2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Variable Configuration Planner for Legged-Rolling Obstacle Negotiation Locomotion: Application on the CENTAURO Robot

Vignesh Sushrutha Raghavan[1,2], Dimitrios Kanoulas[1], Arturo Laurenzi[1], Darwin G. Caldwell[1], and Nikos G. Tsagarakis[1]

Abstract— **Hybrid legged-wheeled robots are able to adapt their leg configuration and height to vary their footprint polygons and go over obstacles or traverse narrow spaces. In this paper, we present a variable configuration wheeled motion planner based on the A* algorithm. It takes advantage of the agility of hybrid wheeled-legged robots and plans paths over low-lying obstacles and in narrow spaces. By imposing a symmetry on the robot polygon, the computed plans lie in a low-dimensional search space that provides the robot with configurations to safely negotiate obstacles by expanding or shrinking its footprint polygon. The introduced autonomous planner is demonstrated using simulations and real-world experiments with the CENTAURO robot.**

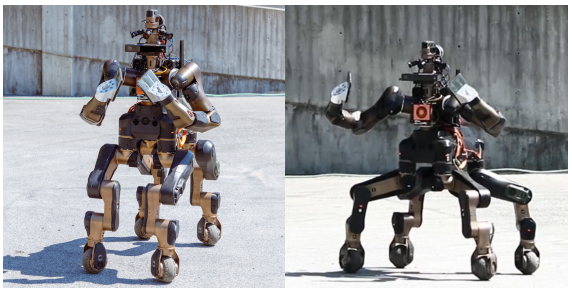


Fig. 1: The hybrid legged-wheeled robot CENTAURO, with variable configurable legs and body.

I. Introduction

In environments cluttered by various obstacles, fast sensing and accurate planning algorithms are needed for safe mobile robot navigation. Wheeled path planning through cluttered spaces may be challenging, infeasible, or computationally expensive due to limited or insufficient clear free spaces. This is particularly true when the vehicle has a fixed wheelbase configuration. A key advantage of legged robots is their capability to negotiate obstacles by stepping on or over them. However, these actions usually require additional expensive planning and caution to ensure footstep safety and maintenance of the robot stability. An advantage of hybrid legged-wheeled robots, such as CENTAURO [1] (Fig. 1), is their ability to vary their leg configuration to modify their footprint polygon, while in rolling motion, to negotiate certain obstacles by going over them as well as navigate in tight narrow spaces.

Wheeled robot navigation planning is a well explored research domain [2]. Two-dimensional navigation planners based on either costmaps or occupancy grids have been extensively developed [3] and implemented on various mobile robotic platforms, such as Turtlebot [4]. The advent of computationally fast and efficient computing hardware enabled autonomous navigation planning, using maps from various Simultaneous Localization and Mapping (SLAM) algorithms. While most of these methods work on robots with fixed footprint polygons, there is a scarcity of algorithms for wheeled robots that have a continuously changing footprint polygon during the planned navigation. Furthermore, for most heuristic algorithms, e.g. A*-based [5], all obstacle points are usually uniformly considered as non-traversable obstacle points.

In this paper, we aim to navigate through and over a series of obstacles that are: lower than the robot pelvis height, objects wider than the "standard robot polygon", and narrow spaces (by expanding the wheelbase over wide objects and narrowing the footprint into tight spaces). This will reduce or eliminate the need to take large deviations from the shortest possible path. If an obstacle is within the limits of the robot capabilities, our introduced method negotiates the obstacle rather than going around it. To achieve this, we propose a wheeled motion planning algorithm based on A*, implemented on our wheeled-legged robot CENTAURO, that takes into consideration the robot's dynamic polygon changing ability. The proposed planner also considers the trade-off between changing robot polygon configurations and travelling longer distances to avoid obstacles. We demonstrate the planner's performance in simulation and in the real-world using the CENTAURO robot to perform the aforementioned agile motions in different obstacle scenarios.

The remainder of the paper is organized as follows. After introducing the CENTAURO robot, we briefly present the related work on configuration changing robots and path planning in Sec. II. In Sec. III, we present our obstacle negotiating A*-based path planner algorithm. This is followed by simulation and experimental results in Sec. IV and finally in Sec. V we conclude with future directions.

A. The CENTAURO Robot

The CENTAURO robot is 42 DoF hybrid legged-wheeled robot, with four 7DoF legs and wheel actuators as end-effectors. We set the maximum stable width of the robot footprint polygon to be 1.1m, the minimum stable width

[1]Humanoids and Human-Centered Mechatronics & Advanced Robotics, Istituto Italiano di Tecnologia (IIT), Via Morego 30, 16163, Genova, Italy {Vignesh.Raghavan, Dimitrios.Kanoulas, Arturo.Laurenzi, Darwin.Caldwell, Nikos.Tsagarakis}@iit.it
[2] Department of Information Engineering, University of Pisa

to be 0.44m, and the maximum safe height of the robot pelvis from the ground to be 1m. The robot has a VLP-16 Velodyne sensor placed on a rotating actuator as its main perception unit on the head, providing dense laser scans of the environment as pointclouds at 40Hz.

II. Related Work

Both 2D and 3D path planners for wheeled and legged robots have been extensively studied in the literature [2], [6], [7]. Several of these methods are based on the A* algorithm [5] to achieve time efficient optimal planning results given a grid-point representation of the environment. The efficiency of A* is due to its heuristic function that estimates costs and directs the planner to look for the cheapest path from a given node to the goal. Other planners such as PRM [8] or RRT [9], and variations or improvements of A* such as ARA* [10], D* [11], have been developed depending on various required planning applications.

