Agile Legged-Wheeled Reconfigurable Navigation Planner Applied on the CENTAURO Robot

Vignesh Sushrutha Raghavan[1,2], Dimitrios Kanoulas[3], Darwin G. Caldwell[1] and Nikos G. Tsagarakis [1]

Abstract— **Hybrid legged-wheeled robots such as the CENTAURO, are capable of varying their footprint polygon to carry out various agile motions. This property can be advantageous for wheeled-only planning in cluttered spaces, which is our focus. In this paper, we present an improved algorithm that builds upon our previously introduced preliminary footprint varying A* planner, which was based on the rectangular symmetry of the foot support polygon. In particular, we introduce a Theta* based planner with trapezium-like search, which aims to further reduce the limitations imposed upon the wheeled-only navigation of the CENTAURO robot by the low-dimensional search space, maintaining the real-time computational efficiency. The method is tested on the simulated and real full-size CENTAURO robot in cluttered environments.**
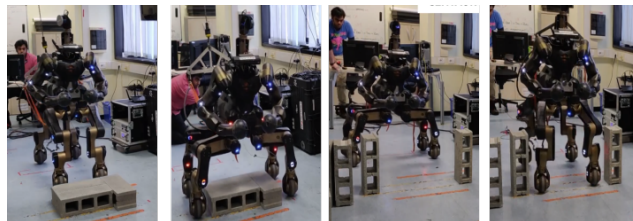
Fig. 1: CENTAURO robot navigation based on plans from the initial Obstacle Negotiating A* planner [2]. The first two images show CENTAURO expanding over a wide obstacle, while the last two show it narrowing into a small space.

I. INTRODUCTION

In cluttered and unstructured environments, autonomous navigation is impeded by obstacles of various shapes and sizes. The spacing of the obstacles can be such that continuous navigation of a wheeled robot with a fixed robot footprint might be impossible. While stepping over obstacles is a potential solution, switching of the control from wheeled to stepping and the planning required to make safe footsteps is time consuming, whereas wheeled only navigation could reduce transit time in some cases. In such scenarios, a robot capable of modifying its height and footprint polygon like the CENTAURO robot [1], may be capable of performing wheeled-only navigation in certain scenarios by expanding over wide objects and narrowing into corridors. While there exist many studies that present planning solutions for wheeled robots, there are few that provide real-time planning for robots with re-configurable footprints.

The CENTAURO robot is 42 DoF hybrid legged-wheeled robot, with four 7DoF legs and wheel actuators as end-effectors. In our previous work [2], we presented an initial solution for re-configurable legged-wheeled navigation for the CENTAURO robot based on a simple symmetric re-configurable rectangle footprint and A* planner. The presented Obstacle Negotiating A* planner provided plans that allowed the robot to negotiate wide, low-lying obstacles by expanding over them or traverse narrow spaces by narrowing into them, as seen from the experimental images in Fig. 1. A rectangular constraint was imposed on the robot footprint

to search for traversable plans. Being an initial version of the algorithm, this rectangular symmetry, as well as the A* algorithm, severely limited the flexibility of the navigation in the following ways:

- Only certain spaces could be traversed due to incomplete exploitation of the agility of the robot. As shown in Fig. 2, if there is not sufficient space between the narrow passage and the wide obstacle, as a simple rectangle is being used for the search, a collision would occur and hence no path will be found. This problem also occurs in the reverse problem of going into a narrow passage after expanding over a wide example.
- The A* search used as the base of the planner does the search on a 8 neighbor grid. Thereby the orientations assumed by the robot are limited to $\frac{n\pi}{4}, n = 0, 1, ..., 8$. So if the robot has to enter a narrow passage angled at for instance 20 deg, no path will be found. The CENTAURO wheels are also capable of omni-directional motion, but this is ignored in the A* search as well.

In this paper, we present an improved version of the Obstacle Negotiating A* planner, such that the aforementioned limitations on CENTAURO's wheeled-only motion which are imposed by the low-dimensional search planner, are eliminated while still maintaining the computationally efficient low-dimensional search. This is done by allowing independent operation of the front and back wheel pairs, as well as incorporating the omni-directional motion. This allows the robot to navigate through even tighter spaces as well as perform more agile motions in the presence of multiple obstacles.

