Obstacle Avoidance for A Legged Robot Based on FFT Control

Xia Yixuan
xiayixuan@u.nus.edu
National University of Singapore

Guillaume Sartoretti
guillaume.sartoretti@nus.edu.sg
National University of Singapore

ABSTRACT

This paper describes a novel method that combines both Central Pattern Generators (CPG) and computer vision for a legged robot to avoid obstacles. In particular, we focus on cases where a legged robot is tasked with traversing a structured (i.e., periodically spaced) obstacle field. We propose a new visual controller based on Fast Fourier Transform (FFT) to improve the performance of legged robots during footfall planning. We first present how to compute the period of obstacles from the image taken by the robot. Then, we propose a bio-inspired locomotion model, based on the CPG framework, and show how to adapt the relevant parameters to enable safe locomotion over repeated obstacles. Finally, we experimentally demonstrate our approach on a hexapod robot, locomotion through obstacles with constant spacing.

KEYWORDS

Hexapod Robot, Fast Fourier Transform (FFT), computer vision, vision-based locomotio, Central Pattern Generator(CPG)

1 INTRODUCTION

Animals are able to negotiate complicated environment with their flexible limbs and bodies based on sensory feedback. Human beings can also coordinate their bodies and limbs to avoid obstacles during their locomotion. On the contrary, the community is working to develop similar controller for legged robots. In this article, we introduce an approach for controlling a legged robot to avoid obstacles based on visual and proprioceptive feedback. Obstacles can refer to both objects above the ground or holes in the ground.

Many inventions are inspired by the behavior of animals [14]. For the control of legged robots, the rhythmic behavior of legged animals serves as an inspiration. The stability and simplicity of this kind of behavior are what the robotics engineers are seeking. In vertebrates, this behavior is actually generated by central pattern generators (CPG). Biologically, CPGs are neural circuits which are able to produce coupled patterns of rhythmic activity without the requirement of feedback or control signals [6]. In other words, CPGs do not need any feedback or input to produce output. For hexapod robots, CPG models are inspired by insect locomotion and are often modeled as coupled oscillators that control the locomotion by directly outputing joint angle values for key articulations.

A CPG model can generate rhythmic activity, which prescribes a given gait for a legged robot. It is then interesting to study the interaction between the periodic legged gaits and periodical holes in the ground. However, the gaits generated by the CPG models needs adaptation to go through challenging environments. In our case, which considers the locomotion of a legged robot over a field containing equally spaced obtascles, we propose to rely mainly on visual feedback to ensure safe locomotion.

Our task can be divided into two parts. The first part is to design a bio-inspired locomotion controller. The second one is to do the image processing and compute the needed information to regulate the locomotion.

In our case, since obstacles are equally spaced, we propose to rely on Fast Fourier Transform (FFT), is a type of algorithm that can compute the frequencies of a input signal rapidly [5]. In this way, we can obtain the frequency of holes by FFT and compute the period to adapt the gait to periodic holes.

The article is structured as follows. We provide a brief introduction about FFT and CPG in Section 2. Section 3 describes how we can calculate the period of holes by image calibration, contour recognition and FFT. Section 4 details the CPG model, the autonomous obstacle avoidance and experimental results on hexapod robot. Section 5 outlines the solved problem, limitation caused by the hexapod robot and suggestion for future work.

2 BACKGROUND

In this section, we present a general background on central pattern generators (CPG) and Fast Fourier Transform (FFT). We first introduce the modeling process of CPG which is related to the locomotion of our legged robot. In particular, this work focus on developing a method for a hexapod robot, due to its increased static stability. And then we detail the FFT from the perspective of periodic image processing.

