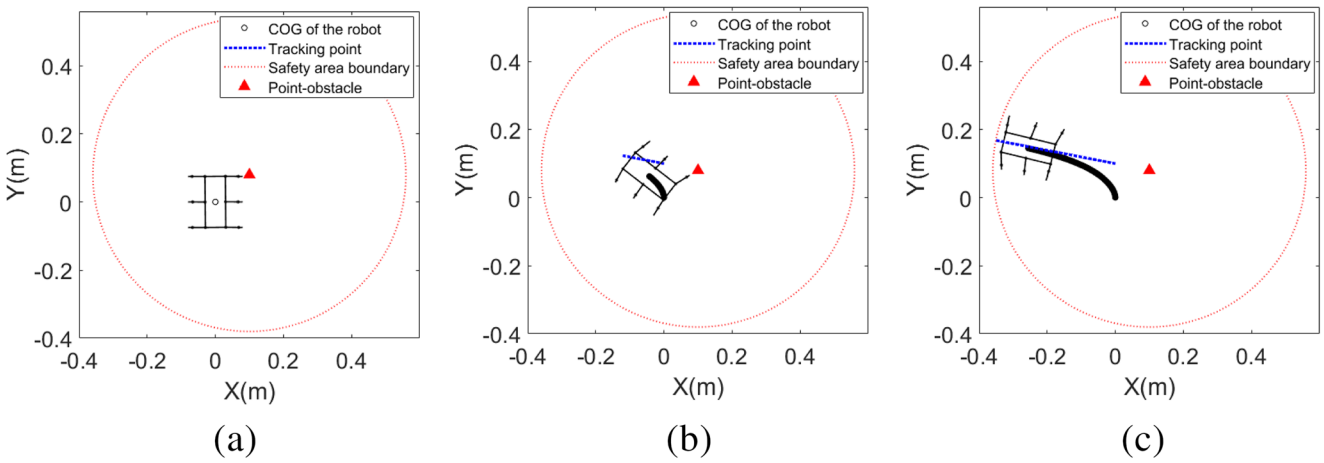

Fig. 20 The motion of the robot in the obstacle avoidance behavior with $c = 0.03$, $\varepsilon = 1$, $v_0 = 0.03$ m/s, $L = 0.1$ m, $\eta = 2/3$, and $T = 1$ s

the corresponding angular velocity (ω), distance error (e_{GA}), and the robot velocity (v), as well as the velocity of the tracking point as a function of time, are depicted in Fig. 18 for several values of ζ. At the beginning of the motion, when the robot's orientation is considerably deviant from the direction of the desired path (the mentioned straight line), the outcome of the controller would result in a more considerable value of ω compared to v. Therefore, an early rotation dominates the motion of the robot as already demonstrated in Fig. 17. Then with a gradual decrease in ω, v is increased to reach its critical value (v_0). Also, after getting adequately close to the goal, the robot begins to reduce its velocity. It is noted that the velocity reduction rate relies on the value of ζ. Although, for the lower values of ζ, the motion of the robot is smoother (without any sharp changes in the velocity) than that for higher ζ, but it takes a relatively longer time for the robot to reach the target point (Fig. 18).

It is noteworthy to mention that the robot shows an interesting motion when the target point is behind the robot (the angle between the robot's orientation and the vector stretching from the COG CoG of the robot to the goal point is greater than $\pi/2$), as exhibited in Fig. 19. In this situation, at first, the robot goes backward and simultaneously rotates, and then it moves forward to reach the target point. This behavior originates from the fact that the tracking point tends to move toward the goal point on a straight line, and this special motion of the robot is required to retain the desired position of the reference point in the body coordinate system. It should be noted that if there is no difference between the head and tail of the robot, one can assign a negative value to L when the goal is

