EURASIP Journal on Embedded Systems

Embedded Systems for Intelligent Vehicles

Guest Editors: Samir Bouaziz, Paolo Lombardi, Roger Reynaud, and Gunasekaran S. Seetharaman

Embedded Systems for Intelligent Vehicles

EURASIP Journal on Embedded Systems

Embedded Systems for Intelligent Vehicles

Guest Editors: Samir Bouaziz, Paolo Lombardi, Roger Reynaud, and Gunasekaran S. Seetharaman

Contents

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 29239, 4 pages
doi:10.1155/2007/29239

Editorial

Embedded Systems for Intelligent Vehicles

Samir Bouaziz,[1] Paolo Lombardi,[2] Roger Reynaud,[1] and Gunasekaran S. Seetharaman[3]

[1] *Institut d' Electronique Fondamentale, Université Paris-Sud XI, Bâtiment 220, 91405 Orsay Cedex, France*
[2] *Institute for the Protection and Security of the Citizen, European Commission Ü Joint Research Centre, TP210, Via Fermi1, 21020 Ispra, Italy*
[3] *Department of Electrical and Computer Engineering, Air Force Institute of Technology, Dayton, OH 45433, USA*

Received 12 June 2007; Accepted 12 June 2007

There is a growing need for some kind of personal driving assistant which is likely to become more acute as the free, independent, and very mobile baby boomers continue to age. Up to 86.5% of the US workforce commutes to work every day through personally owned automobile, often driving alone a car, a van, or a truck, at times in a commute as long as two hours. Urban planning and life style are among the factors, not likely to change in the near future, that make one choose private automobiles over public transportation. Recent developments in Europe have triggered significant increase in car-ownership rates in most of the 27 states of the current enlarged Union from 1995 to 2001—a trend that continues. In short, the man hours spent behind the steering-wheel are continually increasing worldwide, accounting for a lost productivity and increased safety hazards. At the same time, the activities that a driver could do from an isolated automobile have increased, for example, cell phones, televisions, listening to books, mobile computing, among others. If personal assistants can help alleviate some of the driving tasks, it could partially relieve the driver from the required intense attention to the road conditions. It can also help the steadily aging members of the population for whom a personal automobile is the only means of transportation. Intelligent personal driving assistants will improve safety, productivity, and the quality of commute.

Intense research in intelligent transportation systems, over the past 20 years, has produced a wealth of insights into the design challenges and applications of intelligent vehicles. A broad spectrum of published literature in this focus cover smart control, communications, and sensor systems residing on-board a vehicle rather than being centralized in traffic management headquarters or being included in road infrastructures. While infrastructural solutions have remained almost exclusively within the reach of governmental investors, the end-user benefits offered by intelligent vehicles technology are poised to attract private capitals from the vehicle manufacturing industry and eventually hit the consumer market. There is a rich set of opportunities for acquisition, trading, and management of location and time tagged information to support the next generation of intelligent vehicles. Intelligent vehicles advocate for autonomous capabilities, self-regulatory, and self-repairing systems to improve safety, driver comfort, and efficient use of infrastructures. Geographical-position-systems- (GPS-) based navigation, computer vision, radar and laser range sensors, adaptive control, and networking, among the others target problems like traffic flow control, smart communications, pedestrian protection, lane departure monitoring, smart parking facilities, and advanced driver assistance systems (ADAS).

We would require exceptional standards of reliability, quickness of response, and fault-tolerance from these systems, before we accept to delegate part of the intelligence required in driving and navigating on the road. Embedded systems are conceived to meet these standards, and as such they are a necessary step to implement advanced technologies onto a multitude of private vehicles. Ability to stay alert, aware, and to comprehensively factor in related information is required to navigate safely. Latest developments in sensors, distributed information processing, location aware information management, as well as context driven cognitive intelligence all indicate that intelligent vehicles will soon share the public roads with humans within the next twenty years. Embedded systems can carry artificial intelligence for vehicles to be "situated," that is, to specialize itself on the environment and habits of the driver and his or her family. Machine learning techniques based on statistical analyses of operating conditions can enable the device to predict changes in road conditions and consequently adapt to reduce the frequency of critical faults.

Embedded systems are also the natural host for solutions based on distributed intelligence, as opposed to centralized intelligence managed from a headquarters station.

Distributed intelligence can reach locations that centralized services may not be able to reach, for lack of infrastructures, or other structural limits. For example, in alternative to broadcasted wireless networks, smart communication devices can be embedded on vehicles and create a network that "runs on the road," exchanging messages when crossing other vehicles and continuously updating the traffic situation. The most positive scenario would see "intelligent" technology spread independently from governmental intervention through the channels of consumer market, probably in the form of embedded systems. This capillarity would foster a more distributed sharing of responsibilities and costs for introducing new advances of strong social impact. The diffusion of GPS personal navigation systems provides a good example of this scenario: GPS receivers deliver information on traffic jams and can partly redirect the circulation on less-used roads, however, when one buys a personal navigation system is rarely steered by the common good.

In launching this special issue, we aimed to attract discussion and up-to-date results on embedding intelligent systems onto vehicles, spanning applications in localization-based services, ad hoc networking and communication, smart sensors and sensor fusion, embedded vehicle controls, and embedded security. All these are bricks towards building an "autonomous" vehicle, where autonomous refers to the next generation of "automatic" devices. An autonomous vehicle is not necessarily unmanned. Instead, it should be intended as being able to react in a closed loop with the environment it operates in, adapting its behavior to provide improved services and safety to the human driver and other participants of the road environment.

In this context, we have brought together an issue filled with eight exciting articles that represent outstanding developments in this area. These were chosen from a bigger pool through the traditional peer review process. We thank both the authors and the reviewers in making this possible. The papers cover a broad spectrum of research results in: localization, information management, embedded image processing, navigation, context driven reasoning, and so forth as briefly outlined below.

In this issue

The articles presented in this issue cover a broad set of challenges being addressed by the research community in intelligent transportation systems. These have been grouped into four avenues: (1) location-based information and services: acquisition of vehicle location, management and delivery of these data to vehicles in transit; (2) radar-based service for distance and data exchange; (3) embedded image processings: methodology, omnidirectional imaging and stereo; And (4) security.

GPS/low cost IMU/on-board vehicle sensors integrated land vehicle positioning system

Robust, reliable, and swift access to current location of the vehicle is of importance to the autonomous and remote operation of vehicles. A less-obvious use of this data is to control the braking mechanism such antilock brake systems, detection, and avoidance of collision due to uncooperative vehicles sharing the road with oneself. Ability to make use of precisely measured terrain data, including such information as debries, pot-holes, and so forth will be limited to accuracy with which the vehicle can ascertain its self-position. Latest developments in sensor technology offer a variety of compact inertial measurement units (IMU) based on micro-electromechanical systems (MEMS) to acquire reliable information about the vehicles dynamics. Sensor fusion techniques offer a way of integrating such sensor data with that of global position sensors (GPS) extending the dependability of GPS in urban areas where they are known to be unreliable.

The article by J.Gao et al. presents a robust and reliable on-board vehicle sensor fusion based on low cost GPS and IMUS. They demonstrate an increased effectiveness as high as 92.6% in open-sky terrains and 65% in suburban areas based on real-time tests.

Real-time implementation of a GIS-based localization system for intelligent vehicles

In addressing the absolute localization problem by multisensor fusion, one step further is to consider the integration of the advantages brought by local maps from a geographical information system (GIS). Local maps can bear information on landmarks in the local area, for example, particularly visible traffic signs, or outstanding buildings and monuments. Extra exteroceptive sensors like video cameras or laser scanners can be used to locate these landmarks and aid GPS and IMUS with this additional information, thus increasing the precision of vehicles localization capabilities and compensating the defaults of the other sensors. Managing a GIS efficiently is a nonnegotiable prerequisite to this technology. The article by P. Bonnifait et al. describes GPS-IMU Kalman-based fusion and focuses on the problem of retrieving identification numbers (ID) of local maps from a geographical information system (GIS) at a frequency higher than the current commercial standard of 1 Hz. They use an enhanced map representation for efficient road selection and tackle cache memory management.

Broadcasted location-aware data cache for vehicular application

Ability to relate current location of the vehicle does not stop at obstacle detection and collision avoidance. Strategic and pragmatic information such as the weather and traffic conditions several miles ahead on journey, the location of a nearest restaurant, pharmacy, auto-repair shop and hospitals, for example, prove to be valuable at times. Other strategic information includes what is the traffic condition in one or more alternate paths between a nearest exit on our current journey and another several miles ahead. Contrary to the common belief, this information can be acquired—both static and dynamic—and delivered to through a set of well coordinated information services. The vehicles have an opportunity to act as the sensors and trade information, in addition to being able to buy information. Volume of data to be acquired,

integrated, and delivered on demand would require new and efficient paradigms to manage the data in a distributed fashion. The usage value, and average transit time in a locale, and the density of traffic, and so forth determine the geographic extent to which a location tagged data will be proactively cached for a potential on-demand usage at a nearby location. The article by K. Sato and Fukuda outlines a set of metrics such as "scope" and "mobility specification" and model the performance of such a system to study the latency and quality of service obtainable through currently available digital communication infrastructures such as the cell phones.

This paradigm has a significant potential to be embraced by the commercial sector.

Embedded localization and communication system designed for intelligent guided transports

Interest in intelligent vehicle technologies is not limited to the private sector. Also public transportation is looking to enhanced communication and sensors to improve the safety standards. Indeed, trains allow for a higher space capacity and load carrying, thus posing less restriction on experimenting embedded intelligent systems. The metropolitan lines, with their high traffic and many train crossings, provide a challenging scenario for smart communications that somehow replicates the conditions of a road environment but with more controllable parameters. Smart communications can be used to gauge the distance from incoming vehicles, as well as to provide a channel for exchanging data on route conditions in the lines recently visited. Y. Elhillali et al. describe a radar-based multiaccess communication system and provide experimental data coming from a prototype tested on a train.

System platforms-based SystemC TLM design of image processing chains for embedded applications

Video cameras are becoming ubiquitous. In particular, cameras based on CMOS technology have steadily improved over the past decade to offer impressive overall performance over their CCD counterparts in terms of frame-rate and embedded image processing. To be efficiently embedded on intelligent vehicles, vision-based sensing follow the practice of coordinated hardware-software codesign widely consolidated for component design in the automotive industry. For researchers and developers in computer vision, this coordinated design was a more common practice in the early days of image processing, when the massive amounts of image data prompted researchers worldwide to develop ad hoc, parallel hardware to attain quasi-real-time computation. The advent of more powerful personal computers changed the situation, bringing researchers towards more software-oriented solutions. However, embedding vision on real-working vehicle systems necessarily passes through an optimization step involving hardware design. M. D. Cheema et al. outline a methodology for hardware/software codesign of image processing systems and guide the reader step by step through a complete case study, detailing all modeling tools and options they have selected.

Lane tracking with omnidirectional cameras: algorithms and evaluation

In the range of possible camera configurations, catadioptric omnidirectional cameras make an attractive choice for intelligent vehicle applications. A convex mirror projects an all-around view onto the sensor, so that a single camera mounted just behind the windscreen provides a view of both the road environment and the occupants inside the vehicle. One of such cameras provides information simultaneously for vision applications tackling road navigation—lane keeping, obstacle detection, and so forth—and for monitoring driver's activities to deliver services and driver assistance—sleepy drive detection, and so forth. The sensor does not need to be mechanically moved so that no resource conflict arises between different algorithms, and also such a solution minimizes the space occupancy inside the vehicle. However, a catadioptric solution with a convex mirror induces a decrease of resolution, and this factor may spoil the performance of some algorithms well tested for traditional cameras.

S. Y. Cheng and M. Trivedi describe some recent results concerning lane keeping with an omnidirectional color camera. The authors have adapted their own work on lane tracking developed for a traditional pinhole camera model to the omni-directional device, and investigate how the reduced resolution impacts on the performance.

StereoBox: a robust and efficient solution for automotive short-range obstacle detection

Video imagery plays an obvious and critical role in navigating a vehicle. Ready access to mask level design customization of cameras suitable for foveated algorithms and increased availability of high-frame rate video cameras have triggered a resurge in a variety of geometrically designed vision algorithms. Although the LIDAR sensors have proven to be effective in unmanned vehicles in off-road navigating vehicles, they are not suitable for sensing in the presence of human driven vehicles. Moreover, vision is more than sensing the 3D geometry of the scene ahead. It provides rich set cues based on texture and shade information effortless perceived by humans. So we are naturally interested in robust and efficient sensing of obstacles within a short distance from the vehicle based on visual data. Also, it is desirable to deliver the result in a form so as to facilitate integration of LIDAR and other data when available. The article by A. Broggi et al. presents a comprehensive and well-tested recent result on robust and efficient short-range obstacle detection.

State of the art: embedding security in vehicles

Similar to what happened to work stations and personal computers when they gained worldwide connectivity thanks to the Internet, we should expect that future networked vehicles become a target for malicious attacks with the goal of theft or, worse, remote control. Without going that far, today cars already rely on IT security for some applications such as immobilizers or digital tachographs which can be targeted by IT attacks. IT security systems for intelligent vehicles will

take advantage of the knowledge gained in Internet security issues, but of course the scenario is completely different, and the characterization of possible attackers, their motivations, and their means requires a farseeing analysis of automotive-related problems. In the future, security issues will have to be tackled from an early stage of component design, and so the conscience of common attacks and state-of-the-art defenses becomes a significant expertise for anybody working in the sector. In their article, M. Wolf et al. provide an overview on embedding security in vehicles, with an eye to future scenarios and yet-to-come technology.

ACKNOWLEDGMENTS

The guest editors thank Zoran Salcic, Editor-in-Chief of EURASIP Journal of Embedded Systems, for the opportunity to publish this special issue dedicated to intelligent vehicles. They also thank the editorial staff for their continuous support, understanding, and patience and gratefully acknowledge the authors and reviewers for helping them bring together an excellent set of papers. The affiliation of Paolo Lombardi with the European Commission and of Guna S. Seetharaman with the US Air Force does not imply any endorsement of the contents, nor does this article represent any stated or implied policies or technology emphases within the European Commission, the US Air Force, the US Department of Defense, nor the US government.

Samir Bouaziz
Paolo Lombardi
Roger Reynaud
Gunasekaran S. Seetharaman

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 62616, 14 pages
doi:10.1155/2007/62616

Research Article

GPS/Low-Cost IMU/Onboard Vehicle Sensors Integrated Land Vehicle Positioning System

Jianchen Gao, Mark G. Petovello, and M. Elizabeth Cannon

Position, Location, and Navigation (PLAN) Group, Department of Geomatics Engineering, University of Calgary, 2500 University Drive NW, Calgary, AB, Canada T2N 1N4

Received 14 October 2006; Accepted 9 April 2007

Recommended by Gunasekaran S. Seetharaman

This paper aims to develop a GPS, low-cost IMU, and onboard vehicle sensors integrated land vehicle positioning system at low cost and with high (cm level) accuracy. Using a centralized Kalman filter, the integration strategies and algorithms are discussed. A mechanism is proposed for detecting and alleviating the violation of the lateral nonholonomic constraint on the wheel speed sensors that is widely used in previous research. With post-mission and real-time tests, the benefits gained from onboard vehicle sensors and the side slip detection and alleviation mechanism in terms of the horizontal positioning accuracy are analyzed. It is illustrated by all the tests that GPS plays a dominant role in determining the absolute positioning accuracy of the system when GPS is fully available. The integration of onboard vehicle sensors can improve the horizontal positioning accuracy during GPS outages. With respect to GPS and low-cost IMU integrated system, the percentage improvements from the wheel speed sensor are 90.4% for the open-sky test and 56.0% for suburban area real-time test. By integrating all sensors to detect and alleviate the violation of the lateral nonholonomic constraint, the percentage improvements over GPS and low-cost IMU integrated system can be enhanced to 92.6% for open-sky test and 65.1% for the real-time test in suburban area.

1. INTRODUCTION

In recent years, significantattention has been paid to intelligent vehicle systems. Among them are antilock brake systems (ABS), traction control (TC), vehicle stability control (VSC), vehicle safety control such as forward-collision avoidance, to name a few [1]. In these systems, numerous sensors are being employed, and the positioning accuracy and system redundancy have crucial impacts on their performance [2]. Due to the complementary features, GPS/INS integrated systems have been widely used for vehicular positioning and navigation [3–5]. With respect to GPS positioning, centimeter-level accuracies can be achieved by using carrier phase measurements in a double difference approach whereby the integer ambiguities are resolved correctly. However, difficulties arise during significant shading from obstacles such as buildings, overpasses, and trees. To bridge GPS gaps and reduce the INS error growth, many auxiliary sensors, such as compasses, inclinometers, tilt meters, and odometers have been used to provide further external aiding [6, 7]. Typically, the wheel speed sensors are fundamental components of an ABS which is standard equipment on nearly all vehicles [8]. Therefore the integration of the wheel speed senor with GPS/INS has been extensively studied [9–11].

Since the wheel speed sensor can be used to estimate the vehicle's velocity in the forward direction, most of the previous research related to the integration of wheel speed sensor information with GPS/INS applied two nonholonomic constraints on the lateral and vertical directions. These nonholonomic constraints are effective only when the vehicle operates on a flat road and no side slip occurs [12, 13]. The nonholonomic constraints are no longer valid when the vehicle runs off road or on an icy or bumpy road where a larger side slip angle can occur. In a land vehicle positioning system, the violation of the nonholonomic constraints is always accompanied by larger side slip angles [13]. Side slip is a very complicated phenomenon associated with road conditions and high vehicle dynamics (e.g., fast driving, sharp turning as well as high pitch and roll angular rates). It is not easily modeled and estimated. Reference [14] investigated a model-based Kalman filter with GPS velocity measurements to estimate side slip. However, its estimation accuracy relies heavily on the correctness of the model.

As a commonly used system for a land vehicle positioning system, a GPS/INS integrated system harnesses either a tactical grade or low-cost IMU. The high cost of a tactical grade IMU constitutes its main limitation to commercial deployment. The performance of a low-cost IMU degrades quickly

over a short time interval of GPS outages. A larger error drift is not well suited to such a land vehicle positioning system that has a strict requirement on positioning accuracy as the intelligent or autonomous vehicle control system. To meet the requirement of the low-cost and high positioning accuracy for the land vehicle positioning system, a GPS receiver, low-cost IMU, and onboard vehicle sensors are integrated into a land vehicle positioning system. The onboard vehicle sensors come from the vehicle stability control system, including two horizontal G sensors (accelerometers) and yaw rate sensor (i.e., a two-dimensional automotive grade inertial unit) as well as wheel speed sensors. After describing the integration strategy, a mechanism is proposed for the computation of the side slip angle as well as the detection and compensation of the violation of the lateral nonholonomic constraint. The tests for the post-mission and the real-time were conducted. The benefits after integrating the onboard vehicle sensors and using the side slip detection mechanism in terms of the horizontal positioning accuracy are analyzed.

2. COORDINATE FRAME DEFINITIONS FOR ONBOARD VEHICLE SENSORS AND THE LOW-COST IMU

The MEMS low-cost IMU consists of three triads of accelerometers as well as three triads of gyros, which, respectively, measure three-dimensional specific forces and angular rates with respect to the IMU body frame. The low-cost IMU body frame represents the orientation of the IMU axes. The origin of the body frame is at the center of the IMU. The IMU axes are assumed to be approximately coincident with the moving platform upon which the IMU sensors are mounted with the Y-axis pointing towards the front, the X-axis pointing toward the right, and the Z-axis being orthogonal to the X and Y axes to complete a right-handed frame. The low-cost IMU is mechanized in ECEF (earth-centered earth-fixed) frame (e frame) herein although any convenient frame could be used.

The onboard vehicle sensors discussed in this paper include: two rear and two front wheel speed sensors, two G sensors, and a yaw rate sensor. The wheel speed sensors (WSS) are attached to the wheels of the vehicle and provide estimates of the forward velocity in vehicle frame that are susceptible to the change in actual rolling tire radius and the tire longitudinal slip. The vehicle frame is attached to the vehicle at its center of gravity to represent the orientation of the vehicle. The X-axis points towards the right side of the vehicle. The Y-axis points towards the forward direction of the vehicle. The Z-axis is orthogonal to the X and Y axes to complete a right-handed frame. The wheel speed sensor measurements are represented in the vehicle frame.

The G sensors and yaw rate sensor that are placed on the chassis of the vehicle actually constitute a two-dimensional automotive grade inertial unit. The G sensors measure the lateral and longitudinal specific forces, and the yaw rate sensor measures angular rate with respect to the vertical direction of the G sensors/yaw rate sensor unit (GL/YRS). The G sensors and yaw rate sensor (GL/YRS) body frame follows the same definition of the low-cost IMU.

It is an ideal case that the IMU body frame coincides to the vehicle frame. However, due to installation "error" of the IMU, the bore sight of the IMU is misaligned with the vehicle frame in most cases. The tilt angles between the IMU body frame and the vehicle frame will result in some errors when the position or velocity is transformed from one frame to another without taking into account the tilt angles. As such, a calibration algorithm for estimating the tilt angles between the IMU body frame and the vehicle frame is implemented in this paper.

For simplicity, the low-cost IMU and the GL/YRS unit body frames, however, are assumed to be aligned despite tilt angles that may actually exist between these two frames. Theoretically, the tilt angles will lead to constant biases of the accelerometer measurements [15]. In the land vehicle positioning system with a relatively small attitude change rate, this assumption holds in most cases with the biases of IMU and GL/YRS measurements being estimated by the Kalman filter.

3. GPS/LOW-COST IMU/ONBOARD VEHICLE SENSOR INTEGRATION STRATEGY

Figure 1 describes the GPS, low-cost IMU, WSS and GL/YRS integration strategy, which consists of two basic modules, GPS/INS/WSS and GPS/INS/GL/YRS as well as a combined module of GPS/INS/WSS/GL/YRS. All available sensor measurements are combined using a tight coupling strategy at each epoch to obtain a globally optimal solution using one centralized Kalman filter [16]. For the equipment used, the IMU data rate is 100 Hz, and its mechanization equation output rate is set to 10 Hz. The GPS measurements used herein are double-differenced carrier phase, double-differenced Doppler, and double-differenced pseudorange at a 1 Hz rate. The onboard vehicle sensors are sampled at 100 Hz. To make a tradeoff between the system accuracy and the computational load in the real-time test, the vehicle sensor data are thinned to 1 Hz for the update of the centralized Kalman filter. The position, velocity, and attitude information of the integrated system are given by implementing the mechanization equation of the low-cost MEMS IMU in ECEF frame.

Due to the centralized processing approach, the satellite measurements are estimated by using the integrated position and velocity. The raw GPS measurements and the estimated satellite measurements are compared to derive the GPS measurement misclosures in the centralized Kalman filter. When the ambiguities need to be fixed, the float double differenced ambiguities are added to the state vector and estimated in the centralized Kalman filter. The integer ambiguities are resolved using the LAMBDA [17] method with the real-valued ambiguities and their corresponding estimated covariance matrix from the Kalman filter.

The forward-direction velocity in the vehicle frame can be determined from the WSS, while two nonholonomic constraints are applied to the vertical and lateral directions of the

vehicle frame. The wheel speed sensor provides the absolute velocity information to update the centralized Kalman filter. During GPS outages, the nonholonomic constraints, as well as the absolute velocity information, can constrain the velocity and consequently the position drift of the stand-alone INS system.

The nonholonomic constraints imply that the vehicle does not move in the vertical or the lateral directions. The nonholonomic constraints hold only when the vehicle runs on flat road with small side slip. On the bumpy road with a larger side slip, the assumption of nonholonomic constraints is no longer valid.

The wheel speed sensors used in this research are of the passive type. The number of pulses per rotation is measured with the sensor teeth going through a passive magnetic field. The wheel speed is consequently correlated with the number pulses per rotation as well as the radius of the wheel tire. In practical use, the tire size is sensitive to many factors such as a payload, the driving conditions, temperature, air pressure, and the tread wear. Additionally, the IMU body frame does not always coincide with the vehicle frame. Thus, the scale factor of the wheel speed sensor and the tilt angles between the vehicle and body frames are augmented into the state vector of the centralized Kalman filter, as described in Figure 1.

Instead of providing the absolute velocity as the WSS does, the GL/YRS unit derives the lateral and longitudinal velocity with the initial velocity being provided from the integrated system. GL/YRS unit performs a velocity update in its body frame by computing the measurement misclosures (also termed as "innovations") between the integrated velocity (being transformed from ECEF frame into the body frame) and lateral/longitudinal velocity computed from GL/YRS. Similarly to the low-cost MEMS IMU, the biases of the G sensors and yaw rate sensor are augmented into the state vector of the centralized Kalman filter.

As illustrated in Figure 1, the centralized Kalman filter is a closed loop type. It indicates that the relationship between the centralized Kalman filter and the velocity update from either wheel speed sensor or GL/YRS unit are bidirectional. In one way, the GPS update provides an external aiding to limit the INS drift error when GPS is available. During GPS outages, WSS and GL/YRS will continue to update the centralized Kalman filter and bridge GPS data gap. In another way, the estimated error states feedback to the integrated solutions as well as the low-cost IMU, WSS, and GL/YRS measurements. With the feedback information, the integrated position, velocity, and the attitude angles can be corrected by the estimated error states of position, velocity, and the misalignment angles. Also, the estimated accelerometer and gyro biases, WSS scale factor and GL/YRS biases can rectify the IMU and onboard vehicle sensor measurements, respectively.

It has been verified by [11] that the WSS with two nonholonomic constraints can significantly improve the positioning accuracy during GPS outages. The lateral nonholonomic constraint is very close to a real condition with a small side slip, and it is violated with a larger side slip. This constitutes a weak point of GPS/INS/WSS integration module.

The quality of automotive grade GL/YRS is similar to that of the low-cost IMU. Thus, its error will drift at the same rate as the low-cost IMU during GPS outages, and the improvement on the positioning accuracy gained from GL/YRS is less significant than from WSS.

Based on the above analysis, Figure 2 describes this interactive relationship between WSS and GL/YRS. The absolute velocity update from the WSS measurements limits the longitudinal velocity drift error. Consequently, the accuracy of the initial longitudinal velocity for GL/YRS is increased. On another hand, the side slip angle can be calculated from the lateral and longitudinal velocities. The side slip angle information provides a way to detect and alleviate the violation of the lateral nonholonomic constraint. When the side slip angle is smaller than a threshold, it means that the lateral constraint is most likely valid. However, when the side slip angle goes beyond a specific threshold, it indicates that the lateral nonholonomic constraint is violated. To compensate the violation of the lateral nonholonomic constraints, one possible way is to make use of the lateral velocity calculated from the GL/YRS to replace the lateral nonholonomic constraint as will be discussed later.

4. INTEGRATION ALGORITHMS

The development of the integration algorithms includes the derivations of the dynamic and measurement models used in the Kalman filter, as well as the computation of side slip angle, and the mechanism for detecting and alleviating the violation of the nonholonomic constraints used in GPS/INS/WSS/GL/YRS integration strategy.

The error states estimated by GPS/INS centralized Kalman filter include position errors, velocity errors, misalignment angles, the accelerometer and gyro biases, as well as the double-differenced ambiguities ($\Delta\nabla N$), when necessary. The dynamic model for the GPS/INS centralized Kalman filter is given by [3]:

$$\begin{aligned}\begin{bmatrix}\delta\dot{r}^e\\ \delta\dot{v}^e\\ \dot{\varepsilon}^e\\ \delta\dot{b}^b\\ \delta\dot{d}^b\\ \Delta\nabla\dot{N}\end{bmatrix} &= \begin{bmatrix}0 & I & 0 & 0 & 0 & 0\\ N^e & -2\Omega^e_{ie} & -F^e & R^e_b & 0 & 0\\ 0 & 0 & -\Omega^e_{ie} & 0 & R^e_b & 0\\ 0 & 0 & 0 & -\operatorname{diag}(\alpha_i) & 0 & 0\\ 0 & 0 & 0 & 0 & -\operatorname{diag}(\beta_i) & 0\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}\\ &\quad\cdot\begin{bmatrix}\delta r^e\\ \delta v^e\\ \varepsilon^e\\ \delta b^b\\ \delta d^b\\ \Delta\nabla N\end{bmatrix} + \begin{bmatrix}0 & 0 & 0 & 0\\ R^e_b & 0 & 0 & 0\\ 0 & R^e_b & 0 & 0\\ 0 & 0 & I & 0\\ 0 & 0 & 0 & I\\ 0 & 0 & 0 & 0\end{bmatrix}\cdot\begin{bmatrix}w_f\\ w_w\\ w_b\\ w_d\end{bmatrix}\\ &= F_{\text{GPS/INS}}\cdot\delta x + G_{\text{GPS/INS}}\cdot w,\end{aligned} \tag{1}$$

where δr^e is the position error vector, δv^e is the velocity error vector, ε^e is the misalignment angle error vector, δb^b is the vector of the accelerometer bias errors, δd^b is the vector of the gyro bias errors, all of aforementioned error states are 3×1 vectors, $\Delta\nabla N$ is the vector of double difference carrier phase ambiguities, w_f is the accelerometers noise, w_w is the

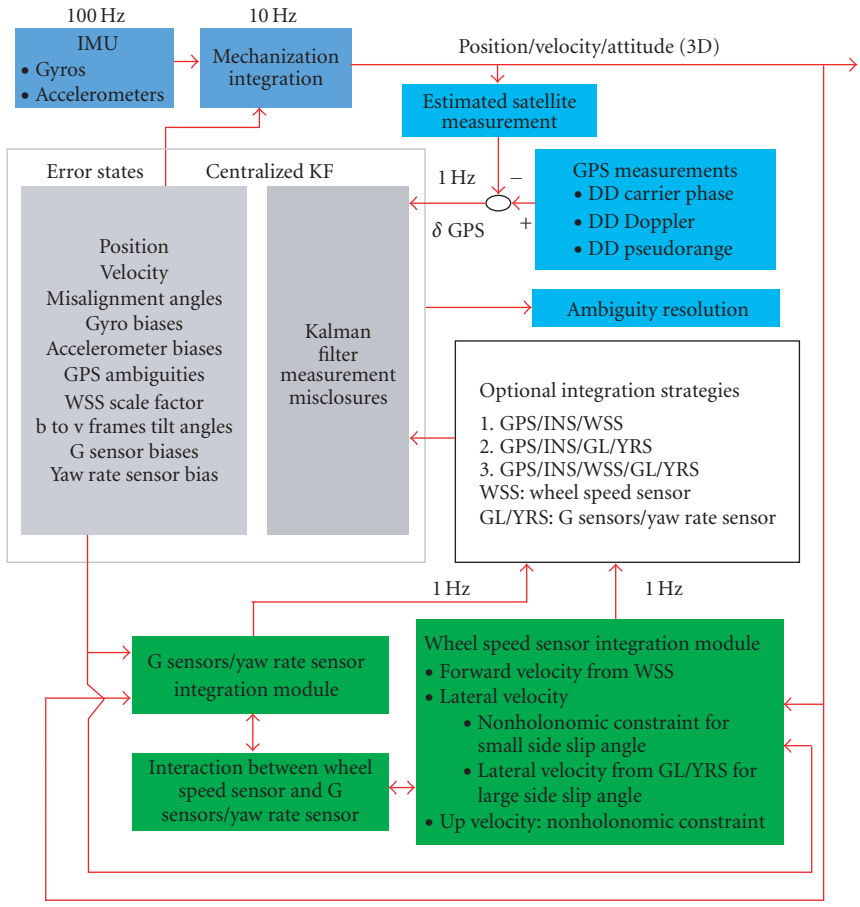


Figure 1: Schematics of GPS/low-cost IMU/WSS/GL/YRS integration strategy.

