SCIENCE CHINA
Technological Sciences

Special Topic: Mechanical Engineering
• Article •

June 2017 Vol.60 No.6: 818–831
doi: 10.1007/s11431-016-9017-6

Obstacle avoidance for a hexapod robot in unknown environment

CHAI Xun, GAO Feng*, QI ChenKun, PAN Yang, XU YiLin & ZHAO Yue

State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China;

Received July 29, 2016; accepted February 20, 2017; published online April 25, 2017

Obstacle avoidance is quite an important issue in the field of legged robotic applications, such as rescuing and detecting in complicated environment. Most related researchers focused on the legged robot's gait generation after assuming that obstacles have been detected and the walking path has been given. In this paper we propose and validate a novel obstacle avoidance framework for a six-legged walking robot Hexapod-III in unknown environment. Throughout the paper we highlight three themes: (1) The terrain map modeling and the obstacle detection; (2) the obstacle avoidance path planning method; (3) motion planning for the legged robot. Concretely, a novel geometric feature grid map (GFGM) is proposed to describe the terrain. Based on the GFGM, the obstacle detection algorithm is presented. Then the concepts of virtual obstacles and safe conversion pose are introduced. Virtual obstacles restrict the robot to walk on the detection terrain. A safe path based on Bezier curves, passing through safe conversion poses, is obtained by minimizing a penalty function taking into account the path length subjected to obstacle avoidance. Thirdly, motion planning for the legged robot to walk along the generated path is discussed in detail. At last, we apply the proposed framework to the Hexapod-III robot. The experimental result shows that our methodology allows the robot to walk safely without encountering with any obstacles in unknown environment.

obstacle avoidance, hexapod robot, terrain map building, path planning, motion planning

Citation: Chai X, Gao F, Qi C K, et al. Obstacle avoidance for a hexapod robot in unknown environment. Sci China Tech Sci, 2017, 60: 818–831, doi: 10.1007/s11431-016-9017-6

1 Introduction

Recently, research on autonomous legged robots [1–5] has gained more and more importance. The foremost advantage of legged robots is that they only need some discrete footholds for locomotion on the ground. In applications such as detecting and rescuing in harsh environment, legged robots show superior performance to wheeled/tracked robots [6,7]. When encountering with an obstacle, legged robots can choose to step on, step over or bypass the obstacle. Most related works are concentrated on the motion generation of the feet and body in order to step on or step over the obstacle. In real application environment, bypassing large obstacles is a fairly common phenomenon for legged robots, while few papers involve the obstacle avoidance approach for legged robots.

Distinguishing the obstacles in unknown environment is a prerequisite for obstacle avoidance. This problem is addressed by terrain map building and obstacle detection. To date, most robots use visual information from a stereo camera [8,9] or distance information from a LIDAR [10,11] (light detection and ranging) sensor to build a terrain map. In order to decrease the data storage space and the computational effort, the point cloud, obtained from the stereo camera or the LIDAR sensor, needs to be converted to a grid discretized map. For example, Kagami et al. [12] introduced a discretized height map where each cell holds the terrain height. Pfaff and Burgard [13] developed a grid-sized map that can represent overhanging objects. An irregular triangular mesh (ITM) is used to represent the terrain by Rekleitis et al [14].

*Corresponding author (email: fengg@sjtu.edu.cn; gaofengsjtu@gmail.com)

Ishigami et al. [15] proposed a range-dependent terrain mapping method, which is based on the sector-shaped cell represented by the cylindrical coordinates.

The geometric feature of the terrain benefits the obstacle detection a lot. However, few works involve building a map which contains detailed geometric features of the partial terrain. The most related and recent work is that of Gutmann et al. [16]. They presented a floor and obstacle height map of the robot surroundings. Each cell in the map is marked as floor or obstacle based on the cell height. However, the whole obstacle cannot be detected and located in their work. The first research target of ours is the same with theirs. While in our paper, a novel grid terrain map model, which contains detailed geometric features of the terrain, is proposed. What's more, the problem of the obstacle detection is solved.

In the real world, both obstacles and robots have complex shapes. If their true shapes are modeled, large amounts of data are needed and the distance computation of them is complicated, which affects the application of the obstacle avoidance method. Generally, some regular geometric models are used to describe obstacles and robots, such as cylinder [17], cuboid [18], elliptic cylinder [19] and so on. In such a way, the data amount is decreased, and the spent time and space for the distance computation is reduced too. In this paper, the cylinder is chosen to describe the objects and facilitate the distance computation. While other typical geometric models are also applicable for the following proposed obstacle avoidance theory.

Once the position and the size of the obstacle are available, a safe obstacle-free path needs to be planned for the legged robots. In the related research fields, the obstacle avoidance path planning for wheeled robots has been studied extensively. Cherubini and Chaumette [20] proposed and validated a framework for visual navigation with collision avoidance for a wheeled mobile robot. Their method guarantees that obstacle avoidance and navigation can be achieved simultaneously. Parhi and Singh [21] proposed an intelligent fuzzy interface technique for the obstacle avoidance control of an autonomous mobile robot. Škrjanc and Klančar [22] presented an obstacle avoidance method for multiple wheeled robots based on Bezier curves, where the velocities and accelerations of the mobile robots are both constrained. Jolly et al. [23] proposed an efficient Bezier curve based approach for the path planning of a wheeled robot in a multi-agent robot soccer system without violating the acceleration limits. Uchiyama et al. [24] proposed a simple obstacle avoidance control approach for a human-operated four-wheeled mobile robot. In their approach, the control input of the human operator is modified in real time to satisfy the nonholonomic constraint and collision avoidance with obstacles. Prassler [25] presented the hardware and control design of the navigation system of a wheelchair robot. Their navigation system enables the robot to move through narrow cluttered and partially unknown environments. However, little researches about the obstacle avoidance for the legged robot has been done.

In this paper, we focus on the obstacle avoidance for a six-legged walking robot. The main contributions of this paper are:

(1) The novel geometric feature grid map (GFGM) is proposed, each cell in the map stores typical geometric features of the partial terrain. Based on the GFGM, the obstacle detection and location algorithm is presented.

(2) The concept of virtual obstacles is introduced to restrict the robot to walk on the detection terrain. The safe conversion pose is defined for the robot to pass through.

(3) The obstacle avoidance path planning method based on Bezier curves is proposed. The method is formulated as an optimization model taking into account the path length subjected to obstacle avoidance.

(4) Motion planning for the six-legged robot is discussed.

(5) We integrate the proposed framework into the Hexapod-III robot. The experiment is carried out and the result validates the theory successfully.

The remainder of this paper is organized as follows: Section 2 provides a brief introduction to the robot system and describes the problem formulation. The GFGM, the obstacle detection algorithm are presented in Section 3. In Section 4, the concept of virtual obstacles and the safe conversion pose are introduced. The obstacle-free path planning method based on Bezier curves is proposed. Section 5 discusses the motion planning method for the six-legged robot. Section 6 describes the experiment and discusses the result. Section 7 summarizes and concludes the paper

2 Robot system overview

As Figure 1 shows, the Hexapod-III robot is a six-DOFS parallel-parallel moving platform integrated walking and manipulating. It is mainly designed for executing emergent tasks, such as detecting and rescuing, in harsh environment. Following, a brief overview of the robot system is given.

Hexapod-III has a long hexagonal body with six similar legs symmetrically distributed. Every leg is a parallel mechanism constructed by three chains as Figure 2 shows. The $U_2P_2S_2$ chain and the $U_3P_3S_3$ chain have the same mechanism structure and dimension size, each one is constructed by a universal joint, a prismatic joint and a spherical joint. Another chain, the U_1P_1 chain constructed by a universal joint and a prismatic joint, is mainly used to bear the external force. The linear movement of the prismatic joint is achieved by the ball-bearing screw, which is actuated by the motor. Every leg has 3 control inputs, which are the lengths of the three prismatic joints. So Hexapod-III has 18 control inputs in total, which can be used to control the body pose and six feet's positions based on its kinematics model. The special parallel-

Figure 1 (Color online) The Hexapod-III robot.

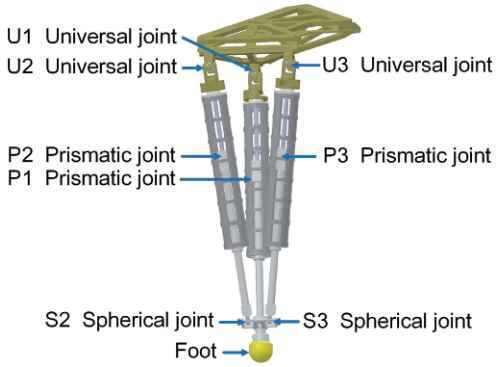

Figure 2 (Color online) The leg mechanical design.

parallel mechanism design concept not only fulfills the DOF requirement but also provides Hexapod-III with outstanding load capacity, walking stability and terrain adaption.

As Figure 1 shows, multiple sensors have been integrated into Hexapod-III, including a stereo camera, a six-axis force/torque sensor, a compass and an IMU. The stereo camera detects the terrain in front of the robot. The six-axis force/torque sensor measures all six components of force and torque of the body. The compass helps the robot navigate the right direction in outdoor environment. The IMU can measure the inclination, the angle velocity and the linear acceleration of the robot.

