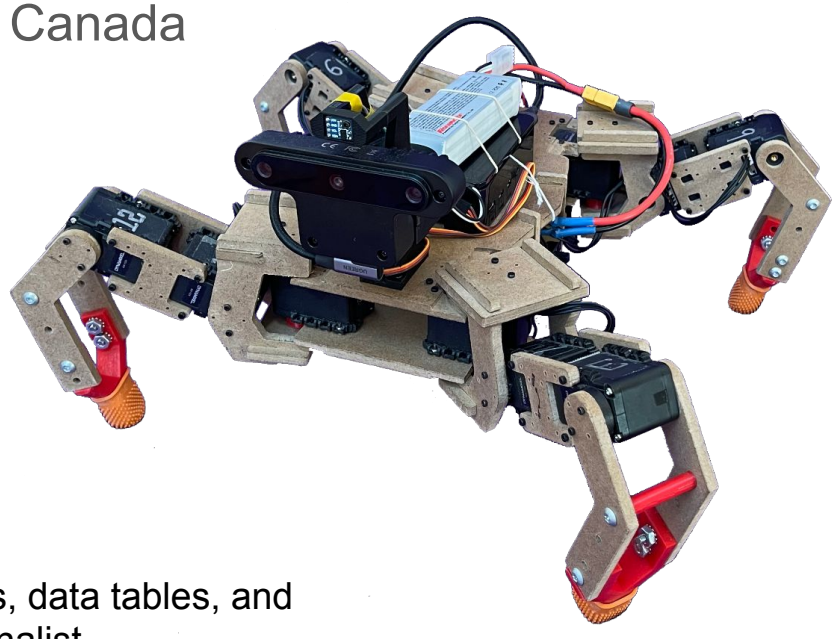
Low-Cost Quadruped Robot with Rough Terrain Traversal, Obstacle Avoidance, and Autonomous Navigation

Henry Zhao

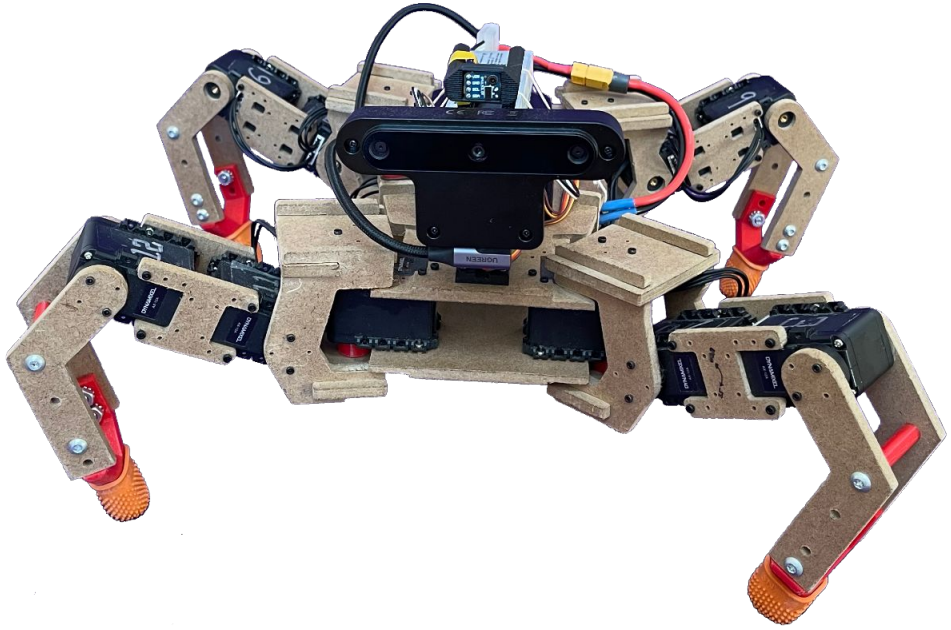
Collingwood School
West Vancouver, B.C, Canada



Unless otherwise noted, all images, data tables, and graphs were taken or created by finalist

Introduction

Legged robots have been researched heavily in past decades due to their agility and ability to traverse challenging man-made and natural terrains. Specifically, they can be used to explore human-hazardous areas, perform automated checkups and maintenance, and even act as household assistants.



