IVCS 2008

Oleg Gusikhin (Ed.)

Intelligent Vehicle Control Systems

Proceedings of the
2nd International Workshop on
Intelligent Vehicle Control Systems - IVCS 2008

In conjunction with ICINCO 2008
Funchal, Madeira - Portugal, May 2008

INSTICC PRESS

Oleg Gusikhin (Ed.)

Intelligent Vehicle Control Systems

Proceedings of the
2nd International Workshop on
Intelligent Vehicle Control Systems
IVCS 2008

In conjunction with ICINCO 2008
Madeira, Portugal, May 2008

INSTICC PRESS
Portugal

Volume Editor

Oleg Gusikhin
Ford Research & Adv. Engineering
U.S.A.

2nd International Workshop on
Intelligent Vehicle Control Systems
Madeira, Portugal, May 2008
Oleg Gusikhin (Ed.)

Printed in Portugal

ISBN: 978-989-8111-34-0
Depósito Legal: 273832/08

Foreword

This volume contains the proceedings of the *2nd International Workshop on Intelligent Vehicle Control Systems* held on May 2008, in Madeira - Portugal, in conjunction with the *5th International Conference on Informatics in Control, Automation and Robotics*. The goal of this workshop is to bring together representatives from academia, industry and government agencies to exchange ideas on state of the art intelligent vehicle systems and future trends.

In recent years, the growing role of informatics in controls is probably most evident in automotive applications. The increasing complexity of modern automotive systems often calls for computational intelligence approaches, whereas traditional control methods are infeasible, ineffective or not economical. Furthermore, with the proliferation of drive-by-wire technologies, advances in sensory, navigation, and wireless communication infrastructure, vehicle controls can now take advantage of the information regarding the state of an environment and a driver, implementing functionalities that are commonly referred to as intelligent.

The workshop opens with a keynote lecture on European efforts for standardization for GPS and map-based driver assistance presented by Christian Ress of the Ford of Europe Telematics & Navigation team who is the 2008 Chairman of the European Advanced Driver Assistance System Interface Specification (ADASIS) Forum. The workshop received 12 submissions. The final program of the workshop is comprised of 9 papers, with 8 double-blind reviewed papers and one invited paper by IEEE Fellow Dr. Ilya Kolmanovsky. The papers represent both academia and automotive industry with topics ranging from case studies from the 2007 Urban Challenge autonomous vehicle competition to stochastic optimization methods for adaptive cruise control.

I would like to thank all of the authors for their contributions to this workshop, members of the program committee and reviewers for providing their support, suggestions and comments on the technical directions, and thorough reviews for the papers. In addition, I would like to acknowledge the assistance from my Ford colleagues Erica Klampfl, Perry MacNeille, Diana Yanakiev and Doug Rhode. Last, but not least, I would like to express my gratitude to the ICINCO Organization Committee, with special thanks to Professor Joaquim Filipe for

inspiration and encouragement for the IVCS workshop and to Marina Carvalho for her tremendous efforts in making this workshop happen.

Oleg Gusikhin
Ford Research & Advanced Engineering
Dearborn, Michigan, U.S.A.

Workshop Chair
Oleg Gusikhin
Ford Research & Advanced Engineering
Dearborn, Michigan, U.S.A.

Program Committee

Plamen Angelov, Lancaster University, U.K.
Trevor Darrell, MIT, U.S.A.
Dimitar Filev, Ford, U.S.A.
TJ Giuli, Ford, U.S.A.
Riad Hammoud, Delphi, U.S.A.
Christian Jones, Affective Media Ltd, U.K.
Ken Kendall, Aston Martin, U.K.
Erica Klampfl, Ford, U.S.A.
Ilya Kolmanovski, Ford, U.S.A.
Anatoli Koulinitch, Visteon, U.S.A.
Urban Kristiansson, Volvo Cars, Sweden
John Krumm, Microsoft, U.S.A.
Dinesh Kumar, RMIT University, Australia
Perry MacNeille, Ford, U.S.A.
Gerard T. McKee, University of Reading, U.K.
Hiroshi Nakajima, OMRON, Japan
Brian Noble, University of Michigan, U.S.A.
Danil Prokhorov, Toyota, U.S.A.
Doug Rhode, Ford, U.S.A.
Diana Yanakiev, Ford, U.S.A.
Prasad Venkatesh, Ford, U.S.A.
Hao Ying, Wayne State University, U.S.A.

Table of Contents

Keynote Lecture

European Standardization for Navigation based Advanced Driver Assistant Systems (ADAS) - The ADASIS Forum

Christian Ress

Telematics & Navigation Research
Vehicle Technologies & Materials
Ford Research & Advanced Engineering Europe

Abstract. With the development of navigation based ADAS functions the interface to access this so-called Electronic Horizon data is of rising importance. In the automotive industry standard interfaces are appreciated to reduce development cost and risk. In order to specify an industry standard interface for providing Electronic Horizon the ADASIS[1] Forum has been launched. The Forum is hosted and coordinated by ERTICO[2] and constitutes of more than 30 members including car manufacturers, navigation system and ADAS suppliers, as well as digital map vendors. The forum's purpose is to:

- Define an open standardised data model and structure to represent map data in the vicinity of the vehicle position (i.e. the Electronic Horizon), in which map data is delivered by a navigation system or a general map data server.
- Define an open standardised API to enable ADAS applications to access the Electronic Horizon and position-related data of the vehicle.

A first version of the interface specification is already available and has been tested and validated within the PReVENT[3] project. The results from PReVENT demontrated successfully the feasibility and interroperability of ADASIS. Nevertheless also some shortcomings have been identified, which are currently addressed by the various Forum's workng groups. In fact, a next version of the protocol specifications is under development and will be transmitted to ISO for becoming an international industry standard.

[1]ADASIS = Advanced Driver Assistance System Interface Specification
[2]ERTICO = European ITS organisation
[3]PReVENT is a European industry research project, that has been co-funded by European Commission within 6th framework.

INVITED PAPER

Discrete-Time Drift Counteracting Stochastic Optimal Control and Intelligent Vehicle Applications

Ilya Kolmanovsky and John Michelini

Ford Research and Advanced Engineering, Ford Motor Company
2101 Village Road, Dearborn, Michigan, U.S.A.
{ikolmano,jmichel1}@ford.com

Abstract. In this paper we present a characterization of a stochastic optimal control in the problem of maximizing expected time to violate constraints for a nonlinear discrete-time system with a measured (but unknown in advance) disturbance modeled by a Markov chain. Such an optimal control may be viewed as providing drift counteraction and is, therefore, referred as the drift counteracting stochastic optimal control. The developments are motivated by an application to an intelligent vehicle which uses an adaptive cruise control to follow a randomly accelerating and decelerating vehicle. In this application, the control objective is to maintain the distance to the lead vehicle within specified limits for as long as possible with only gradual (small) accelerations and decelerations of the follower vehicle so that driver comfort can be increased and fuel economy can be improved.

1 Introduction

In the paper we examine a stochastic optimal control problem motivated by an application of adaptive cruise control to follow a randomly accelerating and decelerating vehicle. For this application, we consider the control objective to maintain the distance to the lead vehicle within specified limits for as long as possible with only very gradual (small) accelerations and decelerations so that to improve fuel economy and increase driver comfort. This and similar application problems can be treated using methods of stochastic drift counteracting optimal control developed in [6].

The paper is organized as follows. In Section 2 we discuss a formulation of the stochastic drift counteracting optimal control problem for a nonlinear discrete-time system with measured (but unknown in advance) disturbance input modeled by a Markov chain. In Section 3 we review the theoretical results [6] pertinent to the characterization and computations of the stochastic optimal control law in this problem. We also present a result to compute expected time to violate the constraints for a fixed control policy, which may be useful in evaluating legacy control solutions. In Section 4 we discuss a simulation example illustrating the application of these methods to a vehicle following, where the lead vehicle speed trajectory is modeled by a Markov chain with known transition probabilities. Concluding remarks are made in Section 5.

2 Problem Formulation

Consider a system which can be modeled by nonlinear discrete-time equations,

$$x(t+1) = f(x(t), v(t), w(t)), \tag{1}$$

where $x(t)$ is the state vector, $v(t)$ is the control vector, $w(t)$ is the vector of measured disturbances, and t is an integer, $t \in Z^+$. The system has control constraints which are expressed in the form $v(t) \in U$, where U is a given set.

The behavior of $w(t)$ is modeled by a Markov chain [3] with a finite number of states $w(t) \in W = \{w^j,\ j \in J\}$. The transition probability from $w(t) = w^i \in W$ to $w(t+1) = w^j \in W$ is denoted by $P(w^j|w^i, \bar{x})$. In our treatment of the problem, we permit this transition probability to depend on the state $x(t) = \bar{x}$. For automotive applications, modeling driving conditions using Markov chains for the purpose of applying Stochastic Dynamic Programming to determine fuel and emissions optimal powertrain operating policies has been first proposed in [4].

Our objective is to determine a control function $u(x, w)$, such that with $v(t) = u(x(t), w(t))$, a cost functional of the form,

$$J^{x_0,w_0,u} = E_{x_0,w_0} \tau^{x_0,w_0,u}(G), \tag{2}$$

is maximized. Here $\tau^{x_0,w_0,u}(G) \in Z^+$ denotes the first-time instant the trajectory of $x(t)$ and $w(t)$, denoted by $\{x^u, w^u\}$, resulting from the application of the control $v(t) = u(x(t), w(t))$, exits a prescribed compact set G. See Figure 1.

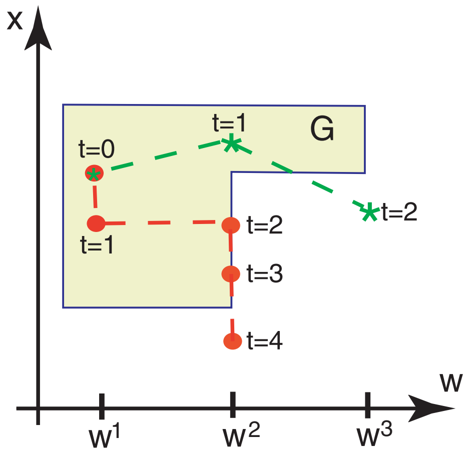


Fig. 1. The set G and two trajectories, $\{x^u, w^u\}$, exiting G at random time instants due to a random realization of $w(t)$. Here $W = \{w^1, w^2, w^3\}$. Note that one of the trajectories exits G at $t = 4$ due to the evolution of $x(t)$ alone, the other trajectory exits G at $t = 2$ due to evolution of both $x(t)$ and $w(t)$.

The specification of the set G reflects constraints existing in the system. Note that $\{x^u, w^u\}$ is a random process, $\tau^{x_0,w_0,u}(G)$ is a random variable and $E_{x_0,w_0}[\cdot]$ denotes the expectation conditional to initial values of x and w, i.e., $x(0) = x_0, w(0) = w_0$. When clear from the context, we will omit the subscript and square brackets around E.

For continuous-time systems, under an assumption that $w(t)$ is a Wiener or a Poisson process, it can be shown [1] that determining an optimal control in this kind of a problem reduces to solving a non-smooth Partial-Differential Equation (PDE). For instance, for a first order stochastic system, $dx = (v - w_0)dt + \sigma \cdot dw$, where w_0 is a constant, w is a standard Wiener process, the control v satisfies $|v| \leq \bar{v}$, this PDE has the form,

$$\frac{1}{2}\sigma^2 \frac{\partial^2 V}{\partial x^2} + \frac{\partial V}{\partial x}(-w_0) + |\frac{\partial V}{\partial x}|\bar{v} + 1 = 0.$$

The boundary conditions for this PDE are $V(x) = 0$ for $x \in \partial G$, where ∂G denotes the boundary of G. The optimal control has the form

$$v = \bar{v} \cdot sign(\frac{\partial V}{\partial x}).$$

Note that this optimal control is of bang-bang type.

As compared to solving the above PDE numerically, the discrete-time treatment of the problem, which is the focus of the present paper, appears to provide a more computationally tractable approach to determining the optimal control. In what follows, we will treat this discrete-time optimal control problem within the framework of optimal stopping [3] and drift counteraction [5], [6] stochastic optimal control.

3 Theoretical Results and Computations

Given a state vector, x^-, and disturbance vectors, $w^-, w^+ \in W$, we define,

$$\begin{aligned} L^u V(x^-, w^-) &\triangleq E_{x^-,w^-}\Big[V(f(x^-, u(x^-, w^-), w^-), w^+)\Big] - V(x^-, w^-) \\ &= \sum_{j \in J} V(f(x^-, u(x^-, w^-), w^-), w^j) \cdot P(w^j \big| w^-, x^-) \qquad (3) \\ &- V(x^-, w^-). \end{aligned}$$

The following theorem provides sufficient conditions for the optimal control law, $u_*(x, w)$:

Theorem 1: Suppose there exists a control function $u_*(x, w)$ and a continuous, non-negative function $V(x, w)$ such that

$$\begin{aligned} &L^{u_*}V(x, w) + 1 = 0, \text{ if } (x, w) \in G, \\ &L^u V(x, w) + 1 \leq 0, \text{ if } (x, w) \in G, u \neq u_*, \qquad (4) \\ &V(x, w) = 0, \text{if } (x, w) \notin G. \end{aligned}$$

Then, u_* maximizes (2), and for all $(x_0, w_0) \in G$, $V(x_0, w_0) = J^{x_0,w_0,u_*}$, $J^{x_0,w_0,u}$ and $E[\tau^{x_0,w_0,u}(G)]$ are finite for any policy u, and the function V, satisfying (4), if exists, is unique.

Proof: The theorem follows as an immediate application of a more general result developed in [6]. More specifically, in [6], a similar result is shown for cost functionals of the form

$$J^{x_0,w_0,u} = E_{x_0,w_0} \sum_{t=0}^{\tau^{x_0,w_0,u}(G)-1} g(x(t),v(t),w(t)),$$

with $g \geq \varepsilon > 0$, of which (2) is a special case with $g = 1$. ■

The following procedure for estimating the expected time to violate constraints for a fixed control law is obtained as an immediate consequence of Theorem 1:

Corollary 1: Given a fixed control law $\bar{u}(x,w)$, suppose there exists a continuous, non-negative function $\bar{V}(x,w)$ such that

$$\begin{aligned} &L^{\bar{u}}\bar{V}(x,w) + 1 = 0, \text{ if } (x,w) \in G, \\ &\bar{V}(x,w) = 0, \text{if } (x,w) \notin G. \end{aligned} \tag{5}$$

Then, $E[\tau^{x_0,w_0,\bar{u}}(G)] = \bar{V}(x_0,w_0)$.

We next consider the application of the value iteration approach to (4), assuming, for simplicity of exposition, that f is continuous in x, and that U is compact. The proofs of subsequent results are similar to [6, 5] and are not reproduced here. We define a sequence of value functions using the following iterative process:

$$\begin{aligned} &V_0 \equiv 0 \\ &V_n(x,w^i) = \max_{v\in U}\left\{\sum_{j\in J} V_{n-1}(f(x,v,w^i),w^j)P(w^j|w^i,x) + 1\right\}, \text{ if } (x,w^i) \in G. \\ &\qquad n > 0. \end{aligned} \tag{6}$$

This sequence of functions $\{V_n\}$ yields the following properties:

Theorem 2: Suppose the assumptions of Theorem 1 hold. Then the sequence of functions $\{V_n\}$, defined in (6), is monotonically non-decreasing and $V_n(x,w^i) \leq J^{x,w^i,u_*}$ for all n, x and w^i. Furthermore, $\{V_n\}$ converges pointwise to $V_*(x,w^i) = J^{x,w^i,u_*}$ and this convergence is uniform if J^{x,w^i,u_*} is continuous.

On the computational side, either value iterations or Linear Programming may be used to numerically approximate the solution to (4).

The value iterations (6) produce a sequence of value function approximations, V_n, at specified grid-points $x \in \{x^k,\ k \in K\}$, and a stopping criterion is $|V_n(x,w^i) - V_{n-1}(x,w^i)| \leq \epsilon$ for all $x \in \{x^k,\ k \in K\}$ and $i \in J$, where $\epsilon > 0$ is sufficiently small. In each iteration, once the values of V_{n-1} at the grid-points have been determined, linear or cubic interpolation may be employed to approximate $V_{n-1}(f(x^k,v^m,w^i),w^j)$, on the right-hand side of (6), where $v \in \{v^m,\ m \in M\}$ is a specified grid for v. Formally, the approximate value iterations can be represented as follows,

$$\begin{aligned} &V_0(x^k,w^i) \equiv 0, \\ &V_n(x^k,w^i) = \max_{v^m, m\in M}\left\{\sum_{j\in J} F_{n-1}(f(x^k,v^m,w^i),w^j)\cdot P(w^j|w^i,x^k) + 1\right\}, \\ &\text{where} \\ &F_{n-1}(x,w^i) = \text{Interpolant}[V_{n-1}](x,w^i) \text{ if } (x,w^i) \in G, \\ &\text{and } F_{n-1}(x,w^i) = 0 \text{ if } (x,w^i) \notin G. \end{aligned} \tag{7}$$

An alternative approach is to seek V in the form,

$$V(x, w^i) = \sum_{l \in L} \theta_l \phi_l(x, w^i),$$

where ϕ_l are specified basis functions satisfying the property that $\phi_l(x, w^i) = 0$ if $(x, w^i) \notin G$. Then relations (4) evaluated over specified grid points $x \in \{x^k, k \in K\}$, $v \in \{v^m, m \in M\}$, and $i \in J$, lead to a Linear Programming problem with respect to $\theta_l, l \in L$:

$$\begin{aligned} &\sum_{l \in L} \theta_l \sum_{k \in K, i \in J} \phi_l(x^k, w^i) \rightarrow \min, \\ &\text{subject to} \\ &\sum_{l \in L} \theta_l \phi_l(x^k, w^i) \geq 1 + \sum_{l \in L} \theta_l \sum_{j \in J} \phi_l(f(x^k, v^m, w^i), w^j) \cdot P(w^j | w^i, x^k), \\ &k \in K, i \in J, m \in M. \end{aligned} \tag{8}$$

There are many aspects, such as selection of the grids and basis functions, which can be exploited to optimize the computations for specific problems. The dependence of the approximation error on the properties of the grid can be established using, for instance, techniques in Chapter 16 of [2].

Once an approximation of the value function, V_*, is available, an optimal control may be determined from the following relation:

$$\begin{aligned} u_*(x, w^i) &\in argmax_{v \in U} \Big\{ 1 + \sum_{j \in J} V_*(f(x, v, w^i), w^j) P(w^j | w^i, x) \Big\}, \\ &\text{or} \\ u_*(x, w^i) &\in argmax_{v \in U} \Big\{ \sum_{j \in J} V_*(f(x, v, w^i), w^j) P(w^j | w^i, x) \Big\}. \end{aligned} \tag{9}$$

4 Vehicle Following Example

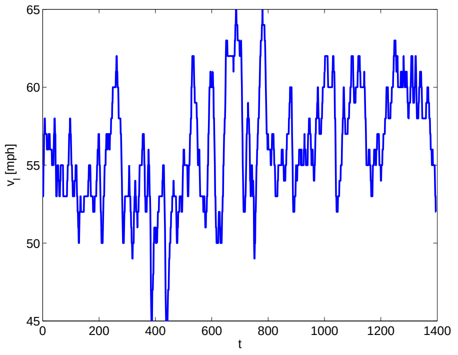
In this section we illustrate the above developments with an example of an intelligent vehicle which uses an adaptive cruise control to follow another, randomly accelerating and decelerating vehicle. In this application, the control objective is to maintain the distance to the lead vehicle within specified limits for as long as possible with only very gradual (small) accelerations and decelerations of the follower vehicle to improve fuel economy and increase driver comfort.

The relative distance between two vehicles minus minimum acceptable distance is denoted by s [m], the velocity of the lead vehicle is denoted by v_l [mph], the velocity of the follower vehicle is denoted by v_f [mph] and ΔT is the sampling time period. Assuming that the acceleration a [mph/sec] of the follower vehicle is a control variable, the discrete-time update equations have the following form,

$$\begin{aligned} s(t+1) &= s(t) + 0.1736 \cdot \Delta T \cdot (v_l(t) - v_f(t)), \\ v_f(t+1) &= v_f(t) + \Delta T \cdot a(t). \end{aligned} \tag{10}$$

The factor 0.1736 is introduced because the velocity units are in miles-per-hour (mph) while the distance is in meters (m). With $x = [s, v_f]^{\mathrm{T}}$, $w = v_l$, and $v = a$ as the control, (10) has the form of (1).

We consider a scenario when the vehicles are driven on a road with average speed of 55 mph, minimum speed of 46 mph and maximum speed of about 66 mph. The update period is fixed to $\Delta T = 1$ sec. The lead vehicle velocity, $w = v_l$, is modeled by a Markov chain with 20 discrete levels uniformly distributed between 46 and 66.0013 mph. The transition probabilities (see Figure 2-right) have been constructed from an experimental vehicle velocity trajectory shown in Figure 2-left.



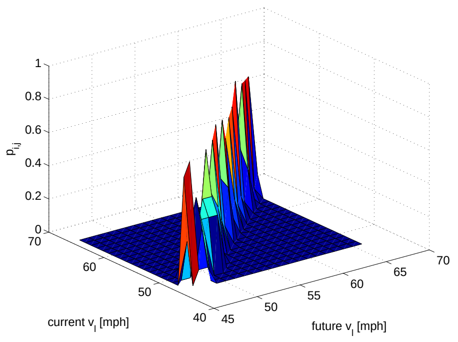


Fig. 2. Left: Experimental vehicle velocity trajectory. Right: Transition probabilities of the Markov chain model of the lead vehicle velocity.

It is desired to maintain the relative distance between two vehicles minus minimum acceptable distance in the range $s \in [0, 20]$ meters. The accelerations of the follower vehicle must be in the range $a \in [-0.5, 0.5]$ mph/sec.

An approximation of the optimal control, u_*, determined using the value iteration approach, is illustrated in Figure 3 while the value function, V_* is illustrated in Figure 4. Note that the u_* and V_* depend on three variables: s, v_f, and v_l. Hence, only the cross-sections of u_* and V_* are shown for a fixed value of v_f. Figure 5 demonstrates numerically the convergence of the value iterations. The grids used were $\{-0.5, -0.25, 0, 0.25, 0.5\}$ for a, $\{46, 47.0527, 48.1054, \cdots, 66.0013\}$ for v_f and v_l, and $\{0, 1.0526, 2.1053, \cdots, 20\}$ for s.

Figures 6 illustrates the time responses when the follower vehicle is controlled with the above approximate stochastic optimal control and when the lead vehicle velocity is a typical realization of the Markov chain trajectory. Note that the accelerations and decelerations of the lead vehicle are up to 2.1 mph/sec, well in excess of 0.5 mph/sec limit imposed on the accelerations and decelerations of the follower vehicle. Figure 7 illustrates the time responses when the lead vehicle velocity is a sequence of non-random accelerations and decelerations.

As can be observed from the plots, the velocity of the follower vehicle, controlled by stochastic drift counteracting optimal control, tracks the velocity of the lead vehicle but with smaller accelerations and decelerations, which satisfy the required limits of 0.5

mph/sec. The controller also enforces the constraints on the relative distance between the vehicles. When the lead vehicle moves at high speed, the follower vehicle increases the relative distance knowing that deceleration of the lead vehicle is more likely and acceleration is less likely. When the lead vehicle moves at low speed, the follower vehicle decreases the relative distance knowing that acceleration of the lead vehicle is more likely and deceleration is less likely. This behavior of the follower vehicle is directionally consistent with the constant headway time policy.

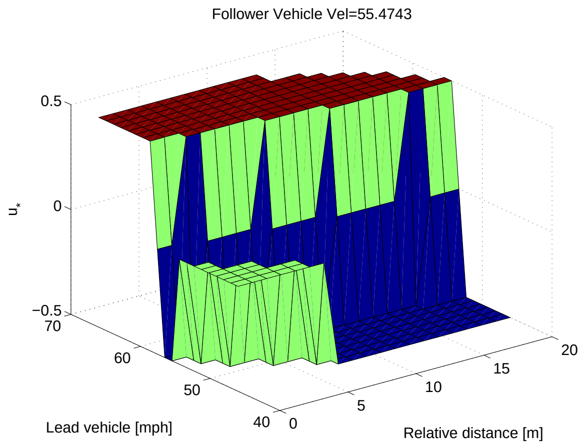


Fig. 3. A cross-section of approximate optimal control.

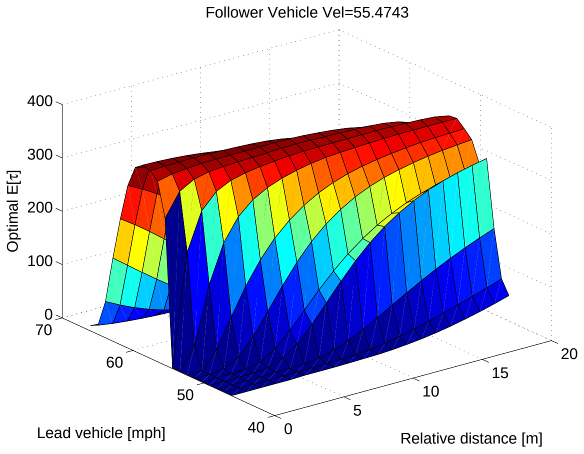


Fig. 4. A cross-section of approximate optimal value function.

Remark 1: The stochastic optimal control maximizes the expected time to violate the constraints, but it cannot entirely eliminate the possibility that the constraints are violated. If the relative distance constraints become violated, a decision needs to be

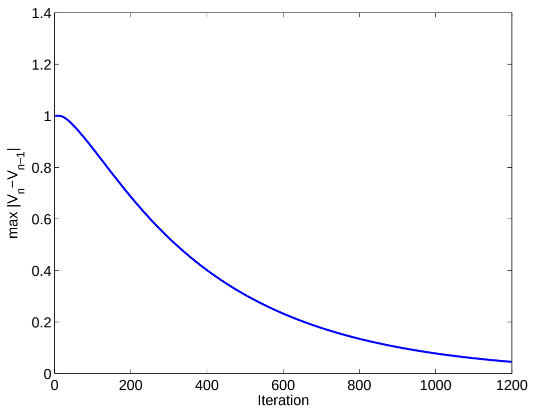


Fig. 5. Maximum of $|V_n(x, w^i) - V_{n-1}(x, w^i)|$ over $x \in \{x^k,\ k \in K\}$ and $i \in J$ as the value iterations progress (i.e., n increases).

made if to discontinue following the lead vehicle since it is too difficult to follow, or to switch to a different controller which may use larger accelerations and decelerations to bring the relative distance and the follower vehicle velocity to values appropriate to re-engage the stochastic optimal controller.

Remark 2: The transition probabilities for the lead vehicle velocity may be estimated on-line by measuring the lead vehicle velocity. Considering that on-board computing power may be limited, fast procedures to approximate u_*, once transition probabilities have been estimated, are desirable. The development of such procedures is a subject of future research.

5 Concluding Remarks

In this paper we presented a method for constructing a stochastic optimal control law in the problem of maximizing expected time to violate constraints for a nonlinear discrete-time system with a measured (but unknown in advance) disturbance modeled by a Markov chain. The resulting control law is referred to as the *stochastic drift counteracting optimal control law*.

A simulation example was considered where an intelligent vehicle follows another, randomly accelerating and decelerating lead vehicle. The control objective in this example was to control the follower vehicle acceleration to maintain the distance to the lead vehicle within specified limits and avoid high accelerations and decelerations so as to improve fuel economy and increase driver comfort. It has been shown that the behavior of the vehicle with the stochastic drift counteracting optimal control law is intuitively reasonable, e.g., the relative distance between the vehicles increases (respectively, decreased) when the lead vehicle is near its maximum (respectively, minimum) speed, as the follower vehicle expects a deceleration (respectively, acceleration) of the lead vehicle.

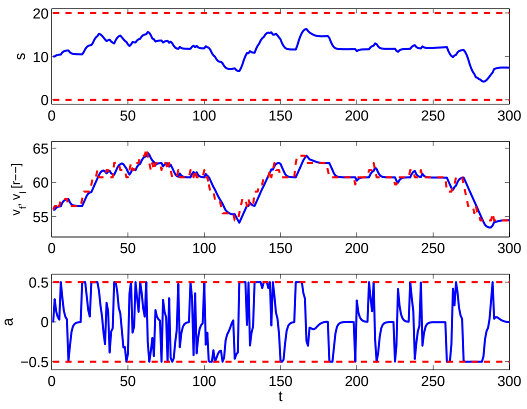


Fig. 6. Relative distance (top), follower and lead vehicle velocities (middle) and follower vehicle acceleration (bottom) in response to random lead vehicle velocity profile. Relative distance constraints and acceleration constraints are indicated on the top plot and bottom plot, respectively, by dashed lines. Dashed lines in the middle plot indicate the lead vehicle velocity.

