Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

17$^{\text{th}}$ International Conference on Process Control 2009

Hotel Baník, Štrbské Pleso, Slovakia, June 9 – 12, 2009

ISBN 978-80-227-3081-5

http://www.kirp.chtf.stuba.sk/pc09

Editors: M. Fikar and M. Kvasnica

Warias, R., Gerke, M.: Target-Oriented Fuzzy-Collision-Avoidance for Vehicles, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 17th International Conference on Process Control '09*, Štrbské Pleso, Slovakia, 638–646, 2009.

Full paper online: http://www.kirp.chtf.stuba.sk/pc09/data/abstracts/063.html

TARGET-ORIENTED FUZZY-COLLISION-AVOIDANCE FOR VEHICLES

R. Warias and M. Gerke

University of Hagen
Department of Electrical Engineering
Control Systems Engineering group
Universitätsstraße 27, D-58084 Hagen, Germany
fax : +49 (2331) 987 – 354

e-mail : Reinhold.Warias@t-online.de
e-mail : Michael.Gerke@FernUni-Hagen.de

Autonomous mobile systems (AMS) mostly navigate on paths created by global path planners. But due to often appearing unforeseen objects the AMS needs a local collision avoidance (LCA) which takes over control functions and guides the AMS around obstacles. Mostly this maneuver results in leaving the preplanned path. Unfortunately it is possible that the AMS cannot find back to the precomputed path because the obstacles' arrangement is too complex. Therefore it is useful to equip the LCA with a target-oriented component during collision avoidance in order to guide the AMS back to the precomputed path. Our approach realizes a fuzzy-based LCA using direct sensor information to generate a steering angle. The results of our examinations show, that the AMS successfully reaches the target-point in most of the cases; without any global path planning, only with the means of fuzzy logic and a target-oriented component.

Keywords: autonomous mobile systems, fuzzy logic, collision avoidance, sensor-based navigation, target orientation

1 INTRODUCTION

In the last years autonomous mobile systems (AMS) got the ability to navigate within partly or completely unknown environments with the help of self localisation, world modelling, path planning and collision avoidance. Unforeseen, perhaps dynamically occuring objects must not endanger the AMS nor abort the complete mission. At this point the local collision avoidance (LCA) takes over AMS-control-functions for maneuvering.

After detecting an obstacle with external sensors the LCA initiates an evasion maneuver to guide the AMS around the object. This maneuver makes the AMS leave the precomputed path. Unfortunately it is possible that the AMS cannot find back to the given path because the obstacles' arrangement is too complex. Therefore it is very useful to equip the LCA with a target-oriented component, which leads the AMS back to the predefined direction.

Our approach realizes a fuzzy-based LCA using direct sensor information to generate a steering angle. Additionally the target-oriented component considers the AMS' present orientation with respect to the direction of the target-point while computing a drift angle. Both, steering angle and drift angle are superposed; the result is the final steering angle of the AMS. The influence of the drift angle is modified by a weight, which depends on the distance to the nearest obstacle: the smaller the distance the lower the weight.

We examined the LCA in a three-step-analysis:

1. configuration of the fuzzy-controller
    - number of memberships
    - superposition of memberships
      ($\Sigma\mu = 1$ or $\Sigma\mu > 1$)

2. type of object-avoidance strategy
   - minimum obstacle distance

   in the first run without, in the second run with automated modification of the rule-set

3. return-strategy
   (finding back to the preplanned path)
   - drift-vector controller

In this paper we point out the strategies; the test runs and their results are documented and discussed in detail in (Warias 2004).

To examine our approach, we developed a windows-controlled simulation software called "AmsSim, Version 1.0" with the help of Microsoft's "Visual C++"-compiler. The software offers lots of possibilities to modify the parameters of fuzzy-controller, environment and sensors. We also created Version 1.1, which uses English instead of German expressions.

The simulation tool and its possibilities to modify the parameters are described in section 2. The fuzzy-controller and its parameters are introduced in section 3 as well as the method of defuzzification which we used in our approach. Section 4 introduces he different strategies of local collision avoidance and explains how they work. In section 5 we have a closed look on the experiments with these strategies and introduce their results in detail. Section 6 gives a short summary of the test results and in section 7 we have a glance on our future plans of creating a simulation environment as well as examining path planning in connection with target-oriented collision avoidance.

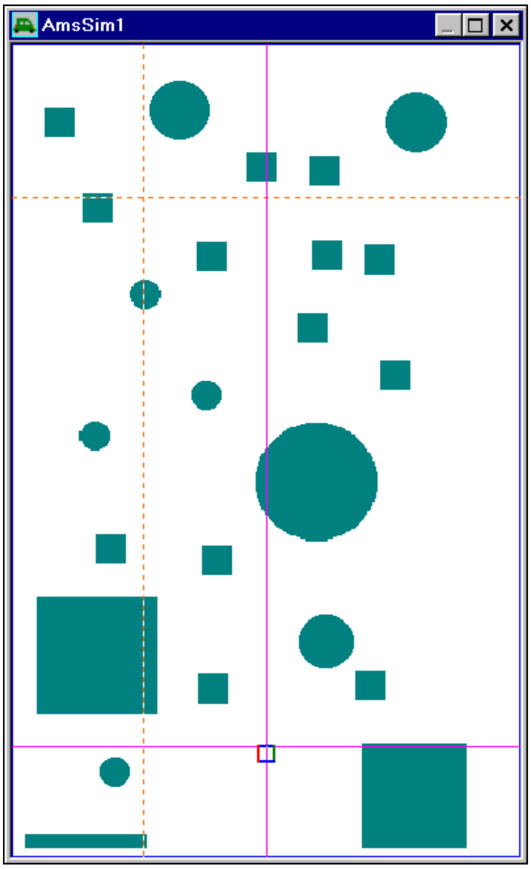

Figure 1: simple environment (Map 1)

