accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009

Autonomous Offroad Navigation Under Poor GPS Conditions

T. Luettel, M. Himmelsbach, F. v. Hundelshausen, M. Manz, A. Mueller and H.-J. Wuensche

***Abstract*— This paper describes an approach for autonomous offroad navigation for the Civilian European Landrobot Trial 2009 (C-ELROB). The main focus in this paper is how to cope with poor GPS conditions such as occurring in forest environments. Therefore the expected and detected problems are stated, and methods how to achieve good autonomous driving performance due to less weigthing of inaccurate GPS information are presented. But first, an introduction how to combine obstacle-avoidance and path following behavior within the reactive so-called tentacles approach is given. Extensions for utilizing visual information for tentacle evaluation as well as the integration of geodetic information to gather information on the road network are presented afterwards. Finally the results gathered from the successful C-ELROB trials are analyzed.**

I. Introduction

Our robot MuCAR-3 (Munich Cognitive Autonomous Robot car, 3^{rd} Generation) participated at this year's Civilian European Landrobot Trial 2009 (C-ELROB) "Autonomous Navigation" scenario for the second time [1]. This scenario is characterized by the hard access path that has to be taken by the vehicle in order to follow the given sparse waypoints. The forest environment produces challenging GPS conditions with major disturbances and outages (see Fig. 1). To achieve a robust system under these circumstances, we need methods like less weigthing of inaccurate GPS information and raise the reactive parts of navigation, without neglecting the difficulties in choosing the right of possibly many driving options at crossings.

Two years ago in C-ELROB 2007, we first used our tentacle approach to reactive robot navigation to cope with the challenges posed by the rough terrain, with great success. For achieving a higher grade of robustness and to minimize the number of manual interventions, many improvements have been made, some of those will be presented in this paper. Evaluating a tentacle now also considers the visual appearance of the environment. Additionally, the tentacle approach now combines local obstacle avoidance with global path following behavior. The target track is generated by matching given waypoints with a road network. Finally, we summarize with our results of C-ELROB 2009.

II. MuCAR-3 Hard- and Software Architecture

A. Vehicle

The present autonomous robot car of UniBw is named MuCAR-3 (Munich Cognitive Autonomous Robot Car 3^{rd} Generation), a VW Touareg vehicle equipped with an automatic gearbox and a very powerful generator. The vehicle has full drive-by-wire modifications and appropriate sensors to allow for autonomous driving.

All authors are with department of Aerospace Engineering, Autonomous Systems Technology (TAS), University of the Bundeswehr Munich, Neubiberg, Germany.
Contact author email: thorsten.luettel@unibw.de

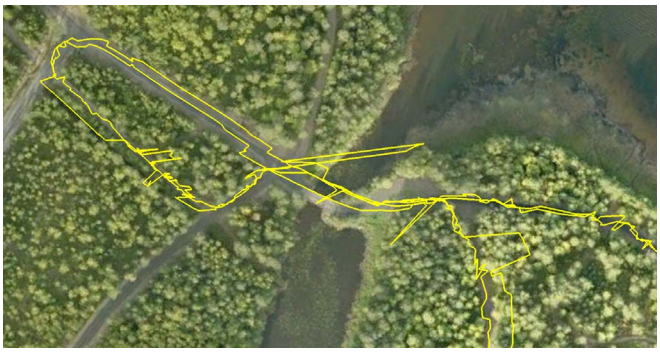


Fig. 1. Aerial view of an extract of C-ELROB's Autonomous Navigation Trial. The yellow line indicates the untrustworthy GPS measurements in wooded offroad terrain, showing errors up to 50 m [all aerial images: www.bing.com].

MuCAR-3 has actuators for steering, throttle, brake, the parking brake and also the gearshift of the automatic gearbox. All can be controlled from the low level computers with electrical signals.

B. Sensors

In addition to the sensor signals which are provided by the stock vehicle's CAN bus such as velocity or steering angle, MuCAR-3 is equipped with additional sensors.

Our main sensor in C-ELROB 2009 scenario is the *Velodyne LIDAR system* (Light detection and ranging), a state-of-the-art high-resolution laser range scanner mounted at the vehicle's roof. It generates a cloud of 100000 3D points at 10 Hz.

