2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Feasible Region Construction by Polygon Merging for Continuous Bipedal Walking*

Chao Li, Xuechao Chen, Hengbo Qi, Qingqing Li, Qingrui Zhao, Yongliang Shi, Zhangguo Yu
Lingxuan Zhao, and Zhihong Jiang

***Abstract*— Feasible regions for continuous walking must provide necessary information for footstep planning, including surrounding landing areas and details about obstacles to be avoided during foot swing. However, the current frame lacks sufficient information to construct a feasible region needed at the current moment due to knee occlusion. To this end, this paper uses polygon merging to construct an information-complete feasible region. This polygon merging refers to merging polygons from the current frame and a specific previous frame. Since the polygon is more concise and efficient than point cloud for environmental representation, construction can be completed quickly without GPU acceleration. Experiments show that the proposed method successfully constructs informative feasible regions within the allowed time frame, enabling the robot to navigate stairs.**

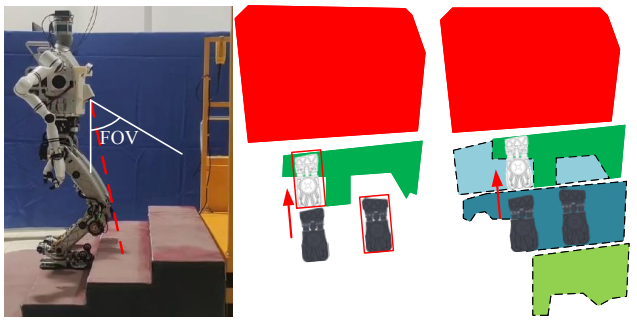
I. INTRODUCTION

Biped robots can traverse through discontinuous planar terrain due to their special structure, which is their strength. Continuous walking without pauses is crucial during navigation, as it enhances the efficiency of the robot's locomotion.

Continuous walking necessitates replanning footsteps at each step to adapt to environmental changes and correct accumulated walking deviations. The feasible region used for footstep planning must provide information about landing areas and obstacles to avoid during the swing process. Existing research on feasible region construction for continuous walking can be categorized into two approaches. The first [1, 2] only uses a current frame for construction. While this method offers adaptability to environmental changes, the feasible region is unsuitable for our robot due to knee occlusion. The second [3] collects a point cloud from previous frames, and generates a massive point cloud to extract the feasible region. However, this method needs high computational costs due to the substantial volume of point cloud data, necessitating GPU acceleration to reduce processing time. Unfortunately, GPUs are unfriendly for bipedal robots due to their high cost and weight.

This paper proposes a framework for constructing information-complete feasible regions without GPU accelerated using polygon merging. To our limited knowledge, this method is the first time used for feasible region construction of biped robots. This method addresses the problem of insufficient information caused by only using the current frame by merging a previous frame with the current frame to construct the feasible region. The previous frame is determined as follows: Convert the left and right foot positions of the next double-support phase into the pixel coordinate system of the previous candidate frame. If the left and right pixels overlap with the planar pixels of the candidate frame, the most recent frame meeting this condition is used to merge and construct the feasible region. Since the polygon information representation is more concise and efficient than point cloud, our method can construct the feasible region quickly without GPU acceleration. Our contribution can be concluded as follows:

1) A framework for feasible region construction by polygon merging is proposed, this framework can be used for robot navigation.
2) A method of polygon merging is used to construct the current informative feasible region, which addresses the problem of the current frame being unable to construct an informative feasible region. To our limited knowledge, this method is the first time used for constructing a feasible region of biped robots.
3) We verified the proposed method on a biped robot



(a) (b) (c)

Fig. 1: Constructing current feasible region through polygon merging. Feasible region extracted from current frame (captured by the robot in state (a)) is uninformative, with the polygons failing to encompass both feet in the subsequent double-support phase (highlighted in the red box of (b)). This phase is critical, as it is the initial state for footstep planning. As shown in (c), merging polygons from current frame with a specific previous frame creates a feasible region containing the necessary information for robot footstep planning.

*This work was supported by the National Natural Science Foundation of China (No. 62073041) and the Postdoctoral Fellowship Program of CPSF (GZC20233399).

All authors are with the School of Mechatronic Engineering, Beijing Institute of Technology, Beijing 100081, China chenxuechao@bit.edu.cn

BHR-8P to climb stairs and achieved good results.

II. RELATED WORK

Most works focus on footstep planning using globalmap to accomplish navigation tasks. some works express the globalmap as HeightMap, Yamamoto et al. [4] represented the environment using a stochastic Elevation-and-Normal Grid (ENG) map, where each grid cell contains elevation, normal, and the standard deviation of measurements. This representation enables the robot to navigate uneven terrain reliably following the traversability assessment. Ueda et al. [5] applied a plane detection algorithm [6] to extract planar pixels from a depth image, representing this information as 2D planar occupancy grids within the 3D space.