2 MODELLING THE SIMULATION COMPONENTS

2.1 *Environment*

The map is the model of the AMS' environment. It is realized as a bitmap which contains the complete arrangement of the obstacles. The AMS does not know this map; it moves in completely unknown environments.

To examine AMS' behaviour in different obstacle arrangements we implemented four maps with different levels of difficulty. Map 1 (Figure 1) contains a simple arrangement of circles and squares.

Map 2 (Figure 2) is endowed with obstacles of different levels of difficulties. There are two regions in the upper area which can only be reached by passing very narrow gaps. Additionally there are a maze, a curved corridor of homogeneous width as well as a short corridor of decreasing width.

With the intention to model a natural-like environment, we arbitrarierly spreaded lots of obstacles with irregular shapes in map 3 (Figure 4).

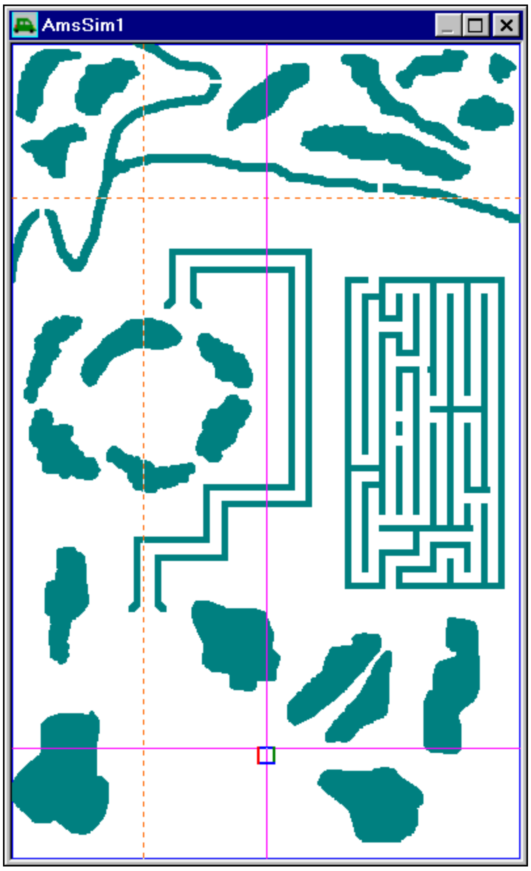

Figure 2: more pretentious environment (Map 2)

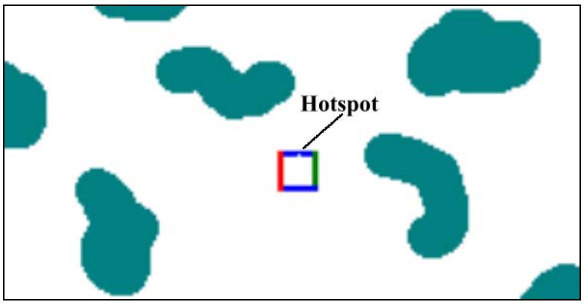

Figure 3: autonomous mobile system

Map 4 (Figure 5) realizes a very pretentious environment. It contains mazes, corridors and a spiral; it is an artifical environment.

Due to a clearer observance and better evaluation of AMS' behaviour we faded not only the AMS itself, but also the crosshairs of world coordinates and target-point as well as the drift vector which connects AMS and the target-point into the map. The user is enabled to set the coordinates of the target-point's crosshairs and of the world's origin with an Environment-Editor.

Additionally every component, with the exception of the AMS, can be faded in or faded out of the map.

2.2 *Autonomous mobile system*

The AMS is reduced to a single dot with the size of one pixel. The dot has no mass at all and no kinematical properties.

Nevertheless the AMS is a non-holonomic component because it cannot move directly in right or left direction but has to drive through a curve. In order to force this behaviour we limited the steering angle to $\pm 45°$. Additionally the AMS cannot go backwards.

Because of a better visibility the AMS is a square with an hotspot (Figure 3) at the bow.

The orientation is illustrated by the green starboard and the red portside. The square is only due to visual reasons. It cannot cause any collision. Collisions are caused only by the hotspot itself.

2.3 *Control desk of the mobile system*

The AMS is controlled via a control desk (Figure 6), which also gives information about the state of the mobile system.

It enables the user to drive the AMS in manual or automatic operation as well as in single-step operation or running it to definable breakpoints. Additionally it delivers state-information such as orientation and drift angle. Furthermore it shows the present step-number, the number of collisions, the step-number of the breakpoint (HS) and the number of steps (RS) from the present position up to the breakpoint.