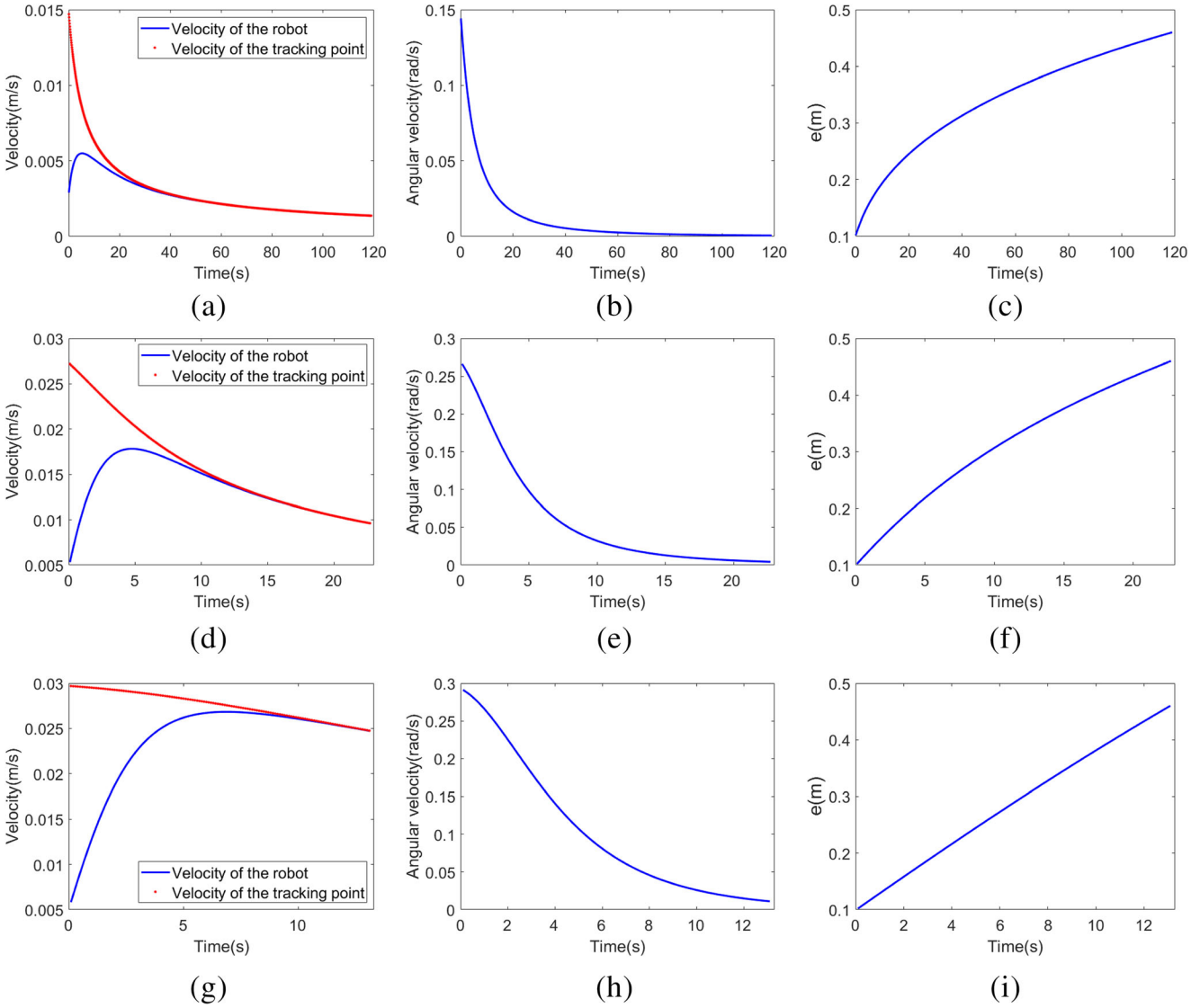

Fig. 21 The variation of the velocities (*left*), angular velocity (*middle*), and distance error (*right*) with respect to time

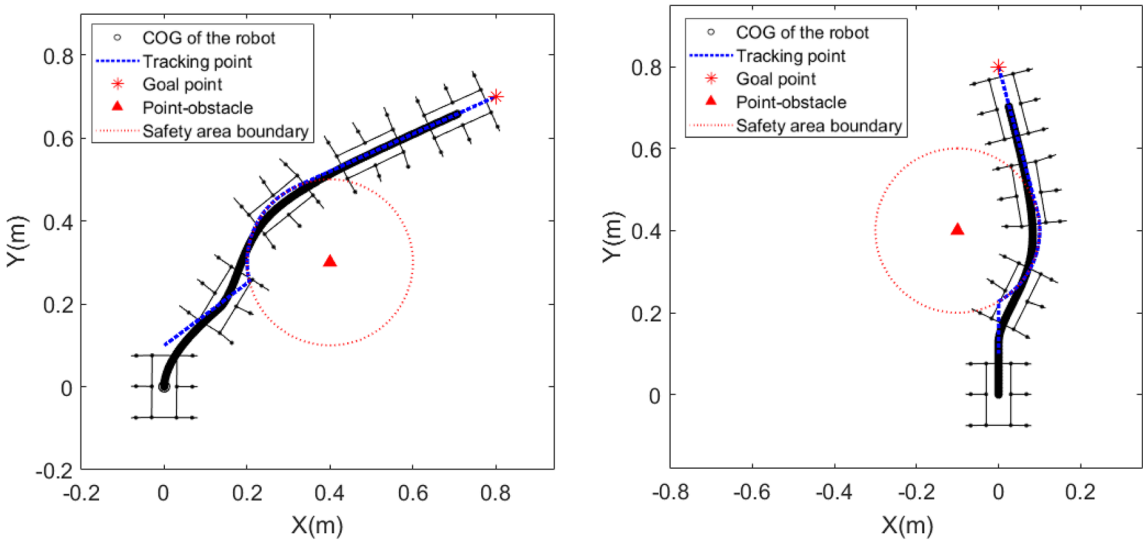

Fig. 22 The StroMotion of the robot controlled by the hybrid automaton in two different conditions with $\zeta = 10{,}000$, $v_0 = 0.03$ m/s, $c = 0.03$, $\varepsilon = 1$, $\lambda = 1$, $L = 0.1$ m, $\eta = 2/3$, $T = 1$ s and $d_O = 0.2$ m

behind the robot. Therefore, the robot can reach the target in a shorter time.

The generated path by the obstacle avoidance mode and the robot's behavior, when a point-obstacle is detected inside the safety area, are demonstrated in Fig. 20. The figure relies on the fact that the obstacle avoidance mode is switched off as the tracking point exits this area.