This paper proposes a novel obstacle avoidance framework for a six-legged walking robot in unknown environment. Legged robots are often used for rescuing and detecting in disaster environment. In such application environment, they need to walk safely without encountering with any obstacles. As shown in Figure 3, in order to express the problem clearly and simply, the ground coordinate system (GCS) is represented by O_G, the robot coordinate system (RCS) is represented by O_R, the camera coordinate system (CCS) is represented by O_C, and the feet coordinate systems (FCS) are represented by $O_{F_1}, \ldots, O_{F_6}$ respectively.

Considering the issue of obstacle avoidance for a legged robot, the robot must have a good knowledge of the surroundings in advance. The stereo camera can obtain the point cloud of the environment. In order to distinguish the obstacles and the normal ground, the point cloud needs to be processed and analyzed. After that, the robot needs find a safe obstacle-free path in the GCS. Concretely, during the whole walking process the distance between the robot and obstacles should always be larger than the minimum safety margin. In general, the robot and obstacles often have complex shapes, which makes it difficult and time-consuming to calculate the distance between them. So some regular geometric models should better be used to represent them effectively. At last, the locomotion of six feet $O_{F_1}, \ldots, O_{F_6}$ needs to be computed to actuate the robot to walk along the safe path.

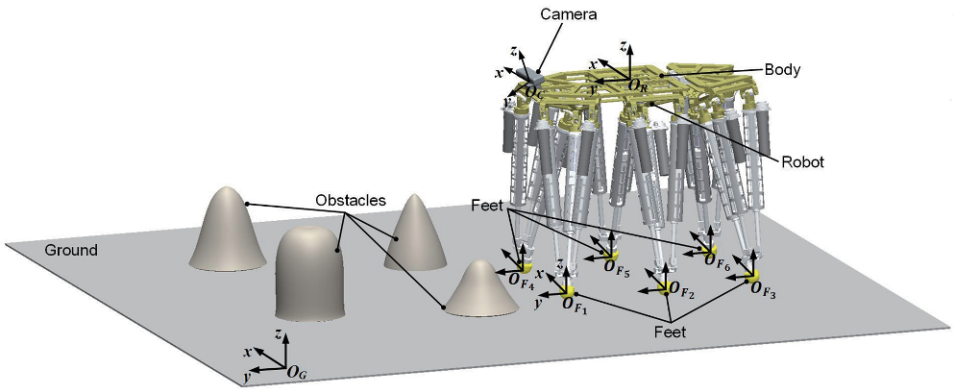

Figure 3 (Color online) Schematic representation of the problem formulation.

3 The geometric feature grid map (GFGM) modeling and the obstacle detection

3.1 The geometric feature grid map (GFGM) modeling

The GFGM describes the terrain using multiple square-shaped grids, and each grid stores the geometric feature of the current object. The GFGM does not need much data storage space and is easy to update at a high rate. The size of the grid is determined by the specific application environment.

As Figure 4 shows, the stereo camera provides the depth image of the environment, and the point cloud $\boldsymbol{P}^C(x^C, y^C, z^C)$ with respect to the CCS is obtained from the depth image. The GFGM is created from the point cloud $\boldsymbol{P}^G(x^G, y^G, z^G)$ in the GCS. $\boldsymbol{P}^G$ can be transformed from $\boldsymbol{P}^C$ using the following equation:

$$\boldsymbol{P}^G = \boldsymbol{T}_R^G \cdot \boldsymbol{T}_C^R \cdot \boldsymbol{P}^C, \tag{1}$$

where $\boldsymbol{T}_C^R$ is the identification matrix between the CCS and the RCS. $\boldsymbol{T}_C^R$ provides the pose information of the CCS relative to the RCS. The methodology of calculating $\boldsymbol{T}_C^R$ has been proposed in our previous work [26]. $\boldsymbol{T}_R^G$ is the transformation matrix providing the pose information of the *RCS* relative to the *GCS*. $\boldsymbol{T}_R^G$ is obtained from the IMU information and the kinematics model of the robot [27,28].

As Figure 5 shows, generally assuming that the size of GFGM is $m \times n$ (m, n are integers), and the grid size is $a \times a$. The size of the terrain represented by the GFGM is $m \cdot a$ long and $n \cdot a$ wide. First of all, the point cloud needs to be mapped into the corresponding grid. $\boldsymbol{P}_i^G$ belongs to grid$^{[p,q]}$ based on the following equation:

$$\begin{aligned} \text{grid}^{[p,q]} = \{\boldsymbol{P}_i^G \mid & (p-1)a \le x_i^G \le pa, \\ & (q-1)a \le y_i^G \le qa, i = 1, \dots k\}, \end{aligned} \tag{2}$$

where k is the number of the point cloud belonging to grid$^{[p,q]}$. The height of grid$^{[p,q]}$ is denoted by $h^{[p,q]}$, which represents the average height of the partial terrain. $h^{[p,q]}$ is calculated as following:

$$h^{[p,q]} = \frac{\sum_{i=1}^{k} z_i^G}{k}. \tag{3}$$

Besides the height, other two indexes are defined to describe the geometric feature of the partial terrain. One is the terrain flatness denoted by $f^{[p,q]}$, the other is the terrain orientation denoted by $o^{[p,q]}$. We propose to use a plane in the grid to approximately describe the surface of the partial terrain. The plane in grid$^{[p,q]}$ is denoted by plane$^{[p,q]}$, which is estimated using the method in our previous work [26]. plane$^{[p,q]}$ is estimated from the point cloud in grid$^{[p,q]}$ and is expressed by the following equation:

$$a^{[p,q]}x^{[p,q]} + b^{[p,q]}y^{[p,q]} + c^{[p,q]}z^{[p,q]} = d^{[p,q]}. \tag{4}$$

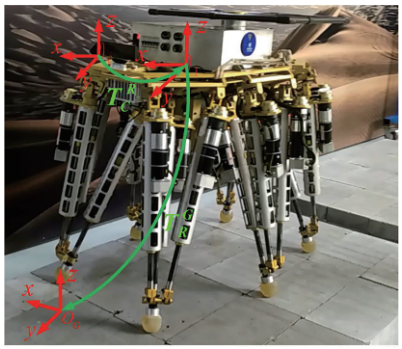

Figure 4 (Color online) The stereo camera system for the GFGM.

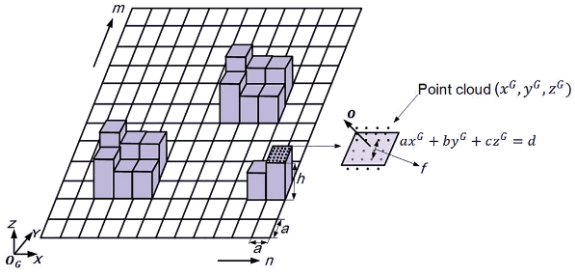

Figure 5 (Color online) Description of the GFGM.

The normal vector of plane$^{[p,q]}$ is denoted by $\boldsymbol{n}^{[p,q]}(a^{[p,q]}, b^{[p,q]}, c^{[p,q]})$. $f^{[p,q]}$ is used to describe the flatness of the partial terrain. It is defined as the average distance of all the point cloud to the plane$^{[p,q]}$. $f^{[p,q]}$ is calculated from the following equation:

$$f^{[p,q]} = \frac{\sum_{i=1}^{k} |a^{[p,q]}x_i^{[p,q]} + b^{[p,q]}y_i^{[p,q]} + c^{[p,q]}z_i^{[p,q]} - d^{[p,q]}|}{k}. \tag{5}$$

In the view of geometry, if the partial terrain in grid$^{[p,q]}$ suddenly changes, plane$^{[p,q]}$ will not fit the terrain surface well, and $f^{[p,q]}$ will have a larger value than that of a normal grid. Using this property, $f^{[p,q]}$ can be used to find the obstacle edge efficiently.

$o^{[p,q]}$ describes the partial terrain's orientation relative to the ground. The ground plane is estimated too, and it is denoted by plane^G which is expressed by the formula $a^G x + b^G y + c^G z = d^G$. The normal vector of $plane^G$ is denoted by $\boldsymbol{n}^G(a^G, b^G, c^G)$. $o^{[p,q]}$ is calculated from the following equation:

$$o^{[p,q]} = \arccos\left(\frac{\boldsymbol{n}^G \cdot \boldsymbol{n}^{[p,q]}}{|\boldsymbol{n}^G||\boldsymbol{n}^{[p,q]}|}\right). \tag{6}$$

Finally, the GFGM is a discrete function of two integers p, q to the partial terrain's geometric feature including the height, the flatness and the orientation

$$\begin{aligned} &\text{GFGM}(p,q) \to (h^{[p,q]}, f^{[p,q]}, o^{[p,q]}) \\ &\quad p = 1, \dots, m, q = 1, \dots, n. \end{aligned} \tag{7}$$

According to eq. (2), GFGM(p, q) can be inversely mapped to the corresponding position in the GCS.

