The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Stereo Camera Based Navigation of Mobile Robots on Rough Terrain

Annett Chilian and Heiko Hirschmüller

Abstract— **A navigation algorithm for mobile robots in unknown rough terrain has been developed. The algorithm is solely based on stereo images and suitable for wheeled and legged robots. The navigation system is able to guide the robot along a short and safe path to a goal specified by the operator and given in coordinates relative to the starting point of the robot. The algorithm uses visual odometry for localization. The terrain is modeled from stereo images and its traversability is estimated. A D* Lite planner is used for efficiently planning a short and safe path by incorporating terrain traversability in the planning process. The robot actively explores its environment as it follows the path to the goal. The algorithm has been tested on a wheel driven mobile robot and on a six-legged walking robot on rough terrain.**

I. INTRODUCTION

Autonomous navigation of mobile robots on rough terrain has been an important field of research during the last years. Many applications demand for autonomous agents which can fulfill special tasks in unknown environments, for example on planetary surfaces, in caves or in disaster areas.

Walking robots can be beneficial on rough terrain because their feet do not need continuous paths on the ground but can step over or on obstacles. However, walking robots suffer from poor odometry and limited payloads. Fig. 1 shows an example for a walking robot.

When navigating on rough terrain, a robot is likely to encounter slip, which causes errors in the robot's odometry and poses a challenge to position estimation. Furthermore, the robot could get stuck in rocky areas or fall off cliffs or steep slopes. To avoid this, the robot has to estimate the traversability of the terrain in order to plan a safe path. While on even ground there is a clear distinction between untraversable obstacles and free space, on rough terrain the degree of traversability can vary continuously between "easily traversable" and "just barely traversable". Since usually no a priori maps are available, the robot needs sensors such as laser scanners or cameras which enable it to model the terrain surface in order to estimate the terrain difficulty. Building a model of the environment is an important task because a map is required for planning a path to the goal point. Since information is collected as the robot moves towards the goal, the robot's knowledge of the environment changes over time. The path planner must be able to replan the path when new information is available.

One of the first systems which was able to autonomously navigate on rough terrain was the Autonomous Land Vehicle (ALV) developed at the Hughes Artificial Intelligence Center in 1987 [2]. It used a laser scanner to build a navigation map and marked areas which would cause invalid vehicle configurations as untraversable.

DLR – German Aerospace Center, Institute of Robotics and Mechatronics, Oberpfaffenhofen, Germany annett.chilian@dlr.de

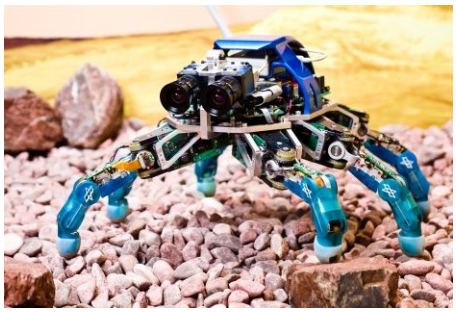


Fig. 1. Example for a walking robot: DLR-Crawler

NASA's Jet Propulsion Laboratory developed the Rocky rovers, which were test platforms for mostly sensor based navigation algorithms [10], [9]. They were prototypes for the Sojourner mars rover, which arrived on mars in 1997. Using a camera and a laser striper, it was able to detect and avoid obstacles along its way [11]. However, these systems only used a binary representation of the environment consisting of obstacles and free space.

The RANGER navigation system [7] is able to estimate the difficulty of traversable areas by computing the configuration a vehicle would have on certain terrain points. This knowledge is then included into the path planning process. Based on RANGER the Morphin algorithm [12] was developed. Using stereo images, it estimates the traversability of regular sized terrain patches by fitting a plane to the data. Morphin uses the traversability measures of these patches to evaluate the safety of possible steering arcs for the rover. These arcs are further evaluated on their use for reaching the specified goal point. The arc with the highest total vote is then commanded to the rover.