In order to further study the impacts of the corresponding controller's parameters on the robot's behavior, the robot velocity (v), angular velocity (ω), and distance error (e_{OA}) as a function of time are drawn in Fig. 21 for several values of c and ε. The figure shows that the velocity of the robot is reduced as the distance between the robot and obstacle is increased. Moreover, it can be perceived that the robot tends to escape from the obstacle with greater velocities for higher values of c and ε, although the motion of the robot is too slow when c and ε are small.

The behavior of the robot in each distinct mode has been studied above. Now, we examine the performance of the complete hybrid automaton and the hexapod robot. To this end, the behavior of the robot in two different situations are depicted in Fig. 22. The figure indicates that

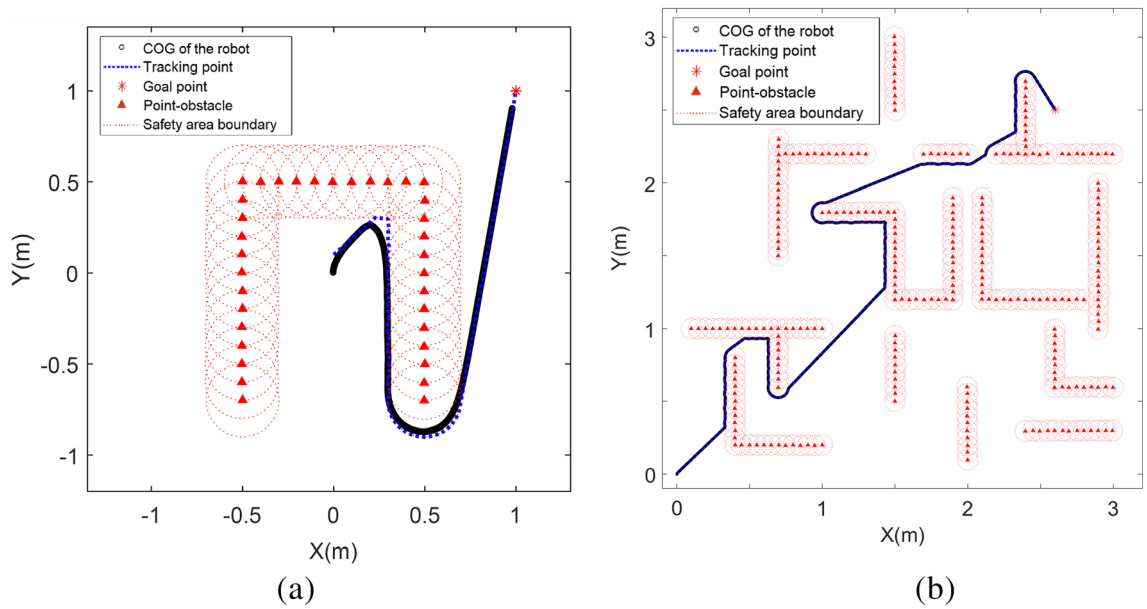

Fig. 23 The path of the robot controlled by the hybrid automaton in a deadlock (*top*) and a maze (*bottom*) map with $\zeta = 10{,}000$, $v_0 = 0.05$ m/s, $c = 0.05$, $\varepsilon = 1$, $\lambda = 1$, $L = 0.1$ m, $\eta = 1/2$, $T = 1$ s and $d_O = 0.2$

