Terrain Optimized Nonholonomic Following of Vehicle Tracks

T. Kopfstedt * **M. Restle** ** **W. Grimm** **

* *Research and Development, Diehl BGT Defence GmbH & Co. KG, Ueberlingen, Germany, (e-mail: Thomas.Kopfstedt@diehl-bgt-defence.de)*
** *Institute of Flight Mechanics and Control, University of Stuttgart, Stuttgart, Germany, (e-mail: werner.grimm@ifr.uni-stuttgart.de)*

Abstract: Unmanned Ground Vehicles (UGV's) have a large field of application. For convoy following or platooning tasks the UGV must be able to follow a path specified by a leading vehicle. This path must be adjusted if it is blocked by an obstacle or not feasible for the following robot due to its nonholonomic constraints. This paper presents a new algorithm for the nonholonomic path planning in rough terrain for leader-follower tasks, whose major advantage is the fast and efficient computation and the consideration for the terrain traversability during the planning phase. It can easily be adjusted to additional conditions like platooning tasks, scenario dependent restrictions or vehicles sizes and is therefore universally applicable to all kinds of UGV's.

Keywords: nonholonomic path planning, robot navigation, terrain based, vehicle following, intelligent vehicle, A*

1. INTRODUCTION

Over the last few years there has been active research on navigation of intelligent vehicles. For Unmanned Ground Vehicles (UGV's) there exists a large field of applications. For example an intelligent trailer could follow a harvester, or a transport vehicle could follow a convoy in conflict regions or in disaster areas after an earth quake or a flood, so that there is no need to endanger human life. To provide this ability the UGV must be able to follow a path specified by a leading vehicle. This path must be adjusted if it is blocked by an obstacle or if it is not feasible for the following robot due to its nonholonomic constraints.

For indoor applications many multiple robot interactions for small scale robots like leader-follower formations (Shi-Cai et al. (2007)) or platooning tasks (Matko et al. (2008)) have been studied. Most of them consider a flat terrain and use identically constructed robots. The indoor environment can be described by only two states - free or blocked by an obstacle. An extension to car-like robots with nonholonomic constraints is made in Graf et al. (2001). There the elastic bands method (Quinlan et al. (1993)) is extended to nonholonomic vehicles.

For outdoor purposes the path planning task has to take into account different traversability conditions of the terrain such as mud, pavement, tall grass, bushes, trees, and rock piles. In Seraji et al. (2002) a terrain-based navigation is presented which uses range dependent traversability indices. An easy way to consider the terrain quality is used in Tay et al. (2002). There the modified distance transform (MDT) algorithm is used on a simple calculation grid that allows the fast calculation of the path. To ensure the feasibility of the path the VFH+ algorithm (Borenstein et al. (1998)) is used to adjust the path *after* planning. This can lead to unexpected situations where the preplanned path is not feasible due to the kinematic constraints.

Another algorithm (Howard et al. (2008)) takes the nonholonomic constraints into account already during the planning phase. A large number of offline precomputed trajectories is used to compute a feasible nonholonomic path. In Lacaze et al. (1998) a state-based sampling strategy is applied to generate path sets that consider global guidance and satisfy environmental constraints while also guaranteeing dynamic feasibility by using a model-predictive trajectory generator. This leads to an enormous computational effort at each planning cycle.

In this paper, a new algorithm for the nonholonomic path planning in rough terrain with additional conditions like leader following is presented. Based on the A* algorithm (Hart et al. (1968)) and a grid map that contains the discretized traversability classes of the terrain. An advanced directional search is developed to consider the nonholonomic constraints. The major advantage is the fast and efficient computation and the easy way to extend it to additional conditions, e.g. platooning tasks, scenario dependent restrictions or vehicles sizes.

This paper is organized as follows: Section 2 describes the experimental vehicle used and gives an overview of the whole control structure. In section 3 the main planning algorithm is presented. Some simulation results are depicted in section 4. Finally some concluding remarks and guidelines on future work are given in section 5.