For visual perception, we use *Camera Platform MarVEye 7* (Multifocal active / reactive Vehicle Eye), which is build up similar to the human eye, with two wide angle cameras and a high resolution tele camera, inertially stabilized by an additional gyro moving the mirror. We use AVT Guppy Firewire color cameras. The image recording of the three cameras is synchronized by a hardware trigger signal, which is synchronized to the LIDAR measurements and the current gaze direction.

We use the RT3003 *Inertial Navigation System* (INS) from Oxford Technical Solutions. The INS consists of a 6-dof IMU (Inertial Measurement Unit) coupled to a differential GPS. Our dual antenna model allows for determination of heading after power-up without moving the vehicle.

accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009

Fig. 2. MuCAR-3 moving over the occupancy grid and accumulating obstacles, with underlying aerial image texture. Dark colors indicate low obstacle probability, light colors indicating high obstacle probability. Note the large transparent grey area not hit by any beam, thus having obstacle probability of 0.5. With this sensor information two bridges can be identified easily, while only the western one was expected according to the aerial image.

C. Computing Architecture

Main computing power is provided by a dual-CPU Intel quad-core Xeon L5420 system, with CPUs specially designed for low energy consumption.

Two hard real-time capable dSPACE computers run the low-level controllers and access the vehicle and camera platform I/O.

We use a Debian Linux operating system with a modified kernel for lower latency; hard real-time performance can't be guaranteed at present. Inter-process communication and data handling in the background is done by KogniMobil realtime-database (Kogmo-RTDB) [2]. All data is time-stamped. Kogmo-RTDB also provides data logging mechanisms.

III. Occupancy Grid Mapping

We use a $2\frac{1}{2}$-D ego-centered occupancy grid of dimension 200m×200m, each cell covering a small ground patch of 0.15m×0.15m. Each cell stores a single value expressing the degree of how occupied that cell is by an obstacle given the current scan only. For calculating the occupancy values, we first inertially correct the LIDAR scan, taking the vehicle's motion into account (exploiting IMU and odometric information). This is done by simultaneously moving the coordinate system of the vehicle while transforming the local LIDAR measurements to global 3D space. After a frame is completed, all points are transformed back into the last local coordinate system of the vehicle, simulating a scan as if all measurements were taken at a single point of time instead of the 0.1s time period of one LIDAR revolution.

Similar to Thrun *et.al.* [3], each cell's occupancy value c_{occ} is then calculated to be the maximum absolute difference in z-coordinates of all points falling into the respective grid cell, $c_{occ} = |c_{z_{max}} - c_{z_{min}}|$.

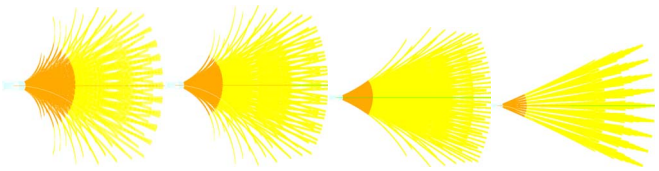


Fig. 3. Some of the tentacles we used at the C-ELROB 2009. Every image shows one set of tentacles to be selected according to robot velocity, with velocity and tentacle length increasing from left to right. Every set contains as much as 1000 tentacles, discretizing a subspace of curvatures, lateral offsets and headings.

IV. Driving With Tentacles

While quite complex approaches to mobile robot navigation exist (eg., solving the SLAM problem [4], methods based on trajectory planning [5]), our research was driven by the search for simplicity: *What is the simplest approach that lets a robot safely drive in an unknown environment?*