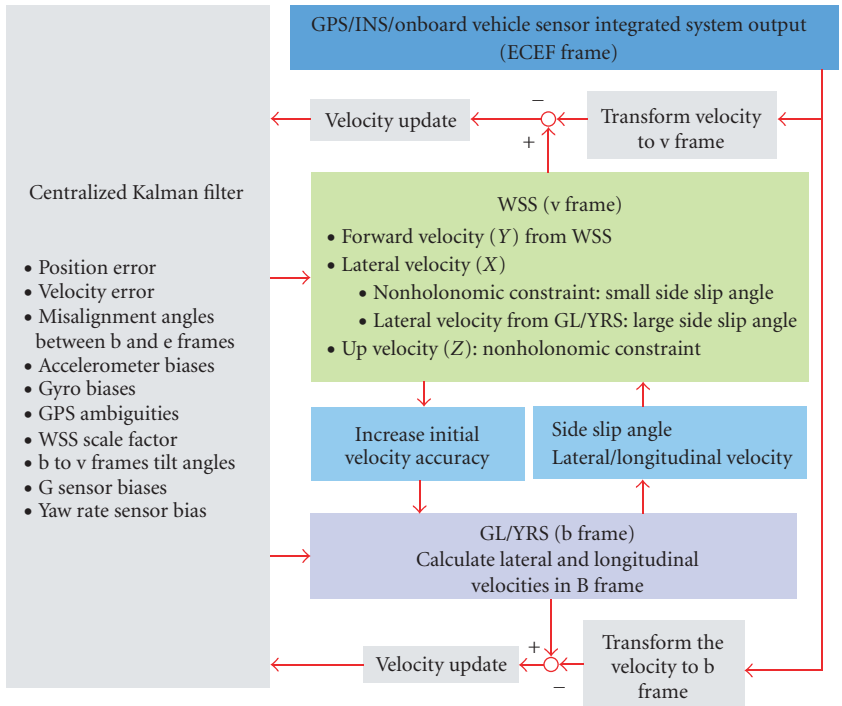


Figure 2: The relationship between WSS and GL/YRS.

gyro noise, diag(α_i) is the diagonal matrix of the time constant reciprocals for the accelerometer bias model, diag(β_i) is the diagonal matrix of the time constant reciprocals for the gyro bias models, w_b is the driving noise for the accelerometer biases, w_d is the driving noise for the gyro biases, R_b^e is the direction cosine matrix between b frame and e frame, F^e is the skew-symmetric matrix of specific force in e frame, N^e is the tensor of the gravity gradients, Ω_{ie}^e is the skew-symmetric matrix of the Earth rotation rate with respect to e frame, δx is the error states vector, and $F_{\text{GPS/INS}}$ is the dynamic matrix for GPS/INS integration strategy, $G_{\text{GPS/INS}}$ is the shaping matrix of the driving noise, and w is the noise matrix.

Equation (1) implies that the bias states for accelerometers and gyros are modeled as first-order Gauss-Markov processes, although any suitable models could be used instead. In terms of the integration strategy shown in Figure 1, the scale factor of WSS and the tilt angle between b and v frames modeled as random constants, as well as the biases of GL/YRS modeled as first-order Gauss-Markov process, are augmented into the error state vector of the centralized Kalman filter to construct the dynamic model, as shown in (2):

$$\begin{bmatrix} \delta\dot{r} \\ \delta\dot{v}^e \\ \dot{\varepsilon}^e \\ \delta\dot{b}^b \\ \delta\dot{d}^b \\ \Delta\nabla\dot{N} \\ \delta\dot{S} \\ \dot{\varepsilon}_{b-v} \\ \delta\dot{b}_{\text{GL}} \\ \delta\dot{d}_{\text{YRS}} \end{bmatrix} = \left[\begin{array}{cccccc|cccc} & & & & & & & & & \\ & & & & & & & & & \\ & & \text{GPS/INS} & & & & & O & & \\ & & & & & & & & & \\ & & & & & & & & & \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta_{\text{GL}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta_{\text{YRS}} \end{array}\right] \cdot \begin{bmatrix} \delta r^e \\ \delta v^e \\ \varepsilon^e \\ \delta b^b \\ \delta d^b \\ \Delta\nabla N \\ \delta S \\ \varepsilon_{b-v} \\ \delta b_{\text{GL}} \\ \delta d_{\text{YRS}} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ R_b^e & 0 & 0 & 0 & 0 & 0 \\ 0 & R_b^e & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} w_f \\ w_w \\ w_b \\ w_d \\ w_{\text{GL}} \\ w_{\text{YRS}} \end{bmatrix}$$
$$= F_{\text{GPS/INS/WSS/GL/YRS}} \cdot \delta x + G_{\text{GPS/INS/WSS/GL/YRS}} \cdot w, \tag{2}$$

where $F_{\text{GPS/INS/WSS/GL/YRS}}$ is the dynamic matrix for GPS/INS/WSS/GL/YRS integration strategy, $G_{\text{GPS/INS/WSS/GL/YRS}}$ is the shaping matrix, δS is the wheel speed sensor scale factor error state, and $\varepsilon_{b-v} = [\delta\alpha \ \ \delta\beta \ \ \delta\gamma]^T$ is the error vector of the tilt angles between the body frame and the vehicle frame corresponding to the X, Y, and Z axes respectively, δb_{GL} is the error vector of the G sensor biases (2×1), δd_{YRS} is the error vector of the yaw rate sensor bias (1×1). β_{GL} (2×1) and β_{YRS} (1×1) are the time constant reciprocals of the first-order Gauss-Markov process model for the GL and YRS biases, respectively, w_{GL} and w_{YRS} are the driving noises for the GL and YRS biases, respectively.

The measurement model in the Kalman filter is generally expressed by (3):

$$z = h(x) + \omega_m, \tag{3}$$

where z is raw measurement, x is estimated state, $h(x)$ is the estimated measurement, and ω_m is the measurement noise.

Most measurement models are nonlinear, and the linearization is needed for the implementation of an extended Kalman filter by (4):

$$z + \left.\frac{\partial z}{\partial x}\right|_{x=x0} \cdot \delta x = h(x_0) + \left.\frac{\partial h}{\partial x}\right|_{x=x0} \cdot \delta x + \omega_m, \tag{4}$$

where x_0 is the value of the estimated state, δx is the estimated error state, $\delta z = \partial z/\partial x|_{x=x0} \cdot \delta x$ is the perturbation of the raw measurement, and $\delta h = \partial h/\partial x|_{x=x0} \cdot \delta x$ is the perturbation of the estimated measurement.

By defining the measurement misclosure as (5),

$$W_z = z - h(x_0), \tag{5}$$

equation (4) can be rearranged by (6):

$$\begin{aligned} W_z = \delta h - \delta z + \omega_m &= \left(\left.\frac{\partial h}{\partial x}\right|_{x=x0} - \left.\frac{\partial z}{\partial x}\right|_{x=x0}\right) \cdot \delta x + \omega_m \\ &= H \cdot \delta x + \omega_m, \end{aligned} \tag{6}$$

where H is the design matrix.

Since the wheel speed is, by definition, in the vehicle frame and the velocities in the integrated system are parameterized in ECEF frame, the WSS update can be either carried out in the e frame by transforming the WSS measurement into the e frame or carried out in the v frame by transforming the integrated velocities into the v frame. In this research, the WSS update is carried out in the v frame. The velocity in the Y direction of the vehicle frame is given by averaging the two rear wheel speed sensormeasurements in (7):

$$v_{\text{WSS}} = \frac{(V_{\text{RL}} + V_{\text{RR}})}{2}, \tag{7}$$

where V_{RR} is the rear right wheel speed sensor measurement, V_{RL} is the rear left wheel speed sensor measurement, and v_{WSS} is the average of the rear wheel speed sensor measurements.

The measurement equation is expressed in (8) with two nonholonomic constraints being applied into the X and Z axes of the vehicle frame:

$$\begin{bmatrix} 0 \\ S \cdot v_{\mathrm{WSS}} \\ 0 \end{bmatrix} = R_b^v \cdot \left(R_b^e\right)^T \cdot v^e + w_m, \tag{8}$$

where v_{WSS} is the wheel speed sensor measurement given by (7), S is the wheel speed sensor scale factor, and R_b^v is the direction cosine matrix between b and v frames calculated by the following:

$$R_b^v = R_3(\gamma) \cdot R_1(\alpha) \cdot R_2(\beta), \tag{9}$$

where α, β, γ are the tilt angles between the b and v frames with respect to the X, Y, and Z axes, respectively.

The perturbation of the left-hand side of (8) is expressed by (10):

$$\delta\left(\begin{bmatrix} 0 \\ S \cdot v_{\mathrm{WSS}} \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ v_{\mathrm{WSS}} \\ 0 \end{bmatrix} \cdot \delta S = V_{\mathrm{WSS}} \cdot \delta S, \tag{10}$$

where $V_{\mathrm{WSS}} = [0 \;\; v_{\mathrm{WSS}} \;\; 0]^T$ is the measurement used for WSS update. It is a 3×1 vector.

The perturbation of right-hand side of (8) is show in (11):

$$\begin{aligned} &\delta\left(R_b^v \cdot \left(R_b^e\right)^T \cdot v^e\right) \\ &\quad = R_b^v \cdot \left(R_b^e\right)^T \cdot \delta v^e + R_b^v \cdot \left(R_b^e\right)^T \cdot V^E \cdot \varepsilon^e - V^V \cdot \varepsilon_{b-v}, \end{aligned} \tag{11}$$

where v^e is the velocity in the integrated system in e frame, and $v^v = R_b^v \cdot (R_b^e)^T \cdot v^e$ is the integrated velocity in v frame, V^E is the skew-symmetric matrix of the integrated velocity in e frame v^e, and V^V is the skew-symmetric matrix of the integrated velocity expressed in v frame v^v.

From (5) and (8), the measurement misclosure is shown in (12):

$$W_z = \begin{bmatrix} 0 \\ S \cdot v_{\mathrm{WSS}} \\ 0 \end{bmatrix} - R_b^v \cdot \left(R_b^e\right)^T \cdot v^e. \tag{12}$$

From (6), (10), and (11), the design matrix is derived from (13):

$$\begin{aligned} W_z &= R_b^v \cdot \left(R_b^e\right)^T \cdot \delta v^e + R_b^v \cdot \left(R_b^e\right)^T \cdot V^E \cdot \varepsilon^e - V^V \cdot \varepsilon_{b-v} \\ &\quad - V_{\mathrm{WSS}} \cdot \delta S + \omega_m = H_{\mathrm{WSS}} \cdot \delta x + \omega_m. \end{aligned} \tag{13}$$

Thus, the design matrix H_{WSS} is summarized by (14):

$$\begin{aligned} &H_{\mathrm{WSS}} \\ &= [O_{3\times3} \;\; Q \;\; Q' \;\; O_{3\times3} \;\; O_{3\times3} \;\; O_{3\times\mathrm{AR}} \;\; -V_{\mathrm{WSS}} \;\; -V^V \;\; O_{3\times2} \;\; O_{3\times1}], \\ &\qquad \text{where } Q = R_b^v \cdot \left(R_b^e\right)^T, \quad Q' = R_b^v \cdot \left(R_b^e\right)^T \cdot V^E. \end{aligned} \tag{14}$$

Equation (14) is a hyper matrix with each submatrix corresponding to the error states defined in (2). O is a zero matrix with the subscripted dimensions and AR is the number of float ambiguities and is equal to zero when all the ambiguities are fixed.

When using the G sensors and yaw rate sensor, the equation of motion in the body frame is shown in (15) [12, 13]. Since the nonholonomic constraint is applied in the vertical direction, the vertical velocity is only coupled with gravity,

$$\begin{aligned} \dot{V}_x^b &= (f_x - b_{\mathrm{GL1}}) - V_y \cdot (r - d_{\mathrm{YRS}}) + g_x^b, \\ \dot{V}_y^b &= (f_y - b_{\mathrm{GL2}}) + V_x \cdot (r - d_{\mathrm{YRS}}) + g_y^b, \\ \dot{V}_z^b &= g_z^b, \end{aligned} \tag{15}$$

where f_x and f_y are the specific force measurements from the G sensors, γ is the yaw rate measurement, V_x^b, V_y^b, V_z^b are the velocities in the b frame, and g_x^b, g_y^b, g_z^b are the gravity elements in the b frame, $b_{\mathrm{GL}} = [b_{\mathrm{GL1}} \;\; b_{\mathrm{GL2}} \;\; 0]^T$ and d_{YRS} are the biases of G sensors and yaw rate sensor, respectively.

The gravity vector in (15) is derived from the gravity vector in the e frame by (16):

$$g^b = \left(R_b^e\right)^T \cdot g^e, \tag{16}$$

where g^e and g^b are the gravity vector in e and b frame, respectively.

By defining

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix}, \qquad J = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{17}$$

equation (15) can be replaced by the state space vector in (18), which simplifies the mathematical analysis

$$\dot{V}^b = M \cdot (f - b_{\mathrm{GL}}) + J \cdot V^b \cdot (\gamma - d_{\mathrm{YRS}}) + g^b, \tag{18}$$

where V^b is the velocity vector in the b frame.

Using the trapezoid method [15], the velocity in the body frame can be integrated from (19) as

$$\begin{aligned} V^b &= V_0^b + \frac{1}{2}(k_1 + k_2) \cdot \Delta t, \\ k_1 &= M \cdot (f_{(0)} - b_{\mathrm{GL}(0)}) + J \cdot V_0^b (\gamma_{(0)} - d_{\mathrm{YRS}(0)}) + g_0^b, \\ k_2 &= M \cdot (f - b_{\mathrm{GL}}) + J \cdot (V_0^b + k_1 \cdot \Delta t) \cdot (\gamma - d_{\mathrm{YRS}}) + g^b, \end{aligned} \tag{19}$$

where V_0^b is the initial velocity that comes from the integrated system, $f_{(0)}$ and $\gamma_{(0)}$ are the G sensors and yaw rate sensor measurements at last epoch, $b_{\mathrm{GL}(0)}$ and $d_{\mathrm{YRS}(0)}$ are the G sensors and yaw rate sensor biases at last epoch, g_0^b is last epoch's gravity vector in the b frame, k_1 and k_2 are parameters for the trapezoid integration, Δt is the integration time interval (defined to be 1 second in this research).

To conduct the GL/YRS update in the b frame, the velocity in the integrated system is transformed from the e frame

into the b frame, and the measurement equation is expressed by (20):

$$V^b = \left(R_b^e\right)^T \cdot v^e, \tag{20}$$

where V^b is the velocity computed from (19), and v^e is the velocity of the integrated system in the e frame.

The perturbation of the gravity vector in (16) can be derived as shown in (21):

$$\delta g^b = \left(R_b^e\right)^T \cdot N^e \cdot \delta r^e + \left(R_b^e\right)^T \cdot G^e \cdot \varepsilon^e, \tag{21}$$

where N^e is the tensor of the gravity gradients, G^e is the skew-symmetric matrix of the gravity vector in the e frame.

Using (19) and (21), the perturbation of the velocity vector V^b is expressed by (22):

$$\begin{aligned}\delta V^b &= \frac{1}{2}(\delta k_1 + \delta k_2) \cdot \Delta t \\ &= \frac{\Delta t}{2}\left(R_b^e\right)^T \cdot N^e \cdot \delta r^e + \frac{\Delta t}{2}\left(R_b^e\right)^T \cdot G^e \cdot \varepsilon^e \\ &\quad + \frac{\Delta t}{2} M \cdot \delta b_{\mathrm{GL}} + \frac{\Delta t}{2} J \cdot \left(V_0^b + k_1 \cdot \Delta t\right) \cdot \delta d_{\mathrm{YRS}}.\end{aligned} \tag{22}$$

The perturbations of the right-hand side of (20) are shown in (23):

$$\delta\left(\left(R_b^e\right)^T \cdot v^e\right) = \left(R_b^e\right)^T \cdot \delta v^e + \left(R_b^e\right)^T \cdot V^E \cdot \varepsilon^e, \tag{23}$$

where V^E is the skew-symmetric matrix of the integrated velocity in the e frame.

Similarly to WSS update, the measurement misclosure can be derived from (5) and (20) as shown in (24):

$$W_z = V^b - \left(R_b^e\right)^T \cdot v^e. \tag{24}$$

Based on (22) and (23), the design matrix related to the GL/YRS velocity update is consequently derived in (25) in terms of (6):

$$\begin{aligned}W_z &= \delta\left(\left(R_b^e\right)^T \cdot v^e\right) - \delta V^b + w_m \\ &= -\frac{\Delta t}{2} \cdot \left(R_b^e\right)^T \cdot N^e \cdot \delta r^e + \left(R_b^e\right)^T \cdot \delta v^e \\ &\quad + \left[\left(R_b^e\right)^T \cdot V^E - \frac{\Delta t}{2} \cdot \left(R_b^e\right)^T \cdot G^e \cdot \Delta t \cdot\right]\varepsilon^e \\ &\quad - \frac{\Delta t}{2} \cdot M \cdot \delta b_{\mathrm{GL}} - \frac{\Delta t}{2} \cdot J \cdot \left(V_0^b + k_1 \cdot \Delta t\right) \\ &\quad \cdot \delta d_{\mathrm{Yaw}} + w_m = H_{\mathrm{GL/YRS}} \cdot \delta x + w,\end{aligned} \tag{25}$$

where $H_{\mathrm{GL/YRS}}$ is the design matrix for the GL/YRS update, which is coupled with the error states of position, velocity, b-to-e frame misalignment angles, and GL/YRS biases. More specifically, the design matrix is

$$\begin{aligned}&H_{\mathrm{GL/YRS}} \\ &= \left[U \quad \left(R_b^e\right)^T \quad U' \quad O_{3\times3} \quad O_{3\times3} \quad O_{3\times \mathrm{AR}} \quad O_{3\times1} \quad O_{3\times3} \quad -\frac{\Delta t}{2} M \quad U''\right], \\ &\text{where } U = -\frac{\Delta t}{2}\left(R_b^e\right)^T N^e, \quad U' = \left(R_b^e\right)^T V^E - \frac{\Delta t}{2}\left(R_b^e\right)^T \cdot G^e \Delta t, \\ &\qquad U'' = -\frac{\Delta t}{2} J\left(V_0^b + k_1 \cdot \Delta t\right).\end{aligned} \tag{26}$$

In this research, the noise power of GL/YRS was determined from a static test by calculating the average standard deviation across 40 evenly spaced 1-second intervals of the static data. When performing integration with the GL/YRS measurements to derive the velocity, the noise in GL/YRS behaves as random walk error because of the integration. The propagation of G sensor noise on the velocity from the trapezoid integration is correlated with the integration time length, that is,

$$\sigma^2_{V^b_{\mathrm{Noise}}} = \sigma_f^2 \cdot \Delta t, \tag{27}$$

where σ_f^2 is the noise power of G sensors, Δt is the time interval for the integration, and $\sigma^2_{V^b_{\mathrm{Noise}}}$ is the variance propagated by the measurement noise.

Considering the integration is performed every 1 second and the initial value comes from the integrated system every 1 second, $\sigma^2_{V^b_{\mathrm{Noise}}} = \sigma_f^2$ herein. Therefore, the velocity variance for the GL/YRS velocity update can be tuned adaptively in terms of variance propagation theory from (19), which is shown in (28),

$$\begin{aligned}\sigma^2_{V^b} &= \sigma^2_{V_0^b} + \frac{1}{4} \cdot \left(\sigma^2_{k1} + \sigma^2_{k2}\right) \cdot \Delta t^2, \\ \sigma^2_{g^b} &= \left(R_b^e\right)^T \cdot N^e \cdot \sigma^2_{r^e} \cdot \left(N^e\right)^T \cdot R_b^e, \\ \sigma^2_{k1} &= M\left(\sigma_f^2 + \sigma^2_{b_{\mathrm{GL}(0)}}\right) \cdot M^T + J\sigma^2_{V_0^b} \cdot \left(\gamma_{(0)} - d_{\mathrm{YRS}(0)}\right)^2 \cdot J^T, \\ &\quad + J \cdot V_0^b \cdot \left(\sigma_\gamma^2 + \sigma^2_{d_{\mathrm{YRS}(0)}}\right) \cdot \left(J \cdot V_0^b\right)^T + \sigma^2_{g_0^b}, \\ \sigma^2_{k2} &= M\left(\sigma_f^2 + \sigma^2_{b_{\mathrm{GL}}}\right) \cdot M^T + J \cdot \left(\sigma^2_{V_0^b} + \sigma^2_{k1} \cdot \Delta t^2\right) \\ &\quad \cdot \left(\gamma - d_{\mathrm{YRS}}\right)^2 \cdot J^T + J \cdot \left(V_0^b + k_1 \cdot \Delta t\right) \cdot \left(\sigma_\gamma^2 + \sigma^2_{d_{\mathrm{YRS}}}\right) \\ &\quad \cdot \left(V_0^b + k_1 \cdot \Delta t\right)^T \cdot J^T + \sigma^2_{g^b},\end{aligned} \tag{28}$$

where $\sigma^2_{V^b}$ is the velocity variance of the GL/YRS, $\sigma^2_{b_{\mathrm{GL}}}$ and $\sigma^2_{d_{\mathrm{YRS}}}$ are the estimated variances of the GL/YRS biases provided by the Kalman filter, $\sigma^2_{b_{\mathrm{GL}(0)}}$ and $\sigma^2_{d_{\mathrm{YRS}(0)}}$ are the variances of GL/YRS biases at the previous step, $\sigma^2_{V_0^b}$ is the initial velocity variance from the integrated system, $\sigma^2_{g^b}$ is the variance of the gravity vector in the body frame, $\sigma^2_{r^e}$ is the position variance in e frame.

Figure 3 describes the geometric relationship between the WSS and GL/YRS, as well a simplified vehicle's bicycle model that contains the rear wheel side slip angle. The rear wheel side slip angle can be calculated in (29) from the transformed velocity in the lateral and longitudinal directions using [18]:

$$\beta_r = \tan^{-1}\left[\frac{V_x^b - L_r \cdot \gamma}{V_y^b}\right], \tag{29}$$

where β_r is the rear wheel side slip angle, L_r is the distance between the GL/YRS and WSS, and V_x^b and V_y^b are the lateral and longitudinal velocity derived from the GL/YRS.

The lateral nonholonomic constraint is most frequently violated when the side slip angle is large. Therefore, the side

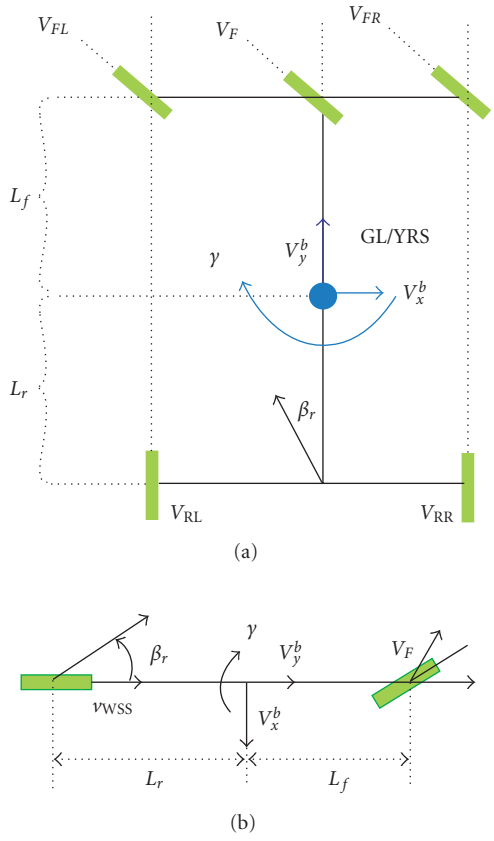


Figure 3: The geometric relationship between WSS, GL/YRS, and the side slip angle.

slip angle provides a way to detect and alleviate the violation of the lateral nonholonomic constraint. This mechanism is designed as described below.

When the side slip angle is smaller than a threshold (5 degrees in this research), it means that the nonholonomic constraint is valid, and the nonholonomic constraints are applied in both the lateral and vertical directions. In this case, (30) is used as the measurement for WSS update:

$$V_{\text{WSS}} = \begin{bmatrix} 0 \\ v_{\text{WSS}} \\ 0 \end{bmatrix}. \tag{30}$$

By contrast, if the side slip angle exceeds a threshold, the violation of the lateral nonholonomic constraint can be replaced by the GL/YRS derived lateral velocity, that is,

$$V_{\text{WSS}} = \begin{bmatrix} V_x^b \\ v_{\text{WSS}} \\ 0 \end{bmatrix}. \tag{31}$$

The longitudinal element is a real value that comes from WSS. The lateral and vertical elements are virtual values that can be either nonholonomic constraints or other values from external measurements. Despite the fact that a lateral velocity can also be given from the INS mechanization output, it cannot be employed as an external or independent measurement to remove the lateral constraint if violated. Otherwise, a correlation or dependence will be introduced when performing external update to the centralized Kalman filter. GL/YRS unit, however, provides redundant and independent measurement for detecting and alleviating the violation of the lateral constraint.

To achieve a high positioning accuracy, it is necessary to switch (30) and (31) in terms of a side slip angle threshold. With a small side slip angle, the lateral constraint is more close to the real situation where the lateral velocity is very small. If (31) is still being used when the lateral constraint is not violated; the error and noise from GL/YRS unit will degrade the positioning accuracy.

During GPS outages, the absolute velocity update from WSS limits the longitudinal velocity drift error and increases the accuracy of initial longitudinal velocity for GL/YRS. Alternatively, GL/YRS unit provides a way to detect and alleviate the violation of the lateral nonholonomic constraint on WSS. This kind of effective cooperation between WSS and GL/YRS can adapt to a variety of driving cases with a high positioning accuracy during GPS outages.

5. TESTS, RESULTS, AND ANALYSIS

To investigate the benefits gained from onboard vehicle sensors, two tests were conducted in an open sky and processed in post-mission and in a suburban area in real time. In this section, the data processing and analysis method is illustrated first. The tests are then described and the results are analyzed, respectively, for each test.

5.1. Data processing and analysis method

The data collected (see Sections 5.2 and 5.3) were processed and the results are analyzed in the following way. First, the satellite DOP (horizontal and vertical dilution of precision), the GPS availability as well as the number of resolved ambiguities are given. In the GPS/low-cost IMU/onboard vehicle sensor integrated system, GPS is the driving factor in terms of system accuracy. When GPS is fully available, GPS plays a dominant role in the integrated system and determines the absolute accuracy of the integrated system. To this end, the GPS availability, namely, the satellite availability in both the base and remote stations is analyzed in all the tests. Also, the satellite DOP which is a measure of the satellite geometry is also shown in each test. Lower DOP values give better position accuracy. The correct and fast ambiguity resolution has crucial effects on the positioning accuracy when the carrier phase measurement is used. In general, correct ambiguity resolution can result in the centimetre-level accuracy. Associated with the numbers of satellites tracked, the number of double difference ambiguities that have been fixed is shown. Second, a reference solution is generated from another independent system such as the GPS/HG1700 IMU (tactical grade) integrated system or GPS/CIMU (navigation grade) system. It is important to know the accuracy of the reference solution. Furthermore, the reference solution

can be generated by either the optimal backward smoothing technique or by a forward Kalman filtering technique. Both the GPS/HG1700 IMU and GPS/CIMU integrated solutions are accurate to the centimetre level, and there is no significant difference in the optimal smoothing and the forward Kalman filtering solutions when GPS availability is good. Therefore, the reference solution in the post-mission open-sky kinematic test was generated by the GPS/HG1700 IMU integrated solution without backward smoothing. However, in the suburban real-time test, both tactical and navigation grade IMUs are susceptible to position and velocity drift due to the frequent masking of satellite signals by trees, buildings, and underpasses. As the CIMU is more accurate than the HG1700 IMU, the reference trajectory was generated by the GPS/CIMU with an optimal backward smoothing technique because it will be more reliable than that generated by the GPS/HG1700 IMU. For the reference generated by GPS/HG1700 solution, its accuracy is shown by the estimated standard deviations of the position. The GPS/CIMU reference solution processed by the Applanix POS Pac software gives the estimated RMS error of the estimated position. The estimated RMS errors are equivalent to the estimated standard deviation assuming the estimated error has zero mean. Third, the performances of four integration strategies are assessed with respect to the reference solution. The integration strategies include GPS/INS without the aiding from the onboard vehicle sensors, GPS/INS/WSS with the nonholonomic constraints applied in the lateral and the vertical directions, and GPS/INS/GL/YRS as well as a combined integration strategy using GPS/INS/WSS/GL/YRS with a detection and alleviation mechanism for the lateral nonholonomic constraint violation.