Some works express the environment as planar regions and execute footstep planning on those regions. Bertrand et al. [7] initially use a rotating UTM-30LX-EW to represent the environment as an OctoMap [8], then perform nearest-neighbor search and clustering within the OctoMap to extract all planar regions. Lee et al. [9] employ pixel segmentation and random sampling to extract plane information from the environment based on fused data from cameras and LiDAR. The above involve planning footsteps within a globalmap and walking according to planned footsteps. However, this approach cannot adapt to changing environments, and the accumulated deviations from walking significantly reduce stability.

Some works use the method of constructing local feasible regions at each step to perform continuous walking. Kumagai et al. [10] represent the global terrain as a HeightField and extract planar information from it. During continuous walking, the robot relies on localization to update local feasible regions and plan footsteps. However, it needs higher demands on stability control during robot walking due to limited localization accuracy, which is challenging for biped position-controlled robots. Moreover, it also cannot adapt to changes in navigation. The Florida Institute for Human and Machine Cognition (IHMC) utilizes clustering and graph search algorithms to extract planar regions from a depth image, achieving acceleration through an NVIDIA GTX 970 GPU, with the extraction of planar regions within a single frame taking only 5-6 ms [11]. Collaborating with other related work [2, 12, 13], they have completed navigation tasks for bipedal robots over rough terrains [1, 2]. However, this approach does not consider the occlusion of the area in front of the feet caused by the robot's bent-knee state and is not suitable for our robot. Bhavyansh et al. [14] use kinematic-inertial state estimates to construct planar surfaces from the planar regions of each frame. Fallon et al. [3] utilize the Kintinuous [15] algorithm to fuse stereo images and employ feature clustering to extract planar regions in the environment accelerated by a Nvidia GTX 680 GPU.

Xu et al. employ polygon merging to construct a terrain map for quadruped robot [16]. Inspired by this, to accomplish continuous walking tasks for a biped robot, we use polygon merging to build a locally feasible region, which only needs a minimal increase in computational load. This method constructs informative feasible regions at the trigger moment.

III. REQUIREMENTS ANALYSIS AND SYSTEM OVERVIEW

A. Requirements of Continuous Walking

The capability to walk continuously in discontinuous planar regions is a distinctive feature of biped robots. However, it needs higher requirements for time consumption of feasible region construction and footstep planning. To respond promptly to the changing environment, the construction of the local feasible region and footstep planning should be completed within one swing period, ensuring refreshing footsteps before lifting a foot next time.

More importantly, continuous walking requires an informative locally feasible region for footstep planning, which includes where the robot can safely land its feet and obstacle information during its swinging process. The latter helps prevent collisions between feet and the local environment. However, the knee occludes the area in front of the feet during walking, preventing the construction of a completely feasible region using only the current frame.

B. System Overview

The proposed feasible region construction framework for continuous walking is shown in Fig.2. In the perception computing unit, a LiDAR-inertial odometry fuses LiDAR data and IMU data to provide a reliable initial guess for GICP [17]. When the robot lifts its foot, it triggers an industrial camera (Helios2) to capture environmental data, generating a point cloud. This point cloud serves two purposes: one is used for registration to obtain an inter-frame pose with higher accuracy, and the other is to extract planar polygonal regions. The feasible region should encompass both feet of the next double-support phase, as this is the initial state of footstep planning. However, the polygons in the current frame do not meet that requirement, we should select a frame from the frame pool that satisfies this requirement to compensate for the shortcomings of the current frame. Generally speaking, there will be many frames that meet that requirement, we chose the most recent frame because its pose relative to the current frame is more accurate. Then, the polygons from the selected frame and the current frame are merged to construct the feasible regions, followed by footstep planning. The planned footsteps are then sent back to the control computing unit to guide the robot in its walking.

IV. RELATIVE POSE ESTIMATION

This section will introduce how to obtain a relative pose between two frames of point clouds captured by an industrial camera. Each step will be explained in the following:

A. LiDAR-inertial Odometry

LiDAR-inertial Odometry will offer an accurate initial estimate for subsequent point cloud registration, significantly speeding up the registration process. Although its information may not be as comprehensive as that provided by a