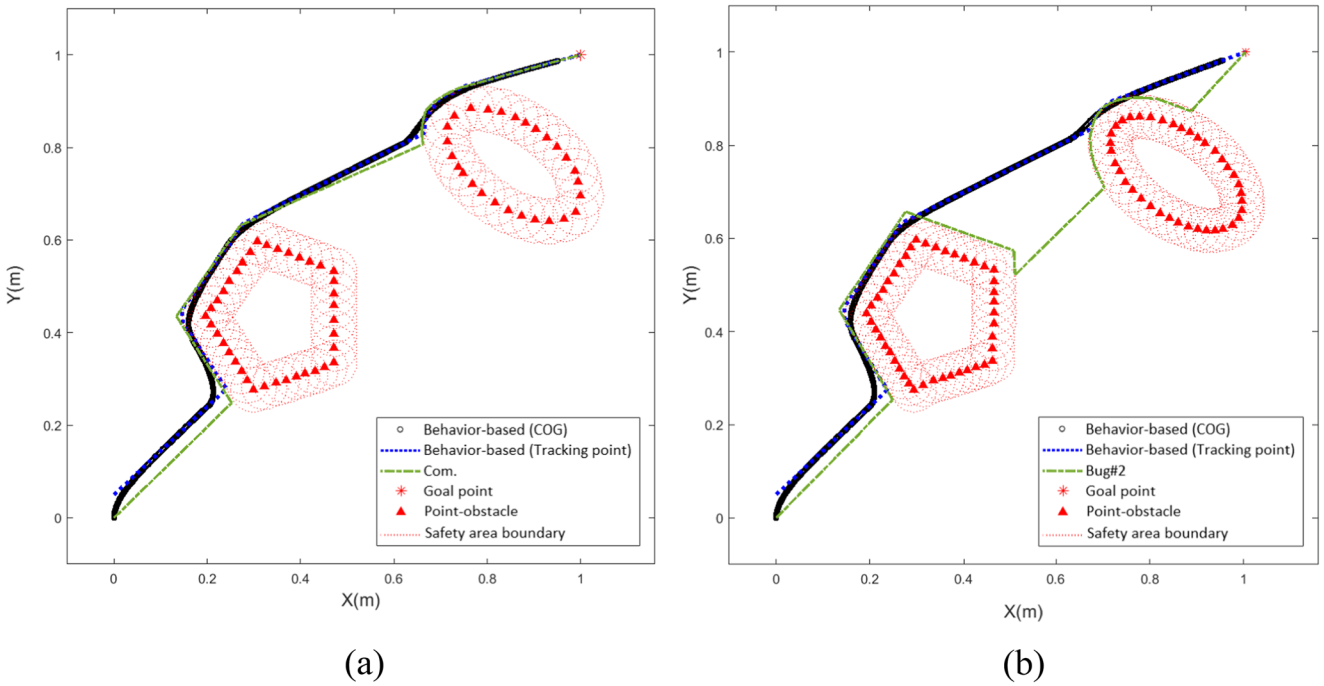

Fig. 24 The path produced by the presented hybrid automaton (behavior-based), *Com.* (*left*) and *Bug#2* (*right*) [40]

in both circumstances, an optimum path would be produced and perfectly tracked by the robot. The robot begins its motion by tracking the path generated by the goal attraction mode until an obstacle is perceived. Then one of the sliding modes (CW or CCW boundary following behaviors) gets activated on the basis of the noted criteria. In this situation, another path is generated to make the robot circle the obstacle until the required conditioned for switching back to the goal attraction behavior is met. Now, a new optimal path from the current position of the robot to the target point is produced and followed until the stopping condition is achieved.

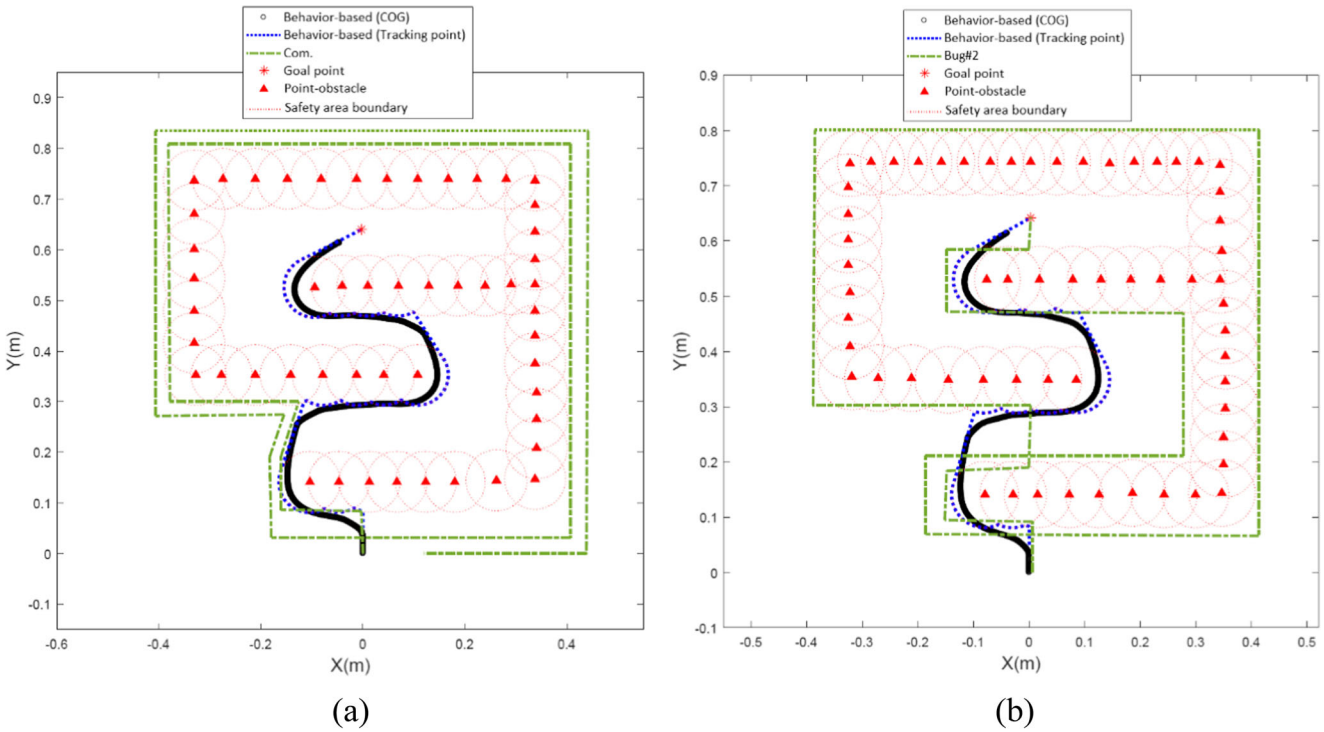

Fig. 25 The path produced by the presented hybrid automaton (behavior-based), *Com.* (*left*) and *Bug#2* (*right*) [40]

Moreover, the performance of the system in a deadlock map with U-shaped and a maze is investigated in Fig. 23. It is noted that the walls in both the U-shaped and maze maps are built of point obstacles with safety distance d_0. In Fig. 23a, the robot begins its motion from inside the U-shaped while the goal point is located outside the U-shaped. According to the figure, the robot would follow the boundary of the obstacles and reach the goal without getting stuck in the deadlock map. Similar to the deadlock map, the algorithm can perfectly handle the maze map by optimally switching between different behaviors as illustrated in Fig. 23b.

