

Fuzzy Logic Control for Automobiles II: Navigation and Collision Avoidance System

A. Riid, D. Pahhomov and E. Rüstern

11.1 Introduction

In 1990, Nguyen and Widrow [1] applied a self-learning neural network to the truck backer-upper problem. The latter has become an acknowledged benchmark in non-linear control since and numerous other techniques have been tried, including genetic programming [2] neurogenetics [3], simplified neural network solution through problem decomposition [4], etc.¹. Computational overhead is usually very high in such applications, e.g. in [5] it is shown that about 20000 of back-up cycles are needed before neural network learns and even then, the backpropagation algorithm may not converge for some sets of training samples.

A simplified version of the control problem (consisting of the cab part only), on the other hand, has been heavily exploited in the field of fuzzy control [7-14]. Apparently, it seems to be one of these cases where the traditional application area of fuzzy logic - knowledge-based control - would be an appropriate solution. Ability to drive a car is a very common skill among people thus it should not be too difficult to find an expert whose verbal instructions would then constitute the core of the control system. Car driving skill, however, is usually learned to a degree where it rarely intrudes on consciousness (the occasions when it does are unusual circumstances like a potential accident or a situation the driver is not used to (i.e. he/she has not yet learnt it). Consequently, it is difficult to extract appropriate rules from the expert because of one's inability to explain how the action behind the steering wheel is exactly related to car positioning and further difficulties in putting it down in terms of fuzzy logic. The design of knowledge-based controller therefore becomes much more difficult than was assumed in the first place. Though the computational load is low, controller design procedure is ill-defined and plagued with the curse of dimensionality that often leads to subpar performance.

¹ Perhaps the most interesting of these contributions is [6], where up to ten trailers are controlled representing those as Takagi-Sugeno models and applying linear matrix inequalities method.

Note that knowledge-based control would be very difficult to apply for the truck and trailer system because normal driving instincts cheat us when attempting to back up a trailer truck to a loading dock. The task is so difficult that a lot of practice is needed to master the skill. And even then, when a truck driver backs up toward a loading dock, he/she will go forward and backward numerous times in order to position the truck at the dock successfully. If the driver is not allowed to make forward movements, successful backing becomes improbable.

In [15] we have shown how the decomposition of the control problem can make controller design very natural and substantially simplify expert knowledge acquisition. In the decomposed view we focus on information concerning car optimal orientation in two-dimensional space (much easier to explain and understand than the minute actions on the steering wheel), which ultimately leads to the efficient solution of the problem. The decomposition principle, according to [16], also helps to tackle the truck and trailer problem.

Current chapter presents our results in vehicle navigation control. In section 11.5, an overview of the complementing collision avoidance system proposed in [17] is given. Implementation issues and implications are discussed in section 11.6 along with the results, including some brand new ones from combining collision avoidance system with truck and trailer control system.

11.2 Problem formulation

Truck position is determined by three state variables x , y and $\Phi_c = [-90^\circ, 270^\circ]$, where the latter is the angle between truck's onward direction and the x -axis (Fig. 11.1). The width and length of the truck are denoted by w and l_c , respectively ($w = 2$, $l_c = 4$).

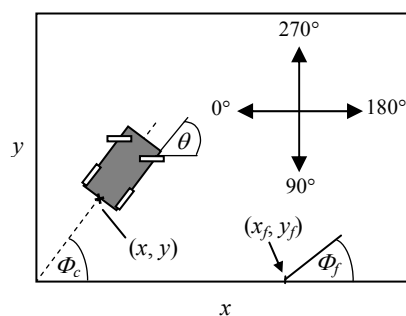


Fig. 11.1. Truck and main variables

The problem is formulated as follows: truck must arrive from the arbitrary initial position (x_0, y_0, Φ_0) to the predefined loading dock (x_f, y_f, Φ_f) . To accomplish this, appropriate steering angle $\theta = [-45^\circ, 45^\circ] = f(x, y, \Phi_c)$ must be provided. Note that the truck moves forward or backward with the fixed speed, thus speed control is not our concern.

Truck kinematics are simulated using the following set of equations

$$\begin{cases} \Phi_c(t + \Delta t) = \Phi_c(t) + (v \cdot \Delta t / l) \tan \theta \\ x(t + \Delta t) = x(t) + v \cdot \Delta t \cos(\Phi_c(t)) \\ y(t + \Delta t) = y(t) + v \cdot \Delta t \sin(\Phi_c(t)) \end{cases} \quad (11.1)$$

where v is the speed of the truck and Δt is the sampling time.