As the side slip angle is a key parameter for the side slip detection, the side slip angles will be given in the following analysis.

5.2. Post-mission test in open-sky area

The onboard vehicle sensor and the low-cost IMU data were collected and logged onto a desktop PC through a serial port at 100 Hz. The GPS base station was set up on a pillar with a surveyed coordinate. For the open-sky kinematic test in post-mission, the GPS base station data was saved onto a flash card. The GPS/HG1700 IMU (tactical grade IMU) integrated solution generated the reference solution. The HG1700 IMU data were time tagged and logged by a NovAtel SPAN system at 100 Hz.

The purpose of the post-mission open-sky kinematic test was to tune the Kalman filter, assess the modeling of sensors and the validity of the integration algorithm, and to assess the performance and positioning accuracy for various integration strategies by simulating GPS outages. Figure 4 gives an overview of the open-sky test that was conducted on March 21, 2006 in Springbank near Calgary, which is an open-sky area with good GPS satellite visibility. The system was run for 10 minutes in static mode for initialization, and approximately for 30 minutes in kinematic mode for positioning and navigation testing with a maximum baseline length of 4 km. Due to a benign environment for the ambiguity resolution, the GPS measurements used in this test include L1 carrier phase, Doppler and the C/A code.

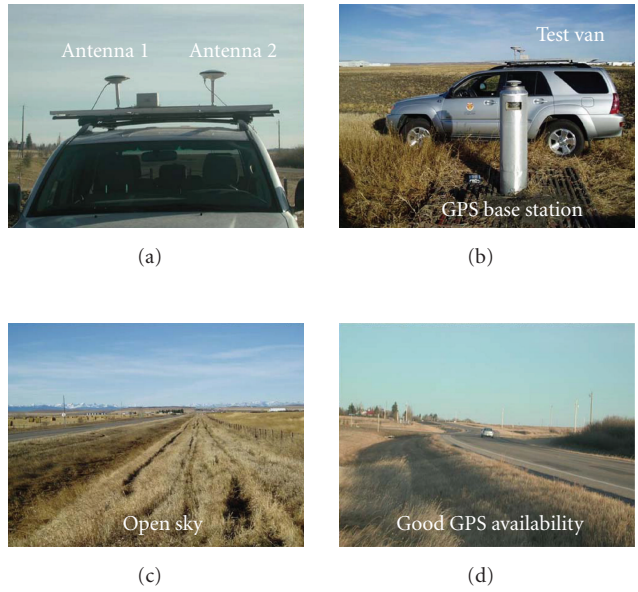


Figure 4: Open-sky test that processed in post-mission.

In the open-sky test, 12 GPS outages were simulated and the horizontal position RMS drift error with respect to the reference solution was computed. Also, the actual position difference and the estimated position standard deviations in the Kalman filter were compared. The estimated standard deviation should have good agreement with the statistics of the actual position difference in an ideal case. In practice, this indicates that the model and parameters in the Kalman filter are well tuned if the estimated standard deviation does not deviate too much from the statistics of the actual position difference.

Figure 5 shows the satellite DOP values, the number of tracked satellites at the base and remote stations (and their difference), and the number of fixed ambiguities. It can be seen that the open-sky test had good GPS availability and satellite geometry, as well as sufficient double difference (DD) ambiguities resolved.

In the open-sky area, the accuracy of the reference solution generated by GPS/HG1700 IMU integrated system is dominated by GPS. As shown in Figure 6, the carrier phase residual is around 2-3 centimetres and the C/A code residual is unbiased. It is reasonable to conclude that GPS ambiguity is correctly resolved. Hence the reference solution is accurate to be at centimetre level.

To investigate the benefits gained from the integration of the onboard vehicle sensors, 12 GPS outages of 40-second duration were simulated. The simulated GPS outages cover a wide range of vehicle dynamics. The side slip angles (by labeling the simulated GPS outage number and by zooming the side slip angles of five simulated GPS outages) are shown in Figure 7. As the open-sky test was conducted in

Table 1: Horizontal position RMS error and the average estimated standard deviation at the end of 40-second GPS outages for the open-sky test that is processed in post-mission.

Strategies	Horizontal position RMS error and average estimated standard deviation at the end of 40-second GPS outages	
	Horizontal position RMS error [m]	Average estimated standard deviation [m]
GPS/INS	30.48	31.98
GPS/INS/GL/YRS	25.00	24.90
GPS/INS/WSS	2.92	3.59
GPS/INS/WSS/GL/YRS	2.26	2.02

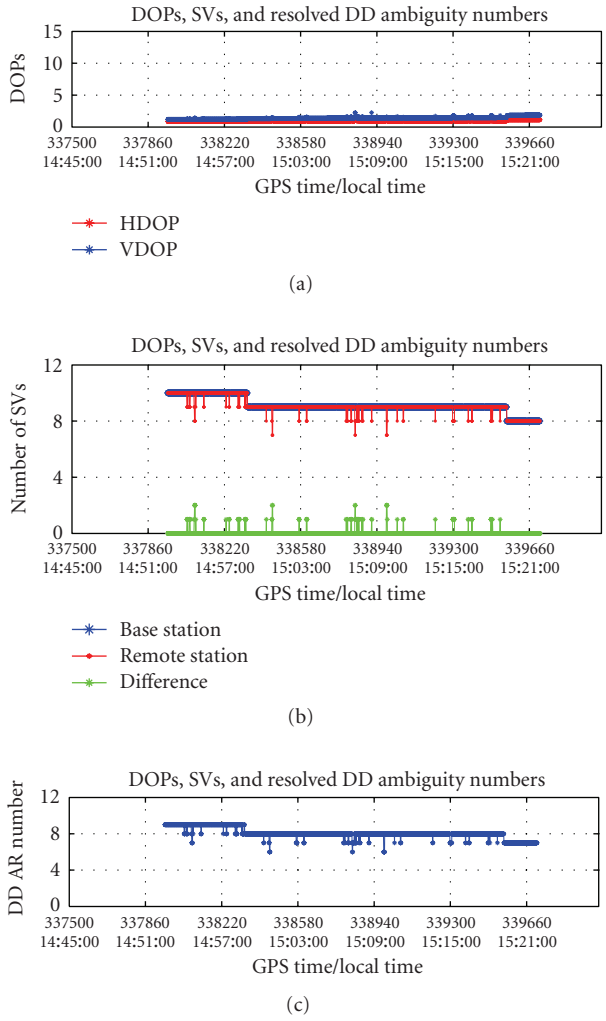


Figure 5: Satellite DOPs, satellite numbers, and the resolved ambiguities in the open-sky test that processed in post-mission.

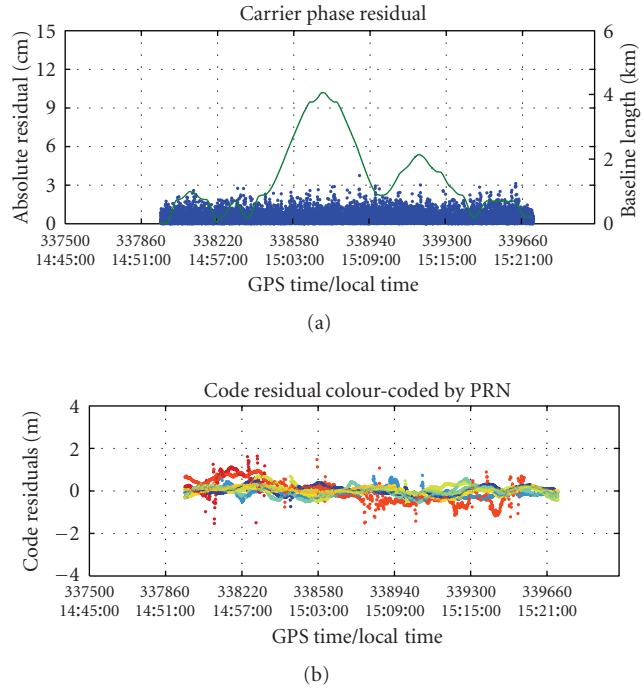
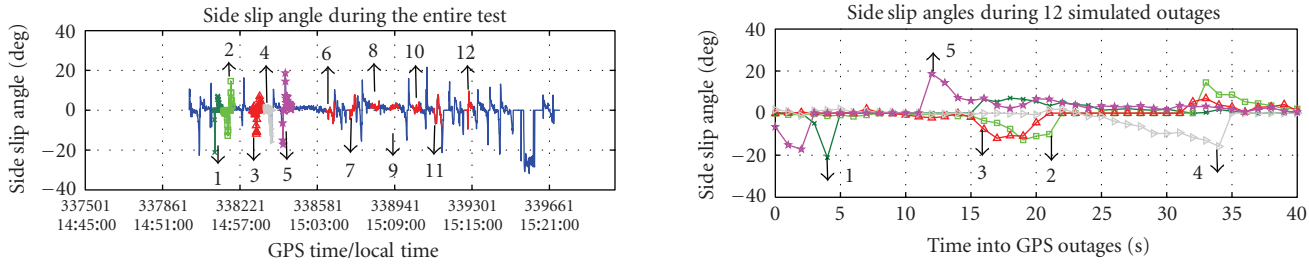


Figure 6: Residuals of carrier phase and C/A code and the baseline length for open-sky test that processed in post-mission.

March at Springbank near Calgary, the icy and bumpy road contributed to the maximum side slip angle at 20 degrees. Figure 8 compares the RMS horizontal position error and the average estimated standard deviation during the 40-second GPS outages with respect to the four proposed integration strategies. The RMS horizontal position error and the average estimated standard deviation are summarized in Table 1.

During GPS outages and without any external aiding from the onboard vehicle sensors, the low-cost IMU drifts very rapidly. However, significant benefits can be gained from the integration of the wheel speed sensor with improvements in the horizontal positioning accuracy of 90.4%. The improvement gained from the integration GL/YRS is less significant than WSS due to the low quality of GL/YRS unit. However, when the WSS and GL/YRS are incorporated (GPS/INS/WSS/GL/YRS strategy) to detect and alleviate the violation of the lateral nonholonomic constraint, the horizontal positioning accuracy can be improved by 92.6%, which is correlated with the degree of the side slip. Due to the well-tuned Kalman filter, the actual horizontal position errors and the average estimated standard deviations in the Kalman filter have good agreement for all integration strategies.



(a) (b)

Figure 7: Side slip angles in the open-sky test that processed in post-mission.

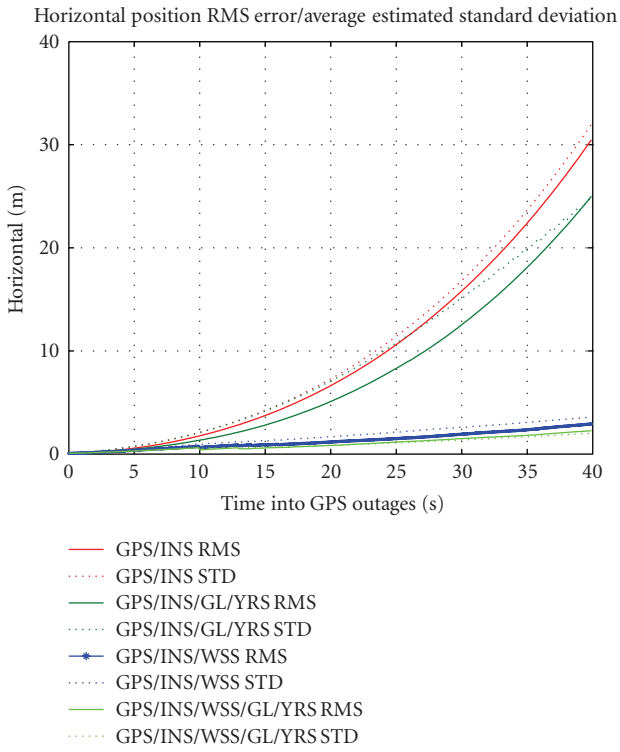


Figure 8: Horizontal position RMS error and average estimated standard deviation during 12 simulated GPS outages for the open-sky test that processed in post-mission.

5.3. Real-time test in suburban area

For real-time test in suburban area, the GPS base station data was broadcast to the remote station via a pair of FreeWave radio link antennas and transceivers. The reference solution was generated by a GPS/CIMU (navigational grade IMU) integrated solution. The CIMU data were collected at 200 Hz by an Applanix POS LS system.

The real-time tests gave an evaluation of the validity of the design of the Kalman filter as well as the impact of various sensor combinations when the satellite signals were masked. The real-time test in suburban area test started and ended in front of the Calgary Centre for Innovative Technology (CCIT) building at the University of Calgary on June 28, 2006. The test was conducted around the campus with a maximum baseline of 2.5 km, and 8 minutes of static mode for the initialization, as well as approximately 20 minutes for the kinematic part. As shown in Figure 9, the GPS base station and radio link antennas were set up on the roof of the CCIT building. Also, partial and complete GPS outages were mainly introduced by the dense foliage, small buildings near the street as well as bridges. Unlike the open-sky area, the multipath error significantly increases in suburban area. To guarantee reliable ambiguity resolution, the widelane carrier phase (rather than L1 in the open-sky test), Doppler and the C/A code measurements were used. The use of widelane measurements is at the cost of amplifying the noise by the linear combination of the L1 and L2 carrier phases. However, it is a tradeoff between fast and reliable ambiguity resolution and an increase in the noise.

Table 2: Horizontal position difference RMS for the real-time suburban area test.

Strategies	Horizontal position difference RMS [m]
GPS/INS	1.09
GPS/INS/GL/YRS	1.05
GPS/INS/WSS	0.48
GPS/INS/WSS/GL/YRS	0.38

For the real-time suburban area test, DOP values, the number of tracked satellites at the base and remote stations (and their difference), and the number of fixed ambiguities are shown in Figure 10. Most of the horizontal DOP are less than two with several cases exceeding five. However, the GPS availability is far from ideal since dense foliage, underpasses, and the buildings near the road introduced partial and complete satellite masking. The difference in the number of satellites tracked between the GPS base and the remote stations indicate the level of signal masking.

Figure 11 illustrates the estimated position accuracies for the reference solution used for the real-time test. It indicates that the estimated accuracy of the GPS/CIMU with optimal backward smoothing is closely related to the GPS availability. When GPS is fully available, the estimated accuracy is comparable to that in the open sky. However, the estimated

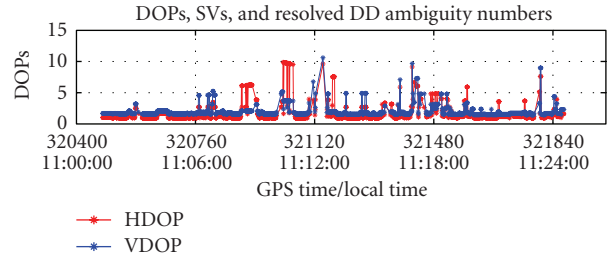
(a)

(b)

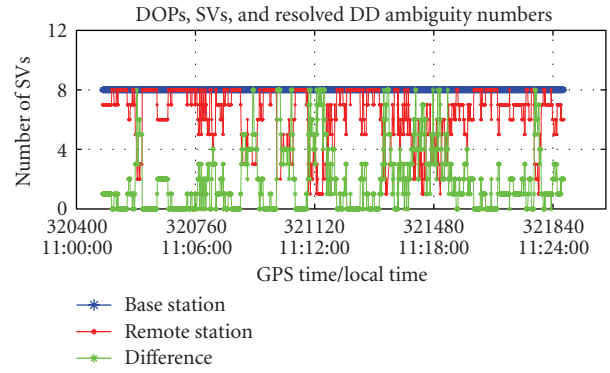
(c)

(d)

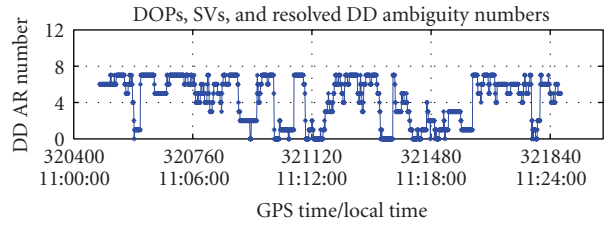
Figure 9: Real-time test in suburban area.



(a)



(b)



(c)

Figure 10: The satellite DOPs, satellite numbers, and the resolved ambiguities in the real-time suburban area test.

accuracy is susceptible to the masking of the satellites as well as the durations of the masking. The longer the duration of GPS blockage, the lower the estimated accuracy. Nevertheless, due to the superior quality of the navigational grade CIMU, the worst case for the estimated accuracy, which is relevant to the masking of GPS signal, is at the decimetre level (10–15 cm) for this test. Its accuracy is much higher than for the low-cost IMU, and thus serves as a good reference solution.

Figure 12 shows the side slip angle during the entire real-time test. As the test was conducted in the summer time on a relatively flat road, the maximum side slip angle was less than 10 degrees, with maximum values being sparsely distributed around the specific epochs at 321100 seconds, 321400 seconds, and 321840 seconds, respectively.

The horizontal position computed from the four integration strategies is compared with the reference solution, as shown in Figure 13. When GPS is fully available, GPS determines the absolute accuracy of the integrated system, and the horizontal position difference for each integration strategy is very small. During GPS outages, the horizontal position difference increases significantly depending on the duration of the outages. By comparing the four integration strategies, the aiding from the WSS can be seen to significantly reduce the position drift as compared to the stand-alone low-cost IMU. The benefits gained form GPS/INS/GL/YRS integration strategy is somewhat limited. However, the horizontal positioning accuracy can be further improved by the GPS/INS/WSS/GL/YRS integration strategy with respect to GPS/INS/WSS integration strategy if a large side slip occurs. This fact can be verified in Figure 13 around the specific epochs at 321100 seconds, 321400 seconds, and 321840 seconds, which are relevant to the large side slip angles. For easy comparisons, Table 2 statistically summarizes the horizontal

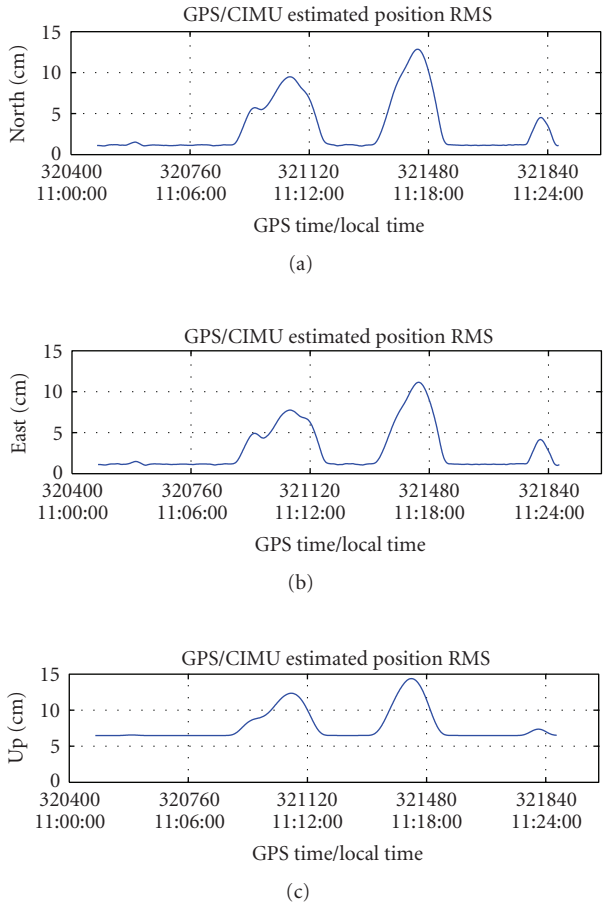


Figure 11: The position accuracy for the reference solution generated by GPS/CIMU integrated system for the real-time suburban area test.

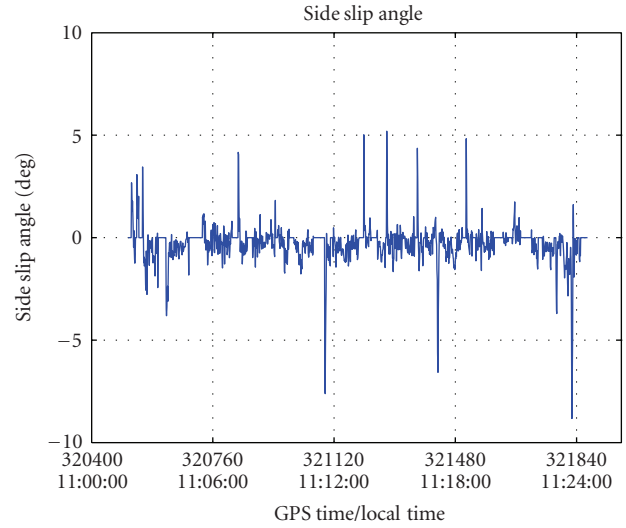


Figure 12: Side slip angle for the real-time suburban area test.

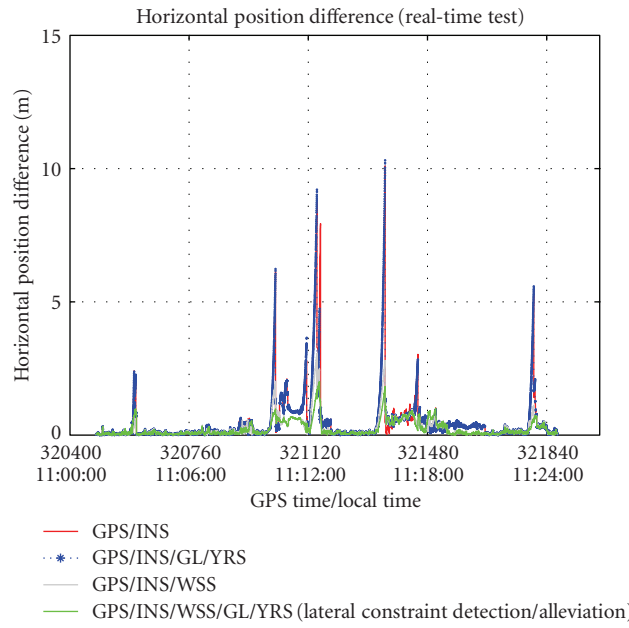


Figure 13: Horizontal position difference between GPS/INS/onboard vehicle sensor integrated output and the reference solution for the real-time suburban area test.

RMS position difference for the four integration strategies. The benefits gained from the integration of the onboard vehicle sensors on the positioning accuracy can be seen clearly.

6. CONCLUSIONS

In this paper, GPS, a low-cost IMU and several onboard vehicle sensors (four wheel speed sensors, two G sensors, and a yaw rate sensor) are integrated using a closed loop centralized Kalman filter. The integration strategies and the integration algorithms were developed. A mechanism was proposed for detecting and alleviating the violation of the lateral nonholonomic constraint on the wheel speed sensor.

It is consistently illustrated by all the tests that GPS plays a dominant role in determining the absolute positioning accuracy of the system when GPS is fully available. The integration of onboard vehicle sensors can enhance the horizontal positioning accuracy during GPS outages.

The improvements from the wheel speed sensor over GPS and low-cost IMU integrated system are 90.4% for the open-sky test (post-mission processing with 12 simulated GPS outages) and 56.0% for suburban area real-time test, respectively.

The improvement from automotive grade GL/YRS unit is less significant than the wheel speed sensor. It is only 18.0% for the open-sky test and 3.7% for suburban area real-time test, respectively. However, the strategy that integrates all sensors to detect and alleviate the violation of the lateral nonholonomic constraints performs best. Percentage improvements on horizontal positioning accuracy reached 92.6% for open-sky test and 65.1% for the suburban area real-time test.

REFERENCES

[1] H. E. Tseng, B. Ashrafi, D. Madau, T. A. Brown, and D. Recker, "The development of vehicle stability control at Ford," *IEEE/ASME Transactions on Mechatronics*, vol. 4, no. 3, pp. 223–234, 1999.

[2] D. M. Bevly, A. Rekow, and B. Parkinson, "Evaluation of a blended dead reckoning and carrier phase differential GPS system for control of an off-road vehicle," in *Proceedings of the 12th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS '99)*, pp. 2061–2069, Sashville, Tenn, USA, September 1999.

[3] M. G. Petovello, "Real-time integration of tactical grade IMU and GPS for high-accuracy positioning and navigation," Ph.D. thesis, UCGE Report 20173, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, 2003.

[4] S. Sukkarieh, *Low cost, high integrity, aided inertial navigation systems for autonomous land vehicles*, Ph.D. thesis, Australian Center for Field Robotics, University of Sydney, Sydney, Australia, 2000.

[5] S. Godha, "Performance evaluation of low cost MEMS-based IMU integrated with GPS for land vehicle navigation application," M.Sc. thesis, UCGE Report #20239, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, 2006.

[6] R. S. Harvey, "Development of a precision pointing system using an integrated multi-sensor approach," M.Sc. thesis, UCGE Report 20117, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, 1998.

[7] J. Stephen, "Development of a multi-sensor GNSS based vehicle navigation system," M.Sc. thesis, UCGE Report 20140, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, 2000.

[8] C. Hay, "Turn, turn, turn wheel-speed dead reckoning for vehicle navigation," *GPS World*, vol. 16, no. 10, pp. 37–42, 2005.

[9] Ph. Bonnifait, P. Bouron, D. Meizel, and P. Crubillé, "Dynamic localization of car-like vehicles using data fusion of redundant ABS sensors," *Journal of Navigation*, vol. 56, no. 3, pp. 429–441, 2003.

[10] J. Kubo, T. Kindo, A. Ito, and S. Sugimoto, "DGPS/INS/wheel sensor integration for high accuracy land-vehicle positioning," in *Proceedings of the 12th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS '99)*, pp. 555–564, Nashville, Tenn, USA, September 1999.

[11] J. Gao, M. G. Petovello, and M. E. Cannon, "Development of precise GPS/INS/wheel speed sensor/yaw rate sensor integrtaed system," in *Proceedings of the National Technical Meeting of the Institute of Navigation (ION NTM '06)*, pp. 780–792, Monterey, Calif, USA, January 2006.

[12] A. Brandit and J. F. Gardner, "Constrained navigation algorithms for strapdown inertial navigation systems with reduced set of sensors," in *Proceedings of the American Control Conference*, vol. 3, pp. 1848–1852, Philadelphia, Pa, USA, June 1998.

[13] G. Dissanayake, S. Sukkarieh, E. Nebot, and H. DurrantWhyte, "The aiding of a low cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 5, pp. 731–747, 2001.

[14] R. Anderson and D. M. Bevly, "Estimation of slip angles using a model based estimator and GPS," in *Proceedings of the American Control Conference*, vol. 3, pp. 2122–2127, Boston, Mass, USA, June-July 2004.

[15] C. Jekli, *Inertial Navigation Systems with Geodetic Applications*, Walter de Gruyter, New York, NY, USA, 2000.

[16] R. G. Brown and P. Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1992.

[17] P. J. G. Teunissen and A. Kleusberg, *GPS for Geodesy*, Springer, New York, NY, USA, 1996.

[18] L. R. Ray, "Nonlinear state and tire force estimation for advanced vehicle control," *IEEE Transactions on Control Systems Technology*, vol. 3, no. 1, pp. 117–124, 1995.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 39350, 12 pages
doi:10.1155/2007/39350

Research Article

Real-Time Implementation of a GIS-Based Localization System for Intelligent Vehicles

Philippe Bonnifait, Maged Jabbour, and Gérald Dherbomez

Heudiasyc UMR CNRS 6599, Université de Technologie de Compiègne, BP 20529, 60205 Compiègne Cedex, France

Received 31 October 2006; Revised 17 March 2007; Accepted 2 May 2007

Recommended by Roger Reynaud

This paper presents a loosely coupled fusion approach that merges GPS data, dead-reckoned sensors, and GIS (geographical information system) data. The GPS latency is compensated for by the DR sensors and the use of an xPPS signal. The fusion of the estimate with the map data is spatial-triggered, while the fusion with the GPS is time-triggered. We present a strategy that relies on pose tracking which is reinitialized when GPS data become incoherent. Particular attention is given to the representation of the road map and to the management of a cache memory for efficiency purposes. We report experiments carried out with our equipped car and a GIS whose characteristics are well adapted to embedded constraints.

1. INTRODUCTION

Dynamic localization is a key issue for intelligent vehicles performing driving assistance tasks or autonomous navigation in the presence of uncertainty and variability in their external environment.

Global navigation satellite-based systems (GNSS) such as GPS, Glonass and, in the near future, Galileo are interesting key components in localization. Unfortunately, when high integrity and high availability are required, a GNSS receiver alone is not sufficient for intelligent vehicles, since satellites can be masked, and since the propagation of the signal can suffer from multitracks. One way of overcoming this problem is to use dead-reckoned (DR) sensors or inertial units. These can be sufficient when outages are brief, but they cannot cope with certain situations, such as in urban areas when navigating in an urban canyon. In such cases, the drift of DR localization can be too large for the needs of the task. This problem can be addressed through the use of extra exteroceptive sensors like video cameras or laser scanners. A number of studies have demonstrated the feasibility of this approach. For instance, Royer et al. [1] have developed a navigation system for a Cycab that can control the trajectory of the vehicle with respect to a learned trajectory, using only a monocular vision system. The principle is as follows: in the learning stage, the vehicle extracts and localizes characteristic features of the environment; while navigating, it matches the features it is detecting in the current view with those that have been learned. A precise localization is subsequently computed using a Ransac method.