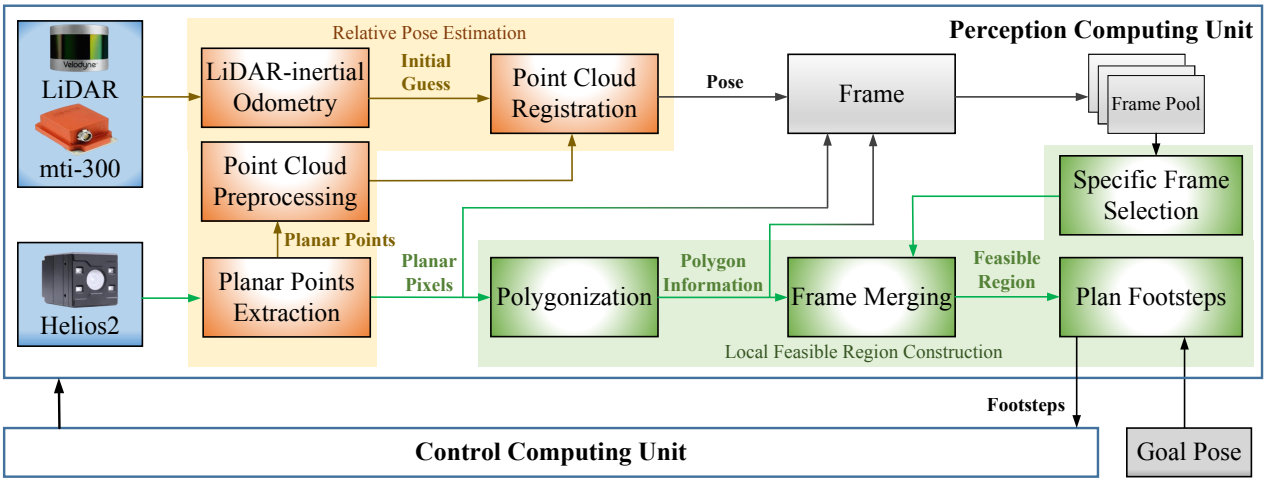


Fig. 2: The framework of our method. An efficient and robust LiDAR odometry system fuses LiDAR points with IMU data, providing a reliable and accurate initial guess for point cloud registration. When the robot lifts its foot, the Helios2 depth camera captures environmental data. This data is processed to extract planar regions from the point cloud, which are then used for registration to get the relative pose between two frames. Additionally, each frame's polygonal contours are extracted from the captured data. A specific frame's polygon is selected from the frame pool and merged with the current frame's polygon, constructing the feasible region for the robot at the moment of foot-lifting.

camera, LiDAR is widely utilized due to its higher robustness. In this paper, we employ the Velodyne VLP16 LiDAR, which has a horizontal field of view (FOV) of 360° and a vertical FOV of 45°. Additionally, it is less susceptible to environmental degradation compared to some LiDARs with smaller fields of view. To ensure the accuracy of odometry output, this paper integrates LiDAR and IMU data using Fast-Lio2 [18] algorithm, a tightly-coupled, efficient, and accurate LiDAR odometry system that utilizes the ikd-Tree data structure. The odometry output frequency of Fast-Lio2 can reach 10Hz with high precision.

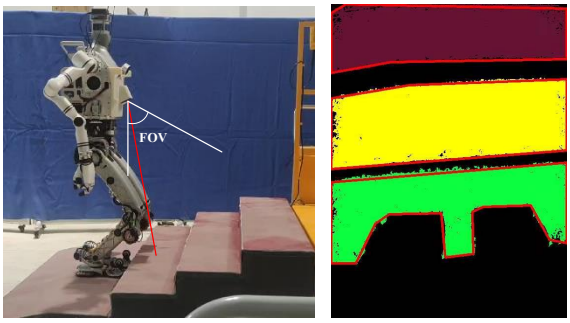
B. Planar Points Extraction

Before extracting planar points, it is necessary to use Gaussian filtering to suppress the impact of noise on the accuracy of the point cloud.

During registration, we found that the registration performance using all points from a frame is not as effective as when only using planar points. Artificial environments typically exhibit numerous planar features, planar point clouds are a relatively noise-free portion of the captured point cloud, resulting in higher registration accuracy when using only planar points. The planar points extraction algorithm is PEAC [19], the result as shown in Fig. 3.

C. Point Cloud Preprocessing

1) Voxel Filtering: The original point cloud is quite dense, significantly increasing the computational load for point cloud registration, with limited improvement in accuracy. To reduce this computational load, we employ voxel filtering to downsample the dense point cloud, approximating the data within a defined three-dimensional grid called voxels. The voxel filter grid size is set to 0.03 m.



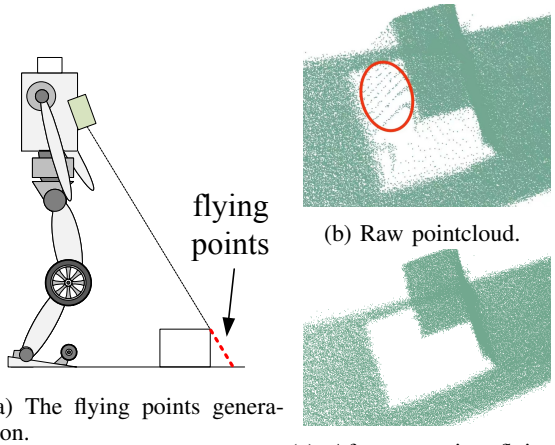
(a) (b)

Fig. 3: planar polygonal regions extraction by PEAC. (a): stairs scene, (b): the brown, yellow, and green colors represent planar points, while the red lines delineate the contours corresponding to the planar regions.