The Problem

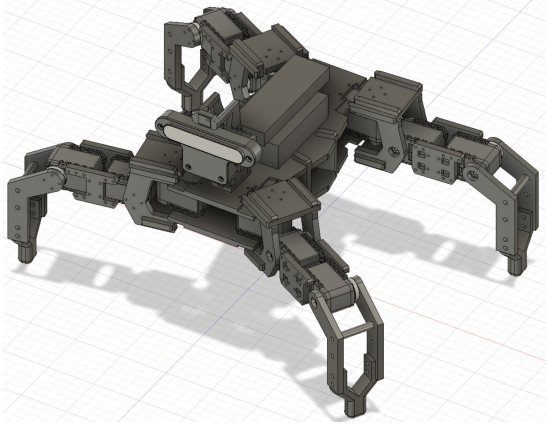
One commercial legged robot product is Spot from Boston Dynamics, which is capable of complex and lifelike movements. However, Spot is very expensive, around **$75,000**, making the cost a barrier for many non-profit or public organizations.

Image credit: extremetech.com

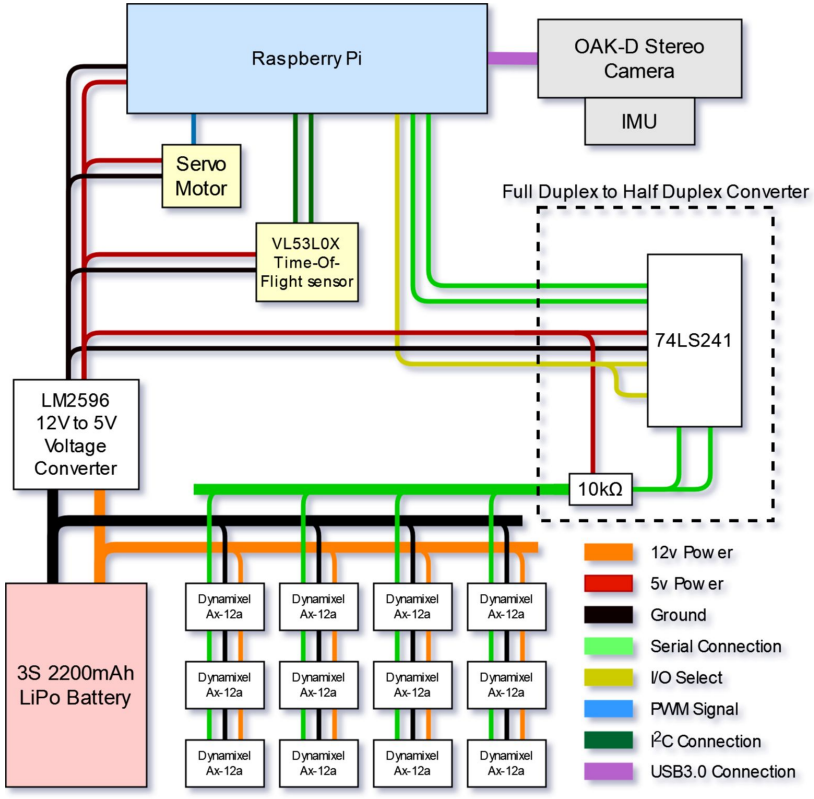
Project Goal

- Create a cost-effective walking robot
- Utilize limited sensors and stereo camera information for a multitude of functionalities:
  - autonomous movement and navigation in unfamiliar terrain
  - obstacle avoidance
  - traversal through a variety of terrain types, like stairs, slopes, and rocks

Hardware and Electronics



3D model of robot in Fusion360



Electronics diagram of robot.

Built upon last year's hardware prototype

- 12 degrees of freedom (3 per leg)
- Running on a Raspberry Pi 4

Additions:

- Oak-D stereo camera for depth perception
  - Detection range: 35cm to 38.4m
  - Outputs depth image
- VL53L0X Time-Of-Flight (LIDAR) sensor
  - Detection range: 5cm to 1.2m
  - Detects objects closer than camera's range
- MG90S servo motor to turn camera and LIDAR
  - Allows for up to 180° view angle
- Rechargeable battery module for untethered and unrestricted autonomous movement
  - 2200mAh 30c 11.1v LiPo battery with 12v to 5v voltage converter
  - Allows for running time of approx. 45 minutes.

The prototype was designed with Fusion360 and manufactured with a CNC and a 3D printer

Standing size: 36cm x 41cm x 26cm (including camera)
Weight: 2kg

Vision System and Ground Detection

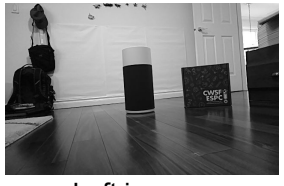
Camera Output

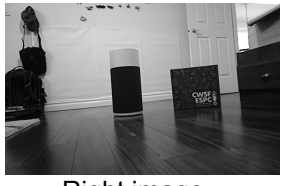
Grayscale and colour images
- For context and object recognition