2. SYSTEM OVERVIEW

The path planning algorithm presented in this paper is implemented in a car-like UGV called MUSTANG Mk 1. In the following, a brief overview of its main features is given.

2.1 Platform kinematics

Fig. 1. UGV MUSTANG Mk 1 from Diehl BGT Defence

The MUSTANG Mk 1 is a medium sized (480kg, 2.2m length) hybrid vehicle for 3-D missions (dirty-dull-dangerous). It participated in the ELROB 2008, listed as CANGURU and can be seen in Fig. 1, equipped with CCD cameras, radio communication, as well as radar based overground sensors, ultrasonic and laser range sensors. It is able to fulfill reconnaissance, transport and observation missions. In the convoy following mode the position of the leader is estimated from either the camera signal or directly from a transmitted GPS signal through a Kalman filter. For self localization the MUSTANG Mk 1 uses an integrated navigation system containing a GPS, an inertial measurement unit (IMU) and a magnetic sensor. The main feature is that all four wheels can be driven and steered individually allowing maximum motion flexibility. For the sake of simplicity it is assumed that the front and the rear axis are always steered with the same amount δ but with opposite directions. This leads to a minimal turning radius, as shown in Fig. 2. Such a vehicle can be approximated using a bicycle model as presented in (Wang et al. (2001)). C is the reference point at the center of gravity, x and y are its Cartesian coordinates. The vehicle length and heading angle are denoted as l and θ respectively.

The bicycle model is represented by the following equations of motion:

$$\dot{x} = v\cos\theta \tag{1}$$

$$\dot{y} = v\sin\theta \tag{2}$$

$$\dot{\theta} = \frac{2\cdot v}{l}\tan\delta\ , \tag{3}$$

where the velocity v and the steering angle δ of the vehicle are the controllable inputs of the system.

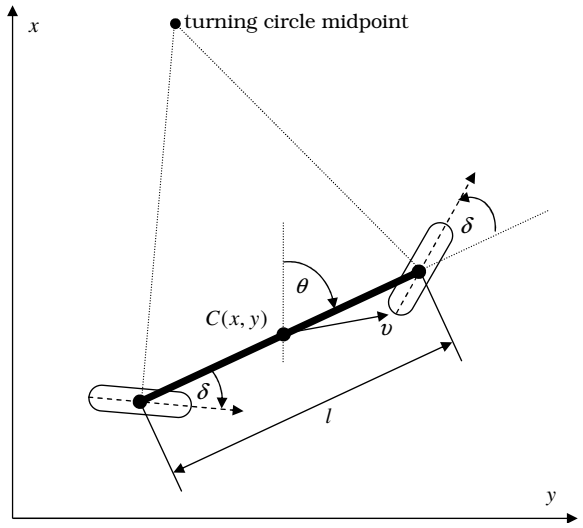

Fig. 2. Simplified bicycle model

2.2 control structure

The control section consists of three parts: the distance controller, the path controller and the collision avoidance, as depicted in Fig. 3. To follow the leader at a desired distance a velocity PID-controller is used. The control objective of the path controller is to ensure that the vehicle will correctly follow the reference path. For robustness reasons a sliding mode path following control is used, similar to the one described in Solea et al. (2006), getting reference values from the path planner described in section 3. The underlying collision avoidance uses the VFH+ method (Borenstein et al. (1998)) to adjust the path controller commands, if an obstacle occurs between two path planning cycles.