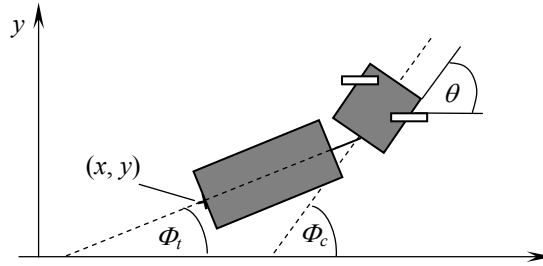


Figure 11.2. Truck and trailer

In full truck backer-upper problem (Fig. 11.2), a trailer is connected to the truck. Inclusion of the trailer brings along an additional state variable - the angle of the trailer (Φ_t). The coordinate point of the moving object is transferred from the cab part to the backside of the trailer because the control goal now is to guarantee proper positioning of the trailer part. It is also assumed that $|\Phi_t - \Phi_c| < 60^\circ$ and in this system, $\theta = [-70^\circ, 70^\circ]$. Length and width of the trailer are $l_t = 4$ and $w = 2$ meters, respectively. The dimensions (l_c) of the cab part are 2×2 m. Other assumptions and control objectives remain unchanged. Current implementation uses the set of equations from [3].

$$\begin{cases} x(t + \Delta t) = x(t) - B \cos(\Phi_t(t)) \\ y(t + \Delta t) = y(t) - B \sin(\Phi_t(t)) \\ \Phi_t(t + \Delta t) = \Phi_t(t) - \arcsin((A \sin(\Phi_c(t) - \Phi_t(t)) / l_t)) \\ \Phi_c(t + \Delta t) = \Phi_c(t) - \arcsin((v \cdot \Delta t \sin(\theta)) / (l_t + l_c)) \end{cases} \quad (11.2)$$

where $A = v \cdot \Delta t \cos(\theta)$, $B = A \cos(\Phi_c(t) - \Phi_t(t))$.

11.3 Vehicle navigation control

The most important component of the control system is the trajectory mapping unit (tmu) that provides optimal truck angle (Φ_r) for the given point in input space determined by its current coordinates x and y , to reach the loading dock situated at $(x_f = 0, y_f = 0, \Phi_f = 90^\circ)$. The information encapsulated in the tmu is very intuitive and can therefore be easily specified as can be seen from Fig. 11.3.

Computation of the actual steering angle θ that would ensure desired car orientation Φ_r is carried out by a PD control loop (Fig. 11.4). The advantages of such configuration when compared to all-in-one approaches [7,12] - where fuzzy controller is required to perform the function $\theta = f(x, y, \Phi_c)$ - become obvious. First, the number of input variables to the fuzzy system is reduced (which allows us to use y (usually neglected for the sake of simplicity in such contributions)). Secondly, the structure of the control system facilitates more efficient control knowledge acquisition and implementation. Finally, feedback in the control loop stabilizes and enhances steering quality.

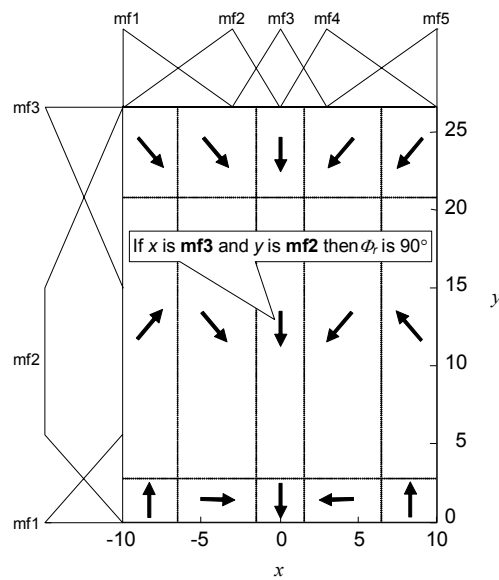


Fig. 11.3. 15 tmu rules responsible for trajectory management. Each small arrow indicates optimal angle of the truck corresponding to the given rule

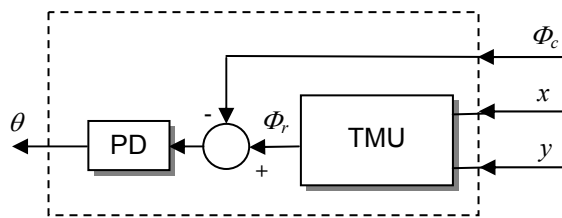


Fig. 11.4. Hierarchical control system

To expand functionality of the control system, the following modifications are made. First, in order to expand its rather small working range ($x = [-10, 10]$, $y = [0, 27]$), the saturation functions are used for input variables, x and y , which limit the input values to the upper and lower bounds of the tmu variables to ensure that car

navigation will be governed by appropriate rules even when x and y appear to be outside of the scope of the tmu.