Morphin provided a basis for the development of the navigation system of the mars exploration rovers Spirit and Opportunity. Their automatic navigation mode uses the local path planner GESTALT [3]. GESTALT also uses arc votes to command steering angles to the rover. However, it estimates the traversability of a grid cell in the terrain by fitting a plane to the terrain data of a rover-sized patch centered around the cell. It computes the slope hazard, roughness hazard, step hazard and border hazard of the center cell from that terrain patch. The steering arcs are assigned hazard votes computed from a weighted sum of cells traversed by the arc. Since GESTALT is a local path planner an additional global Field D* planner was implemented [1]. This planner calculates the waypoint votes for each candidate arc by computing the cost

of the shortest path from the end of each possible steering arc to the goal. Merging hazard and waypoint votes gives a goodness value. The steering arc with the highest goodness value is commanded to the rover.

However, commanding steering angles is only advantageous for wheeled robots. Especially on rough terrain, walking robots can be beneficial. The navigation algorithm presented in this paper is suitable for wheeled and legged robots. It uses a stereo camera as the only sensor. Stereo cameras are light and cheap sensors, which can be used for building terrain models as well as for motion estimation by visual odometry. Unlike wheel or leg odometry, visual odometry is not affected by slip. Thus, it is particularly suitable for rough terrain and can be used for legged robots, which have limited payloads and poor odometry. Furthermore, the navigation system uses a D* Lite path planner for planning global paths considering path length and terrain traversability.

II. DESCRIPTION OF THE NAVIGATION SYSTEM

The navigation system solves the tasks position estimation, terrain modeling, traversability estimation, path planning and motion control, which are presented in detail in the following sections.

A. Depth Image Computation and Position Estimation

Depth images are computed from rectified stereo images using a correlation based multiple-window approach [5]. Position estimation is based on visual odometry with 6 degrees of freedom computed from the left stereo image and the corresponding depth image [6]. A Harris Corner Detector is used to detect features in consecutive left camera images. The features C of the current image are compared with the detected feature points P in the previous image by robust correlation to find an initial correspondence C_i, P_i. Next, an outlier detection is applied to discard wrong correspondences. It is based on computing relative distances of pairs of corresponding corners, reconstructed by using the associated depth image. The rigid transformation between the subsequent camera coordinate systems can be described by a rotation matrix R_r and a translation vector T_r as

$$P_i = R_r C_i + T_r + \epsilon_i, \tag{1}$$

where ϵ_i is the motion error. By minimizing ϵ_i, the rotation R_r and translation T_r can be computed, which gives the relative motion between the two frames. By summing up all relative movements, the absolute rigid transformation R, T between the start camera coordinate system and the current camera coordinate system can be obtained. R and T are given in camera coordinates and have to be transformed into the world coordinate system, which can be rotated and/or shifted with respect to the camera coordinate system.

Since this method does not use feature tracking and does not apply any restrictions to the image positions of corresponding points, it can cope with low frame rates and large motions between images.

To increase the robustness of the visual odometry, the relative movements are not only computed from the last image but from several previous images, which have been stored. The median of the resulting transformations is used to estimate the relative motion. This results in higher accuracy and robustness to blurred images.

Since visual odometry is an incremental position estimation method, errors accumulate over time. To improve the accuracy, a simultaneous localization and mapping (SLAM) approach can be used to store absolute landmark positions for correcting the robot's position estimate when previously detected landmarks are revisited. However, such a method has high computational costs and has not been included in the present work so far.

B. Terrain Modeling

The terrain is represented as a digital elevation model (DEM), which consists of a regular 2D grid. Each grid cell stores a single height value which is the average height value of all terrain points that are located in that grid cell. Unlike a full 3D representation of the terrain, a DEM cannot model multilevel environments. However, a DEM representation has significantly lower computational costs and is sufficient for most applications.