2) Flying Points Removal: Flying points can be regarded as the projection shadows of objects [7, 20], as shown in Fig. 4a. They are commonly found in point cloud data and do not correspond to any physically present objects in the scene. This can impact the accuracy of point cloud registration and may even lead to registration errors. Leveraging the characteristics of flying points, we accurately filter them out based on the following inequality condition:

$$|\mathbf{n}_p \cdot \hat{\mathbf{p}}| \leq T_f \tag{1}$$

where $\mathbf{n}_p$ indicates the normal of point $\mathbf{p}$, $\hat{\mathbf{p}}$ indicates unit direction vector of point $\mathbf{p}$, T_f is a threshold used in this inequality. The points after flying points removal as shown in Fig. 4c.



(a) The flying points generation.

(b) Raw pointcloud.

(c) After removing flying points.

Fig. 4: Example: flying points removal. The points lying within the red circle in (b) are flying points.

D. Point Cloud Registration

As point cloud registration involves iterative processes to converge towards true values, a closer initial registration value leads to faster registration speed and helps prevent being trapped in local minima. In this paper, we employ the initial values obtained from Fast-Lio2 [18] and utilize GICP [17] for inter-frame point cloud registration.

GICP is a point cloud registration algorithm widely used in robot navigation and 3D reconstruction. It refines the registration by minimizing the Mahalanobis distance between the points of one cloud and their corresponding points in another. The use of the Mahalanobis distance enhances the robustness and accuracy of the registration by considering the local surface structures.

V. LOCAL FEASIBLE REGION GENERATION

The local feasible region needs to provide information about suitable landing areas and obstacles to be avoided during swinging process for footstep planning. Unlike other robots, the bent-knee posture during walking can cause the knees to cover the area in front of the robot's feet, as shown in Fig. 3b. Therefore, we cannot rely solely on the data from the current frame to build the locally feasible regions. To address this problem, we merge the planar polygonal regions from two frames to obtain the locally feasible region at the triggering moment.

A. Polygonization

According to the paper [12, 21, 22], polygonal regions are deemed suitable for robot footstep planning based on optimization. Once the landing position for one foot is determined, the robot chooses the most suitable next footstep rather than being limited to a few specific steps based on joint constraints.

Using OpenCV [23, 24], we can extract pixel contours based on the planar pixels, where the 3D points corresponding to these pixels form the contours of a polygon. The result of polygonization is shown in Fig. 3b.

Based on the detection results, we can extract polygonal regions from each frame, each polygon includes the 3D contour points, normal vector, and root mean square (RMS). Therefore, each frame includes inter-frame relative pose and information about polygonal regions. This gathered information is then integrated into the frame pool for the frame merging process in Section V-C.

B. Specific Frame Selection

According to Section III-A, a feasible region must offer obstacle information during swinging process. This necessitates that the region's polygons encompass both feet of the next double-support phase, which serves as the initial state for footstep planning. Unfortunately, the current frame's detected polygons fail to meet this requirement due to knee occlusion, as shown in Fig. 5. Therefore, we select a specific frame to cover this shortage. The steps for selecting a specific frame are as follows: 1) transform the coordinates of both feet in this critical phase into the pixel coordinate system of the candidate frame to obtain their pixel coordinates; 2) determine whether these pixel coordinates overlap with the planar pixels of the candidate frame. A candidate frame is considered suitable for feasible region construction if the pixel coordinates of both feet overlap with the planar pixels region. The details about specific frame selection as shown in Algorithm 1.

Algorithm 1 Specific Frame Selection

Input: $\mathbf{Q_f}$: frame pool; $\mathbf{p}_f$: both feet of the next double-support phase. $\mathbf{K}$: Intrinsic parameters of the industry camera.