Stereo depth image
- Generated by camera
- Used to reconstruct 3D scene

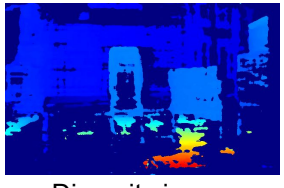
Stereo disparity image
- Used for floor detection
- Calculated using:

$$\frac{focal\ length\ in\ pixels \bullet baseline}{depth}$$


Left image


Right image

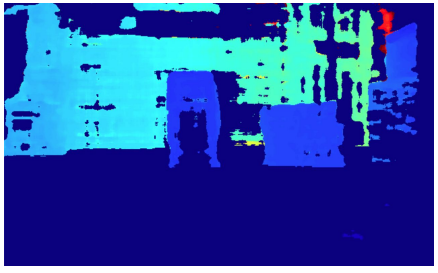

Depth image


Disparity image

Augmented Ground Plane Detection

v-Disparity


v-Disparity


Depth image with floor removed

Use disparity image to calculate v-disparity histogram:

$$v_{dj} = \sum_{i=0}^{cols} h_{ij}, \quad h_{ij} = 1 \ if \ disp_{ij} = d, \ else \ 0$$

The ground plane is represented as a slanted line in the v-Disparity image

A Hough Line Transform is used to detect the most significant non-vertical line segment, which is then mapped to a ground plane in the disparity image.

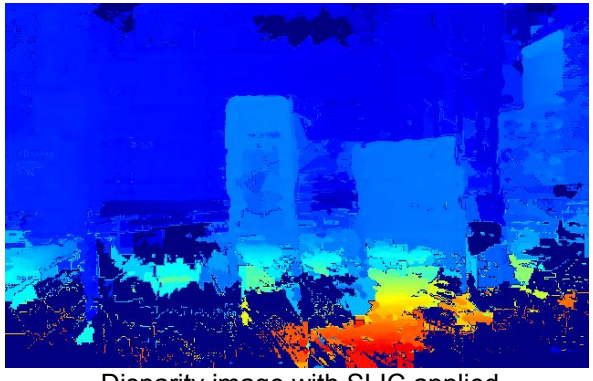
SLIC

Simple linear Iterative Clustering (SLIC) is used in the case where a solid floor cannot be found

Uses the colour image, and finds groups of similarly coloured pixels

Fills in disparity of group to be the average of the group's pixels' disparities

Helps patch up holes left by camera errors


Disparity image with SLIC applied

3D Processing

Depth to Point Cloud → Point Clouds into Panoramic View → Voxelization

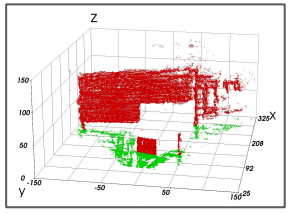
Depth image coordinates to world coordinate system:

$$X_{point} = depth_{pixel}$$

$$Y_{point} = x_{pixel} \cdot \frac{depth_{pixel}}{focal\ length}$$

$$Z_{point} = y_{pixel} \cdot \frac{depth_{pixel}}{focal\ length}$$

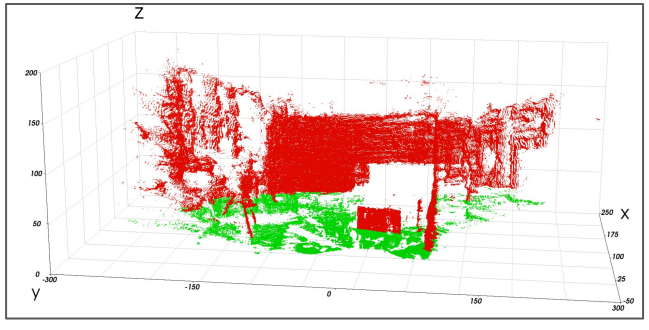
Original image



Point cloud

Obvious noise is removed by truncating points further than 5m

Left (135°) Middle (90°) Right (45°)



Panoramic point cloud

Points in each point cloud are rotated around the z-axis using rotation matrices:

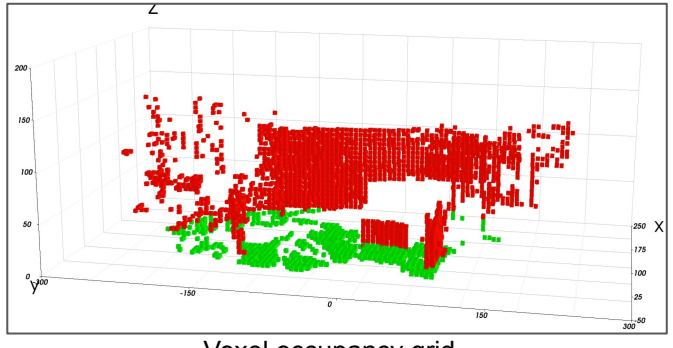
$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X_{point} \\ Y_{point} \end{bmatrix}$$

Where θ is the camera angle for the respective point clouds.