The rest of the paper is organized as follows. First, we review the related work on re-configurable planning (Sec. II). Then, we briefly describe the initial algorithm [2], followed by the detailed description of the introduced modifications

[1]Humanoids and Human-Centered Mechatronics & Advanced Robotics, Istituto Italiano di Tecnologia (IIT), Via Morego 30, 16163, Genoa, Italy. {Vignesh.Raghavan, Darwin.Caldwell, Nikos.Tsagarakis}@iit.it

[2]Department of Information Engineering, University of Pisa.

[3]Department of Computer Science, University College London, Gower Street, WC1E 6BT, UK. d.kanoulas@ucl.ac.uk
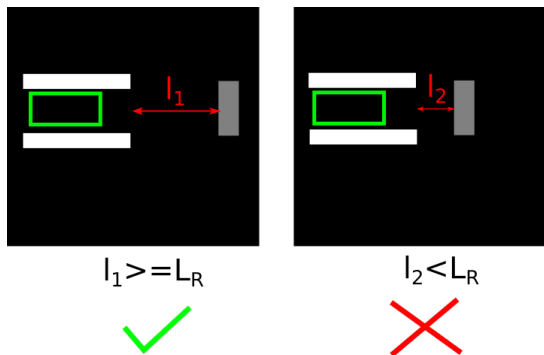
Fig. 2: A scenario demonstrating a drawback of the initial Obstacle Negotiating A* planner [2], where the robot (green rectangle) needs to stand over the low lying obstacle (grey rectangle). l_1 and l_2 represent the length of the space from the end of the narrow passage (two white rectangles) to the start of the wide low lying obstacle. L_R represents the length of the robot in its narrowest configuration (green rectangle).

and improvements (Sec. III). We then present simulated and real CENTAURO robot experimental demonstrations of the improved capabilities of the planner (Sec. IV). Finally, we conclude with some future directions (Sec. V).

II. RELATED WORK

Algorithms such as the A* [3], RRT, [4], and PRM [5], have formed the base of many autonomous navigation planning methods. Improvements to the original algorithms such as the ARA* [6] and D* [7], RRT-Connect[8], and Theta*[9] have been further developed and extensively used in many planning scenarios. Building on these algorithms, many studies have presented re-configurable planning on robots. For instance, in [10], [11], [12], re-configuration is used by the robots to deal with uneven terrain, stairs, and obstacles. Although, these robots were not modifying their support polygon, they were always avoiding obstacles rather than negotiating them. Studies such as [13] (using A*) and [14] presented methods to take advantage of the snake-like nature of some robots, to plan for re-configuring the robot polygon for obstacles avoidance.

A recent study in [15] used a deformable box robot model to plan paths to both avoid overhead and floor obstacles and fit in narrow spaces by modifying the width and height of the robot. This was presented for legged-only motion for obstacle avoidance/negotiation and didnt consider the cost of modifying the robot polygon. While the research on humanoid footstep planning algorithms like [16], [17] has been widespread, recently there have been extensive studies on real-time hybrid legged-wheeled robot motion planning. In [18], [19], a combined driving and stepping motion planning for the Momaro robot was presented. It is based on the ARA* algorithm, where longitudinal robot polygon changes were considered to facilitate driving and stepping onto higher surfaces. Obstacle negotiation or lateral polygon changes were not considered.

In our previous work [2], we reduced the search space by using a rectangle-based search in the A* algorithm. Although the plans restricted the flexibility of the wheeled-only motions, we considered the cost of modifying the robot footprint polygon for wheeled-only motion in the standard A* cost functions. In this way, we controlled whether obstacle avoidance or negotiation is preferred. It was possible to force the robot to both change its polygon only in unavoidable situations as well as to prefer polygon changes if it lead to a shorter traversal path. This work was further combined with a roughness estimation algorithm in [20] to allow for motion planning over low-lying rough surfaces.