2.1 Central pattern generator

The robotic CPG networks are constructed by coupled oscillators. Mathematically, these can be expressed as series of coordinated differential equations. CPG can be used to design cycle gaits, with smooth gait transitions. For an n-legged robot, assume x to be the

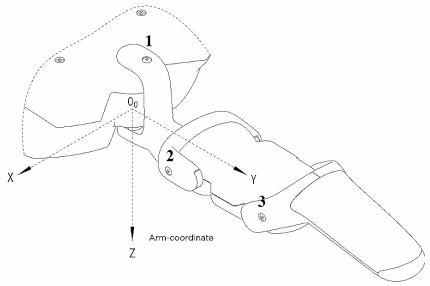


Figure 1: The structure of each leg.(https://documentation.vincross.com/Introduction/hardware.html).

angles of the shoulder joints in the axial plane, and y to symbolize the angle of joints in the sagittal plane. As shown in Fig. 1, x and

y represent the angle for joint 1 and joint 2 respectively. The base CPG model, built upon the simple circular-shaped Hopf oscillator is shown [13]:

$$\begin{cases} \dot{x} = -\omega \cdot y_i(t) + \gamma(\mu^2 - \sqrt{x_i(t)^2 + y_i(t)^2}) \cdot x_i(t) \\ \dot{y} = +\omega \cdot x_i(t) + \gamma(\mu^2 - \sqrt{x_i(t)^2 + y_i(t)^2}) \cdot x_i(t) \\ \quad + (\lambda \sum_j K_{ij} y_j(t)), \end{cases} \tag{1}$$

where μ is the radius of the limit cycle, ω is the angular frequency of the Hopf oscillator, γ indicates the forcing strength and λ is the coupling strength, K is the coupling matrix for gait type selection [10].

The limit cycle in Eq.1 is a circle. However, we can modify it into an ellipse to match the locomotion better. The modified limit cycle function reads:

$$H(x, y) = \left|\frac{x}{a}\right|^d + \left|\frac{y}{b}\right|^d, \tag{2}$$

where d affects the sharpness of the limit cycle's edges, a and b indicate the width and height for the ellipse (limit cycle), so a and b will finally influence the step length and height respectively. In order to traverse holes with different spacing, we can adapt a to the spacing. CPG will converge to the limit cycles decided by Eq. 2 (an example shown in Fig. 2) .

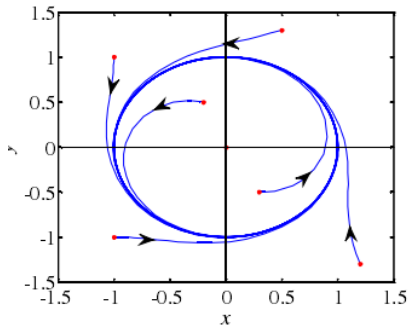


Figure 2: The limit cycle with a=1,b=1, γ=2, and ω=10

2.2 Fast Fourier transform

The Fourier Transform can convert a function (usually in the time domain) into the frequency domain. For image processing, the common choice is DFT (Discrete Fourier Transform), which is the sampled Fourier Transform that only calculates a set of samples which is enough to represent the different frequencies of a whole image [3].

Fast Fourier transform is an algorithm aimed to compute the DFT quickly. It can decompose an image into real and imaginary parts that contain a signal input information in the frequency domain. For a periodic image, FFT can easily get its frequency coefficients. So low frequencies mean the color or grey levels in the image change smoothly, while high frequencies represent faster repeated changes.

The FFT of an image and its inverse are given by[4]:

$$\begin{cases} F(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) e^{-j2\pi(x\frac{m}{M} + y\frac{n}{N})} \\ f(m, n) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} F(x, y) e^{j2\pi(x\frac{m}{M} + y\frac{n}{N})}, \end{cases} \tag{3}$$

where $f(m, n)$ represents the image sample in the spatial domain, $e^{-j2\pi(x\frac{m}{M} + y\frac{n}{N})}$ is the basis function related to F(x,y) in the frequency domain.

3 IMAGE PROCESSING

In this section, we present how the image processing works. We first present the FFT application on periodic image processing to prepare for further discussion. Since we can only get the accurate obstacle repetitions when applying FFT to a bird's eye view of the front surroundings of the robot, we need to first correct the image's perspective. We then apply the image calibration. Therefore, we present how to correct an image based on offline calibration, and then detail the contour recognition to preprocess the image for FFT. Finally, after we apply the FFT, we need to convert the main frequency in pixel to world dimensions.