Secondly, if $(x_f \neq 0, y_f \neq 0, \Phi_f \neq 90^\circ)$ then in order to save ourselves from retuning the tmu each time the final destination is changed, the original tmu in Fig. 11.4 is replaced by the one depicted in Fig. 11.5 that comes with built-in interface between the actual values of variables x, y, Φ_c and those that will be used for the computations in the interfaced tmu (which are denoted by x', y' and Φ_c').

The values of x' and y' are obtained using the following formulas:

$$\begin{aligned} x' &= r \cos(90^\circ - \Phi_f + \Phi_c) \\ y' &= r \sin(90^\circ - \Phi_f + \Phi_c) \end{aligned} \quad (11.3)$$

where

$$r = \sqrt{(x - x_f)^2 + (y - y_f)^2} . \quad (11.4)$$

Setpoint value corresponding to the given destination (x_f, y_f, Φ_f) is computed by

$$\Phi_r = \Phi_r' + \Phi_f - 90^\circ . \quad (11.5)$$

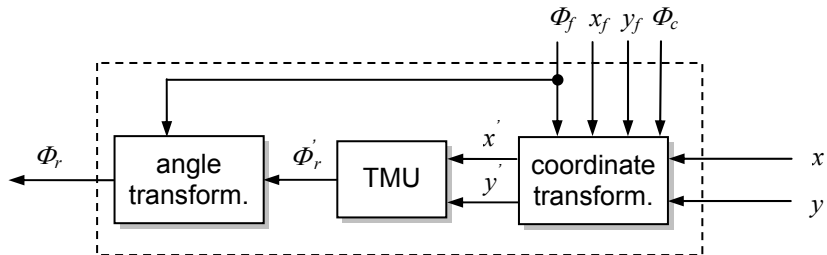


Fig. 11.5. Generalized tmu interface

Third, the tmu should also be extended to cover the negative range of values of y where different actions have to be taken to bring the truck home. To avoid undesirable co-effects from the interpolation of antagonistic rules, the part of the tmu that corresponds to the range of negative values of y , is realized as a separate piece. (Fig. 11.6).

Note that the expanded system can also be used for truck control in forward driving mode if we correct Φ_r by 180° (the reason is obvious) and multiply the PD controller output θ by -1 (the latter is necessary because turning the wheels to the left would turn the car to the right in backward driving mode but would have the opposite result in forward driving mode).

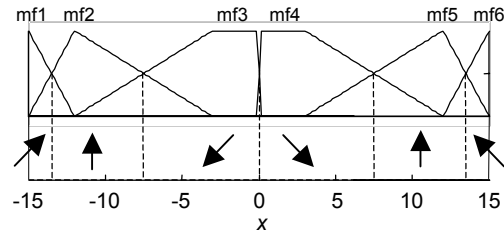


Fig. 11.6. Tmu rules responsible for trajectory management when $y < 0$

Also note that the control system is currently optimized to truck's physical parameters specified in section 11.2. Although the control system is quite robust, substantial increase of car dimensions and/or smaller maximum steering angle would both hamper car maneuverability, which may result in a failure. If this is the case, it is possible to adjust the tmu to the changes with the help of scaling factors applied to tmu inputs x and y . In fact, these scaling factors come handy already in the next section, in the truck and trailer control system.

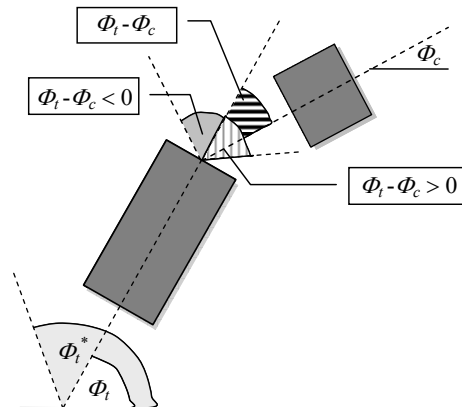


Fig. 11.7. Relationship between truck and trailer angles

11.4 Truck and trailer backing control

With truck and trailer, the output of the tmu is interpreted as desired trailer angle Φ_t^* . From Fig. 11.7 it is obvious that if trailer orientation error $\Phi_t^* - \Phi_t$ is positive, we must increase Φ_t (and similarly decrease it if the error is negative) in order to reduce the error. Φ_t grows when $\Phi_t - \Phi_c$ remains positive. To establish this, a simple joint controller that computes appropriate $(\Phi_t - \Phi_c)^*$ - desired difference between truck and trailer angles against trailer orientation angle - is specified (Fig. 11.8). According to our reasoning, the function realized by the joint controller must be monotonously growing. We have implemented the highly nonlinear curve by the means of fuzzy logic where final membership function values were found

by manual tuning during the backing (as so few variables and rules (five) are involved, the procedure is not overly complicated).