<u>Novel Fast Voxel Occupancy Grid Construction Algorithm</u>

- Points are accumulated into discrete voxels with density information.
- Outliers (based on voxel density) are detected and removed
- Adds in LiDAR information (taken every 15 degrees) to assist obstacle detection

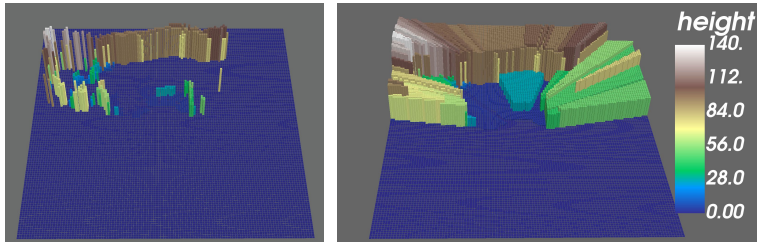
- Optimized processing through fast matrix operations
- Reduces data complexity and noise
- Results in a solidified view of obstacles



Voxel occupancy grid

Slopes and Path Planning

- Inferring Unknowns:
  - “Unknown Areas” — areas behind objects which cannot be seen by the camera
  - Height of those areas are inferred based on the height of the obscuring objects



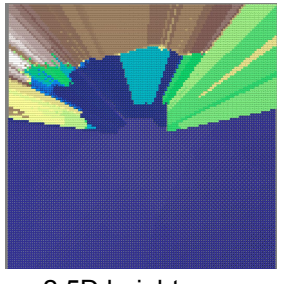
Voxel grid with unknown area inferred

- 3D Voxels turned to 2.5D height map

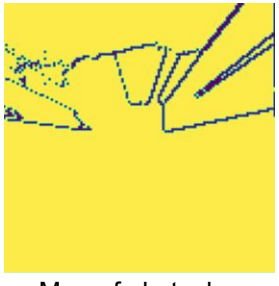
$$\text{slope}_{a \leftrightarrow b} = | h_a - h_b |$$

if $\text{slope}_{a \leftrightarrow b} > t$:

$\text{boundary}_{a \leftrightarrow b} \Leftarrow$ obstacle



2.5D height map

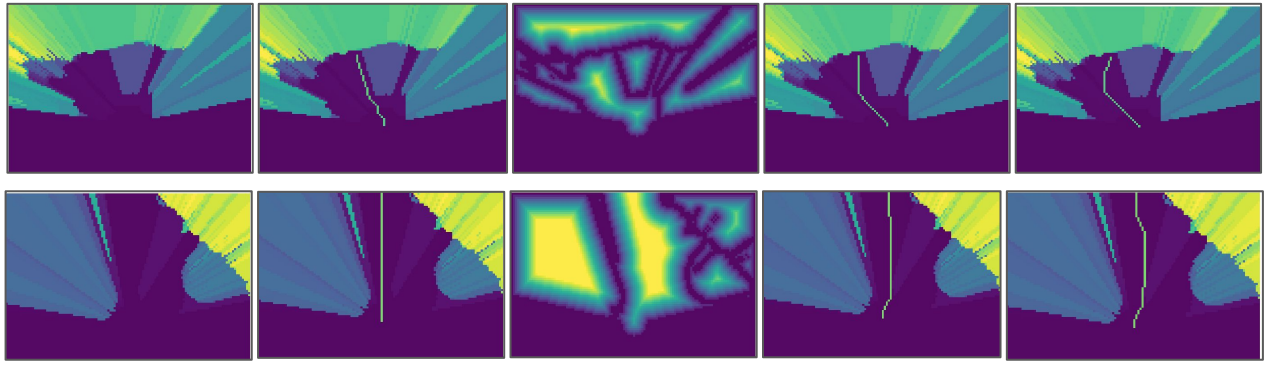


Map of obstacles

Enhanced A* algorithm

$$cost(n) = g(n) + h(n) - d(n) + c(n)$$

- $g(n)$: Path length between s and n
- $h(n)$: Heuristic cost estimation to goal
- $d(n)$: Closest distance between current path to obstacle
- $c(n)$: Slope cost on path from s to n



From left to right: 2.5D map, A* with Manhattan Distance, Enhanced A* map of distances to closest obstacle, Enhanced A* with Manhattan distance, Enhanced A* with Euclidean distance