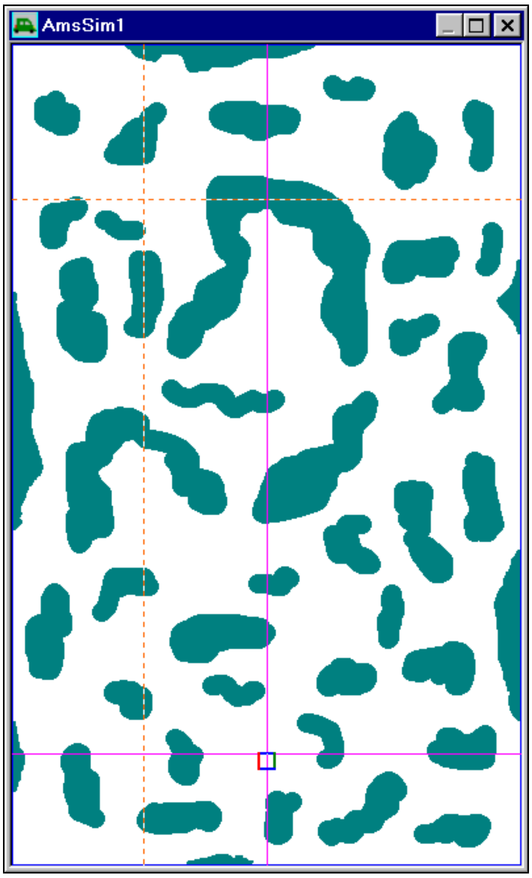

Figure 4: natural-like environment (Map 3)

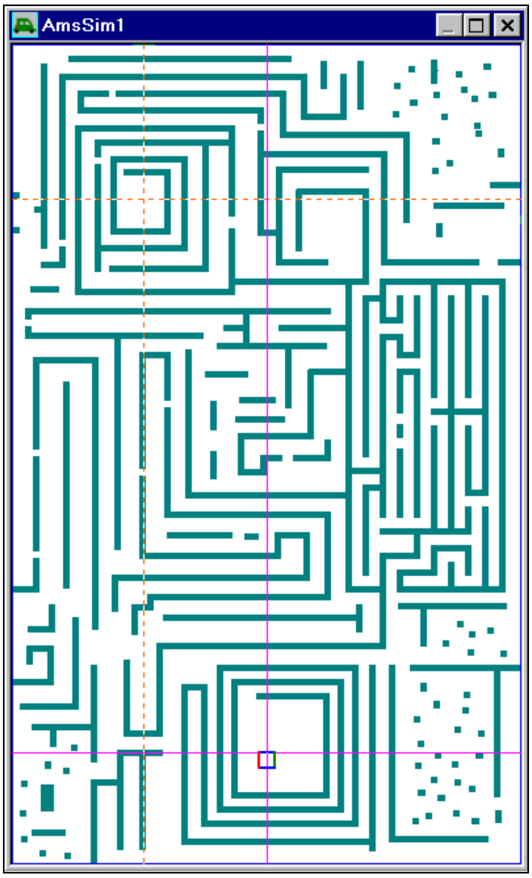

Figure 5: artifical environment (Map 4)

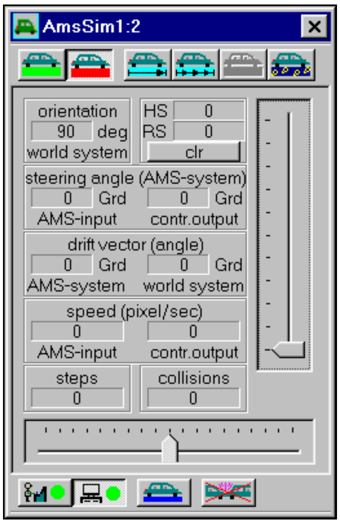

Figure 6: AMS-control-desk

2.4 *AMS-View*

The AMS is assumed to be equipped with an on-board sensor-system for navigation. The AMS-View (Figure 8) shows the environment in the style the mobile system recognizes it.

In our software we simulated a laser scanner with variable range of horizontal view. Figure 8 shows the AMS moving in map 2 (refer to Figure 2) and seeing the entrance of the corridor on the right side. The AMS moves straight towards another obstacle.

The upper buttons of the window enable the user to choose the scanner, which data are displayed in the window.

2.5 *Sensors*

The simulation model of the sensors consists of laser scanners with a range of 190° which is divided up in 528 sectors, 0.36°each. In our approach we solely used sector 15 to sector 514. The distance range can be modified in the interval $l \in [\, 1, 500 \,]$ units of length.

Currently we solely use the bow-scanner. Stern-, starboard- and portside-scanner are also implemented and deliver information which can be represented in the AMS-View (refer to Figure 8). Nevertheless they