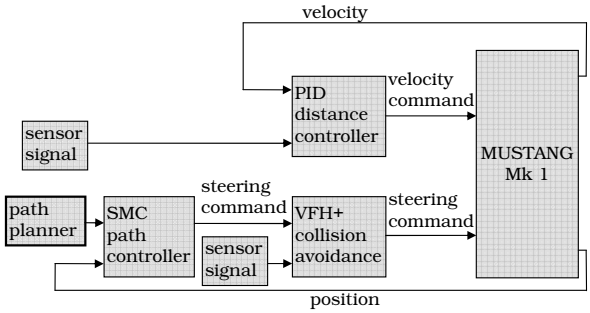

Fig. 3. Control structure

3. THE PATH PLANNING ALGORITHM

The main task of the path planner is to find a feasible, collision free path along the trajectory of the leading vehicle. The planner runs at 2 Hz and has a field of view of 40 m x 40 m mapped on a grid with 0.25 m cell size. At each planning cycle, a new optimal path on this grid is searched with a new specialized A* algorithm that is presented below. The output of the A* search is a polygon of linked linear segments. For a smooth reference path, these segments have to be interpolated using an arbitrary spline representation.

3.1 Consideration of the path of the leader

Onboard sensors measure the actual position only of the leader. To map the track of the leader in a local geo referenced map the leader position must be stored in a buffer. The position data can be illustrated as bread-crumbs $[x_{bc,i},\ y_{bc,i}]$ which are scattered along the path. At each planning cycle, these position nodes are loaded from the buffer and transformed into vehicle coordinates (marked with index $_b$) using

$$\begin{bmatrix} {}_bx_{bc,i} \\ {}_by_{bc,i} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{bc,i} - x \\ y_{bc,i} - y \end{bmatrix} \tag{4}$$

with $i = 1, ..., n$. Starting from the oldest one, each position node is checked if ${}_bx_{bc,i} < 0$. If this is the case, the node lies behind the follower and can be discarded. The first node k for which ${}_bx_{bc,i} > 0$ indicates that from now on the position nodes lie in front of the follower and that they have to be mapped to the local map. If the value lies inside of the map, the corresponding cell has to be marked as a path cell. Then the next position node is checked. If the value is outside of the map, the calculation is finished and the last marked cell will be chosen as the temporary goal for the path planner. Fig. 4 shows the mapping of the position nodes to the local map. The red points are the discardable position nodes, the green ones are necessary for the path planning.

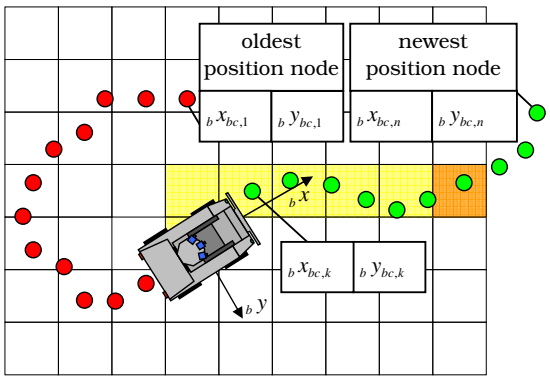

Fig. 4. mapping of the leader path to the local map

Based on the geometrical representation additional path planning conditions can be implemented very easily and efficiently. For example a platooning task can be solved by defining additional makers around the mapped leader path. Fig 4 shows the path for two follower robots in a platooning task as red marked cells.

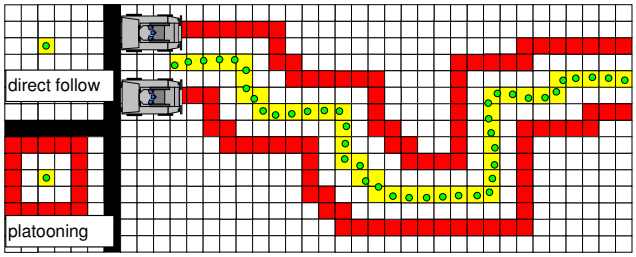

Fig. 5. simple solution for a platooning task

With this method the two follower robots are able to escort the leader with an additional constant lateral offset to the left or to the right side of the leader path.