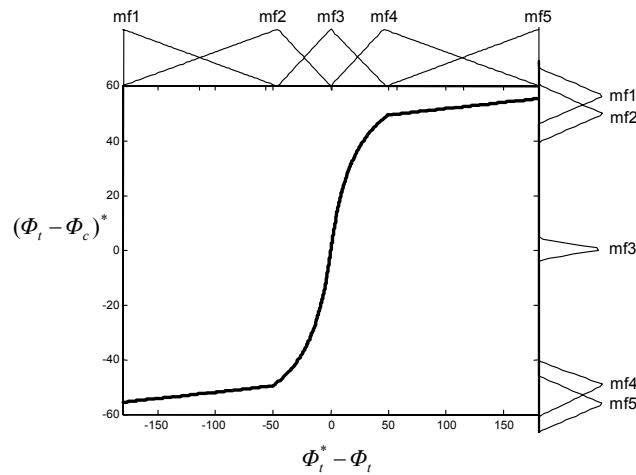


Fig. 11.8. The joint controller

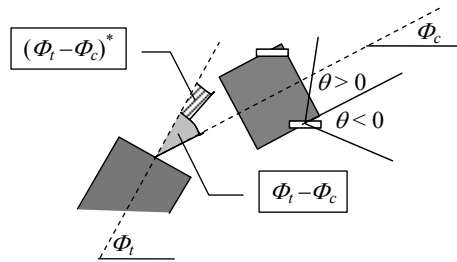


Fig. 11.9. Angles around the joint

Steering angle again can be computed by PD control loop, however, we need to determine the its sign first – i.e. if the actual difference between the truck and the trailer $\Phi_t - \Phi_c$ is larger than the desired one $(\Phi_t - \Phi_c)^*$, we must increase Φ_c (Fig. 11.9) and thus θ must be negative in this case.

The resulting control system is depicted in Fig. 11.10. Note that the parameters of the PD controller are $K_p = -12$, $K_d = -4$ (because of the sign issue) and that variables x and y have been scaled by $k_x = k_y = 0.8$.

11.5 Collision avoidance system

As such, the vehicle or more appropriately, its control system, is ignorant of possible obstacles coming its way (including the bounding walls) and is therefore

unable to avoid consequent collisions. This section explains how the problem can be solved through some extension of the original control system.

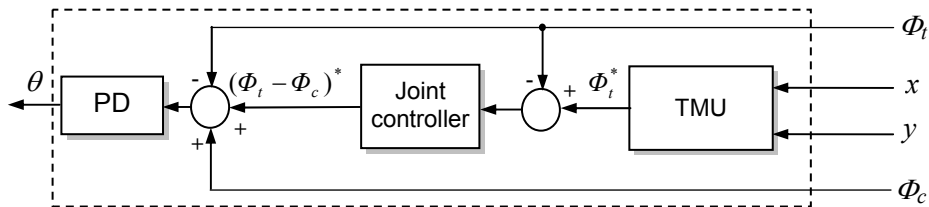


Fig. 11.10. Truck and trailer backer-upper control system

For obstacle avoidance the truck must be first equipped with the devices that make possible object detection. 8 distance sensors are “installed” to truck body and in case of truck and trailer system – to trailer body (although only four of them will be active at a time): left and right rear sensors s_1 and s_2 , right and left side sensors $s_3 - s_6$ and left and ride frontal sensors s_7 and s_8 giving readings d_1-d_8 , respectively (Fig. 11.11).

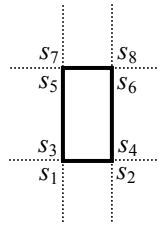


Fig. 11.11. Distance sensor placement scheme

Collision avoidance in backward driving mode works on the following principle: if an obstacle is detected in truck’s way (either d_1 or d_2 is smaller than predetermined detection limit d_{det}), first, the turning direction (left or right) is determined the on the basis of d_1-d_4 (Fig. 11.12).

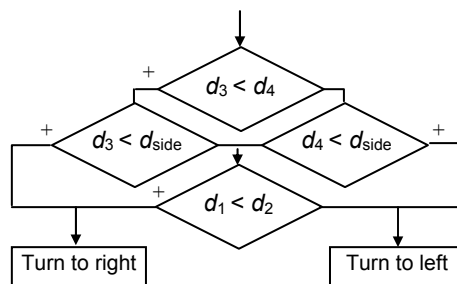


Fig. 11.12. Turning decision making

Based on this decision and orientation of the obstacle (that will be identified according to rear distance sensor readings) a temporary gate is projected onto the bounding wall in car's turning direction (Fig. 11.13). Note that presently, turning direction is chosen according to rear sensor readings only, as there are no relevant obstacles in car side directions. Also note that the opposite turning decision would result in the temporary gate projected onto opposite (upper) bounding wall.