A. Basic Idea

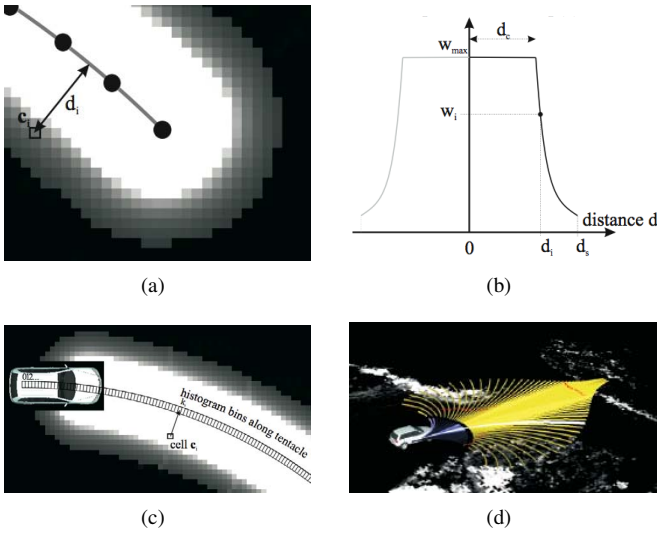
Our basic intention underlying the tentacles approach is to let our robot move within an unknown environment similarly to how a beetle would crawl around and uses its antennae to avoid obstacles. Of course, the idea of using antennae is not new: In fact, one of the first robots, "Shakey" [6], used "cat-whiskers" to sense the presence of a solid object within the braking distance of the vehicle. Braitenberg showed that complicated behavior can emerge by very simple mechanisms [7], and almost all of his vehicles (known under the term Braitenberg vehicles) use sensors resembling an insect's antennae. Some systems in mobile robotics integrate modules that are similar to our approach in that some sort of precalculated trajectories are verified to be drivable [8], [9], [10]. However, compared to the others our approach is closer to reactiveness in the sense of Braitenberg in that our "tentacles" are also used as perceptual primitives. A very detailed process of how those primitives are used to evaluate an occupancy grid is proposed in [11].

Indeed, in our approach the analogue to an insect's antennae are target trajectories representing the basic driving options of the vehicle given the current vehicle state. These driving options that we named "tentacles" then probe the occupancy grid to determine drivability and other properties of the corresponding target trajectory. The main loop of the approach is as simple as follows: In each cycle, first select a subset of tentacles according to the vehicle state. Then, by inspecting the occupancy grid, assign properties to each tentacle and select one of the tentacles as the target trajectory for the vehicle to drive.

B. The Details of a Tentacle

As said, a tentacle basically represents a target trajectory defined in the reference frame of the vehicle. These are represented as point samples p_i on arbitrarily shaped line sequences and denoted by $t_{lsq} = \{p_1, \cdots, p_N\}$. Fig. 3 shows some of the tentacle geometries we used at the C-ELROB 2009.

accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009



(a) (b) (c) (d)

Fig. 4. One tentacle trajectory and mapping of cells based on lateral distance (a), assignments of weights to the mapped cells according to their lateral distance (b) and mapping of cells to longitudinal histogram bins (c). One set of tentacles selected according to current vehicle state and the occupancy grid they relate to (d).

To make evaluation of tentacles based on the occupancy grid as efficient as possible, we make the assumption that both the geometric relation between the vehicle and tentacles, and also the one between the occupancy grid and the vehicle always remain static. If so, the memory offsets to all grid cells influencing the evaluation of a tentacle can be precalculated offline. We make use of this fact by defining, for each tentacle, two sets of grid cells. The first of these sets, t_{narrow}, contains cells within a lateral distance d_{narrow} to the tentacle trajectory t_{lsq}, $t_{narrow} = \{c_1, \cdots, c_M\}$. The second set t_{wide} contains all cells within a larger distance d_{wide}, resp., and additional cell weights according to the corresponding cell's lateral distance to the trajectory.

Finally, to associate each cell index with a longitudinal length on the tentacle trajectory, we longitudinally divide the trajectory into equally spaced bins and record for each cell the index of the bin it was mapped to. Thus, the information collected offline for every cell mapped to a tentacle can be summarized as $c_i = \left\{c_{i_{memindex}}, c_{i_{weight}}, c_{i_{bin}}\right\}$. This is illustrated in Fig. 4 for one tentacle.

C. Tentacle Evaluation and Selection

We will now give a brief description of the properties we evaluate for a tentacle and how a tentacle gets selected for execution based on these properties. The interested reader is referred to [11] for more details on this topic.