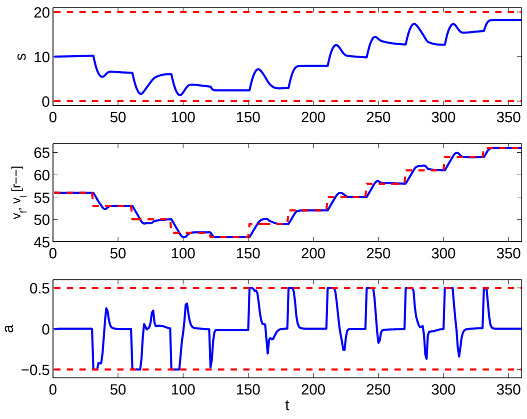


Fig. 7. Relative distance (top), follower and lead vehicle velocities (middle), and follower vehicle acceleration (bottom) in response to non-random lead vehicle velocity profile. Relative distance constraints and acceleration constraints are indicated on the top plot and bottom plot, respectively, by dashed lines. Dashed lines in the middle plot indicate the lead vehicle velocity.

More elaborate vehicle models and lead vehicle speed models can be treated similarly even though, as with any dynamic programming approach, high state dimensions present an obstruction due to "curse of dimensionality." Fast procedures for computing or approximating the stochastic optimal control law, so that it can be reconfigured on-line if the problem parameters or statistical properties of the lead vehicle velocity change, is a subject of future research. While this paper only discussed procedures suitable for off-line computations, these results are already valuable as the resulting stochastic optimal control law can be used as a benchmark for control algorithms developed other approaches, and it can yield valuable insights into the optimal behavior desirable of the follower vehicle. Also, from Figure 3, it appears that u_* does not have a very elaborate form and so it may inspire a simpler rule-based control law which achieves a near optimal performance.

The theoretical results and computational approaches discussed in this paper can have other applications in intelligent vehicle control and manufacturing. Specifically, they may be applicable in other situations where there is a disturbance with statistical properties that can be modeled in advance (e.g., demands of the driver, changes in the environmental conditions, production orders being scheduled, etc.) while pointwise-in-time constraints on certain critical state and control variables need to be enforced. Along these lines, another example application to Hybrid Electric Vehicle (HEV) control has been discussed in [6].

References

1. Afanas'ev, V.N., Kolmanovskii, V.B., and Nosov, V.R. (1996). *Mathematical Theory of Control Systems Design*. Kluwer Academic Publishers.
2. Altman, E. (1999). *Constrained Markov Decision Processes*. Chapman and Hall/CRC.
3. Dynkin, E.B., and Yushkevich, A.A. (1967). *Markov Processes: Theorems and Problems.* Nauka, Moscow, in Russian. English translation published by Plenum, New York, 1969.
4. Kolmanovsky, I., Sivergina, I., and Lygoe, B. (2002). Optimization of powertrain operating policies for feasibility assessment and calibration: Stochastic dynamic programming approach. *Proceedings of 2002 American Control Conference.* Anchorage, AK, pp. pp. 1425–1430.
5. Kolmanovsky, I., and Maizenberg, T.L. (2002). Optimal containment control for a class of stochastic systems perturbed by Poisson and Wiener processes. *IEEE Transactions on Automatic Control.* Vol. 47, No. 12, pp. 2041–2046.
6. Kolmanovsky, I., Lezhnev, L., and Maizenberg, T.L. (2008). Discrete-time drift counteraction stochastic optimal control: Theory and application-motivated examples. *Automatica*, Vol. 44 , No. 1, pp. 177–184.

PAPERS

Distributed Mission and Contingency Management for the DARPA Urban Challenge

Tichakorn Wongpiromsarn and Richard M. Murray

Division of Engineering and Applied Science
California Institute of Technology, Pasadena, CA, U.S.A.
nok@caltech.edu, murray@cds.caltech.edu

Abstract. We present an approach that allows mission and contingency management to be achieved in a distributed and dynamic manner without any central control over multiple software modules. This approach comprises two key elements: a mission management subsystem and a planning subsystem based on a Canonical Software Architecture (CSA). The mission management subsystem works in conjunction with the planning subsystem to dynamically replan in reaction to contingencies. The CSA provides for consistency of the states of all the software modules in the planning subsystem. System faults are identified and replanning strategies are performed distributedly in the planning and the mission management subsystems through the CSA. The approach has been implemented and tested on Alice, an autonomous vehicle developed by the California Institute of Technology for the 2007 DARPA Urban Challenge.

1 Introduction

One of the major challenges in urban autonomous driving is the ability of the system to reason about complex, uncertain, spatio-temporal environments and to make decisions that enable autonomous missions to be accomplished safely and efficiently, with reactive replanning in case of contingencies. Due to the complexity of the system and a wide range of environments in which the system must be able to operate, an unpredictable performance degradation of the system can quickly cause critical system failure. In a distributed system such as Alice, an autonomous vehicle developed by the California Institute of Technology for the 2007 DARPA Urban Challenge, performance degradation of the system may result from changes in the environment, hardware failures, inconsistencies in the states of different software modules, and faulty behaviors of a software module. To ensure safety and mission success, there is a need for the system to be able to properly detect and respond to these unexpected events which affect the vehicle's operational capabilities.

Mission and contingency management is often achieved using a centralized approach where a central module communicates with nearly every software module in the system and directs each module sequentially through its various modes in order to recover from failures. Examples of such a central module are the behavior management module of the TerraMax Autonomous Vehicle [1] and the supervisory controller (SuperCon) module of Alice previously developed for the 2005 DARPA Grand Challenge [2]. A drawback of this approach is that the central module usually has so much functionality and responsibility that it easily becomes unmanageable and error prone as the system gets more complicated. In fact, Team Caltech's failure in the 2005 DARPA Grand Challenge was mainly due to an inability of the SuperCon module to reason and respond properly to certain combinations of faults in the system [2]. This resulted from the difficulty in verifying this module due to its complexity.

The complexity and dynamic nature of the urban driving problem make centralized mission and contingency management impractical. A mission management subsystem and a planning subsystem based on a Canonical Software Architecture (CSA) [3] have therefore been developed to allow mission and contingency management to be achieved in a distributed manner. The mission management subsystem comprising the mission planner, the health monitor and the process control modules works in conjunction with the planning subsystem (the trajectory planner, the follower and the drive control) to dynamically replan in reaction to contingencies. As shown in Figure 1, the health monitor module actively monitors and estimates the health of the hardware and software components to dynamically assess the vehicle's operational capabilities throughout the course of mission. It communicates directly with the mission planner module which replans the mission goals based on the current vehicle's capabilities. The process control module uses the health estimates of individual software modules to automatically restart a software module that quits unexpectedly and a software module that identifies itself as unhealthy. An unhealthy hardware component is power-cycled by the software that communicates with it. The CSA provides for consistency of the states of all the software modules in the planning subsystem. System faults are identified and replanning strategies are performed distributedly in the planning and the mission management subsystems through the CSA directive/response mechanism. Together these mechanisms make the system capable of exhibiting a fail-operational/fail-safe and intelligent responses to a number different types of failures in the system.

Related work includes a holistic contingency management technology [4], a Mission Effectiveness and Safety Assessment (MENSA) technology [5], real-time fault detection and situational awareness [6], the high level controller of the Intelligent Off-Road Navigator [7] and a model-based approach [8]. These approaches rely on having a subsystem, similar to our mission management subsystem, capable of monitoring and assessing unexpected, mission-related events that affect the overall system operation and mission success. This subsystem may also be capable of suggesting a new strategy or operation mode for the planning subsystem or reconfiguring the system in response to these events. The CSA, however, is intended to facilitate these responsibilities of the mission management subsystem. By exploiting the hierarchical structure and integrating the directive/response mechanism into the planning subsystem, the mission management subsystem can assess most of the mission-related events by only reasoning at the level of failure or completion of its directives and the health of the hardware and software components.

The contributions of this paper are: (1) a framework for integrating mission and contingency management into a planning system so that it can be achieved distributedly and dynamically; (2) a complete implementation on an autonomous vehicle system capable of operating in a complex and dynamic environment; and (3) an evaluation of the approach from extensive testing and some insight into future research directions.

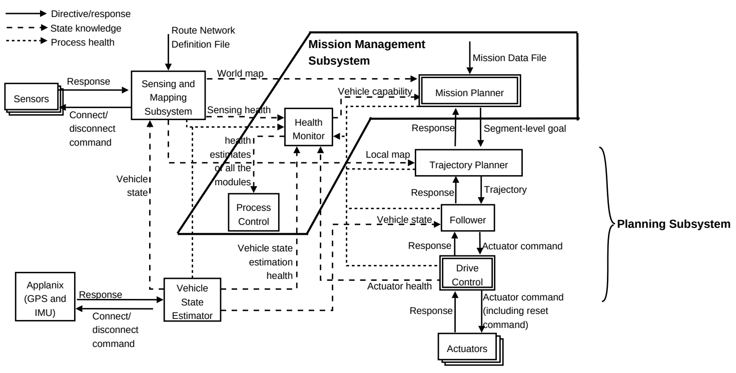


Fig. 1. Alice's mission management and planning subsystems in the Canonical Software Architecture. Boxes with double lined borders are subsystems that will be broken up into multiple CSA modules.

The remainder of this paper is organized as follows. Section 2 introduces the concept of the Canonical Software Architecture. Section 3 describes the mission management subsystem in more detail. Section 4 explains how system faults can be identified and handled distributedly through the CSA. Section 5 presents the results from the 2007 DARPA Urban Challenge's National Qualifying Event and provides a discussion about the advantages and disadvantages of the approach. Section 6 concludes the paper and discusses some future work.

2 Canonical Software Architecture

In many complex systems, the software modules that make up the planning system are responsible for reasoning at different levels of abstraction. Hence, the planning system can be decomposed into a hierarchical framework. A Canonical Software Architecture has been developed to support this decomposition and separation of functionality, while maintaining communication and contingency management. This architecture builds on the state analysis framework developed at the Jet Propulsion Laboratory (JPL) and takes the approach of clearly delineating state estimation and control determination as described in [9], [10], [11] and [12]. To prevent the inconsistency in the states of different software modules due to the inconsistency in the state knowledge, we require that there is only one source of state knowledge although it may be provided in different abstractions for different modules.

There are two types of modules in CSA: estimation modules and control modules. For modularity, each software module in the planning subsystem may be broken down into multiple CSA modules. An example of the planning subsystem in CSA we have implemented on Alice is shown in Figure 1. An estimation module estimates the system state and provides an abstraction of the system state for the corresponding control module(s). A control module gets inputs, performs actions based on the inputs, and delivers outputs. As shown in Figure 2, the inputs consist of state information, directives/instructions (from other modules wishing to control this module) and re-

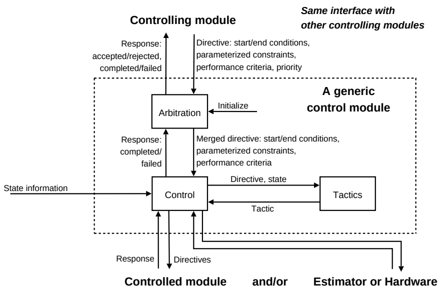

Controlling module
Same interface with other controlling modules
Response: accepted/rejected, completed/failed
Directive: start/end conditions, parameterized constraints, performance criteria, priority
A generic control module
Arbitration
Initialize
Response: completed/ failed
Merged directive: start/end conditions, parameterized constraints, performance criteria
State information
Control
Directive, state
Tactic
Tactics
Response
Directives
Controlled module
and/or
Estimator or Hardware

Fig. 2. A generic control module in the Canonical Software Architecture.

sponses/status reports (from other modules receiving instructions from this module). The outputs are the same type as the inputs, but in the reverse direction (status reports from this module and directives/instructions for other control modules).

For each directive that a control module is designed to accept, the following must be specified: (1) entry condition; (2) exit condition; (3) constraints that must be satisfied during the execution of the directive; and (4) performance criteria (performance or other items to be optimized). The entry and exit conditions define, respectively, what must be true before starting to execute this directive and what must be true to complete the execution of this directive. For each directive received, a response which indicates rejection, acceptance, failure or completion of the directive and the reason for rejection or failure must be reported to the source of the directive. Rejection or failure of a directive occurs when the entry or exit condition is not readily achievable, the deadlines aren't met, or one of the constraints cannot be satisfied.

A CSA module consists of three components: *Arbitration*, *Control* and *Tactics*. It communicates with its neighbors through directives and responses, as shown in Figure 2. *Arbitration* is responsible for (1) managing the overall behavior of the control module by issuing a merged directive, computed from all the received directives, to the *Control*; and (2) reporting failure, rejection, acceptance and completion of a received directive to the *Control* of the issuing control module. We have implemented a simple arbitration scheme, similar to that of the subsumption architecture [13], where the merged directive is simply the received directive with the highest priority. As a future work, one can implement a more complicated arbitration scheme that involves dealing with multiple received directives simultaneously. *Control* is responsible for (1) computing the output directives to the controlled module(s) or the commands to the hardware based on the merged directive, received responses and state information; and (2) reporting failure and completion of a merged directive to the *Arbitration*. *Tactics* provides the core functionality of the control module and is responsible for providing the logic used by the *Control* for computing output directives.

3 Mission Management Subsystem

3.1 Health Monitor and Vehicle Capabilities

The health monitor module is an estimation module that continuously gathers the health of the software and hardware (GPS, sensors and actuators) components of the vehicle and abstracts the information about these devices into a form usable for the mission planner. This form can most easily be thought of as vehicle capability. For example, we may start the mission with perfect functionality, but somewhere along the line lose a right front sensor. The intelligent choice in this situation would be to try to limit the number of left turns at intersection due to the inability to assess oncoming traffic from the right and slow down the vehicle. Another example arises if the vehicle becomes unable to shift into reverse. In this case we would not like to purposely plan paths that require a three-point turn.

From the health of the sensors and sensing modules, the health monitor estimates the sensing coverage. The information about sensing coverage and the health of the GPS unit and actuators allow the health monitor to determine the following vehicle capabilities: (1) turning right at intersection; (2) turning left at intersection; (3) going straight at intersection; (4) nominal driving forward; (5) stopping the vehicle; (6) making a three-point turn; (7) driving in an unstructured region; and (8) navigation in unmapped areas.

3.2 Mission Planner

The mission planner module receives a Mission Data File (MDF) that is loaded before each mission, vehicle capabilities from the health monitor module, position of obstacles from the mapper module and status reports from the trajectory planner module and sends segment-level goals to the trajectory planner module. A segment-level goal specifies the road/zone Alice has to navigate and the constraints, represented by the type of segment (road, zone, off-road, intersection, U-turn, pause, backup, end of mission) which basically defines a set of traffic rules to be imposed during the execution of this goal.

The mission planner is broken up into one estimation and two CSA control modules: the traversibility graph estimator, the mission control and the route planner. The mission control module has three main functions: (1) computing mission goals which specify how Alice will satisfy the mission specified in the MDF; (2) based on the vehicle capabilities, determining conditions (including the maximum speed) under which we can safely continue the mission; and (3) detecting the lack of forward progress and replanning the mission goals accordingly. The route planner module determines segment-level goals to satisfy the mission goals based on the traversibility graph which represents the connectivity of the route network and is determined by the traversibility graph estimator module. Since vehicle capabilities are also taken into account in the determination of the mission goals and the traversibility graph, for example, if the capability for making a left turn decreases due to the failure of the right front sensor, the route involving the least number of these maneuvers will be preferred or if the vehicle is not able to shift into reverse, routes that require a three-point turn will be avoided.

4 Fault Handling in the Planning Subsystem

In the CSA framework, fault handling is embedded into all the modules and their communication interfaces in the planning subsystem hierarchy. Each module has a set of different control strategies which allow it to identify and resolve faults in its domain and certain types of failures propagated from below. If all the possible strategies fail, the failure will be propagated up the hierarchy along with the associated reason. The next module in the hierarchy will then attempt to resolve the failure. This approach allows each module to be isolated so it can be tested and verified much more fully for robustness.

Trajectory Planner. The trajectory planner accepts directives from the mission planner module and generates trajectories for Alice to follow. It comprises four components: the logic planner, the path planner, the velocity planner and the predictor. The logic planner guides the vehicle at a high level by determining the current situation and coming up with an appropriate planning problem (or strategy) to solve. The path planner is responsible for finding a feasible path, subject to the constraints imposed by the planning problem. If such a path cannot be found, an error will be generated. Since Alice needs to operate in both structured and unstructured regions, we have developed three types of path planner to exploit the structure of the environment: the *rail planner* (for structured regions such as roads, intersections, etc), the *off-road rail planner* (for obstacle fields and sparse waypoint regions) and the *clothoid planner* (for parking lots and obstacle fields). All the maneuvers available to the *rail planner* are pre-computed; thus, the *rail planner* may be too constraining. To avoid a situation where Alice gets stuck in a structured region (e.g. when there is an obstacle between the predefined maneuvers), the *off-road rail planner* or the *clothoid planner* may also be used in a structured region. This decision is made by the logic planner. The velocity planner takes the path from the path planner and the planning problem from the logic planner and generates a time parameterized path, or trajectory. The predictor is responsible for predicting the future location and behavior of other vehicles.

The logic planner is responsible for fault handling inside the trajectory planner. Based on the error from the path planner and the follower, the logic planner specifies a different planning problem such as allowing passing or reversing, using the *off-road rail planner*, or reducing the allowable distance from obstacles. The logic for dealing with these failures can be described by a two-level finite state machine (FSM). First, the high-level mode (road region, zone region, off-road, intersection, U-turn, failed and paused) is determined based on the directive from the mission planner and the current position. Each of the high-level modes can be further decomposed to completely specify the planning problem described by the drive state, the allowable maneuvers, and the allowable distance from obstacles.

– **Road Region, Zone Region and Off-Road.** The logic planner transitions to the road region, zone region or off-road mode when the type of segment specified by the mission planner is road, zone or off-road, respectively. The modes and transitions for the road region mode are shown in Figure 3. In the zone region and the off-road modes, passing and reversing are allowed by default. For the zone region mode, the *clothoid planner* is the default path planner and the trajectory is planned such that Alice will stop at the right distance from the closest obstacle, so the only decision that needs to be made by the logic planner is the allowable distance from obstacles For the off-road mode, the drive state (drive or stop) also needs to be determined. As a result, only three and six modes are necessary within the zone region mode

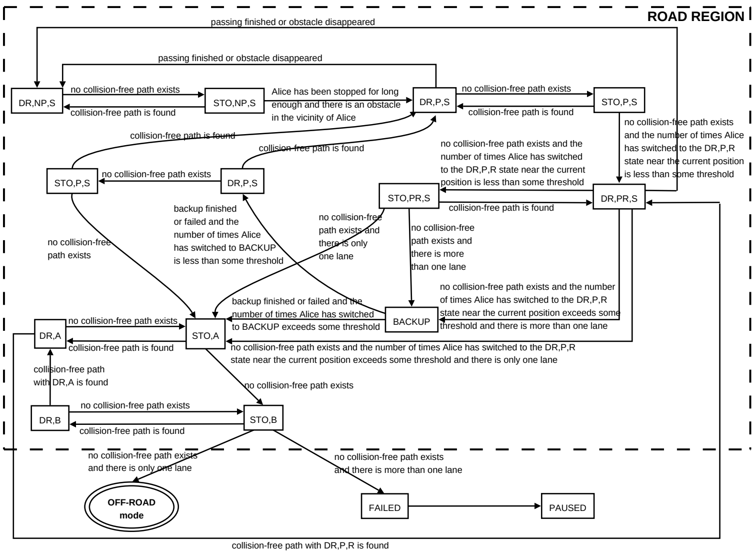

ROAD REGION
passing finished or obstacle disappeared
passing finished or obstacle disappeared
DR,NP,S
no collision-free path exists
collision-free path is found
STO,NP,S
Alice has been stopped for long enough and there is an obstacle in the vicinity of Alice
DR,P,S
no collision-free path exists
collision-free path is found
STO,P,S
no collision-free path exists and the number of times Alice has switched to the DR,P,R state near the current position is less than some threshold
collision-free path is found
collision-free path is found
no collision-free path exists and the number of times Alice has switched to the DR,P,R state near the current position is less than some threshold
STO,P,S
no collision-free path exists
DR,P,S
STO,PR,S
collision-free path is found
DR,PR,S
backup finished or failed and the number of times Alice has switched to BACKUP is less than some threshold
no collision-free path exists and there is only one lane
no collision-free path exists and there is more than one lane
no collision-free path exists
no collision-free path exists and the number of times Alice has switched to the DR,P,R state near the current position exceeds some threshold and there is more than one lane
backup finished or failed and the number of times Alice has switched to BACKUP exceeds some threshold
BACKUP
DR,A
no collision-free path exists
collision-free path is found
STO,A
no collision-free path exists and the number of times Alice has switched to the DR,P,R state near the current position exceeds some threshold and there is only one lane
collision-free path with DR,A is found
no collision-free path exists
DR,B
no collision-free path exists
collision-free path is found
STO,B
no collision-free path exists and there is only one lane
no collision-free path exists and there is more than one lane
OFF-ROAD mode
FAILED
PAUSED
collision-free path with DR,P,R is found

Fig. 3. The logic planner FSM for the road region. Each mode defines the drive state (DR $\equiv$ drive, BACKUP $\equiv$ reverse, and STO $\equiv$ stop for obstacles), the allowable maneuvers (NP $\equiv$ no passing or reversing allowed, P $\equiv$ passing allowed but reversing not allowed, PR $\equiv$ both passing and reversing allowed), and the minimum allowable distance from obstacles (S $\equiv$ *safe or nominal*, A $\equiv$ *aggressive*, and B $\equiv$ *bare or very aggressive*).

and the off-road mode, respectively. The transitions can be easily deduced from those shown in Figure 3.

- **Intersection.** The logic planner transitions to the intersection mode when Alice approaches an intersection. Passing and reversing maneuvers are not allowed and the trajectory is planned such that Alice stops at the stop line. Once Alice is within a certain distance from the stop line and is stopped, the intersection handler, an FSM comprising five modes (reset, wait for precedence, wait for merging, wait for the intersection to clear, jammed intersection, and go), will be reset and start checking for precedence [14]. The logic planner transitions out of the intersection mode when the intersection handler transitions to the go or jammed intersection mode. If the intersection is jammed, the logic planner will transition to the mode where passing is allowed.
- **U-turn.** The logic planner transitions to the U-turn mode when the type of segment specified by the mission planner is U-turn. Once the U-turn is completed, the logic planner will transition to the paused mode and wait for the next directive.
- **Failed.** The logic planner transitions to the failed mode when all the strategies in the current high-level mode have been tried. In this mode, failure is reported to the mission planner. The logic planner then transitions to the paused mode. The mission planner will then replan and send a new directive such as making a U-turn, switching to the off-road mode, or backing up in order to allow the route planner to change the route. As a result, the logic planner will transition to a different high-

level mode. These mechanisms ensure that Alice will keep moving as long as it is safe to do so.

– **Paused.** The logic planner transitions to the paused mode when it does not have any segment-level goals or when the type of segment specified by the mission planner is pause or end of mission. In this mode, the logic planner is reset and the trajectory is planned such that Alice comes to a complete stop as soon as possible.

Follower. The follower module computes actuation commands that keep Alice on the reference trajectory [15]. Although these trajectories are guaranteed to be collision-free, since Alice cannot track them perfectly, she may get too close or even collide with an obstacle if the tracking error is too large. To address this issue, we allow the follower to request a replan from the trajectory planner through the CSA directive/response mechanism when the deviation from the reference trajectory is too large. In addition, we have implemented a reactive obstacle avoidance (ROA) component to deal with unexpected obstacles. The ROA component can override the acceleration command if the projected position of Alice collides with an obstacle. The projection distance depends on the velocity of Alice. The follower will report failure to the trajectory planner if the ROA is triggered, in which case the trajectory planner can replan the trajectory.

Drive Control. The drive control module is the overall driving software for Alice. It receives actuation commands from the follower, determines if they can be executed and, if so, sends the appropriate commands to the actuators. The drive control module also performs checking on the health and operational state of the actuators, resets the actuators that fail, and broadcasts the actuator state. Also included in the role of the drive control module is the implementation of physical protections for the hardware to prevent the vehicle from hurting itself. This includes three functions: limiting the steering rate at low speeds, preventing shifting from occurring while the vehicle is moving, and transitioning to the paused mode in which the brakes are depressed and commands to any actuator are rejected when any of the critical actuators such as steering and brake fail.

5 Results and Discussion

The 2007 DARPA Urban Challenge's National Qualifying Event was split into three test areas, featuring different challenges. In this section, we present the results from Test Area B which was the most challenging test area from the mission and contingency management standpoint. Test Area B consisted of approximately 2 miles of driving, including a narrow start chute, a traffic circle, narrow, winding roads, a road with cars on each side that have to be avoided and an unstructured region with an opening in a fence, navigating and parking at a designated spot in an almost fully occupied parking lot.

In our first attempt, a reasonably conservative vehicle separation distance was used. As shown in Figure 4(a), the logic planner spent a considerable amount of time in the *aggressive* and *bare* modes where the allowable distance from obstacles is reduced. Given the size of Alice, the second largest vehicle in the competition, she had difficulties finishing this course mainly due to the vehicle separation distance problem which

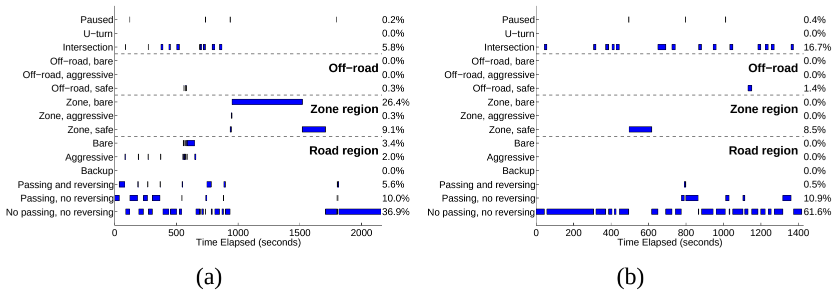


Fig. 4. The logic planner mode during NQE Test Area B (a) run #1 and (b) run #2.

caused her to spend about five minutes trying to get out of the start chute area and more than ten minutes trying to park correctly while keeping the required distance from obstacles. Specifically, the problem was that in the start chute area, there were K-rails less than one meter away from each side of Alice, resulting in a violation of the obstacle clearance requirement for the *safe* or nominal mode, which was set in accordance with the DARPA rules. Alice had to progress through a series of internal planning failures before finally driving with reduced buffers on each side of the vehicle. In the parking lot, there was a car parked right in front of our designated spot and if Alice was to park correctly, she would have to be within two meters of that car; thus, violating the obstacle clearance requirement. Alice ran out of the thirty minute time limit shortly after we manually moved her out of the parking lot.

After the first run, we decided to decrease the required vehicle separation distance and relax the tolerance of reaching waypoints so Alice could complete the course faster. Alice was then able to successfully complete the course within twenty three minutes with only minor errors. The logic planner mode during the second attempt is shown in Figure 4(b).

Despite the failure in completing the first run within the time limit, Alice demonstrated the desired behavior, consistent with what we have seen in over two hundred miles of extensive testing, that she would keep trying different strategies to get closer to completing the mission and she would never stop as long as the system is capable of operating safely. Had she been given more time, the mission control would have detected the lack of forward progress and decided to skip the parking and continue to complete the rest of the mission.

Compared to a centralized approach, our approach to mission and contingency management is a lot more modular. It allows independent development and testing of failure handling in different software modules, which is important for a project with a short development period and a large development team. Most of the bugs can be found at the stage of module test, instead of system integration test. Using different levels of abstraction, our approach greatly simplifies the logic for dealing with failures and makes it easier to identify all the combinations of failures in the system. A drawback of this approach is that all the interfaces need to be clearly defined; thus, it requires putting a substantial amount of effort in the design phase of the project.

6 Conclusions and Future Work

We described Team Caltech's approach to mission and contingency management for the 2007 DARPA Urban Challenge. This approach allows mission and contingency management to be accomplished in a distributed and dynamic manner. It comprises two key elements: a mission management subsystem and a planning subsystem based on a Canonical Software Architecture (CSA). The mission management subsystem works in conjunction with the planning subsystem to dynamically replan in reaction to contingencies. The CSA provides for consistency of the states of all the software modules in the planning subsystem. System faults are identified and replanning strategies are performed distributedly in the planning subsystem through the CSA. These mechanisms make the system capable of exhibiting a fail-operational/fail-safe and intelligent responses to a number different types of failures in the system. Extensive testing has demonstrated the desired behavior of the system which is that it will keep trying different strategies in order to get closer to completing the mission and never stop as long as it is capable of operating safely.