3.1 Fast Fourier Transform application

In order to go through the transverse periodic holes based on visual feedback, we need to analyze the image with periodic holes [1]. That is, we describe the process in this section based on an image featuring repeated patterns. Here, we focus on an image composed of a simple sine wave that prescribes its gray scale:

$$y = \frac{\sin x + 1}{2} \tag{4}$$

where x is a square matrix where every column of it is a arithmetic progression ranging from 0 to 10π with common difference 0.1, and x means the position in pixel of the output image, y is the gray level ranging from 1 to 0. So Eq. 4 is a function with constant period 2π. In the sense of gray level, $y = 0$ means white, while $y = 1$ means black. Since Eq. 4 is a continuity equation, the image will change smoothly. Therefore, we need apply a specific threshold for Fig. 3 to ensure that y only contains 1 and 0, so that the image contains black and white only as shown in Fig. 3 . The black bars indicate the holes to stride over while the white part means ground which is safe to stand on. Because x indicates the position in pixel and the

Figure 3: The image with periodic holes.

common difference is 0.1, so the frequency in pixel is $\frac{2\pi}{0.1} = 63$. So the period of Fig. 3 is 63 pixels, i.e., the distance from the center of first black to the center of the second bar in pixel.

After applying the FFT to the image, we can obtain the frequency of the whole image. Because every column of the image owns the same frequency, we can choose the central column of the frequency matrix to plot. Finally, the single-sided Amplitude of Fig. 3 is generated (shown in Fig. 4). Note that, the frequency of the first peak (also

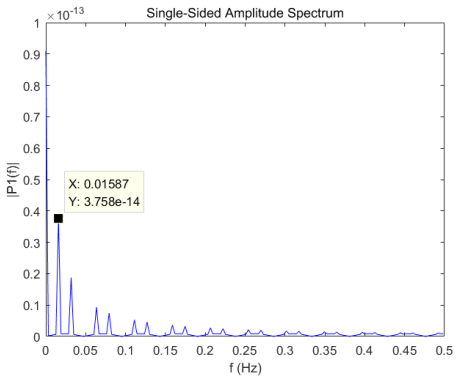


Figure 4: The image with periodic holes.

with highest amplitude) in the image is 0, which is meaningless. So it is the frequency of the second peak that indicates the period of the image. From Fig. 4, we can calculate the period, which as expected, reads:

$$T = \frac{1}{f} = \frac{1}{0.01587} = 63 \tag{5}$$

3.2 Image Calibration

Only when we have access to the bird's eye view of the holes in ground, can we get a meaningful FFT-based analysis. However, the camera is mounted on the robot so it is impossible for it to obtain bird's eye view. Under this circumstances, calibration is needed to correct the perspective as well as fix some distortions caused by the camera.

To recover the perspective, we rely on a known calibration pattern — a chessboard calibration pattern shown in Fig. 5 (b) to carry out the image calibration. We can do the calibration by matching the extracted feature points from Fig. 5 (a) to the known chessboard grid Fig. 5 (b). Moreover, the whole calibration process is an offline process, performed once to calculation the transformed matrix and then simply used online during the locomotion.

The principle of this calibration algorithm is to find the transformation matrix that maps a set of points of the view taken by the robot to the corresponding set of points of the bird's eye view from above of the flat chessboard grid [2].

At the beginning, corner extraction of the images with the two perspectives is needed. The image taken by the robot serves as the input image, while the bird's eye view is the target plane. The corner detection of these images is accomplished by the OpenCV function findChessboardCorners[8]. The reference location of detected corners is computed by the function cornerSubPix. Iterative strategy is applied in this function to find the accurate location of corners in subpixel. The location of corners extracted from the images in different perspectives were marked with white crosses within circles in Fig. 5 (a) (b).

With two sets of coordinates, we can calculate the perspective transformation matrix H between the source and the destination planes[8]:

$$\underbrace{\begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}}_{H} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = k_i \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} \tag{6}$$

x_i y_i is the corners of source image we want to transform, and u_i v_i is the corners of aimed image. The images are 2D, so the third column of coordinates is 1. H are the homogeneous matrix we want. In homogeneous coordinates, (x_i,y_i) is represented by (kx,ky,k) for any k ≠ 0.