Output: $\mathbf{f}$: the selected frame

while $\mathbf{Q_f}$ is not empty **do**
  $\mathbf{f} \leftarrow$ latest frame from $\mathbf{Q_f}$
  transformation from the candidate frame to current frame $\mathbf{T} \leftarrow \mathbf{f}$
  planar pixels $ps \leftarrow \mathbf{f}$
  calculating the pixel coordinates $\mathbf{C}_p$ of both feet in the candidate frame's pixel coordinate system using $\mathbf{T}$ and $\mathbf{K}$.
  if $\mathbf{C}_p$ overlap with ps **then return** $\mathbf{f}$
  else
    remove $\mathbf{f}$ from $\mathbf{Q_f}$
  end if
end while

C. Frame Merging

According to the odometry of the selected frame, the selected frame can be transformed to the coordinate system of the current frame and then merged with the current frame to generate the current feasible region. The merging of two frames is essentially merging of the polygons within both

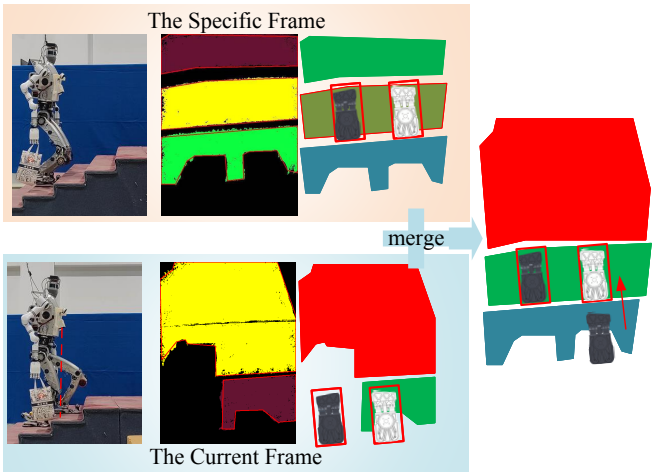


Fig. 5: Construct feasible regions. Bottom left: the robot is about to lift its right foot and step onto the next step. At this moment, the robot's knee has obscured the next step, and the polygon constructed solely based on the current image does not enclose both feet of the next double-support phase (highlighted in the red box). Top Left: by selecting the most recent frame whose polygonal regions enclose both feet at the next double-support phase, and merging it with the current frame, informative feasible regions (Right) at the triggering moment are constructed.

frames. To determine whether two polygons need to be merged, the evaluation involves analyzing the angle between their normals and the orthogonal distance between their centers, and checking if the two polygons overlap. Further details on polygon merging can be found in [16], and the result of frame merging is shown in Fig. 5. The details about frame merging as shown in Algorithm 2.

D. Destination Pose

The construction of the local feasible regions described in this paper is just one component of navigation. In the navigation process, the global guideline generated by any path planning algorithm like [13] is mapped to local feasible regions, and the goal is selected as the point within the local feasible regions that is farthest from the robot, as shown in Fig. 6.

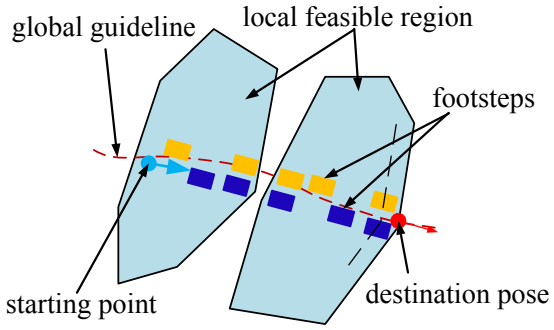


Fig. 6: Destination pose generation

Algorithm 2 Frame Merging

Input: $\mathbf{f}_s$ the selected frame; $\mathbf{f}_c$ the current frame
Output: fr current feasible region

I the number of polygons in $\mathbf{f}_s$
J the number of polygons in $\mathbf{f}_c$
$\mathbf{f}_{sc}$ is obtained by transforming $\mathbf{f}_s$ into the coordinate system of the current frame
for $i = 1 \rightarrow I$ **do**
    NO. i polygon $p_i \leftarrow \mathbf{f}_{sc}$
    $\mathbf{n}_{pi}$ is the normal of p_i
    $\mathbf{c}_{pi}$ is the center of p_i
    for $j = 1 \rightarrow J$ **do**
        NO. j polygon $\mathbf{p}_j \leftarrow f_c$
        $\mathbf{n}_{pj}$ is the normal of p_j
        $\mathbf{c}_{pj}$ is the center of p_j
        if $|\mathbf{n}_{pi} \cdot \mathbf{n}_{pj}| \geq T_n$ **then**
            if $|(\mathbf{c}_{pi} - \mathbf{c}_{pj}) \cdot \mathbf{n}_{pi}| \leq T_d$ **then**
                if p_i overlap with p_j **then**
                    the merged polygon $p_m \leftarrow p_i \bigcup p_j$
                    $fr \leftarrow p_m$
                else
                    $fr \leftarrow p_i$
                    $fr \leftarrow p_j$
                end if
            end if
        end if
    end for
end for
return fr

E. Footstep Planning Phase

1) Footstep planning: Utilizing the extracted local feasible regions mentioned above, we apply our team's developed optimization-based footstep planning algorithm [22, 25] to plan footsteps for navigating to the current destination, as shown in Fig. 6. The planning algorithm is based on greedy and heuristic optimization considering both polygon boundary constraints and robot kinematic constraints derived from its parameters.