3.2 Terrain consideration

Environment sensors like laser range finders calculate the height of the surrounding terrain and pass a height map to the path planner. Therefore, a discretization as for the leader path is not needed. However, not the absolute height is important, rather the height gradient is relevant. Especially when driving over a ridge, the vehicle can tilt over or draggle. Hence, a gradient map is calculated and a threshold is defined. All cells with a higher value than this threshold are declared as blocked. In Fig. 6a)-c) this process is demonstrated with a threshold of 5.

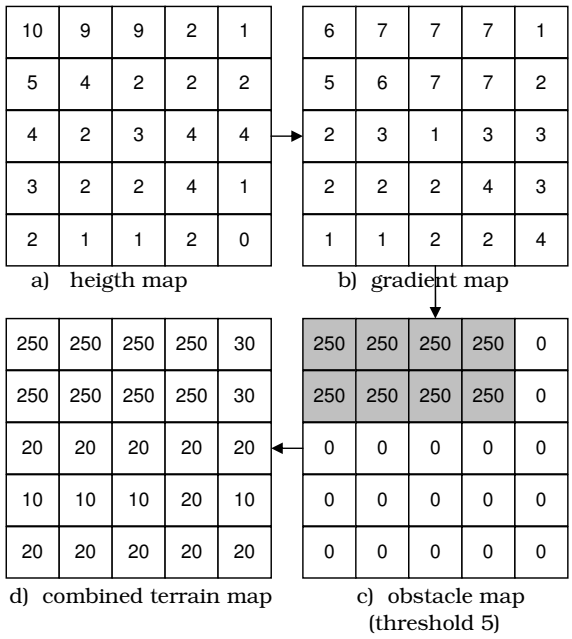

Fig. 6. Generation of the terrain map

In addition to this obstacle map, a terrain traversability map is constructed. This can be done by either using the height gradient or by combining laser data with camera signals and image processing. The latter method is applied in this project. The terrain quality is split into 16 classes as listed in Tab. 1.

Table 1. terrain quality classes

description	value
street - (center of the actual track)	10
street (paved surface, tar)	20
pathway - (center of the actual track)	30
pathway (grit, sand)	40
drivable 1 (flat terrain)	110
drivable 2	120
drivable 3	130
drivable 4	140
drivable 5	150
drivable 6	160
drivable 7	170
drivable 8	180
drivable 9	190
drivable 10 (tall grass, mud)	200
impassable (trees, busher, rocks)	250
unknown	255

These values are combined with the obstacle map as can be seen in Fig. 6d. Now all terrain information can be represented by this combined terrain map.

3.3 Nonholonomic constraint consideration

The standard A* algorithm on a grid map uses a graph representation with 4-point or 8-point connectivity. A cell is reachable, if it is a direct neighbor of the current cell and not blocked by an obstacle, as shown in the left two pictures of Fig. 7. This definition of reachability eads to unfeasible changes in the direction of the path and cannot be used for car-like robots.

To consider these constraints the direction from which a cell has been reached has to be used for an adaptive connectivity. Instead of using four or eight neighbors there are now only three possible neighbors, describing the next cell passed for straight forward driving, driving with the maximum steering angle to the left or driving with the maximal steering to the right. The greater the turning radius, the longer and thinner are these connectivity lines (see the two pictures on the right in Fig. 7). To avoid sharp alteration from left to right corners the connectivity line for straight forward driving has also been stretched.

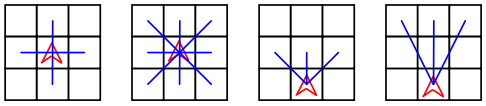

Fig. 7. 4-point, 8-point, and directed 3-point connectivity

By stringing these elements together the whole area can be covered with respect to the nonholonomic constraints and the direction in which the vehicle is orientated during its movement. Thereby a kind of tree structure is generated as it can bee seen on the left side of Fig. 8 in which the colors correspond to the depth of the planning step. The minimum turning radius approximated with this structure is shown on the right-hand side of Fig. 8.