The aforementioned planning algorithms have been used for locomotion of re-configurable robots. For instance, studies such as [12] and [13], have made use of re-configurable robots with tracks to navigate safely in an environment, mainly by avoiding obstacles and climbing onto higher areas in the map. In [14], an arm was used to modify the position of the tracked robot's Center-of-Mass (CoM) in combination with the A* path planner. While uneven terrain, stairs, and slopes were considered, the solution presented did not change the robot footprint polygon. Recently in [15], a modified A* algorithm was presented to modify the shape of a cleaning robot and maximize the cleaning area. While the polygon shape is changed to get through narrow spaces, the planner only avoided obstacles obstructing the floor space by going around them. In [16], a snake-like robot was used to climb stairs or reshape to move around objects, taking advantage of its modular multi-part nature. A "follow the leader module" approach and a computationally rather heavy 6DoF manipulator RRT* planner was used. Due to the leader-follower approach the polygon change was not very evident, even though the robot was capable of climbing stairs and going through narrow spaces. In [17], a multi-modal PRM was used as the simulated planner for the ATHLETE and the HRP2 robots, to determine robot polygon and configurations on flat terrains, undulating terrain, and stairs. The ATHLETE robot which also used a legged wheeled motion similar to the CENTAURO robot, was tested on simulated surfaces. The introduced planner was computationally heavy, while obstacle avoidance and negotiation were not addressed in the simulations presented.

Recently, an ARA* [10]-based planner that combines driving and stepping motions of the MOMARO and CENTAURO robots was presented in [18], [19]. A set of motions, such as one-foot stepping, longitudinal base-frame forward shift, and front/back wheels drive forward/backward, was generated and a hybrid stepping driving motion was executed. With this method the MOMARO and CENTAURO robots were able to climb on to raised surfaces using sequences of driving and stepping motions. While these works execute a longitudinal robot polygon change while climbing and going over short/thin obstacles, they don't modify the robot polygon to negotiate wide obstacles and narrow passages using the legged-wheeled driving motion capability of the CENTAURO robot. In legged-only locomotion for a hexapod, the work presented in [20] introduced a planner that used a deformable bounding box as the model of the robot, to modify the height and width of the robot to avoid overhead and floor obstacles, and narrow spaces. This solution focused on collision avoidance for legged locomotion, whereas our work also considers the cost of changing configurations on a legged-wheeled robot and presents a flexible planner that navigates based on these costs.

In particular in this paper, we focus on wheeled motion planning, applied on the CENTAURO robot. We present a low dimensional search-space planner based on the A* algorithm for legged-wheeled motion, that is capable of determining the extent of 1) robot polygon expansion to go over wide objects and 2) polygon shrinking to fit through narrow paths. This is done by combining costmaps and ground plane filtering extracted from the Octomap [21] environment representation, with 2D image-based obstacle segmentation and safe robot polygon search. Notice that A* (vs. other planners for e.g. PRM/RRT) was selected due to good computational complexity and its flexibility for testing different cost functions.

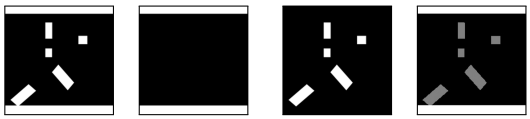
III. Obstacle Negotiating A*

In this section, we describe the steps involved in the execution of the variable configuration A* path planner algorithm demonstrated on the CENTAURO robot, to negotiate obstacles and narrow spaces. First, we generate a map which differentiates between obstacles that can be negotiated by the robot and obstacles to be avoided (Sec. III-A). Following this, we introduce the modified A* planner that considers the robot footprint polygon changes (Sec. III-B) needed for collision-free navigation in the environment (Sec. III-C). This planner uses as its input, the 2D map images created by a segmentation module from the acquired VLP-16 Velodyne's pointclouds.

A. Rough Height-Based Obstacle Segmentation

Laser pointclouds, from the rotating VLP-16 Velodyne, are the main perception input for the algorithm. The pointclouds are accumulated into an Octomap [21], which processes the points by segmenting and filtering out the ground plane. The non-ground points are sent to two costmaps generators[1]. The first costmap considers all points in the filtered pointcloud as obstacles. The second costmap considers only points above a threshold height h_{obs} as obstacles; in our CENTAURO experiments we set this value to $h_{obs} = 0.4$m as a conservative estimate of the height of obstacles easily cleared by the base of CENTAURO. The obstacles are also inflated to a radius that equals half the wheel width for collision safety; in our experiments we considered 5cm of obstacle inflation. The

[1] http://wiki.ros.org/navigation



(a) Image I_a (b) Image I_b (c) Image I_c (d) Image I

Fig. 2: The segmentation process: (a) I_a: all obstacle points, (b) I_b: only tall obstacles i.e the side walls (in this case $I_b = I_d$), (c) I_c: short clearable obstacles, with height less than h_{obs}, (d) I: segmented image map.

two 3D costmaps are then converted to grayscale images—I_a and I_b, respectively—by 2D projection with a resolution of half the robot wheel-size, i.e. in our case 5cm per pixel.