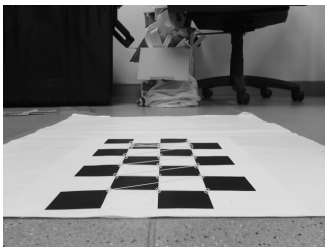
The homogeneous matrix can satisfy the minimum back-projection error:

$$\sum_i \left(x_i' - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 + \left(y_i' - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 \tag{7}$$

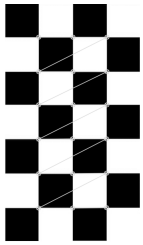
The openCV can warp the perspective of the source image by applying the specified matrix[9]:

$$\mathrm{dst}(x, y) = \mathrm{src}\left(\frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \right) \tag{8}$$

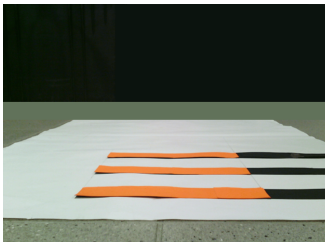
where src is the input image shown in Fig. 5 (c), dst is the output image shown in Fig. 5 (d).



(a) Chessboard grid in the camera



(b) Chessboard grid



(c) Image in the camera

(d) Image after calibration

Figure 5: Image calibration process

3.3 Contour recognition

Sadly, having a perspective-corrected image is not enough to apply our FFT approach. There are two issues related to accuracy that need consideration. The first one is the blank parts surrounding the holes shown in Fig. 5 (d), which can influence the frequency calculation. Second, the image may be misleading if it contains some obstacles on the ground beside the holes. Therefore, we need to recognize the holes and crop the image to ensure no surrounding blank. We then rely on color and contour recognition to crop the image with the four points obtained from the contours.

The whole procedure is as follows: we choose the bars in a rare color (orange in practice) to represent the holes to improve the recognition process and reduce environmental disturbance, so we can apply color recognition to figure out the holes. First, we transform the image from BGR to HSV. Because, for HSV color space, only one channel is enough to describe the color, making HSV intuitive to recognize color [11]. We then set the upper and lower boundaries OpenCV to find the aimed color-orange. To create an image that only includes holes, a mask is needed to cover the rest area that is beyond the aimed color space.

The final step is to find contours of the processed image based on the algorithm which extract the topological structure of a given binary image [15]. This algorithm is an extended version of the border following algorithm [12] which discriminates between outer borders and hole borders. The extensions are: (1) to put a unique mark on each border; and (2) to add a procedure for obtaining the parent border of the currently followed border. If a binary image is stored in the form of the borders and the surroundness relation is extracted by this algorithm, some simple image processing can be done without restoring the original image. Thus the method offers an effective way of storing binary image data.

After finding the contours of the processed image, we should range the contours to pick the vertex of the rectangle boundaries to crop the image. Fig. 6 shows the final image, after 1) perspective correction and 2) cropping.

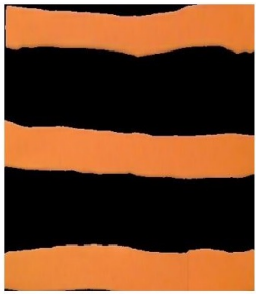


Figure 6: The cropped image.

3.4 Real-world distance computation

We finally need to convert frequencies into real-world distances. The triangle similarity says: the height of an object in the image is proportional to the distance from the object to the camera. Fig. 7 shows that a simplified pinhole camera with pinhole (P) and an image plane (I). The relationship between the parameters is given

Table 1: Focal Distance Calculation

Pixel Length	Distance(cm)	Real Length(cm)	Focal Distance
41	90	5.4	683
45	81	5.4	675
61	61	5.4	689
70	53	5.4	687
84	44	5.4	692

by following equation[7]:

$$oy = \frac{f * H}{Z} \tag{9}$$

where oy indicates the pixel length, Z is the distance from the object to the camera, H is the real height in meters, f is the focal distance. So with the knowledge of the distance to the center of the image,

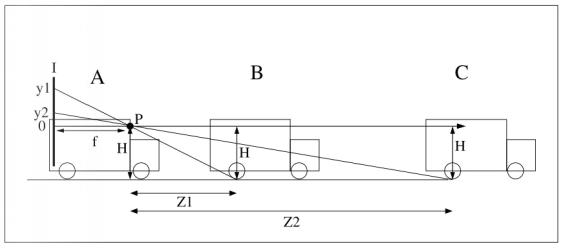


Figure 7: Diagram of the imaging geometry. (https://www.yumpu.com/en/document/read/49159361/vision-based-acc-with-a-single-camera-bounds-on-mobileye/3).

we can convert the pixel distance into a real-world distance:

$$H = \frac{oy * Z}{f} \tag{10}$$

where f can be obtained via multiple experiments (or from the camera datasheet). Table 1 records the pixel size of same object in different distance and the average focal distance is 685. The distance from the camera to the central of image depends on the photographing posture.