An ideal localization system is an embedded system that is able to deal with all of the following technologies: GNSS, DR sensors, and exteroceptive sensors detecting natural landmarks.

When implementing such a system, a key issue is the management of the landmarks that have been characterized and localized previously. The question is now how to organize the landmark information for an intelligent vehicle moving in a large area containing many roads. The usual answer to this question is to group landmarks within local maps, a local map being a set of landmarks considered as a monolithic entity because of (i) memory constraints arising from the use of embedded systems, (ii) the need to download or update a limited amount of data from a remote server, and (iii) the connections that exist between the landmarks, essential to compute a location.

Localization with respect to a digital map describing the road network is an essential task for intelligent vehicles [2, 3]. The user of a vehicle usually specifies its itinerary by indicating the destination address. In this case, the geocoding facility of the GIS is very useful when converting an address like "10, Albert Road" into global (x, y) coordinates.

The GIS can also be used for the management of landmarks by making use of the road descriptions stored in the map database. An efficient technique is to group the landmarks in local maps corresponding to the different roads [4]. Indeed, the local maps can have the same identification numbers (hereafter denoted as ID) as the roads. From a real-time point of view, this is essential since the local maps can be stored in a look aside database (LADB) and easily be retrieved and loaded into the vehicle's memory for localization purposes.

This paper focuses on absolute localization and the ID retrieval problem. In particular, it deals with real-time constraints and positioning integrity. The implementation of the precise localizer is not considered here. Unlike many industrial prototypes that have mainly low frequency (often 1 Hz), this work also deals with the development of high-frequency systems. Moreover, we consider a matching strategy that does not rely on the use of a precomputed route, since our system is not limited to applications dealing only with navigation.

We consider a vehicle equipped with an odometer (we use the ABS sensors of the rear wheels), a fiber-optic gyrometer, an L1 (single-frequency) differential GPS receiver, and embedded GIS software managing a standard road map database (NavTeQ in this particular case [2, 3]).

During the first stage, the DR sensors are merged with the GPS fixes using a loosely coupled Kalman filter approach. Thanks to the predictor/estimator mechanism in the pose tracking process, GPS latency is eliminated. The second stage involves fusion with the map data. A request is sent to the GIS server (embedded or remote) that returns the roads contained in a box whose center and size have been instantiated in the request. Then, a road selection method (also called map matching) is used to select the most likely segment and finally this segment is merged with the state estimate. The management of a road cache memory is an important issue. If it is too small, the road selection can fail. If it is too large, the selection will be uselessly time-consuming. In addition, the management of the map has to be done spatially and just when necessary.

The paper is organized as follows. In Section 2, the localization method fusing GPS and DR sensors is described and the method for compensating the GPS latency is introduced. The fusion of the map data is then presented in Section 3, and the complete algorithm, the map representation, and the road selection strategy are described. We look at the management of the road map's cache memory and show that the right road is always present in the cache, while the worst-case execution time is respected and inaccuracies are taken into account. Finally, Section 4 is devoted to real experiments that illustrate the performance of the fusion process, the latency, and cache management.

2. GPS AND DR SENSORS FUSION

Localization using DR sensors and GPS data can be achieved via a multisensor fusion approach, that is, an approach that explicitly takes into account inaccuracies to handle redundancy and complementarity.

From the real-time point of view, efficient methods are those that rely on state observation because of the predictor/estimator mechanism that can be implemented as a recurrent task (termed *complex task* by Kopetz [5]) that only needs to keep in memory the state vector between two sampling times.

Let us first consider the fusion of DR sensors with GPS. The GPS fixes are projected in a local frame tangential to the surface of the earth.

This fusion problem can be expressed by a discrete state-space representation, sampled with respect to time:

$$\begin{aligned} X(t_k) &= f(X(t_{k-1}), U(t_k)), \\ Y_g(t_k) &= g(X(t_k)), \end{aligned} \tag{1}$$

where

(i) $t_k = t_0 + k \cdot Te$;
(ii) $X(t_k) = [x(t_k) y(t_k) \theta(t_k)]^T$ is the pose (position and heading) of the vehicle in the projected frame;
(iii) the evolution model corresponds to the DR model;
(iv) $U(t_k) = [\delta(t_k) \omega(t_k)]^T$ is the vector of the elementary traveled distance and elementary rotation measured by the DR sensors;
(v) $Y_g(t_k) = [x_g(t_k) y_g(t_k)]^T$ is the GPS measurement vector after projection.

The evolution model of the center of the rear axle can be expressed by

$$\begin{aligned} x(t_k) &= x(t_{k-1}) + \delta(t_k) \cdot \cos\left(\theta(t_{k-1}) + \frac{\omega(t_k)}{2}\right), \\ y(t_k) &= y(t_{k-1}) + \delta(t_k) \cdot \sin\left(\theta(t_{k-1}) + \frac{\omega(t_k)}{2}\right), \\ \theta(t_k) &= \theta(t_{k-1}) + \omega(t_k). \end{aligned} \tag{2}$$

If the GPS antenna is located at the origin of the mobile frame, then the observation equation becomes

$$Y_g(t_k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot X(t_k). \tag{3}$$

A very popular approach for fusing localization data is the Bayesian framework. An extended Kalman filter (EKF) is often used. Unscented Kalman filtering (UKF) [6] is a new approach, very popular currently, because it can more precisely estimate the error covariance than an EKF, especially if this covariance is large with respect to the nonlinearity. In this paper, the equations of the filters are not detailed. For more information, the reader can see [7].

In order to implement a Kalman state observer, we propose to choose the maximum sampling period for two reasons:

(a) reduction of processing workload,
(b) it is well adapted to handle GPS latency.

Tessier et al. [8] have proposed an architecture for the fusion of delayed observations using data buffers: each piece

of data being timestamped and buffered together with the filter estimations. Once a new piece of data appears, the fusion is computed in the past and the current estimation is recomputed from the buffered data. We propose a strategy that compensates for the GPS latency by using an xPPS (xpulses per second) signal, a GPS receiver running in synchronized mode, and a predictor implementation of the state observer. This strategy is more efficient than Tessier's, but works only for small delays. It consists in delaying to the maximum the use of the GPS data in the algorithm in order to leave for it enough time to compute and transmit its results. If the sampling period is higher than the worst-case GPS latency, one solution is to perform the GPS estimation stage first with each new sample (see Figure 1). By this way, the first step at time t_k is to correct the pose $X(t_{k-1})$ using $Y_g(t_{k-1})$. Once this has been done, the prediction stage provides an estimate of state $X(t_k)$ using the DR sensors. This is the output of the filter that works as a one-sample DR predictor.

It should be noted that a good initialization of the Kalman filter, especially of the heading, accelerates significantly its convergence. Moreover, in order to filter GPS jumps due to multitracks (especially in urban areas), a coherence test based on a Mahalanobis distance is applied in the correction stage of the filter [4].

The strategy we have chosen can be expressed as follows. When few GPS fixes are inconsistent with the DR predictions, we suppose that it is the GPS that is faulty. Otherwise, that is, when GPS and DR prediction are inconsistent for a long time, the localizer is reinitialized as shown in Figure 2. The initialization consists in waiting for a new GPS fix. The tracking is done by an EKF, the implementation of which follows the chronogram of Figure 1.

This strategy is also robust when initializing or reinitializing the system with bad GPS data. Let us study the system behavior when this case happens. Two cases can occur with a bad reinitialization, depending on the validity of the information contained in the GST NMEA sentence.

Case 1. The position is bad and the confidence ellipsoid is large. If a new good GPS fix arrives, it will be considered consistent, and therefore used to correct the previous estimate.

Case 2. The position is bad and the confidence ellipsoid is small (inconsistent data). The filter will not be able to detect this at once, since the initial covariance is small. When good GPS fixes are restored, the filter will consider them as bad data until a new reinitialization occurs. It should be noted that multitracks are often very short for a moving receivers. So, the probability of the filter being reinitialized in this case remains low.

To obtain a high positioning frequency (e.g., a video application client working at 60 Hz and requiring positioning data at the same rate), a client-server mechanism can be implemented. Between two low-rate pose computations, the positioning server extrapolates the previously computed pose using the known linear and rotational speeds (cf. (2)).

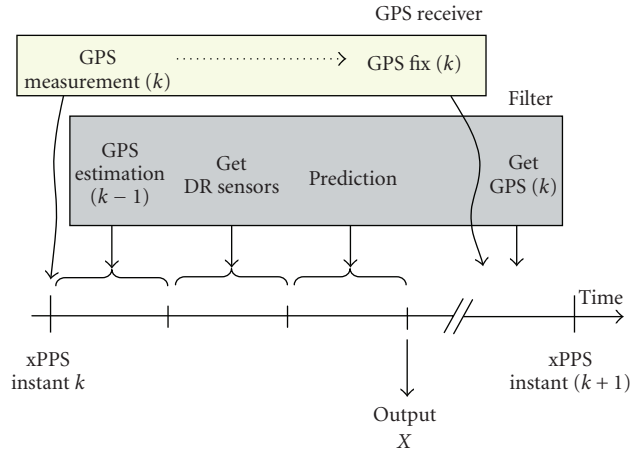


Figure 1: Chronogram of the state observer for GPS latency compensation. Because of the synchronized mode of the GPS receiver, it takes its pseudorange measurements at the xPPS instant but provides a solution when it has completed its computation.

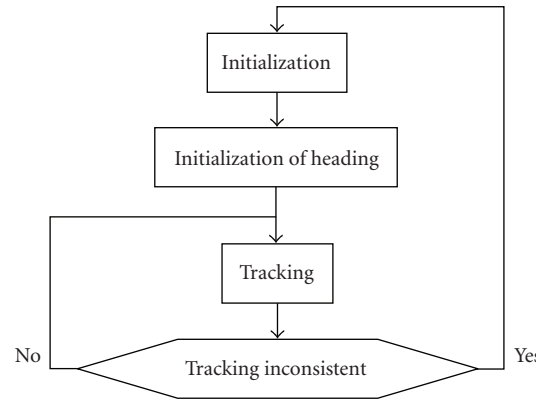


Figure 2: Synopsis of the states of the localizer.

This mechanism also allows the transmission of positioning data to different clients at different request rates, using the same position computations.

If the vehicle is motionless, the heading is not observable. In this case, to avoid error, the previously computed pose is maintained, and no prediction or updating is performed until the vehicle starts to move again. In practice, we check that the odometers' counters have changed since the previous step. This corresponds to a spatial sampling condition equal to the sensor's resolution, which is here about 2 cm.

3. GPS, GIS DATA, AND DR SENSORS FUSION

Because of GPS outages occurring in urban areas for instance, an effective means of correcting DR localization drift is to use map data as an observation in the filtering process. This can be done by serializing the two correction stages, as shown in Figure 3. The GPS correction stage is carried out at

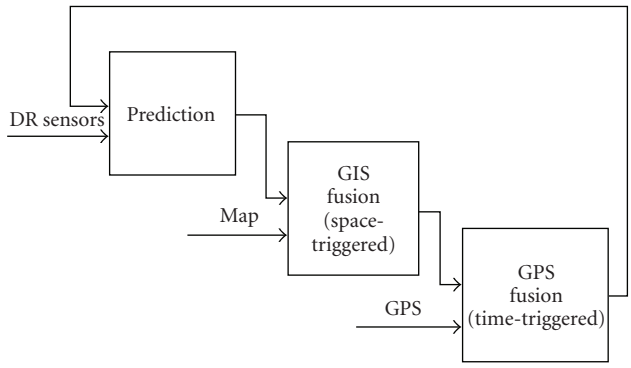


Figure 3: Hierarchical fusion strategy.

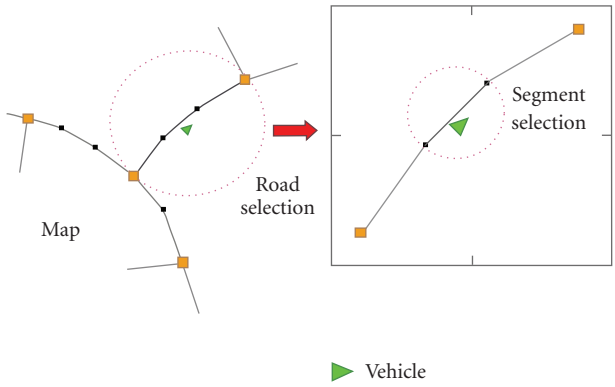


Figure 4: Road selection and segment selection.

the end in order to apply the same GPS latency management strategy as before.

Localization on a map is a problem that can have several solutions, for instance while approaching junctions, when the quality of GPS positioning is poor, or given a map with a poor absolute precision. Particle filtering has shown interesting characteristics in relation to this problem, particularly because of its ability to manage several hypotheses [9]. Nevertheless, its real-time implementation [10] is time consuming and its convergence with a small number of particles is not guaranteed because of the degeneracy problem that can occur. An alternative would be doing a pose tracking with the most likely road. To illustrate this, let us explore the concepts of roads and segments.

A road map is usually a set of digitized roads described by polylines represented by their centerline. Their topological information is very good, while their geometry is often rough. Each junction is represented by a *node*. *Shape points* are used to enhance the geometry description. By definition, a road is a polyline linking two nodes. A segment is defined as the linear interpolation between two points being either nodes or shape points (see Figure 4).

3.1. GIS data fusion

For real-time computation purposes, we adopt a monohypothesis approach: one road only is used in the tracking process. If the filter should select the wrong road, this error is detected thanks to the GPS, and the filter is reinitialized.

An important characteristic of map matching is its *spatial* nature. A time-triggered approach is not well adapted for this problem since it is not elapsed time but traveled distance (also called abscissa curvilinear) that is important for the convergence. Moreover, many approaches rely on data fusion approaches that suppose independence of the errors. If the vehicle is motionless, then the same map data can be used several times, violating the independence hypothesis. For these reasons, map matching can be formulated by a space-triggered state-space description.

Let suppose first that a candidate segment has been selected:

$$\begin{aligned} X(l_i) &= f(X(l_{i-1}), U(l_i)), \\ Y_m(l_i) &= h(X(l_i)), \end{aligned} \tag{4}$$

where

(i) $l_i = l_0 + i \cdot Le$ is ith sample of the traveled distance from the beginning, Le the sampling distance;
(ii) $X(l_i) = [x(l_i)\ y(l_i)\ \theta(l_i)]^T$ is the pose of the vehicle in the frame of the map;
(iii) the evolution model is the DR navigation model;
(iv) $U(l_i) = [\delta(l_i)\ \omega(l_i)]^T$ is the vector of the elementary traveled distance and elementary rotation measured by the DR sensors;
(v) $Y_m(l_i) = [x_m(l_i)\ y_m(l_i)]^T$ is a point (which we term map measurement) that corresponds to the projection of the estimated position onto the most likely segment (see (5) and Figure 5):

$$Y_m(l_i) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot X(l_i). \tag{5}$$

The fusion of the estimate with the map is performed during a Kalman estimation stage. The covariance associated with the map observation is modeled by an ellipsoid around the selected segment as shown in Figure 5 [11]. The center of the ellipsoid is Y_m, the orthogonal projection on the segment of the last estimated location. In the frame associated with the segment, the longitudinal inaccuracy is far greater than the lateral inaccuracy. Theoretically, the longitudinal inaccuracy can be chosen as large as possible, even infinite for a long segment. In practice, we consider a one-sigma value in the order of the length of the segment.

Before doing the correction stage, a consistency test is applied to check if the selected road is correct. If not, a road selection stage or a segment selection stage is carried out.

3.2. Road and segment selections

In order to fuse the road information with the estimated position, the system needs to know the most likely segment.

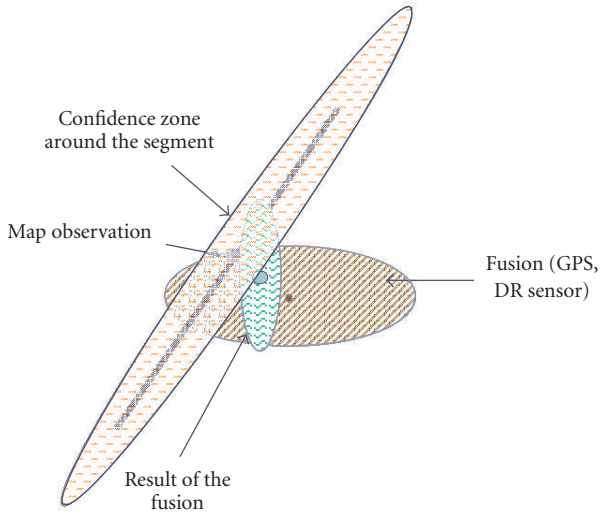


Figure 5: Fusion of the estimate with the selected segment.

The following situations can arise:

(a) there is no selected road (initialization);
(b) the system needs to select the most likely segment of the current road;
(c) the vehicle is approaching the end of a road.

Cases (a) and (c) are road-selection problems, while (b) is a segment-selection issue.

A number of solutions for solving case (a) exist in the literature [11]. Where a high level of integrity is required, the process may be particularly sophisticated, although in this paper, where our concern is efficiency, we propose a simple approach. First, the road segments whose orientation is compatible with the vehicle heading are selected. The heading information stored in the road map structure is used here to accelerate this processing (Section 3.5). Then, in this set, the segment with the smallest distance from the estimated position is chosen.

Now, let us suppose that the selected road is consistent with the current pose. The goal is to select the most likely segment of the road (if the road is a polyline with at least two segments). The most likely segment is simply the closest one.

Finally, suppose that the vehicle is approaching the end of the road. This road selection consists in selecting the most likely road connected to the previously selected road. This connectivity information is once again contained in the array of road IDs connected to the road origin and endpoint, in the road map structure. An efficient strategy is to select the road whose direction corresponds most closely to the heading estimation.

Given that the risk of selecting the wrong road is high when the vehicle's estimated location is close to a node (ambiguity area), we use a careful strategy: the map correction is not computed and the connected road selection is not performed until the vehicle leaves the ambiguity area. This helps to reduce erroneous matches, especially when the map contains large errors.

(A) Initialization(·)
Wait4GPS(·)
Heading_initialization(·)
Δ4MF = 0 //Distance for map fusion
road = select_new_road(x_k, map_cache)
Counter=0 //GPS outliers

(B) Tracking loop trigged at each xPPS
Δ_k=Get_traveled_distance(); $\delta_k = \Delta_k - \Delta_{k-1}$
Ω_k=Get_heading_rotation(); $\omega_k = \Omega_k - \Omega_{k-1}$
If $\delta_k \neq 0$ //the vehicle is moving
Then
(x_{k-1}, Counter) = correction(x_{k-1}, y_gps$_{k-1}$)
If (Counter > threshold) Then Initialization(); break;
End
x_k=prediction(x_{k-1}, δ_k, ω_k)
If ($\Delta_k - \Delta$4MF>Le)
Δ4MF=Δ_k
(Seg, matched_pt) = select_seg(x_k, road)
TH = Map_error + HE; //ambiguity zone size
If (get_dist_to_road_node(matched_pt)>TH) Then
(x_k, Inconsistency)=fusion(x_k, Seg)
If (Inconsistency is bad) then
road = select_new_road(x_k, map_cache)End
Else //zone of ambiguity
road = select_connected_road(X_k, road)
End
End
End
Function new_road = select_connected_road(X_k, current_road)
Selects the road which is connected to road and is most consistent with X_k, Eventually, returns current_road

Function road = select_new_road(x_k, map_cache)
Selects the most consistent road with X_k

Function seg = select_seg(x_k, current_road)
Selects the most likely segment of the current road

Function dist=get_dist_to_road_node(matched_pt)
Returns the distance to the current matched point

Algorithm 1

3.3. Algorithm

The global fusion algorithm is spatial- and time-triggered. It uses the serial fusion strategy shown in Figure 3, implements the GPS latency management described in Section 2, and has the same running modes as the GPS and DR fusion algorithm (Figure 2). We now consider its pose tracking strategy (see stage B Algorithm 1). The fundamental trigger is the xPPS signal obtained from the GPS receiver (the xPPs is assumed always to be available, even during long outages).

The traveled distance and heading rotation are first obtained from the DR sensors. If the vehicle is moving, the correction of the previous prediction is computed using the previous GPS fix. Using a Mahalanobis test, an

incoherence counter (see *Counter*) is incremented (when bad GPS data is detected) or reinitialized (each time a coherent GPS data is received). The predicted state is computed. If the traveled distance since the last fusion with the map is higher than *Le* (spatial sampling period), a segment selection is performed. The Kalman correction stage with respect to the selected segment is performed only if the vehicle is not in the zone of ambiguity (see *function get_dist_to_road_node(matched_pt)>TH*). The zone of ambiguity, defined by the threshold *TH*, includes an estimation of the map error (assumed to be known, e.g., 10 meters) and an estimation of the accuracy of the estimated position (termed HE, i.e., horizontal error). HE is defined to be the 1-sigma error circle, that is, the maximum eigenvalue of the position covariance matrix. If the fusion of the segment is not consistent (see (x_k, *Inconsistency*) = fusion(x_k, *Seg*)), then a new road selection stage occurs. While the vehicle is located in the zone of ambiguity, a connected road is sought (see *select_connected_road*(X_k, *roa*)).

It will be noticed that the tracking mode of the localizer depends only on the coherence of the GPS points with respect to the state estimate. By reinitializing the localizer, this mechanism solves map-matching mistakes.

One should also note that the GPS bad data and the map-matching errors are not handled in the same way, since the GPS outliers can come from a change in satellite constellation, multitracks, and are rather brief, while the matching errors are derived from an erroneous segment-selection mechanism.

3.4. Integrity and behavior analysis of the algorithm

Integrity can be defined as the confidence which can be placed in the correctness of the information supplied by the whole system. The most robust way to ensure that the positioning information is valid is to have a multilayer series of checks [12]. It is important to have integrity checking at the end-user level because this is the only place where all information used to form the position solution are present. For a road vehicle, real-time integrity estimation is very challenging because it has to take into account the degradation of the satellite visibility due to the environment of the vehicle and map errors.

Here, there are several problems to tackle:

(i) convergence of the method,
(ii) tracking divergence detection due to map and positioning errors,
(iii) bad GPS fixes arising from multitracks for instance,
(iv) map cache management which means here to keep the estimate in the cache with a guaranty zone.

The last point is studied in Section 3.6.

3.4.1. Nonobservable situation

If the vehicle is motionless, the heading is not observable. In this case, thanks to the spatial triggering, no computation is done. This prevents any drift.

3.4.2. Road tracking

For simplicity, let us suppose that the road is described by an infinite polyline that corresponds to the right road. If the GPS is available and is coherent with the DR sensors, then the positioning is good and the map matching is trivial. If there is no GPS, the map observation is only able to correct the transversal DR drift [13]. It is well known [14] that lateral drift exceeds longitudinal drift. So, the map observation that we have proposed models correctly this phenomenon, since the map ellipsoid has a small lateral standard deviation and a large longitudinal.

3.4.3. Convergence in case of a bad road selection

Suppose now that the filter has been initialized with an incorrect road (because of a bad road selection in a zone of ambiguity, or because of bad GPS fixes). The only property that can be proved is that if a bad choice occurs, the system is able to detect it after a bounded duration or traveled distance.

There are two cases.

(a) The GPS is good and coherent: if the location is inside a zone of ambiguity, the fusion with the map is not performed and the road-selection error is undetectable. As soon as the vehicle leaves this zone, the fusion with the map will be incoherent and a new road selection is done (*select_new_road()*).
(b) The GPS is poor: as soon as good GPS is restored, the filter will consider it as a bad data and after several steps (counter > threshold), the filter will be reinitialized.

3.4.4. Robustness in relation to GPS outliers

Suppose now that the GPS receiver suffers from multitracks. If this phenomenon is short, then the incoherent fixes will be rejected (counter < threshold). Please note that the map can help in detecting GPS errors. If not, the filter will be reinitialized with bad GPS fixes and remains in that state until good GPS is restored.

3.4.5. Robustness in relation to bad map data

Let us consider the case where the map is very bad (e.g., in intertown situations). In this case, there will never be any map fusion and the system will continuously carry out new road selections without making any map fusion.

3.5. Map representation

In order to facilitate the road selection and fusion processes, the map representation is enhanced. Let us consider its structure.

The extracted map is a vector of road structures; each road structure element contains the following:

(i) road or street *name,*

(ii) *ID*, that is, a unique identifier for each road: this is indispensable for navigation and map-matching purposes,

(iii) *speed limit,*

(iv) *driving direction* information: this indicates whether the road is a one-way segment (from east to west or west to east) or a two-way road. This field also indicates if the road is restricted to pedestrians or cars,

(v) number of shape points that describe the geometry of the road,

(vi) *shape points* coordinates array,

(vii) *heading angles* of the segments defined by the shape points,

(viii) number of connected segments to the road *origin pointn*,

(ix) IDs array of the connected segments to the road origin point,

(x) number of connected segments to the road *endpoint*,

(xi) IDs array of connected segments to the road endpoint.

The connectedness is described by the arrays containing the segments IDs.

3.6. Map cache memory management

Let us suppose that we have a GIS server, which can be either embedded or remote. For an efficient map management, let us consider a map cache memory corresponding to an area whose shape is a square. Such a geographical zone is easy to extract since no distance has to be computed: roads are extracted using only tests on the road segment coordinates. The management of the cache memory is a parallel task with the filter computations.

The center c and the semilength r of the side of the square have to be specified in the request made by the fusion client. When a request is received, all the roads partially or completely included within the square of center c and side length $2r$ are extracted.

For an intelligent map cache memory management, two problems have to be dealt with: center and semilength management.

Since the vehicle continues to move while the map cache requests are made, a predicted request center has to be computed to take into account the worst-case server process and transmission delay (denoted as $T_{\text{GIS}} = T_{\text{Extraction}} + T_{\text{Transmission}}$). The map extraction therefore involves anticipation: we use the vehicle's estimated heading and speed.

Let us denote by $X_c = [x_c \; y_c]^T$ the coordinates of the center of the cache memory. Once the vehicle is close to the limit of the current cache (see Figure 6), a request is sent to the server.

The condition of the request is given by (6) where the distance between the current vehicle position and the center of the cache is compared to semilength of the cache. λ is a parameter necessary for managing the request to the server in realtime. It corresponds to the proportion of r from which a request has to be made before leaving the area of the current cache:

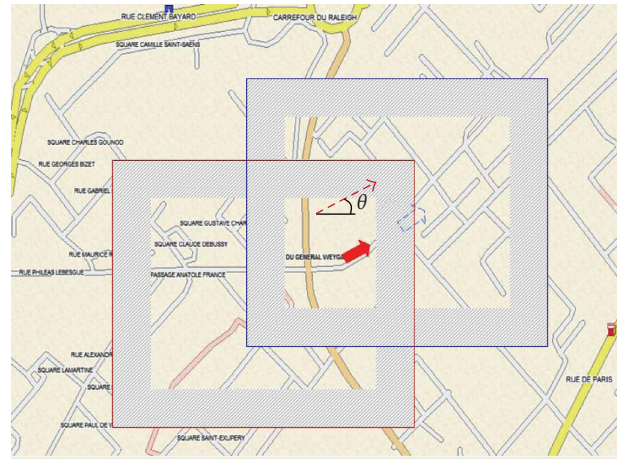


Figure 6: Extrapolation of the cache center while approaching the limit of the current cache (the gray zone corresponds to parameter Δ).

$$||X_c(t_{k-1}) - X(t_k)|| \geq \lambda \cdot r. \tag{6}$$

If the request condition is verified, the new center position is computed using a nominal speed v, for instance 50 Km/h in urban areas:

$$\begin{aligned} x_c(t_k) &= x(t_k) + T_{\text{GIS}} \cdot v \cdot \cos(\theta(t_k)), \\ y_c(t_k) &= y(t_k) + T_{\text{GIS}} \cdot v \cdot \sin(\theta(t_k)). \end{aligned} \tag{7}$$

While the request is being processed, the vehicle does not remain stationary. The risk is that it might leave the current cache area. A security zone is characterized by the following condition on the traveled distance:

$$T_{\text{GIS}} \cdot v \leq (1 - \lambda) \cdot r. \tag{8}$$

This leads to the following condition:

$$\lambda \leq 1 - v \cdot \frac{(T_{\text{transmission}}(r) + T_{\text{extraction}}(r))}{r}. \tag{9}$$

An interesting value is the distance (denoted as Δ) before leaving the region of interest:

$$\begin{aligned} \Delta &= (1 - \lambda) \cdot r, \\ \Delta &> v \cdot T_{\text{GIS}}(r). \end{aligned} \tag{10}$$

Suppose now that the maximum map inaccuracy (denoted as E_M) is known and the localization inaccuracy (denoted as E_L) is estimated in real time by the Kalman filter. In order to have a reliable road selection, Δ must verify

$$\Delta > E_M + E_L + v \cdot T_{\text{GIS}}(r). \tag{11}$$

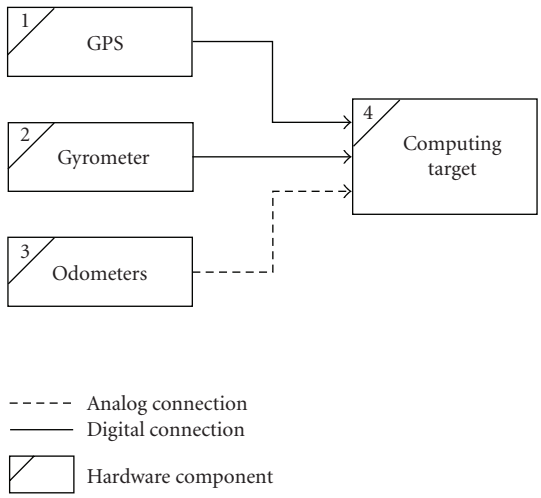


Figure 7: Architecture diagram of the hardware components.