In the resulting greyscale image, the white pixels represent obstacles and the black pixels represent the free space. Given the I_a and I_b greyscale images, a bitwise XOR operation is performed on them. The output is an image $I_c = I_a$ XOR I_b, that roughly consists of obstacle points of height lower than h_{obs}, which can be easily cleared by the robot pelvis. Using the greyscale images I_a and I_c, we generate a fourth image $I_d = I_a - I_c$ to represent only the non-clearable obstacle points. Finally, we create a combined image $I = 0.5 \times I_c + I_d$ (the *segmented image map*), which classifies every point on the map as a free space (black), clearable obstacle (grey) or non-clearable obstacle (white) using 3 colour codes. This image is used as the main 2D map for the introduced variable configuration path planning development and its pixels form the planner's query points. An example of this segmentation on simulated data can be seen in Fig. 2.

B. Robot Polygon Symmetry

At each path planner point of the generated segmented image map, it is necessary to find the shape of the robot polygon that ensures a collision-free motion from a start to a goal point. Hybrid robots like CENTAURO, can expand their polygon, turn in place, and reduce the pelvis height with respect to the ground. So in essence, we require at least a 4DoF footprint polygon calculation assuming that the robot's pelvis roll and pitch are maintained to be zero with respect to the ground. To reduce the complexity of the calculations and to also ensure robust safety of the robot while navigating in close proximity to obstacles, we assume a rectangular symmetry for the robot footprint polygon. Furthermore, the height of the pelvis above the ground is determined by the expanded width of the robot.

We set the sum S_p of the robot footprint polygon width (w_r) and length (l_r), i.e. $S_p = w_r + l_r$, to a constant value based on the limits of the leg joints of the robot. Thus, the length of the robot footprint polygon l_r can be simply determined as: $l_r = S_p - w_r$. Let the maximum and minimum possible robot heights and widths be (h_{max}, h_{min}) and (w_{max}, w_{min}) respectively. The height h_r of the robot for any given footprint polygon is determined by using simple proportions based on the robot footprint polygon width w_r by the following formula:

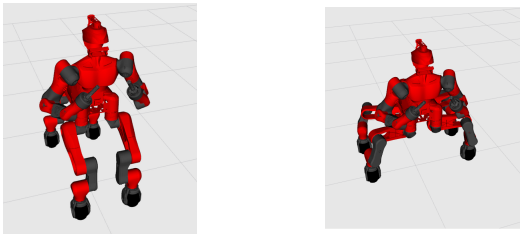


Fig. 3: CENTAURO robot configurations, depicting the most narrow (left) and widest (right) possible footprint polygon.

$$h_r = h_{max} - \frac{(w_r - w_{min})}{(w_{max} - w_{min})} \times (h_{max} - h_{min}) \qquad (1)$$

Using the above imposed symmetry, the calculation of the robot footprint polygon at every instance of planning is now reduced to determining the robot polygon width w_r. Fig. 3 shows the narrowest and the widest possible extreme configurations for the CENTAURO robot, defining the search space used by the introduced planning algorithm.

C. Robot Polygon Calculations for Obstacle Negotiation

From the segmentation module introduced in Sec. III-A, we obtain the image coordinates of obstacles which either can be negotiated by the robot by going over them or they cannot be negotiated (e.g. walls and tall objects) and need to be avoided.

The planner evaluations are based on the segmented image map. In other words, we examine pixels (i.e. image points) for all planner evaluations. The image pixels will henceforth be referred to as *points*. We have three types of points: 1) free space points, 2) points of the negotiable obstacles, and 3) points on obstacles to be avoided. On the other side, the robot can either expand its footprint polygon while lowering its height, narrow its footprint polygon while increasing its height, or maintain its previous configuration.

At a negotiable obstacle point, the robot needs to expand so that the wheels go on either side of the obstacle to successfully go over it. If the point is a free space point, then the robot has three options: 1) it can expand or narrow, if it is near negotiable obstacle points, 2) it can narrow the polygon, if it is near non-negotiable obstacle points, and 3) it can maintain the previous configuration, if the free space point is not near any of the obstacles. The calculations for changing the configuration for each of the above mentioned options will be explained below.

To be capable of safely going over the negotiable obstacle point, we need to determine the robot's footprint polygon width that would allow it to pass over the object. When a point is examined, a small rectangular region of the size of the maximum width (w_{max}) and maximum length (l_{max}) of the robot footprint polygon, centered at the query point is considered. Since the operation uses image points, the number of orientations for motions of the robot pelvis, while moving from a given point to its neighboring points, is 8 (i.e. the total number of immediate image point neighbours). All 8 orientations are searched for valid configurations, and

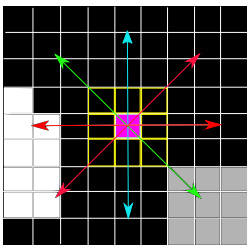


Fig. 4: A search strategy instance. The pink block is the evaluated pixel/point, while the yellow squares and the corresponding arrows represent its 8 neighbors and directions to which the robot will attempt to move to. The red arrow represents possible collision with non-negotiable obstacle, the green represents a collision with negotiable obstacles, and the blue represents the direction with no-collisions.