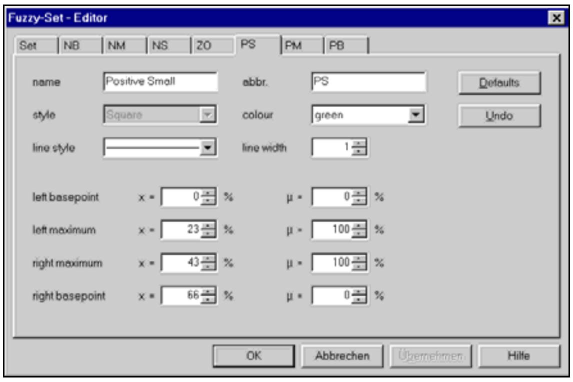

Figure 7: Properties of a membership

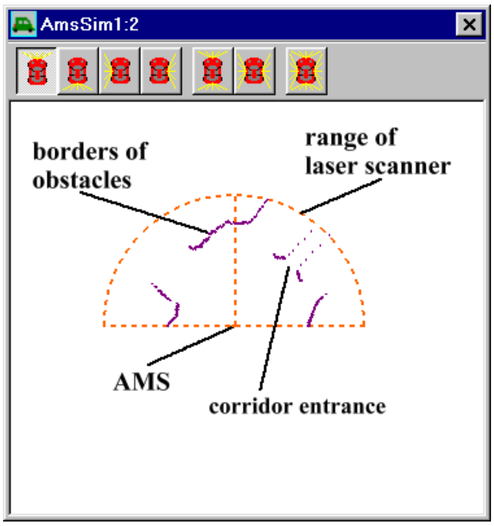

Figure 8: AMS-View

are not considered by the fuzzy-controller yet.

3 FUZZY-CONTROLLER

3.1 *Modelling the controller*

The fuzzy-controller consists of the fuzzy-sets with its memberships, the rule-set, the inference machine and the defuzzification.

The Fuzzy-Set-Editor enables the user to modify the memberships (Figure 7). In our software we consider only trapezoidal memberships including their extremal shapes "triangle" and "square".

Additionally the user is enabled to open a window according to Figure 9 during a trial run.

It shows the complete fuzzy-set of the chosen variable including its present crisp value as a vertical line. All values of the memberships are also displayed in percentage of their maximum values in the lower region of the window. Its upper right area shows the crisp value in both, absolute and normalized numbers.

The Inference/Defuzzy-Editor offers the possibility to modify the methods of the inference-machine.

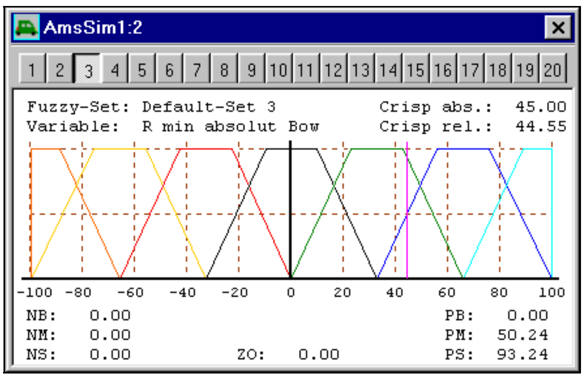

Figure 9: Fuzzy-information at AMS-runtime

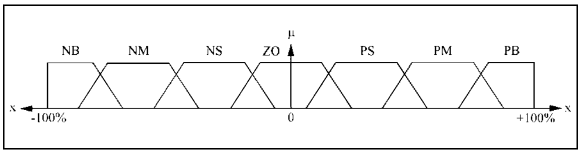

Figure 10: Fuzzy-set with seven memberships

Each strategy of collision avoidance consists of a default rule-set which can be modified by an integrated rule-editor.

3.2 *Memberships*

The input values of the sensors are normalized: the angle to ±π/2, the obstacle distance to the sensor distance range r_{max}.

Both, fuzzy-sets and rule-sets cope partly with seven, partly with five memberships per set.

A set with seven memberships is defined according to Figure 10 with the following abbreviations:

PB	positive big	NS	negative small
PM	positive medium	NM	negative medium
PS	positive small	NB	negative big
ZO	zero		

We also realized a set with five memberships.

The memberships of the controller's variables can be modified by the Fuzzy-Set-Editor (refer to Figure 7). Rule-sets, inference-machine and defuzzification presume either $\mu = 0$ or $\mu = 1$; the usage of intermediate values as well as the input of decimals may be realised later. The present Fuzzy-Set-Editor rounds $\mu \leq 49\%$ to $\mu = 0$ and $\mu \geq 50\%$ to $\mu = 1$.

3.3 *Inference-machine*

The inference-machine firstly works with an operator of aggregation, which realizes a superposition of two memberships. With μ_A and μ_B as input-degrees of fulfillment it computes the output-degree of fulfillment μ_O. The inference-machine is preadjusted to the minimum-operator

$$\mu_O = min\{\mu_A, \mu_B\} \tag{1}$$

and computes the degree of fulfillment for each rule in the rule-set. This degree is combined with the associated membership of the output variable's fuzzy-set. This in turn results in the four specific points of an output-membership.

The Inference/Defuzzy-Editor enables the user to choose different inference methods as well as different operators of aggregation which are decribed in (Tilly 1991).

Secondly the inference-machine has to compute the fuzzy-output with an operator of accumulation. To avoid lots of time-wasting calculations, we used a method, which is similar to the maximum-operator of accumulation and is integrated into the defuzzification.

3.4 *Defuzzification*

Each rule of the inference machine results in an output membership. Figure 11 shows an example with three rules.

The maximum-operator results in the diagram in the "MAX"-line of Figure 11. In our approach we compute the area A_ν and the according center of gravity's x-coordinate x_ν with $\nu = 1 \ldots n$ of each rule-output. Out of these data the defuzzification computes the crisp-value with the formula

$$x_c = \frac{\sum_{\nu=1}^{n} x_\nu A_\nu}{\sum_{\nu=1}^{n} x_\nu} \tag{2}$$

This method has the advantage of both, simple implementation and very low computation time. It has, of course, the characteristic to consider overlapping areas (refer to Figure 11) more than one time.