Extensions of this work include extending the CSA to the estimation side of the system. Incorporating the notion of uncertainty in the CSA directive/response mechanism is also important. Consider a scenario where spurious obstacles are seen such that they completely block the road. Although the map may correctly reflect high uncertainty, the logic planner will still progress through all its modes before finally concluding that it cannot complete the segment-level goal. Failure will then be reported to the mission planner which will incorrectly evaluate the current situation as the road is completely blocked and subsequently plan a U-turn. If the response also incorporates the notion of uncertainty, the mission planner can use this information together with the system health and issue a pause directive instead so Alice will stop and wait for better accuracy of the map.

Another direction of research is to formally verify that if implemented correctly, the directive/response mechanism will ensure the consistency of the states of all the software modules in the system and that the CSA and the mission management subsystem guarantee that Alice will keep going as long as it is safe to do so. Using temporal logic, we were able to formally verify the state consistency for the follower and drive control modules. For the rest of the system, we have only verified the state consistency and the fail-operational/fail-safe capability through extensive testing.

Lastly, it is also of interest to verify that this distributed mission and contingency management approach actually captures all the functionality of a centralized approach such as SuperCon and that it actually facilitates formal verification of the system. We believe that this is the case for many systems in which the central module does not take into account the uncertainties in the system and the environment.

Acknowledgements

The idea of the CSA came from discussions with Robert Rasmussen and Michel Ingham and was implemented by Josh Doubleday. The health monitor module was developed by Chris Schantz. The following individuals have contributed to the development of the

planning subsystem: Joel Burdick, Vanessa Carson, Stefano Di Cairano, Noel duToit, Sven Gowal, Andrew Howard, Magnus Linderoth, Christian Looman, Kenny Oslund, Kristian Soltesz. Special thanks go to the members of Team Caltech without whose contributions this work would not have been possible.
This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract HR0011-06-C-0146, the California Institute of Technology, Big Dog Ventures, Northrop Grumman Corporation, Mohr Davidow Ventures and Applanix Inc.

References

1. Braid, D., Broggi, A., Schmiedel, G.: The Terramax autonomous vehicle. Journal of Field Robotics 23, (2006) 693–708
2. Cremean, L.B., Foote, T.B., Gillula, J.H., Hines, G.H., Kogan, D., Kriechbaum, K.L., Lamb, J.C., Leibs, J., Lindzey, L., Rasmussen, C.E., Stewart, A.D., Burdick, J.W., Murray, R.M.: Alice: An information-rich autonomous vehicle for high-speed desert navigation. Journal of Field Robotics 23 (2006) 777–810
3. Rasmussen, R.D., Ingham, M.D. personal communication (2006)
4. Franke, J., Hughes, A., Jameson, S.: Holistic contingency management for autonomous unmanned systems. In: Proceedings of the AUVSI's Unmanned Systems North America. (2006)
5. Franke, J., Satterfield, B., Czajkowski, M., Jameson, S.: Self-awareness for vehicle safety and mission success. In: Unmanned Vehicle System Technology, Brussels, Belgium (2002)
6. Dearden, R., Hutter, F., Simmons, R., Thrun, S., Verma, V., Willeke, T.: Real-time fault detection and situational awareness for rovers: Report on the mars technology program task. In: Proceedings of the IEEE Aerospace Conference, Big Sky, MT (2004)
7. Chen, Q., Ümit Özgüner: Intelligent off-road navigation algorithms and strategies of team desert buckeyes in the DARPA Grand Challenge 2005. Journal of Field Robotics 23, (2006) 729–743
8. Williams, B.C., Ingham, M.D., Chung, S.H., Elliott, P.H.: Model-based programming of intelligent embedded systems and robotic space explorers. In: Proceedings of the IEEE: Special Issue on Modeling and Design of Embedded Software. Volume 9. (2003) 212–237
9. Dvorak, D., Rasmussen, R.D., Reeves, G., Sacks, A.: Software architecture themes in JPL's mission data system. In: Proceedings of 2000 IEEE Aerospace Conference. (2000)
10. Rasmussen, R.D.: Goal based fault tolerance for space systems using the mission data system. In: Proceedings of the 2001 IEEE Aerospace Conference. (2001)
11. Barrett, A., Knight, R., Morris, R., Rasmussen, R.: Mission planning and execution within the mission data system. In: Proceedings of the International Workshop on Planning and Scheduling for Space. (2004)
12. Ingham, M., Rasmussen, R., Bennett, M., Moncada, A.: Engineering complex embedded systems with state analysis and the mission data system. J. Aerospace Computing, Information and Communication 2, (2005)
13. Jones, J.L., Roth, D.: 4. In: Robot Programming: A Practical Guide to Behavior-Based Robotics. McGraw-Hill (2004)
14. Looman, C.: Handling of dynamic obstacles in autonomous vehicles. Master's thesis, Universität Stuttgart (2007)
15. Linderoth, M., Soltesz, K., Murray, R.M.: Nonlinear lateral control strategy for nonholonomic vehicles. In: Proceedings of the American Control Conference. (2008) Submitted.

Situational Reasoning for Road Driving in an Urban Environment

Noel E. Du Toit, Tichakorn Wongpiromsarn, Joel W. Burdick and Richard M. Murray

California Institute of Technology, Division of Engineering and Applied Science
1200 E. California Blvd, Pasadena, CA 91125, U.S.A.
{ndutoit, jwb, nok}@caltech.edu, murray@cds.caltech.edu

Abstract. Robot navigation in urban environments requires situational reasoning. Given the complexity of the environment and the behavior specified by traffic rules, it is necessary to recognize the current situation to impose the correct traffic rules. In an attempt to manage the complexity of the situational reasoning subsystem, this paper describes a finite state machine model to govern the situational reasoning process. The logic state machine and its interaction with the planning system are discussed. The approach was implemented on Alice, Team Caltech's entry into the 2007 DARPA Urban Challenge. Results from the qualifying rounds are discussed. The approach is validated and the shortcomings of the implementation are identified.

1 Introduction

The problem of robot navigation in urban environments has recently received substantial attention with the launch of the DARPA Urban Challenge (DUC). In this competition, robots were required to navigate in a fully autonomous manner through a partially known environment populated with static obstacles, live traffic, and other robots. In order for the robot to complete this challenge, it needed to drive on urban roads, navigate intersections, navigate parking lots, drive in unstructured regions, and even navigate unstructured obstacle fields. Since the environment was only partially known prior to the race, the robot needed to rely on sensory information to extract the world state, which

Fig. 1. Alice (left), Team Caltech's (right) entry in the 2007 DARPA Urban Challenge.

introduces additional uncertainty into the problem. Furthermore, lack of exact knowledge about the robot's location and the state and intent of dynamic obstacles introduced further uncertainty. Lastly, the robot needed to obey California traffic rules or exhibit human-like behavior when this was not possible.

The urban component of the problem had two effects on the robotic planning problem: first it introduced some structure into the environment that could be used during the planning process. Second, the traffic rules associated with urban driving forced the robots to exhibit specific behaviors in specific situations. These behaviors are at a high level associated with the driving task that is being executed, which include, for example, driving on a road versus driving in a parking lot. While executing a driving task, it is necessary for the vehicle's control system to reason about which traffic rules are applicable at each instant. It was not sufficient to obey all the rules all the time, but in some cases constraints needed to be relaxed for the robot to make forward progress. This reasoning module is what is presented in this work. A related aspect of urban driving is intersection handling [1] and is not discussed here.

Prior work has attempted to solve the problem of reasoning about the robot's correct driving behavior. Most of the work has been related to highway driving, and deciding when a maneuver such as a lane change or emergency maneuver is in order [2–5]. One practical hurdle is managing the complexity of the decision-making module [2] which must decide which rules to enforce and which actions to take. Another problem is taking uncertainty about the situation into account. Sukthankar et al. [2] implemented a scheme based on a voting system, called polySAPIENT. Different traffic objects in the environment (for example another car, an exit on the highway, etc.) would vote for the appropriate action. Using a mitigation scheme, the best action was chosen. Unsal et al. [3] used automata theory for longitudinal and lateral control of the vehicle, and implicitly chooses the best action. Gregor and Dickmanns [4] used a finite state machine (FSM) to decide. Niehaus and Stengel [5] explicitly account for uncertainty in a probabilistic fashion, and use a heuristic method to select the best action.

The main contribution of this paper is the design of a decision module for a robot navigating an urban environment. To manage complexity, this module does not attempt to explicitly reason about all aspects of the environment, but instead makes use of information generated by the path-planning module to guide decisions. The decision module was implemented on Alice, the Team Caltech entry into the DUC (see figure 1). Results obtained during successful DUC qualifying runs are presented. The paper is structured as follows: the overall planning approach is briefly reviewed in section 2, before focussing on reasoning in the logic planner (section 3). An example is given to illustrate usage. Lastly, some results from the qualifying runs for the DUC are presented, with a discussion, recommendations and future work.

2 Overview of Planning Approach

The planning problem involved three driving tasks: road driving, off-road driving, and parking lot navigation. In an attempt to modularize the system for rapid development, the problems of sensing, planning, and control were separated. The planning problem itself was divided into three layers (see figure 2), following the hierarchical architec-

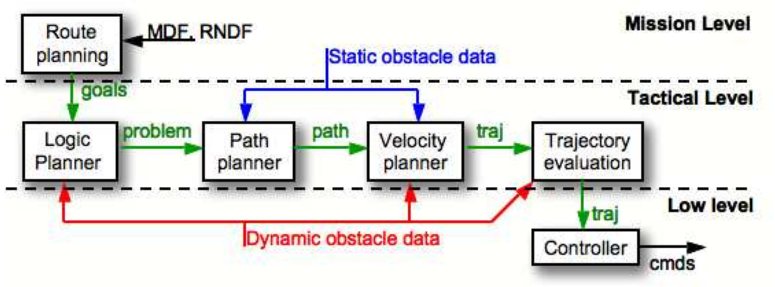


Fig. 2. Planning architecture showing 3 layers used for planning process.

ture dictated by the contingency management approach that was adopted for overall management of Alice's activities [6].

At the mission level, it was necessary to generate a route through the road network, as defined by DARPA through the Road Network Definition File (RNDF). The route planner would specify a sequence of road segment goals to be completed, which would be passed to the tactical planning layer. The tactical planner was responsible for generating a trajectory to some intermediate goal (e.g., a position at the end of a road segment). The reasoning methodology used by the tactical planner is the focus of this paper. The trajectory generated in the tactical planner was in turn passed to a low-level trajectory-following controller, which is documented in [7].

The tactical planner consisted of four parts:

Logic Planner: The logic planner was the reasoning module of the robot. This module had two functions: reasoning about the current traffic situation, and reasoning about intersections [1]. This planner was implemented as a set of finite state machines (FSMs) and would set up a planning problem to be solved. Reasoning about the current traffic state is the focus of this paper.

Path Planner and Velocity Planner: The trajectory planning problem was separated into a spatial and a temporal planning problem in order to simplify these planning problems, and to satisfy the real-time requirement of the planner. Separate path planners were implemented for the three different driving tasks. These path planners were responsible for solving the 2-D spatial path planning problem, accounting for the static component of the environment. The velocity planner time-parameterized the path to obtain a detailed trajectory. This velocity planner adjusted the robot's speed for stop lines along the path, static obstacles on or near the path, the curvature of the path, and the velocities of dynamic obstacles.

Trajectory Analysis (and Prediction): Navigating in urban environments requires the incorporation of the (predicted) future states of dynamic obstacles in the planning problem. Prediction involves two estimation processes: predicting the behavior of the dynamic obstacle, and predicting the future states of the dynamic obstacle. This information can be compared to the robot's planned trajectory to detect future collisions. The generation and use of prediction information will be presented elsewhere.

An important part of the navigation problem was contingency management and internal fault handling. The hierarchical planning architecture lined up well with the contigency management philosophy that was adopted. The Canonical Software Architecture

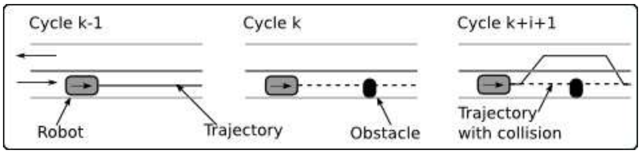


Fig. 3. Example problem: the travel lane of the robot is blocked. Using the failure of the path planner, the logic planner infers that the lane is blocked and relaxes the lane keeping constraint. This allows the robot to execute a passing maneuver.

was adopted, where each module handled its internal faults and the failures propagated from the lower-level modules. When the module could not reach its goal, it would fail to the level above, which would adjust the goal. A complimentary, detailed discussion of contigency management has been presented in [6].

3 Situational Reasoning with the Logic Planner

Situational reasoning is necessary to impose both the traffic rules, and the correct behavior when rules need to be relaxed. For the highway driving case, the environment is very structured, and the behavior of the other dynamic agents that might be encountered by the vehicle is relatively constrained, yet the complexity of the reasoning modules was a problem. One reason for this complexity is because these modules attempt to reason about all components of the environment abstractly. For example, the reasoning module would need to obtain a list of obstacles in the robot's vicinity, and reason about their position (e.g., in lane) in the environment, the context (e.g., static obstacle blocking the lane) and how that may affect the robot (e.g., need to change lanes). Alternatively, much information is obtainable from the path and velocity planners, and could be used to guide the decision process. For example, when the path planner could not find a collision-free path, an obstacle must be blocking the lane. This information could be returned to the reasoning module via a status message, SM, to be used in the decision making. Decision making was avoided while things were running smoothly. For further simplicity, the reasoning module was reduced to a finite state machine (FSM).

Example: To understand the reasoning approach, it is useful to look at an example (see figures 2 and 3). Consider the case of the robot driving down a two-lane, two-way road segment.

Cycle k-1: From the previous planning cycle, no problem was detected by any component of the tactical planner. Imagine now that a static obstacle is detected in the robot's driving lane.

Cycle k: The path planner cannot find a collision-free path that stays in the lane and reaches the goal location. The planner reports the status: $SM = COLLPATH$, and encodes the position of the obstacle in the path structure. From SM, the velocity planner observes the obstacle and plans to bring the vehicle to a stop.

Cycle k+1: The logic planner evaluates SM, and observes that the path contains a collision with a static obstacle. Given the current constraint to stay in the lane, the goal cannot be reached and the lane must be blocked. The appropriate behavior for the robot

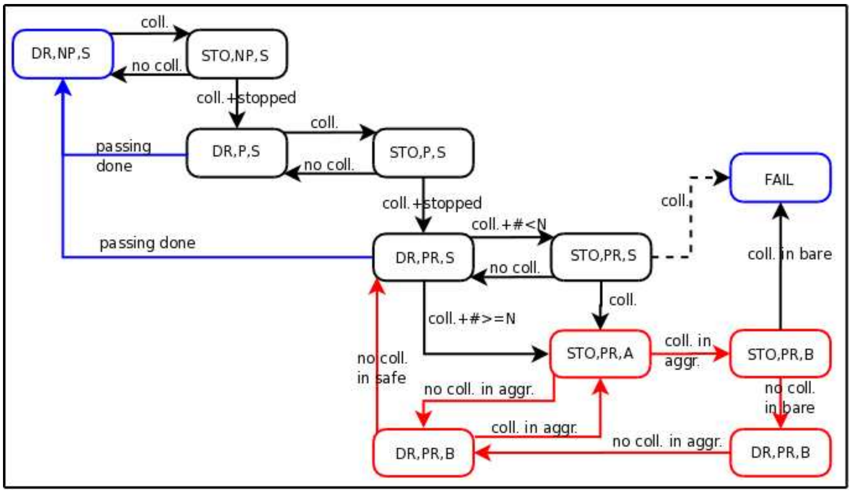


Fig. 4. The logic planner finite state machine for driving in a road region.

would be to drive up to the obstacle and come to a complete stop. Now jump i cycles ahead, to where the robot is stopped.

Cycle k+i+1: Once the robot is stopped, the reasoning module relaxes the constraint to stay in the lane. The path planner searches the adjacent lane. No collision is reported for this new planning problem and the robot is allowed to pass.

Logic Planner: The logic planner was implemented as a finite state machine. For the road navigation, the machine consisted of 10 states denoted by ([M,F,C]). The states constisted of a mode (M), a flag (F), and an obstacle clearance requirement (C). The state machine is illustrated in figure 4. During urban navigation, the robot must interact with static and dynamic obstacles. For planning, the static obstacles required an adjustment of the spatial plan, where as dynamic obstacle required an adjustment of the robot's velocity. Separating the spatial and velocity planning and encoding the dynamic obstacle information on the path, the velocity planner alone could account for the nominal interaction with the dynamic obstacle (such as car following), and the logic did not explicitly have to deal with this problem.

The modes included driving (DR) and stopping for obstacles (STO). The flags included no-passing (NP), passing without reversing (P), and passing with reversing (PR). The obstacle clearance-modes included the nominal, or safe, mode (S), an aggressive mode (A), and a very aggressive, or bare, mode (B). The state machine can be divided into trying to handle the obstacle while maintaining the nominal clearance ([·,·,S]), and being more aggressive. The second option was only invoked when the first failed, and safe operation was guarenteed by limiting the robot speed in these aggressive modes.

The nominal state for road driving, [DR,NP,S], was to allow no passing, no reversing, and the nominal obstacle clearance, termed safety mode. With no obstacles blocking the desired lane, the logic state remained unchanged. When a static obstacle was detected, the path planner would: (i) find a path around the obstacle while staying in lane, (ii) change lanes to another legal lane (if available), or (iii) report a path with a collision. For case (iii), the logic planner would know that a collision free path was not available from the status message (SM), and would switch into obstacle handling mode.

The correct behavior when dealing with a static obstacle was to drive up to it, coming to a controlled stop [STO,NP,S] (refer to figure 4). If at any time the obstacle disappeared, the logic would switch back to the appropriate driving mode. Once the robot was at rest, the logic switched to driving mode, while allowing passing into oncoming lanes of traffic [DR,P,S]. If a collision free path was obtained, then the robot would pass the obstacle and switch back to the nominal driving state once the obstacle had been cleared. If a collision free path did not exist, then the logic would again make sure that the robot was stationary before continuing [STO,P,S]. At this point, either (i) the robot was too close to the obstacle, (ii) there was a partial block and by reducing the obstacle clearances the robot might squeeze by, or (iii) the road was fully blocked. The first case was considered by switching into a mode where both passing and reversing was allowed [DR,PR,S]. If a collision free path was found, the passing maneuver was performed. If a collision was detected and persisted, the robot would again be stopped [STO,PR,S]. At this point, reducing the obstacle clearance and proceeding with caution was considered.

Given the size of the robot (the second largest robot in the 2007 DUC), a major concern was maneuvering in close proximity to static obstacles. To curb this problem, it was desirable to reduce the required obstacle clearances. First the robot switched to aggressive mode, [STO,PR,A]. If a collision free path was found, the robot would drive in this mode [DR,PR,A]. As soon as a path was found that satisfied the nominal obstacle clearance, the logic switched back to [DR,PR,S]. If the robot could not find a collision free path while in aggressive mode, it would reduce the obstacle clearances even further by switching to bare mode [STO,PR,B]. If a path was found, it would drive in this mode [DR,PR,B] until a path was found that did not require this mode. The logic would then switch back to the aggressive mode [DR,PR,A]. If no collision free path could be found, even in bare mode, the conclusion was that the road must be blocked. At this point, the tactical planner could not complete the segment-level goal. In accordance with the contigency management strategy, the tactical planner sent a failure to the route planner, which replanned the route. If the robot was on a one-way road, the route-planner would allow the robot to enter off-road mode, as a last resort.

Since reversing was allowed, it was possible for the robot to get stuck in a cycle of not finding a path [STO,PR,S], then backing up and finding a path [DR,PR,S], driving forward and detecting a collision, backing up again, etc. In an attempt to avoid this cycle and others like it, some transitions were created to exit these loops (from [DR,PR,S]) as part of contigency management.

4 Results and Discussion

The tactical planner, and logic planner, was implemented on Alice, a modified Ford E350 van (see figure 1). The robot was equipped with 24 CPUs, 10 LADAR units, 5 stereo camera pairs, 2 radar units, and an Applanix INS to maintain an estimate of its global position.

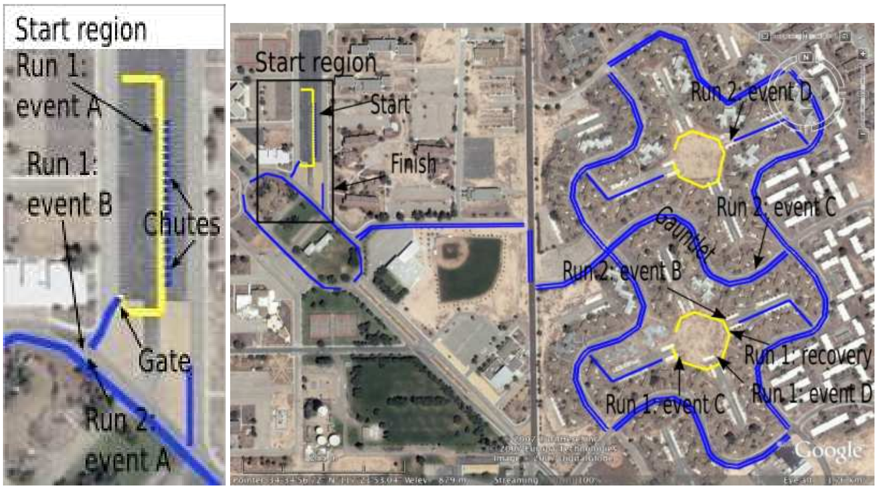


Fig. 5. RNDF and aerial image of Area B.

The NQE consisted of three test areas, which tested different aspects of urban driving. The course of interest here is area B, for which the RNDF, overlayed with aerial imagery, is given in figure 5. The course consisted of approximately 2 miles of urban driving without live traffic and tested the robot's ability to drive on roads, in parking areas, and in obstacle fields. The course was riddled with static obstacles. The robots started in the starting chutes, which were short lane segments, lined with rails. The robot would drive into an open area and proceed to a gate. The gate led to a one-way road, lined with rails, which in turn led to an intersection and the course. The robots would then proceed around a traffic circle and make its way to the parking zone (southern octagonal region). Once through this parking lot, the robot passed through the 'gauntlet', and made its way to the northern zone (obstacle field). From there, it would make its way back to the finish (next to the start area). The results are presented next, followed by a discussion.

4.1 Run 1

The logic states and velocity profile for run 1 are presented in figures 6 and 7, respectively. Four events are indicated on these figures, and the corresponding locations are shown in figure 5. The robot had difficulty exiting the start area (events A and B), but made rapid progress before getting stuck in the parking lot (event C) and was manually reset (event D). It still could not exit the parking area and was eventually recovered from the parking area.

The robot was in the nominal driving state ([DR,NP,S]) only 29.5% of the run (see figure 6). Since the robot got stuck in the parking area (event C and onwards) and ended up spending 34.3% of the run there, it is more useful to consider the logic data up to event C. The robot spend 44.3% of the run up to event C in the nominal driving state, which was still a low number. The logic switched out of the nominal state 8 times to deal with obstacles, of which 5 were in the start area. 42.8% on the run (up to the parking

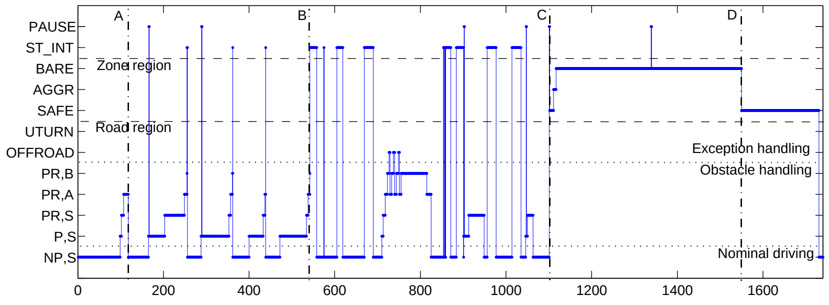


Fig. 6. Logic planner states during run 1 of NQE area B.

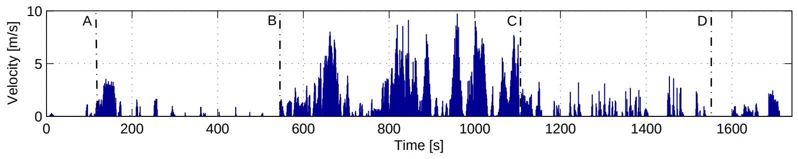


Fig. 7. Velocity profile during run 1 of NQE area B.

area) was spent dealing with obstacles - 30.2% in the nominal obstacle clearance mode, and 12.6% in the more aggressive modes. It also switched out of intersection-handling mode due to static obstacles 3 times and was in exception handling mode 0.65% of the total run.

The robot spent the first 9 minutes in the start area, where it needed to travel through a gate and an alley (event B). The logic correctly switched into the aggressive modes since the alley was too narrow for the robot to pass through while maintaining the nominal obstacle clearance. Unfortunately, the implementation of the switching to intersection-handling mode was lacking, and the obstacle clearance would get reset causing the path-planner to fail again. This happened 3 times in the start area, and the robot was stationary much of the time in this area (see figure 7). The robot swiched to the aggressive modes, and eventually to a failure mode, later in the run (around 700 s) due to a misallignment of the road and the RNDF. The robot got stuck in the parking area since it again could not maintain the necessary obstacle clearances and complete the goal. In this case, even the most aggressive mode was not sufficient.

The team realized that, in order to compete, it needed to adjust it's strategy. It was necessary to be more aggressive around static obstacles, but still maintain operational safety. It was decided to reduce the nominal obstacle clearance to the bare value by default, thereby collapsing the logic for changing this distance in the logic planner. That meant removing the connections between [DR,PR,S]→[STO,PR,A] and [STO,PR,S]→[STO,PR,A] (see figure 4). A connection (shown with a dashed arrow) was added from [STO,PR,S]→[FAIL]. For safety, the planner relied on the velocity planner to slow the robot near obstacles.

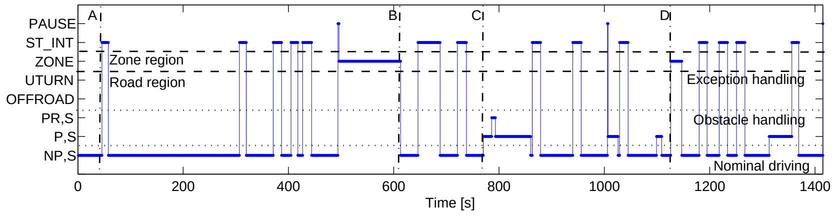


Fig. 8. Logic planner states during run 2 of NQE area B.

4.2 Run 2

The logic states for the second attempt are shown in figure 8. The robot was able to complete the run in a little over 23 minutes. It effortlessly exited the start area (event A), and drove up to and through the parking area (event B). Next, it navigated the ‘gauntlet’ (event C) successfully, before driving through the obstacle field (event D), and on to the finish area.

The robot spent 61.6% of the run in the nominal driving mode, and dealt with obstacles for 11.6% of the time. The robot switched out of the nominal driving mode 4 times to deal with obstacles. It also spent 16.5% of the time in intersection handling mode (14 intersections), and spent 7.93% of the run performing the parking maneuver. The robot spent 1.73% of the run navigating the obstacle field, and had no exceptions.

The time spent in obstacle mode was still worrisome. During the navigation of the ‘gauntlet’, the obstacles were so close together (longitudinally) that the function estimating the completion of the passing maneuver was insufficient. Thus, the robot remained in passing mode during most of this section.

4.3 Discussion

The notion of using the path planner capabilities to assist in the decision making process worked very well, even though the implementation was not perfect. The significant improvement in performance from run 1 to run 2 was due to the effective reduction in size of the robot. Some implementation shortcoming have been mentioned, and are summarized here. It is important to note that these shortcomings are often artifacts of other parts of the system. The logic for switching to intersection handling was fragile since the obstacle clearance mode was reset. Also, estimating whether a passing maneuver was complete was not robust. This was complicated by the path planning approach used. One shortcoming of the approach was not explicitly accounting for uncertainty in the decision process. It had been intended to extend the logic to account for this, but due to the time constraints it was not possible. However, by using the planner components to assist in the decision making, this shortcoming was largely mitigated.

5 Conclusions and Future Work

An approach to situational reasoning for driving on roads in urban terrain was described. In an attempt to manage the complexity of the reasoning module, knowledge from the path planner was used and the reasoning module was implemented as a finite state machine. This module was only invoked when the planner failed to find a solution while satisfying all the constraints imposed by the traffic rules. The reasoning module was implemented as part of a complete (and complex) autonomous system, developed for urban navigation. The performance of the module was discussed based on the results of the two runs in area B during the DUC NQE. The module imposed the correct behavior on the robot in most cases. The failures were a result of the implementation and the size of the robot. Uncertainty was handled implicitly through the use of the planner components to assist in the decision making. Future work includes extending this approach to explicitly account for uncertainty during the decision process.