In this work, instead of using a rectangle symmetry and A* as the base planner, we use trapezium-like symmetry to facilitate independent motions for the front and back pair of wheels, and the Theta* [9] algorithm as a base planner to allow for any-angle robot orientation. Similar to our previous work, we use data from the VLP16 Velodyne lidar sensor which is fixed on the head of the robot, to create costmaps based on point clouds and ground plane filtering extracted from the Octomap [21] environment representation. Two-dimensional (2D) images will be used in planning searches.

III. TRAPEZIUM OBSTACLE NEGOTIATING THETA*

In this section, we present the improved Obstacle Negotiating algorithm. In particular, we briefly describe our previously introduced algorithm [2] and the detailed modifications made to facilitate improvements. An overview of the both methods can be seen in Fig. 3.

A. Obstacle Negotiating A*

Using point clouds from the robot's VLP16 Velodyne lidar sensor, Octomap [21], and costmap functionalities of the software package *move_base*[1], a 2D segmented map image is created, e.g. Fig. 2. In the map image, the black pixels represent free space, grey pixels represent low lying obstacle points (negotiable) easily cleared by the robot base and white pixels represent tall obstacles (non-negotiable). More detailed explanation of the segmented map creation can be found in [2].

We use the 2D segmented map image for all the planning searches and hence the operations are done on pixels. Furthermore, we use intensities of the pixels in the neighborhood of the queried pixel coordinate to determine a collision free valid configuration that the robot can assume to pass through the queried pixel. This operation is executed as follows. With the segmented map, an 8-neighborhood-based grid search is executed on the image points from the start to the goal point. The robot is forced to orient itself in the direction of motion. Furthermore, we impose a constraint on the chosen robot rectangular footprint as follows. The sum of the robot footprint rectangle width (w_r) and length (l_r) is set to S_p, i.e. $S_p = w_r + l_r$. S_p is set to a constant value based on the limits of the leg joints of the robot. Thus, if the width of the robot rectangle is known, the length of the robot footprint
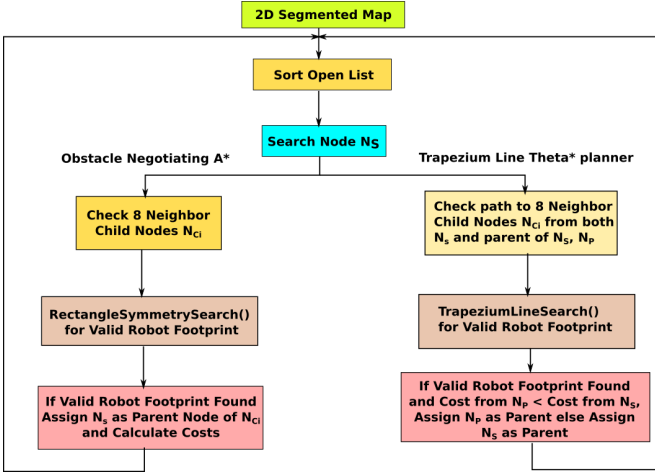
---

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

Fig. 3: Graphical overview of the previously presented Obstacle A* planner and the improved version, namely Trapezium Line Theta* planner.

Fig. 4: Graphical overview of the *RectangleSymmetrySearch*() used in the Obstacle Negotiating A*.

polygon can be simply obtained 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 rectangle is determined by using simple proportions based on the robot footprint rectangle width w_r by the following formula:

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

The search for a valid robot configuration is done by using a rectangular region of interest of image pixels (ROI). First, depending on the orientation of the motion from the node N_S being searched, to the neighborhood child $N_{Ci}, i = 1...8$, the image coordinates in the rectangle ROI are rotated such that they are in the frame as seen by the robot. Then, based on the presence of negotiable and non-negotiable obstacle points, decisions to maintain, expand, or narrow the robot footprint rectangle are made. In the presence of only negotiable obstacles, both expanding and narrowing is tested based on the maximum and minimum horizontal coordinate of obstacles respectively, in the rotated and resolved ROI. In the presence of non-negotiable obstacles, only narrowing is done. Based on the horizontal coordinates chosen, the robot footprint rectangle width is set and checked if it fits within reasonable bounds of robot capabilities. An overview of this rectangular search can be seen in Fig. 4.