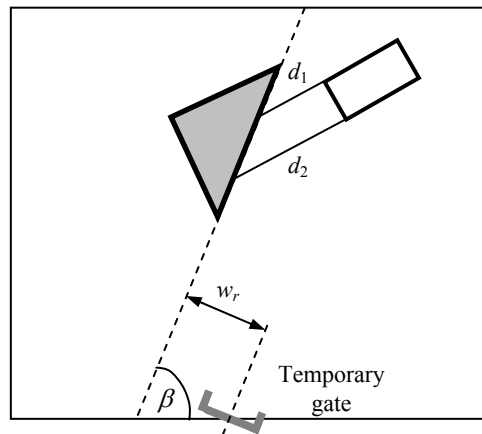


Fig. 11.13. Temporary driving goal

In forward driving mode the algorithm is basically the same, only we are using sensors s_5 - s_8 instead of s_4 , s_3 , s_2 and s_1 , respectively and the obstacle orientation angle β must be corrected by 180 degrees.

Performance of the collision avoidance system is subject to the following parameters

d_{det} – obstacle detection limit

d_{excl} – rear sensor readings larger than d_{excl} will be emulated by $d + w$, where d is the rear sensor reading responsible for obstacle detection.

d_{side} – determines if side sensor information will be used in turning direction determination (Fig. 11.12).

w_r – the distance by which the temporary gate is shifted from the obstacle-projected line (Fig. 11.13).

11.5.1 State transition control

We can speak of two states in car control – normally car moves toward (x_f, y_f, Φ_f) according to the tmu-specified profile (state 0), however, when an obstacle is detected by distance sensors, collision avoidance system (previous section) takes over and switches the tmu to the temporary gate (state 1). This temporary goal remains valid until it is safe to return to the normal state. For overall good performance state transition control must be smooth and effective. This is done in three steps as explained below.

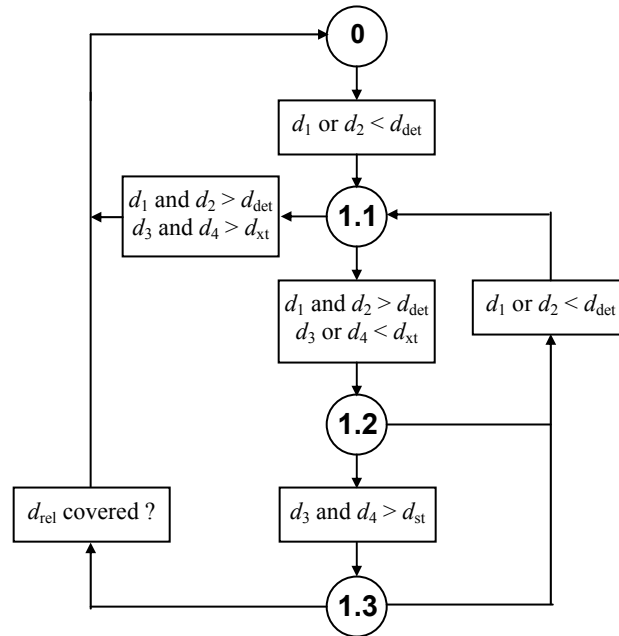


Fig. 11.14. State transition diagram (backward driving mode)

Collision avoidance system takes over if at least one of the sensors in the moving direction of the car registers distance smaller than d_{det} (transition from state 0 to 1.1 in Figs. 11.14 and 11.15). The car then starts to rotate because of the temporary driving goal that has been imposed in the moment of state transition and further state management will be delegated to side sensors if the obstacle comes to the reach of one of them (substate 1.2) - unless a new obstacle is detected and new temporary gate must be computed (return to substate 1.1). Eventual return to state 0 can principally happen if the obstacle is no longer present at car's side (substate 1.3). In certain circumstances, however, it may be necessary to delay the release further (too early release may cause a collision with the obstacle currently being passed), which is controlled by d_{rel} - small distance that has to be covered while in substate 1.3.

State transition is independently regulated by two extra parameters (d_{min} and d_{max}) that determine minimum and maximum time system may spend in state 1, respectively. The aim of d_{min} is to make car control less sensitive to rapid changes of external conditions. The need for d_{max} becomes apparent when a bounding wall or connected obstacles happen to be in car's way because in this case the car cannot break out from state 1 on its own, even if it would be good for the overall goal.