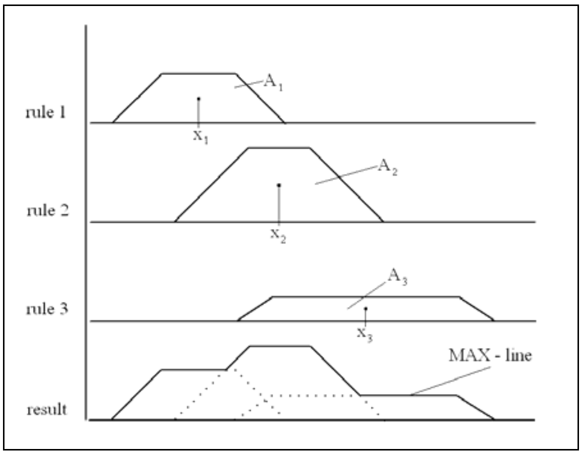

Figure 11: Example-output of three rules

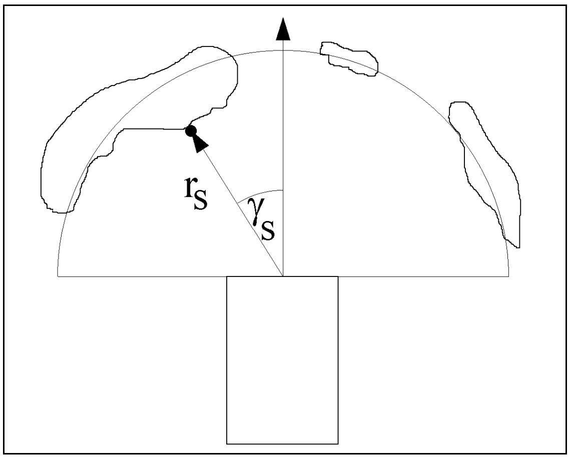

Figure 12: Minimum distance

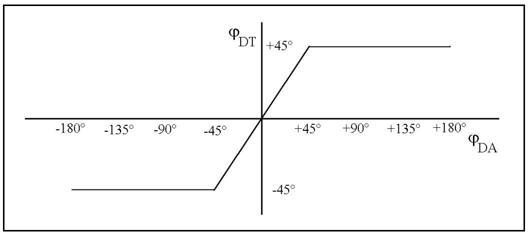

Figure 13: Limitation of the drift angle

4 STRATEGIES OF COLLISION AVOIDANCE

4.1 *Strategy of minimum distance*

The strategy of calculating the minimum distance (Figure 12) evaluates the position of the point which is next to an obstacle.

Out of all obstacle-distances in the range of $-90° \le \gamma_S \le +90°$ the strategy chooses that point, which is next to the AMS. This point is uniquely fixed with its value r_S and its angle γ_S in sensor-coordinates. The data r_S and γ_S are fuzzificated and fed to the controller.

4.2 *Target-point oriented strategy*

With this strategy we want the AMS to reach a target-point without any knowledge about its environment. To get the direction to the target-point, we define a drift vector $\underline{r}_{DW}$, which is computed with the help of the target vector $\underline{r}_Z$ and the position vector $\underline{r}_W$ (Figure 14).

The target vector is fixed and begins at the start-position of the AMS and ends at the target-point. The position vector connects the start-position and the current AMS-hotspot. The difference of these vectors, according to (3), is the defined drift vector, which begins at the AMS' current hotspot and ends at the target-point.

$$\underline{r}_{DW} = \underline{r}_Z - \underline{r}_W \tag{3}$$

These vectors are described in world coordinates.

In our approach the fuzzy controller determines the steering angle Θ_C, based on AMS-coordinates. The AMS knows both, the target-point and its own

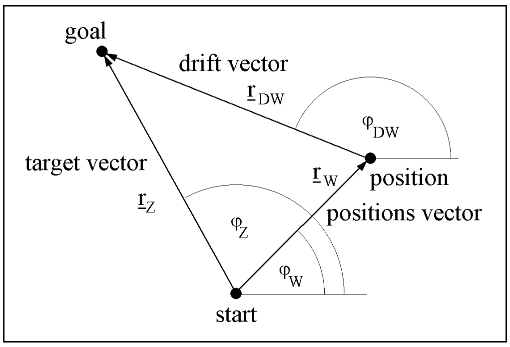

Figure 14: Definition of the drift vector $\underline{r}_{DW}$

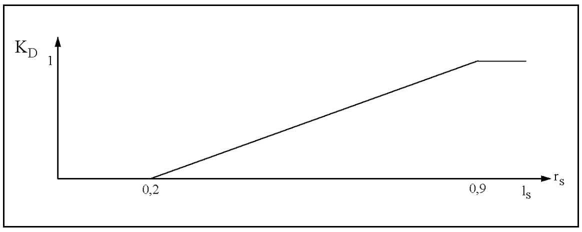

Figure 15: K_D versus obstacle-distance

position as well as its orientation Ψ. Therefore it is able to compute the direction of the drift vector, the drift angle φ_{DW} (Figure 16), which is based on world coordinates, too.