1) Drivability: The most important of all tentacle properties for sure is its drivability, $t_{drivable}$. Indeed, all other properties of a tentacle are only evaluated if the tentacle was considered drivable. For evaluating this property, we only have a look at the cells in t_{narrow} of a tentacle, which we now refer to as the tentacle's classification area. For all cells in this set, we increment the corresponding histogram bin if the cell itself is considered an obstacle, i.e. if the probability of it being occupied exceeds 0.5. We then move along the bins up to the crash distance, which is simply the distance the vehicle needs to stop. If any of the bins values exceeds a threshold $T_{obstacle}$, an obstacle is detected and the tentacle is considered undrivable. If no such bin exists within the crash distance, the tentacle is considered drivable.

2) Distance to Obstacles: This property is closely related to the drivability of a tentacle. However, of all drivable tentacles, we would prefer to select the one which can be driven for longer time, keeping the vehicle away from threatening obstacles. This is reflected by the property $t_{clearness}$, which is just the longitudinal length to the bin where an obstacle was detected (or infinity if there is no obstacle along the tentacle).

3) Flatness: Especially in offroad scenarios, a tentacle should be assigned a value reflecting the flatness of the ground it would lead the vehicle to. Assigning a flatness $t_{flatness}$ to a tentacle is a simple matter of computing the weighted sum of absolute z-coordinate differences of all cells within t_{wide}, making use of the weights $c_{i_{weight}}$ assigned to cells during tentacle construction.

With all drivable tentacles being assigned their properties, it now remains to select one of the drivable tentacles for execution. To be able to fine-tune the influence of individual properties, we decided to combine them all in a weighted sum, choosing the one tentacle minimizing this sum. As a prerequisite, we require all properties be normalized to uniform range. Again, we refer to [11] for details on how the properties are best normalized. Formally, selecting a tentacle is described by

$$t_{selected} = \underset{\{t|t_{drivable}=1\}}{\operatorname{argmin}} \; a_1(1-t_{clearness})+a_2 t_{flatness} \quad (1)$$

where the a_i denote the weights allowing fine-tuning the resulting behavior.

The benefit of precalculated tentacle structures gets apparent in online execution of the algorithm, where we are able to evaluate as much as 1000 different tentacles within the cycle time of the system of 0.1s. In the following of the paper, we will show how high-level knowledge and properties derived from other sensor modalities can be incorporated via the introduction of new summands into this last equation.

D. Incorporation of High-level Knowledge

From what has been told so far, the robot would most probably still not find its way from A to B without incorporation of some high-level knowledge and reasoning, such as deliberately planning a route. Luckily, there is hardly any robotic competition not providing any route information beforehand. This is true for both the DARPA challenges and the annual ELROB robotic competitions, albeit both represent extremes in terms of way point coverage. We will now show how this kind of information is integrated into the tentacles approach that is otherwise reactive by nature.

Suppose we are given a collection of GPS way points, that we connect in sequence to form a sequence of lines similar to the one describing a tentacle trajectory. Two opportunities

accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009

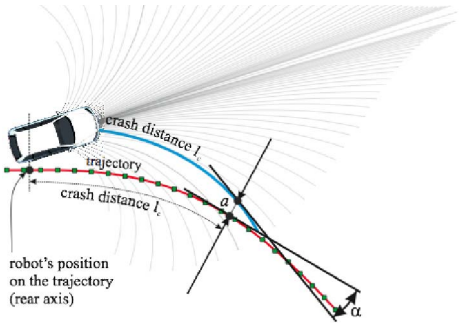
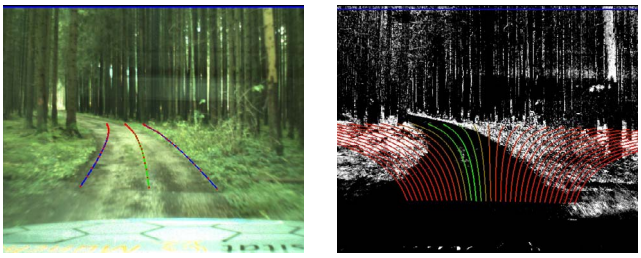


Fig. 5. Evaluating how well a tentacle (*blue*) corresponds to a human-provided target trajectory (*red*, with GPS way point samples shown *green*).



(a) Image of a single tentacle's skeleton line and support area.