Acknowledgements

The work presented here is the culmination of many members of Team Caltech, especially: Vanessa Carson, Sven Gowal, Andrew Howard, and Christian Looman.
This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract HR0011-06-C-0146, the California Institute of Technology, Big Dog Ventures, Northrop Grumman Corporation, Mohr Davidow Ventures, and Applanix Inc.

References

1. Looman, C.: Handling of Dynamic Obstacles in Autonomous Vehicles. Master's thesis, Institut für Systemdynamik (ISYS), Universität Stuttgart, (2007)
2. Sukthankar, R., Pomerleau, D., Thorpe, C.: A distributed tactical reasoning framework for intelligent vehicles. SPIE: Intelligent Systems and Advanced Manufacturing, October, (1997).
3. Unsal, C., Kachroo, P., Bay, J. S.: Multiple Stochastic Learning Automata for Vehicle Path Control in an Automated Highway System IEEE Transactions on System Man, and Cybernetics, Part A: Systems and Humans, January, (1999).
4. Gregor, R., Dickmanns, E. D.: EMS-Vision: Mission Performance on Road Networks. Proceedings of the IEEE Intelligent Vehicles Symposium, (2000).
5. Niehaus, A. Stengel, R. F.: Probability-based decision making for automated highway driving. IEEE Transactions on Vehicular Technology, Volume 43, p. 626-634, August, (1994)
6. Wongpiromsarn, T., Murray, R. M.: Distributed Mission and Contingency Management for the DARPA Urban Challenge. Submitted to: International Workshop on Intelligent Vehicle Control Systems, IVCS (2008)
7. Linderoth, M., Soltesz, K., Murray, R. M.: Nonlinear lateral control strategy for nonholonomic vehicles. In: Proceedings of the American Control Conference. (2008) Submitted.

Electronic Horizon - Providing Digital Map Data for ADAS Applications

Christian Ress, Aria Etemad, Detlef Kuck and Julián Requejo

Ford Research & Advanced Engineering Europe
Suesterfeldstr. 200, 52072 Aachen, Germany
{cress, aetemad1, dkuck1, jrequejo}@ford.com

Abstract. The number of vehicle navigation devices has increased tremendously during the last years. The digital maps of these systems contain a lot of valuable information that provides benefit for other features besides route guidance as well. The area of potential applications reaches from driver information and warning up-to comfort and active safety applications, so-called ADAS[1]. Since built-in sensors are limited to a relatively short range, digital map data can be used to "look" much further into the direction of the vehicle's path. The map data, e.g. road geometry, number of lanes, speed limits, etc. is provided on a vehicle bus system as the so-called Electronic Horizon. European automotive industry has teamed up in the ADASIS Forum to develop a common standardized interface to access the Electronic Horizon. Ford Research & Advanced Engineering Europe has developed a prototype system for Lane Keeping Assistance as one application example.

1 Introduction

Nowadays, modern vehicles are equipped with a variety of sensors to enable safety and comfort features. However, these on-board sensors are limited to a relatively short range of a few hundred meters. On the other hand, digital maps of a navigation system contain valuable information about the road segments lying ahead, such as road geometry, functional road class, number of lanes, speed limits, traffic signs, etc. This data can be extracted and provided to applications as the so-called *Electronic Horizon*. This virtual sensor is now able to provide information within an extended range, see Figure 1. Thus it enables the vehicle to prepare earlier for an up-coming situation.

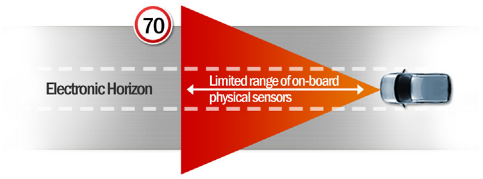


Fig. 1. Range of vehicle sensors.

[1] Advanced Driver Assistance Systems

2 Overview

The support of ADAS requires an accurate knowledge about the road ahead of the vehicle. Since the last years, map vendors already have significantly improved the level of detail and accuracy of digitised data. Up-to-date digital navigation maps contain detailed information about road geometry, topology, and typical attributes such as functional road class, number of lanes, speed limits, traffic signs, etc. Future maps will support even more details depending on the requirements demanded by the automotive industry to support ADAS.

The so-called Electronic Horizon contains road attributes from the digital map as well as vehicle position data. Fig. 2 gives an overview about the principle. The Navigation system, or more generally the *Electronic Horizon Provider*, extracts map data in the vicinity of the vehicle position. All relevant attributes required by applications are extracted and stored locally. The prediction of the *Most Likely Path*, which the vehicle is expected to take, has a significant influence on the quality and reliability of the Electronic Horizon. Ford Research and Advanced Engineering has developed algorithms for calculating the Most Likely Path based on historic information and current vehicle status [2]. The Electronic Horizon is provided to applications on the vehicle's bus system, most likely the CAN bus. The specification of the interface between Electronic Horizon Provider and applications is of special interest for the vehicle manufacturer. In order to reduce development cost and risk, a standardized interface is highly appreciated. As a result of this need, the European ADASIS Forum was launched in the year 2001. Section 4 outlines the interface specification and ADASIS Forum activities.

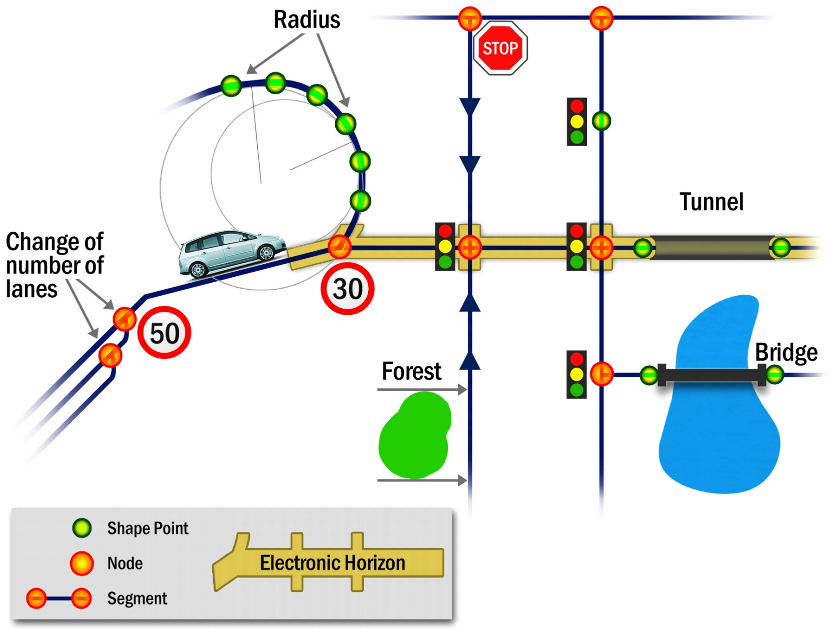


Fig. 2. Electronic Horizon.

3 Electronic Horizon Provider

Ford Research & Advanced Engineering has developed a prototype system to provide an Electronic Horizon, the so-called *Information Manager* [1] [3]. The system has been implemented as a prototype and integrated into test vehicles. The system does not only provide vehicle position and digital map data, but also predicts the Most Likely Path of the vehicle [2]. Fig. 3 gives an overview about the systems architecture. The entire system is designed in a modular way, allowing a maximum of flexibility and providing the opportunity to enhance and modify due to future requirements.

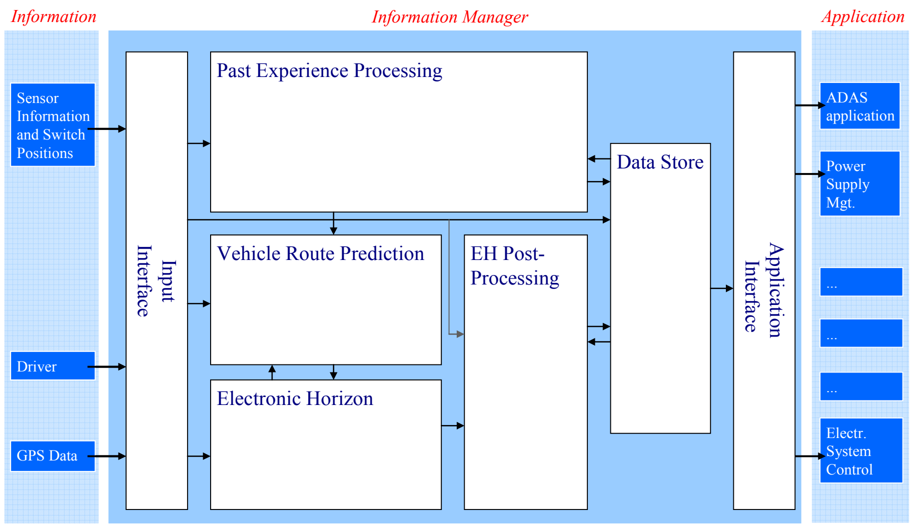


Fig. 3. Systems architecture of Ford's Electronic Horizon Provider (Information Manager).

The following paragraphs introduce the Information Manager in more detail.

3.1 Electronic Horizon & EH Post Processing

The "heart" of the system is a module that retrieves a configurable number of attributes from the digital map in the vicinity of the vehicle (the so-called *Electronic Horizon* module, *EH*). In a first step, the road segments for all possible paths are extracted from the map in the vicinity of the vehicle. Therefore the current vehicle position is estimated by a GPS receiver, and then matched on a road segment. The quality of the Electronic Horizon obviously depends on the correct estimation of the vehicle's path. If the road attributes would be provided for a path the vehicle does not take, the information will be worthless for the application and will cause a re-calculation of the Electronic Horizon. That would consume processing power and time leading to a gap of input data on the application side. In order to avoid re-calculations and to ensure proper operation of the system, the module receives information about the *Most*

Likely Path of the vehicle from *Vehicle Route Prediction* [2]. The EH module then extracts all relevant attributes attached to the road segments only for this single path.

The pre-processed Electronic Horizon is filtered in a second step, the *EH Post Processing,* due to the requirements of the supported applications. The EH Post Processing includes also demonstration software to visualize the Electronic Horizon as shown in Fig. 4. The module can be enhanced with additional sub-modules, e.g. a speed estimator that is calculating a predicted speed profile for the road segments ahead of the vehicle.

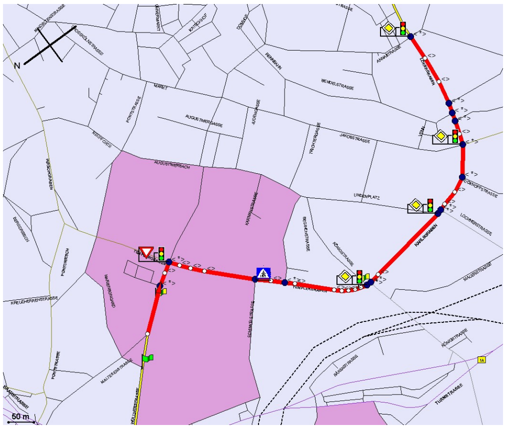


Fig. 4. Visualization of Electronic Horizon.

The calculated Electronic Horizon is then transferred to the *Data Store*.

3.2 Data Store

The *Data Store* is the common database of the system, which stores all data required by the applications as well as needed internally for the *Past Experience Processing*. Incoming data is received from the EH Post Processing as well as vehicle sensor data from the Input Interface. Data is then provided to the *Application Interface* as well as to the Past Experience Processing modules.

3.3 Input/ Output Interfaces

The *Application Interface* provides all data for any connected application in a specified format. The Application Interface supports the ADASIS protocol as described in section 4.2. At the moment, the Application Interface is implemented for the CAN bus only. However, other bus systems can be integrated easily by adding the adequate software.

The *Input Interface* manages all input data for the Information Manager. This includes vehicle sensor data, switch positions and other data available on the vehicle bus system as well as GPS position data. In addition, a driver identification mechanism is used, allowing the system to behave differently depending on the current driver. This information is needed by the *Past Experience Processing* and the *Route Prediction* modules.

3.4 Past Experience Processing

The Past Experience Processing stores the route currently driven by the vehicle, which is subsequently used by the Route Prediction module.

The Past Experience Processing contains a database for storing the collected information, building the basic step for a learning system. This database stores all trip data on segment by segment basis. That is while driving, all road geometry data is broken up into road segments as stored in the digital map. Along with coordinates or keys that allow identifying the segments later, additionally time, date and other data is stored as needed by other supported applications. Fig. 5 illustrates how the map database stores geometry information by splitting up the road network into segments and nodes. When driving to a destination the Past Experience Database receives the sequence of segments. For convenience, the system can either store the whole trip as that sequence or store only the transitions, which are consecutive segment pairs, where the second is always a successor to the first along with the time and date. These segment pairs are also called *decision points*.

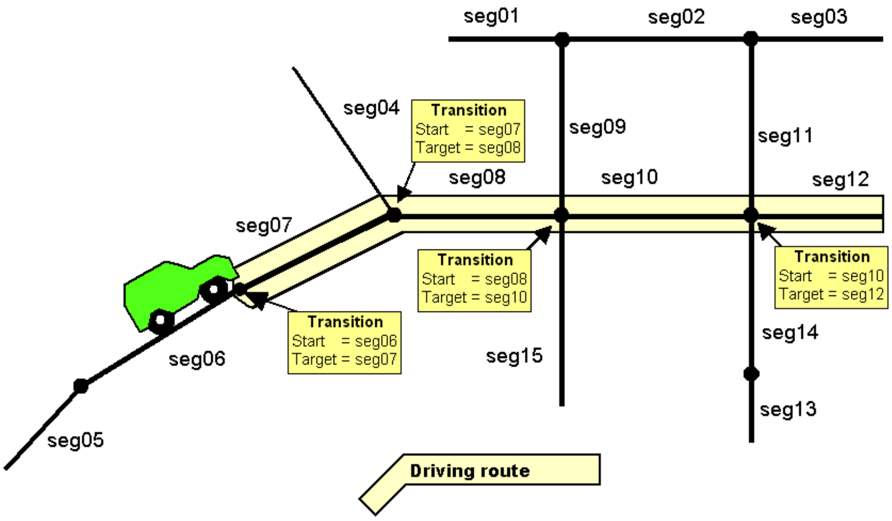


Fig. 5. Storing transitions while driving.

In order to optimize the memory usage, data is stored with an expiration timestamp. Once the past experience data reaches a specified age, the affected records are deleted. The memory management can further be improved by limiting the amount of stored data. Instead of keeping track of complete routes, only the segments that build a so-called decision point are considered (see above). During testing, it has been estimated that 3 -5 Mbytes of storage capacity will be sufficient for achieving a high prediction quality. This amount of memory would allow storing all routes driven by a typical driver during a period of three years.

Additionally, other data might be stored as well, in order to generate profiles and history of vehicle data. For instance, individual travel times can be recorded per each link. This information is then taken into account for future route calculations allowing to calculate an optimum route based on real-world travel times. Nowadays navigation systems only use static values based on the functional road class not matching reality e.g. in terms of traffic jams during rush hours.

3.5 Vehicle Route Prediction

The Vehicle Route Prediction module is responsible for determining the most probable route the vehicle is likely to go. Therefore, a two-way approach is used [2]. First, data from the vehicle's bus system is analyzed to determine a categorized driving situation. For instance, if the vehicle approaches a crossing, the system has to decide which way the driver intends to go. Taking into account turn signal information in combination with the current vehicle speed and distance to the decision point, an intelligent algorithm proposes the way of the vehicle at this crossing.

In a second step, a complete route can be proposed by taking into account the currently driven road segments and comparing these to the stored routes in the Past Experience Processing database. Particular patterns, which occur in a defined accumulation, are compared and used as a decision criterion for determining which route the driver intends to take. In order to optimize processing effort and memory usage no complete routes are compared. Instead only decision points are taken into account, as described in section 3.4, containing the succeeding segment for the one currently driving on. For each successor segment a specific probability is calculated based on the amount of occurrences, day-of-week and time-of-day when driven before. The algorithm chooses the segment with the highest probability as the Most Likely Path.

The estimated Most Likely Path of the vehicle is then provided to the Electronic Horizon module, which uses this information for retrieving a single-path Electronic Horizon, as described above.

4 Interface Specification

A well-defined interface between navigation system providing Electronic Horizon and applications is required. Vehicle manufacturers have a strong interest in using a standard interface specification. In order to develop such a standard interface the *ADASIS Forum* was founded. The following paragraphs introduce the forum itself and its technical approach.

4.1 ADASIS Forum

In 2001 the ADASIS Forum has been launched in Europe in order to specify an industry standard interface for providing Electronic Horizon. Ford Motor Company is playing an active role within that forum and contributing to the interface specification. Since December 2007, Ford has taken over the chairmanship of the forum (C. Ress).

The ADASIS Forum is hosted and coordinated by ERTICO[2] and constitutes to date of more than 30 members including car manufacturers, navigation systems and ADAS suppliers, as well as digital map vendors. The forum's purpose is to:

• Define an open standardised data model and structure to represent map data in the vicinity of the vehicle position (i.e. the Electronic Horizon), in which map data is delivered by a navigation system or a general map data server.

• Define an open standardised API to enable ADAS applications to access the Electronic Horizon and position-related data of the vehicle.

The final specification is currently worked out in the ADASIS Forum and will be transmitted as a next step to ISO for becoming an international industry standard. The first version of the interface specification is already available for forum's members.

During the period from 2004 to beginning of 2008, the successful work of the ADASIS Forum has been supported within the PReVENT project, i.e. MAPS&ADAS sub-project. The objectives for MAPS&ADAS project were to specify, implement, test, and validate the first version of ADASIS specification. Additionally, the development of new digital maps including safety aspects was also a goal of the project.

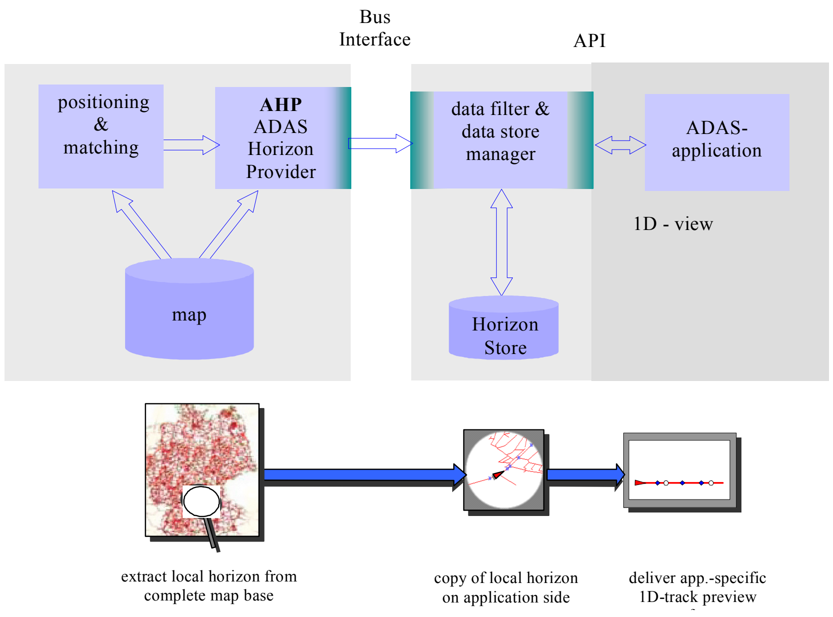

Bus Interface
API
positioning & matching
AHP ADAS Horizon Provider
data filter & data store manager
ADAS-application
1D - view
map
Horizon Store
extract local horizon from complete map base
copy of local horizon on application side
deliver app.-specific 1D-track preview

Fig. 6. ADASIS systems architecture.

[2] European ITS organization

4.2 ADASIS Functional Architecture

The systems architecture defined by ADASIS Forum is shown in Figure 6. It consists of the ADAS Horizon Provider (AHP) on the one side. That retrieves digital map data around the estimated position of the vehicle using GPS and map matching. In the first release of ADASIS only a single path in front of the vehicle is supported. Future versions will also be capable of multiple paths where the ADAS Horizon is provided. The data is then compressed and coded for transmission on the vehicle's bus system. On the application side, an ADAS Horizon Reconstructor (AHR) re-builds the ADAS Horizon from the received messages and provides it to the ADAS application. More than one application is supported but depending on the implementation, each one requires its own Horizon Reconstructor.

ADASIS addresses two interfaces on different levels:

(i) A "low level" interface describing the messages to be transferred on the vehicle bus system. The ADASIS specification is generic for any bus system, the so-called *AGMP (ADASIS Generic Message Protocol)*. Since the CAN bus is the most used bus system in vehicles nowadays, a specific implementation for CAN has been derived, the so-called *ASCP (ADASIS Specific CAN Protocol)*.

(ii) A "high level" interface allowing the applications to access the Electronic Horizon data after being re-built by the Horizon Reconstructor. This is a C code style API.

The developed systems architecture and interface specifications have been implemented as prototypes within the scope of PReVENT MAPS&ADAS by the project partners. Data transmission has been realized on a CAN bus. Tests and validation have been successfully been performed. The results look very promising: the average additional bus load caused by the Electronic Horizon is relatively low on the CAN bus (below 1 percent), and no latency issues or other negative effects for the CAN bus messages have been detected. Different ADAS Horizon Providers and Reconstructors have been developed by the project partners. These have been connected via the CAN bus and their interoperability was also demonstrated successfully.

More information about the ADASIS Forum is available on the internet: http://www.ertico.com/en/subprojects/adasis_forum.

5 Example Application: Lane Keeping Assistance

One of the applications, which have successfully been enhanced by the Information Manager, is *Lane Keeping Assistance*. A prototype system has been implemented and validated within the scope of the European PReVENT project. PReVENT is an industry research project, co-funded by European Commission within the 6th Framework, and has been successfully finalized in March 2008. More information about the PReVENT project is available on the PReVENT web site: http://www.prevent-ip.org .

For Lane Keeping Assistance a camera system is used as the primary sensor to detect if the vehicle is still within the lane. In order to support the image processing in difficult situations, e.g. approaching a curve, driving in bad weather, or low light conditions, Electronic Horizon data is taken into account as an additional virtual sensor. This provides information about lane attributes and geometry of the road ahead. In consequence this enables the lane tracker to evaluate more reliably if the vehicle is still within the lane, or whether an action needs to be taken. This could then be done in the form of a warning or by performing an active steering manoeuvre.

Fig. 7 shows an example of the benefit: ambiguous lane markings by shadows of guardrails are not recognized correctly by the camera itself, see left picture. Using lane width information from a digital map can help the lane tracker to exclude wrong information and to detect the lane markings correctly, see right picture.

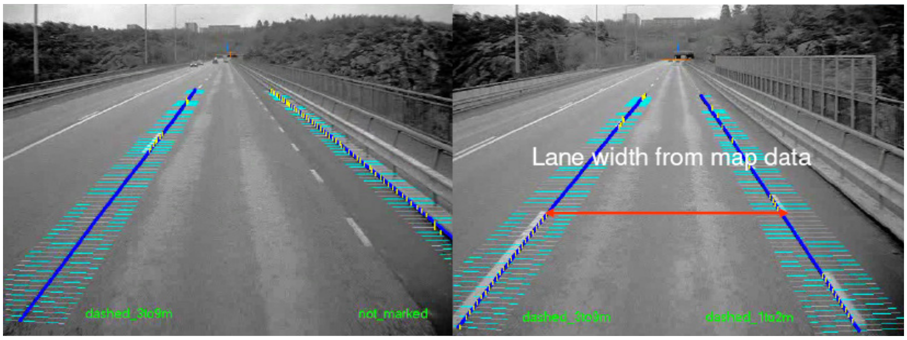


Fig. 7. Lane marking detection with camera only (left) and enhanced with Electronic Horizon (right).

6 Summary and Perspectives

Nowadays navigation systems have the potential to offer much more than solely route calculation and guidance. The digital map provides a lot of information about the route lying ahead of the vehicle that can successfully be used to enable new or enhance existing applications, the so-called Electronic Horizon.

Ford Research & Advanced Engineering has successfully designed and implemented a prototype system providing Electronic Horizon. It provides digital map data as well as the vehicle's position and sensor data to ADAS applications. Special attention has been paid on the calculation of the Most Likely Path the vehicle is expected to take. That allows a high quality of Electronic Horizon data and improves reliability of applications.

As one example how an application can benefit of Electronic Horizon "Lane Keeping Assistance" has been presented. Within the PREVENT project the lane tracker algorithms have significantly been improved with the use of digital map data as an additional "virtual sensor".

However for a series implementation of the system, some restrictions have to overcome in the near future. For instance, the ADASIS specification has to be refined with regard to the test experiences from PReVENT project. Also the Ford prototype system is implemented on a PC with a relatively huge amount of memory and processing power, which has to be replaced by an Embedded Platform for future use in a vehicle or to be integrated into the navigation system.

References

1. Ress, C., Etemad, A., Kuck, D., Boerger, M.: Electronic Horizon - Supporting ADAS applications with predictive map data. Proceedings of ITS World Congress, London, United Kingdom (2006)
2. Etemad, A., Ress, C., Boerger, M.: Generating accurate Most Likely Path data. Proceedings of ITS World Congress, London, United Kingdom (2006)
3. Ress, C.: The potential of digital map data to enhance ADAS functions. Telematics Update Magazine, Issue 40, First Conferences Ltd, London (2007)
4. Requejo, J, Ress, C., Etemad, A., Kuck, D.: Using Predictive Digital Map Data to Enhance Vehicle Safety and Comfort. The Second International Workshop on Intelligent Vehicle Control Systems, Madeira, Portugal (2008)

Using Predictive Digital Map Data to Enhance Vehicle Safety and Comfort

Julián Requejo, Christian Ress, Aria Etemad and Detlef Kuck

Ford Research & Advanced Engineering Europe
Süsterfeldstr. 200, 52072 Aachen, Germany
{jrequejo, cress, aetemad1, dkuck1}@ford.com

Abstract. Navigation systems based on GPS and digital maps have become extremely popular in passenger and commercial vehicles during the last years. The accuracy and amount of information contained in digital maps is constantly evolving, offering a tremendous amount of knowledge about the road network. In addition to providing driving directions, this information may be used on a wide range of technologies. One of these is the so-called Electronic Horizon, an application based on digital maps that helps to overcome the physical limits of many vehicle sensors, such as radar and vision systems. This is accomplished by predicting where the driver is heading, and subsequently extracting from the digital maps information about the predicted path. This paper provides an introduction to the Electronic Horizon, the Most-likely Path concept, and gives some examples of how digital map data can be used for the enhancement of comfort and safety functions.

1 Background

In recent years, navigation and telematics systems based on GPS and digital map databases have become increasingly popular in passenger and commercial vehicles. The accuracy of GPS receivers has improved with the introduction of new chipsets and technologies like the SiRF III and DGPS. At the same time, the road coverage of digital maps has been extended, its accuracy has been improved, and map vendors have committed to offer even further improvements in upcoming generations. In addition to road network topography, digital maps are able to carry large amounts of information describing the characteristics of the road. These include information like speed limit, number of lanes, curvature, slope, tunnels, lane dividers, traffic signs, and many other important attributes.

These digital map attributes can be very useful to many in-vehicle applications beyond the traditional point-to-point driving directions. This paper is divided in two main sections. The first part will introduce the reader to the Electronic Horizon concept. This technology is used for efficiently extracting the most relevant data from digital maps, and presenting it to other applications via a common interface layer. The second part will give an overview of some of these applications and functional enhancements, some which would not be as efficient or even not realizable without the use of digital map data.

2 Electronic Horizon

As modern vehicles are constantly being equipped with additional functionalities, the amount of in-vehicle electronics is equally growing. This not only increases the overall complexity, but it also introduces additional production costs. New technologies like adaptive cruise control require a radar sensor, while other functions like over speed warning rely on camera sensors for the recognition of traffic signs.

Although physical sensors have experienced substantial improvements and cost reduction within the past years, their range is still limited by physical constrains. Despite the accuracy of the traffic-sign pattern recognition, no vision sensor is able to anticipate the speed limit behind the next curve. Other road attributes like the presence of upcoming traffic lights, pedestrian crossings, or the estimating the radius of an upcoming curve can simply not be deduced by using conventional sensors.

Digital maps on the other hand, can be bundled with large amounts of data about the road network. By extracting information from the digital maps, a speed warning system would be able to make decisions based on speed limits several hundred meters ahead. This range goes substantially further than in conventional vision systems, allowing for new or enhanced functionality.

Extracting digital-map information about the road being driven and presenting to other applications constitutes the so-called Electronic Horizon (EH) concept. Figure 1 (left) illustrates how EH interprets digital-map data at the current vehicle position and uses this information for creating a virtual view of the road ahead.