The terrain model is created from the depth images computed by the stereo process. A local map is built from a single stereo image pair by calculating the 3D object points from the depth image, projecting them into the x-y-plane of the world coordinate frame and storing the highest value. To avoid inserting height values into the map which are likely to be erroneous due to noise, only object points within a certain distance from the camera are considered. The local map is attached to the global map using the estimated position of the stereo camera. Fig. 2 illustrates the mapping process for a small terrain patch built from 65 local maps. Existing height values are overwritten by new data.

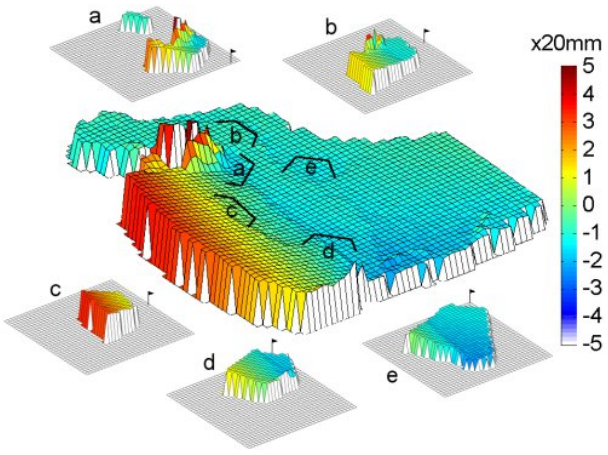


Fig. 2. DEM of the terrain built from 65 stereo views and examples for local maps created from a single stereo views (a-e). Grid resolution: 20 mm

Due to the incremental estimation of the robot's position, errors in calculating the motion accumulate. This can result in artifacts in the terrain model, which have to be considered

during traversability estimation. For this reason, the terrain model not only stores the height value for each grid cell but also the frame number indicating the "age" of the height value. This will be used in the traversability estimation process.

C. Traversability Estimation

Based on the DEM, each cell has to be assigned a danger value d ($d \in \{[0,1], \infty\}$) describing the terrain difficulty. A cell is traversable if the robot is not exposed to critical terrain hazards irrespective of its orientation given its center is located in that cell. Thus, the robot can be treated as a point in further computations. A danger value of $d = 0$ stands for completely flat, smooth terrain, which can be traversed by the robot most easily. Higher danger values are assigned to areas which are harder to pass. A value of $d = 1$ describes terrain which is just barely traversable for the robot. Untraversable regions are assigned $d = \infty$. Unknown areas are assumed to be traversable but are assigned a high danger value of $d = 1$.

There are three potential hazards: steep slopes, high terrain roughness and high steps. Each of these criteria must be estimated from the DEM. If one of the hazard criteria exceeds the corresponding critical value, the cell is marked as untraversable. The critical values s_{crit}, r_{crit} and h_{crit} are the maximum slope, roughness and step height which the robot can traverse without tipping over or getting stuck. For traversable cells the danger value is computed from the three types of hazards as

$$d = \alpha_1 \frac{s}{s_{crit}} + \alpha_2 \frac{r}{r_{crit}} + \alpha_3 \frac{h}{h_{crit}}, \tag{2}$$

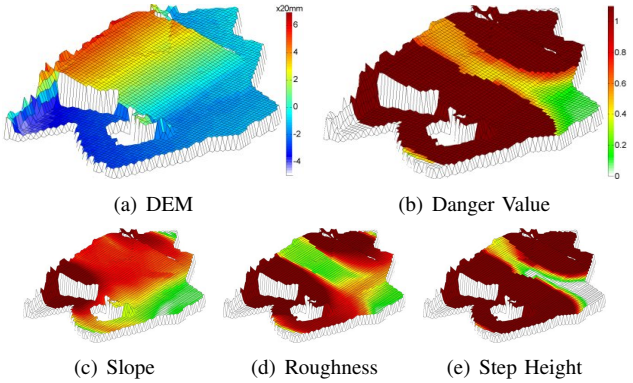
where α_1, α_2 and α_3 are weight parameters which sum up to 1.