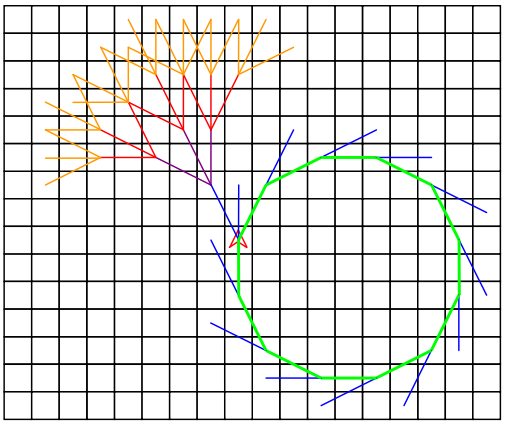

Fig. 8. Nonholonomic path generation

It is possible that a cell is reached by multiple branches from different directions. Associated to these directions are different suitable neighbor cells. Therefore in addition to the x and y coordinates of the cell the direction from which the cell has been reached has to be stored during the planning phase.

This extension allows it to specify not only the x and y coordinates of the goal, but also the orientation at this point. The two dimensional problem of finding a path in a plane is indeed expanded to a three dimensional one (x, y, direction). But because there are only three instead of eight neighbors at each step and due to the longer branches [1] there are fewer cells to check. Therefore the computational effort is not much higher than in the two dimensional case without considering the nonholonomic constraints.

Finally the size of the vehicle has to be taken into consideration. This can easily be done by not only checking the cells which are covered by the connectivity lines, but also check all the cells surrounding them with a radius equal to the robot size as shown in Fig. 9.

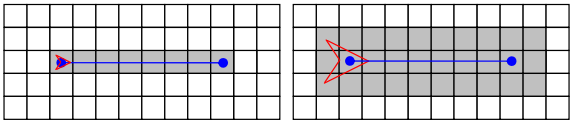

Fig. 9. Consideration of the vehicle size

3.4 Planning progress

The first step at each planning cycle is the calculation of the local maps as illustrated in Fig. 6. In the second step, these maps have to be combined to a planning map. To adjust the priority between terrain optimization and deviation of the leader path, some weighting factors have been used. The pseudocode of this algorithm looks as follows:

```
1:  compute terrain map
2:  map leader path to local map
3:  compute start point
    (last point which was used by the path controller or
    the current follower position in the initial step )
4:  add start point s to open list O
5:  do
6:    pick best point a from O such that f(a) ≤ f(j), ∀j ∈ O
7:    ∀ neighbors n of a which are not in closed list c check
8:    if neighbor i is reachable, with i = 1...n
      (i is reachable if it is connected by one of the three
      branches and not blocked by an obstacle)
9:      compute path cost g(i)
        including weighted terrain traversability
10:     if i ∉ open list O
11:       add i to open list O
12:     else if g(i_new) < g(i_old)
        (point has already been reached, check if actual
        path is shorter than the old one)
13:       update i's backpointer to a
14:     end if
15:   end if
16: while O ∉ ∅ or goal has been reached
```

With the path cost function g, the A* heuristic h the priority value is defined as $f = g + h$.

This has to be done in every planning cycle. If the size of the local map is chosen carefully with respect to the velocity of the robot, it can be ensured that the following

[1] For a vehicle with a turning circle of about 4m and a grid map whose grid size is 0.25m, the branches cross 3 cells.

robot never reaches the goal point of the actual planning cycle. Thus the path can be continuously planned and tracked.

Fig. 10 shows the output of an A* search for one planning cycle in which only obstacles (marked as black cells) as well as start and goal point are considered. The leader path in this example is generated by a holonomic object, e.g. a human operator. Using the standard implementation, the generated path has sharp turns and is not feasible for a car-like vehicle. The path differs from the leader path each time the leader path makes a movement which does not directly lead to the final position.