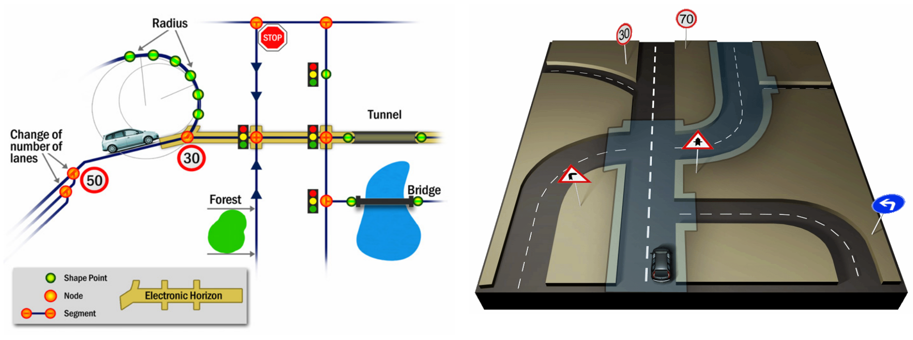


Fig. 1. Electronic Horizon and the Most-likely Path concept.

The task of extracting digital information from the roads with the virtual horizon becomes exponentially complex as network density increases. In highly dense areas with multiple intersections, such as large cities, the amount of possible paths grows rapidly. Extracting map information about every possible path would be unfeasible and constitute a waste of system resources. As a matter of fact, most applications that would benefit from EH are not interested in all possible scenarios, but rather on the most-likely one.

Figure 1 (right) illustrates the most-likely path concept (MLP) in a scenario with multiple intersections. Several methods exist for estimating the most-likely path. These vary in accuracy, complexity, and required system resources. If the driver is using the navigation system for driving directions, the destination address and the corresponding planned route are readily known. Other methods include static predictions based on road classes, or more complex dynamic calculations. Nevertheless, regardless the computational method, MLP is effectively able to reduce a 360 degree electronic horizon such as that shown in Figure 2, down to a single virtual linear road. Armed with a unique path, EH can now discard all not-so-likely alternatives, and extract as much digital information as possible about the most-likely path. All relevant information is then filtered, sorted, and presented using a standard interface to other applications [1]. It should be obvious that the feasibility of EH and the use of digital map data is highly dependent on the accuracy of the MLP predictor. If the vehicle energy management module bases its charging cycle decision on a wrong prediction, the benefit of EH would not be as significant. Ford Research Centre Aachen has developed an MLP algorithm based on past experience. This method is able to learn the driving patters of the vehicle owner, creating a driving history that may be used in subsequent drives. After a few days of learning, this technique has been able to achieve accuracy levels of up to 99% under typical driving behavior [2].

3 ADAS Application Support

The high level of accuracy achieved by MLP in combination with past experience has permitted the development of a very robust EH. The information extracted from EH has been used by different applications, some for enhancing performance, while in other situations for providing new functionality. Some applications currently being researched by the automotive industry will be presented in the following sections.

3.1 Curve Speed Warning

Accidents due to high speeds in curves have always been a concern to drivers. Slippery surfaces, hidden curves, and sharp turns are some examples of the many dangerous situations that could arise when driving curves. In other scenarios, driving in curves at high speed might not be necessarily dangerous, but it could provide discomfort to the driver or its passengers.

Curve-speed warning technology (CSW) has been developed with the goal of identifying these uncomfortable or potentially dangerous situations, and being able to warn the driver with enough time to react. CSW is a non intrusive technology, based on standard digital maps that are commonly present in GPS-based navigation systems. Since the application does not require additional physical sensors, its additional functionality is provided at a minimum cost. The process of identifying hazardous situations is divided into different phases. First, the system needs to predict the road that is most likely to be taken by the driver. Once the route is identified, the system accesses the digital maps and retrieves information about the shape and characteristics of the upcoming curve. The combination of path prediction and extraction of information from the digital maps is provided by the electronic horizon module (EH). Figure 2 illustrates the CSW principle based on gathering the shape points that describe an upcoming curve, and using this information to estimate the road curvature.

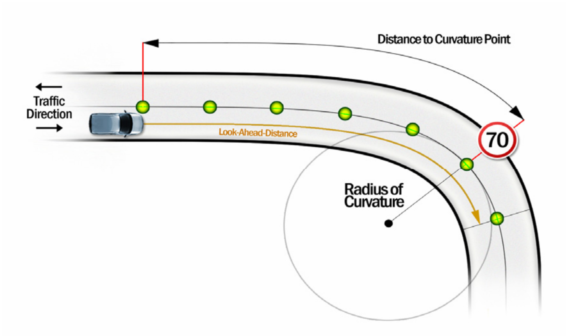


Fig. 2. Curve Speed Warning curvature estimation.

Future generations of digital maps will incorporate curvature information as native road attributes, simplifying the real-time computational requirements during driving.
In addition to road curvature, CSW considers other factors such as weather conditions and estimates of road friction. With this information in hand, the maximum recommended speed for the curve is estimated. If the vehicle is approaching the curve at a speed higher than the recommended value, the system could either warn the driver of the potential hazard, or actively inhibit further acceleration of the vehicle. In order to avoid annoying the driver with unnecessary warnings, the alert threshold is dynamically adjusted based on the driving style. This allows the system to recognize whether the driver prefers a sportive or a more comfort-oriented driving style, adjusting itself without user intervention.
In this way, curve-speed warning is effectively able to increase the driver awareness in dangerous scenarios, therefore reducing the likelihood of accidents.

3.2 Lane Keeping Support

Automotive OEM and suppliers have been actively looking for ways to improve the overall vehicle safety using cameras and visual sensors. Lane Keeping Support (LKS) is a camera-based technology which uses road pictures and image processing algorithms for determining whether the vehicle is driving within lane boundaries.
This technology is of significant importance not only for passenger vehicles, but also for commercial vehicles, where drivers are subject to long periods of driving. LKS is able to assist the driver in several dangerous scenarios, such as sleepiness, tiredness, or simple distractions.

These algorithms typically work by recognizing the lane markings painted on the road and computing the intended direction of travel. In parallel, the algorithm analyses the steering wheel angle, vehicle speed, and other parameters, and determines the actual direction of travel. If the intended direction of travel differs from the actual direction of travel, the system is able to spot a lane departure scenario.

Each LKS implementation may react in response to lane departures in different ways. This could be either by means of an acoustic alarm, an active intervention on the steering wheel, or a combination of both. Active intervention systems are usually

very subtle, providing only a steering guidance that is overridden as soon as the driver resumes control of the situation.
Despite its potential, there are several scenarios that could easily confuse the lane detection algorithm. Figure 3 (left) shows a typical lighting problem due to the low dynamic range of typical camera sensors. In this example, exiting from a tunnel in a sunny day exposes the camera to a simultaneous dark and bright situation that is not able to handle properly. As a result, LKS is not able correctly determine the lane markings beyond the tunnel, forcing to temporarily disable the safety feature.

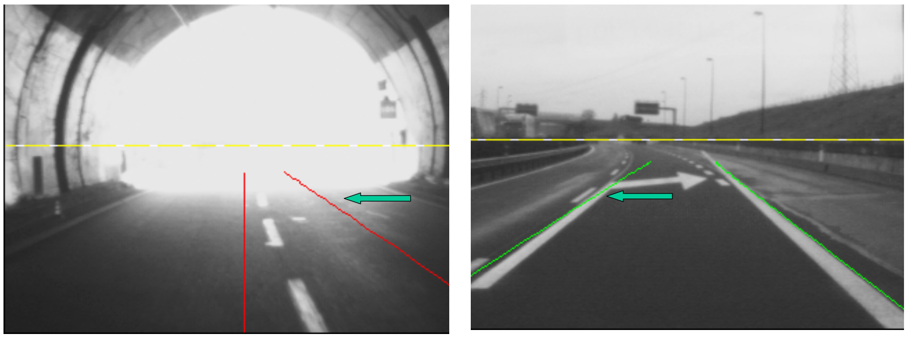


Fig. 3. Lane detection problems due to lighting conditions and lane expansion.

The right side of figure 3 shows another problematic scenario due to changes in the lane configuration. In this example, a new lane opens in the highway, confusing the algorithm for detecting lane markings. Such a situation would trigger a false lane departure alarm or even worse, an incorrect active intervention. Lanes that merge or split, highway entrances and exits, bicycle lanes, and pull-in lanes are just some examples of the many situations that could confuse the lane recognition algorithm. One way to improve accuracy of the system under these scenarios is with the support of digital map data and EH. Next generations of digital maps, as well as some prototypes used within research, already incorporate attributes with the number of lanes, type of markings dividers, exits and entrances in highways, and tunnels. By combining vision sensor information with digital map data, it is possible to predict an upcoming situation that could confuse the lane detection algorithm. In these examples, the lane departure system would know from the digital map data that the vehicle is in a situation that would likely confuse the vision algorithm. As a countermeasure, the system could either use an alternative strategy or temporarily inhibit the false alarm.

By incorporating knowledge about the road ahead, LKS can reduce the amount of false positives, therefore reducing the driver annoyance, increasing the acceptance of the technology, and improving the overall vehicle safety.

3.3 Adaptive Cruise Control

Automatic cruise control has been present in series vehicles for many years, as a way to increase driving comfort in those situations where a constant speed is desired, such as driving in highways. With the introduction of radar and laser sensors in vehicles, next generation cruise control systems have been developed. In addition to maintaining a constant speed, adaptive cruise control (ACC) uses radar or laser sensors to monitor the speed of the vehicle ahead, as well as the distance separating them. If the difference in speed and distance is considered to be unsafe, ACC could warn the driver, reduce the target speed, or even actively brake the vehicle.

Despite being a leap forward in comparison to standard cruise controls, ACC can easily get confused under various driving scenarios. Figure 4 shows one these potentially dangerous situation. In this example, the driver had programmed his cruise control to drive at a certain speed. After a while, the vehicle encountered a slower vehicle ahead. In order to maintain a safety distance, ACC reacted by reducing the vehicle speed. After some time, now the driver desires to leave the highway, pulling into the exit lane. At this point, ACC will see the slower vehicle disappear and as a result, it would accelerate back to the target speed. This behavior is probably the opposite of what the driver expects when exiting from a highway and it could represent a very dangerous situation.

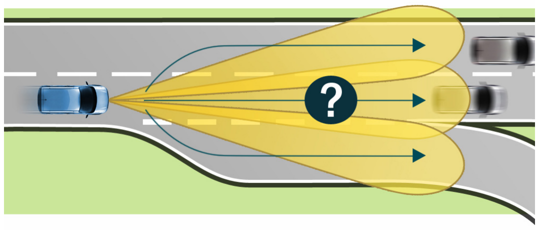


Fig. 4. Adaptive Cruise Control problem when exiting from a highway.

One way to support ACC in these confusing situations is with digital map information provided by EH. If the system had known that the driver was likely to exit the highway, it would have not accelerated. Even in the event of an incorrect prediction, by simply knowing about a highway exit at the current location, ACC would have waited until past the exit, in order to make sure that the driver is still in the highway.

By providing additional information about the road, EH is effectively able to assist ACC, improving its overall effectiveness and user experience.

3.4 Adaptive Front Lighting

Headlamps have always been an important item during vehicle design. Automotive OEMs have provided customers with headlights whose height may be easily adjusted with the push of a button, cooler and brighter lamp colors, and extra lights for aiding in parking maneuvers. Despite its usefulness, these solutions offer limited functionality and often require some manual intervention by the driver.

In contrast to fix lighting, adaptive front lighting (AFL) is able to adapt the shape, height, intensity, area covered, and other aspects of the light beam to fit each particular driving scenario. This permits AFL to customize the lighting to each particular driving situation, improve overall visibility and vehicle safety.

Despite its potential, determining the correct driving scenario and predicting the shape of the road headed by the vehicle constitute major obstacles for an effective

AFL. This information can be extracted by the Electronic Horizon, enabling the use of interesting configurations not possible otherwise.
Figure 5 shows different situations in which AFL in combination with EH is able to provide some assistance. While waiting on a traffic light, AFL can adjust its lighting pattern to match the intended behavior. If EH predicts a left turn, AFL would focus its beam on that direction, with the possibility of an additional side LED-light for spotting crossing pedestrians. This provides not only a clearer view of the road, but it also shows the driver's intention to other vehicles, and it provides additional pedestrian protection.

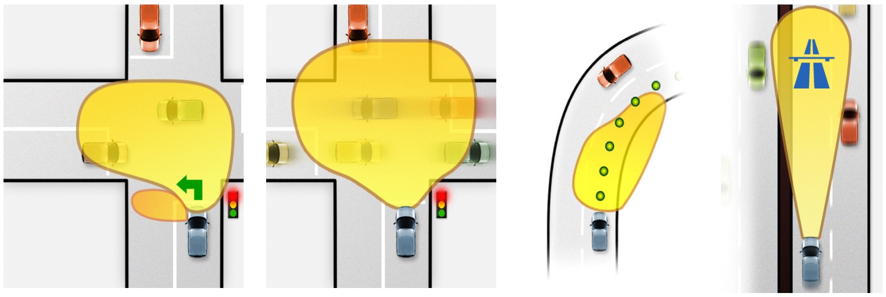


Fig. 5. Various scenarios suitable for Adaptive Front Lighting.

If driving straight, AFL would extend the range slightly further, offering a more comfortable view in the direction of travel.

Using this technology in curves or winding roads provides a better view of the road, while inducing the driver to look in the direction headed by the vehicle. In addition, it reduces the blinding effect experienced by forthcoming vehicles, offering an increase in comfort, as well as in overall safety. Highways can be easily identified by EH and, depending on external factors like the presence of other vehicles, it may switch to a long range mode for optimal lane view, or a more conservative low-beam configuration.

3.5 Hybrid Vehicle Support

With the introduction of hybrid vehicles into the market, considerable effort has been devoted to maximizing their efficiency. The main concept behind hybrids is to combine an electrical motor with a traditional gasoline engine, profiting from the advantage offered by each device. On one hand, regenerative braking is used for recovering energy that would otherwise be lost as heat. This energy is stored in a high-capacity battery and later is reused by the electric motor. This could be used for briefly driving the vehicle at low speeds, where the gasoline engine would not be as efficient. Alternatively, the electric motor may provide a complementing torque to the power-train, reducing the work of the gasoline engine. Despite its advantages, accomplishing an optimal energy management strategy is a difficult task. For instance, driving downhill or braking with a fully charged battery would constitute a waste of valuable energy. Likewise, starting a charging cycle before a hill would probably overload the engine and require an interruption of the charging phase. Using stored energy as a complementary torque close to a pedestrian crossing might not represent the best choice, as this energy could possibly be better used for accelerating the vehicle after the crossing. Many possible scenarios and strategies are conceivable for hybrid vehicles, and determining an optimal decision is dependent on multiple factors. Nevertheless, using digital map data can be of great value to the energy management strategy. Information coming from EH such as traffic lights, pedestrian crossings, and road inclination can be used for optimizing the assessment. Figure 6 illustrates EH in action. After predicting the most-likely path, EH gathers information about slope (shown in the figure with color bars), traffic lights, and crossings up to 1km ahead.

Armed with a profile of the road ahead, the vehicle is able to compute more efficient engine and charging strategies, ultimately leading to lower fuel consumption.

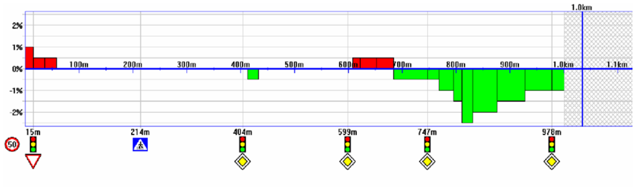


Fig. 6. Electronic horizon supporting hybrid vehicles.

3.6 Driver Awareness

One important safety application of electronic horizon is the inducing of higher driver awareness, with the immediate consequence of an improved overall safety. In contrast to other safety technologies like CSW, driver awareness intends to identify potentially dangerous scenarios based exclusively on digital map data and current vehicle location. These scenarios could be specific locations that the driver should be aware of, such as kindergartens, schools, pedestrian crossings, or dangerous intersections.

Another important example of driver awareness is the identification of the so-called hot-spots. These are locations statistically known to be problematic or have previously incurred in a large number of accidents. These could be specific highway entrances, lanes that enter an avenue without a merge area, difficult left turns on two-way roads, narrow bridges, roads susceptible to icing, etc. However, in order to provide valuable information ahead of time, it is essential to implement a highly reliable electronic horizon using precise digital map data, and an accurate most-like path prediction.

3.7 Green Driving Support

The rising prices of oil, in combination with increasing pressure from governments to reduce vehicle emissions have become a major concern to both drivers and manufactures. The automotive industry is investing considerable effort in reducing fuel consumption levels, such as with the introduction of hybrid vehicles and better engine management. At the same time, the transportation business is quite aware of the fact

that driving style has a large impact in fuel consumption. These concerns have introduced the concept of *green driving* or *green navigation*.
One way to support green driving is by computing routes to destination that would result in the lowest fuel consumption. Drivers are usually aware of the fact that driving on highways is more efficient than driving through the city. However, other factors such as hills or mountains, speed limits, traffic lights, and types of intersection play an important role in finding the green route.

Determining the green route is difficult for the driver, particularly if he is not familiar with the area. Quite often, the route with the lowest fuel consumption does not represent the fastest route. And to make things harder, routes with best fuel economy are not necessarily the routes with lowest fuel consumption. This makes an educated guess a difficult task. Figure 7 shows two real-world alternatives computed by the driving-directions portal Map24. The first alternative shown on the left was provided as the fastest route, with a distance of 13 Km and an estimated travel time of 19 minutes. The fuel consumption was predicted to be 0.9 Liters, giving a fuel economy of 6.67 L/100 Km. The second route was provided as the shortest route, with a distance of 10.4 Km, a travel time of 21 minutes, and a fuel consumption of 0.7 L. This configuration gives a fuel economy of 7.17 L/100 Km. Other examples could show an opposite scenario, where the lowest fuel consumption is achieved by the fastest route.

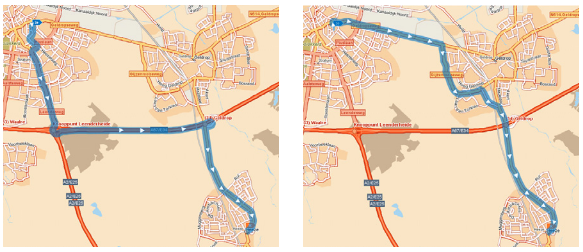


Fig. 7. Two route alternatives from A to B (source map24).

In any case, digital map data is essential for estimating the greenest route. This computation is carried by combining slope, road class, and surface type information, as well as a profile describing the consumption characteristics of the vehicle.

Although the green route is typically computed before departure –by entering the destination into the navigation system–, EH is also able improve the fuel consumption during driving. With knowledge about the road ahead, such as slope and road class information, green route is able to optimize the engine and energy management strategies. This effectively results in a reduction of the overall fuel consumption, and consequently lower emissions and a greener driving.

4 Conclusions

The introduction of navigation systems in passenger vehicles has opened the door for a large number of applications. In addition to the highly deployed driving guidance, digital maps contain a tremendous amount of information that could support existing applications, as well as enable new technologies not realizable otherwise. This paper has provided an overview of some map-based applications currently being researched in the automotive industry. Despite their potential, it should be emphasize that the effectiveness and acceptance of these technologies is highly dependent on precise map data and an accurate electronic horizon. Map vendors are working in cooperation with the automotive industry to provide the necessary road attributes in their next-generation maps. At the same time, new generation of GPS technologies are offering better precision, and the accuracy of digital maps is being constantly improved.

As next-generation maps and new technologies become available to consumers, more map-based applications are expected to be deployed in vehicles. Due to their low implementation cost, these features are very attractive to OEMs. These will not only offer additional driving comfort, but also improve the overall vehicle safety.

References

1. Ress, C., Etemad, A., Kuck, D., Boerger, M.: Electronic Horizon - Supporting ADAS applications with predictive map data. Proceedings of ITS World Congress, London, United Kingdom (2006)
2. Etemad, A., Ress, C., Boerger, M.: Generating accurate Most Likely Path data. Proceedings of ITS World Congress, London, United Kingdom (2006)
3. Ress, C.: The potential of digital map data to enhance ADAS functions. Telematics Update Magazine, Issue 40, First Conferences Ltd, London (2007)
4. Ress, C., Etemad, A., Kuck, D., Requejo, J.:). Electronic Horizon – Providing digital map data for ADAS applications. The Second International Workshop on Intelligent Vehicle Control Systems, Madeira, Portugal (2008)

Intelligent Refueling Advisory System

Erica Klampfl[1], Oleg Gusikhin[1], Kacie Theisen[1], Yimin Liu[1] and T. J. Giuli[2]

[1] Ford Research & Advanced Engineering, Systems Analytics & Environmental Sciences Department, RIC Bldg, MD #2122, 2101 Village Rd., Dearborn, MI 48124, U.S.A.
{eklampfl, ogusikhi, ktheise1, yliu59}@ford.com

[2] Ford Research & Advanced Engineering, Vehicle Design & Infotronics Department RIC Bldg, MD #3137, 2101 Village Rd., Dearborn, MI 48124, U.S.A.
tgiuli@ford.com

Abstract. Recent advances in wireless communication technologies have led to numerous new developments that take advantage of network access, ranging from real-time information delivery essential for many driving decisions to harvesting off-board computing power for remote vehicle diagnostics. Specifically, with the ever increasing fuel cost, there is growing popularity of services that provide information to drivers on current gas prices and alerts on upcoming changes to gas prices. This paper demonstrates how to take currently available technology one step further by providing a proactive refueling advisory system based on minimizing fuel costs over routes and time. The system integrates vehicle data, a navigation system, and internet connectivity to supplant the existing vehicle low fuel warning with a comprehensive decision support system on refueling choices.

1 Introduction

Recent advances in wireless communication technologies have led to numerous developments that take advantage of internet connectivity. These developments support a wide range of industries and applications. Focusing on the personal transportation sector, harvesting computing power accessible over the net can be used for remote vehicle diagnostics. Another example is real-time information delivery used onboard vehicles to assist drivers with decisions such as where to find the nearest gas station. A popular extension of this is an emerging service that offers access to current retail gasoline prices. With ever increasing fuel prices, drivers are paying closer attention to their refueling strategies, making when and where to refuel a more important and complex question than in the past. Many drivers are willing to drive a few extra miles to get a cheaper gas price. Drivers may even be willing to only purchase a few gallons of gas today when refueling is necessary with expectations of lower gas prices the next day; this results in more frequent stops, but lower total fuel costs.

The model we introduce in this paper offers users substantial benefits by identifying the best possible refueling strategy over an entire route and multiple time periods, as opposed to the best strategy based on current gas prices in a small local area. It includes how to determine the best refueling strategy for a driver by considering their route, the gas stations and gas prices along the route, the starting fuel level, good estimates of fuel consumption along the route, and user preferences.

Inputs to the system can come from several different sources. A driver's route can be determined in either of two ways: it can be manually input or for frequently driven routes it can be predicted from past driving patterns (see [8] and [16]). Once the route has been determined, gas stations along the route can be identified along with current gas prices. Current gasoline prices can also be used to estimate future fuel prices by a forecasting model. By connecting to the vehicle's internal network, the current fuel level can be obtained along with the vehicle's average fuel economy, which can be used to form a good estimate of fuel consumption for new routes. For frequently driven routes, previously collected data from the vehicle internal network can be used determine route specific fuel consumption.

We will first present background in Section 2 and then give an overview of our system in Section 3. We then provide details on how we forecast future gas prices and determine the refueling strategy in Sections 4 and 5, respectively. We present an example in Section 6 and discuss future work and conclude in Section 7.

2 Background

In the United States, there are currently two main sources of current gasoline prices: credit card transactions and networks of gas price spotters. GasBuddy [10] and GasPrice-Watch [11] are examples of the latter approach. Both companies offer a web site and mobile application where the user can look up gas prices for free. The gas prices reported are collected from a network of gas price spotters who are usually volunteers. Consequently, neither company can guarantee the accuracy of the information they provide. Oil Price Information Services (OPIS) [19] obtains their data from credit card transactions, which makes it more reliable than other sources.

When a driver is deciding whether or not to stop to purchase gas, they usually contemplate if the gas prices will be going up or down in the next couple of days. Gas prices at the pump largely depend on wholesale prices and are subject to cyclical fluctuations. While some people follow fluctuations of retail gasoline prices and may have reasonable predictions on when prices will rise or fall, the majority of drivers do not know the likelihood of gas prices going up or down. There are a number of popular web sites that offer the general population an educated guess on the future direction of gas prices; one example is The Gas Game [20].

To assist drivers in making more informed refueling decisions, many car navigation system manufacturers offer gas price services delivered to the vehicle through mobile phone data services or satellite radio broadcast. Examples include the TomTom 920T [17], Dash express [3], and nüvi 780 by Garmin [9] (most get their prices from OPIS). Vehicle manufacturers, like Ford Motor Company, are even taking this one step further by offering their factory installed navigation system with SIRIUS Travel Link [7]. In addition, some researchers have explored notifying drivers of the cheapest place to refuel when their current gas level drops below a certain threshold (see [13] and [15]).

In comparison to the privately owned vehicle sector, the commercial trucking industry has considered complex decisions about when and where truck drivers should stop for gas as a part of their route planning process for quite some time due to the tight correlation between profits and fuel cost. They have developed optimization models to assist with refueling decisions as each truck's driving route is known and network infrastructures are in place to communicate with truck drivers on the road: an example is Expert Fuel developed by Integrated Decision Support [14].

Our approach differs from the above in the several ways. First, we use forecasted gas prices, so that we can consider future days on the route, making the best decision over multiple days. While forecasting has been used for other consumer oriented help applications, such as predicting when online customers should purchase airline tickets [6], it is a new approach for providing refueling strategies. Another way that our approach is different is that our routes do not have to be predetermined, as we can get routes directly from the vehicle GPS. In addition, we get other data directly from the vehicle internal network, such as current fuel level and fuel consumption. Also, we can recalculate on the fly a new strategy if the driver deviates from the original plan. Finally, our system acts as a proactive fuel gauge that notifies the driver before they are approaching the gas station where they should stop (current low on fuel gauge warnings provide data only relative to the internal fuel tank).

3 System Overview

Our system combines web and in-vehicle components to assist the driver in minimizing their fuel purchase costs, as seen in Figure 1. To use the system, a driver must first register on a web portal and enter their fuel purchase preferences, such as how many times per week they are willing to stop for gas and which brands they prefer. The web site also assists the driver in mapping out their frequently traveled routes. The driver then creates an approximate travel plan for the next week, which the system uses to identify potential refueling stations. Future versions of our system will predict the driver's future driving routes without any input from the driver

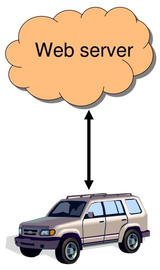


Fig. 1. System consisting of web to vehicle connection.

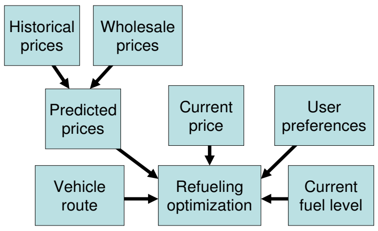


Fig. 2. This figure illustrates the data flow used in the refueling strategy optimization algorithm.

To get the most out of our system, the driver's vehicle should include a navigation system and a means of connecting the vehicle to the internet, such as a telematics platform (e.g., OnStar [18], BMW Assist [1]) or a cell phone with data service (e.g. GPRS or EVDO). The vehicle maintains a connection to the web server and periodically uploads the vehicle's current fuel level and GPS location. An optimization algorithm (see Section 5.1) on the web server uses the fuel level to calculate when and where the driver should refuel and how many gallons to purchase. If the algorithm determines that the driver should stop, it sends this information to the vehicle's navigation system and alerts the driver.

Figure 2 shows the main data sources our system uses to calculate a refueling plan. Future fuel prices for individual fueling stations are predicted using a forecasting algorithm (see Section 4) that takes into account historical prices as well as wholesale prices. Historical and current prices are obtained from a database maintained by OPIS. User preferences and vehicle routes are obtained from the web portal, and the current fuel level is directly obtained from the vehicle.

4 Forecasting Model

As previously mentioned, if the consumer had more information on gasoline prices at each gas station along the specified route and predicted prices for the next few days, determining a refueling plan that would minimize the total fuel cost over the trip would be achievable. One important aspect in determining the consumer's refueling strategy is an econometrical model that reasonably predicts daily gasoline retail prices at each station on the route. Many studies show that the retail gasoline price is highly correlated with wholesale gasoline price, according to the production and distribution system of gasoline in the U.S [2]; this provides the foundation for our approach.