The slope s of a cell is calculated by fitting a plane in a circular region around the cell with a diameter corresponding to the maximum diameter of the robot. The angle between the plane normal and the z-axis of the global coordinate frame gives the slope inclination s. The terrain roughness r is calculated as the standard deviation of the terrain height values from the computed plane in the circular region around the cell.

The step height h is computed in two steps. First, local height differences within a square window of several (e.g. 11×11) grid cells are computed for all cells in the circular region. If the maximum height difference between any cell in that window and the center cell of the window is greater than the critical step height h_{crit} and the slope between the corresponding two terrain points is higher than the critical slope s_{crit}, the maximum height difference is stored as the temporary step height of the central cell of the window. Second, the step height of the central cell of the circular region is computed as

$$h = min(h_{max}, h_{max} \cdot \frac{n_{st}}{n_{crit}}), \tag{3}$$

where h_{max} is the maximum temporary step height in the circular region, n_{st} is the number of cells in the circular region whose temporary step heights are higher than the critical step height and n_{crit} is the valid number of cells (e.g. 50) with a temporary step height higher than the critical step height. This method for calculating the step height also detects small steep slopes as steps and is robust to missing terrain information.



(a) DEM (b) Danger Value

(c) Slope (d) Roughness (e) Step Height

Fig. 3. Danger value computation from the criteria slope, roughness and step height ($s_{crit} = 20°$, $r_{crit} = 30$ mm, $h_{crit} = 50$ mm, $\alpha_1 = 0.5$, $\alpha_2 = 0.25$, $\alpha_3 = 0.25$)

Fig. 3 illustrates the computation of the danger value from the three criteria. It shows that the step height is well suited for detecting whether a cell is traversable or not, but it provides little information about the difficulty of the traversable cells. By contrast, the slope and roughness criteria can fail to detect untraversable cells but are better suited for estimating the difficulty of traversable cells.

The terrain traversability is reestimated every time a new local map is added to the global map. It is not necessary to recompute danger values for the complete global map. Only the cells within the area of the new local map surrounded by a border of the size of the robot have to be reestimated.

As mentioned above, artifacts can be present in the DEM, which must not be detected as terrain hazards (ref. Fig. 4). For this reason, only height values which were detected within a certain range Δf of frame numbers are considered in the traversability estimation process. When reestimating the danger of a cell that was detected impassable in the previous estimation step, height values of a greater frame range $\Delta f' > \Delta f$ are considered than when reestimating a previously traversable cell. Practical tests showed that taking only height values within Δf into account often causes untraversable cells to be detected as traversable cells because nearby hazards are not covered by the images. Thus, also considering older height values within $\Delta f'$ for reestimating the traversability of previously untraversable cells gives more realistic results.

The traversability of a cell is only computed if a sufficient number of height values is present in the circular rover-sized region around the cell. In addition to the danger value, a certainty value is calculated as the percentage of available height values in the circular region.

D. Path Planning

Based on the danger map, the robot can plan a path to the goal point. Since the robot does not have an a priori map

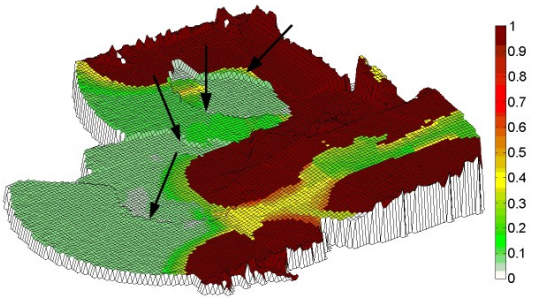
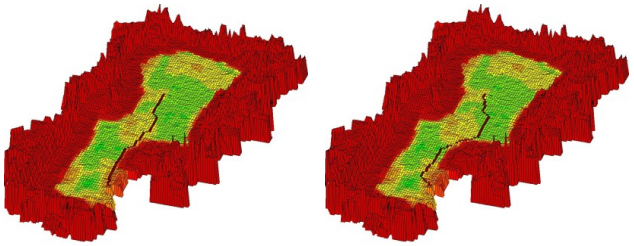


Fig. 4. Traversability estimation of a DEM with artifacts caused by errors in position estimation: By considering only height values with frame numbers within $\Delta f = 20$ for passable and $\Delta f' = 200$ for impassable cells artifacts are not detected as hazards.