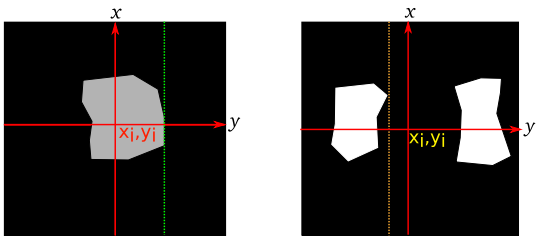


Fig. 5: The process of polygon width selection for two cases: expanding over a negotiable obstacle (left) and narrowing between two obstacles (right). In the left image, the green line is the maximum y-coordinate of the robot polygon expansion to safely negotiate the object. In the right image, the orange dotted line is the minimum y-coordinate of the robot polygon shrinking to safely go in between two obstacles.

this is intuitively illustrated in Fig. 4. In particular, consider the examined negotiable obstacle point with coordinates $p_i = (x_i, y_i)$ and angle θ with respect to the horizontal axis. All the points in the rectangular Region-of-Interest (RoI) centered at p_i, are rotated by θ and normalized such that a relative origin lies at p_i. In other words, the coordinates of the points in the RoI are obtained with respect to p_i and thus, are represented exactly as the robot sees them when oriented by θ at p_i. If the RoI consists of only negotiable obstacle points, we find the maximum absolute horizontal pixel coordinate (i.e. the relative y-component) from the normalized coordinates of the obstacle points. This determines how much the robot has to expand to go safely over the obstacles. This process is repeated for all the negotiable obstacle points and all θ's, at the start of the algorithm.

If the point is a free space point, with the RoI having only negotiable obstacle points, we consider the options of both expanding or narrowing the polygon. Expanding is done as described in the previous case. For narrowing, instead of finding the maximum horizontal coordinate, we find the minimum horizontal coordinate. This determines how much the footprint polygon needs to be compressed to can avoid all the obstacle points. This is repeated when the RoI around the free space point has non-negotiable obstacles. Fig. 5 depicts the selection of the width w_r of the robot for the expanding and narrowing cases. If the minimum or maximum horizontal coordinate values fit within thresholds set by the narrowest and the widest robot footprint polygons, then the free space point is traversable. Similarly, the points on negotiable objects are considered traversable for a given θ, if the maximum horizontal coordinate is within the thresholds. Otherwise, for that given θ the point is considered non-traversable.

D. Incorporation into the A* Algorithm

Having determined the robot footprint polygon calculations for obstacle negotiation given a segmented image map, we incorporate these into the A* path planner. To determine the path with the lowest cost from a start to a goal point in A*, two functions are used: 1) the function to calculate the cost of reaching the node $g(x,y)$ and 2) the heuristic measure $h(x,y)$, which is usually the distance of the query point to the goal point.

Consider $[x_c, y_c]$ to be the child neighbour node being queried, $[x_p, y_p]$ the corresponding parent node, and $[x_g, y_g]$ the goal node. If the point related to the $[x_c, y_c]$ node is close to obstacles or it is on a negotiable obstacle, then the θ angle corresponding to the motion from the parent to the child node is checked for a valid available configuration. In such a case, the costs will be evaluated, otherwise the path will be ignored. For our algorithm, the function $g(x,y)$ will be calculated as follows:

$$\begin{aligned} g(x_c,y_c) &= g(x_p,y_p) + W_t \times \frac{|\theta_{ent} - \theta_{ext}|}{2\pi} + W_c \times (|\delta w|) \\ \theta_{ent} &= acos\left(\frac{x_p - x_c}{\sqrt{(x_p-x_c)^2 + (y_p-y_c)^2}}\right) \end{aligned} \tag{2}$$

where $||$ is the absolute value, W_t is the weight on the turning cost, W_c is the weight on the cost of configuration change, θ_{ext} is the angle at which the robot exits the node prior to the parent node $[x_p, y_p]$ or the angle at which it enters the parent node, θ_{ent} is the angle at which the robot enters the child node $[x_c, y_c]$, and $(|\delta w|)$ is the change in robot footprint polygon width that is required for obstacle collision-free negotiation at node $[x_c, y_c]$ and it is divided by the maximum permissible horizontal robot axis (width) change.

The heuristic function $h(x_c, y_c)$ is the standard Euclidean distance as follows:

$$\begin{aligned} h(x_c,y_c) &= \sqrt{(x_g-x_c)^2 + (y_g-y_c)^2} + W_g \times \frac{|\theta_{goal} - \theta_{ent}|}{2\pi} \\ \theta_{goal} &= acos\left(\frac{x_g - x_c}{\sqrt{(x_g-x_c)^2 + (y_g-y_c)^2}}\right) \end{aligned} \tag{3}$$

where W_g is the weight on the cost of turning away from the direction of the goal. Intuitively, by having higher weights on θ's or turning motions in both functions, we force the planner to prefer paths with few turns to reach the goal.

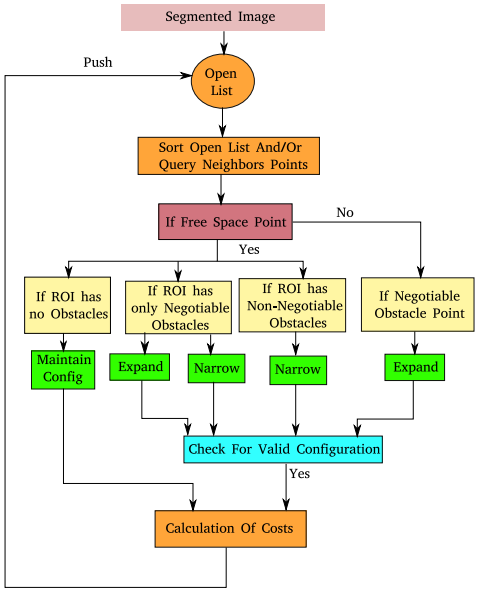


Fig. 6: Obstacle Negotiating A* block decision diagram. The blocks in orange are parts of the standard A* algorithm whereas the rest are the modifications presented in this paper.