(b) Rating a sparse tentacle set according to the introduced visual cue (red skeleton lines encode non-drivable tentacles).

Fig. 6. Visual tentacle evaluation in a forest scene.

immediately arise on how to consider this trajectory for tentacle selection. The first one would be to use the distance of a tentacle evaluated at some look-ahead length to this target trajectory as another tentacle property $t_{targetdistance}$. This would obviously mean selecting a tentacle leading the vehicle right onto track. However, this is not the desired behavior in most of the cases. It happens that buildings or dense vegetation prevent the GPS signals from reaching the sensors on board the vehicle at all. Even worse, GPS measurements might deteriorate due to signals being reflected. In any case, blindly following a given GPS target trajectory is no reasonable choice. This is especially true if two successive way points as given cover far distances and the resulting line could describe anything but the road, country track or street really in-between.

A more reasonable strategy would thus be to go into the same direction as the target trajectory without trying to actually approach it. The according property of a tentacle $t_{targetdirection}$ would then be calculated as the angular difference between the tangents at both trajectories at a given look-ahead distance and involves matching the vehicle onto the target trajectory.

In formulating an appropriate tentacle property, we want to be flexible in trading-off tentacles leading to the track and tentacles going into the same direction as the given track. This is again formulated by a weighted sum of both properties as

$$t_{target} = b_1 t_{targetdistance} + b_2 t_{targetdirection} \qquad (2)$$

contributing another summand to Eq. 1. In both cases, the look-ahead distance is taken to be the distance the vehicle needs to halt. The evaluation of both tentacle properties – leading to the path and following its direction – is illustrated in Fig. 5.

Note that by requiring tentacles to be drivable for selection, tentacles that lead well to the GPS target trajectory but are occupied by obstacles will not be executed.

V. Evaluating Tentacles By Visual Appearances

Up to now the road following capabilities of the tentacles approach without using dense GPS-Points are limited to scenarios with significant obstacles or non-flat surfaces along what humans would otherwise easily recognize as roads. In order to overcome these limitations we extended the tentacles approach to also incorporate visual features. There is a vast number of works on road detection and following algorithms. Basically, all these works try to identify the road according to visual features special to roads, like road boundaries [12], [13], texture and color [14]. The visual feature we propose goes the other way around: Instead of trying to recognize a road in an image, we try to segment non-drivable pixels in an image.

We segment non-drivable terrain utilizing the saturation channel of the HSI (hue saturation intensity) color space. The feature we propose makes use of the knowledge we gained – that the road surface may show all kinds of colors but there will be only a vanishing amount of brightly colored pixels on a road's surface. In the opposite, this means that almost all brightly colored areas will most likely correspond to non-drivable area (see Fig. 6(b)).

In order to rate each tentacle according to the visual feature, we do a perspective projection of the tentacles into the camera frame coordinates. For all coordinate transformations we use homogeneous transformation matrices which include the vehicle roll and pitch angles measured by an IMU as well as the gaze direction of the camera platform MarVEye7 derived from the steering angle of MuCAR-3 via the ackermann steering geometry and a simple pinhole camera model.

Subsequently, we sum all weighted saturation pixel values within the support area (see Fig. 6(a)) and normalize the sum by the number of considered pixels N_t to get the visual quality of each tentacle (see Fig. 6(b)).

Like in Subsec. IV-D we incorporate the visual quality t_{visual} of a tentacle into the tentacle evaluation via a weighted sum

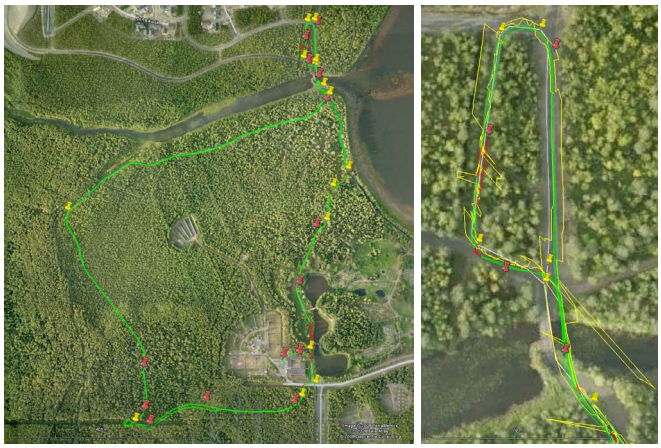
$$t_{visual} = c_1 t_{saturation} \qquad (3)$$

contributing another summand to Eq. 1.