4 HARDWARE EXPERIMENTS

In this section, we first introduce the hexapod robot applied in this project and its characters. We then detail the main parameters of the CPG model, e.g., the coupling matrix and the CPG offsets. Finally, we compare the performance of closed-loop locomotion to open-loop locomotion.

4.1 Robot specifications

The robot used in this project is a hexapod robot shown in Fig. 8, named HEXA from Vincross. Each leg consists of three joints shown in fig 1, which contributes to great flexibility. The first joint is aligned with the yaw axis, and the intermediate and distal joints aligned with the roll axis. The robot has a camera which enables the robot to obtain onboard visual feedback, e.g., to plan paths and avoid obstacles.

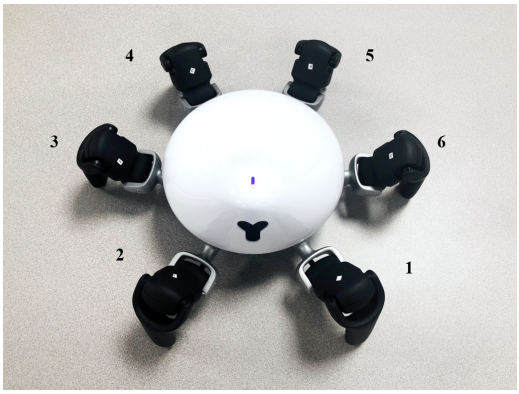


Figure 8: Numbering convention for the hexapod legs.

4.2 CPG implementation

In order to improve the performance of locomotion, we define offsets cx and cy for each leg. We use the modified limit cycle function:

$$H_c(x,y) = \left|\frac{x-c_x}{a}\right|^d + \left|\frac{y-c_y}{b}\right|^d = 1 \tag{11}$$

where c_x defines the horizontal offsets, c_y defines the vertical offsets.

By applying $H_c(x,y)$(x,y) into the original model for CPG network, we obtain the refined equations:

$$\begin{cases} \dot{x}_i(t) = -\omega \cdot \partial H_{y_i} + \gamma \left(1 - H_{c_i}\left(x_i(t), y_i(t)\right)\right) \cdot \partial H_{x_i} \\ \dot{y}_i(t) = +\omega \cdot \partial H_{x_i} + \gamma \left(1 - H_{c_i}\left(x_i(t), y_i(t)\right)\right) \cdot \partial H_{y_i} \\ \qquad + \left(\lambda \sum_j K_{ij}\left(y_j(t) - c_{y,j}\right)\right. \end{cases} \tag{12}$$

$$\text{with } \partial H_\zeta = \frac{\partial H_{c_i}}{\partial \zeta_j}\left(x_i(t), y_i(t)\right)$$

For HEXA, we set the initial offset (the center of limit cycle for each leg) to be:

$$\begin{aligned} c_{x_0} &= \begin{bmatrix} -\frac{\pi}{4} & -\frac{\pi}{4} & 0 & \frac{\pi}{4} & \frac{\pi}{4} & 0 \end{bmatrix} \\ c_{y_0} &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{13}$$

Generally speaking, these offsets will remain constant during the whole locomotion process, since they only depend on the robot's morphology. We then need to choose the robot's gait. An alternating tripod gait suits periodic holes well, because it is quasi-static. This gait is bio-inspired from the locomotion of insects; three legs stand on the ground while three legs moving. To carry out this gait, the following coupling matrix K is used:

$$K = \begin{bmatrix} 0 & -1 & -1 & 1 & 1 & -1 \\ -1 & 0 & 1 & -1 & -1 & 1 \\ -1 & 1 & 0 & -1 & -1 & 1 \\ 1 & -1 & -1 & 0 & 1 & -1 \\ 1 & -1 & -1 & 1 & 0 & -1 \\ -1 & 1 & 1 & -1 & -1 & 0. \end{bmatrix} \tag{14}$$