This has an additional advantage that in front of negotiable obstacles, the algorithm would prefer to expand and move forward rather than turning away from the goal without a configuration change due to the lower cost. Furthermore by increasing W_c, we can force the robot to look for paths with fewer configuration changes.

The total cost of the node is the sum of the two functions $h(x,y)+g(x,y)$. Following this, the standard A* planner steps are followed (using a sorted open and closed list of nodes to explore the map), such that a path with a lower cost and a shorter travel distance is followed. The overall block diagram representing the sequence of steps can be seen in Fig. 6.

On the implementation level, each *node* in the planner is represented as a structure, which consists of the following variables:

- The x and y coordinates of the point.
- A boolean array of 8 elements, indicating whether at the θ angle that corresponds to the element, a valid robot footprint polygon configuration exists.
- A double array of 8 elements, indicating a valid robot width w_r to the corresponding θ of the element.
- A boolean to indicate whether the point is an obstacle.
- A boolean to indicate if the point is traversable.
- The associated costs of the node.

IV. Experimental Results

In this section, we present the experimental and computational results with our planner, on the simulated and real CENTAURO robot.

A. Simulation Results

We first evaluate our obstacle negotiating A* planner on multiple image-based simulations (Figs. 7-9). The runs were performed in a generated scenario with multiple low-height objects of different sizes within a 8.5m wide corridor.

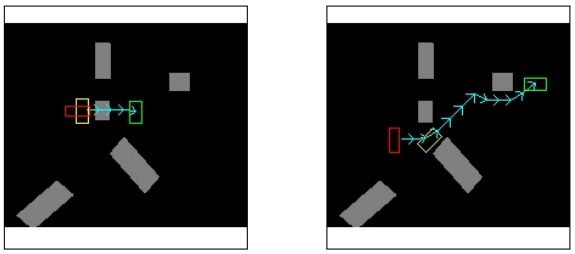


Fig. 7: Two evaluated scenarios of path planning with footprint polygon expansion and narrowing. On the left, the robot is expanding before a wide object to continue towards the goal. On the right, the robot initially in an expanded configuration, is narrowing to go between two objects to reach the goal.

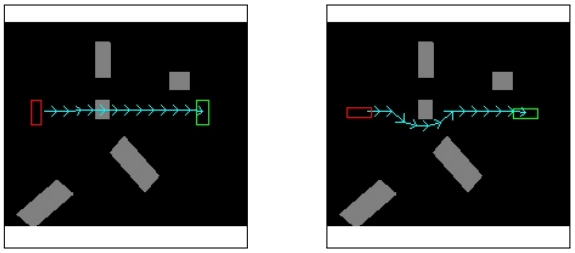


Fig. 8: Two cases of planning decisions based on the robot's starting configurations. On the left, the robot starts in the expanded configuration, and thus a straight path over the obstacle is planned. On the right, the robot starts in the narrow configuration, and chooses against expanding, instead going through the narrow gap taking advantage of its configuration.

As described in Sec. III-A, we obtain two images, one consisting of points below the height h_{obs}, and the other with tall obstacles. Finally, we obtain a rough segmented image map (Fig. 2), where grey represents small/negotiable obstacle points, black represents free space points and white represents tall obstacles.

Fig. 7 depicts the general capacity of the introduced planner to expand and narrow the footprint polygons en-route to the target. In the figure, the yellow rectangle represents the change in robot configuration. The red rectangle and the green rectangle represent the start and end robot positions and configurations respectively. The blue arrows depict the direction and path of robot motion.

An instance of subtle effectiveness of the planner can be seen in the Fig. 8. The robot chooses to go over the object as the robot is already in an expanded configuration. On the other side, when the robot starts in a narrow configuration, it chooses to go around the object as the turning costs are comparatively less than the action of expanding over the object.

To illustrate further the effectiveness of the planner, it is executed with different weights W_c, on the robot configuration costs. The starting and goal positions considered can be seen in Fig. 9, where we show the plans with two different weights on the robot configuration change. Table I summarizes the comparative results of: 1) planning computation times (note that the implementation is not-

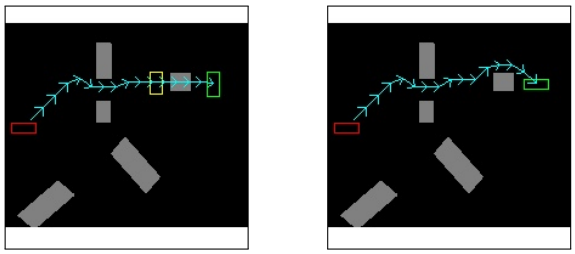


Fig. 9: Two planning cases of varying configuration change weights. On the left, $W_c = 3$, while on the right the weight is higher, i.e. $W_c = 7$.

TABLE I: Image-Based Planner Simulations Results

Configuration Change Cost (W_c)	Final Planner Cost	Path Length (m)	Planning Time (s)	Plan Execution Time (s)
1	160.06	9.072	1.88	69.282
3	162.67	9.072	2.73	69.282
5	164.20	9.072	3.81	69.282
7	161.00	9.960	4.16	91.180

optimized), 2) possible plan execution times (based on path length and velocity-time-distance calculations), and 3) the final path cost. For these simulations we let $W_g = W_t = 3$, while W_c is varying. As the weight on the configuration change cost is increased to $W_c = 7$, the planner chooses to go around the object instead of expanding and going over it. This results in a longer path with increased planning and execution time.