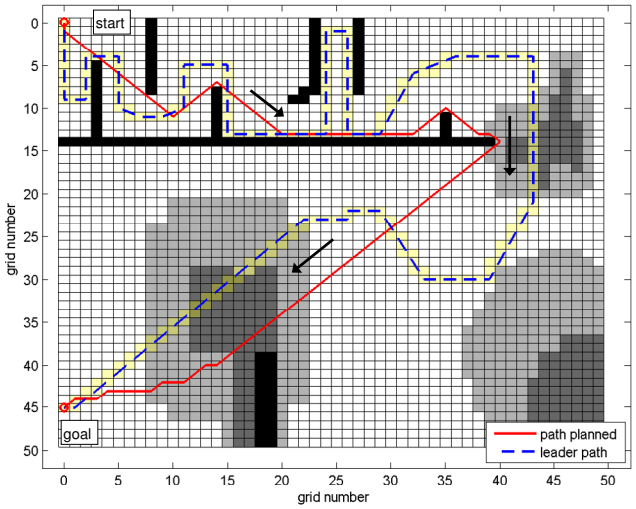

Fig. 10. Standard A* path search

Fig. 11 demonstrates the capabilities of the extensions of the the A* algorithm made in this paper. The generated path has no sharp turns and is therefore feasible. The path of the leader is tracked with high accuracy, and is only adjusted, if it was not feasible.

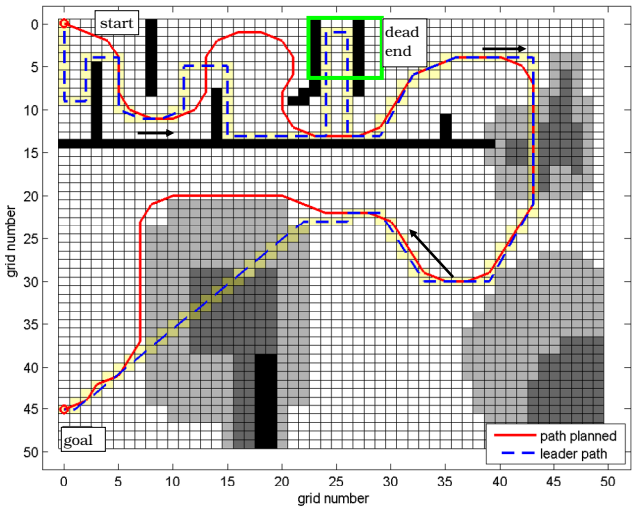

Fig. 11. New nonholonomic, terrain optimized path plan

This can be seen at the top of Fig. 11 where the leader makes a sharp change of direction. Instead of following the leader directly, the planner considers the minimum turning radius and generates a path with a larger arc. There is also a dead end. If the robot would follow the leader through this narrow gap, it would not be able to get out of it without driving backwards. The algorithm realizes that problem and chooses another feasible path. Furthermore, the traversability of terrain is considered, this can be seen at the bottom of Fig. 11 where the leading vehicle crosses a 'bad' terrain (e.g. a hill) that is marked with the dark background color of the cells. In this case the leader path has been left and a better path which avoids the 'bad' terrain has been chosen.

4. SIMULATION RESULTS

For verification purposes a simulation testbed has been implemented in Matlab Simulink. To use the same C source code as on the MUSTANG Mk 1, C-S-functions were used. To generate a representative input source for the path planner a test run with the MUSTANG Mk 1 has been made. Thereby the GPS data has been recorded, so that the testbed can use this data as the GPS position of the leading vehicle, which shall be followed. The following robot is simulated using the model described in section 2 with an additional delay on the input values to account for the actuator dynamics.

In the first example (Fig. 12) the mission task is to follow the leader with a distance of 6m with an initial position error of $x_0 = 4m$, $y_0 = 0m$ and $\theta_0 = 30°$. The leader path follows the street and therefore the planner does not have to adjust with respect to the terrain. The task is just to follow the leader path without cutting corners as it would be done when directly following the leader.