The variable $obs^{[p,q]}$ is defined to describe whether grid$^{[p,q]}$ belongs to an obstacle. $obs^{[p,q]}$ is computed using the following equation:

$$obs^{[p,q]} = w_1 \cdot \frac{\delta_h^{[p,q]}}{N_{\delta_h}} + w_2 \cdot \frac{o^{[p,q]}}{N_o} + w_3 \cdot \frac{f^{[p,q]}}{N_f}, \tag{8}$$

where $\delta_h^{[p,q]}$ is the height difference between the partial terrain and the ground. w_1, w_2 and w_3 are the weighting factors which sum up to 1. They are used to assign specific priorities to corresponding geometric features. N_{δ_h}, N_o and N_f are normalization factors, and each factor is a larger value.

The obstacle map (OM) is formulated as a discrete function of two integers p, q

$$OM(p,q) = \begin{cases} 1, & \text{if } obs^{[p,q]} \geq \Delta_{obs}, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

where Δ_{obs} is the threshold used to judge whether grid$^{[p,q]}$ is a part of an obstacle. As Figure 6 shows, the *OM*'s value 1 represents the grid belongs to an obstacle, the *OM*'s value 0 represents the grid belongs to the ground. Corresponding to the GFGM, OM is a 2D grid map whose size is $m \times n$, and the value of each grid is 1 or 0.

3.2 The obstacle detection

The OM describes whether a grid is a part of a complete obstacle. A complete obstacle consists of several connecting grids. It is necessary to extract the contour of the complete obstacle, and find its position in the GCS. For this purpose, the obstacle detection method is proposed here.

As Figure 7 shows, partial obstacles in a same row are detected first. Then the connection relation table of partial obstacles is established. Thirdly, traversing the relation table, if a partial obstacle in the current row is connected to another partial obstacle in the adjacent row, they are judged to belong to a same complete obstacle; if not connected, a new obstacle is produced. Fourthly, finding the edges of the obstacle, and using an appropriate geometric model to envelope the obstacle.

The following algorithms show the method in detail:

Algorithm 1 The algorithm for detection of partial obstacles in a same row

procedure PARTIALOBSTACLEDETECTION

n_{po}: **partial obstacle counter**,

$e_l(n_{po})$: left edge of obstacle n_{po},

$e_r(n_{po})$: right edge of obstacle n_{po},

$row(n_{po})$: row index of obstacle n_{po},

for i=1 to m do,

for j=1 to n do,

if $OM(i, j) = 1$ and $OM(i, j-1) = 0$,

$n_{po} = n_{po} + 1$,

$row(n_{po}) = \boldsymbol{i}$,

$e_l(n_{po}) = \boldsymbol{j}$,

if $OM(i, j) = 1$ and $OM(i, j+1) = 0$,

$e_r(n_{po}) = \boldsymbol{j}$,

return n_{po}, $e_l(n_{po})$, $e_r(n_{po})$, $row(n_{po})$,

where n_{po} is the counter of partial obstacles. e_l and e_r are the left edge index and the right edge index of the partial obstacle. *row* is the row index that the partial obstacle is located in. The partial obstacle is detected based on the value of *OM*. The algorithm in Algorithm 1 detects the number and the edges of partial obstacles in the same row.

Algorithm 2 The algorithm for the establishment of the connection relation table

procedure CONNECTIONRELATION

reltab: connection relation table

for i=1 to n_{po} do,

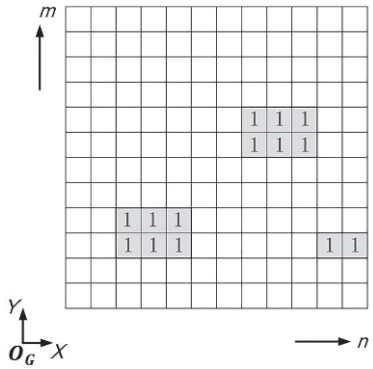

Figure 6 Description of the OM.

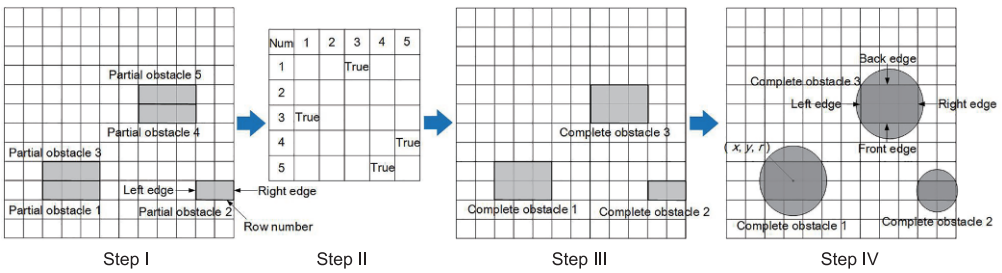

Figure 7 (Color online) Illustration of the obstacle detection method.

for j=i+1 to n_{po}–1 do,
if $row(i)$==$row(j)$+1 and $e_l(i) \le e_r(j) + 1$ and $e_r(i) \ge e_l(j) - 1$,
$reltab[i][j]$=true,
$reltab[j][i]$=true,
return $reltab$,
where $reltab$ is the connection relation table which records whether two partial obstacles in adjacent rows are connected. The algorithm in Algorithm 2 establishes the connection relation table by comparing the edges of two partial obstacles in adjacent rows.

Algorithm 3 The algorithm for detection of the complete obstacle
procedure COMPLETEOBSTACLEDETECTION
n_{co}: complete obstacle counter,
$g(n_{co})$: stack array of the complete obstacle,
for i=1 to n_{po} do,
for each $i \notin g(n_{co})$,
push i in $g(n_{co})$,
n_{co}=n_{co}+1,
for j=1 to size($g(n_{co})$) do,
for each $reltab[g(n_{co})][k]$ == $true$,
push k in $g(n_{co})$,
return n_{co}, $g(n_{co})$,
where n_{co} is the number of complete obstacles. g is a stack array which stores partial obstacles belonging to the same complete obstacle. The algorithm in Algorithm 3 detects the complete obstacle by analyzing the connection relation in $reltab$.

The output results of the algorithm in Algorithm 3 are the number and detailed geometric information of the grids belonging to a whole complete obstacle. Based on the output results in Algorithm 3, several regular geometric models can be used to envelope and represent the obstacle. In this paper, the cylinder model is used just as an example to better discuss the whole obstacle avoidance theory. Algorithm 4 shows the detailed algorithm of obstacle representation using the cylinder model.

Algorithm 4 The algorithm for obstacle representation
procedure OBSTACLEREPRESENTATION
$C_{co}(n_{co})$: circle enveloping the obstacle bottom
for i=1 to n_{co} do,
for each partial obstacle in $g(n_{co})$,
E_l=min(e_l),
E_r=max(e_r),
E_f=min(row),
E_b=max(row),

$$C_{co}\left(n_{co}\right) = \left[\left(\frac{E_l + E_r}{2} \times a, \frac{E_f + E_b}{2} \times a\right), \frac{\sqrt{(E_r - E_l)^2 + (E_b - E_f)^2}}{2} \times a\right],$$

return $C_{co}(n_{co})$
where E_l, E_r, E_f and E_b are the left edge index, right edge index, front edge index and back edge index of the complete obstacle. $C_{co}(x, y, r)$ is denoted as a circle with proper center (x, y) and radius r, which envelopes the bottom of the complete obstacle.

Figure 8 describes the obstacle representation using the cylinder model. The obstacle represented by a cylinder model is denoted by $B(x, y, r, h)$, which is defined by four parameters. x, y, r determine the position and size of the cylinder bottom, which is computed by the obstacle representation algorithm in Algorithm 4. The height of the cylinder is denoted by h, which is the maximum height of all the grids contained in the cylinder.

4 Obstacle avoidance method

We will now present the method of planning a safe path in unknown environment with obstacles. The planned path should ensure that the robot can walk safely without encountering with any obstacles. In the following subsections, we will first propose the concept of the virtual obstacle, which will be used in the following path planning method to restrict the robot to walk on the safe detection terrain. Then the safe conversion pose is defined for the robot to reach. The safe conversion pose is far from the obstacle and the unknown terrain, which guarantees the robot's safety. At last, the Bezier curve is used to plan a safe path which is obstacle-free and passes through the safe conversion poses.

4.1 The virtual obstacle

The concept of the virtual obstacle is proposed to add the FOV (field of view) constraint of the robot. Vision sensors usually have the limited FOV, so only part of the terrain in front of the robot can be detected. As shown in Figure 9, the detected obstacle, which is noted as the real obstacle, is represented by the solid cylinder as mentioned in Section 3. $B_{ri}(x_{ri}, y_{ri}, r_{ri}, h_{ri})$ denotes the real obstacle on the detection terrain.

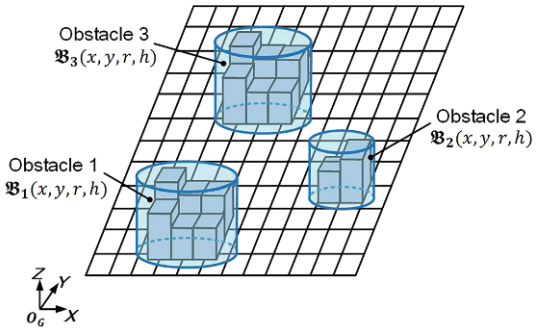

Figure 8 (Color online) Obstacle representation using the cylinder model.