The management of the semilength of the square is the second most important issue in cache management. If r is too small, then the road selection can fail due to localization and map errors. Otherwise, if r is too large, then

(i) the memory cache retrieval can be long (especially if the GIS server is remote);
(ii) the size of the cache can exceed the maximum size of the target's RAM;
(iii) the road-selection procedure can be uselessly time-consuming.

In summary, the cache memory management depends mainly on parameters Δ and r, which have to be determined regarding the implementation. A case study will be presented in the next section.

4. EXPERIMENTS

4.1. Embedded real-time platform

In this section, we describe the different hardware and software components used to test and validate our localization system.

The hardware components are separated into two categories: sensors and computing resources. The architecture diagram in Figure 7 shows the interaction between these components.

The components of Figure 7 are as follows.

(i) A computing target (4): a Shoebox PC with a low electrical power consumption processor (Intel Pentium M 1.5 GHz, 512 MB RAM), with a 12 V power supply and running Windows XP.
(ii) A GPS receiver (1): a Trimble AG132 DGPS with EGNOS/RTCM corrections. It is connected to the computer via a serial RS232 link at 38400 bauds.
(iii) A gyrometer 2: a fiber-optic KVH ECore2000 connected via a serial RS232 link at 9600 bauds.

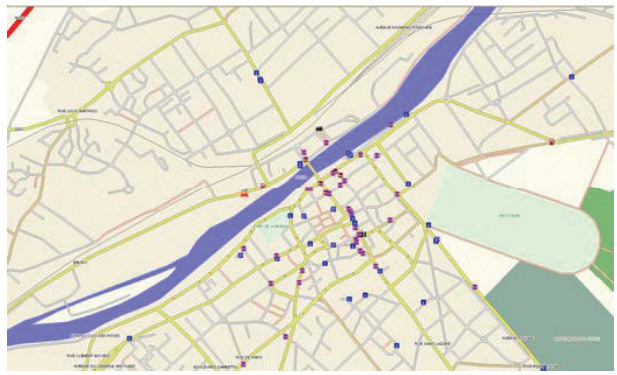


Figure 8: Map rendering using BeNomad SDK.

(iv) Two odometers whose values are obtained thanks to the ABS sensors of the rear wheels of the car. The sinusoidal signal generated by the rotation of each wheel is applied to a national instruments 16-bit counter. This provides measurements of the distances covered by the two wheels (1 top corresponds to 2 centimeters).

The interfacing programs were developed in C++, and all the data are timestamped and stored in a binary format for rapid prototyping purposes. Our goal is to have timestamps as close as possible to the sensor measurements in order to fit the model. To obtain the data via the serial link, we used an asynchronous driver which enables the data to be timestamped upon their arrival at the port and eliminates the decoding time. For the GPS, two kinds of data are received:

(a) on the Rx pin, NMEA0183 frames which contain the navigation data;
(b) an xPPS signal on the ring indicator pin.

The GIS used by the map-matching module is based on a software development kit (SDK) provided by BeNomad [15], see Figure 8. This SDK is completely object-oriented and cross-platform (Windows, Linux). It is also available for embedded targets such as PDAs or smart phones running Windows CE and Windows Mobile. The maps are size-optimized and provided in the SVS (scalable vector system) file format. For our prototype, we used a NavteQ geographical database converted to SVS format. The SVS format is very compact: the file size for the whole of the town of Compiegne is only 68 KB, and that for the entire OISE department is only 3 MB.

4.2. GPS latency experiments

Real-time experiments were carried out on a specialized test track in Versailles using one of our experimental cars (Figure 9).

In order to compute estimation errors, a L1/L2 Thales Navigation GPS receiver was used in a post-processed kinematic mode working with a local base (a Trimble MSi 7400). This system was able to give reference positions with a 1 Hz sampling rate. Since the constellation of the satellites was sufficiently good throughout the

Figure 9: Used experimental car and experimental test track.

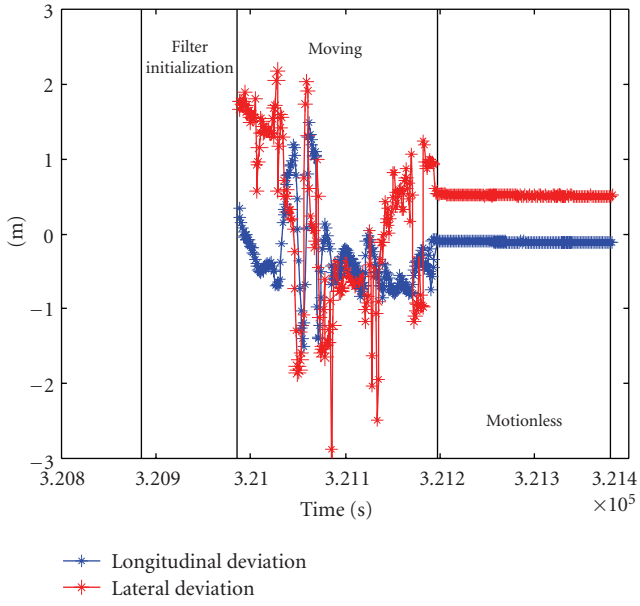


Figure 10: Filter output errors (maximum speed 90 Km/h).

trials, all the kinematic ambiguities were fixed. Accuracy was therefore guaranteed to within a few centimeters. The synchronization between this reference and the outputs of the localizer was achieved using the GPS timestamps.

Before implementing the filter, we measured the latency of the Ag132 receiver using an oscilloscope. We observed a 180 millisecond maximum latency corresponding to the worst case, that is, with 8 satellites used in the computation. Therefore, in order to implement the GPS latency management of Section 2, the receiver was tuned to 5 Hz, which guarantees that the GPS data are available before the next xPPS signal.

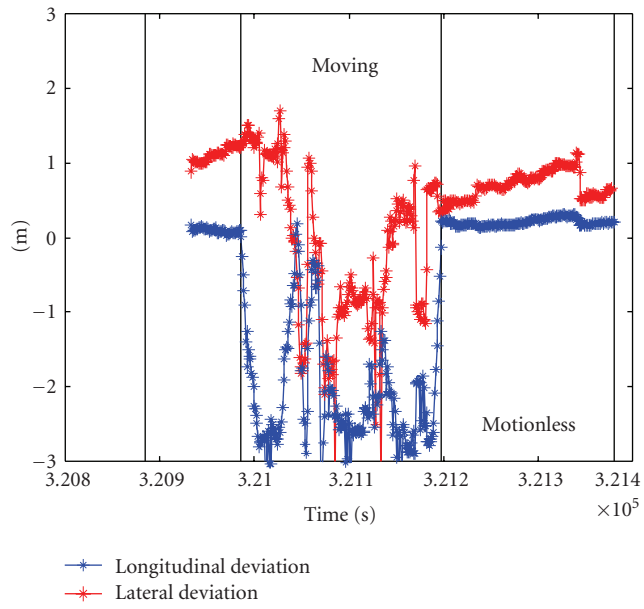


Figure 11: GPS output errors on the same trial.

Figures 10 and 11 are courtesy of Kais et al. [16], featuring the output of our localizer and the rough GPS files. Theses plots correspond to the longitudinal and lateral errors in a Frenet's frame.

Itcan be seen in Figure 11 that the longitudinal GPS error can be as high as 3 meters because of its latency, while this error is significantly reduced by the filter thanks to the predictor and the synchronization strategies used. Moreover, it is important to notice the efficient filtering of the multisensor localizer while the vehicle is motionless.

4.3. Map cache experiments

As a source digital map, we used a NavteQ database. In this map, coordinates are integers in centimeters, in the French Lambert 93 coordinate system.

We took a position in a downtown area (city hall of Compiègne). This position is the center of the GIS request. The semilength r of the square search has been instantiated from 10 m to 1 Km with a 1 m step.

Figure 12 shows the map-extraction time-consuming process and the road-selection time-consuming process versus the client request radius. For each plot, a second-order approximation was made, plotted in bold blue for each subplot. As a matter of fact, theses processes are dependent on the size of the area of interest, proportional to r^2.

We note that the two processes are very efficient and also that the road selection time is relatively small when compared to the map extraction time. Even if r is large, the road selection duration never exceeds 0.5 millisecond, which confirms that it is negligible compared to the sampling period of tracking process (Te = 200 milliseconds).

Figure 13 plots the size of the extracted map cache regarding the request radius. Once again, an expansion of the

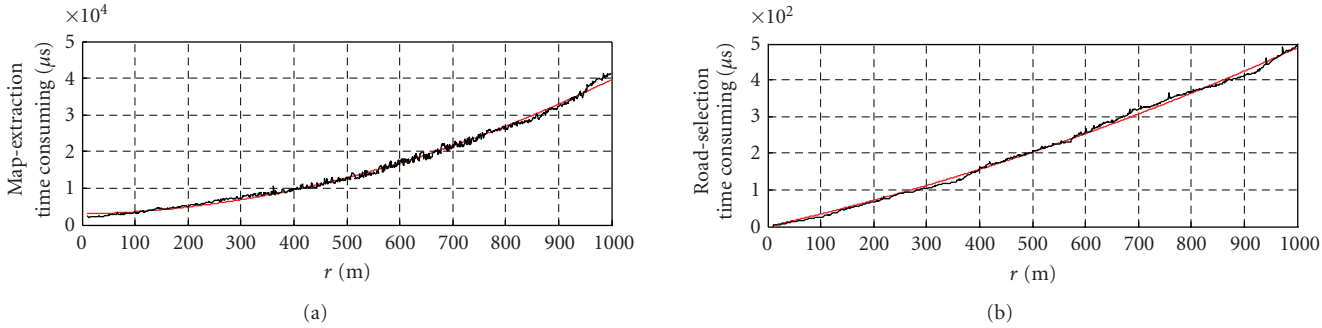


Figure 12: Map-extraction and road-selection time consuming for an urban area.

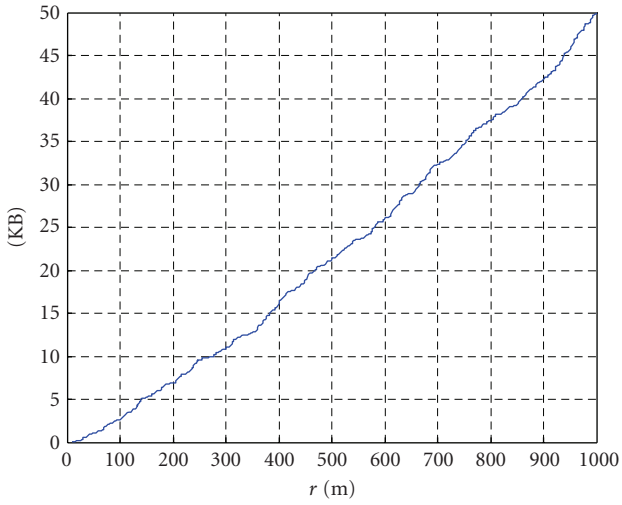


Figure 13: Map cache size regarding the size r of the request.

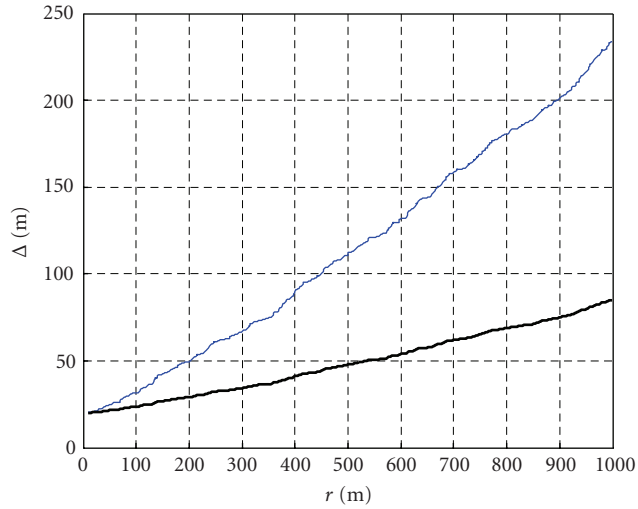


Figure 14: Δ versus r in the case of a remote server: in bold black 3G com link and in thin blue GPRS link.

search area implies a four-fold increase in the size of the map cache. Moreover, we can see that for a maximum radius of 1000 m, the cache size does not exceed 50 KB, which is not a constraint for the real-time target under consideration.

These results indicate that neither the size nor the computation times (extraction and road selection) are hard constraints for the cache management. r can be set very high.

Let us consider now the computation of Δ.

Suppose that the maximum map inaccuracy is $E_M = 15\,\text{m}$ and the localization inaccuracy is $E_L = 5\,\text{m}$ (as indicated by the experiments described in Section 4.2).

First of all, let us assume that the GIS server is embedded in the target and provides its data by using a shared memory or the middleware SCOOT-R [17] for instance. We choose $r = 1000\,\text{m}$.

On our target, the local transmission time between the server and the client was measured $T_{\text{transmission}}(1000\,\text{m}) = 745$ microseconds, and $T_{\text{map extraction}} = 41$ milliseconds (from Figure 12). Suppose that the vehicle speed equals 30 m/s = 108 km/h. Thanks to (11), we can compute that the request has to be sent at a distance $\Delta = 15+5+(745\cdot10^{-6}+41\cdot10^{-3})\cdot 30 = 21\cdot 25\,\text{m}$ only before leaving the cache area. Therefore, the map cache management is mainly constrained by the localization and map errors.

Let us suppose now that the GIS server is remote and let us consider 2 kinds of network: 3G (3rd generation) and GPRS (general packet radio service) connection [18]. For realistic considerations, we took the half of the maximum bandwidth: for GPRS, we took the half of 115.2 Kb/s, and for 3G, the half of 384 Kb/s.

Figure 14 shows Δ versus r that gives the distance to the cache limit that it is necessary to respect in order to obtain reliable road selection. This plot was estimated numerically by using (11) where $T_{\text{extraction}}(r)$ was the second-order approximate of the time extraction process in function of r, and $T_{\text{transmission}}(r)$ was the transmission delay of the extracted map within the network. This delay depends on the size of the map and on the bandwidth of the network.

In summary, the methodology for cache memory management is as follows.

(i) Estimate the road selection duration versus r.
(ii) Choose a value of r such that the duration is compatible with
  (a) the sampling period of the localizer,
  (b) RAM size of the target,
  (c) the map error,
  (d) the localizer error.
(iii) Compute Δ corresponding to the implantation of the GIS server.

In our case, the road selection routine not being very time consuming, and the size of the cache memory not being a limitation, we suggest using a large cache area (r in the order of 1 km) in order to obtain a reliable behavior.

5. CONCLUSION

This paper has considered the real-time implementation of a localization system that uses DR sensors, GPS, and GIS data following a loosely coupled paradigm. Such a system is a key component for intelligent vehicles since it can constitute the basis for precise localization or the development of advanced driving assistance systems. Its main outputs are the pose of the vehicle in the projected frame of the map and the ID of the most likely road in case of successful map matching (otherwise an off-road situation is detected).

An efficient implementation relies on pose tracking using Kalman filtering, after a good initialization. Because of the GPS latency, we have proposed a strategy that involves delaying the GPS correction stage until last, the filter outputting DR predictions with small latency.

The multisensor data fusion problem has been modeled as a space-time problem with two pose trackers. Because of the GPS receiver, a time-triggered approach is necessary. The trigger is the xPPS signal. For the road-selection problem, the spatial nature is essential. Therefore, a mechanism for triggering the filter according to the vehicle displacement has been developed. At each fusion step, a coherence test is applied. If the GPS is incoherent for several samples, the localizer is reinitialized. This guarantees that wrong map matches can be corrected.

A key aspect of this system is the map representation for an efficient road selection. We have seen that our proposed representation is very pertinent to this consideration. Another issue is the map cache memory management that is performed in parallel by considering spatial, rather than temporal, conditions. This is essential for an embedded system, since such a strategy gives rise to computations only when necessary. Two aspects have to be dealt with: the center and the semilength of the square search area. They are linked together and we have identified the key points. The first step consists in evaluating the duration of the road selection with respect to the size of the area. Then a compromise has to be found between map-matching integrity and workload (in case the GIS server is embedded) or workload and communication (if the GIS server is remote).

This prototype is currently used for the management of visual landmarks memory. Thanks to the use of the filter, it is possible to send requests to the positioning server at a video rate.

One perspective of this work is to use an electronic horizon to manage the cache memory. An electronic horizon is a graph of the accessible roads from the current pose.

ACKNOWLEDGMENT

This research has been carried out within the framework of the European FP6 Integrated Project CVIS, *Cooperative Vehicle Infrastructure Systems*, started in February 2006 for 4 years.

REFERENCES

[1] E. Royer, J. Bom, M. Dhome, B. Thuilot, M. Lhuillier, and F. Marmoiton, "Outdoor autonomous navigation using monocular vision," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '05)*, pp. 1253–1258, Edmonton, Canada, August 2005.
[2] http://www.navteq.com/.
[3] http://www.teleatlas.com/.
[4] M. Jabbour, Ph. Bonnifait, and V. Cherfaoui, "Enhanced local maps in a GIS for a precise localisation in urban areas," in *Proceedings of the 9th IEEE International Conference on Intelligent Transportation Systems (ITSC '06)*, pp. 468–473, Toronto, Ontario, Canada, September 2006.
[5] H. Kopetz, "Time-triggered real-time computing," in *Proceedings of the 15th World Congress of the International Federation of Automatic Control (IFAC '02)*, IFAC Press, Barcelona, Spain, July 2002.
[6] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Proceedings of the 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls*, Orlando, Fla, USA, April 1997.
[7] R. G. Brown and P. Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering: With MATLAB Exercises and Solutions*, John Wiley & Sons, New York, NY, USA, 1996.
[8] C. Tessier, C. Cariou, C. Debain, F. Chausse, R. Chapuis, and C. Rousset, "A real-time, multi-sensor architecture for fusion of delayed observations: application to vehicle localization," in *Proceedings of the 9th IEEE International Conference on Intelligent Transportation Systems (ITSC '06)*, pp. 1316–1321, Toronto, Ontario, Canada, September 2006.
[9] F. Gustafsson, F. Gunnarsson, N. Bergman, et al., "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.
[10] S. Niklas, "Real time implementation of map aided positioning using a Bayesain approach," M.S. thesis, Linköping University, Linköping, Sweden, December 2002.
[11] M. E. El Najjar and Ph. Bonnifait, "A road-matching method for precise vehicle localization using belief theory and Kalman filtering," *Autonomous Robots*, vol. 19, no. 2, pp. 173–191, 2005.
[12] T. Walter and P. Enge, "Weighted RAIM for precision approach," in *Proceedings of the 8th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS '95)*, vol. 2, pp. 1995–2004, Palm Springs, Calif, USA, September 1995.
[13] S. Wijesoma, K. W. Lee, and J. I. Guzman, "On the observability of path constrained vehicle localisation," in *Proceedings*

of the 9th IEEE International Conference on Intelligent Transportation Systems (ITSC '06), pp. 1513–1518, Toronto, Ontario, Canada, September 2006.

[14] A. Kelly, "Linearized error propagation in odometry," *The International Journal of Robotics Research*, vol. 23, no. 2, pp. 179–218, 2004.

[15] http://www.benomad.com/.

[16] M. Kais, Ph. Bonnifait, D. Bétaille, and F. Peyret, "Development of loosely-coupled FOG/DGPS and FOG/RTK systems for ADAS and a methodology to assess their real-time performances," in *Proceedings of IEEE Intelligent Vehicles Symposium (IV '05)*, pp. 358–363, Las Vegas, Nev, USA, June 2005.

[17] K. Chaaban, M. Shawky, and P. Crubillé, "A distributed framework for real-time in-vehicle applications," in *Prceedings of the 8th IEEE International Conference on Intelligent Transportation Systems (ITSC '05)*, pp. 925–929, Vienna, Austria, September 2005.

[18] P. R. Muro-Medrano, D. Infante, J. Guillo, J. Zarazaga, and J. A. Banares, "A CORBA infrastructure to provide distributed GPS data in real time to GIS applications," *Computers, Environment and Urban Systems*, vol. 23, no. 4, pp. 271–285, 1999.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 29391, 11 pages
doi:10.1155/2007/29391

Research Article

Broadcasted Location-Aware Data Cache for Vehicular Application

Kenya Sato,[1] Takahiro Koita,[1] and Akira Fukuda[2]

[1] *Department of Information Systems Design, Doshisha University, 1-3 Tatara-Miyakodani, Kyotanabe-Shi, Kyoto 610-0321, Japan*
[2] *Graduate School of Information Science and Electrical Engineering, Kyushu University, 744 Motooka, Nishi-Ku, Fukuoka 819-0395, Japan*

Received 15 October 2006; Revised 7 March 2007; Accepted 17 April 2007

Recommended by Gunasekaran S. Seetharaman

There has been increasing interest in the exploitation of advances in information technology, for example, mobile computing and wireless communications in ITS (intelligent transport systems). Classes of applications that can benefit from such an infrastructure include traffic information, roadside businesses, weather reports, entertainment, and so on. There are several wireless communication methods currently available that can be utilized for vehicular applications, such as cellular phone networks, DSRC (dedicated short-range communication), and digital broadcasting. While a cellular phone network is relatively slow and a DSRC has a very small communication area, one-segment digital terrestrial broadcasting service was launched in Japan in 2006, high-performance digital broadcasting for mobile hosts has been available recently. However, broadcast delivery methods have the drawback that clients need to wait for the required data items to appear on the broadcast channel. In this paper, we propose a new cache system to effectively prefetch and replace broadcast data using "scope" (an available area of location-dependent data) and "mobility specification" (a schedule according to the direction in which a mobile host moves). We numerically evaluate the cache system on the model close to the traffic road environment, and implement the emulation system to evaluate this location-aware data delivery method for a concrete vehicular application that delivers geographic road map data to a car navigation system.

1. INTRODUCTION

As technologies of mobile computings and communications have become highly extensive and functional, computer systems equipped in a vehicle such as navigation systems with wireless communication have been popularized recently to contribute to improving road transportation, efficiency, and comfort. These systems generally provide drivers with traffic information, weather report, and other individualized data at the driver's request such as information on restaurants, amusement parks, landmarks, hospitals, and so forth through cellular phone networks. The wireless networks are relatively slow and unstable compared with wired communication networks.

Digital broadcasting for mobile hosts has been available recently; for example, one-segment broadcasting service [1] began in Japan on April 1, 2006. This broadcasting service for mobile hosts refers to the single segment set aside out of a total of 13 segments for customizable mobile broadcasting in each of Japan's home TV terrestrial digital channels. One current use for one-segment broadcasting is digital TV programs for mobile phones, portable devices, car navigation systems, and so on. In addition, data broadcasting has been specified [2] in the Association of Radio Industries and Businesses (ARIB), and some other application examples have also been proposed [3].

The broadcast services for mobile are additional candidates for disseminating location-aware data for vehicular applications. Using this broadcast method to deliver location-aware data is more scalable and less expensive in comparison with cellular phones. However, this broadcast delivery method has the drawback that a mobile host, which is part of an in-vehicle computer system, needs to wait for the required data items to appear on the broadcast channel.

This high performance digital broadcasting for mobile hosts has been available recently. However, broadcast delivery methods have the drawback that clients need to wait for the required data items to appear on the broadcast channel. In order to reduce the time a mobile host needs to wait, and to receive and store data effectively on the mobile host,

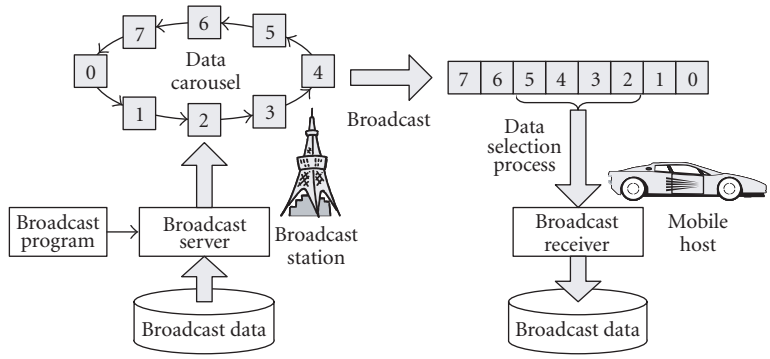


Figure 1: Broadcast model outline.

a caching mechanism is necessary. The mobile host does not have to wait for the data item to appear on the broadcast channel if the data is stored in a cache. The idea of caching broadcast data is not new and there are existing proposals for data delivery to a mobile host [4]. Generally, a least recently used (LRU) method has been adopted for data replacement policies, although Acharya et al. proposed the PIX and PT methods [5] to invalidate useless data in a cache and prefetch useful data among the broadcast data. Although these methods are optimal policies, they are impractical since they all require complete knowledge of the access probabilities and comparisons of each value for all cached data. Barbara and Imielinski proposed another strategy [6] in which broadcast data are categorized into synchronous/asynchronous, and stateful/stateless. Jing et al. proposed a method based on bit sequences [7] to effectively send invalidation reports that are organized as a set of bit sequences with an associated set of time stamps.

In recent several years, there are many caching schemes proposed for broadcast data in mobile computing environments. Applications that employ broadcast data delivery mainly are Internet TV programs for mobile users. Chow et al. proposed a distributed group-based cooperative caching scheme [8] by capturing data affinity of individual peers and mobility patterns in a mobile broadcast environment. Ercetin and Tassiulas developed the method for joint cache management and scheduling problem [9] for satellite-terrestrial broadcasting. In their two-stage broadcast data delivery system, a main server broadcasts information to local stations and the local stations act as intermediate stages and transfer information to mobile users. Birk and Tol presented ISCOD approach [10] from a server to multiple caching clients over a broadcast channel. This method is based on high-speed forward broadcast channel and a slow reverse channel. These approaches are not effective for location-aware data dissemination without uplink methods in specific mobile computing environments (e.g., car navigation systems).

In this paper, we propose a cache system to reduce the waiting time specially for location-aware data. With the cache system, a data item is prefetched and replaced at an appropriate timing according to the mobility specification. We numerically evaluate the cache system on the model close to the traffic road environment, and implement the emulation system to evaluate this location-aware data delivery method for a concrete vehicular application that delivers geographic road map data to a car navigation system.

We believe that the methods described above are not always effective for receiving location-aware data on a mobile host and that there could be more effective caching methods for a vehicular application. We propose a cache system especially for caching location-aware data through broadcast data delivery. With this cache system, a data item a mobile host is interested in is prefetched and replaced at an appropriate time according to the mobility specifications; a schedule that a mobile host is expected to travel.

2. BROADCAST DATA MANAGEMENT

2.1. Broadcast scheme

There are two kinds of methods to deliver data to mobile hosts through wireless communication; one is push based and the other is pull based. In pull-based data delivery, a mobile host can explicitly request specific data items to the information center. The limitation of this pull-based data delivery is not scalable; each mobile host allocates its own communication channel to the information center. In push-based data delivery, data are repetitively broadcasted to a mobile host population having no specific request. Mobile hosts monitor the broadcast and retrieve the data items they are interested in when the data items appear on the broadcast channel. Push-based data delivery is suitable in cases in which information is transmitted to a large number of mobile hosts with overlapping interests, because this delivery is scalable and the performance does not depend on the number of mobile hosts listening to the broadcast. However, one of the limitations is that access is only sequential; mobile hosts must wait until the required data items appear on the channel.

Figure 1 shows the outline of data dissemination from a broadcast station to a mobile host. The broadcast data items are location-aware data such as point-of-interest information, traffic information, weather reports, and so on. The broadcast station repeatedly transmits broadcast data items as a data carousel during the scheduled broadcasting

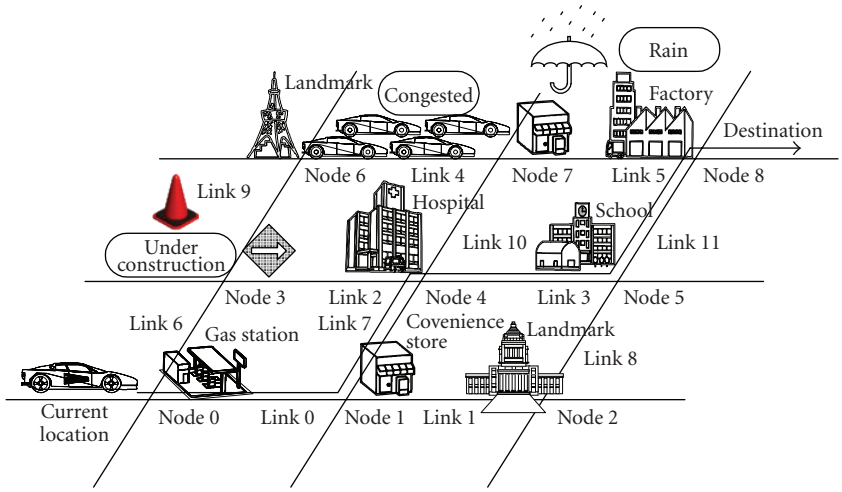


Figure 2: Example of location-aware data.

time period. The caching of data items in a mobile host's local storage is important for improving retrieval performance and for data availability. The mobile host receives and caches data items in its local storage, and the data items become available before the start of a scheduled broadcasting period. Therefore, the mobile host does not have to wait for the data items to appear on the broadcast channel if the data is stored in the cache. Generally, due to the limited size of local storage in the mobile host to cache broadcast data items, the mobile host selects only those data items it needs, and the other data items are not stored in the local storage.