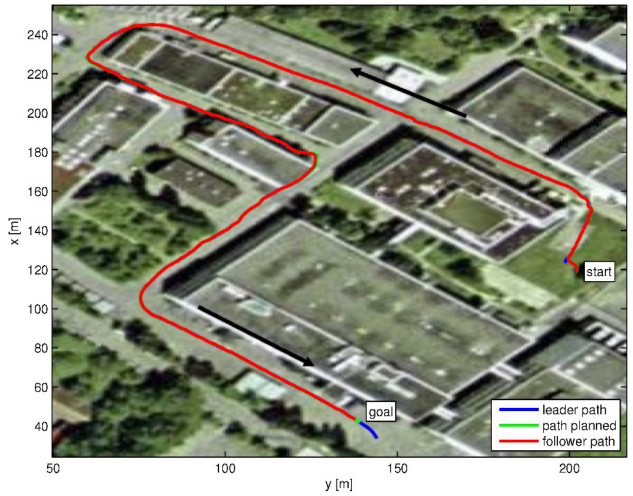

Fig. 12. x -y plot of the path (leader, planned and follower) with initial pose error of $x_0 = 4m$, $y_0 = 0m$ and $\theta_0 = 30°$

Fig. 13 shows the lateral error being defined as the distance between the vehicle and the closest point on the planned path. Due to the nonholonomic planning, the planned path can be followed with a root-mean-square deviation of only $\sigma_e = 0.1m$.

The corresponding steering commands can be seen on Fig. 14. They never reach or exceed the mechanical limit of $\delta_{max} = 20°$. This indicates that the directed 3-point connectivity elements are a good approximation of the real kinematics of the following robot.

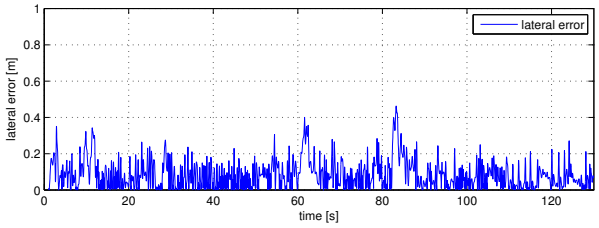

Fig. 13. Lateral path error

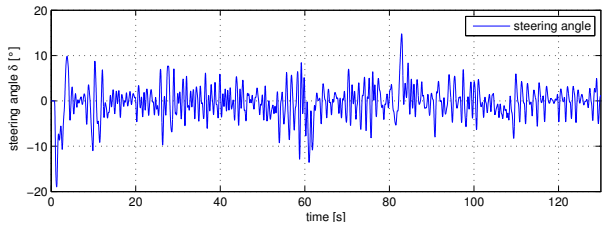

Fig. 14. Steering commands

To test the terrain optimization a simple test was made considering a human operator who moves on an arc and passes a tree nearby, shown as the blue line in Fig. 15. The planner has to adjust this path to avoid any collision with the tree. The red line in Fig. 15 shows that this challenge is correctly solved by the developed path planner. The path is adjusted where a collision would happen and is retained to the original path otherwise, so that an accurate following is achieved. The tradeoff between accurate following and terrain based adjustment can be regulated due to weighting factors in the planner.