For the Obstacle Negotiating A*, the standard cost functions of A* namely, $g()$ and $h()$ are modified as follows:

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

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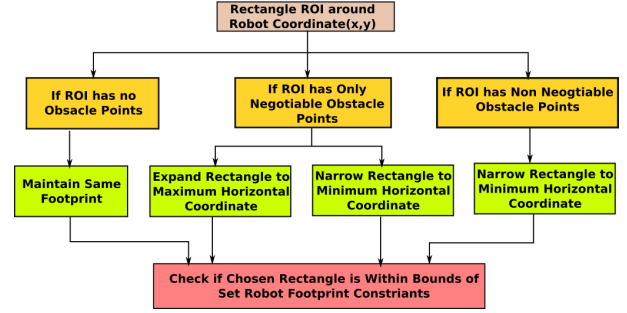
$$\theta_{goal} = acos\left(\frac{x_g - x_c}{\sqrt{(x_g - x_c)^2 + (y_g - y_c)^2}}\right) \qquad (3)$$

where $[x_c, y_c]$ is the child neighbour node being queried, $[x_p, y_p]$ the corresponding parent node or searched node N_s, and $[x_g, y_g]$ the goal node. The $|.|$ 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. W_g is the weight on the cost of turning away from the direction of the goal. A more detailed analysis on the decision making process and the usage of weights can be found in our previous work [2].

B. Trapezium Line Theta* Planner

For operations of the proposed improved planner, the segmented map, the imposed constraint on the rectangle, as well as the cost functions are reused. As mentioned earlier, the planner searches are done on 2D image pixels, and the pixel coordinates and intensities of the pixels in neighborhood of the queried point are used to determine valid robot configurations. In the improved planner we proceed similarly with the following differences. The first of the key difference is that, instead of a simple singular rectangular symmetry for the whole robot, similar symmetry rules are applied separately to the front and back pair of wheels, allowing independent operation of the two pairs of wheels as follows.

First, we extend the rectangular symmetry introduced in our previous work to two separate pair of wheels, by creating a trapezium like calculation (Fig. 5). Consider a target point (x, y) during planning. Let the orientation of entry or search be θ and the constant S_p introduced before be expressed in pixels as S_{pp}. Consider a square grid of size $2 \times S_{pp}$. Let the grid be expressed in a coordinate frame
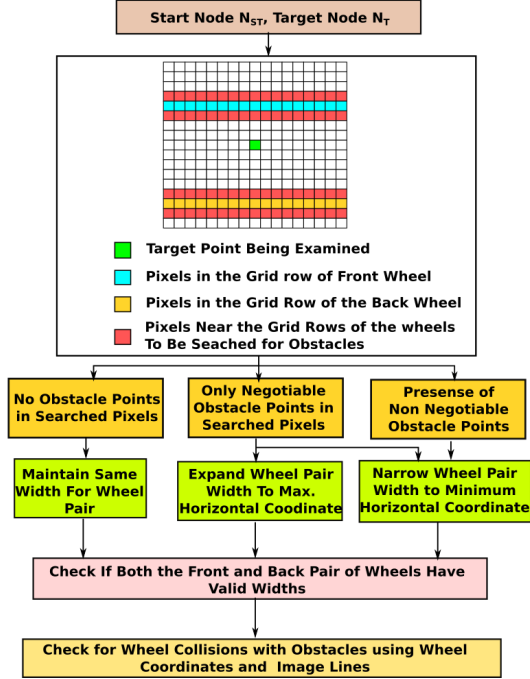
TrapeziumLineSearch()

| Start Node N_{ST}, Target Node N_T |

Legend:
- ■ (green) Target Point Being Examined
- ■ (cyan) Pixels in the Grid row of Front Wheel
- ■ (yellow) Pixels in the Grid Row of the Back Wheel
- ■ (red) Pixels Near the Grid Rows of the wheels To Be Searched for Obstacles

| No Obstacle Points in Searched Pixels | Only Negotiable Obstacle Points in Searched Pixels | Presense of Non Negotiable Obstacle Points |

| Maintain Same Width For Wheel Pair | Expand Wheel Pair Width To Max. Horizontal Coodinate | Narrow Wheel Pair Width to Minimum Horizontal Coordinate |

| Check If Both the Front and Back Pair of Wheels Have Valid Widths |

| Check for Wheel Collisions with Obstacles using Wheel Coordinates and Image Lines |