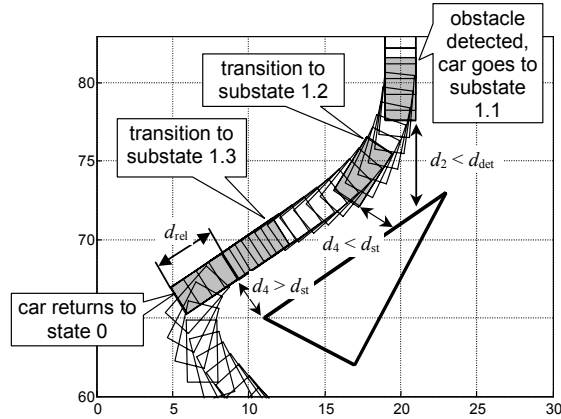


Fig. 11.15. State transition in obstacle avoidance

11.6 Results and discussion

In this section simulated results with the designed navigation and collision avoidance system (mostly in the more challenging backward driving mode) obtained in various environments are presented. One of these environments contains obstacles of various shapes (Fig. 11.18), the other two are the labyrinths (Figs 11.19a and 11.20). In order to perform the requested tasks we need to determine proper settings of d_{side} , d_{det} , d_{excl} , d_{st} , w_r , d_{rel} , d_{min} and d_{max} . These values mostly depend on the dimensions of the controlled object and the nature of the task (character of obstacles). Note that listed parameters influence only the performance of the collision avoidance system, as the ability to reach the preset destination depends solely on the t_{mu} which has currently been optimized for given truck or truck and trailer system dimensions.

It is possible to detect internal dependencies in parameter settings. w_r must be chosen such that the car would leave reasonable distance between itself and the obstacle wall (about few car widths), presently it is 4m for the truck and 6m for the truck and trailer system. The value of the detection limit d_{det} partially derives from the former i.e. it must be large enough (in current setting this parameter equals 6 and 12 meters, respectively) to provide smooth trajectory from the detection point to the point where w_r is established (Fig. 11.16a). On the other hand, it is important to keep this parameter as small as possible not to make the car to react to the obstacles at longer distances (Fig. 11.16b). Obviously, d_{st} (in current setting 9 and 15 meters) must be larger than w_r to make sense and similar to d_{det} (the reason becomes apparent in Fig. 11.17a). d_{side} must also be in the same range (in current setting 8 and 11 meters) as its too large value can lead to incorrect strategical decisions (see Fig. 11.17b). Release distance d_{rel} is equal to 0 in backward driving mode but 2 meters in forward driving mode because the turning radius of the car is smaller in the latter case. The remaining parameters - $d_{min} = 5$, $d_{max} = 80$, $d_{excl} = 30$ - are obtained rather empirically and/or in a few test drives.

Numerically, the driving results depicted in Figs. 11.18-20 can be evaluated by the combined measure proposed in [15]. This measure, however, evaluates the final position of the car that primarily depends on the t_{mu} (its ability to provide smooth and effective car trajectories has been documented in [15, 16]). The performance of the collision avoidance system has binary character (success/failure) and can easily be identified from respective Figs.

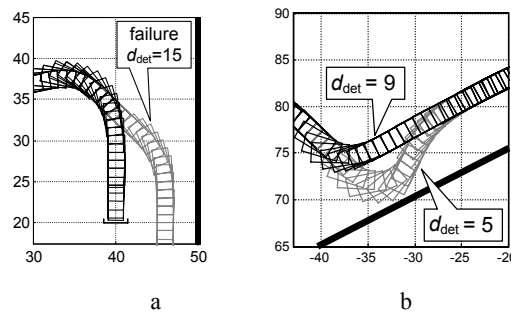


Fig. 11.16. Settings for d_{det} and d_{side} : a. too large value of d_{det} may connect the car with irrelevant obstacles (wall at right); b. Non-smooth trajectory in substate 1.1 resulting from too small value of d_{det} (compared to $w_r = 6$)

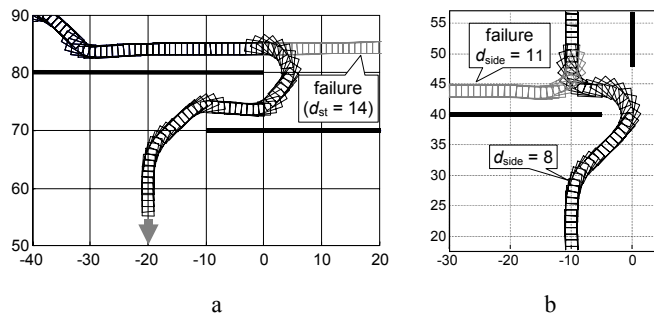


Fig. 11.17. Settings for d_{st} and d_{side} : a. car does not come out of substate 1.2 if d_{st} is too large; b. car fails to break out from the maze if the setting of d_{side} is improper

Despite good performance in presented simulations, it is clear that there exist limits on what the proposed control system can do. First – the test assignments are based on the reasonable assumption that they must be solvable under current capacities of the control system. E.g. there must be enough space for the car to perform necessary maneuvers.