In addition to the image-based simulations, in Fig. 10 we also show a visualization of the simulated robot executing planner paths. We demonstrate the capacity of the robot to expand over a wide obstacle and narrow into a passage for a single instance of planning. The robot was placed in front of a wide object in a corridor, which narrows after the object. The simulation results for robot using the obstacle negotiating A* can be seen in Fig. 10. While the standard A* planner fails to provide a solution as the space is too limited for the robot to go around the obstacle, the obstacle negotiating A* generates a safe plan even in the tight space, and directs the robot to the target.

B. Real-World Results

To experimentally validate the capabilities of the planner developed in this paper we used the CENTAURO robot. For the robot control, the XBotCore [22] middleware was used for the joint, sensor, and computer communication. The interface used to command the position of wheels and the overall motion of the robot is the *CartesianInterface* [23]. The planner supplies the *CartesianInterface* with the wheel (x,y) positions with respect to the pelvis, along with the (x,y,z) positions and the yaw value of the pelvis with respect to the global frame. In real time, the *CartesianInterface* determines the trajectories of the joints needed to execute the required configuration and the pelvis position for the robot. The planner runs on an Intel Corei7-6700 PC with 24 GB RAM.

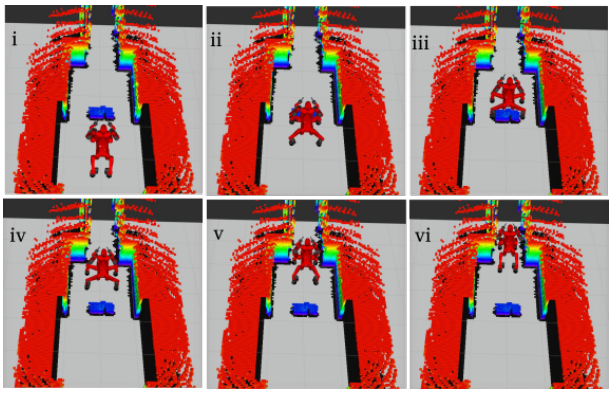


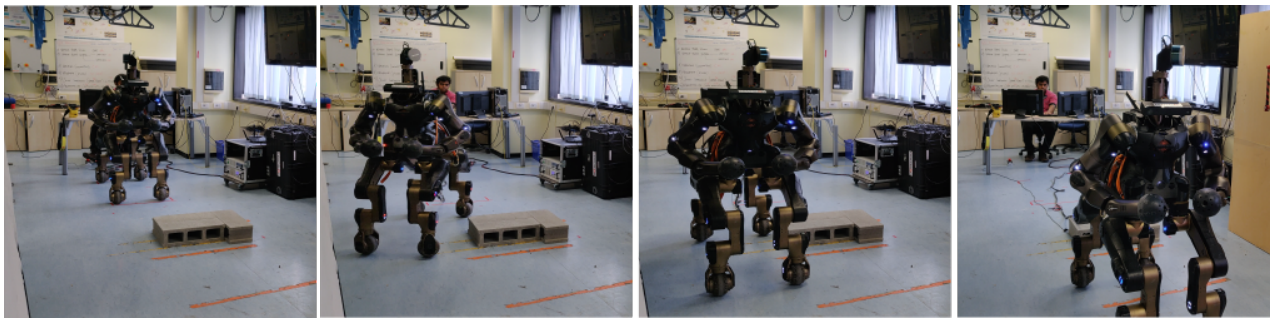
Fig. 10: Sequence of the simulated CENTAURO robot executing the plans given by the obstacle negotiating A* planner. The robot first expands over a wide object and then shrinks into the narrow corridor.

Four lab experiments were conducted, to demonstrate the capabilities of the obstacle negotiating A* planner on the CENTAURO robot. For all the experiments we let $W_g = W_t = 2$ and $W_c = 3$, to minimise the number of turns away from the goal and also force the robot to only change its configuration when there are no suitable free paths, thereby preferring small immediate detours over configuration changes. These weights were tuned on the segmented image map from the real experiment data.