7 Comparative Study

This section concerns with the comparison between the behavior-based navigation and similar methods, such as the Bug and the potential field algorithms. To begin with, Fig. 24 presents the generated path by the "common sense Bug algorithm", abbreviated as *Com.*, the *Bug#2*, and the behavior-based method in the same setting. As demonstrated in the Fig. 24a, the trajectories generated by the behavior-based and the *Com.* method are fairly close. The robot starts moving toward the goal on a straight line until it perceives an obstacle; then it follows the boundary of the obstacle until the direction to the target is free [40, 41]. However, in the *bug#2* method, the scenario is somewhat different. In this approach, the robot would follow the boarder of the obstacle as long as it reaches the imaginary line, which is stretched from the start point to the goal, and is closer to the target point than the hit-point, where the robot meets an obstacle for the first time (Fig. 24b) [40, 41]. It is worth mentioning since the performance of the *Bug#1* method is relatively poor, we do not use this algorithm in this section. In fact, in the *Bug#1*, the robot has to discover every obstacle that meets by circling around it in order to come up with the closest position to the goal [40, 41].

Nonetheless, as shown in Fig. 25, the performance of the Bug algorithm is not satisfactory in complex maps. In other words, the *Com.* algorithm may not even reach the goal in some settings as illustrated in Fig. 25a. On the other hand, the presented behavior based strategy can handle complicated surroundings, and guide the robot toward the target point on a relatively short path.

It should be noted that the modified versions of the Bug algorithm, such as *DistBug*, *Rev1*, and *Rev2*, are able to cope with sophisticated environment by utilizing some techniques like remembering the former hit-points' distance to the goal [40]. However, the modified Bug algorithms are more computationally expensive compared to the presented simple behavior-based method.

Now, the performance of the presented planning algorithm is compared to that of the potential field approach. Figure 26 shows the generated path by both the behavior based and potential algorithm in two different maps. According to the

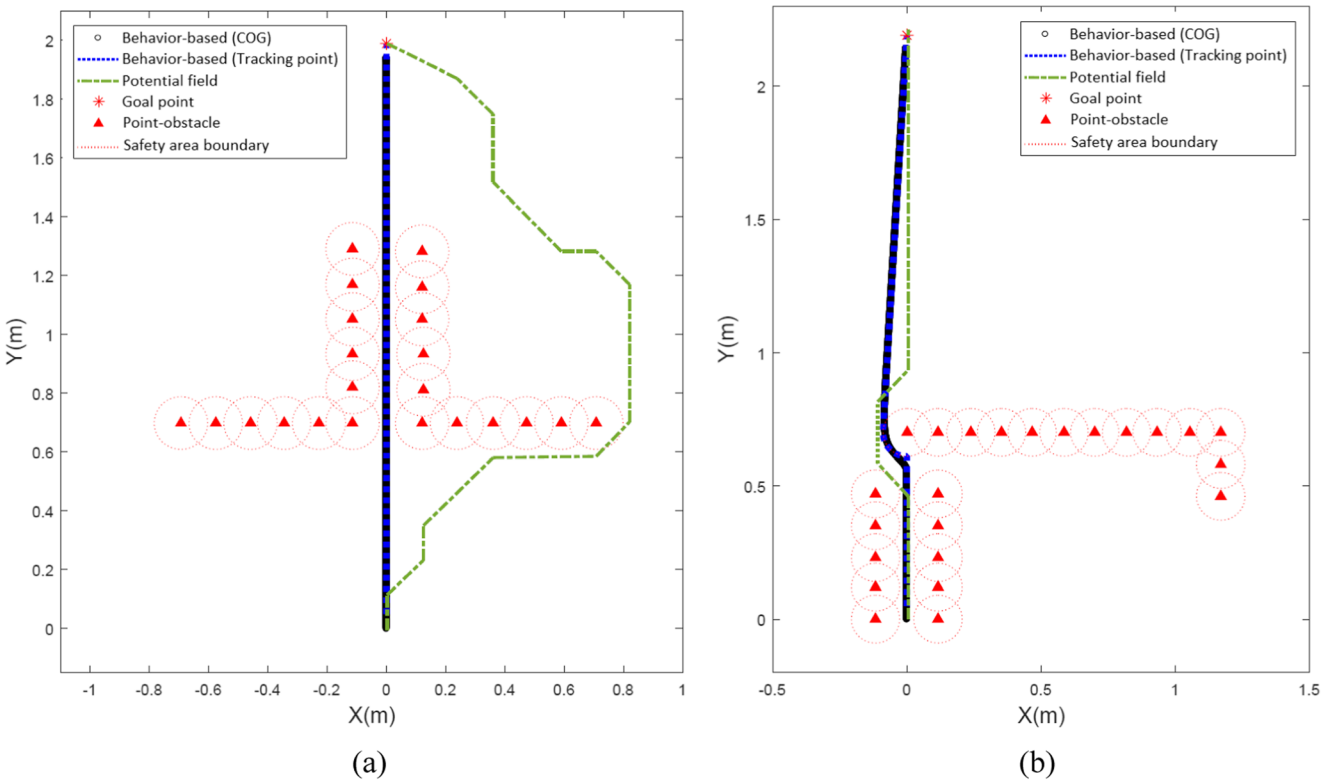

Fig. 26 The path produced by the presented hybrid automaton (behavior-based) and potential field [42]