Fig. 5: Graphical Overview of the *TrapeziumLineSeach()* used in the new planner presented in this paper.

oriented at θ and centered at the target point. Hence the grid coordinates run from $(-S_{pp}, -S_{pp})$ to (S_{pp}, S_{pp}). We extract image coordinates of the pixels corresponding to this rectangular grid using a simple homogeneous transformation. Thus, we have image coordinates of the pixels in the square grid centered at (x, y) as well as the corresponding local square grid coordinates in the frame centered at (x, y) and oriented at θ. Let the current width between the front pair of wheels in pixels be w_{fp} and the back pair of wheels be w_{bp}. So the x (forward with respect to the robot) coordinate of the front wheel and back wheel in the local grid coordinates can be simply calculated as $(S_{pp} - w_{fp})$ and $-(S_{pp} - w_{bp})$. So if we extract all the points in the grid in the $(S_{pp} - w_{fp})^{th}$ and $-(S_{pp} - w_{bp})^{th}$ row, we know of all the grid points in the horizontal neighborhood of the wheel. So along with the aforementioned rows, we also extract the grid points and the corresponding image pixels of their rows above and below them. A graphical view can be seen in Fig. 5.

In terms of code, we merely condense the local grid coordinates and the corresponding image pixel coordinates into a 2D vector array of length $(2 \times S_{pp})^2$ and run vector search in the indices corresponding to the grid rows of the wheel pairs, making the search less cumbersome. Hence, we have all the image coordinates, corresponding image pixels, and corresponding local grid coordinates of all the pixels near the two pairs of wheels.

Similar to the *RectangleSymmetrySearch* (Fig. 4), we test for obstacle points and depending on the three afore-mentioned scenarios (i.e., no obstacle, negotiable obstacle,

non-negotiable obstacle) that can be seen in Fig. 5, we decide whether to expand, narrow, or maintain the same configuration. The difference this time is that the front and back pair of wheels operate independently. Once horizontal coordinates of obstacles are obtained from the local grid coordinates, the width of the front and back pair of wheels is set and checked for collisions.

The second key difference from the previously presented work is that we use the Theta* search. Hence, in addition to the standard A* search, there is an extra step where it is also checked if we can travel from the parent of the searched node N_S, namely N_P, to the child target node N_{Ci} with the above calculated front and back wheel pair configuration, without any collisions. Depending on the costs, N_P is chosen as the parent node for N_{Ci} if the cost is lesser, else N_S is chosen as the parent, thus giving an any angle traversal capability as the global angle of the motion direction from N_P to N_{Ci} need not be $\frac{n\pi}{4}, n = 0, 1...8$.

The third key difference is that we also check for omni-directional traversal, meaning that the robot heading can be not oriented in the direction of motion. If a valid path without collisions is found, then, maintaining the same orientation and rolling in an omni-directional manner is preferred over changing the heading of the robot.

The fourth and final difference/addition to this algorithm to aid the Theta* aspect of the presented planner, is the simple collision detection implemented based on image pixels. Once we obtain the robot configuration for the target point, we extract a line of pixels between the previous and current wheel image coordinates, and check if any of the pixels are part of an obstacle. If they are, then the configuration and path is discarded.