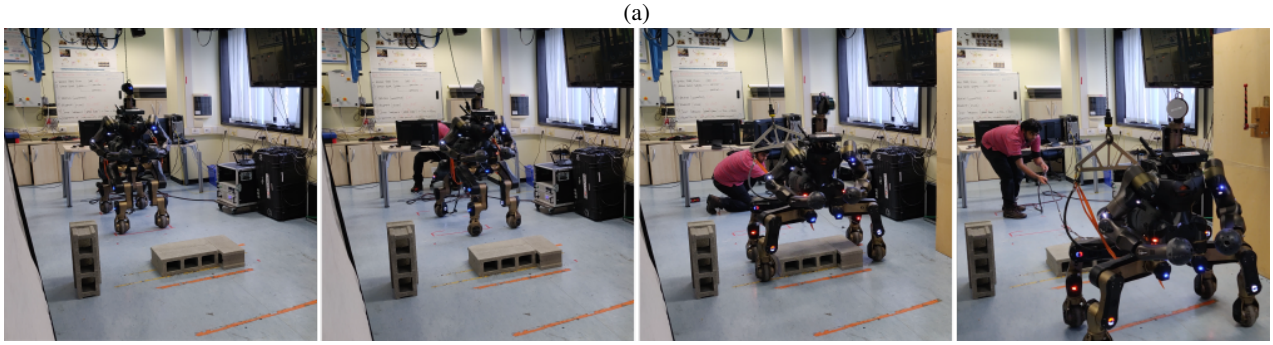
The first set of two experiments were conducted in two almost identical environments (Fig. 11). In the first experiment, as seen in the Fig. 11-(a), there is enough free space on the right (from the robot's perspective) of the robot to take a turn and go around the set of bricks on the floor. Whereas in the second experiment (Fig. 11-(b)), we put two extra bricks to close the safe free path on the right side of the set of bricks. As expected, in the first experiment the robot goes around the bricks, whereas in the second experiment, the robot turns to align itself properly with the set of low-lying bricks, expands, and then goes over them to the goal point. The total width of the low-lying bricks was approx. 70cm. The planner instructed the robot to expand its footprint polygon to a width of around 90cm and traverse safely over the obstacle.

The third experiment, as shown in Fig. 12, demonstrates the execution of a plan where the robot needs to narrow into a tight path starting from an expanded configuration. The space in between the two stands of bricks was around 75cm and the initial width of the robot polygon was 100cm. The robot moves in an expanded configuration up towards the narrow gap and then narrows itself to fit between the bricks.

The fourth and final experiment is shown in Fig. 13. In this experiment, the robot performs two configuration changes to reach its goal point, which is set at the final brick lying on the ground. Firstly it narrows its wheelbase to navigate through the tight passage consisting of two tall bricks and then expands so that the wheels do not collide with the wide brick lying on the ground. The width of the narrow passage was approx. 80cm and the width of the final brick was 50cm. The robot starts in a wide configuration (polygon



(a)



(b)

Fig. 11: Real world experiments with the CENTAURO robot going around and going over an obstacle.

Fig. 12: Sequence of the CENTAURO robot narrowing into a passage, after starting from an expanded configuration.

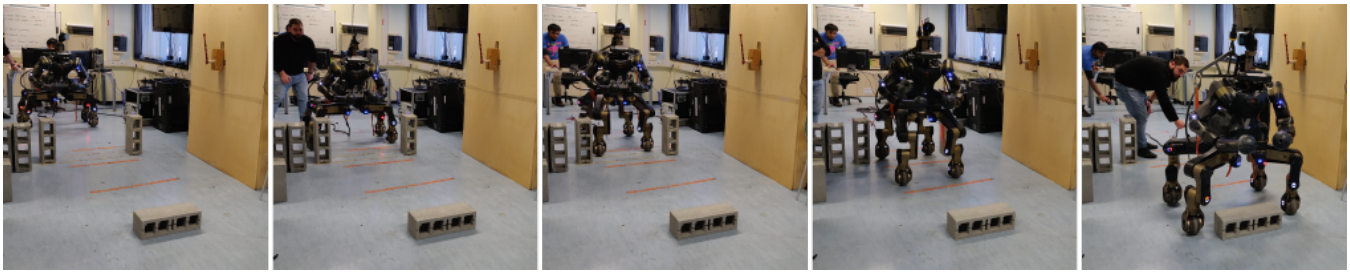


Fig. 13: A planned path where narrowing and expanding of the robot footprint polygon are executed.

width 100cm), ending with a narrower but still expanded polygon width of 76cm as the final obstacle is not as wide as it was in the first two experiments.

In Fig. 14 we report: 1) computation times, 2) final path costs, and 3) the path lengths travelled by the robot for the four experiments conducted on the CENTAURO robot. One important observation is that in case of the first two experiments (the going around and the expand experiments), the cost of expanding is higher, but the planning time and the path length are significantly lesser for the case where the robot performs expanding, rather than avoiding the obstacle. Furthermore, from the last two experiments, it can be seen that the planning time is doubled for the fourth experiment when compared to the third experiment. This is due to the computation of planning for multiple configuration changes as well as increased path length.

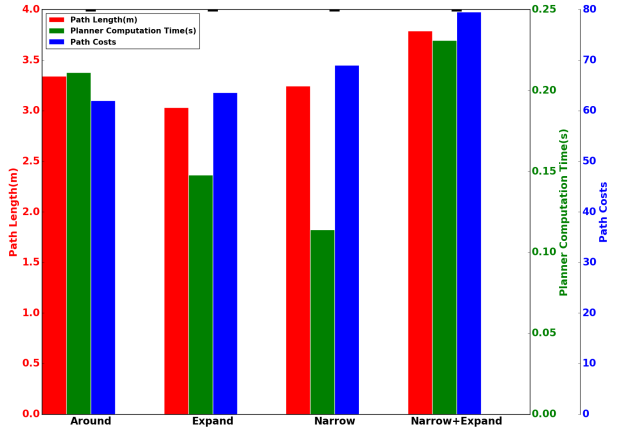


Fig. 14: Plots of the path length, planner computation times, and path costs for the 4 CENTAURO experiments.

V. Conclusion and Future Work

In this paper, we presented a modified version of the A* planner, named obstacle negotiating A*, that was capable of negotiating obstacles by widening the robot to traverse over them, as well as narrowing the robot footprint polygon in tight spaces. The effect of modifying the weight on the cost of the robot footprint polygon configuration change was studied through simulations. Four experiments were performed on the CENTAURO robot demonstrating the capability of the planner to provide paths over low obstacles and through narrow spaces.