The rest of the CPG parameters are as follows:

$$\begin{aligned} a &= \begin{bmatrix} 0.22 & 0.22 & 0.32 & 0.22 & 0.22 & 0.32 \end{bmatrix} \\ b &= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \\ d &= 2, \gamma = 1, K = 0.3. \end{aligned} \tag{15}$$

The outputs for the CPG model only includes x and y, which are the angles of proximal and intermediate joints. The remaining flexible parameter z are angles of distal joints. Considering the trajectory of the end-effector without z is a curve, while a straight line is better to control the orientation of the robot. To avoid instability, the flexible joint angle (z) should fix this defect. It can transform the curve into line parallel to the robot's heading. The calculation can be accomplished by the inverse kinematics function given the other angles x and y.

4.3 Photographing posture setting

Since the onboard camera is rigidly-mounted, Fig: 9(a) shows that the robot can only look straight ahead rather than look down during the regular locomotion. Thus, the robot cannot easily observe the holes on the ground. Therefore, we propose to rely on a photographing behavior before every step to let the robot look down and take pictures. During the locomotion, the plane of robot is horizontal, while for photographing, we need to implement an inclined plane. The beginning and ending posture are the same for every loop. Therefore, the position of the end-effectors should keep the same to avoid slippage between the photographing posture and the ready posture.

We can derive the positions of end-effectors at the ready posture from the angles given by Eq.12. It becomes an inverse kinematics problem to determine the angles that suits photographing. However, the IK equations show that for the required position, there is a unique solution. Therefore, the slippage is unavoidable and the only solution is to reduce it. The experiments on robot and Forward Kinematic Equations help us to work out a feasible solution for photographing with small slippage shown in Fig:9(b).

(a) Ready posture (b) Photographing posture

Figure 9: Posture

4.4 Experimental results

When there is no visual feedback for the robot's locomotion, the robot will keep the locomotion with the initial parameter shown in Section 4.2. So the step length keeps 10 cm (computed by kinematic equation), because it is a open-loop locomotion. In this case, the robot will definitely step into the holes whenever the interval is not 10 cm.

We then test the performance of the robot, using our proposed close-loop controller. In this experiment, the distance between every two holes is 9.5 cm. And the robot is initially positioned at the gap between the first hole. The robot pauses to take photographs and finish the image processing in 1.5 seconds. The computational spacing result for the first image is 9.3 cm, which is very close to the ground truth. Therefore, we need to modify a of the limit cycle to ensure that the step length equals to 9.3cm. With the inverse kinematics, we can translate this length in Cartesian space to joint space. Based on the range of angles, we can get a to satisfy the required step length. So the robot is able to go through the periodic holes.

However, it is difficult to find a successful case, because this approach and robot are highly noise sensitive. Also, if the interval of holes is unsuitable, even for perfect image processing and computation, the locomotion also breaks. Full video: https://youtu.be/c417cjfTbVE

5 DISCUSSION

In this paper, we presented a FFT control for obstacle avoidance, which is generally accomplished by image processing and CPG model refinement.

The two main problems about this work are image calibration and the fixed vision system. In order to address the calibration problem, chessboard-pattern based calibration is performed. Theoretically, the smaller the grid is, the more accurate the calibration will be. However, if the grid is too small, OpenCV cannot find the remote corners of chessboard under the constraint of the images' quality. For the problem of vision system, the photographing posture can fix it. However, due to the uniqueness of ready posture, slippage is hard to avoid. So the risk of stepping into the holes is increased.

One more limitation comes from the robot. The size of footstep defined by the holes will influence the gait. With the increase of step length, the gap between two neighboring legs shown in Fig. 10 becomes smaller. But when the gap is smaller than the width of holes, the robot will definitely step into the holes.

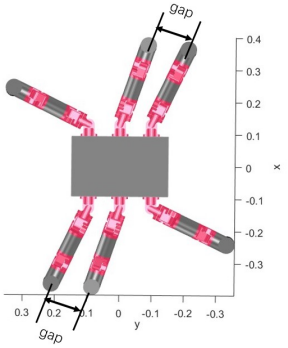


Figure 10: The gap between two neighboring legs.