2.2. Digital terrestrial broadcast for mobile host

The digital terrestrial broadcasting system in Japan applies ISDB-T (integrated services digital broadcasting-terrestrial) with OFDM (orthogonal frequency division multiplex), which is the standard for digital terrestrial television broadcasting and digital terrestrial sound broadcasting. OFDM modulation is effective for single frequency networks, and is robust to multipath interference.

The signal in the transmission channel consists of 13 OFDM segments (6 MHz spectrum) whose parameters can be selected independently of each other, for example, one HDTV (12 segments) + mobile service (1 segment), or three SDTV (3 × 4 segments) + mobile service (1 segment). The one-segment digital broadcasting service uses the middle segment of the 13 segments to transmit and enables high error tolerant reception for mobile receivers. One current service of the one-segment broadcasting is digital TV programs transmitted in a H.264 (MPEG-4 AVC) format at QVGA (320 × 240 pixels) resolution. The total bit rate is approximately 312 kbps with DQPSK modulation and 1/2 inner convolution error correction, 416 kbps with DQPSK modulation, and 2/3 inner convolution error correction, 624 kbps with 16 QAM modulation, and 1.4 Mbps with 64 QAM modulation. Since the ISDB-T specification includes a data broadcasting function, we believe that the service would be applicable for delivering location-aware data to vehicular applications.

3. LOCATION-AWARE DATA DELIVERY

3.1. Location-aware data

We refer to location-aware data as information regarding hospitals, gas stations, landmarks, and so forth which shown in Figure 2 are usually managed on mobile hosts. The location-aware data are basically dependent on geographic locations. Therefore, traffic information and weather reports are also included in the category. Moreover, we also define a scope of location dependent data as the available area of the data. For example, the scope of traffic information or weather reports is the area where the information or the report is referred.

Figure 3 shows that the convenience stores at the places B and F have a narrow scope, while the hospital at the place C has a wider scope. Some traffic information generally still have larger scope. Weather reports have an even wider scope for each information. Data are supposed to be available on a mobile host in the case it is located within the scope of the data, otherwise data are not available when it is not within the scope. The idea of the scope is useful to decide the timing when the data is prefetched and replaced in a cache on a mobile host.

3.2. Mobility specification

Mobility specifications are composed of the current location, the destination location, the link (road) list during the time a mobile host moves from the current location to the destination, and the time data when a mobile host passes through nodes and links. We assume a mobile host can measure its current location, the current time, the direction in which it is moving, and its mobility specifications by acquiring inputs from the following functions: a GPS receiver; geographic

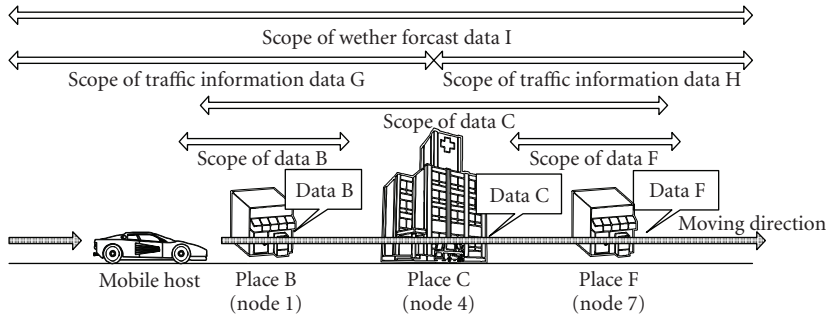


Figure 3: Scope of location-aware data.

road network data stored on a CD/DVD-ROM or hard disk drive; a speed meter; an angular velocity sensor; and a route calculation program that automatically calculates the shortest travel time route from the vehicle's current location to the desired destination.

Mobility specification data is generated using the route calculation program. Users can also individually set a desired route that need not be the route with the shortest travel time. We assume that a mobile host moves according to the mobility specifications previously set by the route calculation program or individual users.

4. CACHING ALGORITHM

4.1. *Caching mechanism*

The basic concept of the scope is useful for deciding the timing of the data to be prefetched and replaced in a mobile host's cache. In the case of geographic road map data, the scope of small-scale (wide area) map data is defined as large and the scope of large-scale (detailed) map data is small.

Suppose that a mobile host tries to make use of data items that are not stored in the cache; mobile hosts need to wait for the data items to appear on the channel. Generally, mobile hosts need to wait an average of half a broadcast period to receive a specific data item. To eliminate waiting time, prefetching on a cache is preferable for mobile hosts. Because of a mobile host's memory size limitations, useless data must be replaced in order to receive new data.

We consider the case that a mobile host moves in the direction of the arrow on the road network shown in Figure 2. There exist the location-aware data on the road network, and the mobile host is supposed to use the data item regarding each place on the road. In this situation, the caching and replacement policy in this paper is explained in the following. We use the simple straight route for the explanation, although the route of the mobile host in Figure 2 is not straight. Figure 4 shows the procedure of caching data items.

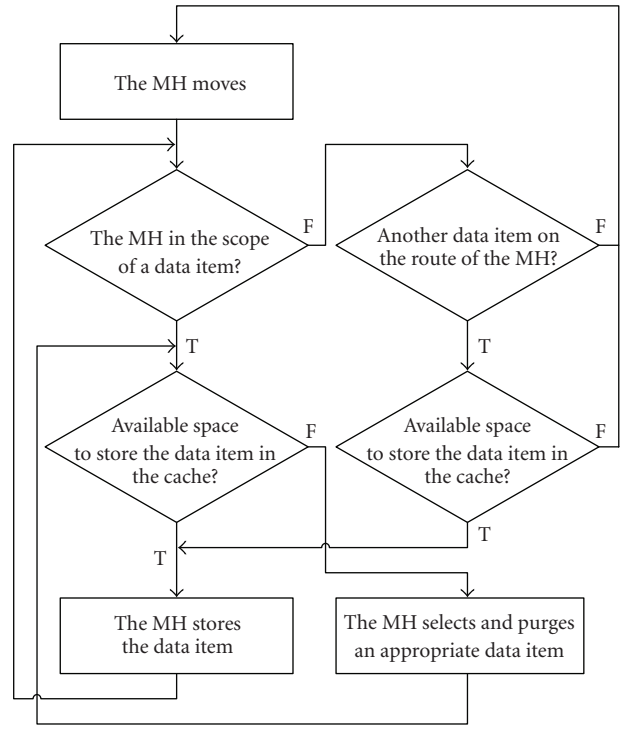


Figure 4: Procedure of caching data items.

4.1.1. *Prefetching policy*

As shown in Figure 5, the mobile host moves in the direction of the arrow and stores the location-aware data A, B, C, and D regarding facilities located at A to D, which are all further along the mobile host's route. In this example, the mobile host is supposedly implemented with a cache for four sizes of data items. In this case, the size of each data item is the same. With the prefetching policy, the mobile host stores the data items in the cache when the mobile host approaches a particular place and that location enters the scope of the data. Moreover, in a case where the cache is not full of data items, the mobile host can prefetch further data items relating to places further along their route, even though the mobile host is not in the scope of the other data. The mobile host does not cache data items that are not located on routes that follow the mobility specification of the mobile host.

When prefetching data items in the cache, mobile hosts do not need to wait until the data items arrive at the mobile hosts. With push-based delivery, the mobile host stores the

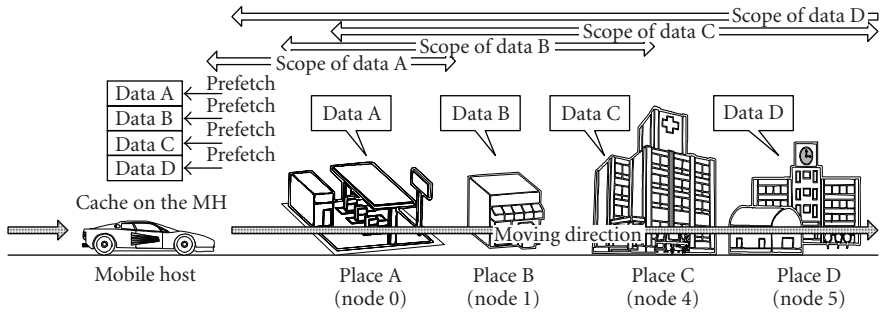


Figure 5: Prefetching of location-aware data.

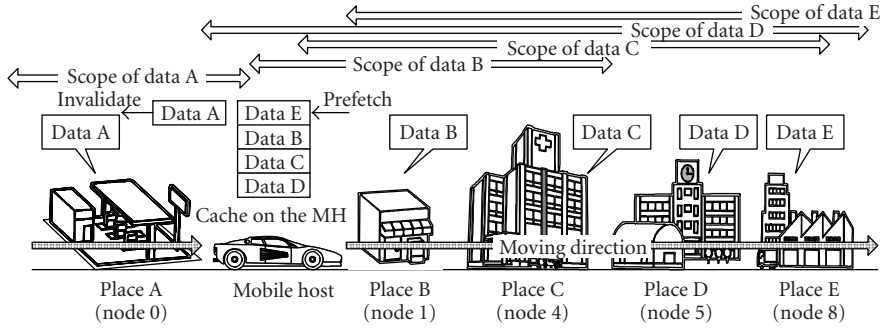


Figure 6: Replacement of location-aware data.

required data items that appear on the channel, before they use the data. With pull-based delivery, the mobile host automatically sends a request to receive data at a certain location where it enters the scope of data items. Besides the facility data shown in this example, this mechanism is also useful for caching geographic road map data.

4.1.2. Replacement policy

The replacement policy is explained in Figure 6. Suppose the mobile host continues to move and pass through place A after the mobile host prefetches data relating to places A to D. When the mobile host approaches place E and that place enters the scope of the data, the mobile host tries to prefetch data E. In this situation, since the cache has no space for data E, one of data items A to D needs to be replaced to store the new data. We assume there is little chance that the driver will make a U-turn and that the mobile host will want to access data A after it passes through place A. Therefore, with the replacement policy, data A is replaced because the mobile host has already passed through it and it exits the scope, when approaching place E.

It is known that the LRU (least recently used) replacement policy, with which the data replaced is the one that has been unused for the longest time, is effective in a general cache system. However, we believe that the LRU policy is not effective for accessing location-aware data. In Figure 5, data A, B, C, and D are stored in the cache in that order because the mobile host approaches the places in the same order. In addition, when the mobile host passes through place A, the mobile host accesses data A. Therefore, if the LRU policy is adopted, data B is supposed to be replaced because data B is not accessed for the longest period among the data A to D, although there is a good possibility that the data B will be accessed next. When the mobile host arrives at the place B, it restores data B using either push-based or pull-based delivery. This situation is very ineffective.

With this policy, the cache data is replaced more effectively compared with the LRU in the case of location-aware data, because the cache system checks the moving direction of the mobile host and the location and scope of each data item.

5. EVALUATION OF THE CACHE SYSTEM

5.1. Evaluation method

To simply evaluate the cache system, we use a mathematical method to measure the total of miss penalty for which each client has to wait until the required data appear on the channel. If the required data is already prefetched and stored in the cache, the penalty time is estimated at zero. Generally prefetching schemes are limited both by prediction accuracy and by the penalty for misprediction. The former depends on a selection method of prefetching items and the size of the cache, and the latter is the time elapsed from the moment

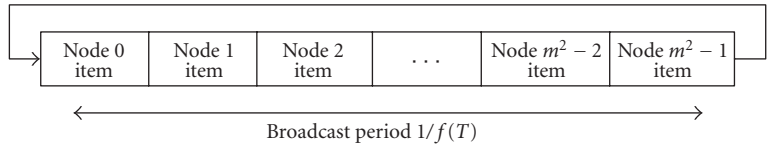


Figure 7: Broadcast method (flat organization).

a client expresses its interest to an item to the appearance of the item on the broadcast channel. Therefore, we adopt caching methods and cache size as evaluation parameters.

5.2. Evaluation model

The road network model shown in Figure 2 is used to evaluate the cache performance of the three kinds of cache model corresponding to the information of a mobile host: (1) random data cache to prefetch data items at random in the cache, (2) neighbor data cache to prefetch data items, which are location-aware data, around the current location of the mobile hosts, and (3) routed data cache to prefetch data items on the route that the mobile host is expected to follow. In the case that the location of a mobile host is unknown, the random data cache is used, and in the case that only the location of a mobile host is known, the neighbor data cache is used. The routed data cache proposed in this research is used when the location and the route of a mobile host are known by mobility specification. The evaluation includes the following conditions.

(1) The road network model is composed of simply meshed m nodes $\times$ m nodes. Each node is located at the same distance; each link is the same length l.
(2) A mobile host enters the road network from the corner of the network; that is the start point. The goal of a mobile host is a cater corner to the start point. Speed of a mobile host, v, is the same from the start point to the destination. A mobile host makes a random choice of one of the shortest routes from the start point to the goal.
(3) A data item related to each node in the road network, that is a location-aware data, is broadcasted with a flat organization shown in Figure 7. The size of each data item S_{data} is the same, and the frequency of the broadcast is f. The average of the period of time that an interesting item appears on the broadcast channel is $1/(2f)$. The client interests the data items on the route which the mobile host follows.
(4) A client prefetches data items on its cache. The size of the cache is S_{cache}. Since the cache replacement policy is described before, we assume the replacement policy of cached data is optimal in this evaluation.

5.3. Evaluation result

We evaluate the average of miss penalty for each method, P_{random}, P_{neighbor}, and P_{routed} for the three cache policies: (1) random data cache, (2) neighbor data cache, and (3) routed data cache. The average of miss penalty when the data is not in the cache is $1/(2f)$. The number of nodes when a mobile host passes from the current location to the destination is $2m - 2$, where the client takes each location-aware data. The speed of a mobile host, v, is shown by nfl, where n is the number of links through which a mobile host passes for a unit period of time.

The average of total miss penalty for each cache policy is the product of the probability to miss the cache for each method, the average of miss penalty $(1/2f)$, and the number of nodes that a mobile host pass through $(2m - n)$,

$$
\begin{aligned}
P_{\text{random}} &= \frac{(m^2 - s)^+}{m^2} \cdot \frac{1}{2f} \cdot (2m - 2), \\
P_{\text{neighbor}} &= \frac{\left((1 + 4\sum_{i=1}^{n} i) - s\right)^+}{1 + 4\sum_{i=1}^{n} i} \cdot \frac{1}{2f} \cdot (2m - 2), \\
P_{\text{routed}} &= \frac{(n - s)^+}{n} \cdot \frac{1}{2f} \cdot (2m - 2),
\end{aligned}
\tag{1}
$$

where a^+ is $\max(a, 0)$, s is $S_{\text{cache}}/S_{\text{data}}$, and n is flv; the larger n is, the faster the mobile host moves. The case of the neighbor data cache is approximate value under the condition which the value of m is much large compared with the value of n. The evaluation result for the three data caching policies is shown in Figures 8, 9, and 10, in the condition that $m = 50$, $f = 0.5$, and $n = 1$, $n = 3$ and $n = 5$, respectively. The routed data cache we propose has much smaller penalty with a small size of cache in comparison with the random data cache and the neighbor data cache.

5.4. Emulation system

To study location-aware data delivery method using data broadcasting, we implemented the emulation model shown in Figure 11. The emulation model consists of two kinds of components: a broadcast data server as a broadcast station and a broadcast data receiver on a mobile host. Both the server and receiver functions are implemented as application programs on PCs. These PCs are connected to a single IP network over Ethernet or WiFi radio channel. Since there could be multiple mobile hosts receiving broadcast data within the broadcast area of a broadcast station, the multiple broadcast data receivers connected to the network can receive broadcast data from the broadcast data server at the same time.

In this research, we set up our target vehicular application as an off-board car navigation system receiving geographic road map data through a wireless broadcast channel.

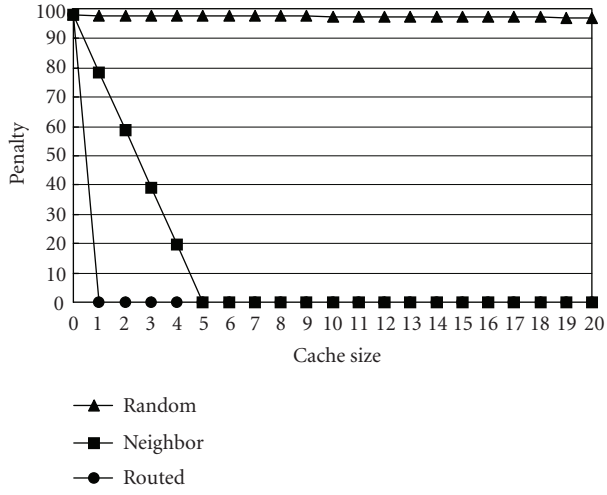


Figure 8: Penalty of cache method ($n = 1$).

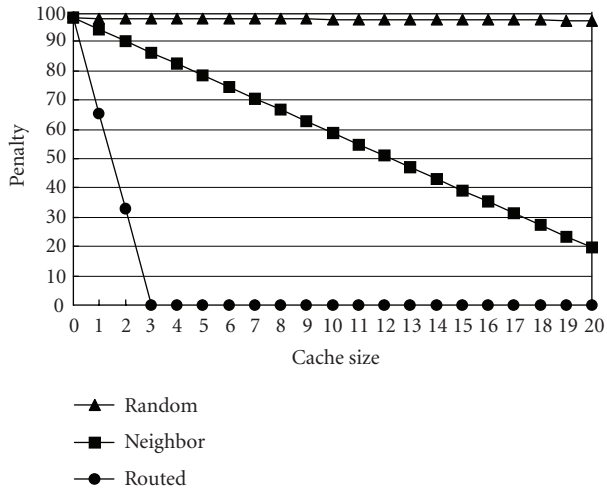


Figure 9: Penalty of cache method ($n = 3$).

The broadcast data server broadcasts map data items and the broadcast data receiver in the mobile host receives some of the map data items that the mobile host requires. The criteria by which the mobile host selects the data items are mobile host's mobility specifications, which we will describe later.

5.4.1. Broadcast data server

The broadcast data server contains location-aware data in the broadcast data storage. The broadcast transmitter reads data items from the data storage, packetizes the information to broadcast the data items, and broadcasts data to the network that adopts a broadcast program. When broadcasting information, the broadcast transmitter activates functions regarding the packet size, the broadcast period, and the selection of data items for the broadcast program. The broadcast transmitter transmits data items without any request from any broadcast data receiver.

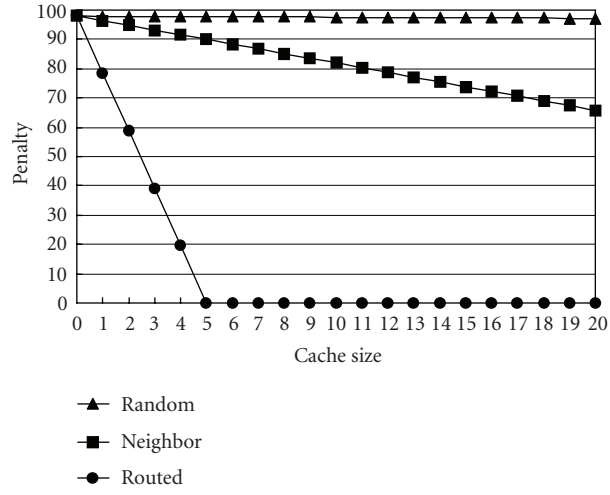


Figure 10: Penalty of cache method ($n = 5$).

Data delivery through broadcasting is implemented as a datagram broadcast with UDP (user datagram protocol) as a transport layer protocol over an IP (Internet protocol) network. UDP provides no guarantees to the broadcast transmitter application for data item delivery and the broadcast transmitter retains no state on the UDP datagrams once sent.

When broadcasting geographic road map data is tiled with $m \times m$ mesh boundaries, as shown in Figure 12, a broadcast data receiver can access specific items from the broadcast data; in this case, the map data items that relate to the current location of the mobile host. The access time is the average time that elapses from the moment a mobile host requires certain data items to the receipt of these items on the broadcast channel. The broadcast data should be organized so that access time is minimized. Under a general broadcasting mechanism, it is impossible to take into account all broadcast data items required by all mobile hosts.

In this research, we adopt the simplest way to organize the transmission of broadcast data, which we call flat organization. There is no priority of any of the square-meshed map data items. The broadcast program is arranged as a flat data carousel as shown in Figure 7. The square-meshed map data items are broadcast in the following order: mesh 0, mesh 1, mesh 2, ..., and mesh $m^2 - 1$. The mesh 0 item is broadcast again after the mesh $m^2 - 1$ is sent.

5.4.2. Broadcast data receiver

The broadcast data receiver described in Figure 11 in a mobile host receives the data items broadcast by the broadcast data server. It is possible that there are multiple mobile hosts within a certain broadcast area, and each mobile host receives exactly the same data from the broadcast station. The broadcast tuner in the broadcast receiver depacketizes the received data and checks for errors. The storage manager selects some of the received data items according to requests from the data

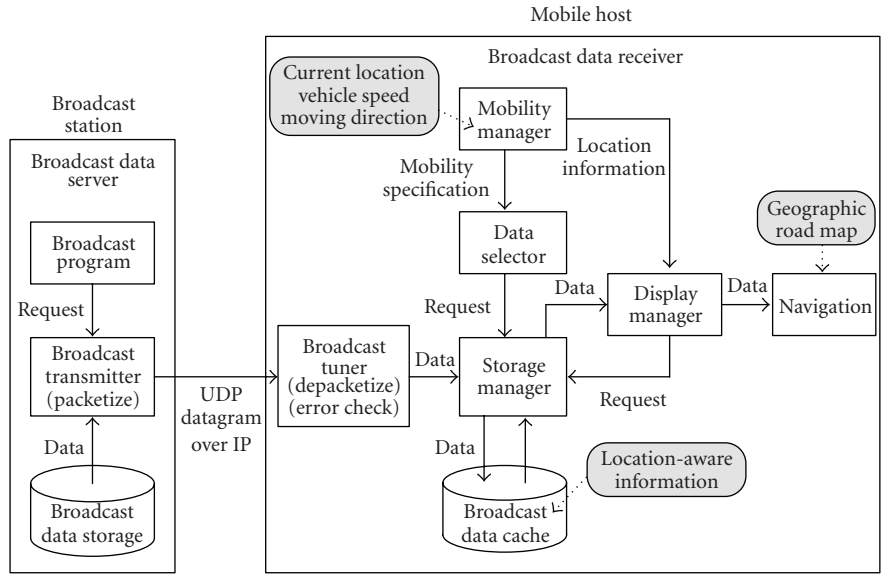


Figure 11: Emulation model for data transfer.

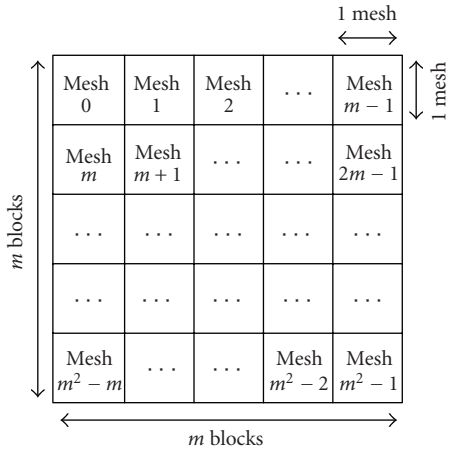


Figure 12: Map data tiles with $m \times m$ mesh boundaries.

selector, and stores the data items into the broadcast data cache. The location function module manages the current position, the moving direction, and the mobility specifications. The data selector receives this information and sends requests to the storage manager about which broadcast data items need to be kept in the cache, using the caching algorithm described in the next subsection. The broadcast data items that a mobile host does not need are not stored in the cache; however, another mobile host may require these data items.

The storage manager provides the display manager with data items from the broadcast data cache. The display manager has information about which data items are required at a certain time, and sends these data items to the display; geographic road map images then appear on the display. Unnecessary data items kept in the broadcast data are purged by the storage manager in response to requests from the data selector that relates to mobility specifications.

5.5. Broadcast data selection

Selection of the map data items depends on the mobile host's mobility specification (current location, moving speed, moving direction, and so on). For simplicity, we adopt a simple mobility specification to obtain square-meshed geographic map data. As the mobile host moves, the selected map data items change. Suppose the mobile host has already stored map data items around its current location, and the mobile host then requires map data items for the location it is currently moving towards. The mobile host needs to predict its future location using mobility specification, and select and store the map data items around the new location from the broadcast data before the mobile host arrives at the location. In addition, the map data items for the backward area of the mobile host will be purged from the cache.

In our system, the faster a mobile host moves, the larger the map data area is stored in the cache, to diminish the risk of no data items being stored around the new current location. Examples of map data item selection are illustrated in Figures 13 and 14. Figure 13 shows the cache area in a case where the vehicle speed is $v = l/T$, the mesh size of the map data is $l \times l$, and broadcast period is T.

Suppose a location coordinate of the current mobile host's position is (0,0), the data items $(-1,1)$, $(0,1)$, $(1,1)$, $(-1,0)$, $(0,0)$, $(1,0)$, $(-1,-1)$, $(0,-1)$, and $(1,-1)$ are stored while the mobile host moves at the vehicle speed $v = l/T$. When the mobile host moves from (0,0) to (0,1) according to the mobile specification, the data items $(-1,-1)$, $(0,-1)$, and $(1,-1)$ would not be necessary, while $(-1,2)$, $(0,2)$, and $(1,2)$

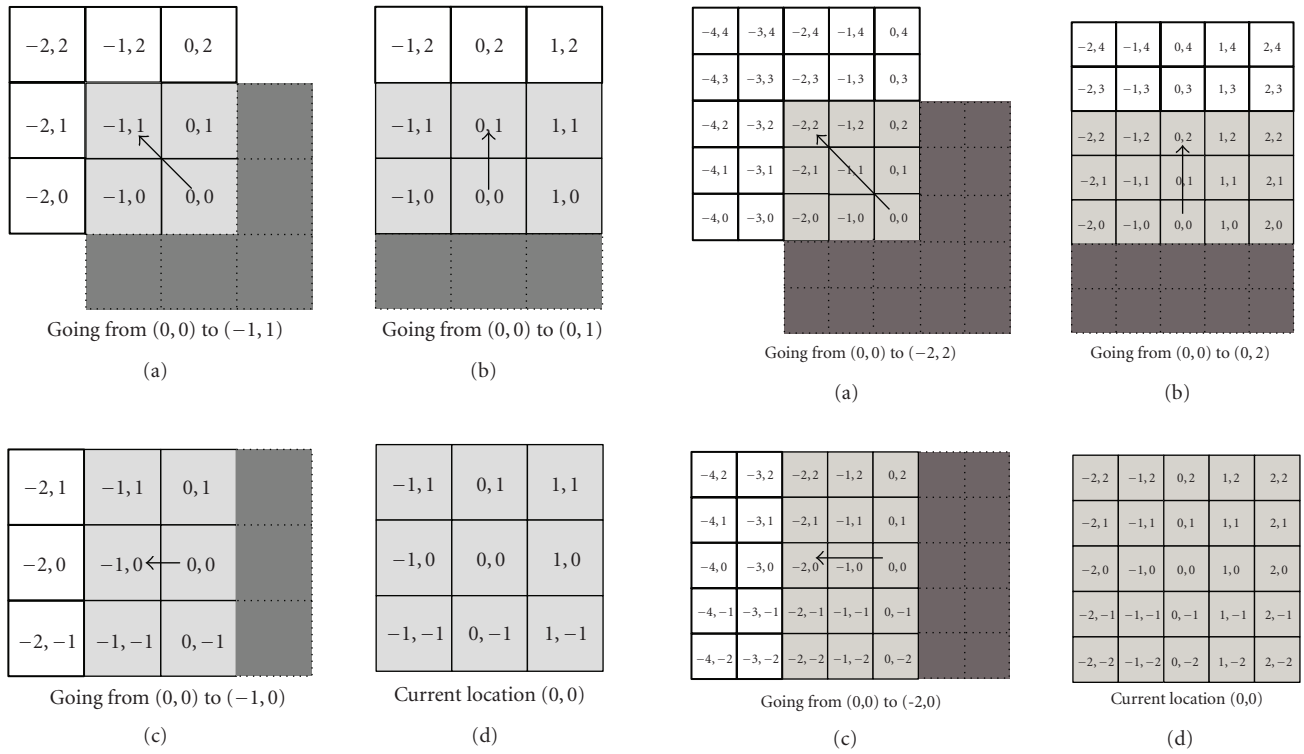


FIGURE 13: Data cache area ($v \simeq l/T$).

need to be stored. When the mobile host moves to (−1,1) according to the mobile specification, the data items (1,1), (1,0), and (−1,−1), (0,−1), and (1,−1) are not necessary, while (−2,2), (−1,2), and (0,2), (−2,1), (−2,0) need to be stored. In this case, the vehicle speed should be $v = \sqrt{2}l/T$. Figure 14 shows the cache area of the map data when the vehicle speed is $v = 2l/T$.

6. DATA BROADCAST EXPERIMENT

6.1. Broadcast map data

We performed a data broadcast experiment with our implemented system to emulate a location-aware data delivery method for a vehicular application using data broadcast. A broadcast data server PC used as broadcast station, and multiple broadcast data receiver PCs used as mobile hosts are connected to the IP network described in Section 5.4. The parameters of data broadcast under this experiment are shown in Table 1.