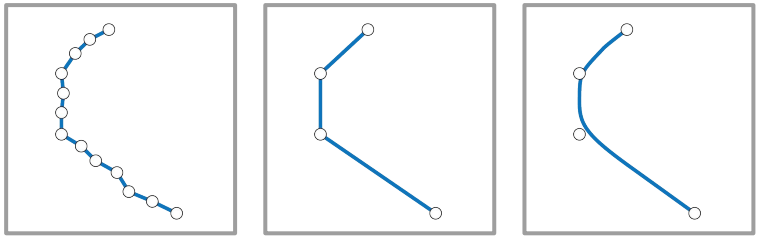
- Slope cost is calculated as 10x the cumulative change in altitude

- Parts of the path with an increase in height are marked as slopes
- Those sections are traversed using the rough terrain gait

Path Generation and Procedural Gait

Curved Path Generation:

- Utilize Ramer–Douglas–Peucker algorithm to simplify path
- Converted into bezier curves for smoothness.

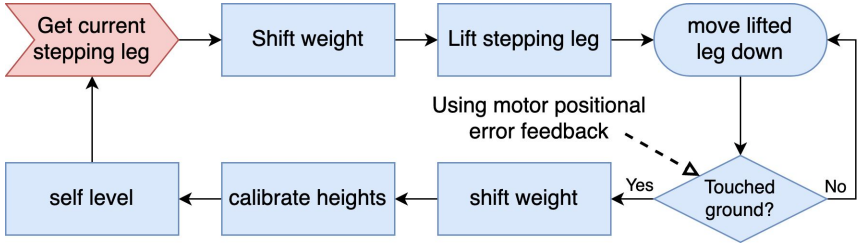


Procedural Gait Generation:

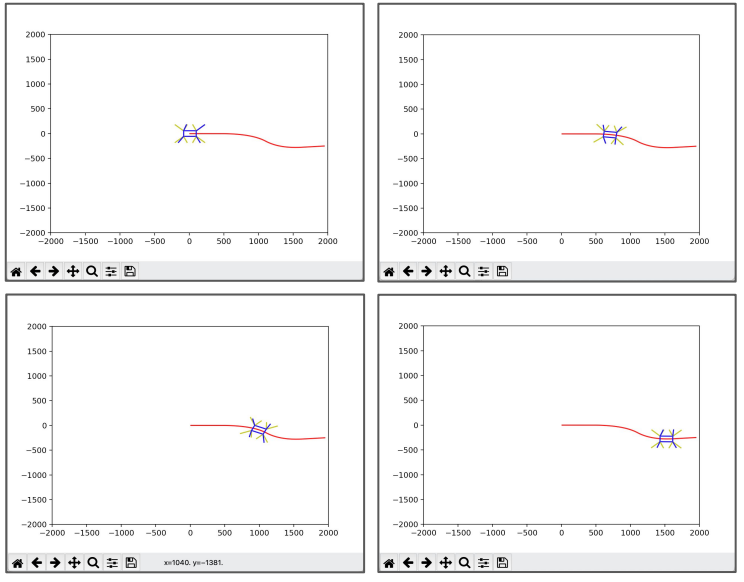
- Sets a reference leg to ensure that the robot turns without overextending a leg

- Always tries to find the optimal gait
  - Uses the longest stride, and thus, the fastest speed