Moreover, although the designed subsystems have highly cooperative nature, there are situations where it is impossible to perform the expected task because of the conflict of interest of theirs. Fig. 11.19b presents one such case. At point a, the obstacle is recognized and car starts its usual routine to avoid it. At point b, it encounters another obstacle and acts accordingly. The car is released at point c but because of the position of the global goal (loading dock) the car performs left turn

and returns to the point it has already visited, apparently unable to find a solution to the problem.

For such situations, another extension or “higher level of intelligence” of the control system is needed. It should be able to recognize a failure and make more general decisions about the actions to be taken, i.e. when the car recognizes that it arrives in the position it has been before, it should force the system to solve the problem in alternative way and apply it in the right moment (in point c).

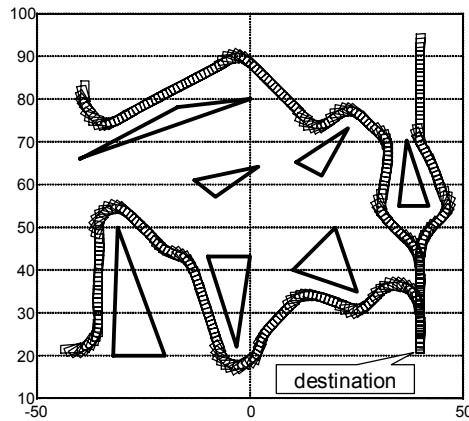


Fig. 11.18. Driving the truck backwards in the field

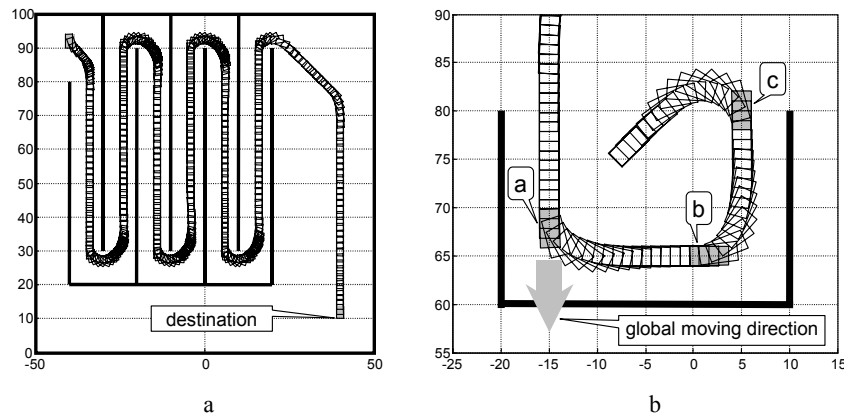


Fig. 11.19. Results: a. forward-driven truck in the labyrinth; b. a possible failure situation

Secondly, the existing system performs in the noise-free environment. In real life, however, all computations that depend on measured data are subject to noise. This affects both car position determination and decisions based on distance sensor information.

The material in this chapter appears as the groundwork for car (or similar moving objects, such as mobile robots) navigation system model-scale implementation and although it is possible to simulate the noise as well, at this

stage of the project we do not know yet what kind of noise corresponds to the conditions we are going to encounter in real life. It is possible to make the existing system less sensitive to noise if the need arises, either by sensor information processing, appropriate tmu modification or more conservative (i.e. larger) system parameter values (this actually has been tested with white noise and found to be useful for collision avoidance system).

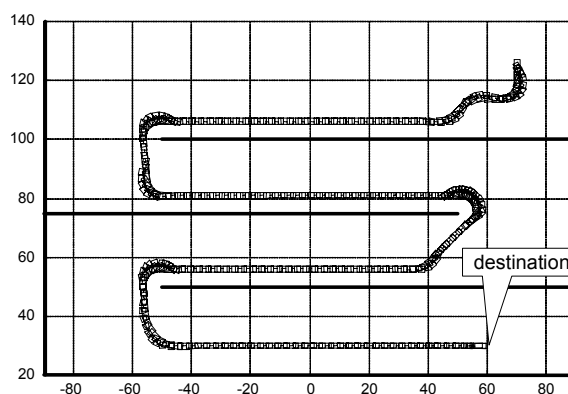


Fig. 11.20. Backward-driven truck and trailer system performance in the labyrinth