As in the case of the original study the height of the pelvis of the robot above the ground is determined by the width. Only in this case we obtain two heights h_f and h_b based on w_f, w_b (front and back wheel pair widths in m) and the height calculations in Eq. 1. The pelvis height H_p and pitch θ_{pitch} are calculated as follows, in such a way that the pelvis may pitch towards the pair of wheels that has expanded. This is done to maintain the stability of the robot.

$$
\begin{aligned}
&\text{If } h_b >= h_f \\
&H_p = h_f + \frac{(S_p - w_b) \times (h_b - h_f)}{2 \times S_p - w_f - w_b} \\
&\theta_{pitch} = \tan^{-1}\left(\frac{(h_b - h_f)}{(2 \times S_p - w_f - w_b)} \right) \\
&\text{else} \\
&H_p = h_b + \frac{(S_p - w_f) \times (h_f - h_b)}{2 \times S_p - w_f - w_b} \\
&\theta_{pitch} = -\tan^{-1}\left(\frac{(h_f - h_b)}{(2 \times S_p - w_f - w_b)} \right)
\end{aligned}
\tag{4}
$$

In the next section, we present simulated and real robot experiments, depicting the advantages of the presented improved planner.
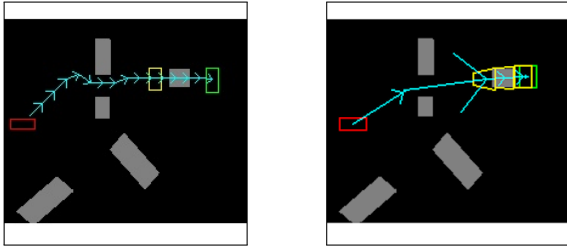
Fig. 6: Plans by the Obstacle Negotiating A* planner (left) and the Trapezium Line Theta* Planner (right) for similar start and goal points. Red polygon represents the starting point, the green polygon represents the goal, the yellow polygon represents instances of configuration change.

IV. SIMULATIONS AND REAL ROBOT EXPERIMENT

For both simulated robot and real robot experiments, an obstacle safety distance equal to the wheel width (in our case 10cm) was considered during the creation of costmaps and planning searches. The *CartesianInterface* [22] was used to command the joint trajectories of both the simulated and real CENTAURO robot [1].

A. Simulations

We first present an image-based simulation comparison of the Obstacle Negotiating A* and Trapezium Line Theta* algorithm. For both simulated plans, we experimentally set $W_c = 1, W_t = W_{gt} = 3$. The results of the plans generated can be seen in Fig. 6. The path length of the plan from Obstacle Negotiating A* was 9.042m and the time it took to compute the plan was 1.88sec. Whereas, the Trapezium Line Theta* planned a path of length 7.33m and the plan was computed in 0.650 sec. In spite of having more computations due to extra collision checks and front and back wheel pixel searches, the Trapezium Line Theta* had shorter computation time. Furthermore, as can be seen, the path length of the first planner was longer as the improved planner used omni-directional searches to ignore turns and changes to robot heading thereby having a more smooth motion, unlike the first case where 3 changes to the robot heading were needed.

We now present two simulations to explicitly demonstrate the improved capabilities of the planner to allow agile wheeled-only motions on the simulated CENTAURO robot. For the following simulations and the real experiment, the two turning weights W_t and W_g where set to 1.0 and W_c was set to 0.5, so as to give more preference to changing robot configuration over turning and taking longer paths with no re-configuration. The first simulation sequence can be seen in Fig. 7, where the robot has to traverse a passage angled at 60 deg. The sequence shows the robot travelling omni-directionally, expanding only the front wheel pair in the beginning, then, turning approximately 18 deg and expanding the back wheels to safely go over the low lying obstacle. Our previous planner would not have been capable of planning a path of this orientation and allow such an agile motion in a tight space.
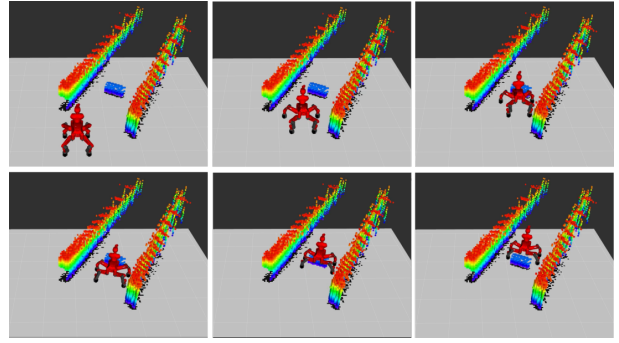