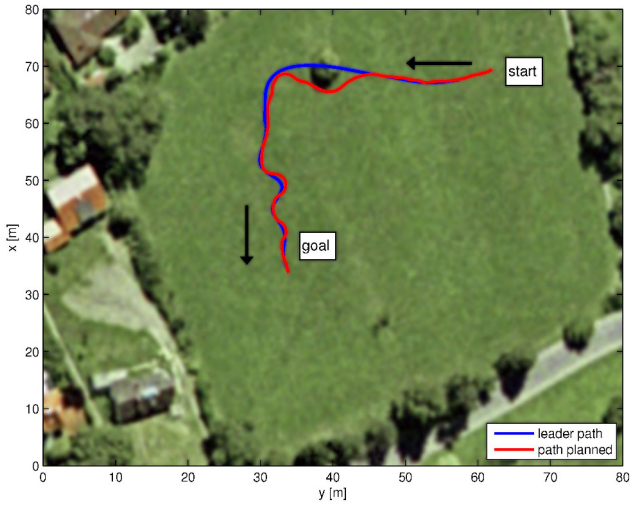

Fig. 15. Terrain based path adjustment

5. CONCLUSION

The path planning algorithm presented in this paper combines the low computational effort of a grid based A* path planner with the feasibility in presence of nonholonomic constraints, like restricted minimum turning radius. Moreover a solution to incorporate the terrain traversability has been found which is essential for path planning in rough terrain. The fast execution together with the local planning horizon allow to run the planner at a high frequency to deal with dynamic environments. Extensive simulations have shown the power of this algorithm so that it now can be implemented on the UGV MUSTANG Mk 1 for further development.

REFERENCES

Ulrich, I., Borenstein, J. (1998). *VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots*, Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 16-21, 1572-1577.

Graf, B., Hostalet Wandosell, J. M. (2001). *Flexible Path Planning for Nonholonomic Mobile Robots*, 4th European workshop on advanced Mobile Robots (EUROBOT'01), Fraunhofer Institute Manufacturing Engineering and Automation (IPA), Lund, Sweden, Sept. 19-21, 199-206

Hart, P., Nilsson, N., Raphael, B. (1968). *A formal basis for the heuristic determination of minimum cost paths*, IEEE Transactions on Systems Science and Cybernetics, Volume 4, 100-107.

Howard, T.M., Green C.J., Kelly, A., Ferguson, D. (2008). *State space sampling of feasible motions for high-performance mobile robot navigation in complex environments*, Journal of Field Robotics Vol. 25, No. 6-7, 325-345.

Lacaze, A., Moscovitz, Y., DeClaris, N., Murphy, K. (1998). *Path Planning for Autonomous Vehicles Driving Over Rough Terrain*, Proceedings of the ISIC/CIRA/ISAS Conference, Gaithersburg, MD, September 14-17, 50-55.

Matko, D., Klancar, G., Blazic, S., Simonin, O., Gechter, F., Contet, J.M., Gruer, P. (2008). *The application of reference-path control to vehicle platoons*. In ICINCO, Portugal, 145-150.

Quinlan, S., Khatib, O. (1993). *Elastic Bands: Connecting Path Planning and Control*, IEEE International Conference on Robotics and Automation, Vol. 2, Atlanta, USA, 802-807.

Seraji, H., Bon, B. (2002). *Multi-range traversability indices for terrain-based navigation*, International Conference on Robotics and Automation, Vol. 3, Washington D.C., USA, 2674-2681.

Shi-Cai, L., Da-Long T., Guang-Jun, L. (2007). *Robust Leader-follower Formation Control of Mobile Robots Based on a Second Order Kinematics Model*, Acta Automatica Sinica, Vol. 33, No. 9, 947-955.

Solea, R., Nunes, U. (2006). *Trajectory Planning with Velocity Planner for Fully-automated Passenger Vehicles*, Proceedings of the Intelligent Transportation Systems Conference (ITSC), Toronto, Canada, 474-480.

Tay, A., Jiang, S., Ibanez, G. J., Chan, C.W. (2002). *Autonomous vehicle navigation strategies - localized navigation with a global objective*, Proceedings of the IEEE of International Conference on Information Technology and Application.

Wang, D., Qi, F. (2001). *Trajectory Planning for a four-wheel-steering Vehicle*, Proceedings of the IEEE International Conference on Robotics & Automation, Seoul, Korea, May 21-26, 3320-3325.