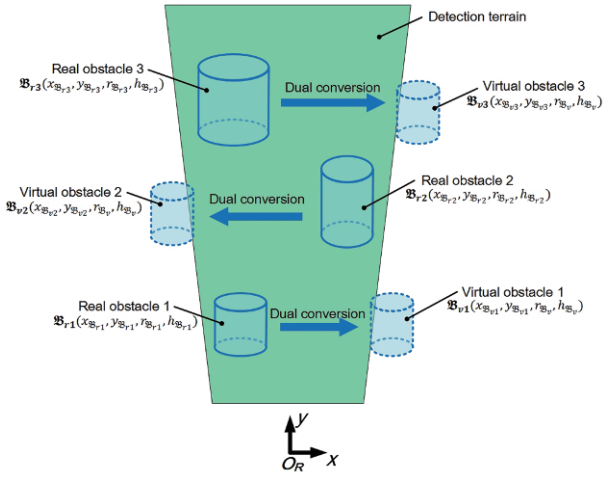

Figure 9 (Color online) Dual conversion from the real obstacle to the virtual obstacle.

The virtual obstacle does not exist in the real world. Similar to the representation of the real obstacle, the cylinder model is used again to represent the virtual obstacle in quantity. As shown in Figure 9, the virtual obstacle, depicted by the dotted cylinder and denoted by $\mathfrak{B}_{vi}(x_{vi}, y_{vi}, r_v, h_v)$, appears in the dual form together with the real obstacle. The virtual obstacle is regulated to have a unified size. The radius r_v and the height h_v of the cylinder are predefined values. The position of the virtual obstacle is related to that of the corresponding real obstacle. The virtual obstacle is used to avoid the robot to walk on the unknown terrain, so its position is located at the edge of the detection area. A real obstacle divides the detection terrain into two spaces, one is narrow, the other one is relatively wide. The wide space is chosen for the robot to pass through in order to guarantee its safety. So the corresponding virtual obstacle is located at the other side of the wide space. The positon of the virtual obstacle is calculated by the position conversion of the corresponding real obstacle as following:

$$x_{vi} = \begin{cases} x_{ri} + d_{ei} + r_v, & \text{if } x_{ri} < 0, \\ x_{ri} - d_{ei} - r_v, & \text{if } x_{ri} \geq 0, \end{cases} \qquad y_{vi} = y_{ri}, \tag{10}$$

where d_{ei} denotes the distance between the real obstacle and the edge of the detection terrain. $x_{ri} < 0$ means that the real obstacle is on the left side of the robot, the corresponding virtual obstacle is located on the right side of the robot. And $x_{ri} > 0$ means that the real obstacle is on the right side of the robot, the corresponding virtual obstacle is located on the left side of the robot. Eq. (10) is established with respect to the RCS. In the following subsections, all the obstacles' positions are transformed in the GCS.

4.2 The safe conversion pose

The robot needs to pass through spaces formed by real obstacles and corresponding virtual obstacles safely in order to reach the target position to execute tasks. Considering the situation of the robot walking through a real obstacle and the corresponding virtual obstacle. Firstly the robot gets closer and closer to the real obstacle and the virtual obstacle. Then after reaching a conversion pose, the robot walks farther and farther from both obstacles. The robot at the conversion pose is the nearest to both obstacles, it is dangerous for the robot. So it is very essential for the robot to find a safe conversion pose to guarantee its safety as much as possible during the whole walking process.

In general, the robot should not be prone to be close to either one of the both obstacles. The robot maybe encounter with the obstacle if it is too close to the obstacle, because the measurement error from the stereo camera and the kinematic error from the robot exist inevitably. In the view of geometry, distances from the real obstacle and the virtual obstacle to their midpoint are equal. So the midpoint of the real obstacle and the virtual obstacle is the most appropriate and safest selection as a conversion point for the robot. The robot has a long hexagonal body, the heading angle passing through the conversion point should guarantee that the distances from the entire body to both obstacles are equal. In order to satisfy the above constraint, the robot must pass though the conversion point along the direction which is vertical to the line formed by both obstacles.

Based on above analysis, the safe conversion pose $\mathcal{P}_s(x_s, y_s, z_s, \alpha_s, \beta_s, \gamma_s)$ is defined and shown in Figure 10. (x_s, y_s, z_s) and $(\alpha_s, \beta_s, \gamma_s)$ denote the robot's position and orientation. α_s, β_s, γ_s are rotation angles along the fixed x-, y- and z-axis respectively. On the level ground, z_s, α_s, β_s are initialed with appropriate values and remain constant. γ_s, x_s, y_s can be calculated from the following equations:

$$\begin{aligned} \gamma_s &= \arctan\left(\frac{y_v - y_r}{x_v - x_r}\right), \\ x_s &= \frac{x_r + x_v + \cos\gamma_s \cdot (r_r - r_v)}{2}, \\ y_s &= \frac{y_r + y_v + \sin\gamma_s \cdot (r_r - r_v)}{2}, \end{aligned} \tag{11}$$

where γ_s is the heading angle of the robot passing through the safe conversion point, it is vertical to $\mathfrak{B}_r\mathfrak{B}_v$ linear direction. The position of the safe conversion point is denoted by (x_s, y_s), which is the midpoint of $\mathfrak{B}_r$ edge and $\mathfrak{B}_v$ edge along the $\mathfrak{B}_r\mathfrak{B}_v$ linear direction.

4.3 Computation of the distance to obstacles

The geometric representation of the robot is indispensable for computing the distance to obstacles. The representation method using regular geometric models can efficiently decrease the complexity and time consuming for the distance computation. In this paper, the cylinder model is still selected to represent the robot because of its efficiency and simplicity.

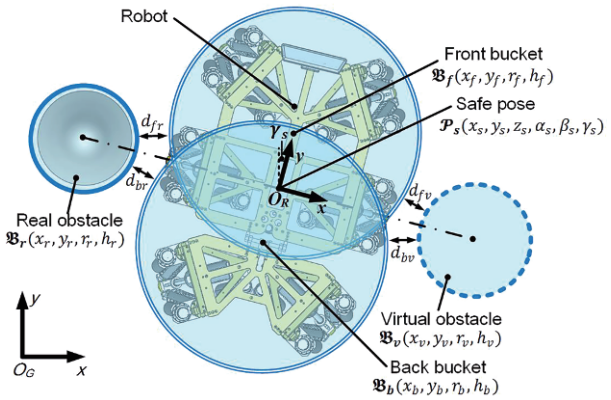

Figure 10 (Color online) Definition of the safe conversion pose and the distance between the robot and the obstacle.

The Hexapod-III robot has a long hexagonal body, two cylinders are used to envelope its entire body as Figure 10 shows. The front cylinder $\mathfrak{B}_f(x_f, y_f, r_f, h_f)$ and the back cylinder $\mathfrak{B}_b(x_b, y_b, r_b, h_b)$ respectively envelop the front part and the back part of the robot. In order to insure that the entire body is contained in $\mathfrak{B}_f$ and $\mathfrak{B}_b$ when the robot walks, the size and the position of $\mathfrak{B}_f$ and $\mathfrak{B}_b$ are calculated previously based on the real geometric model of the robot. Whenever the robot moves, the positions of $\mathfrak{B}_f$ and $\mathfrak{B}_b$ are updated with new values.

The distance to obstacles will be used to generate an obstacle avoidance path in the following section. For this reason, the formal definition of the distance between the robot and the obstacle is presented here. As shown in Figure 10, when the robot is passing through one pair of obstacles, four distances d_{fr}, d_{fv}, d_{br} and d_{bv} are defined. d_{fr} and d_{fv} are the distances between the front part of the robot to the real obstacle and the virtual obstacle. d_{br} and d_{bv} are the distances between the back part of the robot to the real obstacle and the virtual obstacle. d_{fr} is calculated from eq. (12), and other three distances d_{fv}, d_{br} and d_{bv} can be calculated using similar equations

$$d_{fr} = \sqrt{(x_f - x_{ri})^2 + (y_{fr} - y_{ri})^2} - r_{fr} - r_{ri}. \tag{12}$$

4.4 Obstacle avoidance path planning

Turning to the issue of obstacle avoidance path planning for the robot, a smooth obstacle-free path from the robot's initial pose to the safe conversion poses should be found as Figure 11 shows. The safe conversion poses are generated from real obstacles and the corresponding virtual obstacles as mentioned in Section 4.2. Distances from the robot to obstacles, which are defined in Section 4.3, should be larger than the minimum safety distance in order to satisfy the constraint of obstacle avoidance. The Bezier curve is used to parameterize the smooth obstacle-free path. The basic insight of the thought here is parameterizing the path as a Bezier curve initially, then accomplishing the obstacle-free path planning and optimization simultaneously.

Bezier curves were first proposed by the French engineer Pierre Bézier for designing the body shape of automobiles. Bezier curve does not pass through all the control points used to define it. It only passes through the starting and ending control points. Bezier curve is completely contained in the convex hull constructed by its control points. The tangent vectors at the starting and ending points are directly along the first and last span of the convex hull. Modifying the positions of the control points can change the global shape of the Bezier curve. These properties are very useful for planning an obstacle avoidance path.