2) Obstacle Points Calculating: During planning process, we also need to calculate obstacle points that need to be avoided during swing phase. According to the motion characteristics of biped robot, the normal vector of the swing plane is always perpendicular to the direction of gravity.

Firstly, it is necessary to calculate the normal vector $\mathbf{n}_s$ and its directed distance d_s to the origin of swing plane based on the endpoints of swing process, as well as the geometric relationship between swing plane and gravity.

The intersection between the swing plane and the polygon within the feasible region defines the obstacle points. These points are where the swing plane cuts through the contours of the polygon. Before calculating them, it is necessary to consider the polygons within the feasible region separately. The normal vector of a polygon within the feasible region is $\mathbf{n}_p$. Then, the normal vector $\mathbf{v}$ of the intersection line can be

calculated based on the normal vectors of the swing plane and the polygon.

$$\mathbf{v} = \mathbf{n}_s \times \mathbf{n}_p \tag{2}$$

After that, the endpoints of the intersection line can be calculated based on the endpoints of the swinging process.

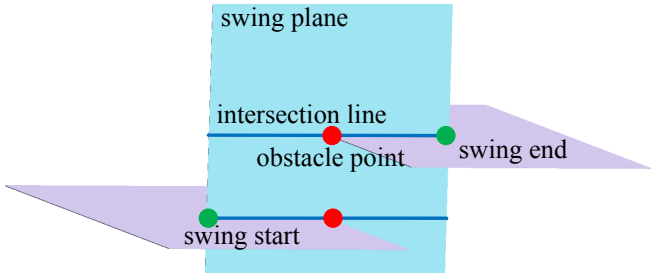


Fig. 7: Calculate obstacle points.

Based on the obstacle points calculated by all polygons in the feasible region, the foot trajectory can be planned, and the related details can be found in [26].

VI. EXPERIMENT

A. Platform

The position-controlled biped robot (BHR-8P) has 26 degrees of freedom (DoF), which are 6 DoF per leg and 7 DoF per arm. The robot weighs 75 kg and has a height of 1.65 m. The control computer is a NUC8 with an Intel i5-8259U processor, and the control rate is 250 Hz. The perception computer is a PN51 with an AMD R7-5700U processor.

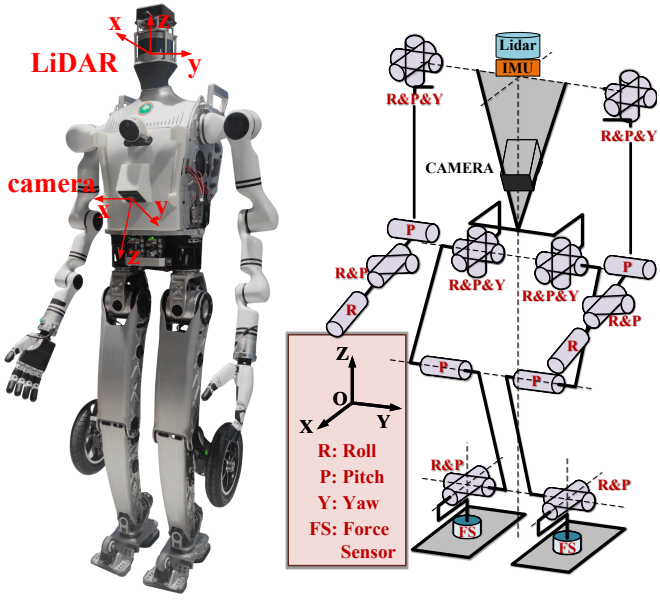


Fig. 8: Depiction of BHR-8P. (a) BHR-8P. (b) Abstracted model of BHR-8P

Due to the requirements for high accuracy and robustness for bipedal walking, we use an industrial camera (Helios2) instead of consumer-grade stereo cameras (e.g., D455) or depth cameras (e.g., L515) to collect terrain data. The Helios2 provides exceptional accuracy and robustness in diverse sun-lighting conditions. It can generate a robust 640*480 depth image with a FOV of 69°×51°. This camera can capture environmental data at a specific moment through a trigger mode, rather than at a fixed frequency (e.g., 10 Hz).

As shown in Fig. 8, three perception devices are installed on the robot: a LiDAR (VLP-16C) and an IMU (XSENSE mti-300) are mounted horizontally on the robot's head, while an industrial camera (Lucid Helios2) is installed on the robot's abdomen, tilted at 27 degrees in the vertical direction, with a mounting height of 1.02m.