(a) $\beta = 3$ (b) $\beta = 10$

Fig. 5. Paths planned with different values of β.

of the environment, its knowledge about the terrain changes over time. The path planner must be able to adapt the path to changes in the map in an efficient way. Thus, a path planner of the D* family was chosen. D* developed by Stentz [13] is the dynamic version of the A* graph based path planning algorithm. These search algorithms find the minimum cost path to a goal vertex in a graph by first searching those vertexes which most likely appear to lead to the goal. In contrast to A* planners, D* is able to modify previous search results locally and is thus more efficient when dynamic replanning is required. For the present navigation system a D* Lite [8] path planner was used, which is simpler and more efficient than the classic D* algorithms.

To apply the D* Lite planner to the grid map, the map has to be considered as a graph. The grid cells are the vertexes of the graph and edges connect vertexes which correspond to adjacent cells in the grid map. As for the A* algorithm, a cost function and a heuristic distance function must be implemented for the D* Lite path planner. However, D* Lite plans a path in opposite direction from the goal vertex G to the start vertex S. The cost function $c(N, N')$ describes the cost for moving from vertex N to its neighbor N'. The heuristic distance function $h(N, S)$ is an estimate of the costs remaining to reach the start vertex from the current vertex N. The formulation of the cost function defines the optimality of a path. Often, a path is optimal if it is the shortest path to the goal. In the present work, not only the path length but also the traversability of the path cells should be taken into account. Thus, the cost function for going from vertex N to its neighbor N' is

$$c(N, N') = \sqrt{(N_x - N'_x)^2 + (N_y - N'_y)^2} + \beta \cdot d(N'). \quad (4)$$

The first term describes the distance between the vertexes and the second term denotes the danger value of the destination vertex weighted by $\beta > 1$. The bigger the value of β is chosen the longer paths are accepted if they go through safer cells (ref. Fig. 5). The costs of going to an untraversable cell are ∞.

The heuristic distance function is important for the efficiency of the planner. For the planner to be optimal it must fulfill the monotony condition

$$h(N, S) \leq c(N, N') + h(N', S). \quad (5)$$

That implies that $h(N, S)$ must not overestimate the true costs $h^*(N, S)$ of going from N to S along the shortest path. Since the minimum danger value of a cell is 0, the heuristic distance function must be

$$h(N, S) = \sqrt{(N_x - S_x)^2 + (N_y - S_y)^2}, \quad (6)$$

which is the direct distance from N to S. Due to triangle inequality, this function also fulfills the monotony condition.

In the beginning, the robot does not have any information about its environment. It plans an initial path, which is the direct path to the goal. As the robot follows this path, it collects information about the environment. If assumptions about the traversability of the path cells are proven wrong by new data, the path is replanned from the robot's current cell.

E. Motion Control

Path following is achieved by a simple proportional controller which commands a forward speed and a turn speed to the robot and is independent from the robot's kinematics.

Due to the narrow field of view of the stereo camera, often there is not enough information about the terrain to calculate a danger value with high certainty. Thus, the robot has to actively explore the area surrounding the path by exploration turns. If the certainty value of a path cell which is in range of the cameras is lower than 1, the surroundings of the path cell have to be explored by an exploration turn. During an exploration turn the robot turns over an angular range of 2ϵ so that the cameras cover the rover-sized circular region around the path cell being explored. Since a certainty value of 1 is hard to reach in practice, a set of rules about when exploration turns are permitted has been established: Between two exploration turns

- the distance between the path cell to be explored and the previously explored path cell must be at least l

and one of the following conditions must hold:

- The robot must have passed a distance of at least l.
- The path cells to be explored in two subsequent exploration turns must be at an angle of at least ϵ given the current robot position is the angular point.
- The path must have been replanned.