The third-order Bezier curve is used to plan an obstacle avoidance path as Figure 12 shows. A single segment planning, during which the robot moves from the current safe conversion pose $\mathcal{P}_{si}(x_{si}, y_{si}, z_{si}, \alpha_{si}, \beta_{si}, \gamma_{si})$ to the next safe conversion pose $\mathcal{P}_{sj}(x_{sj}, y_{sj}, z_{sj}, \alpha_{sj}, \beta_{sj}, \gamma_{sj})$, is analyzed here. The third-order Bezier curve has four control points $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$. The starting point $A = [x_{si}, y_{si}]^{\mathrm{T}}$ and the ending point $D = [x_{sj}, y_{sj}]^{\mathrm{T}}$ are located at the positions of $\mathcal{P}_{si}$ and $\mathcal{P}_{sj}$. The second control point $B = [x_{Bi}, y_{Bi}]^{\mathrm{T}}$ and the third control point $C = [x_{Ci}, y_{Ci}]^{\mathrm{T}}$ are computed using the optimization method. These four control points $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ determine the global shape of the Bezier curve, which needs to satisfy constraints of obstacle avoidance and the heading angles of the robot at $\mathcal{P}_{si}$ and $\mathcal{P}_{sj}$. The third-order Bezier curve $w(u)$ passing through these four control points $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ is defined as

$$w(u) = [x(u), y(u)]^{\mathrm{T}} = (1-u)^3 A + 3u(1-u)^2 B + 3u^2(1-u)C + u^3 D, \tag{13}$$

where u is the normalized time variable and equal to t / T. T is the time the robot spends moving from $\mathcal{P}_{si}$ and $\mathcal{P}_{sj}$. Eq. (13) is expanded and rearranged to the form of a third order polynomials in u as

$$x(u) = \left(-x_{si} + 3x_{Bi} - 3x_{Ci} + x_{sj}\right)u^3 + (3x_{si} - 6x_{Bi} + 3x_{Ci})u^2 + (-3x_{si} + 3x_{Bi})u + x_{si}, \tag{14a}$$

$$y(u) = \left(-y_{si} + 3y_{Bi} - 3y_{Ci} + y_{sj}\right)u^3 + (3y_{si} - 6y_{Bi} + 3y_{Ci})u^2 + (-3y_{si} + 3y_{Bi})u + y_{si}. \tag{14b}$$

As Figure 12 shows, the control point $\boldsymbol{B}$ is determined by the robot heading angle γ_{si} at $\mathcal{P}_{si}$ and the length d_{si1} of the first span of the Bezier convex hull. And the control point $\boldsymbol{C}$ is determined by the robot heading angle γ_{sj} at $\mathcal{P}_{sj}$ and the length d_{si2} of the last span of the Bezier convex hull. The control points $\boldsymbol{B}$ and $\boldsymbol{C}$ are calculated from the following equations:

$$x_{Bi} = x_{si} + d_{si1}\cos\gamma_{si}, \tag{15a}$$

$$y_{Bi} = y_{si} + d_{si1}\sin\gamma_{si}, \tag{15b}$$

$$x_{Ci} = x_{sj} - d_{si2}\cos\gamma_{sj}, \tag{15c}$$

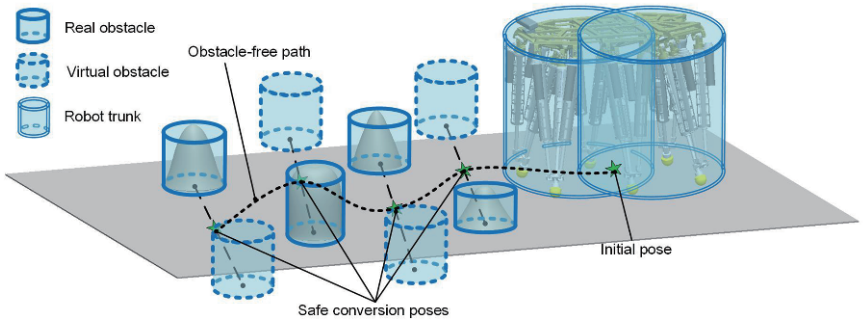

Figure 11 (Color online) Illustration of the obstacle avoidance path planning.

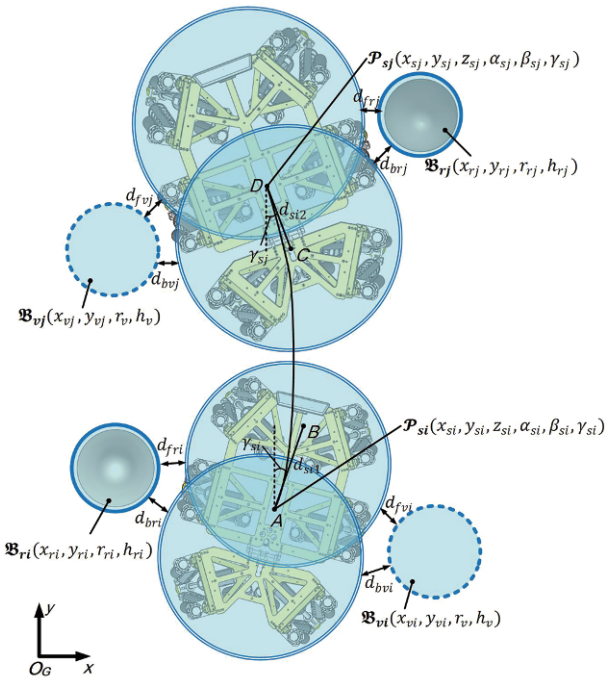

Figure 12 (Color online) Path planning based on the Bezier curves.

$$y_{Ci} = y_{sj} - d_{si2}\sin\gamma_{sj}. \tag{15d}$$

For the known starting pose P_{si} and the calculated ending pose P_{sj} of the robot, the shape of the Bezier curve changes with the variables d_{si1} and d_{si2}. What's more, the length of the Bezier curve is directly related to d_{si1} and d_{si2} too. So d_{si1} and d_{si2} are two important parameters which can be used to optimize the Bezier curve which is obstacle-free and has a relatively short length simultaneously. The length of the Bezier curve $w(u)$ is denoted by $\xi(u)$ and is computed by the following equation:

$$\xi(u) = \int_0^1 v(u)\mathrm{d}u, \tag{16}$$

where $v(u)$ is the tangent velocity and can be calculated by the following equations :

$$v(u) = \sqrt{\dot{x}^2(u) + \dot{y}^2(u)}, \tag{17}$$

$$\dot{x}(u) = (-3x_{si} + 9x_{Bi} - 9x_{Ci} + 3x_{sj})u^2 + (6x_{si} - 12x_{Bi} + 6x_{Ci})u + (-3x_{si} + 3x_{Bi}), \tag{18a}$$

$$\dot{y}(u) = (-3y_{si} + 9y_{Bi} - 9y_{Ci} + 3y_{sj})u^2 + (6y_{si} - 12y_{Bi} + 6y_{Ci})u + (-3y_{si} + 3y_{Bi}), \tag{18b}$$

where $\dot{x}(u)$ stands for $\frac{\mathrm{d}x(u)}{\mathrm{d}u}$ and $\dot{y}(u)$ stands for $\frac{\mathrm{d}y(u)}{\mathrm{d}u}$.

The safety margin used to avoid a collision between the robot and the obstacle, denoted by δ_s, is defined as the minimum acceptable distance between the robot and the obstacle. And the following conditions of obstacle avoidance are obtained. As Figure 12 shows, when the robot moves away from the current ith pair of obstacles, distances from the robot to B_{ri} and B_{vi} should be larger than δ_s, $d_{fri}(u) \geq \delta_s$, $d_{fvi}(u) \geq \delta_s$, $d_{bri}(u) \geq \delta_s$ and $d_{bvi}(u) \geq \delta_s$, $0 \leq u \leq 1$. When the robot walks close to the next jth pair of obstacles, distances from the robot to B_{rj} and B_{vj} should be larger than δ_s too, $d_{frj}(u) \geq \delta_s$, $d_{fvj}(u) \geq \delta_s$, $d_{brj}(u) \geq \delta_s$ and $d_{bvj}(u) \geq \delta_s$, $0 \leq u \leq 1$. Fulfilling above criterions means the robot will not encounter with obstacles when it walks from P_{si} to P_{sj}. The obstacle avoidance path planning problem can be formulated as an optimization problem as follows:

$$\begin{aligned}
&\text{minimize } \xi(u)\\
&\text{subject to:}\\
&d_{fri}(u) \geq \delta_s,\ d_{fvi}(u) \geq \delta_s,\\
&d_{bri}(u) \geq \delta_s,\ d_{bvi}(u) \geq \delta_s,\\
&d_{frj}(u) \geq \delta_s,\ d_{fvj}(u) \geq \delta_s,\\
&d_{brj}(u) \geq \delta_s,\ d_{bvj}(u) \geq \delta_s,\quad 0 \leq u \leq 1.
\end{aligned} \tag{19}$$