- Built upon a crawling gait to maximize stability



- Visualization tool created to debug and to ensure proper gait is generated



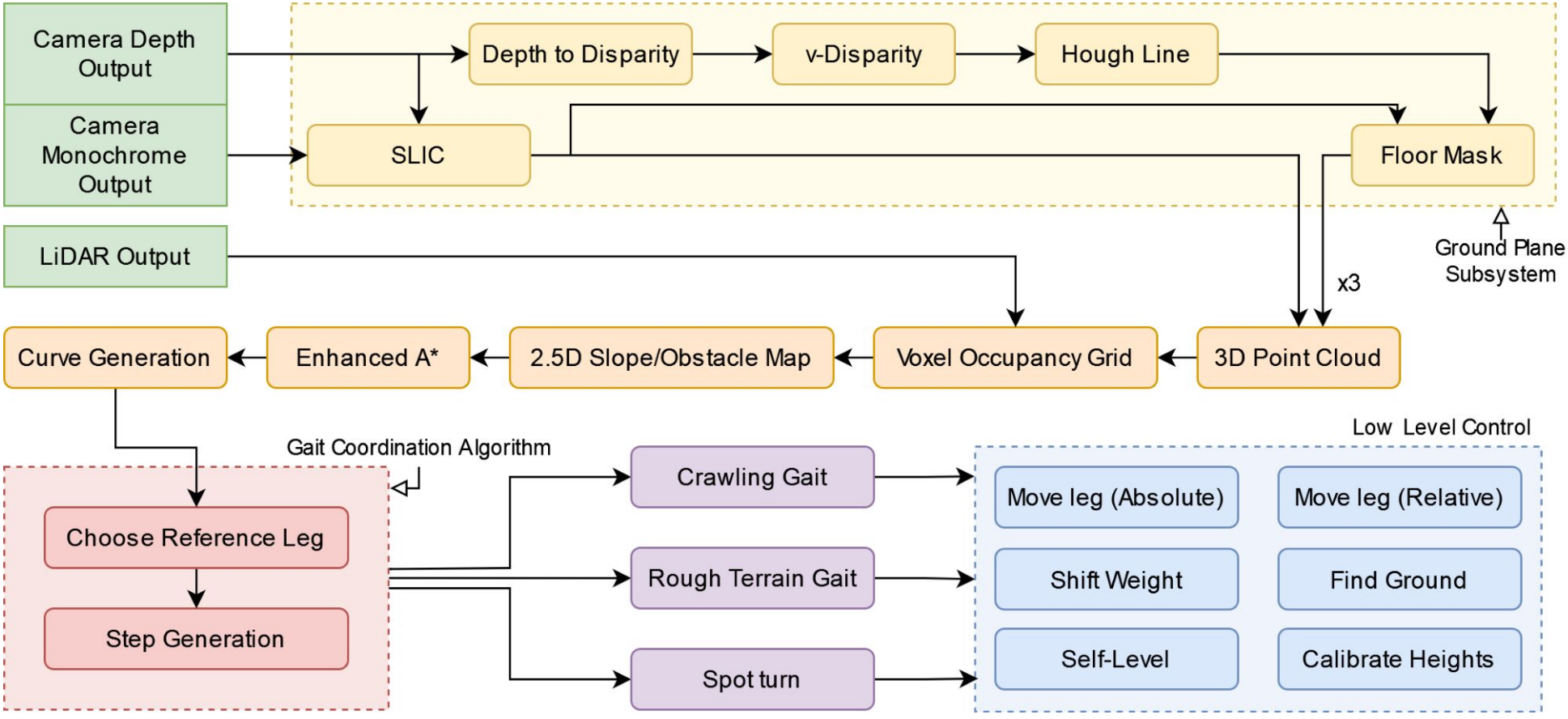
Simulated procedural gait traversing the path

Rough terrain walking system

- Uses motor's torque sensors and IMU information
  - For ground detection and self-levelling

- Ensures ground contact and stability when walking on uneven terrain

Overall Control System



Cycle Control :

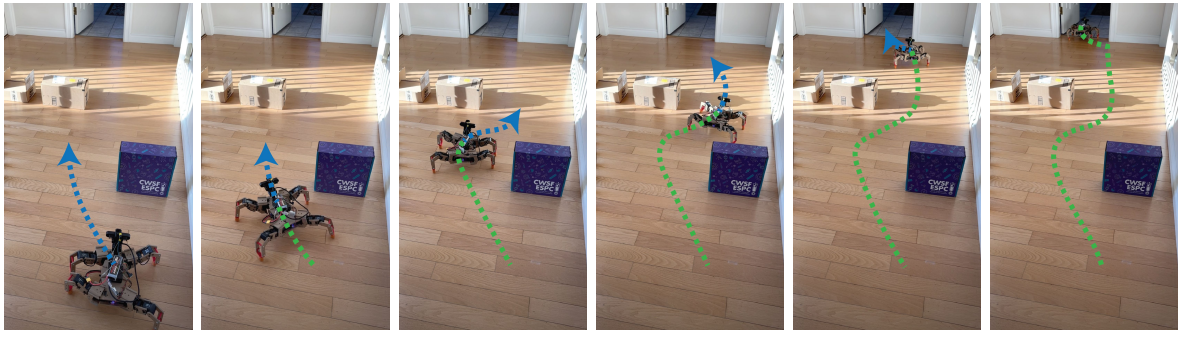
- Based on the path and terrain information, the system picks proper walking mode.
- Walks 2m between camera images
- In the case that no path is found, the system triggers a spot turn of 90° clockwise and restarts the cycle
- The system resets position and restarts the cycle under certain error conditions, for example depth pictures with too many undetected pixels.

Testing & Results

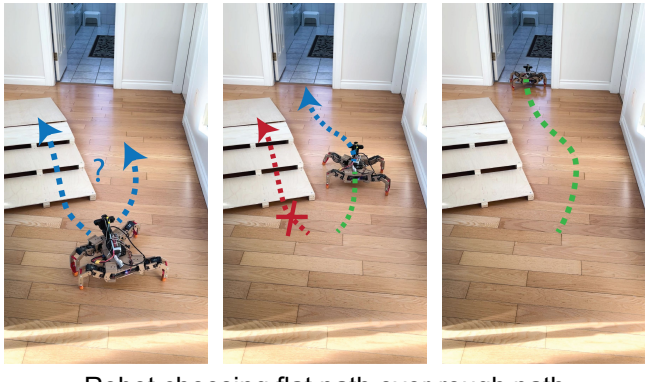
Basic instruction testing: straight line walking with various gaits, spot turn, self-levelling, ground detection.