Fig. 26a, the robot guided by the behavior-based technique would approach the target directly, while the path produced by the potential field method navigate around the obstacle to the goal [42]. As a result, even though both navigation strategies can fulfill the completeness requirement, the behavior-based method presents better properties over the other approach. This originates form the underlying limitation of the potential field planning algorithm in narrow passages. In other words, the repulsive potential of the obstacles would make the robot guided by the potential field method prevent entering narrow pathways. It is noteworthy to point out that both the potential field and behavior-based methods would generate almost similar paths when the robot begins its motion from the inside of a narrow passage as illustrated in Fig. 26b [42]. In fact, in this condition, the potential field algorithm finds the path through the narrow passage to have the lowest potential; whereas in other circumstances (Fig. 26a), the algorithm would possess other options to choose.

8 Conclusions

The major contribution of this research is the behavior-based navigation of an autonomous hexapod robot. It was demonstrated that the navigation algorithm could be represented by a hybrid automaton, in which each node is related to a different robot behavior. The nodes corresponding to the goal attraction and avoid obstacle behaviors were the primary nodes of the automaton, and the transitions between them were occurred using hard switches, which may cause a chattering phenomenon in the system. Thus, the automaton was modified by inserting additional nodes between the primary behaviors. The added nodes were the sliding dynamics of the system on the switching surface that were associated with the clockwise and counter-clockwise boundary following behaviors by which the robot could perfectly bypass obstacles. Plus, this behavior enabled the robot to perform well in the deadlock maps or mazes. Also, it was illustrated that the navigation algorithm could generate the shortest path, which was a straight line to the target point in the goal attraction behavior. Since the navigation algorithm was based on the trajectory generation for a reference point, the robot was converted to a point mass robot using two transformations. At first, the complex hexapod robot was converted to a unicycle robot, and then it was converted to a point mass robot. It was demonstrated that one can freely apply this navigation algorithm or any other planning methods to a complicated mobile robot using this procedure. Additionally, utilizing this approach, there is no need to employ a controller for controlling the orientation of the robot. In fact, as presented, one can easily guide a mobile robot by merely controlling a reference point. Moreover, this implementation methodology provided us with the opportunity to alter the smoothness of the motion of the robot or the path following precision by only adjusting a single parameter.

The effects of the controllers' parameters on the behavior of the robot were also investigated. It showed that there was a contradiction between the smoothness of the robot's motion and its accuracy in following the reference point as well as the reaction speed and the sensitivity of the robot to obstacles.

Simulation tests were executed to validate the performance of this approach. The results confirmed that the behavior-based control by the hybrid automaton would perfectly navigate the autonomous hexapod robot with and arbitrary stable gait pattern.

The comparative study illustrated that not only this navigation strategy can perfectly guide a mobile robot in various settings, but also it is cost efficient. While other planning algorithms that are introduced in the literature either have poor efficiency in complex surroundings or are computationally expensive.

The best future work of this study would be the experimental analysis of the system. In addition, one can also try to augment the navigation algorithm or the sensors in order to have a better perception of the environment and find the most optimal path for reaching the target point in complicated settings.

Appendix 1

$$J = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ 0 & J_{32} & J_{33} \end{bmatrix}$$

where,

$$J^i_{11} = (l_{3i}\cos(\gamma_i - \beta_i) + l_{2i}\cos(\beta_i) + l_{1i})\cos(\alpha_i),\ J^i_{12} = (l_{3i}\sin(\gamma_i - \beta_i) - l_{2i}\sin(\beta_i) + l_{1i})\sin(\alpha_i)$$
$$J^i_{13} = -(l_{3i}\sin(\gamma_i - \beta_i))\sin(\alpha_i),\ J^i_{21} = (l_{3i}\cos(\gamma_i - \beta_i) + l_{2i}\cos(\beta_i) + l_{1i})\sin(\alpha_i)$$
$$J^i_{22} = (l_{2i}\sin(\beta_i) - l_{3i}\sin(\gamma_i - \beta_i))\cos(\alpha_i),\ J^i_{23} = (l_{3i}\sin(\gamma_i - \beta_i))\cos(\alpha_i)$$
$$J^i_{32} = -l_{2i}\cos(\beta_i) - l_{3i}\cos(\gamma_i - \beta_i),\ J^i_{33} = l_{3i}\cos(\gamma_i - \beta_i)$$

Authors Contributions M. Khazaee: Methodology, investigation, writing – original draft, writing - review & editing. M. Sadedel: Supervision, project administration, conceptualization. A. Davarpanah: Formal analysis, visualization, software.