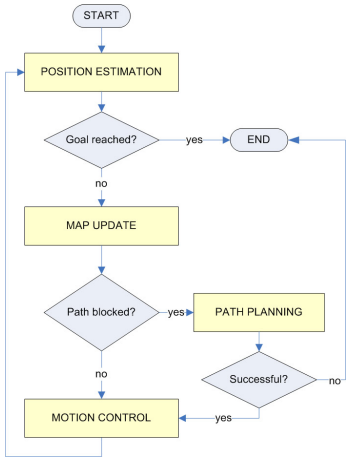


Fig. 6. Diagram of the navigation algorithm

These rules are necessary to avoid that the robot repeats exploration turns because the certainty value does not reach 1. For the hardware used in the practical tests the values were chosen to be $\epsilon = \pi/8$ and $l = 0.2$ m.

The navigation procedure finishes successfully when the robot reaches the specified goal point due to its estimated position. Since the robot cannot reach the goal point exactly, a tolerance area around the goal point must be defined. The size of the tolerance area depends on the map resolution and the distance the robot travels within one navigation step. In the present work, the tolerance area is a circle with a radius of 40 mm. As soon as the robot's estimated position is inside that tolerance circle, the robot stops.

F. Summary of the Navigation Algorithm

Fig. 6 shows a diagram of the navigation algorithm. In the beginning, the goal coordinates are given to the robot and the robot plans an initial path to goal. The main loop starts with capturing a stereo image and estimating the robot's position by visual odometry. If the robot's estimated position is within the tolerance area around the goal, the robot stops and the algorithm terminates. Otherwise the danger map is updated using the robot's position and the depth image computed in the position estimation step. If cells of the planned path are affected by the changes in the danger map, the path is replanned. When path planning is successful or when the path does not have to be replanned, the motion control module is activated. It decides whether the robot can follow the path or has to explore the environment by an exploration turn and it commands a forward and turn speed to the robot. Then a new stereo image is taken. If the path planner cannot find a path to the goal, the robot stops and the algorithm terminates with showing an error message to the operator.

III. PRACTICAL TESTS

The navigation algorithm was implemented and tested on a wheeled and a legged robot. The wheeled platform is a commercial Pioneer 3-DX robot with two driven wheels and a caster wheel. The second mobile robot is the DLR-Crawler [4] – a six-legged walking robot of about 50 cm in diameter developed at DLR Institute of Robotics and Mechatronics. It can walk using different gait patterns and reflexes. Both robots are equipped with equal stereo camera pairs. To determine the true positions of the robots at any time, an external tracking system was used. It consists of three infrared ARTtrack cameras and a reflecting target body, which was attached to the robots.