Integrated testing: single to multi-obstacle avoidance, in a variety of path shapes, stairs and slopes traversal, flat vs. sloped path planning.

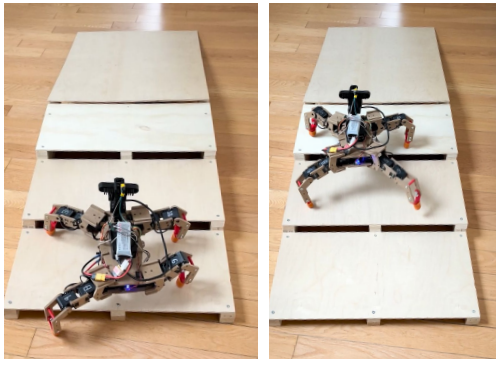
Real-world testing: Indoor and outdoor navigation.



Robot avoiding two boxes in s-shaped path



Robot choosing flat path over rough path

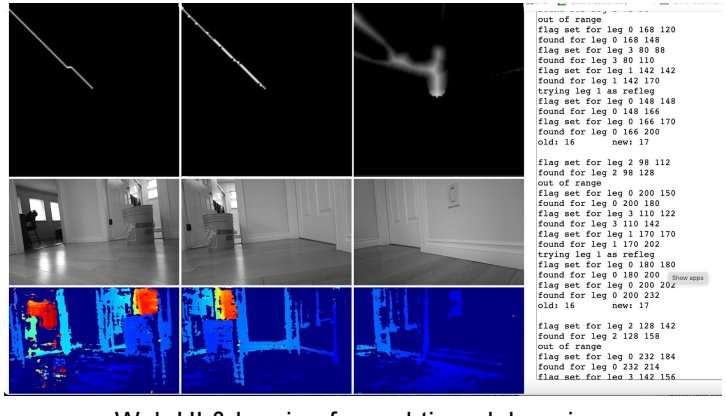


Robot walking over stairs

Robot Performance	
Trot Gait Speed	0.5m/s
Crawl Gait Speed	0.25m/s
Rough Terrain Gait Speed	0.1m/s
Max Slope Degree	16°
Max Stair Height	10cm
Straight Line Accuracy	±5cm
Spot Turn Accuracy	±6%
Vision Processing Time	2s

Testing Tools:

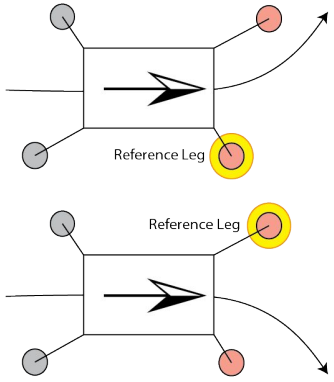
- Virtual gait visualizer
- Preconfigured images for repeatability
- Embedded testing GUI for monitoring



Web UI & logging for real-time debugging

Discussion

Notable challenges and problems:



1. During the development of the procedural gait algorithm, a problem was encountered where certain legs would overextend while walking. It was at this point that the virtual gait visualizer was created, to assist in the debugging of this sub-system. After many tests, an algorithm, where a “reference leg” would be set during turns, was implemented, and solved the problems relating to overextending legs.

2. During initial testing of the vision system, it was found that the stereo camera could not detect featureless surfaces and objects, for example, a pure white wall, as well as objects too close. This would lead to the robot attempting to walk through an object in the way. To solve this, a short-ranged LiDAR was added, to complement the camera in detecting close and featureless objects.


White wall


Object too close

3. Due to computational power constraints on the Raspberry Pi, the enhanced A* algorithm took over 10s to calculate the distance from a position to the nearest object. This greatly slowed down the vision processing time. After research into more optimized methods, a modified algorithm, employing obstacle contours and the built-in einstein sums function, was developed, resulting in a ten-fold time reduction to less than 1s on the Raspberry Pi.

$$(A \cdot B)_{ij} = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$$

To Einstein Notation

$$(A \cdot B)_{ij} = A_{ik} \cdot B_{kj}$$

To numpy.einsum(‘jk,kj->ij’...)

Current Limitations:

- At current stage, the robot can’t take continuous images, and thus cannot detect moving objects during its walking cycle.
- A local occupancy map is not generated, which could result in the robot returning to already explored areas instead of choosing a different route.
- Due to the motor’s communication speed, ground detection and self-levelling components are slow, making the rough terrain gait around 3 times slower than the crawling gait.