The above problem is an optimization problem with inequality constraints. The penalty function is used to transform the constrained optimization problem to an unconstrained optimization problem. The constraints are added to the objective function in the form of penalty parameters. In such a way, the violation of the constraints is penalized. The penalty function $p(d)$ is defined as follows:

$$p(d)=\begin{cases}\dfrac{1}{2}\left(\dfrac{1}{d}-\dfrac{1}{\delta_s}\right)^2, d<\delta_s,\\ 0\ , \quad d\geq\delta_s,\end{cases} \tag{20}$$

where d is the distance from the robot to the obstacle. When the distance d is less than δ_s, the value of $p(d)$ increases as d decreases. $p(d)$ is equal to 0 whenever d is larger than the safety margin δ_s. It means the distance between the robot and the obstacle is safe enough. Finally, the transformed unconstrained optimization problem $F(d_{si1}, d_{si2})$ is formulated as the following expression:

$$\text{minimize}\, F(d_{si1}, d_{si2}) = \xi(u) + \sum_{k=1}^{8}\sigma_k p(d_k(u)),\ 0\leq u\leq 1, \tag{21}$$

where d_{si1} and d_{si2} are the lengths of the first and last span of the Bezier convex hull. $\xi(u)$ is the length of the Bezier curve. σ_k is the magnification factor for penalizing the violation of the constraints. k is from 1 to 8, and $d_k(u)$ stands for $d_{fri}(u)$, $d_{fvi}(u)$, $d_{bri}(u)$, $d_{bvi}(u)$, $d_{frj}(u)$, $d_{fvj}(u)$, $d_{brj}(u)$ and $d_{bvj}(u)$, respectively. Choosing larger value of σ_k can increase the penalization of the corresponding constraint violation. If a specific σ_k is larger than others, it means the corresponding constraint condition has a higher priority.

5 Robot motion planning

The legged robot is a complex system with multiple inputs and multiple outputs. So motion planning for legged robots is more complicated than their wheeled counterparts. In this section, the motion planning method of controlling the robot to walk along the planned path is presented.

The robot uses the tripod gait for walking in complex environment. Although any stable gait can be applied, the tripod gait is the fastest stable gait for a hexapod robot. The tripod gait divides six feet into two groups, O_{F_1}, O_{F_3}, O_{F_5} and O_{F_2}, O_{F_4}, O_{F_6}. Each group of three feet form a triangle. During walking, the triangle has two states, the support state and the swinging state. Both triangles can be in support state at the same time, but they cannot be in swinging state simultaneously. Figure 13 shows the gait timing sequence, the foot in support state is depicted by a bold solid line, the foot in swinging state is depicted by a bold dashed line. In order to improve the walking speed, the timing sequence planner is regulated as only one triangle is in support state at any time, while the other triangle must be in swinging state.

One complete walking process has two stages as shown in Figure 13. In the first walking stage (WS-I), O_{F_1}, O_{F_3} and O_{F_5} are fixed on the ground to actuate the robot to move along the planned path. At the same time, lifting O_{F_2}, O_{F_4} and O_{F_6}, swinging them to next three footholds. In the second walking stage (WS-II), O_{F_2}, O_{F_4} and O_{F_6} are fixed on the ground continuing to actuate the robot to move along the planned path. At the same time, proceeding to lift O_{F_1}, O_{F_3} and O_{F_5}, and swinging them to next three footholds.

Figure 14 shows the motion planning model of the robot. ${}^{G}\mathcal{T}_{R}({}^{G}x_{R}, {}^{G}y_{R}, {}^{G}z_{R}, {}^{G}\alpha_{R}, {}^{G}\beta_{R}, {}^{G}\gamma_{R})$ represents the robot trajectory in the GCS, which can be obtained from the planned path as indicated in Section 5. ${}^{R}_{G}\boldsymbol{T}$ is the transformation matrix from the GCS to the RCS, which can be calculated from ${}^{G}\mathcal{T}_{R}$. The foot trajectory in the GCS is denoted by ${}^{G}\mathcal{T}_{F}({}^{G}x_{F}, {}^{G}y_{F}, {}^{G}z_{F})$. The trajectory of the swinging foot is regulated as a semi-ellipse curve as shown in Figure 14. U_1, U_2 and U_3 represent the universal joints, P_1, P_2 and P_3 represent the prismatic joints as mentioned in Section 2. Each leg of the robot has three inputs l_1, l_2 and l_3, which are the lengths of P_1, P_2 and P_3. So the robot has 18 inputs in total, and l_1, l_2, l_3 of all six legs need to be calculated to actuate the robot to walk along ${}^{G}\mathcal{T}_{R}$. ${}^{L}_{F}\boldsymbol{T}$ is the transformation matrix from the FCS to the leg coordinate system (LCS). Eq. (22) shows the detailed expression of ${}^{L}_{F}\boldsymbol{T}$.

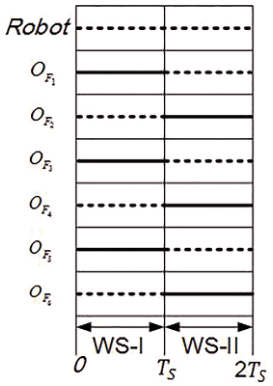

Figure 13 Gait sequence for the robot.

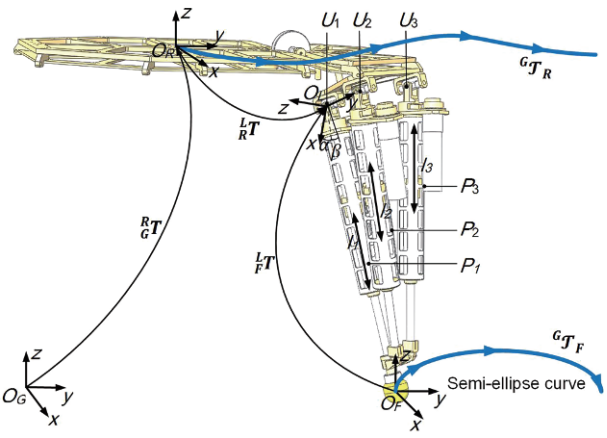

Figure 14 (Color online) Motion planning model of the robot.

$$ {}^{L}_{F}\boldsymbol{T} = \begin{bmatrix} \cos\alpha\cos\beta & -\cos\alpha\sin\beta & \sin\alpha & l_1\cos\alpha\cos\beta \\ \sin\beta & \cos\beta & 0 & l_1\sin\beta \\ -\sin\alpha\cos\beta & \sin\alpha\sin\beta & \cos\alpha & -l_1\sin\alpha\cos\beta \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{22} $$

where α and β are rotation angles of U_1 along two vertical directions. α, β and l_1 can be calculated from the following equations:

$$ l_1 = \sqrt{{}^{L}x_F{}^2 + {}^{L}y_F{}^2 + {}^{L}z_F{}^2}, \tag{23} $$

$$ \alpha = \arctan\left(\frac{{}^{L}z_F}{{}^{L}x_F}\right), \tag{24} $$

$$ \beta = \arcsin\left(-\frac{{}^{L}y_F}{l_1}\right), \tag{25} $$

where ${}^{L}\mathcal{T}_F({}^{L}x_F, {}^{L}y_F, {}^{L}z_F)$ is the position of the foot in the LCS. ${}^{L}\mathcal{T}_F$ can be calculated by substituting α, β and l_1 into eq. (22). The other two inputs l_2 and l_3 are obtained from the following equations:

$$ l_2 = |{}^{L}\mathcal{T}_F - {}^{L}\boldsymbol{U}_2|, \tag{26} $$

$$ l_3 = |{}^{L}\mathcal{T}_F - {}^{L}\boldsymbol{U}_3|, \tag{27} $$

where ${}^{L}U_2$ and ${}^{L}U_3$ respectively represent the spatial positions of U_2 and U_3 in the LCS. $|\ |$ represents the length of the vector.

The following equation is used to transform the foot trajectory $\mathcal{T}_F$ from the GCS to the LCS:

$$ {}^{L}\mathcal{T}_F = {}^{L}_{R}\boldsymbol{T}\,{}^{R}_{G}\boldsymbol{T}\,{}^{G}\mathcal{T}_F, \tag{28} $$

where ${}^{R}_{G}\boldsymbol{T}$ is the transformation matrix from the GCS to the RCS. ${}^{L}_{R}\mathcal{T}$ is the transformation matrix from the RCS to the LCS, which is determined by the mechanical structure dimension of the robot.

As mentioned above, in WS-I, O_{F_1}, O_{F_3} and O_{F_5} are fixed on the ground. And the corresponding trajectories ${}^{G}\mathcal{T}_{F_1}$, ${}^{G}\mathcal{T}_{F_3}$ and ${}^{G}\mathcal{T}_{F_5}$ in the GCS are constant. At the same time, O_{F_2}, O_{F_4} and O_{F_6} are lifted and swung along the semi-eclipse trajectories ${}^{G}\mathcal{T}_{F_2}$, ${}^{G}\mathcal{T}_{F_4}$ and ${}^{G}\mathcal{T}_{F_6}$ respectively. Then ${}^{L}\mathcal{T}_{F_1}$, ${}^{L}\mathcal{T}_{F_2}$, ${}^{L}\mathcal{T}_{F_3}$, ${}^{L}\mathcal{T}_{F_4}$, ${}^{L}\mathcal{T}_{F_5}$ and ${}^{L}\mathcal{T}_{F_6}$ are obtained from eq. (28). The inputs of O_{F_1}, O_{F_2}, O_{F_3}, O_{F_4}, O_{F_5} and O_{F_6} are calculated from eqs. (23), (26) and (27). In WS-II, all the input lengths of the prismatic joins can be calculated in the same way as WS-I.