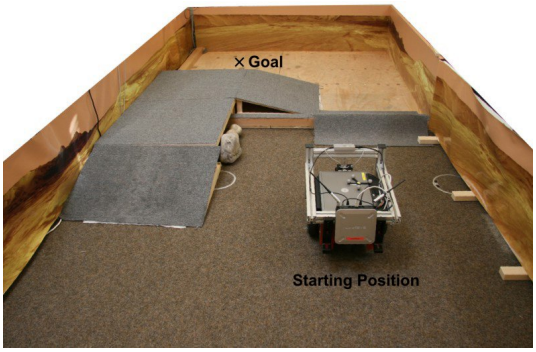


Fig. 7. Test environment for the Pioneer robot

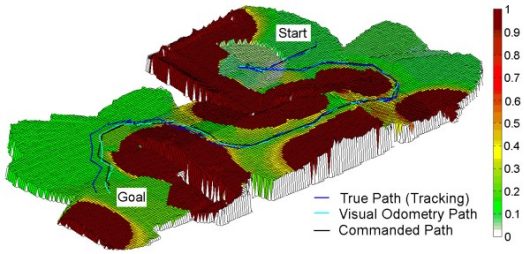


Fig. 8. Resulting danger map and traveled paths

A. Test on a wheeled robot

For testing the algorithm on the Pioneer robot, an environment with ramps and platforms was created (ref. Fig. 7). The goal coordinates are given to the robot relative to its starting position (0.8 m to the left, 2.8 m forward). The only way for the robot to reach the goal is finding a path over the ramps and platforms. Since the robot does not have any information about its environment, the initial path is planned directly to the goal. The robot starts traversing this path until it encounters an obstacle which blocks the direct path to the goal. The path planner replans the path to avoid the obstacle. Every time the planned path is blocked by a previously undetected obstacle, the path is replanned. By doing so, the robot finds the way to the goal and stops when the goal point is reached with sufficient accuracy.

The robot performed eight successful runs in this environment. Fig. 8 shows the DEM with danger values and the commanded path, the traveled path measured by visual odometry and the traveled path measured by the external tracking system for one run. The robot traveled 8.4 m on average to reach the goal. After successful termination of the navigation algorithm, the average distance between the true robot position and the goal position is 120 mm or 1.4% in relation to the path length. The distance between

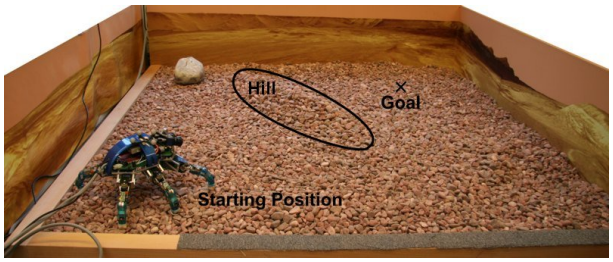


Fig. 9. Test environment for the DLR-Crawler

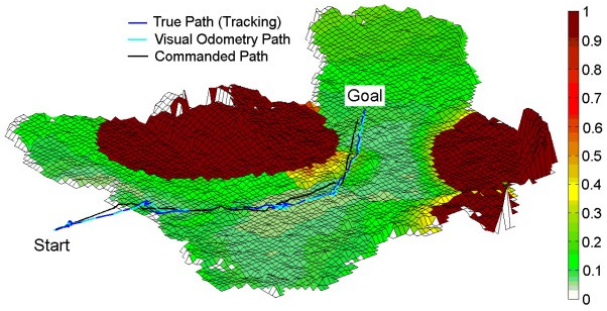


Fig. 10. Resulting danger map and traveled paths

the robot's true position and its estimated position based on visual odometry is 108 mm or 1.3% in relation to the traveled distance.

B. Test on a six-legged walking robot

For tests on the DLR-Crawler an uneven environment was built of gravel (ref. Fig. 9). The small hill in the middle of the testbed is too steep for the crawler. The goal point is located 1.4 m in front of the robot behind the hill. The robot has to detect that the hill is untraversable and find a path around the hill to the goal point. For this test the DLR-Crawler uses a fixed tripod gait pattern with leg stretch reflexes.

Fig. 10 shows the danger map and the commanded path, the traveled path measured by visual odometry and the traveled path measured by the external tracking system. The distance traveled by the robot in a single run is 2.5 m and the accuracy of reaching the goal point is 72 mm or 2.9% in relation to the traveled distance. The distance between the robot's true position and its estimated position based on visual odometry is 29 mm or 1.2%.

The non-optimized implementation of the navigation algorithm runs at an average frame rate of 1 Hz on a standard 2.2 GHz processor.