In this work, the mapping, segmentation, and planning was done all at once at the start of the experiments and paths of lengths up to 3.8m were planned on the real robot. In the future, we intend to interface the algorithm with a SLAM algorithm and plan for longer paths, while including re-planning functionality in case of dynamic environments or noisy data. Furthermore, we intend to remove the dependency on the robot symmetry when planning and to optimize the code to reduce planning times. While the turning angles were maintained to 8 discrete orientations, incorporating a continuous orientation system would give more flexibility. We would also like to consider the energy costs involved in staying in a particular robot configuration rather than just the cost on changing the robot polygon. While the current mapping solution works for obstacles on flat surfaces, we aim at incorporating a more detailed height map for sloping surfaces. Future work would also investigate the possibility of maintaining the low-dimensional search of the planner, while planning the motion for each of the wheeled legs individually. Such a planner would allow the highly agile CENTAURO robot to navigate in more densely populated spaces.

Acknowledgment

This work is supported by the CENTAURO & CogIMon (no: 64483 & 644727) EU projects. The Titan Xp GPUs used for this research was donated by the NVIDIA Corporation. The authors would like to thank Paulo Guria for his help with the experiments.

References

[1] N. Kashiri *et al.*, "CENTAURO: A Hybrid Locomotion and High Power Resilient Manipulation Platform," *IEEE Robotics and Automation Letters (RA-L)*, 2019.

[2] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[3] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The Office Marathon: Robust Navigation in an Indoor Office Environment," in *IEEE Int. Conf. on Robotics and Automation*, 2010.

[4] "Turtebot - Willow Garrage," http://www.willowgarage.com/turtlebot.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[6] A. Hornung, D. Maier, and M. Bennewitz, "Search-based Footstep Planning," in *IEEE International Conference on Robotics and Automation (ICRA) Workshop*, 2013.

[7] D. Kanoulas, A. Stumpf, V. S. Raghavan, C. Zhou, A. Toumpa, O. Von Stryk, D. G. Caldwell, and N. G. Tsagarakis, "Footstep Planning in Rough Terrain for Bipedal Robots Using Curved Contact Patches," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 1–9.

[8] L. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[9] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," 1998.

[10] E. A. Hansen and R. Zhou, "Anytime Heuristic Search," *Journal of Artificial Intelligence Research*, vol. 28, pp. 267–297, 2007.

[11] A. Stentz, "Optimal and Efficient Path Planning for Partially-Known Environments," in *IEEEE International Conference on Robotics and Automation (ICRA)*, 1994, pp. 3310–3317.

[12] M. Brunner, B. Brüggemann, and D. Schulz, "Motion Planning for Actively Reconfigurable Mobile Robots in Search and Rescue Scenarios," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2012, pp. 1–6.

[13] M. Menna, M. Gianni, F. Ferri, and F. Pirri, "Real-time Autonomous 3D Navigation for Tracked Vehicles in Rescue Environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 696–702.

[14] M. Norouzi, J. V. Miro, and G. Dissanayake, "Planning High-Visibility Stable Paths for Reconfigurable Robots on Uneven Terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 2844–2849.

[15] A. Le, V. Prabakaran, V. Sivanantham, and R. Mohan, "Modified A-Star Algorithm for Efficient Coverage Path Planning in Tetris Inspired Self-Reconfigurable Robot with Integrated Laser Sensor," *Sensors*, vol. 18, no. 8, p. 2585, 2018.

[16] L. Pfotzer, S. Klemm, A. Rönnau, J. M. Zöllner, and R. Dillmann, "Autonomous Navigation for Reconfigurable Snake-like Robots in Challenging, Unknown Environments," *Robotics and Autonomous Systems*, vol. 89, pp. 123–135, 2017.

[17] K. Hauser and J.-C. Latombe, "Multi-Modal Motion Planning in Non-Expansive Spaces," *The International Journal of Robotics Research*, vol. 29, no. 7, pp. 897–915, 2010.

[18] T. Klamt and S. Behnke, "Planning Hybrid Driving-Stepping Locomotion on Multiple Levels of Abstraction," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 1695–1702.

[19] T. Klamt, D. Rodriguez, M. Schwarz, C. Lenz, D. Pavlichenko, D. Droeschel, and S. Behnke, "Supervised Autonomous Locomotion and Manipulation for Disaster Response with a Centaur-like Robot," *arXiv preprint arXiv:1809.06802*, 2018.

[20] R. Buchanan, T. Bandyopadhyay, M. Bjelonic, L. Wellhausen, M. Hutter, and N. Kottege, "Walking Posture Adaptation for Legged Robot Navigation in Confined Spaces," *IEEE Robotics and Automation Letters (RA-L)*, 2019.

[21] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013.

[22] L. Muratore, A. Laurenzi, E. M. Hoffman, A. Rocchi, D. G. Caldwell, and N. G. Tsagarakis, "Xbotcore: A Real-Time Cross-Robot Software Platform," in *IEEE International Conference on Robotic Computing (IRC)*, 2017, pp. 77–80.

[23] A. Laurenzi, E. M. Hoffman, L. Muratore, and N. G. Tsagarakis, "CartesI/O: A ROS Based Real-Time Capable Cartesian Control Framework," in *IEEE Int. Conf. on Robotics and Automation*, 2019.