11.7 Conclusions

In this chapter, a fuzzy logic enhanced navigation and collision avoidance system for truck or truck and trailer systems has been described. It assumes that vehicle coordinates and its orientation in respect to x -axis can be determined and is based on the fuzzy trajectory mapping unit that provides smooth car trajectory management independent of object's initial position or the position of the loading dock. It is suggested that for performing more demanding tasks than demonstrated here, additional blocks adding more "intelligence" are required. The latter is made possible by the modular structure of the control system that also enables us to tune subsystems of the control system responsible for different tasks individually without jeopardizing overall performance.

11.8 Acknowledgement

The research has been partially supported by Estonian Science Foundation, grant no. 5170.

11.9 References

- [1] D. Nguyen and B. Widrow, "The truck backer-upper: An example of self-learning in neural network," *IEEE Contr. Syst. Mag.*, 1990, Vol. 10, No. 2, pp. 18-23.
- [2] J.R. Koza, "A genetic approach to the truck backer upper problem and the intertwined spirals problem". *Proc. Int. Joint Conf. Neural Networks*, Piscataway, NJ, Vol. 4, 1992, pp. 310-318.
- [3] M. Schoenauer and E. Ronald, "Neuro-genetic truck backer-upper controller. Proc. *First Int. Conf. Evolutionary Comp.*, Orlando, FL, USA, 1999, pp. 720-723.
- [4] R.E. Jenkins and B.P. Yuhua, "A Simplified Neural Network Solution Through Problem Decomposition: The Case of the Truck Backer-Upper", *IEEE Trans. Neural Networks*, Vol. 4, No. 4, 1993, pp. 718-720.
- [5] S. Kong and B. Kosko, "Comparison of fuzzy and neural truck backer-upper control systems". *Proc. IJCNN*, Vol. 3, 1990, pp. 349-358.
- [6] K. Tanaka, T. Kosaki and H.O. Wang, "Backing Control Problem of a Mobile Robot with Multiple Trailers: Fuzzy Modeling and LMI-Based Design". *IEEE Trans. Syst., Man, Cybern. Part C*, Vol. 28, No. 3, 1998, pp. 329-337.
- [7] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. on System, Man, and Cybernetics*, Vol. 22, No. 6, 1992, pp. 1414-1427.
- [8] P.A. Ramamoorthy and S. Huang, "Fuzzy Expert Systems vs. Neural Networks – Truck Backer-Upper Control Revisited," *Proc. IEEE*, 1991, pp. 221-224.
- [9] J.J. Shann and H.C. Fu, "A fuzzy neural network for rule acquiring on fuzzy control systems", *Fuzzy Sets and Systems*, Vol. 71, 1995, pp. 345-357.
- [10] A. Ismail and E.A.G. Abu-Khousa, "A Comparative Study of Fuzzy Logic and Neural Network Control of the Truck Backer-Upper System," *Proc. IEEE Int. Symp. On Intelligent Control*, Dearborn, 1996, pp. 520-523.
- [11] D. Kim, "Improving the fuzzy system performance by fuzzy system ensemble", *Fuzzy Sets and Systems*, Vol. 98, 1998, pp. 43-56.
- [12] S.-G. Kong, B. Kosko, "Adaptive Fuzzy Systems for Backing up a Truck-and-Trailer," *IEEE Trans. on Neural Networks*, Vol. 3, No. 5, 1992, pp. 211-223.
- [13] I. Dumitrache and B. Catalin, "Genetic learning of fuzzy controllers," *Mathematics and Computers in Simulation*, Vol. 49, 1999, pp. 13-26.
- [14] M.-C. Su and H.-T. Chang, "Application of neural networks incorporated with real-valued genetic algorithms in knowledge acquisition," *Fuzzy Sets and Systems*, vol. 112, 2000, pp. 85-97.
- [15] A. Riid and E. Rüstern, "Fuzzy logic in control: truck backer-upper problem revisited" *Proc. 10th IEEE International Conference on Fuzzy Systems*, Melbourne, Vol. 1, 2001, pp. 513-516.
- [16] A. Riid and E. Rüstern, "Fuzzy Hierarchical Control of Truck and Trailer" *Proc. 8th Biennial Baltic Electronic Conf.*, Tallinn, Estonia, 2002, pp. 141-144.
- [17] A. Riid and E. Rüstern, "Car Navigation and Collision Avoidance System with Fuzzy Logic", *Proc IEEE International Conference on Fuzzy Systems*, Budapest, Vol. 3, 2004, pp. 1443-1448.