In a case where the packet size is 2 kBytes and the data transmit interval is 100 milliseconds, the valid bit rate for data transmission becomes 160 kbps (approximate half as effective as of 312 kbps). Because the IP network bit rate (more than a megabit per second) is much faster than the one-segment digital terrestrial broadcasting bit rate (about hundred kilobits per second), a certain transmit interval is set between the broadcast data. The mesh size of the map data used in this experiment is level 2, which we call the L2 mesh; a single L2 mesh of the map data covers (2 km × 2 km). When the

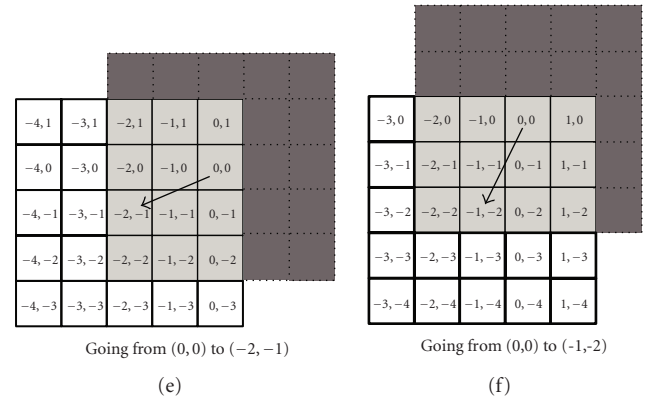


FIGURE 14: Data cache area ($v \simeq 2l/T$).

map area is a 49 (7 × 7) meshed map, the broadcast period becomes approximately 100 seconds, and the total broadcast area is 14 km × 14 km.

6.2. Map display

Figure 15 shows an example of an emulation model display. The meshed map data items are broadcast from the broadcast data server to multiple mobile hosts, and the mobile host receives the map data items it needs. To achieve a location-aware data delivery method for a vehicular application with digital broadcasting, we implemented the map display function of a car navigation system and a mobile host's location emulation program on a PC. The current location of the mobile host is shown as a center small circle on the map, which

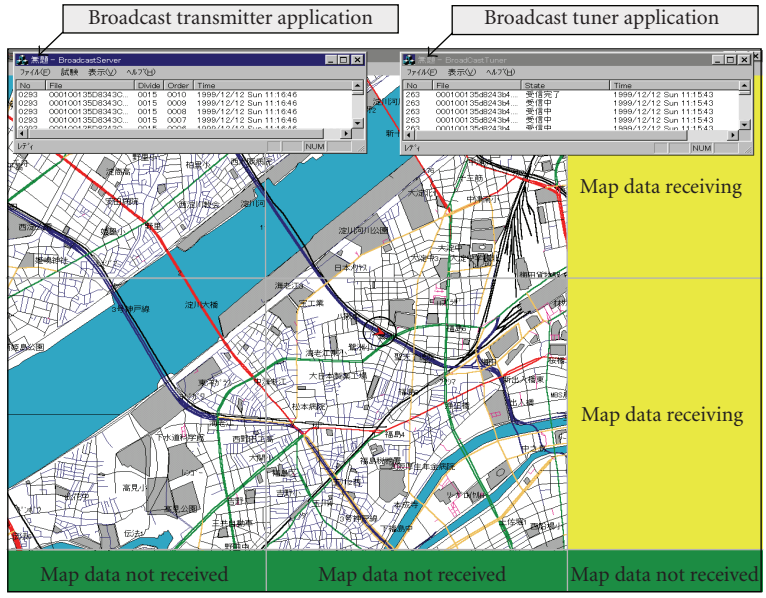


Figure 15: Simulation for data broadcast.

Table 1: Emulation parameters

Emulation item	Parameter
Packet size	2 kbytes
Data transmit interval	100 milliseconds
Valid bit rate	160 kbps
Mesh size of the map	2 km mesh (2 km × 2 km area)
Broadcast area	14 km × 14 km (49 meshes)
Data size per mesh	25 kbyte–50 kbyte
Broadcast cycle	Approx. 100 seconds
Cache size	9, 25, 49 meshes

is a vehicle locator mark. As the mark moves along the prerecorded route stored as a PC data, the broadcast data receiver application receives meshed map data relating to the direction in which the vehicle is moving. The data items for the areas that are shown as dark colored parts of the meshed data inside the display (at the bottom of the figure) have not yet been received, and the data items for the areas that are shown as more brightly colored parts (right of the figure) are in the process of being received, which means that these parts of the map appear shortly.

6.3. Evaluation

When average vehicle speed of the mobile host was 70 km/h ($v \simeq l/T$), the packet size was 2 kBytes, and the data transmit interval was 100 milliseconds, we confirmed there was no delay in displaying the appropriate map data following the mobile host's movement. We also confirmed there was no delay when the average vehicle speed of the mobile host was 140 km/h ($v = 2l/T$) where the size of the cache is 1.25 MBytes (50 kBytes × 25 meshes).

Using numerical evaluation without considering mobility specification, the required broadcast bit rate would be approximately 16.7 Mbps, if there was no cache in the mobile host, where broadcast area of a certain digital terrestrial broadcast station is 100 km × 100 km, and maximum vehicle speed is 120 km/h. Because the real available broadcast bit rate is 160 kbps, the required cache size is 5 MBytes. The required cache size depends on the broadcast area and the maximum supported vehicle speed.

7. CONCLUSION AND FUTURE WORK

These systems equipped in a vehicle can provide drivers with point-of-interest information, traffic information, weather reports and so on as well as driving directions through relatively slow and expensive cellular phone networks. Meanwhile high performance digital broadcasting for mobile hosts has been available recently.

In order to deliver location-aware data to a vehicle through broadcast channels, we proposed a new cache system to effectively prefetch and replace cached data using mobility specifications, which is a schedule according to the direction in which a mobile host moves. In this paper, we implemented an emulation system to evaluate our location-aware data delivery method using a cache system for a concrete vehicular application, which delivers geographic road map data to a car navigation system. Through our experiments, we confirmed that the method worked.

In this research we adopted an Ethernet and a WiFi radio channel as the IP network, which both have very small error rates. However, the error rate with a real digital terrestrial broadcast must be bigger than in the network we used

for our experiment. Conducting an evaluation that includes error rates over the broadcast channel will be the subject of our future work.

REFERENCES

[1] ARIB STD-B31 Version 1.5, "Transmission System for Digital Terrestrial Television Broadcasting," Association of Radio Industries, and Businesses, 2003.

[2] ARIB STD-B38 Version 1.3, "Coding, Transmission and Storage Specification for Broadcasting System Based on Home Servers," Association of Radio Industries, and Businesses, 2006.

[3] K. Matsumura, K. Usui, K. Kai, and K. Ishikawa, "Location-aware data broadcasting: an application for digital mobile broadcasting in Japan," in *Proceedings of the 11th ACM International Conference on Multimedia (MM '03)*, pp. 271–274, Berkeley, Calif, USA, November 2003.

[4] E. Pitoura and G. Samaras, *Data Management for Mobile Computing*, Kluwer Academic Publishers, Norwell, Mass, USA, 1998.

[5] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks: data management for asymmetric communication environments," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 199–210, San Jose, Calif, USA, 1995.

[6] D. Barbara and T. Imielinski, "Sleepers and workaholics: caching strategies in mobile environments," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1–12, Minneapolis, Minn, USA, May 1994.

[7] J. Jing, O. Bukhres, A. Elmagarmid, and R. Alonso, "Bit-sequences: a new cache invalidation method in mobile environments," Tech. Rep. CSD-TR-94-074, Department of Computer Science, Purdue University, West Lafayette, Ind, USA, 1995.

[8] C.-Y. Chow, H. V. Leong, and A. T. S. Chan, "Distributed group-based cooperative caching in a mobile broadcast environment," in *Proceedings of the 6th International Conference on Mobile Data Management (MDM '05)*, pp. 97–106, Ayia Napa, Cyprus, May 2005.

[9] O. Ercetin and L. Tassiulas, "Push-based information delivery in two stage satellite-terrestrial wireless systems," *IEEE Transactions on Computers*, vol. 50, no. 5, pp. 506–518, 2001.

[10] Y. Birk and T. Kol, "Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2825–2830, 2006.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 79095, 8 pages
doi:10.1155/2007/79095

Research Article

Embedded Localization and Communication System Designed for Intelligent Guided Transports

Yassin ElHillali,[1] Atika Rivenq,[1] Charles Tatkeu,[2] J. M. Rouvaen,[1] and J. P. Ghys[2]

[1] *Departement Opto-Acousto-Electronique (DOAE), Institute des Etalons de Mesure Nationaux IEMN, Université de Valenciennes et du Hainaut Cambresis (UVHC), Le Mont Houy, 59313 Valenciennes Cedex 9, France*
[2] *Institut National de Recherche sur les Transports et leur Sécurité (INRETS), 20 rue Elisée Reclus, 59650 Villeneuve déAscq Cedex, France*

Received 14 October 2006; Accepted 16 February 2007

Recommended by Samir Bouaziz

Nowadays, many embedded sensors allowing localization and communication are being developed to improve reliability, security and define new exploitation modes in intelligent guided transports. This paper presents the architecture of a new system allowing multiuser access and combining the two main functionalities: localization and high data flow communication. This system is based on cooperative coded radar using a transponder inside targets (trains, metro, etc). The sensor uses an adapted digital correlation receiver in order to detect the position, compute the distance towards the preceding vehicle, and get its status and identification. To allow multiuser access and to combine the two main functionalities, an original multiplexing method inspired from direct sequence-code division multiple access (DS-CDMA) technique and called sequential spreading spectrum technique (SSS2) is introduced. This study is focused on presenting the implementation of the computing unit according to limited resources in embedded applications. Finally, the measurement results for railway environment will be presented.

1. INTRODUCTION

Localization and communication systems become increasingly more important to ensure the common transport safety that is maritime [1, 2], airway, or terrestrial. Actually all boats and the planes are already equipped with systems based on a transponder which allows the localization and data exchange. For example, in the maritime transport domain, a system called automatic identification system (AIS) is deployed. This system equips all chips with a device using a GPS receiver to estimate the boat position and a VHF transponder to broadcast this position and other information to all chips around. However, in guided transport domain, no system is actually able to ensure these functionalities.

In the present paper, a new system, called Communication, Detection and Identification of Broken-Down Trains (CODIBDT), is proposed to optimize the exploitation mode inside automatic guided transports. Indeed, a traffic perturbation occurs when a train is broken down along the line. It is then necessary to accost [3], in safety conditions, this broken train by another train. The line is divided in parts called districts of about 1 km. When a train is in a district, it is declared to be engaged. No coach can go in until the train leaves it. This is the security system in the current networks. If the real time distance between the trains was known, the accosting phase duration between the two vehicles could be reduced significantly. This distance could be transmitted to the exploitation center, which is in charge of procedure management. This measurement should be provided in different environments where the train moves like free area, viaduct, and subway tunnel.

However, in a subway tunnel, due to the multipath reflections, a conventional radar system analyzing signal echoes on an obstacle is inefficient. In fact, as shown in Figure 1, the radar receives multiple echoes especially if an obstacle is closer than the train targeted. In such case, it is difficult to detect the right obstacle among all these echoes.

The designed cooperative radar CODIBDT overcomes these problems and its principle relies on a transponder system: transmitters and receivers equip, respectively, the front and the rear of each train. Another advantage is that it not only provides a real time distance measurement, but also allows communication with high data flow between the sensors. Then it could be helpful to develop many applications among which exchange information such as audio-video records in order, for example, to increase security feeling and

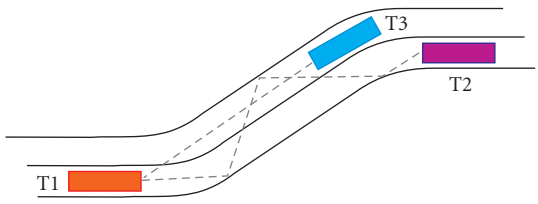


Figure 1: The problems occurred in a subway tunnel.

quality of service inside trains (wireless Internet). For this purpose, an appropriate multiplexing method for this sensor has been proposed to favor high data flow and robustness according to signal-to-noise ratio (SNR) criterion.

This paper is focused on developing hardware and software implementation of this system developed using flexible components such as FPGA. Finally, the results obtained with the implemented mock-up are presented in free space area and in tunnel.

2. THE PRINCIPLE OF THE PROPOSED CODIBDT SYSTEM

The implemented system has a broadband of about 100 MHz that can be used. We propose to develop a new coding algorithm to exploit this band in order to establish high data rate communications between trains and operator centers. The CODIBDT sensor is able

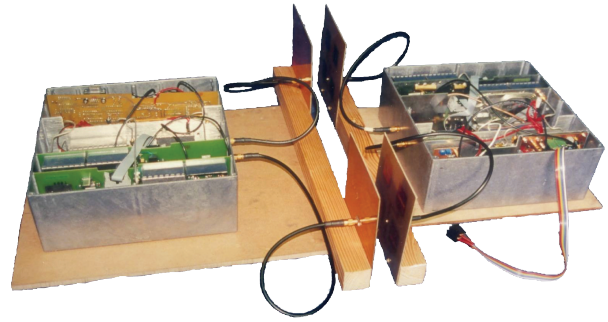
(i) to detect the position, get the identification and the status of the train,

(ii) to compute, in real time, the distance towards the preceding vehicle,

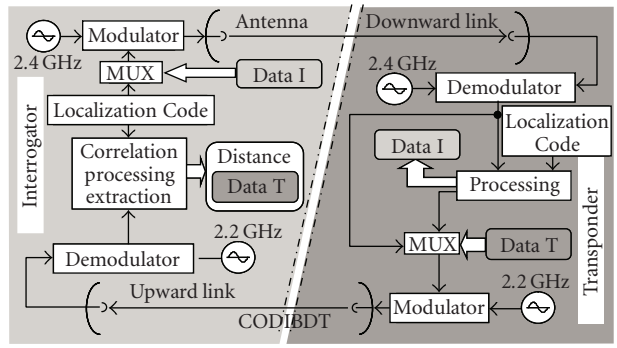
(iii) to allow high data rate communications for exchanging data information between trains.

Its principle relies on a transponder system using an interrogator/responder pair (see Figure 2(a)) which equips, respectively, the front and the rear of vehicle. As shown on Figure 2(b), the first vehicle (interrogator) sends a signal at a frequency of 2.2 GHz, towards the preceding vehicle (responder). This signal, which has its own radar code, is a binary pseudo random sequence (BPRS). It is received by the second vehicle ahead. The sensor of this vehicle ahead process and sends a replica of the received signal that is amplified, filtered and filled out with data at the same time. These data contain information about its identification (or identity), its working mode or state (broken-down or not, failure status), and so forth. The new signal sent at 2.4 GHz frequency is received by the interrogator that is able to deduce the intertrain distance and to recover the data sent by the responder (identification, status (broken-down or not)).

The frequency choice is an important item, because it depends on the line configuration and the possibility of resolving both effects of masking and multipath, which strongly affect the resulting signal. The present choice is settled in the



(a) The CODIBDT radar mock-up



(b) The CODIBDT transmitter/receiver design architecture

Figure 2

range of 1–10 GHz band. For low power transmitter consumption, we choose industrial, scientific and medical (ISM) band for our sensor on (2.2 GHz and 2.4 GHz).

Such a cooperative radar system for which the target becomes active like in a transponder, the proposed system has great advantages among others.

(i) It works in each kind of environment: free space, subway tunnel or viaducts areas. In the later case, conventional radar systems based on distance measurement using signal echoes on obstacles proves inefficient.

(ii) Moreover, the pseudorandom sequence (BPRS) used, combined with a correlation receiver, are very adapted to the detection of signals over noisy communication channels and can be generated easily.

On the following paragraphs, this paper will present characteristics and performances in terms of BER and data rate of the system.

3. PRESENTATION OF THE MULTIPLEXING TECHNIQUE

This paragraph is focused on technical solutions to develop the new communication feature and optimize the combination of the two main functionalities: localization and high data rate communication. In order to provide this combination with high speed data flow, different coding methods [4] were tested and one of them is presented hereafter. Indeed,

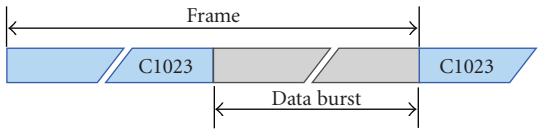


Figure 3: General structure of a frame sent with the coding technique.

Figure 4: Detailed structure of the frame sent by the SSS2 technique.

Table 1: Number of code according to register length.

Register length	3	4	5	6	7	8	9	10
Number of different orthogonal code	2	2	6	6	18	16	48	60

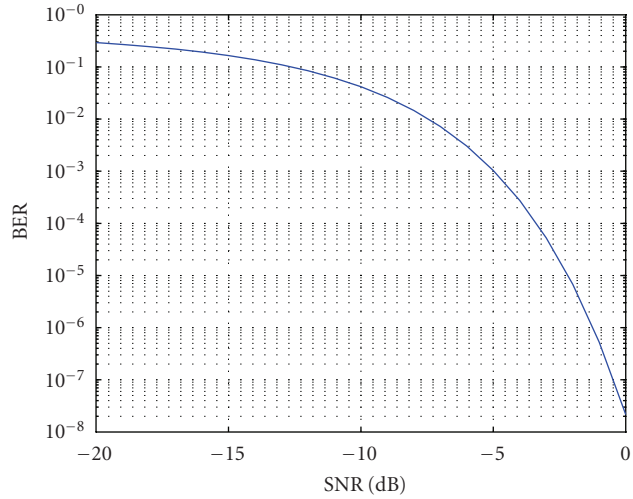


Figure 5: The BER obtained with SSS2 technique.

this method allows a continuous refreshing of the measurement of distance and also ensures a sufficient flow rate for communication with a suitable BER.

The technique is inspired from the DS-CDMA [5] and uses families of orthogonal codes (Binary Pseudo-Random Sequence—BPRS) with two different lengths. The first one has code length of 1023 bits (C1023) intended for the localization and the second is constituted by short codes of 31 bits long (C31) dedicated to the communication.

Different codes families (BPRS codes, Gold codes, Kasami codes) were studied for use in this system and were compared according to the number, the length, and the maximum of their crosscorrelation. These sequences look like a noise and so have a spread spectrum. The selected codes have the same length: 2^{n-1} [6, 7], an autocorrelation peak and a low level only for the crosscorrelation. The BPRS, also called m-sequences, presents an autocorrelation with a peak at 2^{n-1} and $a-1$ level elsewhere. They have good performances even when the signal to noise ratio is very low. Their implementation is simple. They could be easily generated using shift registers with XOR feedback. The number of these codes per family is a function of their length is presented in Table 1.

These families are considered as the reference in this study.

As we can see on Figure 3, the method consists of sending periodically the code of localization to ensure a regular renewal of the distance measurement. We propose to insert between two codes of localization a variable structure of coded data burst. Between two localization codes we insert 1023 bits, which can be divided into several short codes.

The proposed coding technique is entitled SSS2 for Sequential Spectrum Spreading using 2 codes.

The spreading with the C1023 is used to assume localization function. The second one is used to code data communications with the C31 in the classical DS-CDMA (Direct Sequence CDMA) [5, 7–9]. This technique allows us to send 33 bits of data between two codes of localization. The length of the first code is chosen to reach the required distance (about one kilometer) and due to important number of reply codes (60). The length of the second code affects the rate of communication, if we choose a shorter one, we will have a higher rate but the robustness will decrease significantly. Multiple simulations have been done and the length of 31 bits seems to be a good trade-off between the data rate and the robustness to noise. Figure 4 shows the standard structure of the frame transmitted by this method.

To calculate the distance, the correlation between the received signal and the reference codes (C1023) is computed. The correlation peak allows the synchronization process. Then, to recover data, a second correlation between the received signal and the C31 code is used.

4. PERFORMANCES

The SSS2 technique has been simulated in additive white Gaussian noise (AWGN) channel in order to evaluate its performances in terms of data flow rate and bit-error rate (BER).

On Figure 5, the bit-error rate corresponding to several signal-to-noise ratio values, obtained by simulations (with sufficient number of iterations) is given for this technique. The SNR is defined as

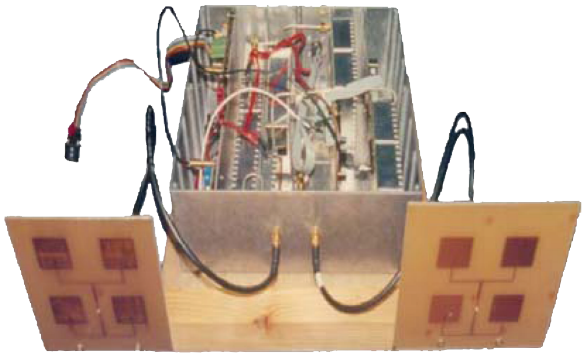
$$\mathrm{SNR} = 10\log\left(\frac{E}{\sigma^2}\right), \tag{1}$$

where E is the maximum power transmitted by the radar and σ is the standard deviation of noise.

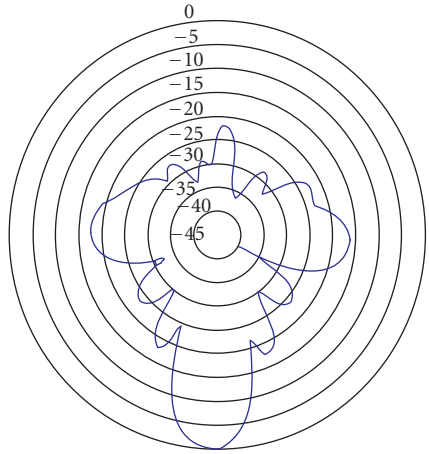
Simulation results show that, in AWGN channel, SSS2 technique is robust to noisy environments (i.e., SNR less than -2 dB). Moreover, a BER of 10^{-5} can be reached with SNR equals to -2 dB with this method.

Concerning the data flow rate, it could be estimated as the following:

$$\text{data flow} = \frac{\text{number of bits sent}}{\text{time}} = \frac{N}{2 * L_c/f}, \tag{2}$$



(a) The patch antenna used in our system



(b) The antenna radiation pattern

Figure 6

where N is the number of data bits sent periodically, $\mathbf{L}_c$ is the length of the localization code and f is the signal frequency.

Furthermore to ensure periodical renewal of the distance measurement, we choose to limit the data frame length to 1023 (as the localization code). And because we spread the data with a code length 31, the maximum numbers of bits which could be sent is limited to 33 bits/frame,

$$\text{data flow} \approx 1.6\,\text{Mbps}. \tag{3}$$

In this case, the data flow which could be reached is about 1.6 Mbps for a clock of 100 MHz. This data flow rate associated to the robustness of this technique in noisy environments (BER of 10^{-5} with SNR greater than -2 dB) makes this multiplexing method very interesting for our application.

Concerning the localization characteristics, it gives a resolution in distance, which is between 1.5 meter and 3 meters depending of the clock frequency used (50 MHz or 100 MHz). The maximal range obtained is about 800 meters in tunnels and 700 meters in free space.

Moreover, the radar detection is physically limited in low range, under 10 or 15 m, due to the recovery time of the sensor.

The actual laboratory mock-up integrates a multiplexing SSS2 technique using flexible components like FPGA [6] as described in Figure 6(a). We use 2×2 patches antennas for each link with a beam aperture of 20° to operate in curves. Figure 6(b) show the radiation pattern of each antenna.

Table 2 gives a summary of performances of the whole radar sensor.

The resolution and range in free space and tunnel are the same of about 1.5 meters for a clock frequency of 100 MHz, and we can reach 700 meters maximum range in free space and 800 meters in tunnels. The range of ours system in tunnel is greater that in free space because the behavior of the tunnel is like a "wave guide" for the frequencies used by ours system.

The preliminary results of simulations confirm the performance of the SSS2 technique (weak BER and sufficient high-speed information exchange).

Table 2: Performances of CODIBDT.

Coding	SSS2
Maximum flow	1.6 Mbps
SNR for BER = 10^{-5}	−2 dB
Range in free space	700 m
Range in subway	800 m
Resolution at 100 MHz	1.5 m
Sensor characteristics	
Antenna aperture	15°
Antenna type	2×2 patches
Antenna size (cm)	12×12

5. CODIBDT IMPLEMENTATION

5.1. Architecture choices

In order to estimate the C1023 flight time between the interrogator and the responder, a local peak is detected in the calculated cross-correlation between the received signal and the reference (C1023). To compute this correlation, the first solution is to use a conventional DSP processor. So, we have to estimate the number of operations needed per second. Indeed, the maximum frequency of the transmitted signal is about 50 MHz (or 100 MHz) and the received signal has to be sampled at least twice per chip. So, the signal to be processed has a given rythm of about 100 MHz (or 200 MHz) and for each chip, at least 1023 MAC (Multiplication and accumulation) are needed to calculate the intercorrelation. Due to the fact that DSP processors carry out a MAC operation by clock edge, a processor which runs up to 102.3 GHz or (204.6 GHz) is required. However, such a processor does not exist on the market yet. For these reasons, we mother choose new generation components such as FPGA which propose a more flexible and easily reconfigurable structure and where treatments may be massively parallelized.

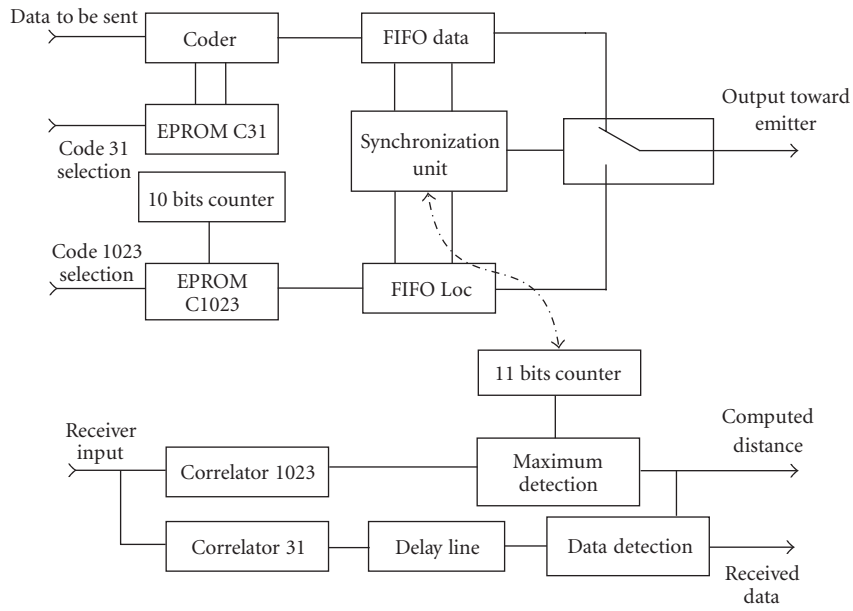


Figure 7: Different modules implemented in the FPGA component of the interrogator.

So the computing unit needed for calculating the correlation as well as the detection unit will be implemented on FPGA components. The correlation unit is composed by a barrel of parallel multipliers and accumulators. Thus, the system can run as fast as the frequency of the received signal (i.e., in real time). Moreover the detection unit is programmed such a "state machine." In our design the biggest element, which consumes the largest resources of the FPGA, is the correlator module. Multiple architectures to implement this module is developed to optimize the resources consumption according to limitation imposed by the specification or the embedded applications.

5.2. Global implementation of the CODIBDT process

As shown on the previous paragraphs, the proposed system is made of a couple of microwave transmitting and receiving equipments fixed on each train (resp., interrogator and responder). The transmitting equipment includes a modulator and a demodulator, respectively, at 2.2 GHz and 2.4 GHz frequencies and includes also a computing unit composed by an ADC—analogue-to-digital converter—and FPGA device. The receiving equipment is similar but the modulator will run at 2.4 GHz and the demodulator at 2.2 GHz. The localization-communication procedure will be made in several successive steps, which can be summarized as follows.

The interrogator will build the global frame and send it towards the responder at 2.2 GHz.

The responder demodulates the signal at 2.2 GHz and identifies the localization frame, then it replaces the interrogator data frame by his data frame.

The new global frame will be sent to the interrogator at 2.4 GHz.

Besides the interrogator, the computing unit will calculate the correlation between the received signal and the different code (C1023 and C31) in order to estimate the fly time and decode the data frame.

The working of the computing unit will now be described.

5.3. The interrogator computing unit

As shown in Figure 7, the interrogator computing unit can be divided into two principal blocks: the transmitting block (at the top of the figure), and the receiving block (at the bottom). It has different inputs and outputs such as

(i) data input,
(ii) C31 and C1023 code selection,
(iii) received signal which is plugged into the ADC output,
(iv) signal output,
(v) estimated distance and received data output.

It contains different modules as the following.

(i) EPROM's where the two different BPRS codes used are stored.
(ii) Coder module: to spread the data with data code.
(iii) Data FIFO where spreaded data will be stored.
(iv) FIFO Loc where localization code will be copied.
(v) Synchronization unit which builds the global frame by synchronizing the read operation for the two FIFOs.
(vi) Some counters: 10 bits counter to transfer the localization code from EPROM to FIFO loc, and 11 bits counter used as a time references (reference counter).
(vii) Two correlators.

(viii) Peak detection to detect the peak present in the correlation result between the received signal and the localization code.
(ix) Data detection.

The communication localization process will start in the interrogator FPGA by constructing the burst to be sent. The coder component will modulates the C31 code stored in the EPROM and put it in "FIFO Data" and the 10 bits counter transfer the C1023 stored in the EPROM into the "FIFO Loc."

When the reference counter is reset to zero, the synchronization unit deals with orchestrating the sending of the localization code present in "FIFO LOC;" followed by 33×C31 codes modulated by the data present in the "FIFO data."

This signal will be received by the responder and will be amplified, modified and sent back towards the interrogator.

Besides the interrogator the module "correlator 1023" calculates the intercorrelation between the received signal and reference code C1023 and in the same time the "correlator 31" module calculates an intercorrelation between this signal and reference code C31.

When "maximum detection" module detects a peak in the correlation results with C1023, the value present in the "11 bits counter" is raised up. This value represents the flight time of the radar signal. Then the reception of the data is performed also, by estimating the sign of the correlation result with code C31. The "delay line" module is used to synchronize the results of both correlators; because there are different response times of about 10 chips and 5 chips.