Conclusions

- The robot is capable of fully autonomous navigation without any prior information about the environment. It can avoid obstacles, detect sloped paths, and choose different walking modes for different terrain types.

- A vision system, as well as augmented ground detection, novel fast voxel occupancy grid construction, enhanced A* path planning and gait generation algorithms have been developed, along with an overarching control system.

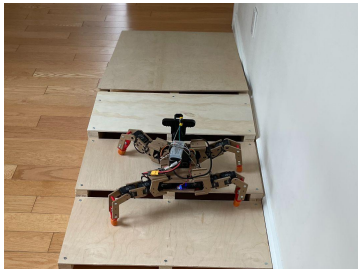
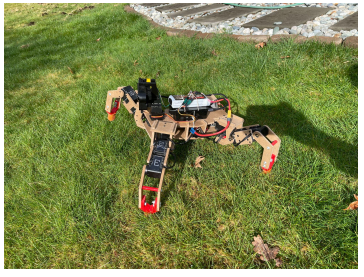
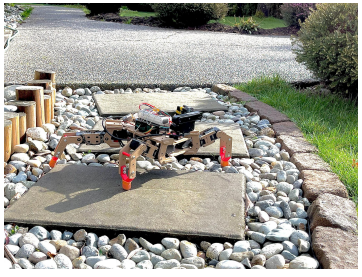
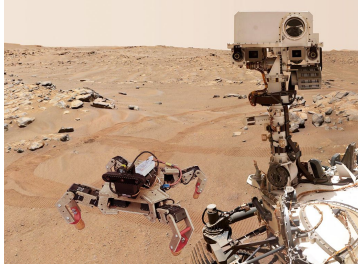
- The robot uses minimal sensors to achieve full functionality, resulting in an overall cost of **less than $700**.

Significance

- In the future, this robot can be used in hazardous terrain exploration, as well as domestic assistance.
- Can be used when a swarm of such robots are needed, for example in search and rescue.
- Affordable yet intelligent robots provide unique opportunities and features for organizations and individuals to use in everyday life.

Future Improvements:

- Ability to merge data from multiple locations and rotations for full-scale mapping
- More extensive motor and sensor feedback system - allowing for reactions in unexpected situations
- Add platform for various attachments
- Object detection and recognition for complex and dynamic tasks

Background credit: mars.nasa.gov

References

- Hassani I, Maalej I, Rekik C “Robot path planning with avoiding obstacles in known environment using free segments and turning points algorithm”. Math Probl Eng 2018(pt7):1–13
- Jun Zhao, Mark Whitty and Jayantha Katupitiya “Detection of Non-flat Ground Surfaces Using V-Disparity Images” The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems
- H. Ghazouani, M. Tagina, R. Zapata “Robot Navigation Map Building Using Stereo Vision Based 3D Occupancy Grid” Journal of Artificial Intelligence: Theory and Application (Vol.1-2010/Iss.3)
- L. Garrote, J. Rosa, J. Paulo, C. Premebida, P. Peixoto and U. J. Nunes "3D point cloud downsampling for 2D indoor scene modelling in mobile robotics," IEEE ICARSC, 2017, pp. 228-233
- H. Harms, E. Rehder, T. Schwarze and M. Lauer "Detection of ascending stairs using stereo vision," 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 2496-2502
- Le, A.V.; Prabakaran, V.; Sivanantham, V.; Mohan, R.E. “Modified A-Star Algorithm for Efficient Coverage Path Planning in Tetris Inspired Self-Reconfigurable Robot with Integrated Laser Sensor”. Sensors 2018, 18, 2585
- H. Lategahn, W. Derendarz, T. Graf, B. Kitt and J. Effertz "Occupancy grid computation from dense stereo and sparse structure and motion points for automotive applications," 2010 IEEE Intelligent Vehicles Symposium, 2010, pp. 819-824
- G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing and S. Kim "MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot", 2018 IEEE/RSJ IROS, 2018, pp. 2245-2252
- Lewis, M.A., Bekey, G.A. “Gait Adaptation in a Quadruped Robot”, Autonomous Robots 12, 301–312 (2002).
- František Duchoň, Andrej Babinec, Martin Kajan, Peter Beňo, Martin Florek, Tomáš Fico, Ladislav Jurišica “Path Planning with Modified a Star Algorithm for a Mobile Robot”, Procedia Engineering, Volume 96, 2014, Pages 59-69

Videos

Project Video:

https://youtu.be/q5yGvO9uQ58

Demo Video:

https://youtu.be/8fbxrIsXeXM