We present a simple forecasting approach that predicts future retail gas prices at individual stations when the following limited data is available: wholesale regional gas prices, one-month future market wholesale prices, current station retail gas prices, and historical retail station gas prices. To determine which wholesale price to use as the explanatory variable in our forecasting model, we considered several wholesale prices to see which provided the best correlation to retail gas prices. There are four regional wholesale prices in the U.S.: mid-continent, Los Angeles, New York, and Gulf coast markets. These four regions and the one-month future market wholesale prices have the following correlation with retail prices at gasoline stations in Michigan: 0.89, -0.71, -0.89, 0.92, and -0.91, respectively. In addition, we tested the correlations between the average of two or more wholesale prices and the retail gasoline prices: all values were less than 0.92. As a result, we chose the Gulf coast wholesale price as our explanatory variables in the model, which yielded the best correlation to retail gas prices.

The Energy Information Administration (EIA) reported in 1999 [5] that the downstream gasoline price responses caused by upstream price changes can be represented by the following equation:

$$D_t = \beta_0 + \sum_i (\beta_i U_{t-i}) + \varepsilon. \tag{1}$$

Here D_t is the price offered to each gas station by the supplier at time t; it is a moving average of the wholesale prices U_{t-i} at time t for the previous days i such that $i = 1, \ldots, 6$. The error term is ε, which also represents the downstream markup.

The markup of each gasoline station π_t at time t is the retail gasoline price p_t at time t at each gasoline station minus the price D_t paid to its suppliers; that is,

$$\pi_t = p_t - D_t. \tag{2}$$

In this study, we do not use the coefficients β provided by the EIA report, but estimate coefficients $\overline{\beta}$ by applying an Ordinary Least Square (OLS) linear regression [12]: the retail gasoline price at time t is the dependent variable, and explanatory variables are the wholesale prices U_{t-i} for $i = 1, \ldots, 6$, and the monthly dummy prices, M_k, where $k = 1, \ldots, 12$ represents January through December, respectively, for any day t. The results of this OLS linear regression are in Table 1. Applying the regression coefficients, $\overline{\beta}$ and $\overline{\theta}$, we can obtain the estimated suppliers' prices $\overline{D_t}$:

$$\overline{D_t} = \overline{\beta_0} + \sum_i (\overline{\beta_i} U_{t-i}) + \sum_k (\overline{\theta_k} M_k). \tag{3}$$

Table 1. Relationship between retail gas and mid-continent wholesale prices: $R^2 = 0.97$, ***Statistically significant at 99% confidence level, *Statistically significant at 90% confidence level.

Variable	Coefficient ($\overline{\beta}$)	Standard Error
U_{t-1}	0.002 ***	0.001
U_{t-2}	0.001	0.001
U_{t-3}	0.001	0.001
U_{t-4}	0.001	0.001
U_{t-5}	4.531 e-04	0.001
U_{t-6}	0.003***	0.001
M_2	-0.180 ***	0.014
M_3	-0.090***	0.020
M_4	-0.039*	0.031
M_5	0.020	0.038
M_6	0.089***	0.033
M_7	0.088***	0.032
M_8	0.102 ***	0.025
M_9	-0.012	0.030
M_{10}	-0.012	0.030
M_{11}	0.001	0.041
M_{12}	0.009*	0.006
constant	0.950***	0.059

So, the estimated markup of each gasoline station at time t is $\overline{\pi_t} = p_t - \overline{D_t}$. However, we randomly chose some gasoline stations to study their markup and found some interesting markup patterns. For each day of the week, even if the markup $\overline{\pi_t}$ is different for every t, it is always higher at the end of a week and lower at the beginning of a week. Since it is not feasible to test each gasoline station's markup, we make the assumption from our comprehensive testing that gasoline prices at each gasoline station have a weekly pattern; without loss of generalization, we calculate the average markup for each weekday, providing us with seven average markups for each gasoline station. We call this average markup for each weekday $\tilde{\pi_n}$ for $n = 1, \ldots, 7$; i.e. for each day of the week n, we calculate the average markup for n using historical data for n. For example, to calculate $\tilde{\pi_n}$ for $n = 1$ (Monday), we take the average markup over all Mondays ($\overline{\pi_t}$ such that t is a Monday) in our historical data. Hence, we can predict the gas price for any station one day in advance by

$$\overline{p_{t+1}} = \overline{D_{t+1}} + \tilde{\pi_n}, \tag{4}$$

where $n = 1, \ldots, 7$ represents Monday through Sunday, respectively, for any day t. When retail gas price for time t is not available for a specific station, we can also use equation (4) to estimate the missing price.

To forecast the gasoline prices at one specific gasoline station more than one day in advance, we can do so using $\overline{p_{t+i}} = \overline{D_{t+1}} + \pi_{\tilde{n+i}}$, for $i = 1$ to m, where m is the number of days ahead for which the forecast is desired. Because we do not know the wholesale or suppliers' prices on days $t+i$, when $i > 1$, the accuracy of the forecasting results will decrease as i increases.

The graphs below show the prediction results from the simple price forecasting model described above for gas purchases using 2007 daily data. Figure 3 and 4 display average daily retail gasoline prices (averages include prices for 931 stations) and predicted gasoline prices in Michigan. Figure 3 shows the average daily retail gasoline prices compared to the predicted gasoline prices based on the one-day ahead prediction for Michigan over 2007. Figure 4 includes five predicted prices which are one-day ahead, two-day ahead, three-day ahead, four-day ahead and five-day ahead, respectively, for the month of November 2007. For example, when using the three-day ahead predictions, we would use November 1, 2007 data to predict the price of gas on November 4, 2007. The mean of absolute residual values (i.e., the absolute difference between real price and predicted price) for the one to five day ahead predictions are approximately 3.2 cents, 2.8 cents, 2.9 cents, 3.1 cents, and 3.3 cents; the standard deviation of the absolute residual values is 2.3 cents, 2 cents, 2.1 cents, 2.2 cents and 2.4 cents, respectively. Figure 5 show the predicted daily gasoline prices at one stations in Michigan as an example to demonstrate the accuracy of predicted prices for an individual gas station.

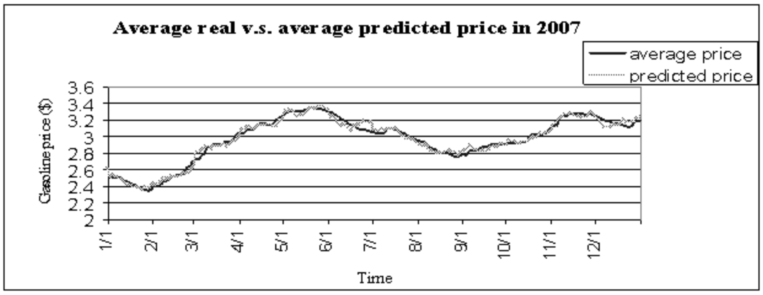


Fig. 3. Average retail price compared to average predicted retail price for 931 gasoline stations.

Some papers discuss factors that impact daily or weekly gasoline price fluctuation [4], but few researchers have studied forecasting models for daily gasoline prices. Compared to those models, our approach for price prediction provides a simple way when limited data is available to estimate daily gasoline price at each gasoline station with forecasting results that are within a reasonable range to the retail prices. In future research, we plan to develop a more advanced forecasting model that will require more data, but will be able to include effects of other factors on retail gasoline prices, such as a gasoline station's brand, distance to competitors, station characteristics, regional income level, etc. Moreover, we will use an autoregressive integrated moving average

(ARIMA) model [12] to better understand the lag terms $t-i$ (the number of days before t that are considered) and their coefficients.

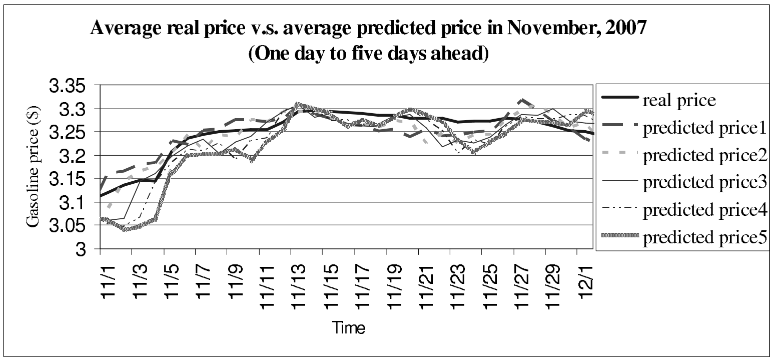


Fig. 4. Average retail price and 5 average predicted prices of 931 stations in November, 2007.

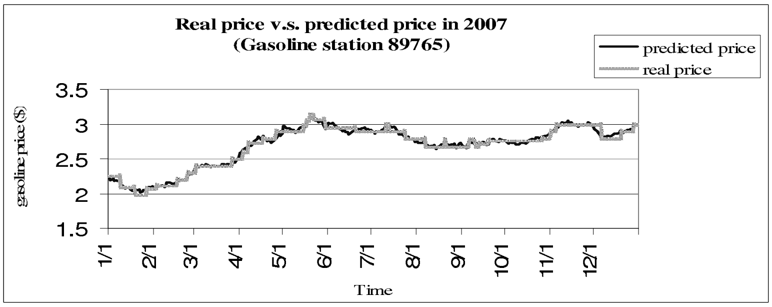


Fig. 5. Real retail price and predicted retail price at gasoline station 89765 in 2007

5 Solution Techniques

Once we have the forecasted gas prices, we are ready to use these as cost inputs to the objective function in our optimization model. There are a number of different approaches for solving the optimal refueling strategy problem from heuristical approaches to discrete optimization methods. In this paper, we first describe how to model the refueling strategy optimization problem as a Mixed Integer Program (MIP) [21]. Next, we briefly explore two simple heuristics: the first has the driver stop at the cheapest gas station in the area whenever the driver runs out of gas, and the second recommends a refueling stop within a certain tank capacity range.

5.1 Mixed Integer Program

We first describe the MIP optimization technique, introducing the input information, discussing the variables, the objective function, and ending with a detailed description of the constraints and overall model formulation.

Input Information. In this section, we describe the parameters that are inputs to the MIP model. We divide these inputs into the following categories: inputs obtained from vehicle internal network, inputs implicit from route specifications, inputs defined by user to specify preferences, and inputs from the forecasting model. We define a route to be a multi-day trip plan of around one week. After a week, the accuracy of the forecasted gas prices diminishes. However, the week can be a rolling horizon, where the model is solved every day for a new week time period.

We get the following information from the vehicle internal network:

$\mathcal{M}$ax $=$ maximum number of gallons the gas tank holds
$\mathcal{MPG}$ $=$ miles per gallon (note that this could be different depending on the route)
$\mathcal{G}_0$ $=$ initial amount of gas in the vehicle at the beginning of the route

As discussed in Section 3, driver routes are continually being collected on the vehicle internal network to populate the user's most frequently driven routes or can be set up on the web portal by the driver for routes not previously driven.

Knowing these routes, we can determine the following inputs:
n $=$ number of gas stations in route
$\mathcal{S}$ $=$ set of gas stations in route $= \{1, 2, \ldots, n\}$
m $=$ number of driving days or time periods in the route
$\mathcal{D}$ $=$ set of days $= \{1, 2, \ldots, m\}$
d_i $=$ the distance from the starting point of the route to gas station i
ξ_i $=$ the distance from each station i back to the original route
NPI_t $=$ the new period index $\forall \;\; t \in D$. The gas stations are sorted in the order in which the driver will encounter them along the route. NPI_t identifies the first gas station encountered on day t. For example, $NPI_1 = 1$ is the index for the first gas station in the first time period. If on the second day ($t = 2$) the first gas station to visit is station 256, then $NPI_2 = 256$.

In addition to the routes, the user specifies the following preferences through the web portal:

$\mathcal{M}$in $=$ minimum number of gallons of gas the driver wants to have in the fuel tank at any given time
MST $=$ maximum number of stops over the entire route (note that this should take into consideration the number of miles driven so that the problem does not become infeasible)
MSD $=$ maximum number of stops in one day of the trip (note that this should take into consideration the number of miles driven so that the problem does not become infeasible)

The last input involves the cost of gas at each station. If the gas station occurs along the route on the current day, then the cost of gas is the current gas price. If the gas station occurs along the route on a future day or the current day's price is not available, then it comes from the forecasting model.
c_i $=$ cost of gas at station i.
We keep track of the first station in each time period, so we get the corresponding forecasted gas prices $\overline{p_{t+i-1}}$ for days $i = 2, \ldots, m$ for each station.

Variables. This model contains both binary and continuous variables. The binary variables help make choices of when to stop, whereas the continuous variables determine how much gas to get at a station.

- The first variable, x_i, is a binary variable that determines whether or not the driver should stop at gas station i. In other words,
  $x_i = \begin{cases} 1 \text{ if stop at gas station } i \\ 0 \text{ otherwise} \end{cases} \quad \forall i \in \mathcal{S}.$
  Recall that the time period is embedded in this variable as we keep track of it by the value NPI_t.
- The second variable, y_i, is the amount (in gallons) of gas purchased from station i for every $i \in \mathcal{S}$.

Objective Function. The objective is to minimize the total amount spent on gas when traveling on a specific route over a certain number of days. The user preferences of how many times they are willing to stop also comes into play by adding a weighted penalty to the number of stops. If the user does not set a preference for the number of stops, then we let $\alpha = 0$; that is, we don't impose a penalty. The objective function is as follows

$$\min \sum_i (c_i y_i + \alpha x_i) \tag{5}$$

Constraints. In this section, we will discuss the constraints that are involved in guaranteeing that the driver does not run out of gas, that the driver does not get more gas than the fuel tank can hold, and if the driver stops at a station than the driver must get gas.

The first constraint specifies that the vehicle must never run out of gas. So, when the vehicle gets to station i, the amount of gas that the vehicle had at station $i-1$ needs to be enough to get to station i and still have the minimum gallons required. Note that we add the distance ξ_j to account for the distance that the driver takes if they stop at some station $j < i$ to get back to their route.

$$\mathcal{M}in \leq \mathcal{G}_0 - \frac{d_i}{\mathcal{MPG}} + \sum_{j<i} y_j - \sum_{j<i} \frac{\xi_j}{\mathcal{MPG}} x_j \quad \forall \; i \in \mathcal{S}. \tag{6}$$

The second constraint guarantees that the vehicle will never have more gas than the amount held by the fuel tank. So, for every station visited,

$$\mathcal{G}_0 - \frac{d_i}{\mathcal{MPG}} + \sum_{j\leq i} y_j - \sum_{j<i} \frac{\xi_j}{\mathcal{MPG}} x_j \leq \mathcal{M}ax \quad \forall \; i \in \mathcal{S}. \tag{7}$$

We also constrain the number of stops per route; this accounts for how much gas the vehicle's tank can hold so that the problem does not become infeasible.

$$\sum_{i \in S} x_i \leq MST \tag{8}$$

Or, if we only want a maximum number of stops per day, we address it by the following two constraints. The first constraint is for all time periods except for the last one, and the second constraint covers the last time period:

$$\sum_{i\in S: i\geq NPI_{t-1} \& i<NPI_t} x_i \leq MSD \;\; \forall \; t \in \mathcal{D}\setminus 1, \qquad \sum_{i\in S: i\geq NPI_m \& i\leq n} x_i \leq MSD. \quad (9)$$

The last two constraints are linking constraints that guarantee if the driver does not stop at station i to get gas then no gallons should be purchased; that is,

$$y_i \leq (\mathcal{M}ax - \mathcal{M}in)x_i \;\; \forall \; i \in \mathcal{S} \text{ and } x_i \leq y_i \;\; \forall \; i \in \mathcal{S}. \quad (10)$$

Bounds. The variable x_i is a binary variable. The variable y_i is a real number that must be greater than or equal to zero, but less than or equal to the size of the fuel tank minus the preference of how much fuel should always be left in the tank. Note that the upper bound on the y_i variable is implied by the first linking constraint in (10). The bounds are as follows: $x_i \in \mathcal{B} \;\; \forall \; i \in \mathcal{S}$ and $0 \leq y_i \leq \mathcal{M}ax - \mathcal{M}in$ such that $y_i \in \Re \;\; \forall \; i \in \mathcal{S}$.

Problem Formulation.

$$\begin{aligned}
&\min_{x_i, y_i \;\forall i \in \mathcal{S}} \sum_i (c_i y_i + \alpha x_i) \\
\text{s.t.}\quad & \\
&\mathcal{M}in \leq \mathcal{G}_0 - \tfrac{d_i}{\mathcal{M}PG} + \textstyle\sum_{j<i} y_j - \sum_{j<i} \tfrac{\xi_j}{\mathcal{M}PG} x_j \;\; \forall \; i \in \mathcal{S} \\
&\mathcal{G}_0 - \tfrac{d_i}{\mathcal{M}PG} + \textstyle\sum_{j\leq i} y_j - \sum_{j<i} \tfrac{\xi_j}{\mathcal{M}PG} x_j \leq \mathcal{M}ax \;\; \forall \; i \in \mathcal{S} \\
&\textstyle\sum_{i\in S} x_i \leq MST \\
&\textstyle\sum_{i\in S: i\geq NPI_{t-1} \& i<NPI_t} x_i \leq MSD \;\; \forall \; t \in \mathcal{D}\setminus 1 \\
&\textstyle\sum_{i\in S: i\geq NPI_m \& i\leq n} x_i \leq MSD \\
&y_i \leq (\mathcal{M}ax - \mathcal{M}in)x_i \;\; \forall \; i \in \mathcal{S} \text{ and } x_i \leq y_i \;\; \forall \; i \in \mathcal{S} \\
&x_i \in \mathcal{B} \;\; \forall \; i \in \mathcal{S} \text{ and } 0 \leq y_i \leq \mathcal{M}ax - \mathcal{M}in
\end{aligned}$$

5.2 Heuristics

Two different heuristic approaches are considered as possible solution techniques. The first approach has the driver stop for gas whenever the fuel gage hits a minimum allowable fuel level. For example, if this minimum level is 2 gallons then the driver will stop at the gas station right before the fuel level drops to 2 gallons. When stoping for gas the driver always refuels to the maximum tank capacity. This method may not give the lowest total fuel cost over a trip, however it will result in the minimum number of stops possible to complete the trip. The second method is a Greedy heuristic where the driver continues along their given route until the fuel gage registers a predetermined amount. As an example, let this predetermined amount be half a tank of gas. At this point, the heuristic checks all of the given gas stations between half a tank and the minimum allowable fuel level. The station with the lowest gas price in this range is selected as the refueling point, and when the driver stops, they fill their tank to the maximum capacity. The driver then continues along the route until the fuel level reaches half a tank, at which point the process repeats itself. This method should show some improvement in fuel costs compared with the previous heuristic and will eliminate the possibility of purchasing small amounts of gas at any given stop.

6 Example Scenario

A sample scenario was created to demonstrate the various solution techniques and to compare resulting MIP solutions when user preferences are considered. In this scenario, a five day trip has been planned in which there are 1532 possible gas stations to stop at along the route. For each gas station, the forecasted price of gas and the distance from the starting point is known. The driver's vehicle averages 22 miles per gallon, holds a maximum 15 gallons of gas, and is starting the trip with 5 gallons of fuel. The driver preferences are not to let the fuel level drop below 2 gallons and no stopping for gas more than twice in one day or more than six times over the entire trip.

Table 2. Scenario Results for Optimization Methods.

Optimization					Optimization with Penalty				
Day	ID	Distance	Price ($)	Buy	Day	ID	Distance	Price ($)	Buy
1	28	15.571	2.799	10.708	1	41	26.685	2.799	11.213
1	171	115.498	2.799	4.542	2	367	284.007	2.837	10.544
3	484	373.707	2.821	6.507	4	853	544.649	2.719	8.098
4	853	544.649	2.719	8.098	5	1240	722.803	2.698	7.539
5	1240	722.803	2.698	7.539	N/A				
Total Trip Fuel Cost: $ 103.398					Total Trip Fuel Cost: $ 103.655				

Table 2 displays the following results for the scenario: a listing of on which days to stop, at which gas stations to stop, the price of gas at each station, and how much gas should be purchased at each station. The first column (Optimization) provides the solution when the objective function is to minimize to cost of fuel over the trip and α is set to zero such that there is no penalty for the number of stops. The second column (Optimization with Penalty) provides the results for the optimization model that includes a penalty in the objective function each time the driver has to stop for gas; this enforces user preferences on how many times a day and per trip they are willing to stop (see equation 5). Here we set $\alpha = 1$, but the value of α could vary depending on the number of gas stations considered in the route. If we compare the results for the optimization models with different objective functions, we see the difference caused by the penalty in the objective function: it eliminates an additional stop on the first day for a slight increase in overall cost of around $\$.25$.

Table 3 shows the results for both heuristic methods when applied to this scenario. The first column gives the solution when the driver stops for gas right before their fuel gage registers the minimum gas allowed (2 gallons in this scenario). The second column gives the results of the greedy heuristic, which looks for the lowest priced gas between when the fuel gage registers half a tank and 2 gallons.

Table 4 compares the results of the four approaches. The heuristic, which has the driver stop at the closest gas station when they reach 2 gallons of gas in their tank, results in the fewest stops and eliminates only partially filling up the gas tank when stoping. However, this technique does result in the highest total fuel cost which is 6.44% higher than the original optimization technique.

Table 3. Scenario Results for Heuristic Methods.

Stop At 2 Gallons					Greedy Heuristic				
Day	ID	Distance	Price ($)	Buy	Day	ID	Distance	Price ($)	Buy
1	101	65.892	2.999	12.995	1	41	26.685	2.799	11.213
2	432	350.930	2.946	12.956	2	381	290.172	2.837	11.977
4	1061	636.879	2.877	11.442	4	924	563.492	2.719	12.424
N/A					5	1446	829.094	2.698	1.78
Total Trip Fuel Cost: $ 110.06					Total Trip Fuel Cost: $ 103.945				

Table 4. Technique Comparison.

	Optimization	Optimization with Penalty	Stop at 2 Gallons	Greedy
# Stops	5	4	3	4
Stops Fill < 10 Gallons	4	2	0	1
Total Cost ($)	103.398	103.655	110.06	103.945
% Over Lowest Cost	N/A	0.25%	6.44%	0.53%

7 Conclusions

With fuel prices on the rise in the United States, drivers are increasingly concerned with their refueling strategy. This paper presented a model to assist drivers with decisions regarding where and when to refuel along with how much fuel to purchase. The model takes into account various factors, such as user preferences (e.g. how often the driver is willing to stop for gas) and vehicle specific information (e.g. the current fuel level). Gas stations along a specified route can be identified and gas prices corresponding to future days of the route can be determined using the forecasting method discussed in Section 4. Once this information has been aggregated, it is used by the Mixed Integer Programming model presented in Section 5.1 to provide the driver with an optimal refueling strategy. As shown in section 5.2 heuristical approaches can also be used to determine a refueling strategy.

Future versions of this model can be extended to include additional driver preferences or economic factors. For example, by considering flex-fuel vehicles, the refueling decision becomes even more complex as the analysis would include tradeoffs between the price per gallon of fuel, average miles per gallon associated with a particular route and vehicle, and possibly the CO2 effects for multiple fuel types. In addition, we could move towards a more advanced forecasting model and provide a refueling strategy that would take into consideration multiple routes for a single origin destination pair.

The proposed model offers substantial benefits over existing approaches. First, it provides an optimal refueling strategy using forecasted fuel prices over multiple time periods and locations as opposed to a local strategy based on a small geographical region and the current fuel prices in that area. In addition, the vehicle internal network can be used to monitor the current fuel level and consumption to offer proactive advice instead of waiting for the driver to request help from the system. Finally, over time the system can learn a driver's regular routes and associated fuel consumption to provide more accurate recommendations in the proposed refueling strategy.

Acknowledgements

We would like to thank Jenny Lin at Corporate Economics of Ford Motor Company for providing the 5 series of wholesale prices.

References

1. BMW. BMW website. `http://www.bmwusa.com/Standard/Content/Owner/BMWAssist/Default.aspx`, 2008.
2. Borenstein, Cameron & Gilbert. Do gasoline prices respond asymmetrically to crude oil price changes? `TheQuarterlyJournalofEconomics,Vol.112,1997`, 1999.
3. Dash. Dash website. `http://www.dash.net/`, 2008.
4. Eckert, & West. Retail gasoline price cycles across spatially dispersed gasoline stations. `JournalofLawandEconomics,April,2004`, 2004.
5. Energy Information Administration. Price changes in the gasoline market. `http://tonto.eia.doe.gov/FTPROOT/petroleum/0626.pdf`, 1997.
6. Farecast. Our Technology and Data website, 2008. `http://www.farecast.com/about/ourTechnology.do`.
7. Ford Motor Company. Popular Ford SYNC System Expanded; New SIRIUS ”Travel Link” Takes Navigation to New Level website, 2008. `http://media.ford.com/article_display.cfm\?article_id=27489`.
8. J. Froehlich and J. Krumm. Route predictions from trip observations. *Society of Automotive Engineers (SAE) 2008 World Congress*, April 2008.
9. Garmin. `https://buy.garmin.com/shop/shop.do?pID=13479`, 2008.
10. GasBuddy. GasBuddy website. `http://www.gasbuddy.com/`, 2008.
11. GasPriceWatch. GasPriceWatch website. `http://www.gaspricewatch.com/`, 2008.
12. William Greene. *Econometric Analysis*. Prentice Hall, 2007.
13. E. Horvitz, P. Koch, and M. Subramani. *Mobile Opportunistic Planning: Methods and Models*, pages 228–237. Lecture Notes in Computer Science. Springer Berlin, 2007.
14. Integrated Decision Support. Planning and Execution Solutions for Truckload Carriers: Expert Fuel website, 2008. `http://www.idscnet.com/software/fuel.htm`.
15. E. Kamar, E. Horvitz, and C. Meek. Mobile opportunistic commerce: Mechanisms, architecture, and application. In *Proceedings of AAMAS 2008*, Estoril, Portugal, May 2008.
16. A. Kapoor and E. Horvitz. Experience sampling for building predictive user models: A comparative study. In *Proceedings of CHI 2008*, Florence, Italy, April 2008.
17. Ludovic Privat. TomTom to offer real-time fuel price to GO 920. *GPS Business News*, January 10 2008.
18. OnStar. OnStar website. `http://www.onstar.com/`, 2008.
19. OPIS. OPIS website. `http://www.opisnet.com/`, 2008.
20. The Gas Game. The Gas Game website. `http://www.thegasgame.com/`, 2008.
21. L. Wolsey. *Integer Programming*. John Wiley & Sons, Inc., 1998.

A Predictive Controller for Object Tracking of a Mobile Robot

Xiaowei Zhou, Plamen Angelov and Chengwei Wang
Intelligent Systems Research Laboratory
Lancaster University, Lancaster, LA1 4WA, U. K.
p.angelov@lancs.ac.uk

Abstract. In this paper a predictive controller for real-time target tracking in mobile robotics is proposed based on adaptive/evolving Takagi-Sugeno fuzzy systems, eTS. The predictive controller consists of two modules; i) a conventional fuzzy controller for robot motion control, and ii) a modelling tool for estimation of the target movements. The prediction of target movements enables the controller to be aware and to respond to the target movement in advance. Successful prediction will minimise the response delay of the conventional controller and improve the control quality. The model learning using eTS is fully automatic and performed 'on fly', 'from scratch'. Data are processed in 'one-pass' manner, therefore it requires very limited computational resource and is suitable for on-board implementation on the mobile robots. Predictions are made in real-time. The same technique also has the potential to be used in the process control. Two reference controllers, a controller based on the Mamdani-Type fuzzy rule-base, and a controller based on the simple linear model, are also implemented in order to verify the proposed predictive controller. Experiments are carried out with a real mobile robot Pioneer 3DX. The performance of the three controllers is analyzed and compared.

1 Introduction

The main objective of the object tracking is controlling the robot to maintain a constant distance and heading to the mobile object being tracked [1]. A simple first priciple controller can be used for this purpose based on the linearisation of the problem [2]. Alternatively, in a pursuit of more accurate tracking, a fuzzy controller can be applied. A fine tuned fuzzy controller [3] can achieve higher accruacy comparing to the simple linear controllers. However, one problem that the conventional controllers are facing is that the controller generates the manipulated value (control command) according to the observation of system status at the current and the past time instants while the purpose of the control is to minimise the observed error in the forthcoming (future) time instant. Taking into account the dynamic nature of the target system, this delay, in response may lead to larger errors. For this reason, a predictive controller which is able to predict the behavour of the target system is recommended in such cases [4]. Instead of a response to the directly observed measurements, the so called model-based predictive controller (MBPC) makes the control decision based on the predicted values. Therefore, a predictive model is an indispensable part of any MBPC scheme [4]. In [5] a Takagi-Sugeno (TS) fuzzy model has been used as a model predictor. This model, however, was pre-trained off-line and was with a fixed strtucture. The eTS concept introduced recently [6]-[8] allows the TS fuzzy model to be designed on-line, 'on fly' during the process of control and operation. This is especially suiatble and convenient for applications such as robotics where the autonomous mobile robots may be required to operate in a completely unknown, dynamic, or harsh environment [9].