Availability of Data and Materials Owing to the fact that the data supporting the findings of this article is going to be utilized for future studies, participants of this research do not agree for their data to be shared publicly.

Funding The authors proclaim that they did not receive any funding for this Study.

Declarations

Ethical Approval Not applicable.

Consent to Participate Not applicable.

Consent to Publish Not applicable.

Competing Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Tariwala, S. Grid Navigation and Path Planning Algorithm Using a Proposed New Greedy Approach. Doctoral dissertation, Sciences (1914)
2. Karmore, S.L.P.S.: Design and Development of Sonar Based Autonomous Robot for Localization and Mapping for Potentially Unsafe Areas (1918)
3. Pruski, A., Rohmer, S.: Robust trajectory for Mobile robot. WIT Trans. Inf. Commun. Technol. **1** (1970)
4. Stentz, A., Hebert, M.: A complete navigation system for goal acquisition in unknown environments. Auton. Robot. **2**(2), 127–145 (1995)
5. Weisbin, C.R., de Saussure, G., Einstein, J.R., Pin, F.G., Heer, E.: Autonomous mobile robot navigation and learning. Computer. **22**(6), 29–35 (1989)
6. Lawitzky, G.: A navigation system for cleaning robots. Auton. Robot. **9**(3), 255–260 (2000)
7. Mathisen, S.G., Leira, F.S., Helgesen, H.H., Gryte, K., Johansen, T.A.: Autonomous ballistic airdrop of objects from a small fixed-wing unmanned aerial vehicle. Auton. Robot., 1-17 (2020)
8. Woosley, B., Dasgupta, P., Rogers, J.G., Twigg, J.: Multi-robot information driven path planning under communication constraints. Auton. Robot., 1–17 (2019)
9. Wheeler, D.O., Koch, D.P., Jackson, J.S., Ellingson, G.J., Nyholm, P.W., McLain, T.W., Beard, R.W.: Relative navigation of autonomous GPS-degraded micro air vehicles. Auton. Robot., 1-20 (2020)
10. Kortenkamp, D., Bonasso, R.P., Murphy, R.: Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems. MIT Press (1998)
11. Arkin, R.C., Arkin, R.C.: Behavior-Based Robotics. MIT Press (1998)
12. Egerstedt, M.: Behavior based robotics using hybrid automata. In: International Workshop on Hybrid Systems: Computation and Control, pp. 103-116. Springer, Berlin (2000)
13. Axelsson, H., Wardi, Y., Egerstedt, M., Verriest, E.I.: Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. J. Optim. Theory Appl. **136**(2), 167–186 (2008)
14. Reina, G., Foglia, M.: On the Mobility of all-Terrain Rovers. Ind. Robot. Int. J. (2013)
15. Nagatani, K., Noyori, T., Yoshida, K.: Development of multi-DOF tracked vehicle to traverse weak slope and climb up rough slope. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2849-2854. IEEE (2013)
16. Sadedel, M., Yousefi-Koma, A., Khadiv, M., Mansouri, S.: Investigation on dynamic modeling of SURENA III humanoid robot with heel-off and heel-strike motions. Iran. J. Sci. Technol. Trans. Mech. Eng. **41**(1), 9–23 (2017)
17. Chen, S.C., Huang, K.J., Chen, W.H., Shen, S.Y., Li, C.H., Lin, P.C.: Quattroped: a leg–wheel transformable robot. IEEE/ASME Trans. Mechatron. **19**(2), 730–742 (2013)
18. Sadedel, M., Yousefi Koma, A., Iranmanesh, F.: Heel-off and toe-off motions optimization for a2D humanoid robot equipped with active toe joints (2). Modares Mech. Eng. **16**(3), 87–97 (2016)
19. Agheli, M., Nestinger, S.S.: Force-based stability margin for multi-legged robots. Robot. Auton. Syst. **83**, 138–149 (2016)
20. Sadedel, M., Yousefi-Koma, A., Khadiv, M., Mahdavian, M.: Adding low-cost passive toe joints to the feet structure of SURENA III humanoid robot. Robotica. **35**(11), 2099–2121 (2017)
21. Sadedel, M., Yousefikoma, A., Iranmanesh, F.: Analytical dynamic modelling of heel-off and toe-off motions for a 2d humanoid robot. J. Comput. Appl. Mech. **46**(2), 243–256 (2015)
22. Zhong, G., Chen, L., Jiao, Z., Li, J., Deng, H.: Locomotion control and gait planning of a novel hexapod robot using biomimetic neurons. IEEE Trans. Control Syst. Technol. **26**(2), 624–636 (2017)
23. Manglik, A., Gupta, K., Bhanot, S.: Adaptive gait generation for hexapod robot using genetic algorithm. In: 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), pp. 1-6. IEEE (2016)
24. Khudher, D., Powell, R.: Quadratic programming for inverse kinematics control of a hexapod robot with inequality constraints. In: 2016 International Conference on Robotics: Current Trends and Future Challenges (RCTFC), pp. 1-5. IEEE (2016)
25. Gao, H., Liu, Y., Ding, L., Liu, G., Deng, Z., Liu, Y., Yu, H.: Low impact force and energy consumption motion planning for hexapod robot with passive compliant ankles. J. Intell. Robot. Syst. **94**(2), 349–370 (2019)
26. Zhu, Y., Guo, T., Liu, Q., Li, Q., Yan, R.: A study of arbitrary gait pattern generation for turning of a bio-inspired hexapod robot. Robot. Auton. Syst. **97**, 125–135 (2017)
27. He, B., Xu, S., Zhou, Y., Wang, Z.: Mobility properties analyses of a wall climbing hexapod robot. J. Mech. Sci. Technol. **32**(3), 1333–1344 (2018)
28. Stasse, O., Verrelst, B., Vanderborght, B., Yokoi, K.: Strategies for humanoid robots to dynamically walk over large obstacles. IEEE Trans. Robot. **25**(4), 960–967 (2009)
29. Michel, P., Chestnutt, J., Kuffner, J., Kanade, T.: Vision-guided humanoid footstep planning for dynamic environments. In: 5th IEEE-RAS International Conference on Humanoid Robots, pp. 13-18. IEEE (2005)
30. Havoutis, I., Ortiz, J., Bazeille, S., Barasuol, V., Semini, C., Caldwell, D.G.: Onboard perception-based trotting and crawling with the hydraulic quadruped robot (HyQ). In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 6052-6057. IEEE (2013)
31. Stelzer, A., Hirschmüller, H., Görner, M.: Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. Int. J. Robot. Res. **31**(4), 381–402 (2012)
32. Belter, D., Skrzypczyński, P.: Rough terrain mapping and classification for foothold selection in a walking robot. J. Field Robot. **28**(4), 497–528 (2011)
33. Agheli, M., Nestinger, S.S.: Inverse kinematics for arbitrary orientation of hexapod walking robots with 3-dof leg motion. In: 15th International Association of Science and Technology for Development (IASTED) Conference on Robotics and Applications (RA 2010), Cambridge, MA, Nov, pp. 1-3. (2010)
34. Bahrami, A., Tafaoli-Masoule, M., Bahrami, M.N.: Active vibration control of piezoelectric Stewart platform based on fuzzy control. Int. J. Mater. Mech. Eng. (IJMME). **2**(1), 17–22 (2013)
35. Åström, K.J.: Theory and applications of adaptive control—a survey. Automatica. **19**(5), 471–486 (1983)