B. Climbing Stairs Experiment

Walking on discontinuous planar regions is a major difference between biped robots and conventional wheeled robots. Stairs, as common indoor environments with discontinuous planar regions, are a typical scenario for robot navigation challenges. In this scenario, the stairs have a width of 1.4 m, a tread depth of 0.3 m, and a riser of 0.1 m.

During walking, the robot constructs a feasible region when lifts a foot. For our robot, the feasible region constructed only using the current frame image cannot provide sufficient information for footstep planning, as shown in the second row in Fig. 9, the polygons of the current frame do not encompass both feet of the next double-support phase. After polygon merging, the frame polygons that address this shortcoming are merged with the current frame to build a feasible region with sufficient information., as shown in the third row in Fig. 9. Polygons are a very concise and efficient way to represent the environment. They can ensure that polygon merging can complete the construction of the feasible region within the allowable duration without GPU acceleration.

The time consumption for feasible region construction is shown in Table I, the step of polygonization takes almost no time, and the time consumption for footstep planning is $203 \sim 252ms$. These indicate the sum of feasible region construct and footstep planning is less than the swing period (1.6s). Our algorithm is single-threaded and cannot be accelerated using GPUs or multithreading. We compare the proposed method's time consumption to existing approaches (Table II), our approach enables the construction of a feasible region with essential information without GPU acceleration. While the methods [11] are fast, they fail to account for knee occlusion, resulting in incomplete information about constructed feasible regions. The methods [3, 11] are unsuitable for bipedal robots because they rely on GPU acceleration, which results in additional cost and weight.

The experiment indicates that, despite the obstruction caused by the robot's knees to the area in front of its feet, polygon merging creates an informative feasible region.

VII. CONCLUSION

Constructing an informative feasible region is crucial for the continuous walking of biped robots. However, factors such as the obstruction of the frontal area by the knees make the current frame insufficient to meet that requirement for region construction. This paper used polygon merging to

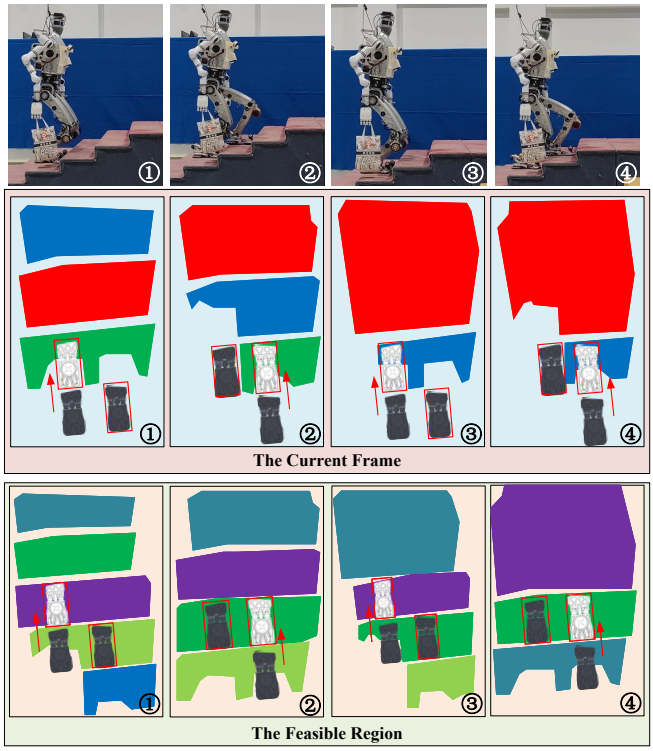


Fig. 9: Continuous walking. The feasible region would be constructed while lifting a foot, both feet of the next double-support phase are highlighted in the red box. Row 1st: BHR-8P walking up stairs. Row 2nd: a feasible region is constructed only using the current frame, which is unsuitable for our robot, as its polygons do not enclose both feet of the next double-support phase. Row 3rd: an informative feasible region is constructed by polygon merging.

TABLE I: The Time Consumption for Feasible Region Construction

Gaussian Smoothing	$60 \sim 65ms$
Planar Points Extraction	$147 \sim 244ms$
Voxel Filtering	$9 \sim 11ms$
Flying Points Removal	$2 \sim 6ms$
Point Cloud Registration	$130 \sim 240ms$
Specific Frame Selection	$3 \sim 5ms$
Polygon Merging	$5 \sim 10ms$
totoal	$356 \sim 581ms$

construct a feasible region with the necessary information by merging polygons from the current frame and the specific previous frame. Our method avoided excessive computational overhead because representing the environment as polygons is more concise and efficient. Experiments demonstrated that our method can construct information-complete feasible regions within the allowable duration without GPU acceleration, addressing the occlusion problem in the area in front of the feet caused by knee bending.