Fig. 7: Simulation Sequence of the CENTAURO robot moving to a goal behind a low lying object in a corridor angled at 60 deg. The robot travels omni-directionally till it turns approx. 18 deg to expand the back wheels to go over the low lying object safely.
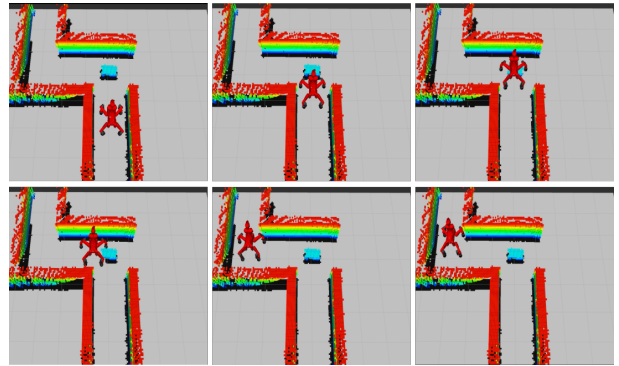

Fig. 8: Simulation sequence of the CENTAURO robot traversing through right angled corridors in the presence of a low lying object.

The second simulation sequence as seen in Fig. 8, depicts the robot traversing a corridor with right angle turns with ease by completely utilising omni-directional motion. Furthermore, throughout the whole passage it only expands the front wheels and not the back wheels as expanding both pair of wheels increases the cost. Hence, it finds a solution which only needs configuration change for the front pair of wheels, demonstrating the effectiveness of the independent front and back pair operation. Our previous work would have failed to find a plan over the low lying obstacle at the end of the first passage as it would not have enough space to expand the full footprint of the robot before reaching the object.

B. Real Experiment On the CENTAURO Robot

We demonstrate our improved Trapezium Line Theta* planner on the real CENTAURO robot. A slightly cluttered environment with bricks was created. The environment consisted of a small narrow passage with minimum width of 75cm followed by a low-lying wide obstacle of width 50cm. The distance between the end of the narrow passage and the low-lying brick is 80cm. As can be seen from Fig. 9, the robot is not aligned with the narrow passage and not enough space is available on the sides. The robot first aligns
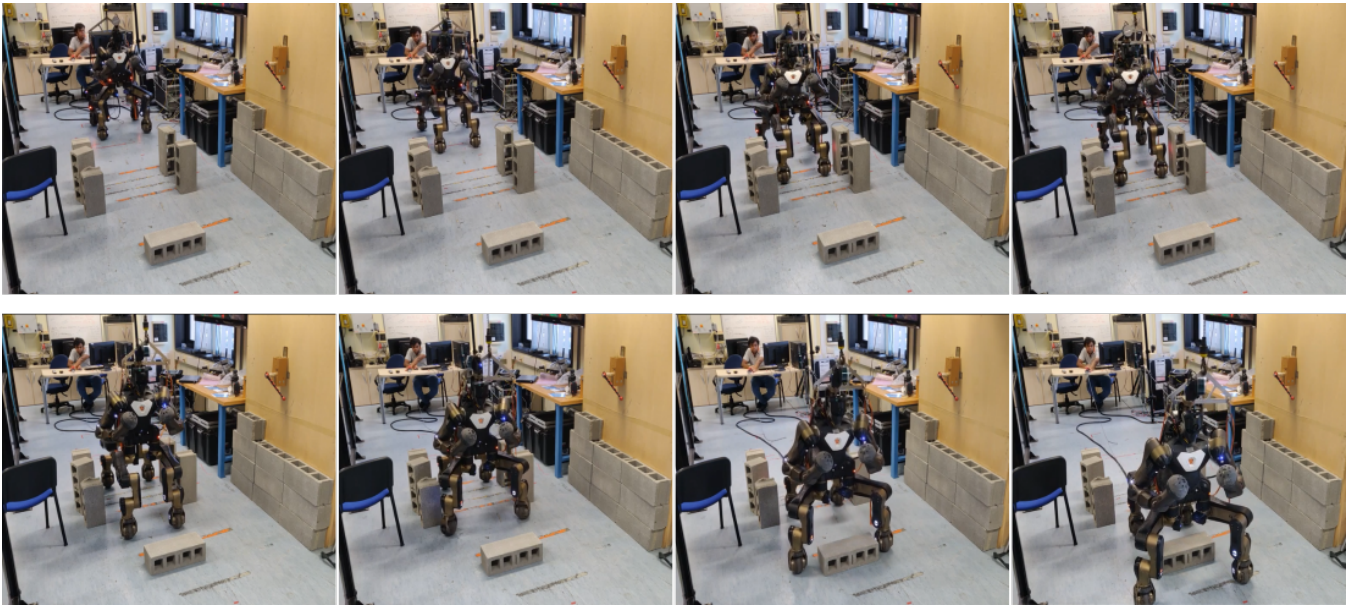
Fig. 9: Sequence of the CENTAURO robot moving through a narrow passage to a goal point on the low-lying object.