VI. Incorporation of GPS and Map Knowledge

Problems with GPS position measurements in a forest environment were mentioned above and can be seen in Fig. 1 – during the ELROB events of 2007 and 2008 we learnt the lesson "*Never Trust GPS*". In this section we describe the extensions for reduced GPS influence in our appoach.

accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009



(a) (b)

Fig. 7. Recorded track during the trial. Green: autonomous driven parts. Red: manual intervention in critical parts. Yellow: Raw GPS. For comparison: White: road network. Red markers: given waypoints. Yellow markers: network-matched waypoints, augmented with attraction radii.

(a) Northern part of the trial: the eastern bridge is not included in the aerial image.

(b) Southern part of the trial: all roads from the traffic circle to the west are only available in the GIS data.

Fig. 8. Road network from commercial GIS data, matched with the aerial image. Orange lines: lanes. Blue dots: crossings.

A. Weighting GPS Position Information

Because of the known problems with GPS measurements, we turned away from using GPS information directly.

An *Egomotion Estimation* process collects all available data which is provided by the several sensors and uses the time-stamped measurements for improvement of the egomotion estimation. Currently we include measurements from the INS, vehicle's odometry and air suspension. Estimation itself is done by Extended Kalman Filter techniques, the state vector consists of 23 states. Different system models are used for prediction in the substates, e.g. a bicycle model for lateral motion. Among other substates, the vehicle's position and orientation are estimated both in a fixed global (GPS-) coordinate system (*EGOPOS*) and in a drift space (*DRIFTPOS*).

For all raw sensor data processing, like inertial correction of LIDAR scans, we utilize the drift space coordinate system. Thus GPS position errors don't influence local behavior of the robot, and local movements are represented much more accurate even over a longer period.

But in autonomous navigation, there are decisions to make where you need absolute position information like GPS: where to go on a free area or where to drive if there are divergent possibilities like at crossings?

In our reactive approach, we do not make such decisions, and the robot's behavior is entirely determined by the weighting of the different properties $t_{clearness}$, $t_{flatness}$, t_{visual}, t_{target} of all drivable tentacles. This weighting has been determined empirically during the testing periode. Setting $t_{flatness}$ and t_{visual} equal proved because one sensor can't outvote the other completely this way. But we figured out that one constant parametrization won't work in all situations. Thus we actually use two different sets of weights.

For the first, our approach weights the *EGOPOS* only marginally during normal driving, so that the reactive mechanisms outvote GPS information if there are differently classified tentacles. In doing so, tentacle properties $t_{clearness}$, $t_{flatness}$, t_{visual} are weighted by 1.0 whereas the weight for t_{target} utilizing *EGOPOS* is reduced to 0.2. Thus, tentacle selection is only marginally influenced by the given GPS target trajectory as shown in Subsec. IV-D. The robot will go on its way while avoiding obstacles.

While heading forward this way is the right decision most of the time, when the robot comes to places requiring a decision about the way to take, e.g. at crossing, raising the influence of the GPS target trajectory is helpful. Within an attraction radius of 20 m around the estimated crossing position the weight of target trajectory t_{target} is raised to 1.0 to equal the weights of the other tentacle properties. Thus, tentacles leading into the same direction as the target track are selected with higher probability, so that at crossings the right branch is taken. Still, the underlying obstacle avoidance guarantees safety even with this comparatively intense GPS usage.

Note that if the vehicle is mapped into the inside of a waypoint's attraction radius, the robot's speed is reduced to achieve a better maneuverability.

B. Track Generation

A priori only a couple of sparse GPS waypoints are known, which are located always behind a crossing (red markers in Fig. 7). They can be used to turn into the right direction. Challenging is the fact, that there often are hundreds of meters between the given waypoints, so that the connecting air line has a complete different course than the real path, also additional crossings can be included.

Adding knowledge derived from a map or a geographical information system (GIS) can improve detail and shape of the resulting track significantly.