36. Van, M., Mavrovouniotis, M., Ge, S.S.: An adaptive backstepping nonsingular fast terminal sliding mode control for robust fault tolerant control of robot manipulators. IEEE Trans. Syst. Man Cybern. Syst. **49**(7), 1448–1458 (2018)
37. Filippov, A.F.: Equations with the Right-Hand Side Continuous in x and Discontinuous in T. In: Differential equations with discontinuous righthand sides, pp. 3–47. Springer, Dordrecht (1988).
38. Egerstedt, M., Johansson, K., Lygeros, J., Sastry, S.: Behavior based robotics using regularized hybrid automata. In: Proceedings of the 38th IEEE conference on decision and control (Cat. No. 99CH36304), vol. 4, pp. 3400-3405. IEEE (1999)
39. Egerstedt, M.: Control of Mobile Robots. Retrieved from https://www.coursera.org/learn/mobile-robot (n.d.)
40. McGuire, K.N., de Croon, G.C.H.E., Tuyls, K.: A comparative study of bug algorithms for robot navigation. Robot. Auton. Syst. **121**, 103261 (2019)
41. Lumelsky, V., Stepanov, A.: Dynamic path planning for a mobile automaton with limited information on the environment. IEEE Trans. Autom. Control. **31**(11), 1058–1063 (1986)
42. Chen, Y., Liang, J., Wang, Y., Pan, Q., Tan, J., Mao, J.: Autonomous mobile robot path planning in unknown dynamic environments using neural dynamics. Soft. Comput. **24**(18), 13979–13995 (2020)

Mostafa Khazaee received his B.S. degree in Mechanical Engineering from Qom University of Technology, Qom, Iran in 2015 and M.S. degree in Applied Mechanics from Tarbiat Modares University, Tehran, Iran, in 2018. Since 2018, he has been working in the department of Mechanical Engineering at Tarbiat Modares University, Tehran, Iran as a research assistant. His research interests include autonomous mobile robots, dynamical systems, machine learning, and artificial intelligence.

Majid Sadedel received his B.S. degree from Amirkabir University of Technology, formerly called the Tehran Polytechnic, Tehran, Iran, in 2009, his M.S. degree from Sharif University of Technology, Tehran, Iran, in 2011, and his Ph.D. degree from Tehran University, Tehran, Iran, in 2016, all in Mechanical Engineering. Since 2017, he has been with the department of Mechanical Engineering at Tarbiat Modares University, Tehran, Iran, where he is currently an Assistant Professor. His research interests include dynamic modeling, motion planning, gait optimization and stability control of humanoid robots.

Atoosa Davarpanah received her B.S. degree in Mechanical Engineering from Shiraz University, Shiraz, Iran in 2013 and M.S. degree in Applied Mechanics from Shiraz University of Technology, Shiraz, Iran, in 2016. Since 2016, she studies in Tarbiat Modares University, Tehran, Iran for Ph.D. degree in Mechanical Engineering. Her research interests include, dynamic modeling, trajectory planning, and optimization of soft robots.