TABLE II: Comparison of the Proposed Method against Existing Perception Methods

Related Work	Time/ms	Informative	Computing Unit	
			CPU	GPU
Fallon et al. [3]	615	✓	3.30GHz	GTX 680
Mishra et al. [11]	5	×	3.50GHz	GTX 970
Ours	581	✓	**3.7GHz**	**-**

REFERENCES

[1] Duncan Calvert et al. "A Fast, Autonomous, Bipedal Walking Behavior over Rapid Regions". In: *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*. IEEE. 2022, pp. 24–31.

[2] Bhavyansh Mishra et al. "Perception engine using a multi-sensor head to enable high-level humanoid robot behaviors". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 9251–9257.

[3] Maurice F Fallon et al. "Continuous humanoid locomotion over uneven terrain using stereo fusion". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pp. 881–888.

[4] Takanobu Yamamoto and Tomomichi Sugihara. "Responsive navigation of a biped robot that takes into account terrain, foot-reachability and capturability". In: *Advanced Robotics* 35.8 (2021), pp. 516–530.

[5] Ryohei Ueda et al. "Biped humanoid navigation system supervised through interruptible user-interface with asynchronous vision and foot sensor monitoring". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 273–278.

[6] Alexander JB Trevor et al. "Efficient organized point cloud segmentation with connected components". In: *Semantic Perception Mapping and Exploration (SPME)* 10.6 (2013), pp. 251–257.

[7] Sylvain Bertrand et al. "Detecting usable planar regions for legged robot locomotion". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 4736–4742.

[8] Armin Hornung et al. "OctoMap: An efficient probabilistic 3D mapping framework based on octrees". In: *Autonomous robots* 34 (2013), pp. 189–206.

[9] Inho Lee et al. "Camera-laser fusion sensor system and environmental recognition for humanoids in disaster scenarios". In: *Journal of Mechanical Science and Technology* 31 (2017), pp. 2997–3003.

[10] Iori Kumagai et al. "Perception based locomotion system for a humanoid robot with adaptive footstep compensation under task constraints". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 713–719.

[11] Bhavyansh Mishra et al. “GPU-Accelerated Rapid Planar Region Extraction for Dynamic Behaviors on Legged Robots”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 8493–8499.

[12] Robert J Griffin et al. “Footstep planning for autonomous walking over rough terrain”. In: *2019 IEEE-RAS 19th international conference on humanoid robots (humanoids)*. IEEE. 2019, pp. 9–16.

[13] Stephen McCrory et al. “Humanoid path planning over rough terrain using traversability assessment”. In: *arXiv preprint arXiv:2203.00602* (2022).

[14] Bhavyansh Mishra et al. “Efficient Terrain Map Using Planar Regions for Footstep Planning on Humanoid Robots”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 8044–8050.

[15] Thomas Whelan et al. “Real-time large-scale dense RGB-D SLAM with volumetric fusion”. In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 598–626.

[16] Zhi Xu et al. “Polytopic Planar Region Characterization of Rough Terrains for Legged Locomotion”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 8682–8689.

[17] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. “Generalized-icp.” In: *Robotics: science and systems*. Vol. 2. 4. Seattle, WA. 2009, p. 435.

[18] Wei Xu et al. “Fast-lio2: Fast direct lidar-inertial odometry”. In: *IEEE Transactions on Robotics* 38.4 (2022), pp. 2053–2073.

[19] Chen Feng, Yuichi Taguchi, and Vineet R Kamat. “Fast plane extraction in organized point clouds using agglomerative hierarchical clustering”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 6218–6225.

[20] James Patrick Marion. *Perception methods for continuous humanoid locomotion over uneven terrain*. 2016.

[21] Robin Deits and Russ Tedrake. “Footstep planning on uneven terrain with mixed-integer convex optimization”. In: *2014 IEEE-RAS international conference on humanoid robots*. IEEE. 2014, pp. 279–286.

[22] Zhifa Gao et al. “Global footstep planning with greedy and heuristic optimization guided by velocity for biped robot”. In: *Expert Systems with Applications* 238 (2024), p. 121798.

[23] Satoshi Suzuki et al. “Topological structural analysis of digitized binary images by border following”. In: *Computer vision, graphics, and image processing* 30.1 (1985), pp. 32–46.

[24] Urs Ramer. “An iterative procedure for the polygonal approximation of plane curves”. In: *Computer graphics and image processing* 1.3 (1972), pp. 244–256.

[25] Zhifa Gao et al. “Autonomous Navigation with Human Observation for a Biped Robot”. In: *2021 IEEE International Conference on Unmanned Systems (ICUS)*. IEEE. 2021, pp. 780–785.

[26] Thomas Corbères et al. “Perceptive Locomotion through Whole-Body MPC and Optimal Region Selection”. In: *arXiv preprint arXiv:2305.08926* (2023).