Then the drift angle φ_{DW} is transformed into AMS-coordinates with (4).

$$\varphi_{DA} = \varphi_{DW} - \Psi \tag{4}$$

The value $|\underline{r}_{DW}|$ of the drift vector is not of relevance.

Figure 16 shows that φ_{DA} can be of any value of the interval $\varphi_{DA} \in [\,-180°, +180°\,]$, while the AMS' steering angle Θ is limited to the interval $\Theta \in [\,-45°, +45°\,]$. Therefore we also limited the drift angle to ±45° according to Figure 13.

So the resulting steering angle Θ of the AMS is computed by the controller's output Θ_C and the transformed drift angle φ_{DT} according to (5).

$$\Theta = \Theta_C + K_D * \varphi_{DT} \tag{5}$$

In (5) we introduced the weight $K_D \in [\,0.0, 1.0\,]$ in order to vary the influence of the drift angle. So the final value of K_D is fixed at AMS-runtime.

If there are no obstacles at all the AMS goes straight to the target-point with $K_D = 1$. If the AMS moves very close to an obstacle, it does not consider the drift angle with $K_D = 0$. These facts show, that the obstacle distance r_S is a suitable criterion to calculate the weight K_D. Figure 15 shows the curve of K_D versus the obstacle distance r_S, which is normalized by the sensor distance range l_S of our approach.

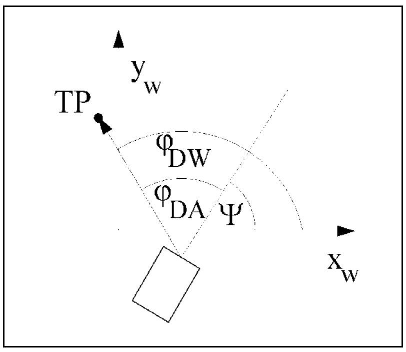

Figure 16: Drift angle of the AMS

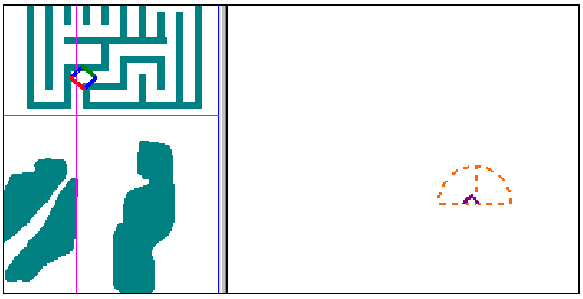

Figure 17: Left turn in a right bend

5 EXPERIMENTS AND RESULTS

5.1 *Analysis of fuzzy-sets and memberships*

The examination of the fuzzy-sets in our first experiments showed, that a fuzzy-controller with seven memberships per variable does not perform satisfactory. Because of the steep ramps, a small change of input values results in a big change of the output value.

Overlapping memberships with $\Sigma\mu > 1$ work disadvantageously, too. In this case upcoming non-linearities seem to initialize big changes of the output variable.

It turned out, that such type of fuzzy-controller was not an appropriate solution at large distance ranges of the sensors.

In principle the results could be improved by reducing the sensor's distance range r_{max}. Because of the normalization of the obstacle distance to the sensor distance range, a reduction of r_{max} equals a reduction of the obstacle density.

5.2 *Strategy of minimum distance*

The origin of the world coordinate system was positioned in front of the maze entrance of map 2 (refer to Figure 2). With a controller design of static rule-set, low sensor distance range and seven linearized memberships per set, the AMS entered the

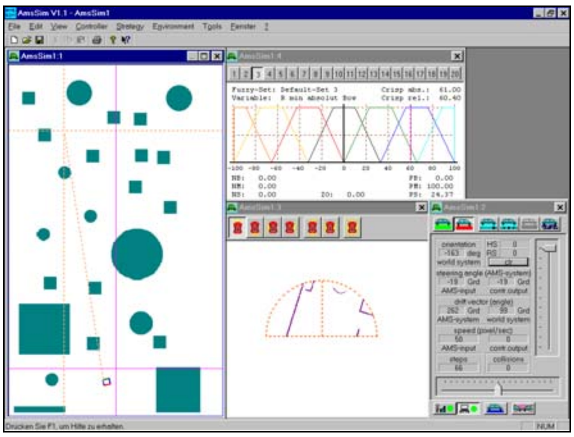

Figure 18: Main screen of our application

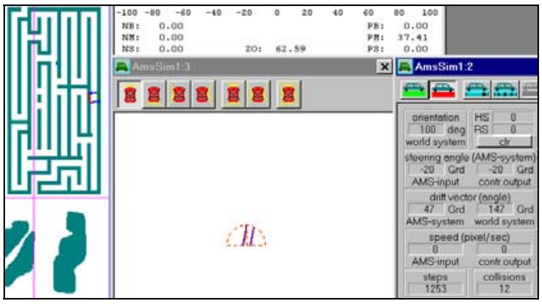

Figure 19: Coping with the maze

maze, but did not recognize the bend (Figure 17). Instead of turning to the right, it turned to the left (here at step 33). Then it left the maze through the entrance. During a mission of 68 steps there were 2 collisions which occured at step 33 and step 34.