The main problems in controllers design [10] are; i) their stability; ii) their tuning. The former problem is not treated in this paper. The latter one is usually approached in off-line mode and also from the point of view of adpative control theory [2] which is well developed for the linear case [11]. In a dynamcially changing environment eTS fuzzy systems have their advantage of flexibility and open structure. Moreover, they have been used in conjunction with so called indirect learning proposed by Psaltis in 1998 [12] described in [6] and [12]. While Psaltis and Anderson et al. [14] used off-line pre-trained and with fixed structure neural networks for their indirect learning scheme in [6] and [13] evolving FLC is used that learns 'on fly', 'from scratch' based on the operational data alone and no pre-training.

In this tracking problem, the desired velocity of the two side wheels of the robot is controled. The distance, d and the angle to the moving target, θ are measured at each sampling time, Figure 1. The objective of the control is to maintain a predefined distance to the target so that the target is closely followed without a collison (reference distance, d_{ref}). A heading angle of 0^o to the target is also required.

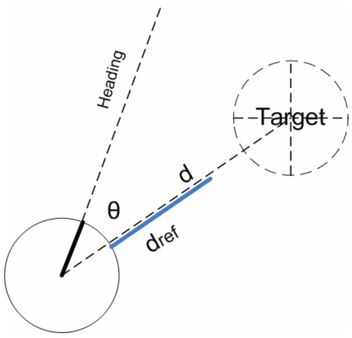


Fig. 1. Target tracking by a mobile robot.

The structure design of the conventional controller used as a basis benchmark for this test of target tracking by the mobile robot is illustrated in Figure 2. The current state described by the distance to the target, velocity of both wheels of the mobile robot (left and right) measured by the sensors mounted on the mobile robot Pioneer 3DX is fed back to the controller. The controller has a fixed structure and parameters that are determined based on common knowledge of the problem.

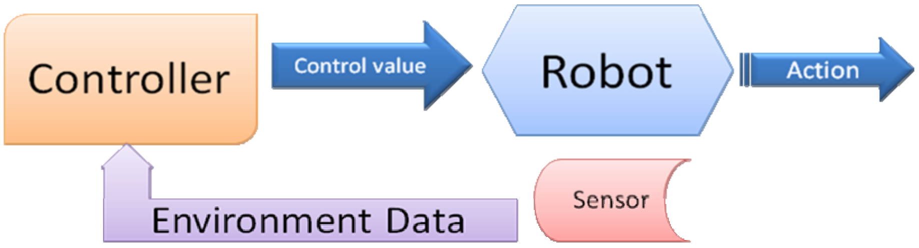


Fig. 2. Controller schematic.

1.1 First Principles-based Controller

The first principles-based controller used for this task is based on the explicit linear description of the problem. In order to follow the moving target, the acceleration of the robot is assumed to be proportional to the distance to the target, d. Due to the inertia of the real systems it takes a short period of time after a velocity command is received by the motor the desired velocity to be reached. Therefore, the velocities of both wheels (left and right) are selected as control values. The turning of the robot is achieved by control of the velocity difference between the left and right wheels. When the velocity of the left wheel is higher than the velocity of right wheel, the robot makes a right turn and vice versa. Based on these principles, the wheel velocity control model is described by the following equations:

$$V_{left} = V_f + V_l$$
$$V_{right} = V_f + V_r \qquad (1)$$

It consists of two components; V_f, the component for maintaining d_{ref} and the pair of velocities, V_l, and V_r which determine the heading of the mobile robot. The two components are defined by equations (2)-(3) below, also illustrated in Figure 3. Figure 3a) and 3b) illustrate the linear components which describes the control response in proportion to the distance and heading difference to the target respectively. When the distance to the target is Far the velocity component, V_f is set to *High*, which leads to a larger acceleration of the robot. While the distance is below d_{ref} (set in our experiments to 400mm), the velocity component, V_f is set to *Negative*.

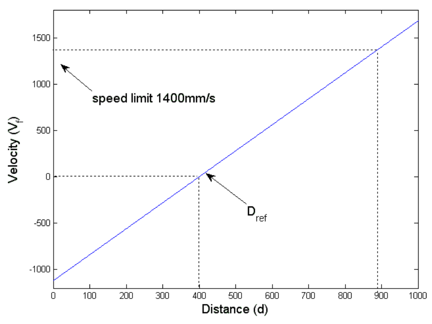


Fig. 3a. Distance component.

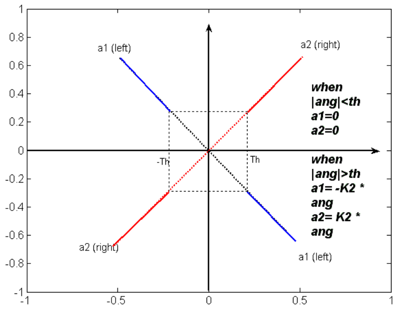


Fig. 3b. Angle component.

$$V_f = k_1(d - d_{ref}) \tag{2}$$

$$V_l = \begin{cases} 0 & |\theta| < \overline{\theta} \\ k_2\theta & |\theta| > \overline{\theta} \end{cases} ; \quad V_r = \begin{cases} 0 & |\theta| < \overline{\overline{\theta}} \\ -k_2\theta & |\theta| > \overline{\overline{\theta}} \end{cases} \tag{3}$$

Where $\overline{\theta}$, $\overline{\overline{\theta}}$ are threshold values; k_1 and k_2 are coefficients.

1.2 Fuzzy Controller

In an attempt to achieve a more flexible and accurate tracking, a Mamdani type fuzzy logic controller (FLC) has also been implemented [10]. It consists of five fuzzy rules:

The fuzzy rule base of the Mamdani type FLC:

Rule 1:
IF** (d is Crash)* ***AND *(θ is Negative)*
THEN** (V_l is Quick Back)* ***AND *(V_r is Quick Back)*
Rule 2:
IF** (d is Close)* ***AND *(θ is Straight)*
THEN** (V_l is Slow Back)* ***AND *(V_r is Slow Back)*
Rule 3:
IF** (d is proper)* ***AND *(θ is Small Positive)*
THEN** (V_l is Hold)* ***and *(V_r is Hold)*
Rule 4:
IF** (d is Not Far)* ***AND *(θ is Small Negative)*
THEN** (V_l is Slow Forward)* ***AND *(V_r is Hold)*
Rule 5:
IF** (d is Far)* ***AND *(θ is Positive)*
THEN** (V_l is Slow Forward)* ***AND *(V_r is Quick Forward)*

Each rule describes a typical situation during the tracking task. Real-time readings are obtained to form an input vector. The closeness from the measured input vector to the prototypes (focal points) of each fuzzy rule is calculated based on triangular membership functions illustrated in Figure 4. The result is aggregated to form the degree of firing for each rule and normalised and aggregated further to form the overall output of the FLC [10]. The antecedent part of the fuzzy rules is defined by linguistically interpretable terms that describe the distance (Figure 4) and angle; the consequent fuzzy sets are defined in respect to the velocity.

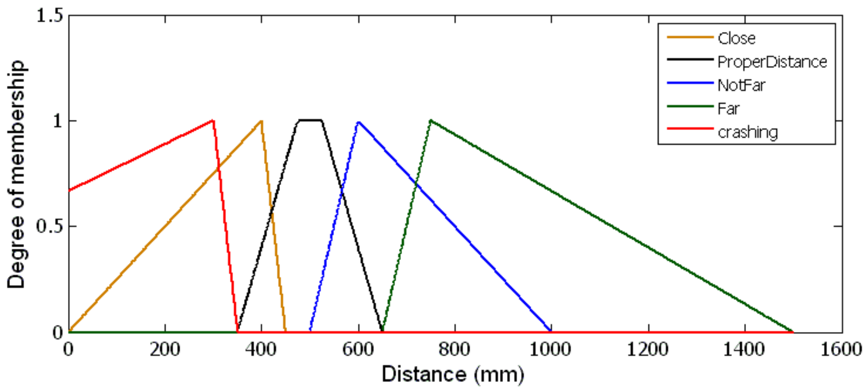


Fig. 4. Fuzzy Sets for Distance.

The fuzzy controller is tuned by an off-line optimization testing a group of randomly chosen fuzzy sets settings.

1.3 Predictive Controller

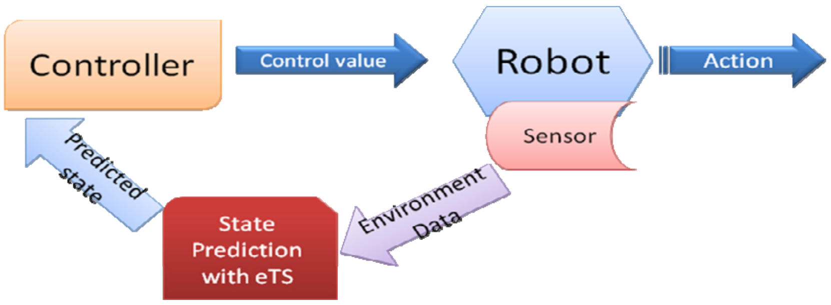
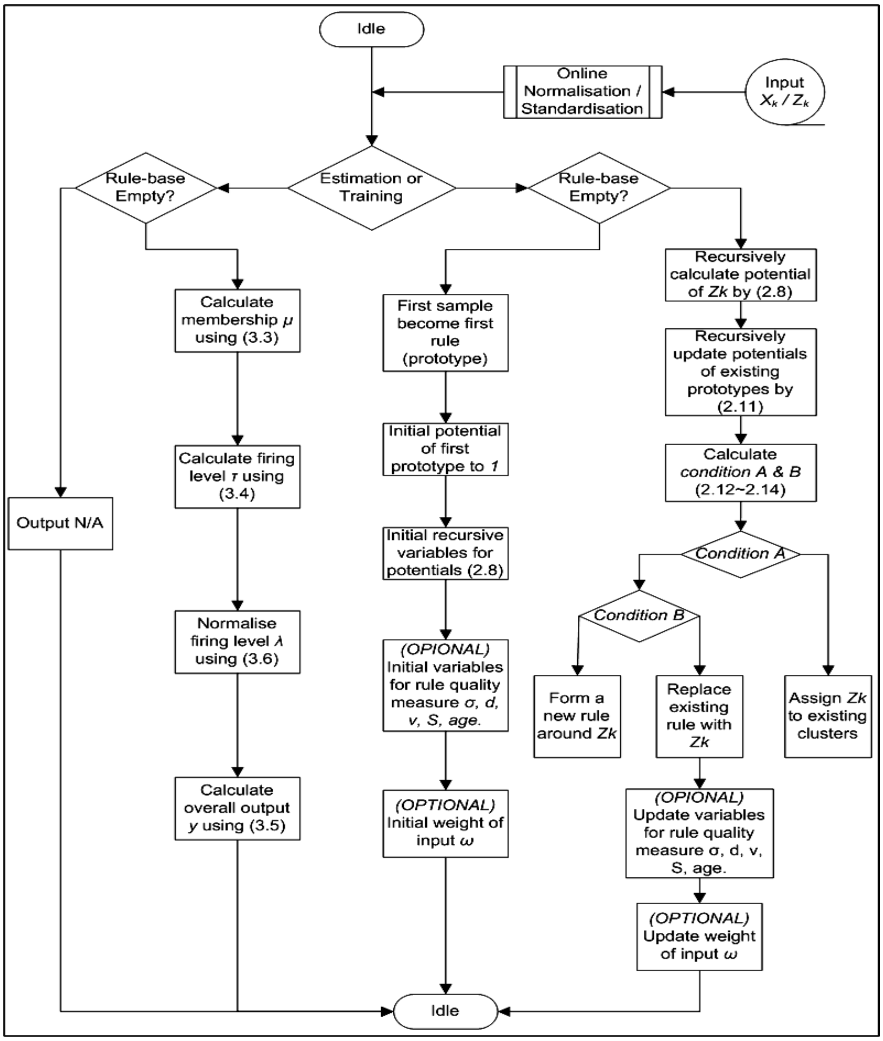


Fig. 5. System Structure of the predictive controller.

In the design of the MBPC, a prediction module is added to the FLC described above. The prediction module is based on eTS [6-8] and aims to predict the distance and angle to the moving target one time instant ahead based on the information of current distance, angle, and velocity of both wheels. These predicted values are then fed to the FLC instead of the readings of the distance and angle at current step. The MBPC then determines the control values in the same way, but based on the predicted values. This leads to minimisation of the tracking error caused by the delay in the response in velocity due the time required by acceleration. The evolving Takagi-Sugeno predictor is described in more details elsewhere [6-8] and is sketched in the following diagram.



Idle
Online Normalisation / Standardisation
Input X_k / Z_k
Rule-base Empty?
Estimation or Training
Rule-base Empty?
Recursively calculate potential of *Zk* by (2.8)
Calculate membership μ using (3.3)
First sample become first rule (prototype)
Recursively update potentials of existing prototypes by (2.11)
Initial potential of first prototype to *1*
Calculate *condition A & B* (2.12~2.14)
Calculate firing level τ using (3.4)
Output N/A
Initial recursive variables for potentials (2.8)
Condition A
Condition B
Normalise firing level λ using (3.6)
(OPIONAL) Initial variables for rule quality measure σ, *d*, *v*, *S*, *age*.
Form a new rule around *Zk*
Replace existing rule with *Zk*
Assign *Zk* to existing clusters
Calculate overall output *y* using (3.5)
(OPTIONAL) Initial weight of input ω
(OPIONAL) Update variables for rule quality measure σ, d, v, S, age.
(OPTIONAL) Update weight of input ω
Idle

Fig. 6. Flow chart of the evolving Takagi-Sugeno (eTS) Predictor Algorithm.

2 Experimental Study

2.1 The Robots

The experiment is carried out with a Pioneer3 DX mobile robot [15] equipped with an onboard PC and a laser ranging device. The laser scans a fan area of 180 degrees and returns the distance and headings to the closest obstacle in this fan area. The detectable range of the laser is [150mm, 10,000mm]. In the experiment, another mobile robot played the role of the moving object to be tracked, following automatically a predefined routine (see Figures 7 and 8). There is no external links such as GPS and the wireless data connection is used only to download data. Thus, the task is performed fully unsupervised by the mobile robot.

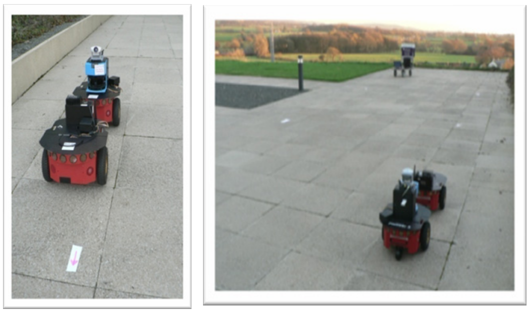


Fig. 7. The robots and the experiment.

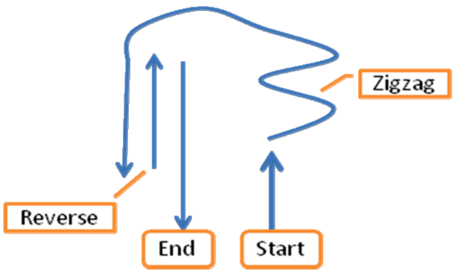


Fig. 8. Route of the target object.

2.2 Experimental Settings

Four variables were measured in real-time:

1 distance to the object;
2 angle to the object;
3 the real velocity of the left wheel
4 the real velocity of the right wheel of the robot being controlled.

The sampling frequency is about 10Hz (100ms per sample). The control values are generated at each sampling interval.

Table 1. An example of the data collected in real-time with the control outputs.

Time	***d*, mm**	**θ,º**	**Real Left, mm/s**	**Real Right, mm/s**	**Ctrl Left, mm/s**	**Ctrl Right, mm/s**
0	205.295	-5.04	-110	-118	-763.923	-799.234
200	201.297	-5.53	-42	-10	-773.8	-812.551
400	207.336	-16.9	-43	-88	-716.216	-835.137
600	216.334	0.068	-93	-133	-750.034	-749.551
801	207.263	10.47	-107	-167	-739.24	-812.532
1001	246.715	3.49	-282	-274	-676.127	-651.682

The experiment was carried out outside Infolab21, Lancaster University, UK. For each of the tested controllers a group of ten tests were carried out along the same test route as shown in Figure 8. During the test the target object performed a series of behaviours including acceleration, deceleration, turning, reversing, etc. The mobile robot that is performing the tracking task has the controllers uploaded on its on-board PC written in C language. The tracking task is performed fully automatically. Only the laser ranging device was used to measure both the angle and the distance. The velocity is measured by the tachometer (odometer) of the robot [15].An example of the measured distance and angle difference to the target is illustrated in figure 9a) and 9b). Several pre-tests were also carried out to find the suitable parameters for the FLC. The distance and the angle to the target were measured in real-time. The discrepancy between the real observation and the target values of distance and angle has been used to calculate the errors. The mean absolute error (MAE), the standard deviation (STD) and the root mean square (RMSE) are used as the criteria for the comparison of the three controllers.

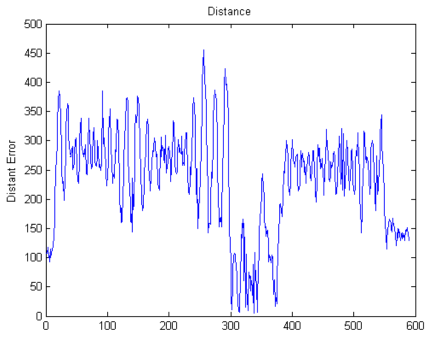


Fig. 9a) Distance measured in real-time.

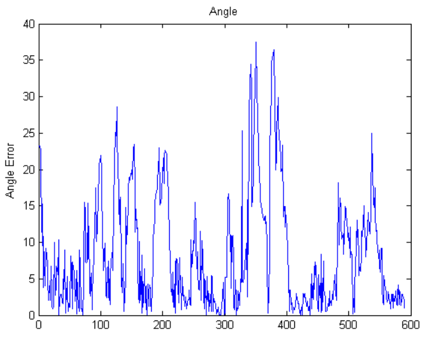
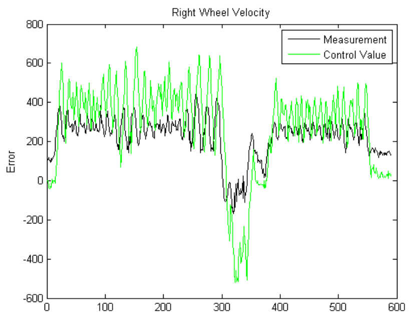


Fig. 9b) Angle measured in real-time.



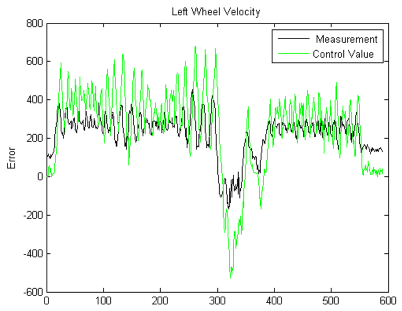


Fig. 10. Control values versus real observations for velocity of the wheels.

3 Results Analysis and Conclusions

The results are tabulated in table 2. They show that the prediction module in the predictive controller has assisted the fuzzy controller to achieve better control precision in terms of the distance and to some extent in terms of the tracking angle minimising the delay in control response. Note that as shown in table 1 and Figure 10, there is some overshot (the control values generated by the fuzzy controller and the predictive controller are larger than the desired velocity of the wheels). This is because it has already taken into account the response delay in time required for the acceleration/deceleration.

Table 2. Result Comparison.

	***d*, mm**			**θ°**		
	RMSE	MAE	STD	RMSE	MAE	STD
First Principles Controller	83.3	129.2	110.1	4.8	9.35	8.14
FLC	70.2	120.3	113.7	4.9	7.43	7.34
MBPC	**65.2**	**112.5**	119.9	**4.8**	7.53	7.09

In table 2, on can see that the angle tracking by the FLC is worse than that of the First principles-based controller. To improve on this aspect, more rules describing the response to different observation in angles can be added to the fuzzy controller to achieve higher control accuracy. Off-line techniques such as ANFIS [16] can be used in order to get the optimal parameters of the fuzzy controller for the task.

In the future, real time image classification [9] and tracking techniques [17] can be integrated with the proposed predictive controller. In this way, image-based information can be used by the prediction module of the MBPC which is expected to further improve the precision.

References

1. Liu P. X., M. Q.-H. Meng (2004) Online Data-Driven Fuzzy Clustering with Applications to Real-Time Robotic Tracking, IEEE Transactions on Fuzzy Systems, vol.12, No 4, 2004, pp.516-523.
2. Astrom K. and B. Wittenmark (1984) Computer Controlled Systems: Theory and Design, Prentice Hall: NJ USA, 1984.
3. Carse B., T.C. Fogarty, A. Munro (1996) Evolving Fuzzy Rule-based Controllers using GA, Fuzzy Sets and Systems, v.80, pp.273-294.
4. Clarke D, Advances in Model-based Predictive Control, Oxford University Press, Oxford, UK, 1994.
5. Babuska R (1998) Fuzzy Modelling for Control, Kluwer Publishers, Dordrecht, The Netherlands.
6. Angelov, P., Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems. Berlin, Germany: Springer Verlag, 2002.

7. Angelov, P., Filev, D., “An Approach to On-line Identification of Takagi-Sugeno Fuzzy Models”, IEEE Transactions on System, Man, and Cybernetics, part B - Cybernetics, vol.34, No1, 2004, pp.484-498. ISSN 1094-6977.
8. Angelov P and X Zhou (2006) Evolving Fuzzy Systems from Data Streams in Real-Time, In Proc. 2006 International Symposium on Evolving Fuzzy Systems, Ambelside, Lake District, UK, IEEE Press, pp.29-35, ISBN 0-7803-9719-3.
9. Zhou. X., P. Angelov, An Approach to Autonomous Self-localization of a Mobile Robot in Completely Unknown Environment using Evolving Fuzzy Rule-based Classifier, First 2007 IEEE International Conference on Computational Intelligence Applications for Defense and Security, April 1-5, 2007, Honolulu, Hawaii, USA, pp.131-138.
10. Yager R R and D P Filev (1993) Learning of Fuzzy Rules by Mountain Clustering, Proc. of SPIE Conf. on Application of Fuzzy Logic Technology, Boston, MA, USA, pp.246-254.
11. Kailath, T, Linear systems, Prentice Hall, US, 1980.
12. Psaltis D, A Sideris, A A Yamamura (1998) A Multilayered Neural Network Controller, Control Systems Magazine, vol. 8, pp. 17-21.
13. Angelov P P (2004) A Fuzzy Controller with Evolving Structure, Information Sciences, ISSN 0020-0255, vol.161, pp.21-35.
14. Andersen H.C., F.C. Teng, A.C. Tsoi (1994) Single Net Indirect Learning Architecture, IEEE Transactions on Neural Networks, v.5 (6), pp.1003-1005.
15. Pioneer-3DX (2004) User Guide, ActiveMedia Robotics, Amherst, NH, USA.
16. Jang, J.-S., RANFIS: adaptive-network-based fuzzy inference system, IEEE Transactions on System, Man, and Cybernetics, vol.23, No3, 1993, pp.665-685.
17. Angelov P., R. Ramezani, X. Zhou, Autonomous Novelty Detection and Object Tracking in Video Streams using Evolving Clustering and Takagi-Sugeno type Neuro-Fuzzy System, 2008 IEEE Intern. Conf. on Fuzzy Syst. within the IEEE World Congress on Computational Intelligence, Hong Kong, June 1-6, 2008, to appear.

Floating Car Observer – Approaches for Traffic Management Strategies by Analysing Oncoming Vehicles

Franziska Wolf[1], Sebastian Naumann[2], Christoph Engel[3] and René Schönrock[4]

Institut f. Automation und Kommunikation e.V. Magdeburg
Werner-Heisenberg-Str. 1, 39106 Magdeburg, Germany
[1]Franziska.Wolf@ifak.eu, [2]Sebastian.Naumann@ifak.eu
[3]Christoph.Engel@ifak.eu, [4]Rene.Schönrock@ifak.eu

Abstract. For traffic management systems, the knowledge of urban traffic conditions is essential. The strategy of the Floating Car Observer (FCO) is to collect traffic data via moving traffic sensors. The data of oncoming cars and trucks is detected using a travelling public transport vehicle. Different acquisition strategies are evaluated to establish a low cost application, capable of real time usage. Using the reflection of the infra-red emitter on number plates, traffic data information about the speed and the average density of oncoming traffic is acquired. The properties of the hardware prototype and the implementation of the software strategies are discussed and first research results are presented.

1 Motivation

Today knowledge of the inner-city traffic state is a prerequisite for a functioning traffic management both in private and public transport. Data as speed and traffic density is more and more used for applications on IT systems in public traffic management. These services provide dynamic arrival time prognosis for passengers or sophisticated traffic management strategies for public and private transport. Therefore widespread acquisition methods for spatial and temporal traffic data are desired. The core problem offering such information services is the absolute knowledge of current and future traffic conditions. Traffic processes can be described by several parameters, such as the time headway τ, the traffic flow q and the local speed vl. The spatially and temporarily complete automatic acquisition of these parameters is desirable, however due to economical and technical reasons it is often not achievable.

1.1 Current Methods of Measuring Traffic Data

Currently the acquisition methods for traffic management systems rely on local detectors such as induction loops, infra-red and video detectors at fixed cross sections ([1] [2]) of a road (see ① in Fig. 1). Unfortunately, these local measurements do not allow the clear determination of traffic states. A second method is the instantaneous observation (see ② in Fig. 1), however except for observation flights (see ④ in Fig. 1) this method has only little practical relevance.

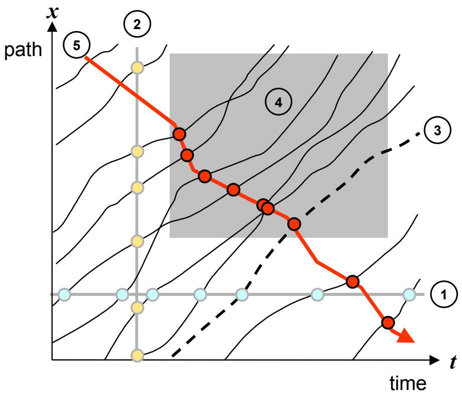


Fig. 1. Path time diagram of traffic course parameters.

An alternative to measuring traffic parameters with stationary devices is the use of Floating Car Data (FCD). A vehicle is used as a mobile traffic sensor for permanently transferring its own traffic data such as speed and position. As shown in ③ in Fig. 1, kinematical parameters of single vehicles such as path x, velocity v and acceleration a can be measured and travel times along road sections can be determined directly. Thus FCD receives a small amount of information because the data only describes the vehicle's own driving course and does not represent traffic conditions in a broad spectrum. If stuck in a traffic hold-up, the usage of a FCD vehicle can not offer information about the length of the hold-up and the estimated time lag for the travel time prognosis. FCD is dependent on the traffic situation which it is going to measure.

1.2 Floating Car Observer

A broader data acquisition method will be the automatic observation of oncoming traffic in order to obtain traffic data. The main idea is that public transport vehicles shall carry devices which can observe traffic conditions on the opposite oncoming lanes. Equipped with low-cost devices, vehicles will be used as Floating Car Observers (FCO) monitoring their road environment and traffic flows along the public transport route network [3]. The advantage of the FCO approach is shown in Fig. 1. The ongoing course ⑤ shows that traffic density k(x) and speed v(x) can be measured directly. The FCO measures the traffic data without being part of the traffic situation. For example the starting points and the length of traffic hold-ups on the oncoming traffic lanes can be observed and used to improve arrival time prognosis of the public transport distributed by various services.

By using this data, traffic diversions can be adapted for present traffic situations.

In the future the traffic data acquired by the FCO module will be sent by GPRS to a traffic management control centre. Additionally the FCO's geographical position, driving direction and speed will be used to generate a continuous overview of the network`s current traffic conditions. In thefollowing, the simulation set up to evaluate various potential FCO devices is presented. It will be tested if the device data can be

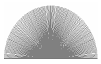
used for signal processing algorithms for speed and traffic density calculation. The most promising technical strategy will be used for a hardware and software prototype of a FCO.
In this paper, methods of observing vehicle flows in order to gain information about traffic conditions will be outlined. In Section 2 the theoretical background of a system observing oncoming traffic using a Floating Car Observer (FCO) is describes. The simulation framework to evaluate different sensors to set up a hardware prototype is outlined. Furthermore it can be shown that the theory of observing oncoming traffic can be realised. In Section 3, a description of the FCO's hardware design is presented. The strategy of using an infra red (IR) emitter combined with a camera to capture specific traffic observing images will be described. The currently developed image processing algorithms to detect traffic vehicles will be described along with the expected results. A conclusion and an outlook at the upcoming steps will conclude the article.