6 Experiment

The proposed obstacle avoidance theory has been implemented on the Hexapod-III robot. The whole algorithm is coded using C++programming language, which is running in a real-time Linux system on the Beckhoff CX2040 industrial computer (2.1 GHz, 4 cores). In order to reduce the computation consumption in the real application, we use the real-time segment planning method, which consists of two threads. In a single segment planning process, the obstacle is detected and the safe conversion pose is computed in the first thread, then the second thread generates an obstacle-free path to the safe conversion pose and controls the robot to walk. During walking, the first thread still perceives the environment in real time to detect whether there are new obstacles in the generated path. If new obstacles appear, the second thread makes the robot stop, the current segment planning process ends and a new segment planning based on the immediate environmental information is started. If there are no new obstacles, the current segment planning process ends after the robot reaches the safe conversion pose safely, and a new segment planning is started.

The experimental setup is shown in Figure 15. Several obstacles are placed randomly. An external tracking system API T3 Laser Tracker is used to measure the position (x, y, z) and the rotation orientation (pitch, roll, yaw) of the robot. API T3 is integrated with a smart sensor, which tracks the spherical retro reflector (the target ball shown in Figure 15) mounted on the robot.

Figure 16 shows the original point cloud captured by the stereo camera. The point cloud in Figure 16 has been transformed in the GCS, and obstacles are a cluster of scattered points. For the GFGM, we use a size of 120×120 with cells of 2.5 cm side length. The GFGM is centered on the start position of the robot. During the experiment, the grids are shifted according to the immediate pose of the robot calculated by the kinematics model.

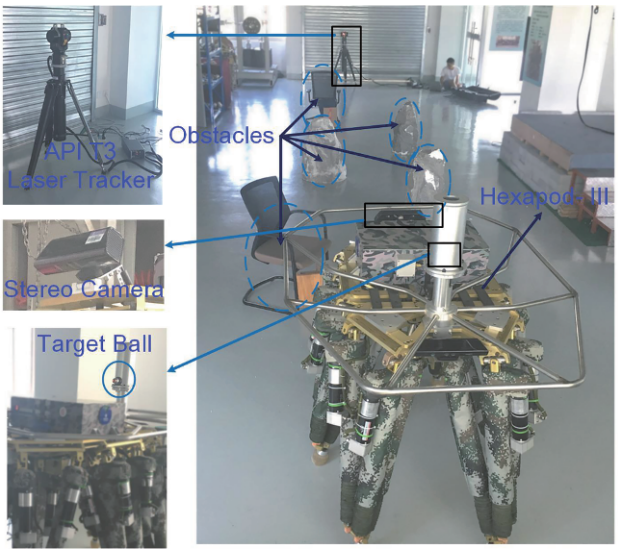

Figure 15 (Color online) The experimental setup.

Table 1 The real obstalces and the virutal obstalces

	Real obstacle					Virtual obstacle				
	$\mathfrak{B}_{r1}$	$\mathfrak{B}_{r2}$	$\mathfrak{B}_{r3}$	$\mathfrak{B}_{r4}$	$\mathfrak{B}_{r5}$	$\mathfrak{B}_{v1}$	$\mathfrak{B}_{v2}$	$\mathfrak{B}_{v3}$	$\mathfrak{B}_{v4}$	$\mathfrak{B}_{v5}$
x (m)	1.80	3.51	5.15	6.69	8.59	1.80	3.51	5.15	6.69	8.59
y (m)	0.48	–0.80	0.18	–1.09	–0.01	–1.35	1.11	–1.68	0.77	–1.87
r (m)	0.23	0.31	0.27	0.26	0.26	0.30	0.30	0.30	0.30	0.30
h (m)	0.45	0.71	0.65	0.67	0.53	0.8	0.8	0.8	0.8	0.8

Figure 17 shows the obstacle detection result based on the GFGM. A total of 5 obstacles are detected, and they are represented by the cylinder model. The detailed geometric parameters of the real obstacles are listed in Table 1. Based on our theory, the corresponding virtual obstacles are generated, and their detailed geometric parameters are also listed in Table 1.

The body trajectory during the whole experimental process is shown in Figure 18. The solid cylinders represent the real obstacles, and the dotted cylinders represent the corresponding virtual obstacles. The body trajectory, which is depicted by the solid line, is obtained by the API T3 measuring the position of the target ball on the robot body. Figure 18 shows the generated path successfully passes through the correctly computed safe conversion poses with adequate safety margin.

The image flow of the experiment is shown in Figure 19. The image flow clearly shows that the robot walks along the generated path and passes through a set of safe conversion poses without encountering with any obstacles. The experimental result successfully validates our theory and its applicability.

Figure 20 shows the immediate lengths of the pristimatic joins. As metioned above, each leg has three control inputs, which are the lengths of the prismatic joints actuated by three motors. The immediate lengths are calculated based on the real-time motors' records. Figure 21 shows the trajectories of 6 feet in space. The trajectories depicted in Figure 21 are obtained using the kinematics model of the robot based on the immediate lengths of the pristimatic joints.

7 Conclusions

In this paper, we have presented a novel obstacle avoidance framework for a hexapod robot. The target of our framework is to solve three key issues: The terrain map building and the

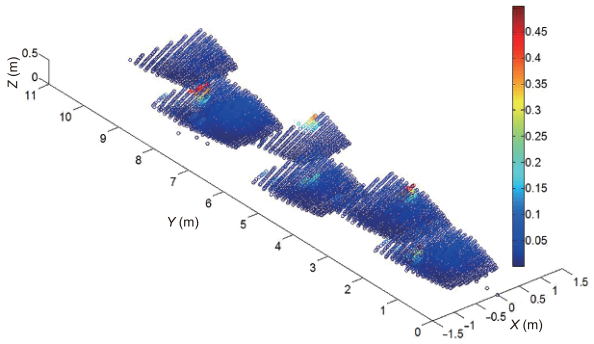

Figure 16 (Color online) Point cloud of the experimental environment.

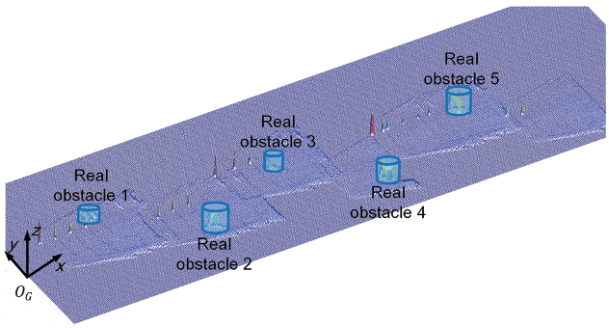

Figure 17 (Color online) Obstacle detection result based on the GFGM.

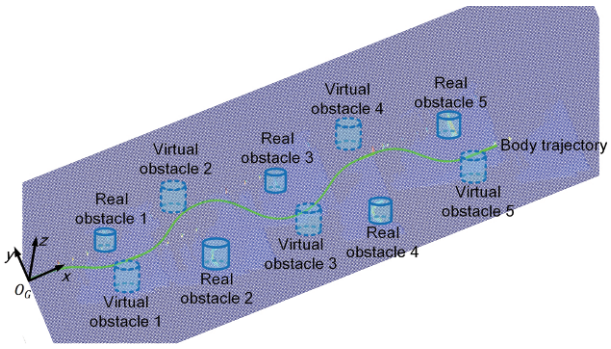

Figure 18 (Color online) Body trajectory tracked by the API T3.

Figure 19 (Color online) Image flow of the experiment.

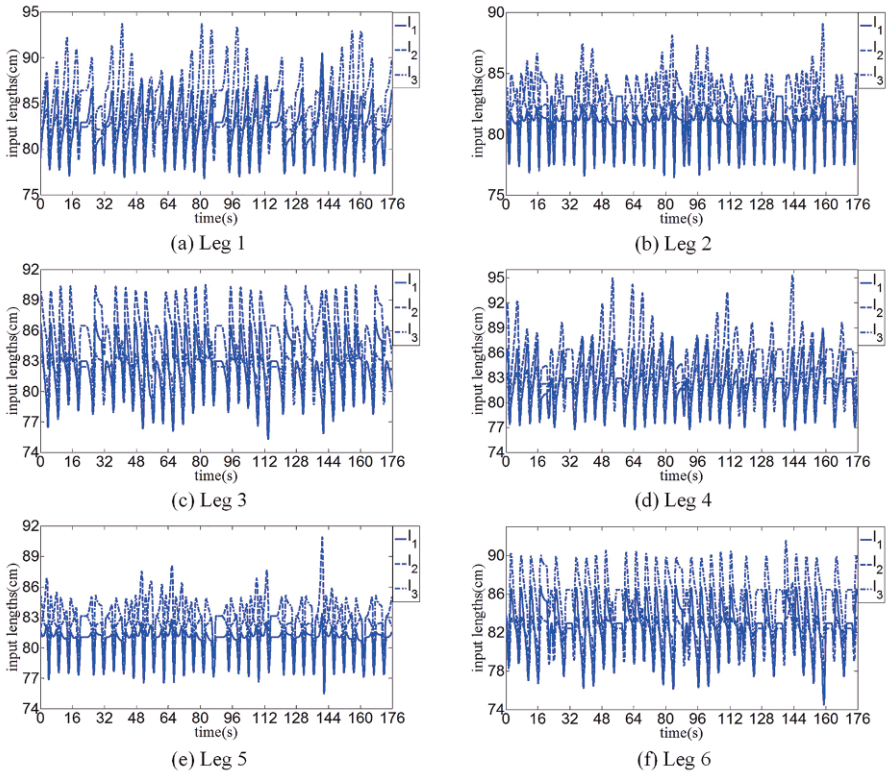

Figure 20 (Color online) The immediate lengths of the prismatic joints.