The fuzzy-controller with five memberships per set and a dynamical rule-set lead the AMS through the entrance, caused some collisions, but moved, under strong oscillations, freely within the maze. According to Figure 19 the corridor's dead end was recognized and after describing two circles and causing one collision the AMS turned and went upwards.

The mission consisted of 2030 steps and caused 17 collisions. The change to larger laser distance ranges resulted in a mission of 2950 steps with 42 collisions.

With this type of controller the AMS moved within both, the corridor and the maze of map 2, even with larger distance ranges of the laser sensor. Therefore the fuzzy-controller with five memberships per set and dynamical rule-set is the optimal solution for the strategy of minimum distance. Occuring oscillations and collisions in an acute-angled corridor are characteristics of this strategy.

5.3 *Target-point oriented strategy*

The configuration of the fuzzy-controller consisted of five memberships per set and a dynamical rule-set. The sensor was adjusted to large distance range.

After starting in map 1 (refer to Figure 1) the controller led the AMS at the first three steps away

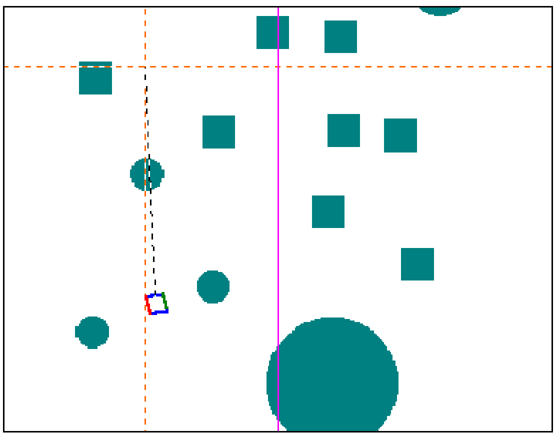

Figure 20: AMS controlled by the drift vector

Figure 21: Approach to go between square and circle

from the left small square in order to prevent a collision. In this case the controlling solely lay in the hands of the LCA.

After 17 steps the drift vector got more and more influence and and guided the AMS to the left until it drove through the passage between the left small square and the right big circle according to Figure 21. The drift vector pulled the AMS to the target-point like a "rubber band".

The small circle is located in the path; therefore the AMS drove a roundabout way and finally moved straight to the target-point (Figure 23).

The mobile system showed target-oriented behaviour and moved without any collision.

With a reduction to low sensor distance range the mobile system took immediately course to the target-point, controlled by the drift vector. According to Figure 22, it evaded the small circle and reached the goal (Figure 24).

In the environment of map 2 the AMS was pulled to the corridor (Figure 25).

But with our strategy the mobile system turned to the right, detected the passage between corridor and maze and used it according to Figure 26.

It reached the target-point after approximately 600 steps.

At low sensor distance range this strategy fulfilled all expectations. The mobile system set course to the target-point. It evaded the obstacles and eventually reached the goal. At medium and high sensor distance range difficulties occured in some cases. Here the distance to an obstacle may be so small that its crisp value is too close to zero and only uses the ZO-membership. The modification of the fuzzy-set

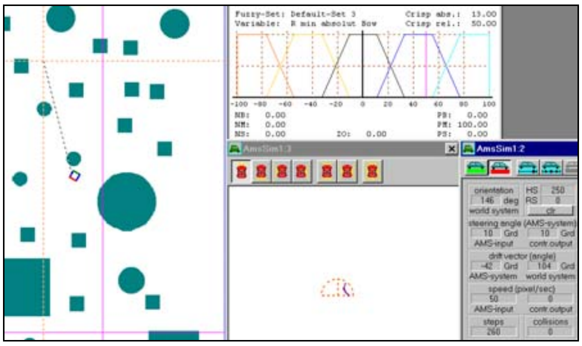

Figure 22: Evading the small circle

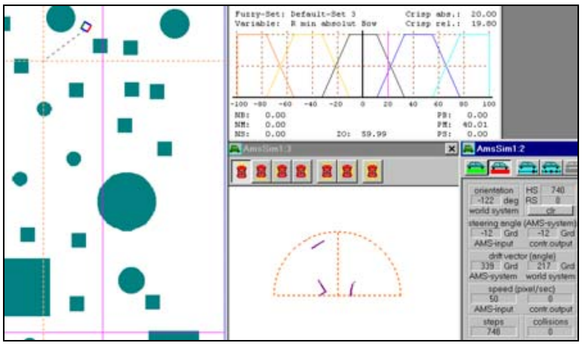

Figure 23: Course to the target-point

may probably avoid these problems.

The strategy of minimum distance with dynamical rule-set and dynamical superposition of the drift angle is able to guide a mobile system safely through an environment to a target-point.

Occuring insufficiencies are a problem of optimisation, which potentially could be solved by modifying the curve of the weight K_D (refer to Figure 15). Another approach could use not only the obstacle distance r_S but also the angle γ_S to compute the weight K_D.

6 SUMMARY OF THE TEST RESULTS

Our approach began with the examination of simple strategies, which only had to make the AMS move in an unknown enviroment without any collisions. It turned out, that it was sufficient, to choose the point of minimum obstacle distance as a criterion. The optimisation was realized by the choice of appropriate fuzzy-sets and a dynamical rule-set.

The target-point oriented strategy, which was also based on the strategy of minimum distance worked very fine by using a dynamical rule-set and dynamical superposition of the drift angle. The occuring insufficiencies might be a problem of optimisation, too.