By using different layers of "Shapefile" data, a data format established in geodesy and commercially available for most regions, we generate a lane network for the corresponding region. Shapefile layers not only describe roads, but also small forest trails or additional information of the environment like lakes. For an excerpt of the utilized lane network see Fig. 8. Each road is split into two lanes (orange lines in Fig. 8)

accepted for publication at
3rd Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV),
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009

for handling different directions in planning, the lanes are connected among each other at the crossings (blue dots in same figure).

The given GPS waypoints are matched onto the lane network automatically. Then the shortest route is planned covering all waypoints as matched to the road network. In case the air line distance between to given waypoints is much smaller than the planned route, we use the direct connection between those pairs of points. Crossing points from road network are endued with an attraction radius just like the given waypoints are. Afterwards, a manual check of the planned track is performed making use of an underlying aerial image. This is advisable to check for variations of lane network and aerial image, and to modify the track if necessary.

A generated track can be found in Fig. 7(b). The white line describes the utilized lane segments, yellow markers on the track represent matched waypoints that are augmented with an attraction radius.

VII. Results and Conclusions

The approach described in the sections before has proved itself in C-ELROB taking place in Oulu/Finland in June 2009.

We took part in the scenario called "Autonomous Navigation". It consisted of a round track with a lenght of approx. 2.6 km, which had to be passed twice, totalling 5.2 km. For judging performance of the participating robots, the most important aspect was its grade of autonomy, but also time required (limited to 1 h) and distance travelled.

The course was described by a total of 34 waypoints given in UTM coordinates, consisting of paved and unpaved, unmarked roads, gravel, grass paths and trails. Wide parts led across the forest, so we had to face changing lighting conditions and frequent GPS outages. Because inspection of the course was disallowed before the trial, and the provided aerial maps (see Fig. 7(a)) were produced before some roads and tracks had been constructed, a couple of waypoints seemed to be located in the wood by visual comparison with the aerial images.

Since the trials took place in public area, there were dynamic obstacles like vehicles, bicycles or pedestrians our vehicle had to cope with.

The given UTM waypoints (red markers in Fig. 7) were matched onto an road network from commercially available GIS data (white line). By this, some waypoints were moved, especially to describe the crossings in a better way (yellow markers). Also information about the formerly unknown parts of the track could be gathered (see Fig. 8). The matched points were postprocessed manually, where the influence of the given UTM point south of the new eastern bridge was removed because this bridge could not be seen on the aerial images, whereas the road network showed two bridges due to GIS data.

MuCAR-3 managed the difficult challenge successfully. Our robot was the only vehicle completing the requested distance of 5.2 km, undercutting the time limit by far with

(a) Turning into the grass trail. (b) Taking the new eastern bridge, detected by sensor information only.

Fig. 9. MuCAR-3 during the C-ELROB 2009 trial.

Fig. 10. A narrow grasstrail, classified as too narrow for driving.

43 min, including 2.5 min of data analysis after finishing. 81 % of time were driven fully autonomously, which equates to 95 % of distance. Fig. 7(a) shows the recorded track, with parts of the track which required manual intervention colored red.

The critical parts of the track mostly consisted of parts being narrower than the car (Fig. 10). The security mechanisms stopped the vehicle because up to now there is no differentation possible between mild obstacles like high-grown grass and hard obstacles like branches of a tree. This parts are mainly located in the northest and the south-east part. Also at few challenging turnoffs intervention was necessary (Fig. 9(a), see also right-corner in middle-left part of Fig. 7(b)), because turning from asphalt into high grass was inhibited by visual rating of the tentacles. The simple reason for this is that the branch to take did never get into the field of view of the cameras.

Since most of the track lead through the woods, there are non-negligible errors in GPS position, as the yellow line in Fig. 7(b) shows. These errors especially change when leaving the forest and getting information of more satellites. We measured position errors up to 50 m, which were fully compensated by our egomotion estimation process fusing information from INS with odometry and other information.

Summarizing, it has proved in the trials that less confidence in GPS measurements is advisable especially under poor GPS conditions like in the described forest environment. Fusing environment perception of LIDAR and vision in the tentacle approach enables the robot to drive safely with less need for accurate GPS information.