So the maximal period of the holes is limited by the gap. Additionally, if the distance between holes is too small, there will be little space for the end-effector to step in. Therefore, the minimal period is limited by the width of end-effector. In our case, we experimentally found that the appropriate period should be within 9-13 cm.

6 CONCLUSION

In this paper, we present a novel method to control the locomotion of legged robot to avoid periodic obstacles based on FFT processing. We review the knowledge about CPG and FFT which are the basis of our control. For image processing, we first implement the FFT in the imaginary periodic image of holes to prove that FFT is suitable to calculate the frequency and period of image. However, to apply FFT, we need to do some pretreatment for the image taken by the robot, e.g., calibration and cropping based on the contour recognition. Finally, we need to translate the distance in pixel to the real-world distance. For CPG model, we apply suitable parameters for HEXA—the hexapod robot applied in this article. In order to take pictures of ground, we design a photographing posture. Then we modify the posture by experiments to smooth out the end-effector slips.

In order to improve the accuracy of gait adaptation, we can upgrade our closed-loop controller in the future. For example, we envision that the robot could read the color of the image central point to make up the possible error made by previous step. If the the color keeps same for every loop, means the CPG model just match the holes. However, if the color changed, we need to carry out an further analysis about the positions of holes in the image to decide the adaptation of step length, and ensure steady safe locomotion. We can also consider how to compute the interval when the frequency of obstacles changes suddenly.

REFERENCES

[1] Alessandro Bevilacqua, Alessandro Gherardi, and Ludovico Carozza. 2008. Automatic perspective camera calibration based on an incomplete set of chessboard markers. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. IEEE, 126–133.

[2] Ming-Yee Chiu and Remi Depommier. 2002. Method and arrangement for camera calibration. US Patent App. 09/912,069.

[3] James W Cooley, Peter AW Lewis, and Peter D Welch. 1969. The fast Fourier transform and its applications. *IEEE Transactions on Education* 12, 1 (1969), 27–34.

[4] Paul Heckbert. 1995. Fourier transforms and the fast Fourier transform (FFT) algorithm. *Computer Graphics* 2 (1995), 15–463.

[5] M. Heideman, D. Johnson, and C. Burrus. 1984. Gauss and the history of the fast fourier transform. *IEEE ASSP Magazine* 1, 4 (1984), 14–21.

[6] Auke Jan Ijspeert. 2008. Central pattern generators for locomotion control in animals and robots: a review. *Neural networks* 21, 4 (2008), 642–653.

[7] Yu Liu, Shuping Liu, Yang Cao, and Zengfu Wang. 2016. Automatic chessboard corner detection method. *IET Image Processing* 10, 1 (2016), 16–23.

[8] OpenCV. 2019. findHomogeneous. https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html?highlight=.findhomography#cv.FindHomography [Online; accessed 4-August-2019].

[9] OpenCV. 2019. warppersepective. https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html#void%20warpPerspective(InputArray%20src,%20OutputArray%20dst,%20InputArray%20M,%20Size%20dsize,%20int%20flags,%20int%20borderMode,%20const%20Scalar&%20borderValue) [Online; accessed 4-August-2019].

[10] Ludovic Righetti and Auke Jan Ijspeert. 2008. Pattern generators with sensory feedback for the control of quadruped locomotion. In *2008 IEEE International Conference on Robotics and Automation*. IEEE, 819–824.

[11] Adrian Rosebrock. 2014. OpenCV and Python Color Detection. https://www.pyimagesearch.com/2014/08/04/opencv-python-color-detection/ [Online; accessed 4-August-2014].

[12] Azriel Rosenfeld. 1976. *Digital picture processing*. Academic press.

[13] Guillaume Sartoretti, Samuel Shaw, Katie Lam, Naixin Fan, Matthew Travers, and Howie Choset. 2018. Central pattern generator with inertial feedback for stable locomotion and climbing in unstructured terrain. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1–5.

[14] Emily Shiffer. 2019. How Animals Shaped Our Modern World. https://www.popularmechanics.com/science/animals/g28912650/animal-inspired-technologies/?slide=1 [Online; accessed 4-September-2019].

[15] Satoshi Suzuki et al. 1985. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing* 30, 1 (1985), 32–46.