Our examination shows, that a mobile system using the strategy of minimum distance with dynamical rule-set and dynamical superposition of the drift angle is able to move without any collisions in an unknown environment and, with some exceptions,

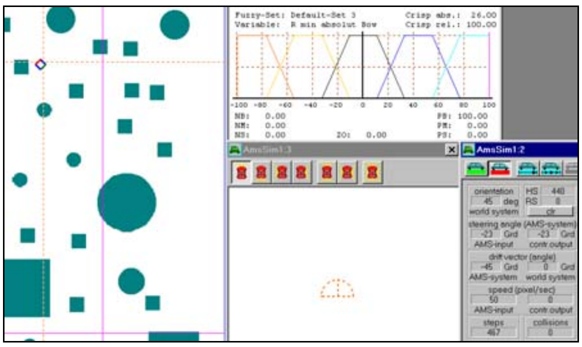

Figure 24: The target-point is reached

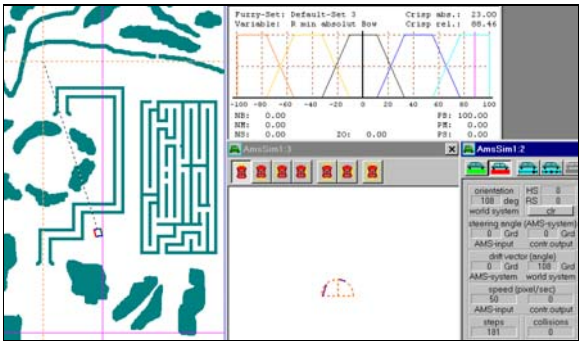

Figure 25: Course to the corridor

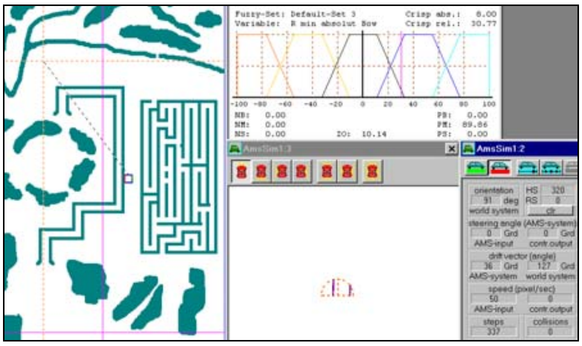

Figure 26: Using the passage

reaches the target-point.

This strategy has to be augmented and optimised.

7 FUTURE WORK

All examinations of our approach took place in a 2D-environment; the software was written with the Visual C++ - compiler.

Lately we decided to turn to the MatLab-Software because of the better possibilities of simulation with Simulink. Additionally we want to do a further step as well and enhance the software to cope not only with 2D- but also with 3D-environments.

Furthermore we intend not only to integrate natural landscape maps but also a global path planner. We want our new overall strategy working as follows:

1. Make use of a high resolution environmental map with height-information
2. Fix start- and target-point
3. Plan a global path
4. Start and move the AMS on the planned path to the goal
5. Avoid unforeseen obstacles and find back to the planned path with the strategy proposed in this paper

The simulated mobile system may be an aircraft or a car respectively. The visualisation should be realised within a VRML-world.

We hope being able to report on some results of our new approach on next year's session.

8 REFERENCES

Börcsök, J. (2000). *Fuzzy Control; Theorie und Industrieeinsatz*, 1. Auflage, Verlag Technik, 2000, Berlin

Demel, P., S. McCormac (1995). Echtzeit-Fuzzy-Kollisionsvermeidung für autonome Fahrzeuge. *5. Workshop Fuzzy Control des GMA-UA, Dortmund, 16.-17.11.1995* Serie Forschungsberichte der Fakultät für Elektrotechnik, Universität Dortmund, Bd. 295

Frank, H., J. Kahlert (1994). *Fuzzy-Logik und Fuzzy-Control*, 2. Auflage, Vieweg Verlag, 1994, Braunschweig

Guldner, J. (1995). *Intelligentes hierarchisches Regelungskonzept für autonome mobile Robotersysteme.*, Fortschr.Ber. VDI Reihe 8 Nr. 509 , VDI-Verlag, 1995, Düsseldorf

Hinkel, R. (1989). Konzeption eines echtzeitfähigen autonomen, mobilen Systems. *Vom Fachbereich Informatik der Universität Kaiserslautern genehmigte Dissertation*

Puttkamer, E.v., T. Specht, R. Trieb (1993). Lokale Kollisionsvermeidung durch Ansteuerung der maximalen Hindernisdistanz für einen autonomen mobilen Roboter. In: *Autonome mobile Systeme, 9. Fachgespräch des SFB 331 – Informations-verarbeitung in autonomen mobilen Handha-bungssystemen,* TU München, 28.-29. Okt. 1993, Seite 227 - 238

Tilli, T. (1991). *Fuzzy Logik; Grundlagen, Anwendungen, Hard- und Software*., 3. Auflage, Franzis Verlag, 1993, München

Warias, R. (2004). Entwicklung von Fuzzy-Kollisionsvermeidungs-Strategien einschließlich zugehöriger Simulations-Software für mobile Roboter. *Diploma thesis,* University of Hagen.