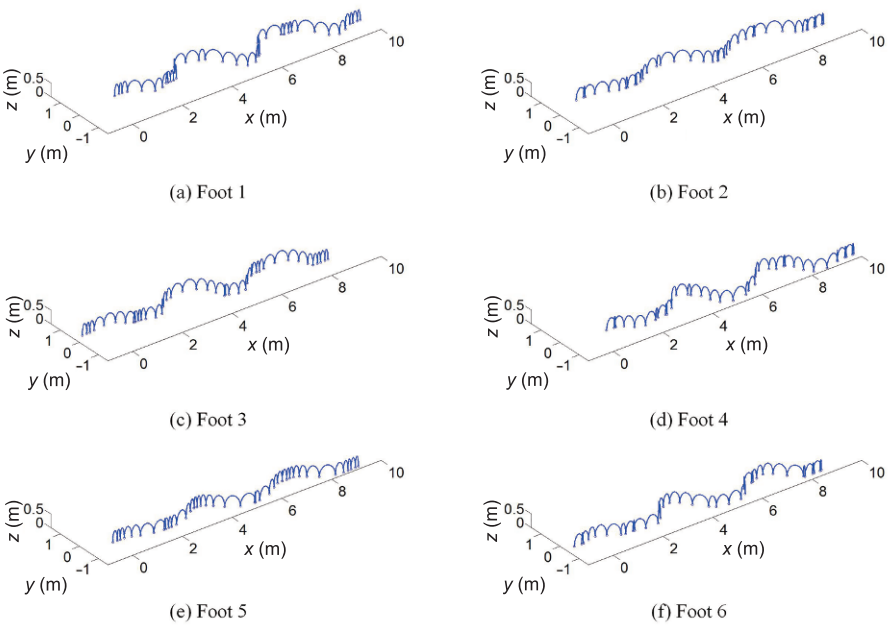

Figure 21 (Color online) Trajectories of different feet.

obstacle detection, the obstacle avoidance path planning, the motion planning for the hexapod robot. These issues have seldom been involved in the related legged robot research.

The theoretical contributions of this paper are summarized as follows. For the terrain map building, the novel GFGM is introduced. Each grid in the GFGM stores typical geometric features of the terrain, which is helpful to the obstacle detection. For the obstacle detection, the detailed algorithm based on the GFGM is described. For the obstacle avoidance path planning, the concept of virtual obstacles is proposed to avoid the robot to walk on the unknown terrain, and the safe conversion pose is defined for the robot to pass through safely.

A safe path based on Bezier curves is generated by solving an optimization problem taking into account the path length subjected to the constraint of obstacle avoidance. For the motion planning, the gait planning method integrated the robot kinematics is presented. Finally, the proposed obstacle avoidance framework is applied to the Hexapod-III robot, and the method's theory and practicality has been validated by the experiment.

This work was supported by the National Basic Research Program of China (Grant No. 2013CB035501).

1 Perrin N, Stasse O, Lamiraux F, et al. Weakly collision-free paths for continuous humanoid footstep planning. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). San Francisco: IEEE, 2011. 4408–4413
2 Raibert M, Blankespoor K, Nelson G, et al. Bigdog, the rough-terrain quadruped robot. In: Proceedings of the 17th World Congress. Seoul, 2008. 10822–10825
3 Kolter J Z, Ng A Y. The stanford littledog: A learning and rapid replanning approach to quadruped locomotion. Int J Robot Res, 2011, 30: 150–174
4 Stelzer A, Hirschmüller H, Görner M. Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. Int J Robot Res, 2012, 31: 381–402
5 Belter D, Skrzypczyński P. Rough terrain mapping and classification for foothold selection in a walking robot. J Field Robot, 2011, 28: 497–528
6 Mae Y, Arai T, Inoue K, et al. Application of a 'limb mechanism' robot to rescue tasks. Adv Robot, 2002, 16: 529–532
7 Virk G S, Gatsoulis Y, Parack M, et al. Mobile robotic issues for urban search and rescue. IFAC-PapersOnline, 2008, 17: 3098–3013
8 Bogdan Rusu R, Sundaresan A, Morisset B, et al. Leaving Flatland: Efficient real-time three-dimensional perception and motion planning. J Field Robot, 2009, 26: 841–862
9 Chilian A, Hirschmuller H, Gorner M. Multisensor data fusion for robust pose estimation of a six-legged walking robot. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). San Francisco: IEEE, 2011. 2497–2504
10 Cho K, Baeg S H, Park S. 3D pose and target position estimation for a quadruped walking robot. In: Proceedings of 2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). Jeju, 2013. 466–467
11 Okada Y, Nagatani K, Yoshida K, et al. Shared autonomy system for tracked vehicles on rough terrain based on continuous three-dimensional terrain scanning. J Field Robot, 2011, 28: 875–893
12 Kagami S, Nishiwaki K, Kuffner J J, et al. Vision-based 2.5 D terrain modeling for humanoid locomotion. In: Proceedings of the 2003 IEEE International Conference on Robotics and Automation. Taipei: IEEE, 2003. 2141–2146
13 Pfaff P, Burgard W. An efficient extension of elevation maps for outdoor terrain mapping. Field Serv Robot, 2006, 25: 195–206
14 Rekleitis I, Bedwani J L, Dupuis E, et al. Autonomous over-the-horizon navigation using LIDAR data. Auton Robot, 2013, 34: 1–18
15 Ishigami G, Otsuki M, Kubota T. Range-dependent terrain mapping and multipath planning using cylindrical coordinates for a planetary exploration rover. J Field Robot, 2013, 30: 536–551
16 Gutmann J S, Fukuchi M, Fujita M. A floor and obstacle height map for 3D navigation of a humanoid robot. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. Barcelona: IEEE, 2005. 1066–1071
17 Mu Z, Xu W, Gao X, et al. Obstacles modeling and collision detection of space robots for performing on-orbit services. In: Proceedings of 2014 4th IEEE International Conference on Information Science and Technology. Shenzhen: IEEE, 2014. 461–466
18 Matthies L, Brockers R, Kuwata Y, et al. Vision-based obstacle avoidance for micro air vehicles using disparity space. In: Proceedings of the 2014 IEEE International Conference on Robotics and Automation. Hong Kong: IEEE, 2014. 3242–3249
19 Silva E C, Costa M F, Erlhagen W, et al. Superquadrics objects representation for robot manipulation. In: Proceedings of International Conference of Numerical Analysis and Applied Mathematics 2015. Rhodes: AIP, 2016
20 Cherubini A, Chaumette F. Visual navigation of a mobile robot with laser-based collision avoidance. Int J Robot Res, 2013, 32: 189–205
21 Parhi D R, Singh M K. Intelligent fuzzy interface technique for the control of an autonomous mobile robot. Proc Inst Mech Eng Part C-J Mech Eng Sci, 2008, 222: 2281–2292
22 Škrjanc I, Klančar G. Optimal cooperative collision avoidance between multiple robots based on Bernstein-Bézier curves. Robot Auton Syst, 2010, 58: 1–9
23 Jolly K G, Sreerama Kumar R, Vijayakumar R. A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits. Robot Auton Syst, 2009, 57: 23–33
24 Uchiyama N, Dewi T, Sano S. Collision avoidance control for a human-operated four-wheeled mobile robot. Proc Inst Mech Eng Part C-J Mech Eng Sci, 2014, 228: 2278–2284
25 Prassler E. Navigating a robotic wheelchair in a railway station during rush hour. Int J Robot Res, 1999, 18: 711–727
26 Chai X, Gao F, Pan Y, et al. A novel identification methodology for the coordinate relationship between a 3D vision system and a legged robot. Sensors, 2015, 15: 9519–9546
27 Pan Y, Gao F. A new 6-parallel-legged walking robot for drilling holes on the fuselage. Proc Inst Mech Eng Part C-J Mech Eng Sci, 2013, 228: 753–764
28 Yang P, Gao F. Leg kinematic analysis and prototype experiments of walking-operating multifunctional hexapod robot. Proc Inst Mech Eng Part C-J Mech Eng Sci, 2013, 228: 2217–2232