Some explanatory video impressions of the trial and its difficulties can be found on our website [15].

During the trials some topics for future work appeared. The first is determination of "optimal" weighting factors for tentacle properties in all situations. We currently use two

fixed sets of weighting factors which have shown to fail in some situations. Dynamic adaption of weights – using e.g. learning based methods – could improve our approach a lot. For example, one could select the tentacle closest to the path taken by a human driver in each step. If this tentacle's visual property t_{visual} was significantly larger than its flatness $t_{flatness}$, the weight for t_{visual} would be increased and the one for $t_{flatness}$ decreased accordingly.

Another topic is the precision of tentacle navigation. In this approach to offroad navigation a new driving primitive is selected every cycle. There will always be a control deviation, increasing with dynamically changing driving primitives. This will cause problems in narrow environments where high precision control is needed. Using a traditional planning approach, generating a safe trajectory at a greater look-ahead distance as target track for the lateral controller, could improve the quality of control considerably. To keep the independency from GPS, this had to be done within the DRIFTPOS coordinate system of course.

VIII. Acknowledgements

The authors gratefully acknowledge funding by DFG excellence initiative research cluster *Cognition for Technical Systems* – CoTeSys [16], partial support of this work by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) within the Transregional Collaborative Research Centre 28 *Cognitive Automobiles* [17], as well as generous funding by the Federal Office of Defense Technology and Procurement (BWB).

References

[1] ELROB – The European Robot Trial. [Online]. Available: http://www.elrob.org

[2] M. Goebl and G. Färber, "A Real-Time-capable Hard- and Software Architecture for Joint Image and Knowledge Processing in Cognitive Automobiles," in *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, June 13-15 2007.

[3] S. Thrun, M. Montemerlo, and A. Aron, "Probabilistic Terrain Analysis For High-Speed Desert Driving," in *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[4] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, June 2001.

[5] N. Sariff and N. Buniyamin, "An Overview of Autonomous Mobile Robot Path Planning Algorithms," in *4th Student Conference on Research and Development, 2006. SCOReD 2006.*, 27-28 June 2006, pp. 183–188.

[6] N. J. Nilsson, "Shakey The Robot," AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Tech. Rep. 323, Apr 1984.

[7] V. Braitenberg, *Vehicles: Experiments in Synthetic Psychology*. MIT Press, 1984.

[8] A. Kelly and A. T. Stentz, "Rough terrain Autonomous Mobility - Part 2: An Active Vision, Predictive Control Approach," *Autonomous Robots*, vol. 5, pp. 163 – 198, May 1998.

[9] P. Lamon, S. Kolski, R. Triebel, R. Siegwart, and W. Burgard, "The SmartTer for ELROB 2006 a Vehicle for Fully Autonomous Navigation and Mapping in Outdoor Environments," Autonomous Systems Lab, Ecole polytechnique Fdrale de Lausanne, Switzerland, Autonomous Intelligent Systems, Albert-Ludwigs-University of Freiburg, Germany, Tech. Rep., 2006.

[10] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the DARPA Grand Challenge: Research Articles," *Journal of Robotic Systems*, vol. 23, no. 9, pp. 661–692, 2006.

[11] F. von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H.-J. Wuensche, "Driving with Tentacles: Integral Structures for Sensing and Motion," *International Journal of Field Robotics Research*, 2008.

[12] E. D. Dickmanns and B. D. Mysliwetz, "Recursive 3-D Road and Relative Ego-State Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 199–213, February 1992.

[13] B. Southall and C. Taylor, "Stochastic road shape estimation," *Proceedings of 8th IEEE International Conference on Computer Vision*, vol. 1, pp. 205–212, 2001.

[14] U. Franke, H. Loose, and C. Knöppel, "Lane Recognition on Country Roads," in *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium*, June 2007, pp. 99–104.

[15] Autonomous Systems Technology (TAS), University of the Bundeswehr Munich. [Online]. Available: http://www.unibw.de/lrt8/medien-en

[16] Cognition for Technical Systems – CoTeSys. [Online]. Available: http://www.cotesys.org

[17] Transregional Collaborative Research Center 28 – Cognitive Automobiles. [Online]. Available: http://www.kognimobil.org