itself with the passage omni-directionally, while preparing to narrow the front pair of wheels to a width of 50cm from an initial width of 70cm, to safely move into the passage. Halfway into the passage, the back wheels narrow so that they can pass safely into it. When the front wheels are out of the passage and close to the wide low-lying obstacle, the front pair of wheels alone expand to a width of 80cm. It is to be noted that before expanding, the full length of the robot is 110cm, while the space between the end of the narrow passage and the beginning of low-lying obstacle is 80cm. Hence, our previous planner failed to provide valid plans as it was unable to find enough space for the rectangle to expand without the back wheels crashing into the bricks of the passage. Whereas, the improved algorithm, only expands the front wheels and facilitates the reaching of the goal. The planned path length is approximately 3.49m and it is computed in 0.0634 seconds.

V. CONCLUSIONS AND FUTURE WORK

In this work, we presented a improved version of the wheeled-only Obstacle Negotiating A* algorithm. The presented Trapezium Line Theta* algorithm was able to overcome the issues of fixed robot orientations and absence of valid plans due to insufficient space while going from a narrow passage to over a wide obstacle and vice-versa. This was achieved by using a trapezium-like search to find valid widths for the front and back pair of wheels independent of each other, thereby giving rise to a more agile motion planner. Simulations and real experiments were presented to highlight the improvement.

The authors are well aware that while the improvement does reduce the limitations of wheeled only planning in cluttered spaces, not all environments can be solved with such a low-dimensional and symmetry imposing wheeled-only planner. The future work will include combination of the presented planner with a stepping motion planner such that, in the case where no valid wheeled motion is found, a stepping motion planning is carried out. By using the wheeled motion planner in conjunction with a stepping motion planner, the robot will be able to traverse increasingly more difficult/tighter cluttered spaces. Such a planner will allow the robot to reach a task space in spite of the presence of clutter between its starting point and the target task space. Furthermore, since the wheeled-only planner computes plans for path lengths of $9 - 11$m within a maximum of $1 - 2$ seconds, wheeled motion planning in the presence of dynamic obstacles as well as repeated re-planning can be carried out without much loss of time and this improvement will be worked upon in the future.

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] V. Raghavan, D. Kanoulas, A. Laurenzi, D. Caldwell, and N. Tsagarakis, "Variable configuration planner for legged-rolling obstacle negotiation locomotion: Application on the centauro robot," in *IEEE Int. Conf. on Robotics and Automation*, 11 2019.
[3] 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.
[4] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," 1998.
[5] 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.

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

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

[8] J. J. Kuffner and S. M. LaValle, "RRT-connect: An Efficient Approach to Single-Query Path Planning," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2000, pp. 995–1001.

[9] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta^*: Any-angle path planning on grids," in *AAAI*, vol. 7, 2007, pp. 1177–1183.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] D. Kanoulas, C. Zhou, A. Nguyen, G. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Vision-based foothold contact reasoning using curved surface patches," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 121–128.

[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] V. Suryamurthy, V. Sushrutha Raghavan, A. Laurenzi, N. Tsagarakis, and D. Kanoulas, "Terrain segmentation and roughness estimation using rgb data: Path planning application on the centauro robot," in *IEEE-RAS International Conference on Humanoid Robots*. IEEE-RAS, 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] 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.