2 FCO Design - Simulation and Evaluation

In preparation for running a computer simulation of the FCO, a model of a public transport vehicle can be equipped with simulated FCO devices such as laser scanner, ultrasonic and video camera systems. These simulated and evaluated devices are already broadly used applications for supporting driver-assistance ([4][5]) and safety applications [6] for example in nighttime situations [7]. However so far they are hardly used for oncoming traffic detection and observation. By altering the configuration of the FCO devices, the different detection rates of the sensors are measured while the vehicle travels through a virtual public transport network, built using a microscopic traffic model of a real existing traffic scenario. Based on empirical values, the simulation evaluates whether the resolution of the devices is appropriate to achieve a detection of the oncoming traffic. For example it was measured to which quantity and distance the FCO sensor types could capture the oncoming vehicles due to the frame rate, detection field and measure values.

In order to equip vehicle fleets of public transport companies with FCO devices, also economical aspects of the systems have to be considered. In this evaluation the sensor types are simulated in a FCO simulation and evaluated as follows:

Table 1. Tested sensor types simulated and evaluated on usability for FCO development.

FCO-sensor type	Function	Field of detection and measure values (Hz)	Range (m)	Disadvantages for FCO usage	Advantages for FCO usage
Light scanner	The sensor sends a directed laser light beam. The reflection is used to calculate distance	1, 4, 16, 64	20	Discrimination of traffic vehicles hardly achievable Only 1-D resolution	Low acquisition costs
Laser scanner	The laser scanner contains a rotating unit, which sends pulsed laser beams	 19, 38, 75	150	Discrimination of traffic vehicles hardly achievable High acquisition costs	Wide range of 150m High spatial and temporal resolution
Ultrasonic/ Radar	The ultrasonic sensor sends sound/ electromagnetic waves where the detection field forms a club. The reflection is used to calculate distances	10	10 / 50	Discrimination of traffic vehicles hardly achievable Calibration of measuring field required	Low acquisition costs Much experience of this technology in traffic monitoring
Active video camera	The camera system uses pulsed IR lights to recognise retro-reflecting materials, e.g. number plates.	30 to 60	NN (weather conditions)	Detection range depending on weather conditions	Low acquisition costs Discrimination of traffic vehicles achievable

The evaluation showed that the ultrasonic system has an insufficient usability due to the limited range and specific formed detection field. The laser and ultrasonic sensors may need a second sensor to enable a sufficient range and detection accuracy for traffic data capturing. Due to high acquisition costs, the laser scanner can also not be used for broad traffic monitoring. The camera approach uses positions and sizes of detected objects for data reconstruction. Due to better discrimination of detected objects, a camera system is higher rated than a laser or an ultrasonic system.

The camera based system has been rated to be the most promising system due to its technical properties, low cost and mounting capabilities. The main advantage of the camera system is its potential for detecting particular objects, cars in this case. However image streams received by a camera system normally are not very significant: but by using an active exposure system the images' quality and therefore the initial position for object recognition can be improved. The usage of such camera systems for traffic monitoring has been proposed, for example for automatic number plate recognition systems [8]. A special system using a similar strategy was originally developed for the recognition and safety improvement of non-motorised, vulnerable road users such as pedestrians or cyclists [9][10]. Equipped with special coated retro-reflectors they could be recognised in IR exposed image streams by object detecting algorithms. Normally cars and trucks are not equipped with such special coated retro-reflectors of this quality, but they provide a slightly retro-reflecting number plate. These number plates have proportions from height to width which are in most countries unique among road signs and other reflecting objects in the road environment. It is essential to identify these reflecting number plates because their shapes will be used for traffic data calculation. By scanning the image streams of the camera, the size and movement of the detected number plates are used for vehicle monitoring and speed reconstruction using geometrical strategies such as intercept theorems.

The following section describes the set up of a hardware and software prototype of a FCO device for visual traffic observation. The usage of embedded infra-red light for detecting number plates of cars in video streams will be dealt with. The set up and configuration of the specified camera system is also described. The current research strategy of the vehicle detecting image processing and speed reconstruction algorithm will be described and implementation methods will be outlined. First results based on images of the simulation are presented.

3 Technical Design of the FCO Prototype

The architecture of the FCO prototype is described in Fig. 2 and contains the following modules:

- CMOS-camera with optical filter for achieving the raw image stream
- Infra-red-emitter for lighting the scenery
- Micro controller

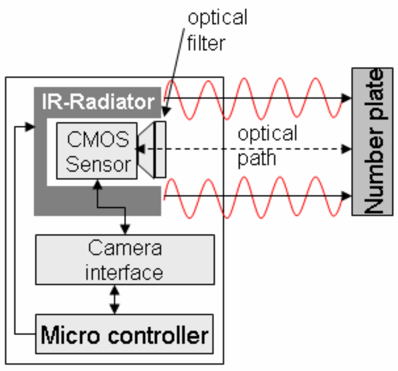


Fig. 2. FCO camera module for recognizing number plates.

The low cost camera module is based on CMOS technology and includes a 12 bit ADC (analogue-to-digital converter). 30 monochrome 12 bit raw images with a resolution of 648*488 pixels can be captured per second. The emitted IR light has a narrow-banded spectrum with a central wavelength of 950 nm. This compact spectrum of the IR emitter enables the usage of a special optical filter in order to improve the quality of the images received by the camera system. As pointed out in Fig. 3 the optical filter reduces the receivable spectrum of the camera to the wavelength from 900 nm to 1000 nm. That reduces the distracting sun light in order to improve the signal-to-noise ratio and therefore makes it easier to detect objects which reflected the IR light..

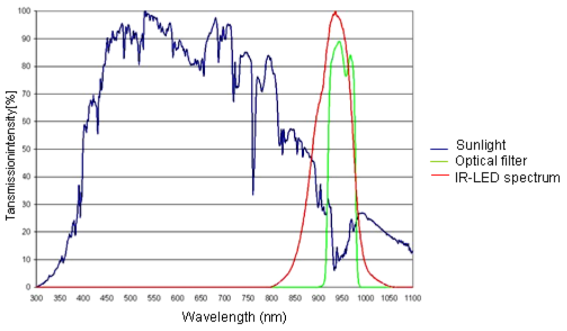


Fig. 3. Spectral sensibility of CMOS sensor with and without optical filter.

The 32 bit ARM microcontroller synchronises the switching and the power control of the IR LED emitter. Provided by the digital camera via a serial I²C interface, the micro controller also runs the image processing and object recognition and tracking algorithms. An overview of the software design and the main modules is described in the following section.

4 Image Processing and Object Recognition Strategies

In this section the basic image processing algorithm will be outlined. The algorithm uses a 12 bit raw data image for input provided by the camera system. During the exposure, the aperture of the camera and the IR emitter are synchronised with each other in order to create video images with good detectable objects. The on and off switching of the infra-red emitter can be synchronised to the camera in two different ways, shown in Table 2.

Table 2. Two methods of IR embedded image processing.

	Function	Image	Advantages	Disadvantages
Stripe Algorithm	Every second line of a frame is exposed by the infra-red emitter. Generates bright and dark striped pattern for reflecting objects.		Small amount of data	Lower resolution. Distortion of reflections possible.
Two Frame Algorithm	Every second frame is exposed by the infra-red emitter. Generates bright patterns for reflecting objects		Allows a more accurate calculation	High amount of data

Both algorithms are able to achieve specified video frames. The *Stripe Algorithm* allows a fast processing rate, but a lack of exposure can lead to an insufficient object recognition. The *Two Frame Algorithm* allows a more accurate calculation but needs double the amount of data of the first strategy. It has to be researched which strategy is the best for the following object recognition algorithm shown by the application flow in Fig. 4. The process flow starts with the exposure of traffic scenery to create raw data images to enable clearly visible highlighted areas based on IR reflection on vehicle number plates. These rectangular areas have to be detected by the algorithm in each frame. The next step is to track each of these rectangles from frame to frame. This is done using position and size of the rectangles as similarity measures. Traffic data of the detected cars can be computed using the number plates' course through the images. It has been analysed for the image processing algorithm done by the micro controller, whether the usage of Java or the specified Open CV libraries for C++ can accomplish these tasks best. These steps are marked red in the application flow.

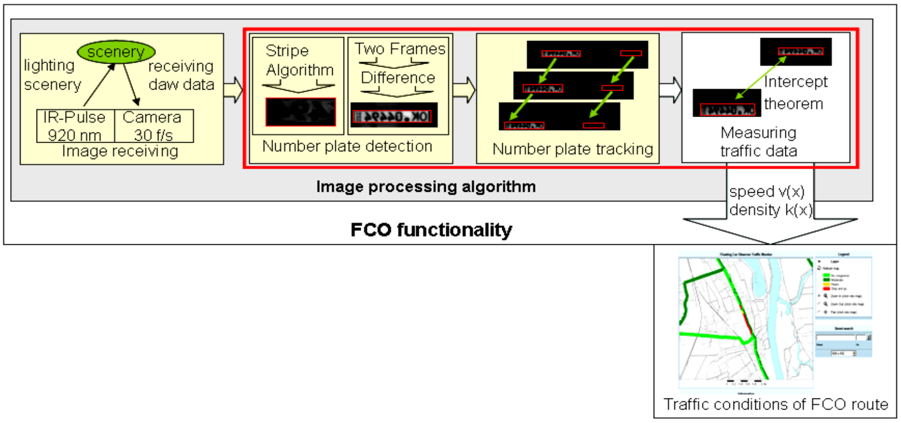


Fig. 4. Vehicle detection and data reconstruction of FCO.

The following pseudo code presents the steps of the rectangle detection in the raw data images, the number plate tracking and the measurement of the traffic data. For the following FCO algorithm in pseudo code, the following variables are to be mentioned:

- lrf, lrc – list of the rectangles found in the current frame, the rectangles' course
- h(NPfn), h(NPf1) – height of the rectangle in the first/ last captured frame
- h(NPR) – real height of a number plate
- fd – focal distance of the camera system used
- hit - threshold condition fulfilled

FCO algorithm describing the number plate detection, tracking and traffic data calculation.

```
FCO ALGORITHM
for (every recorded frame) do
{
  while (the lower right corner of the frame is not reached)
  {
    browse image;
    if (hit)
    {
      create rectangle with a dimension of 0;
      while (new hit in the area of the rectangle)
      {
        rectangle increased until includes new hit
    }
    }
    if (r NOT (ratio 52:11 OR 34:20))
    {
      remove r;
    }
  }
  store all remaining rectangles of frame as list lrf;
  for (the last added rectangle (rlast in lrc)) do
  {
    for (every rectangle (r in lrf)) do
    {
      if ((dist(center of rlast and r)) < thresholddist)
      {
        add r to lrc;
        remove r from lrf;
      }
    }
  }
  for (every rectangle remaining in lrf) do
  {
    create new list lrci with rectangle as first element;
  }
  for (every list lrci without a new rectangle added,
       rectangle course is assumed as finished) do
  {
    s1 = h(NPf1) / h(NPR) * fd;
    sn = h(NPfn) / h(NPR) * fd;
    t = tn - t1
    s = sn - s1
    v = s/t;
    print out v;
    delete lrci;
  }
}
```

In order to track the patterns in each image, previous knowledge about the movement of the number plates has to be used. Because of the high frame rate of up to 30 frames/sec, the position of a rectangle in one frame can be assumed similar in the next frame. While tracking the rectangles of the video frames, the change of size and position of the rectangles from frame to frame is used to reconstruct the speed of the detected vehicles using intercept theorems.

Recent test results show that the algorithm works well for simulated streams and encouraging results for first captured images of the FCO hardware prototype could be achieved.

Before the FCO module can be integrated into a public transport vehicle, broad field tests will be necessary. For this a test vehicle is equipped with the required measuring and computing technology. The equipped test vehicle will be used for broad test scenarios covering data acquisition in varying seasonal weather situations at selected places.

5 Conclusions and Future Prospects

This paper offers a new approach to acquire data of actual traffic situations using a Floating Car Observer (FCO). The FCO captures data of the traffic situation of oncoming vehicles, such as positions and lengths of traffic hold-ups. The captured data will be used to enhance travel time prognosis for public traffic management systems. A simulation framework was used to evaluate different traffic observing devices (laser, ultrasonic, camera) in order to to set up a traffic monitoring FCO prototype. The evaluation derives from detection rates, but also from real time capability of the computing process and economical aspects of the devices. The system chosen to set up a FCO prototype comprises of video processing and infra-red emitting modules. The reflection of the infra-red beams on vehicles' number plates shall be used to gain information about speed and average density of the oncoming traffic. The algorithm presented detects the highlighted patterns of reflection and tracks them. Using intercept theorems, the courses of the number plate reflections are used to extract traffic data such as traffic density and speed. The algorithm is currently implemented and tested on various platforms. In future this traffic information will improve management systems of the public transport to inform passengers about broad traffic situations and estimated arrival times.

At present the set up of the FCO prototype is used for image capturing of traffic scenarios. These real images have been used together with synthetic simulation images to develop the presented algorithm. In this respect currently different platforms and system frameworks are tested for the implementation of the algorithm. The software has to be evaluated on different traffic road situations, seasonal and weather conditions. Analysis of the detection rates under different weather and seasonal conditions will reveal their implementable capacities and limits. Additionally it will be examined as to which traffic condition the system can be best applied to and which kind of traffic data is the most significant for traffic management systems. Consolidating the most effective data into global strategies will be the next step towards advanced public and private traffic management systems.

References

1. Minakata, T. Taniguchi, Y. Shiranaga, H. Nishihara, T.: Development of Far Infra-red Vehicle Sensor System, Sei Technical Review – English edition, ISSU 58, Sumitomo electric industries LTD, Japan (2004) 23-27
2. Sumiya, N.; Fujihira, K.; Kamijo, S.: Incident detection system by sensor fusion network employing image sensors and supersonic wave, Proceedings of the Intelligent Transportation Systems Conference, 06. IEEE Volume, (2006), 1066 - 1071
3. Hoyer, R.; Czogalla, O; Naumann, S.: Improvement of real time passenger information by floating car observers, Proceedings of the 13th World Congress on Intelligent Transport Systems, London (2006)
4. Nagata Akihito, Ozawa Shinji, Yanagawa Hirohiko, "Approach Vehicle Detection from the Rear Side Using Fish-eye Camera", Papers of Technical Meeting on Information Processing, IEE Japan, Vol. IP-07; No. 1-13; (2007), 65-70
5. Wannes van der Mark, Dariu M. Gavrila, "Real-time dense stereo for intelligent vehicles", Intelligent Transportation Systems, IEEE Transactions on Publication Date: March 2006, Vol 7, Issue 1, 38-50
6. Lages, Dr. U.: Laser Sensor Technologies for Preventive Safety Functions. IBEO Publication 12th International Symposium ATA EL 2004
7. Graf, T.; Seifert, K.; Meinecke, M.; Schmidt, R.:Advanced Night Vision Systems Concepts, Conference Title 12th World Congress on Intelligent Transport Systems, 12th World Congress on Intelligent Transport Systems, San Francisco, 06. -11.11. 2005, Proceedings
8. Sulehria, H.; Zhang, Y.; Irfan, D: Mathematical Morphology Metodology for Extraction of Vehicle Number Plates, International Journal of Computers, Issue 3, Volume 1, 2007
9. Hoyer, R.; Schönrock, R.: Approach to the Recognition of non-motorised Road Users, Proceedings of the 8th World Congress on Intelligent Transport, Sydney (2001)
10. Hoyer, R.; Herrmann, A.; Schönrock, R: Modelling of vehicle actuated traffic lights. In: Schulze, Th. Et al.: (Hrsg.): Simulation und Visualisierung 2004. Society for Modeling and Simulation International SCS-European Publishing House, Erlangen, San Diego, 2004, S. 359 – 369; ISBN 3-936150-30-3.

A Solution to the Parallel Parking Problem

Sinan Öncü and Mürvet Kırcı

Faculty of Electrical & Electronic Eng.
Technical University of Istanbul, İstanbul, Turkey
oncusi@itu.edu.tr, ucerm@itu.edu.tr

Abstract. Nowadays parts of a car have been designed form in which make us easy driving. One of the driving problems is parallel parking. This study gives a solution with an algorithm to the parallel parking problem. Simulation of the solution including all cases in the parallel parking has been made based on this algorithm. Implementation of the simulation has been made as a robot car. The approach has been integrated into an automated parking system, and implemented a modified car. It shows the potential to be integrated into automobiles.

1 Introduction

This work is related with a mobile robot car (MCR) solving parallel parking problem. Parallel parking is a method of parking a vehicle in parallel to the other car. It is proposed to develop a parking helper system which is possible apply to the real car. Autonomous parking is necessary sensing environmental conditions, calculating the dynamic paths. In this work implementation of parallel parking algorithm on mobile robot car will be presented. The parallel parking involves many problems such as recognition of driving circumstances, maneveuring and vehicle control. In the literature there are some works related with this subject. Yasunobu and Murai have purposed a controller based on hierarchical fuzzy control.[1] Jenkin and Yuhas have reported a simplified neural network controller.[2] These algorithms are based on the kinematics data to formulate intelligent controllers.Lo et.al have presented an automated parallel parking strategy for a vehicle like robot[3]

In our system works in three step; scanning, calculating and parking. In scanning step the requiring parking area is scanned by infra red sensors. Then in second step the car is moved to a suitable parking area where the parking maneveurs can begin. The required maneveurs for the parking step are calculated dynamically. In the last step MCR follows the path to the suitable location. Parallel parking algorithm and implementation on the MCR is discussed in section 2. In the section 3.application of the algoritm is explained. The application consists of two steps. First,a visual computer program running in PC environment is developed in order to simulate the approach above. In the second part, hardware implementation of the parallel parking algorithm on a MCR and modules that are used will be explained.

2 Parallel Parking Algoritm

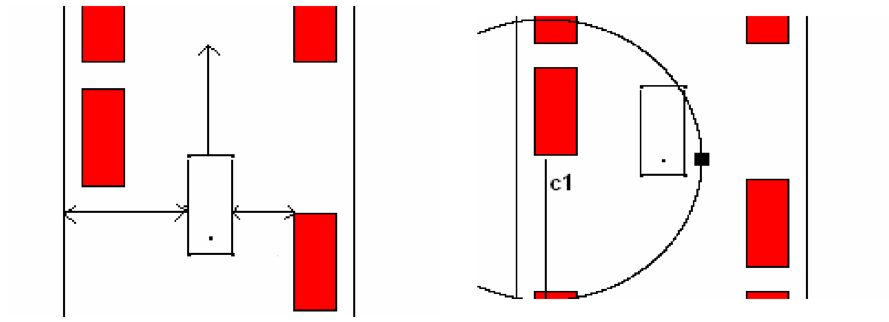
A mechanical system consists of different compenents. In this case where the deformations of the body are significant compared to the motion of the body as a whole body is called a rigid body The idealization allows us to reduce the equations needed to describe a rigid body.[4] In this work the system has been assumed as a nonholonomic system.[5-7] In this method datas from the sensors and closed loops for dynamic circumstances are used. In this section it is given an algorithm to solve parallel parking problemof MCR.There are three step in this method.

a) Scanning Step

In scanning step it is determined potential places using closed loop controls so that it does not hit around objects and borders. MCR is parallel y axis in initial case. Every action loop for parking is correspond to a displacement 1cm direction in +y axis. As the angle of the steering was zero degree, x axis and the angle of the position of the MCR do not change. The action of the MCR is forward. It seeks a convenient parking place acting in low velocity. Sensors mounted to the left and right side of MCR give necessary knowledge for seeking. The sensor in font of the MCR also gives some knowledge about a possible accident undesired will be done. After every action loop correspond to a scanning 1cm has been finished the variables of the parking area are updated. Distance between MCR and other parked vehicles are also noted in memory. When the variable of the parking area has exceeded a fixed value specified with dimensions of MCR, this fixed value is pointed as a potential target. The center of the circle C1 is settled on a line perpendicular to this point on the coordinate plane.The center of the circle C1 is calculated by following formulas:

$$C1x = fx + \Delta x - R$$
$$C1y = fy \qquad (1)$$

Where Δx displacement. If parking area has suitable dimensions it will be passed to the calculating step



(a) (b)

Fig. 1. (a) The car is seeking a park place. (b) The center of the circle C1 is being calculated.

b) Calculating Step

When the convenient parking area has been found the MCR passes to this step. The aim of the MCR is to arrive a good position for beginning car parking maneverous. This Position is the point specified as a target in the scanning step. Firstly, it is defined a circle which is described points MCR be able to arrive on coordinate plane. Coordinates of the center of this circle are calculated as follows:

$$C2x = fx + R$$
$$C2y = fy \tag{2}$$

To find a point is an intersection of these circles MCR acts forward. Distance between two circles is 2R at this tangent point. Therefore tangent point is

$$\sqrt{(C1x.C2x)^2 + (C1y.C2y)^2} = 2R \tag{3}$$

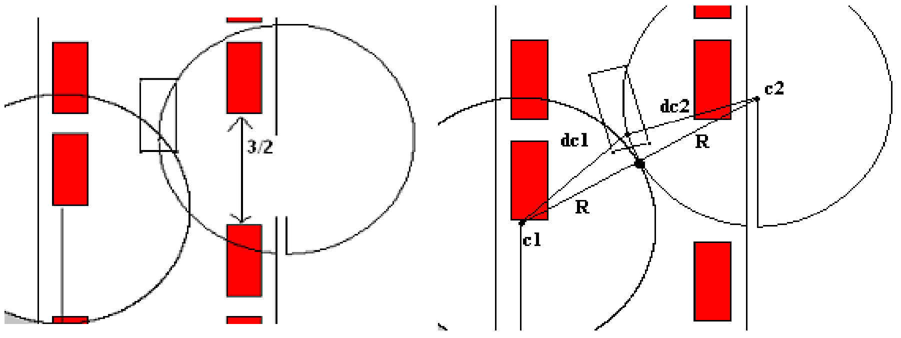


Fig.2. (a) The MCR has found the convenient park place. (b) Parking Maneveurs are being calculated.

Then MCR arrives to the tangent point acting backward with a specified steering angle. At tangent point following equations are satisfied

$$dc1 + dc2 = 2R$$
$$dc1 = \sqrt{(fx - C1x)^2 + (fy - C1y)^2}$$
$$dc2 = \sqrt{(fx - C2x)^2 + (fy - C2y)^2} \tag{4}$$

When MCR has reached to the tangent point C1,C2 and F(fx,fy) are situated on the same line. Then MCR realizes steering maneverous inverse direction with same angle value. So it follows arc which is occurred by C1 and passes between the circles Finally the MCR is backed up with the determined steering angle until its orientation angle is again parallel to the y-axis. The sideways displacement obtained can be observed in the figure 2 (a) and (b).

c) Parking Step

In the parking maneveurs the same approach is used that was explained calculating step with sideways displacement resulting in the final location where the MCR is desired to be lacated after the parking is completed. Different from the calculating step greater steering angle values are used during parking maneveurs to increase maneveuring capability of the car. The parking maneveurs for the given scenerio are shown in the following figure 3.

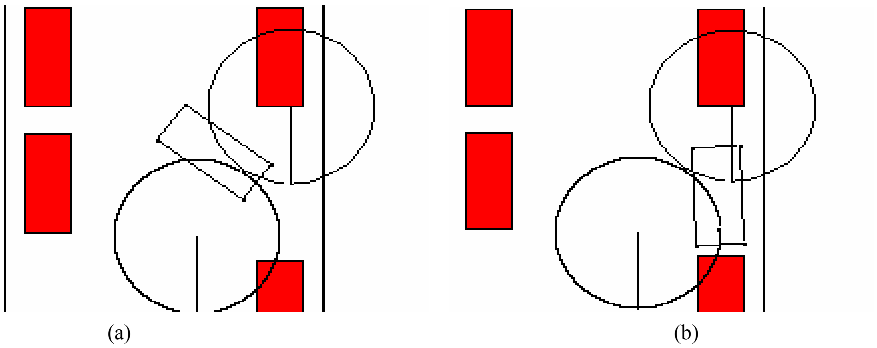


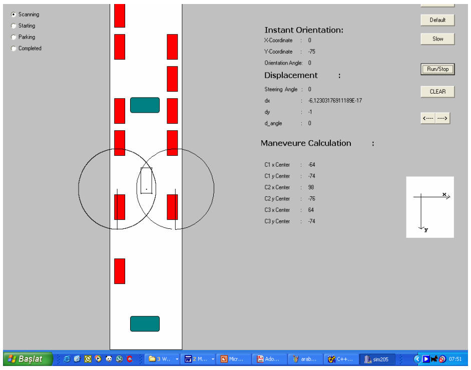
Fig.3. (a) The MCR is doing parking maneveurs. (b) The MCR is arriving to the final location.

3 Application

Application of the parallel parking algorithm on a MCR consists of two steps. First, a visual computer program running in PC environment is developed in order to simulate the approach above. The program allows the user to generate random scenarios for parallel parking. In the first section of this chapter the simulation program is explained In the second part , hardware implementation of the parallel parking algorithm on a MCR and modules that are used will be explained.

3.1 Simulation

In the simulation program, physical model of a MCR is generated first. The coordinates of the reference point F(fx,fy) is the main control point. The corner points of the vehicle are generated with referencing to the reference point.The kinematic model is implemented on this model. After each cycle of the main program loop, coordinate parameters of the reference point are calculated depending on the previous values and steering angle. On this car model four sensors were modelled which obtain the enviromental data from the simulation screen by checking color changes Which is an abstraction of the real world operation of sensors.



Instant Orientation:
X-Coordinate : 0
Y-Coordinate : -75
Orientation Angle: 0

Displacement :
Steering Angle : 0
dx : -6,12303176911189E-17
dy : -1
d_angle : 0

Maneveure Calculation :
C1 x Center : -64
C1 y Center : -74
C2 x Center : 98
C2 y Center : -76
C3 x Center : 64
C3 y Center : -74

Fig.4. A screen shot from the from the simulator.

A random scenario generator program has been developed in order to simulate the environment. For decision making tasks a program simulation te controller is generated which obtaines the enviromental data, makes decision and sends the control signals to the motion module Simulator. In the figure.4. a screen shot from the simulator running in PC enviroment is seen.

3.2 Hardware

Realizing the parallel parking algorithm on a MCR requires obtaining enviromental data, making decision and driving the car to perform the task.The requirements are satisfied using three modules of hardware:

1.) The sensor module
2.) The Microcontroller module
3.) The motion module

For the parallel parking of the MCR SharpGP2D120 distance measuring infra-red sensors were used to obtain enviromental data.These IR sensors operate between 4~30cm distances which is sufficient for our 20x50cm robot. The experiments showed that sensor data resolution is sufficient in the interval 4-20cm which covers the critical ditances for the parallel parking problemof a MCR wit dimensions 20x50cm. In this project Microchip PIC16F877 micro controllers were chosen because of their large memory size, built in ADC, low price and incircuit programming capability. The motion module consists of two stepper motors and their driver chips, the ULN2003A. Stepper motors are used for both steering and driving of the robot because accurate control on the motion of the robot is needed for the controller software to predict the displacements of the robot body.

A MCR has been developed with 20x50 cm dimensions. This a four wheeled rear-driven car with front-steering wheels. It is equipped with 2 stepper motors. One of the stepper motors is used for driving and the other is used for controlling the steering system. 4 IR sensors are placed on the robot.

4 Conclusions

The parallel parking algorithm introduced in capter 2 has been realized on the car-like robot (MCR) using closed loop control and path following methods. The approach was verified with experimental studies. The method can be used in different sized cars by modifying relevant parameters (car length, width etc.) in the controller software.

The approach solves the parallel parking problem for a general case. The approach can be implemented on more complex scenarios which include various parking spaces with the use of hybrid sensor systems for better modeling of the environment.

References

1. Yasunobu, S., Murai, Y., Parking Control Based on Predictive Fuzzy Control in Proc. Of IEEE Int. Conf. On Fuzzy system, Jun (1994),pp 1338-1341
2. Jenkins, R.E., Yuhas, B.P., A simplified Neural Network solution Through problem decomposition:The case of the truck Backer-upper, IEEE Tran.on Neural Network, (1993, Vol.4.,No.4.
3. Lo, Y. K., Rad, A.B., Ho,M. L., Automatic Parallel Parking , IEEE Journal of Robotics and Automation, (2003)
4. Pars, L., A Ttreatise on AnaliticalDynamics, Heinemann, London, (1965)
5. Laugier,C., Paromtchick, I.E., Garnier, Ph., Sensor Based Control Architecture for a Car-like Vehicle, International Conference on Intelligent Robots and Systems, October, (1998)
6. Jiang, K., Seneviratne, L.D., A Sensor Guided Autonomous Parking System for Non-holonomic Mobile Robots, International Conference on Robotics & Automation, May, (1999)
7. Geng, G., Geary, G. M., Development Of A Path Planning System For AGVs, Industrial Technology, (1994), Proceedings of the IEEE International Conference

Author Index

Proceedings of the 2nd International Workshop on
Intelligent Vehicle Control Systems - IVCS 2008
ISBN: 978-989-8111-34-0
http://www.icinco.org