IV. CONCLUSION

A stereo camera based navigation system for rough terrain was presented which is suitable for wheeled and legged robots. The stereo camera is the only sensor and thus used for position estimation and terrain modeling. The traversability of the terrain is estimated by computing a danger value from the criteria slope, roughness and step size. For efficient replanning a D* Lite path planner was implemented. The terrain traversability is considered in the path planning process to plan a short and safe path. The robot follows the planned path and actively explores its environment to improve the traversability estimation of the terrain.

The practical tests showed that the presented navigation system works well on wheeled and legged robots. The errors of reaching the goal points are less than 3% of the traveled paths on traverses of a few meters length.

In future, the localization error has to be limited by storing landmarks of the environment or employing an additional tilt sensor for absolute pitch and roll estimates. Furthermore, a foothold planner for walking robots has to be developed which computes footholds locally as the robot moves along the planned path. The foothold planner can act as a second layer upon the presented navigation system and will enable walking robots to take full advantage of their walking capabilities.

V. ACKNOWLEDGMENTS

We would like to thank Martin Görner for his help in the practical tests.

REFERENCES

[1] J. Carsten, A. Rankin, D. Ferguson, and A. Stentz. Global Path Planning on Board the Mars Exploration Rovers. *Aerospace Conference, 2007 IEEE*, pages 1–11, March 2007.

[2] M. Daily, J. Harris, D. Keirsey, D. Olin, D. Payton, K. Reiser, J. Rosenblatt, D. Tseng, and V. Wong. Autonomous cross-country navigation with the ALV. *International Conference on Robotics and Automation, 1988. Proceedings., 1988 IEEE*, 2:718–726, April 1988.

[3] S. B. Goldberg, M. W. Maimone, and L. Matthies. Stereo vision and rover navigation software for planetary exploration. *Aerospace Conference Proceedings, 2002. IEEE*, 5:2025–2036, March 2002.

[4] M. Görner, T. Wimböck, A. Baumann, M. Fuchs, T. Bahls, M. Grebenstein, C. Borst, J. Butterfass, and G. Hirzinger. The DLR-Crawler: A Testbed for Actively Compliant Hexapod Walking Based on the Fingers of DLR-Hand II. In *IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems*, pages 1525 – 1531, September 2008.

[5] H. Hirschmüller, P. R. Innocent, and J. Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*, 47(1-3):229–246, April-June 2002.

[6] H. Hirschmüller, P. R. Innocent, and J.M. Garibaldi. Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics. In *Proceedings of the 7th International Conference on Control, Automation, Robotics and Vision*, pages 1099–1104, December 2002.

[7] A. Kelly. *An Intelligent Predictive Control Approach to the High-Speed Cross-Country Autonomous Navigation Problem*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, September 1995.

[8] S. Koenig and M. Likhachev. Improved fast replanning for robot navigation in unknown terrain. In *Proceedings of the International Conference on Robotics and Automation*, pages 968–975, 2002.

[9] S. L. Laubach, J. Burdick, and L. Matthies. An autonomous path planner implemented on the Rocky7 prototype microrover. In *IEEE Conference on Robotics and Automation*, pages 292–297, 1998.

[10] D. P. Miller, R. S. Desai, E. Gat, R. Ivlev, and J. Loch. Reactive navigation through rough terrain: experimental results. In *Proceedings of the 1992 National Conference on Artificial Intelligence*, pages 823–828, 1992.

[11] A. H. Mishkin, J. C. Morrison, T. T. Nguyen, H. W. Stone, B. K. Cooper, and B. H. Wilcox. Experiences with operations and autonomy of the Mars Pathfinder Microrover. *Aerospace Conference Proceedings, 1998. IEEE*, 2:337–351, March 1998.

[12] R. Simmons, L. Henriksen, L. Chrisman, and G. Whelan. Obstacle avoidance and safeguarding for a lunar rover. In *Proc. AlAA Forum on Advanced Developments in Space Robotics, Madison WI*, 1998.

[13] A. Stentz. Optimal and efficient path planning for partially-known environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3310–3317, 1994.