5.4. The responder computing unit

Besides the responder, to ensure the function of localization, a copy of the received signal is sent back to the interrogator. And in order to exchange data, we exploit the C1023 code sent by the interrogator to synchronize the two components. To ensure that, we compute an intercorrelation between the received signal and code C1023. The detection unit algorithm will take care to detect a local maximum in a guard interval. The presence of one peak indicates that a data frame is being sent. Once the synchronization peak is detected, the sign corresponding to the second correlator peak will be estimated. If the transponder has some data to transmit, we wait until a C1023 peak is detected; then, instead of sending a copy of the received signal, the transponder will send the package of modulated C31 present in the "FIFO data."

At the first interrogator stage, the correlation function is calculated using the C1023 code (Figure 8). The peak position determines the distance and the synchronization for the data frame. At the second stage, a second correlation is calculated with the C31 code to detect data information as by the DS-CDMA decoding technique.

6. EXPERIMENTAL RESULTS

Some trials have been carried out with the preliminary mock-up in real life conditions to evaluate the localization and the communication functions. The measurements have been made in the different environments the radar maybe used. Figure 9 shows the mock up placed in the front of the vehicles.

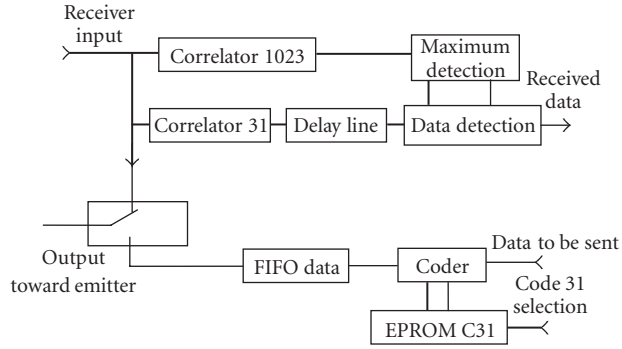


Figure 8: Different modules implemented in the FPGA component of the interrogator.

Figure 9: Measurement made in the tunnel using the realized mock-up.

An example of the received signal from the transponder located 100 meters far from the interrogator is shown on Figure 10.

We can note on this graph that there are many interferences with other systems working in the same frequency band, that is, 2.2 GHz to 2.4 GHz.

The architecture of this radar is efficient in these conditions and avoids the interference effects. In fact, Figure 11 shows the performances of the correlation tools associated to BPRS codes. The corresponding peaks could be easily detected.

Figure 12 presents a zoom on the first 4000 samples of the signal shown on Figure 10. It corresponds to a signal processed with a signal analyzer using an oversampling ratio of about 40. The signal has a rythm of about 50 MHz. The intrinsic central processing unit includes two ADC that can work at 100 megasamples per second. An oversampling ratio of about 2 or 4 could there be reached.

On Figure 13, the normalized intercorrelation result of the received signal with the code C1023 is presented together to the time reference. The delay time between the two signals corresponds to the flight time relative to the distance.

On Figures 14 and 15, the result obtained after the correlation between the received signal and the localization code

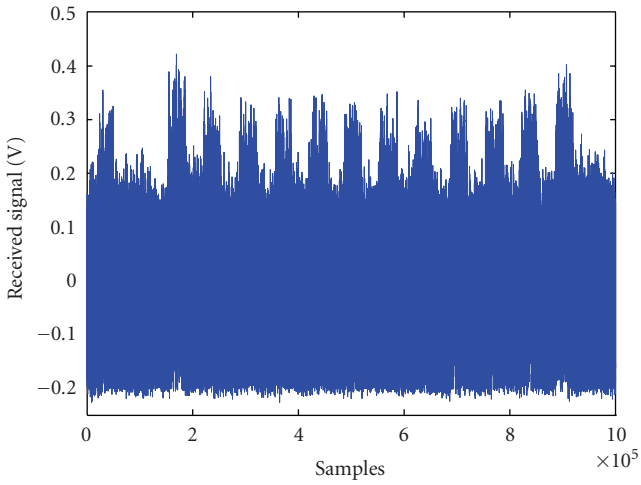


Figure 10: Received signal target at 100 meters.

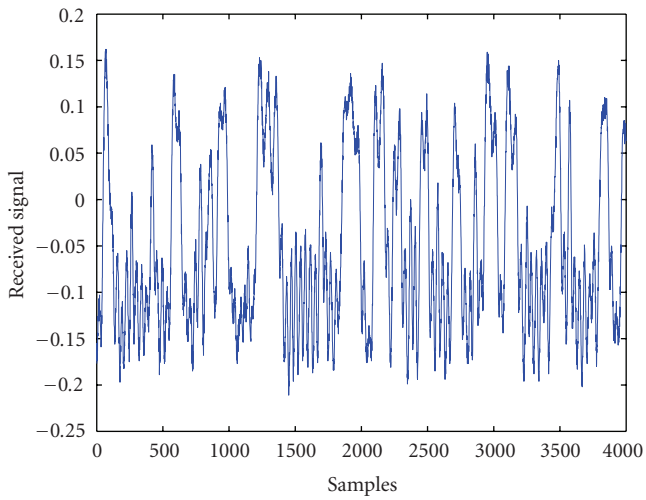


Figure 12: Received signal zoom first 4000 samples.

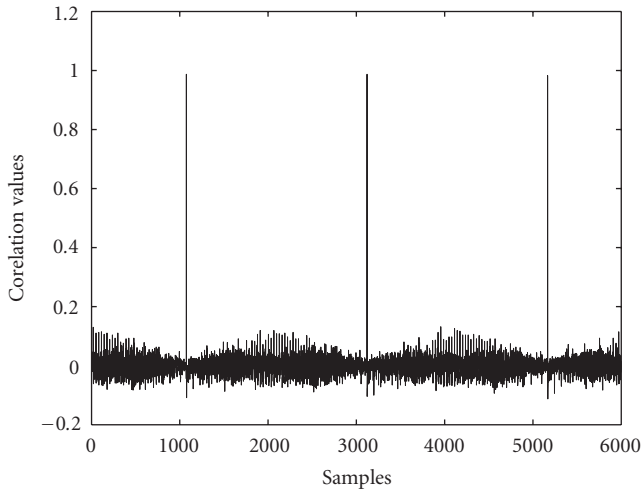


Figure 11: Correlation result with C1023.

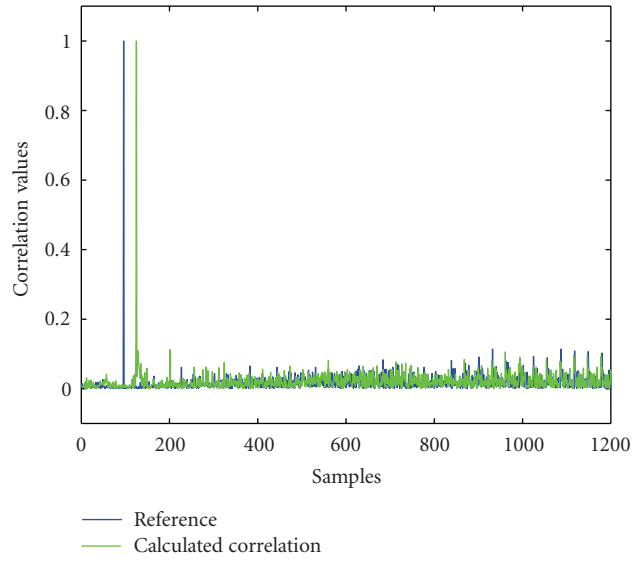


Figure 13: Correlation result with C1023.

C1023 (black color) and data code C31 (gray color) are represented.

We can see on Figure 14 that, between two localization codes, a series of data sent could be extracted easily. Moreover, on Figure 15, the data peaks are periodically distributed spaced of 31 chips. Between the localization peak and the first data peak, only a 26 chips delay exists (instead of 31) due to the difference in response times between localization correlation and data correlation. This difference, as we mentioned previously, is about 5 chips.

7. CONCLUSION

In this paper, new cooperative radar dedicated to automatic guided trains is presented. This sensor allows two functionalities: localization and high data flow communication. To combine these functionalities, original multiplexing methods called SSS2 have been proposed. This technique is inspired from CDMA base and uses successively two coding frames to ensure the multiplexing between the localization and the communication part and at the same time to give automatically multiuser access. With this method, the CODIBDT sensor achieves interesting performances in terms of localization range that is about of 800 m in subway tunnel and 700 m in open space with resolution of 1.5 m. However, the communication between vehicles is established with flow data rate up to 1.6 Mbits/s.

Many simulations have been computed to look further the system's performance in terms of computing time and complexity. And in order to validate simulations results, a mock-up have been build outfitted with flexible component like FPGA devices. This FPGA device contains the computing

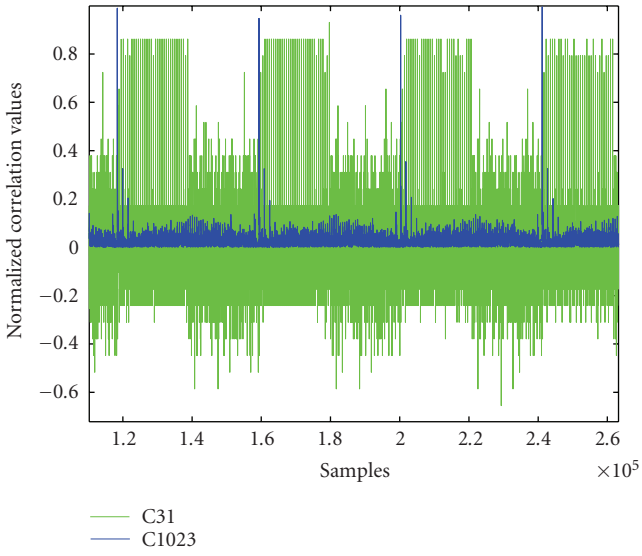


Figure 14: Correlation result with C1023 (black) and C31 (gray).

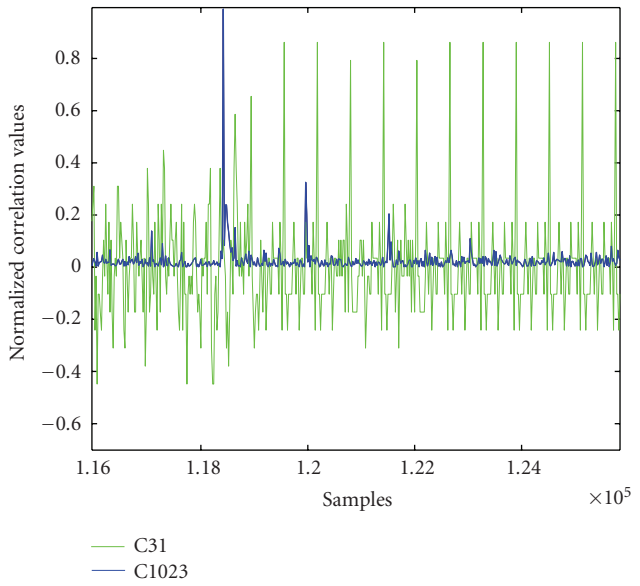


Figure 15: Correlation result with C1023 (black) and C31 (gray) zoom of Figure 14.

unit of the whole system (interrogator and responder) including also the coding technique and the detection algorithm. Future works will be oriented to multiplexing technique enhancement. Higher data flow rates could be reached by the same system using other coding method. Simulations of these methods will be performed with real channel model corresponding to free area and tunnel.

REFERENCES

[1] J. King, "The security of merchant shipping," *Marine Policy*, vol. 29, no. 3, pp. 235–245, 2005.

[2] T. Wahl, G. K. Høye, A. Lyngvi, and B. T. Narheim, "New possible roles of small satellites in maritime surveillance," *Acta Astronautica*, vol. 56, no. 1-2, pp. 273–277, 2005.

[3] B. Fremont, A. Menhaj, P. Deloof, and M. Heddebaut, "A cooperative collision avoidance and communication system for railway transports," in *Proceedings of IEEE Conference on Intelligent Transportation Systems (ITSC '00)*, pp. 216–221, Dearborn, Mich, USA, October 2000.

[4] Y. Elhillali, C. Tatkeu, A. Rivenq, and J. M. Rouvaen, "Enhancement and implementation of a localization and communication system dedicated to guided transports," in *Proceedings of the 6th International Conference on ITS Telecommunications (ITST '06)*, pp. 596–599, Chengdu, China, June 2006.

[5] T. Ottosson, "Coding, modulation and multiuser decoding for DSCDMA systems," *Doktorsavhandlingar vid Chalmers Tekniska Hogskola, 1343*, p. 192, 1997.

[6] C. Tatkeu, P. Deloof, Y. Elhillali, A. Rivenq, and J. M. Rouvaen, "A cooperative radar system for collision avoidance and communications between vehicles," in *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC '06)*, pp. 1012–1016, September, Toronto, Canada 2006.

[7] C. Tatkeu, Y. Elhillali, A. Rivenq, and J. M. Rouvaen, "Evaluation of coding's methods for the development of a radar sensor for localization and communication dedicated to guided transport," in *Proceedings of the 60th IEEE Vehicular Technology Conference (VTC '04)*, vol. 3, pp. 2244–2247, Los Angeles, Calif, USA, September 2004.

[8] R. Dixon, Ed., *Spread Spectrum Techniques*, IEEE Press, New York, NY, USA, 1976.

[9] J. Glas, "Spread Spectrum Techniques," Delft University of Technology, 1996, http://cobalt.et.tudelft.nl/~glas/ssc/techn/techniques.html.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 71043, 14 pages
doi:10.1155/2007/71043

Research Article

System-Platforms-Based SystemC TLM Design of Image Processing Chains for Embedded Applications

Muhammad Omer Cheema,[1,2] Lionel Lacassagne,[2] and Omar Hammami[1]

[1] *EECS Department, Ecole Nationale Superieure de Techniques Avancees, 32 Boulevard Victor, 75739 Paris, France*
[2] *Axis Department, University of Paris Sud, 91405 Orsay, France*

Received 18 October 2006; Accepted 3 May 2007

Recommended by Paolo Lombardi

Intelligent vehicle design is a complex task which requires multidomains modeling and abstraction. Transaction-level modeling (TLM) and component-based software development approaches accelerate the process of an embedded system design and simulation and hence improve the overall productivity. On the other hand, system-level design languages facilitate the fast hardware synthesis at behavioral level of abstraction. In this paper, we introduce an approach for hardware/software codesign of image processing applications targeted towards intelligent vehicle that uses platform-based SystemC TLM and component-based software design approaches along with HW synthesis using SystemC to accelerate system design and verification process. Our experiments show the effectiveness of our methodology.

1. INTRODUCTION

Embedded systems using image processing algorithms represent an important segment of today's electronic industry. New developments and research trends for intelligent vehicles include image analysis, video-based lane estimation and tracking for driver assistance, and intelligent cruise control applications [1–5]. While there has been a notable growth in the use and application of these systems, the design process has become a remarkably difficult problem due to the increasing design complexity and shortening time to market [6]. A lot of work is being done to propose the methodologies to accelerate automotive system design and verification process based on multimodeling paradigm. This work has resulted in a set of techniques to shorten the time consuming steps in system design process. For example, transaction-level modeling makes system simulation significantly faster than the register transfer level. Platform-based design comes one step forward and exploits the reusability of IP components for complex embedded systems. Image processing chain using component-based modeling shortens the software design time. Behavioral synthesis techniques using system-level design languages (SLDLs) accelerate the hardware realization process. Based on these techniques, many tools have been introduced for system-on-chip (SoC) designers that allow them to make informed decisions early in the design process which can be the difference in getting products to market quicker. The ability to quickly evaluate the cross-domain effects of design tradeoffs on performance, power, timing, and die size gives a huge advantage much earlier than was ever achievable with traditional design techniques.

While time to market is an important parameter for system design, an even more important aspect of system design is to optimally utilize the existing techniques to meet the computation requirements of image processing applications. Classically, these optimization techniques have been introduced at microprocessor level by customizing the processors and generating digital signal processors, pipelining the hardware to exploit instruction-level parallelism, vectorizing techniques to exploit data-level parallelism, and so forth. In system-level design era, more emphasis has been on the techniques that are more concerned with interaction between multiple processing elements instead of optimization of individual processing elements, that is, heterogeneous MPSoCs. HW/SW codesign is a key element in modern SoC design techniques. In a traditional system design process, computation intensive elements are implemented in hardware which results in the significant system speedup at the cost of increase in hardware costs.

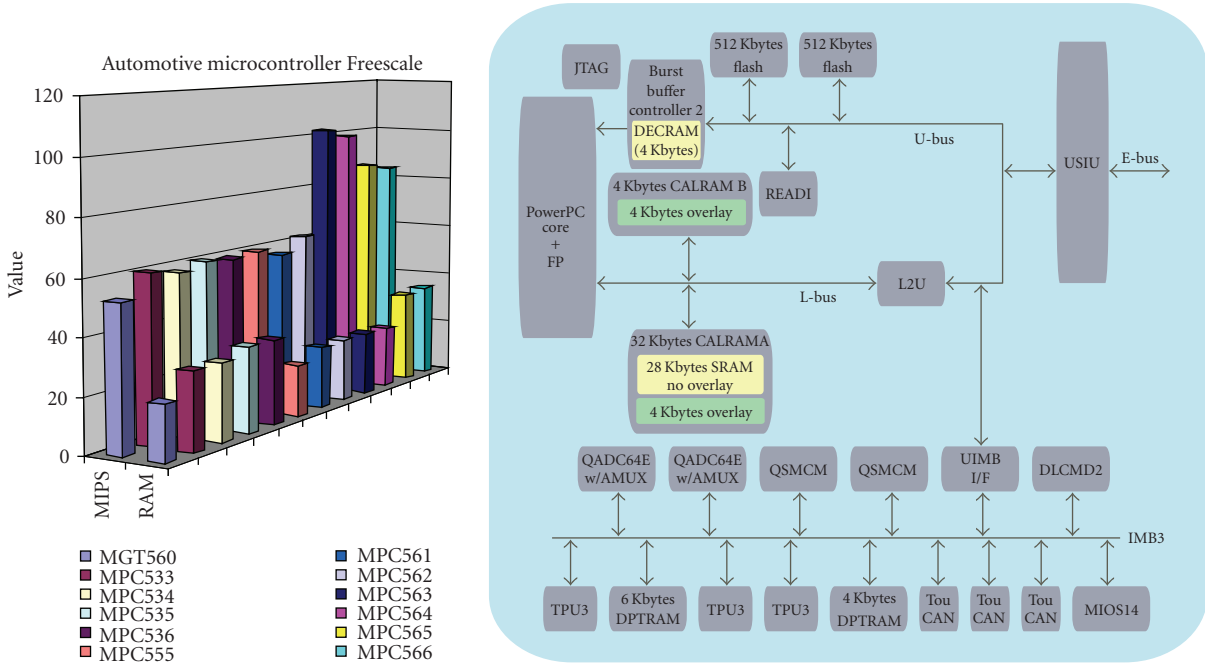


Figure 1: Freescale MPC controllers: (a) MIPS/embedded RAM, (b) MPC 565 block diagram.

In this paper, we propose an HW/SW codesign methodology that advocates the use of the following.

(i) Platform-based transaction-level modeling to accelerate system-level design and verification.
(ii) Behavioral synthesis for fast hardware modeling.
(iii) Component-based SW development to accelerate software design.

Using these techniques, we show that complex embedded systems can be modeled and validated in short times while providing satisfactory system performance.

Rest of the paper is organized as follows. Section 2 presents related work. Section 3 overviews the general vehicle design methodology and establishes a direct link with our proposal. Section 4 describes a very recent SystemC TLM platform: the IBM PowerPC evaluation kit. Section 5 explains our system design methodology and Section 6 describes the experiment environment and results. Future work and a proposed combined UML-SystemC TLM platform are described in Section 7. Finally, Section 8 concludes.

2. RELATED WORK

When designing embedded applications for intelligent vehicles a whole set of microcontrollers are available. An example of such an offer comes from Freescale [7] with their PowerPC-based microcontrollers (Figure 1).

However, although diverse in the MIPS and embedded RAM these microcontrollers do not offer enough flexibility to add specific hardware accelerators such as those required by image processing applications. The PowerPC core of these microcontrollers is not sufficient in this peripherals intensive environment to exclusively support software computation intensive applications. It is then necessary to customize these microcontrollers by adding additional resources while keeping the general platform with its peripherals. A system-design approach is needed. Our work is based on three different aspects of system design. Although some work has been done on each of these aspects at individual level, no effort has been made to propose a complete HW/SW codesign flow that gets benefit out of all these techniques to improve the system productivity. In the following sections, we will present the related work done on each of these domains. Transaction-level modeling based on system-level design languages has proven to be a fast and efficient way of system design [8–10]. It has been shown that simulation at this level is much faster [8] than register transfer level (RTL) and makes it possible for us to explore the system design space for HW/SW partitioning and parameterization. The idea of transaction-level modeling (TLM) is to provide in an early phase of the hardware development transaction-level models of the hardware. Based on this technique, a fast-enough simulation environment is the basis for the development of hardware and hardware dependent software. The presumption is to run these transaction-level models at several tens or some hundreds of thousand transactions per second which should be fast-enough for system-level modeling and verification. A lot of work has been done on behavioral synthesis. With the evolution of system-level design languages, the interest in efficient hardware synthesis based on behavioral description of a hardware module has also been visible. A few tools for behavioral SystemC synthesis [11, 12] are available in the market.

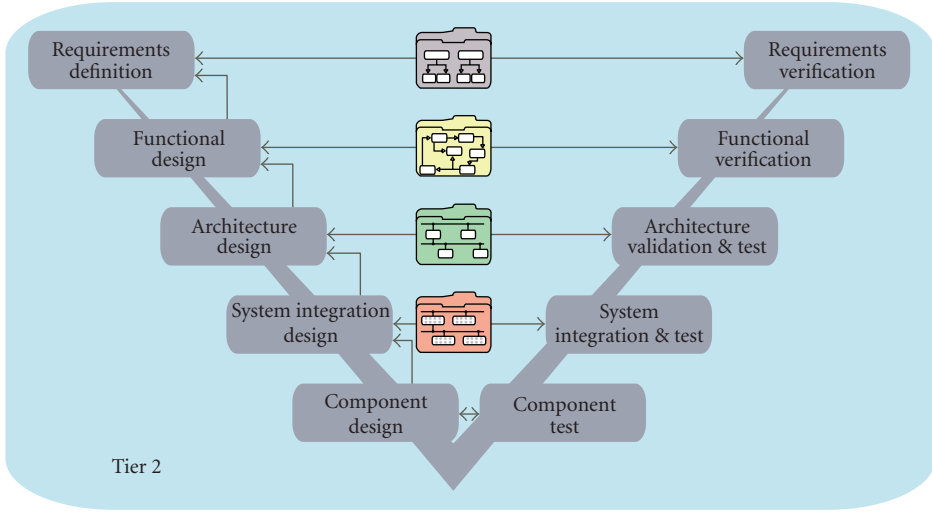


Figure 2: V design cycle.

For a system designer, behavioral system is very attractive for hardware modeling as it has shown to result in a lot of productivity improvements [10]. On the other hand, image processing chain development is a relatively old technique for software development that uses component-based software design to accelerate the software development process [13, 14]. On another side, UML-based design flows [15–21] have been proposed whether or not with SystemC [22–27] as an approach for fast executable specifications. However, to the best of our knowledge no tools have been proposed which combine UML- and SystemC TLM-based platforms. In this regard, additional work remains to be done in order to obtain a seamless flow.

3. GENERAL VEHICLE DESIGN METHODOLOGY

Vehicle design methodology follows the V-cycle model where from a requirements definition the process moves to functional design, architecture design, system-integration design, and component design before testing and verifying the same steps in reverse chronological order (Figure 2).

In the automotive domain, system integrator (car manufacturers) collaborate with system designer (tier 1 supplier, e.g., Valeo) while themselves collaborate with component designers (tier 2 supplier, e.g., Freescale); see (Figure 3).

This includes various domains such as electronics, software, control, and mechanics. However, design and validation requires a modeling environment to integrate all these disciplines. Unfortunately, running a complete multidomain exploration through simulation is unfeasible. Although component reuse helps somewhat reduce the challenge, it prevents from all the possible customizations existing in current system-on-chip design methodologies. Indeed, system on chip makes intensive uses of various IPs and among them parametrizable IPs which best fit the requirements of the application. This allows new concurrent design methodologies between embedded software design, architecture, intermicrocontroller communication and implementation. This flattening of the design process can be best managed through platform-based design at the TLM level.

4. PLATFORM-BASED TLM DESIGN PROCESS

Platforms have been proposed by semiconductor manufacturers in an effort to ease system-level design and allow system designers to concentrate on essential issues such as hardware-software partitioning, system parameters tuning, and design of specific hardware accelerators. This makes the reuse of platform-based designs easier than specific designs.

4.1. *Platforms and IBM platform driven design methodology*

The IBM CoreConnect platform [28] described in Figure 4 allows the easy connection of various components, system core, and peripheral core to the CoreConnect bus architecture.

It also includes IPs of PLB to OPB and OPB to PLB bridges and direct memory access (DMA) controller, OPB-attached external bus controller (EBCO), universal asynchronous receiver/transmitter (UART), universal interrupt controller (UIC), and double data rate (DDR) memory controller. Several other peripherals are available among them CAN controllers. The platform does not specify a specific processor core although IBM family of embedded PowerPC processors connection is straightforward. This platform which mainly specifies a model-based platform have all associated tools and libraries for quick ASIC or FPGA platform design. System core and peripheral core can be any type of user-designed components whether hardware accelerators or specific peripherals and devices.

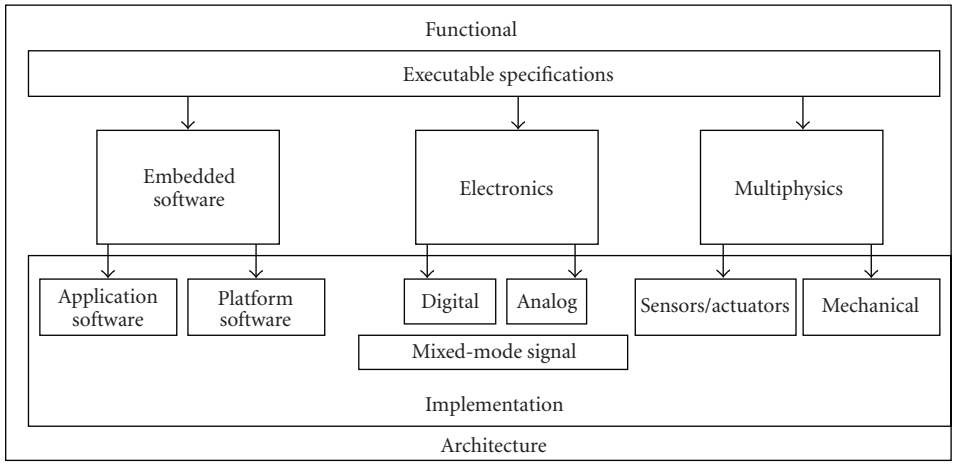


Figure 3: Decomposition.

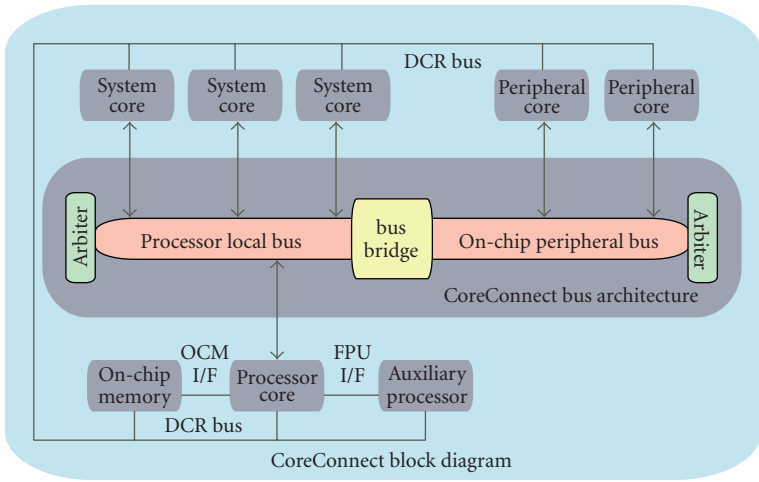


Figure 4: IBM CoreConnect platform.

4.2. *IBM SystemC TLM platform*

The SystemC IEEE standard [29] is a system-level modeling environment which allows the design of various abstraction levels of systems (Figure 5). It spawns from untimed functional to cycle accurate. In between, design space exploration with hardware-software partitioning is conducted with timed functional level of abstraction. Using the model-driven architecture (MDA) terminology [30] we can model computation independent model (CIM), platform-independent model (PIM), and platform-specific model (PSM). Besides, SystemC can model hardware units at RTL level and be synthesizable for various target technologies using tools such as Synopsys [11] and Celoxica [12], which in turn allows multiobjective SystemC space exploration of behavioral synthesis options on area, performance, and power consumption [31] since for any system, all three criteria cannot be optimally met together.

This important point allows SystemC abstraction-level platform-based evaluation taking into account area and energy aspects, and this for proper design space exploration with implementation constraints. In addition to these levels of abstraction, transaction-level modeling and abstraction level [8, 9] have been introduced to fasten simulation of communications between components by considering communications exchange at transaction level instead of bus cycle accurate levels. Benefits of TLM abstraction-level design have been clearly demonstrated [8, 9].

Using the IBM CoreConnect SystemC modeling environment PEK [32], designers are able to put together SystemC models for complete systems including PowerPC processors, CoreConnect bus structures, and peripherals. These models may be simulated using the standard OSCI SystemC [29] runtime libraries and/or vendor environments. The IBM CoreConnect SystemC modeling environment TLM platform models and environment provide designers with a

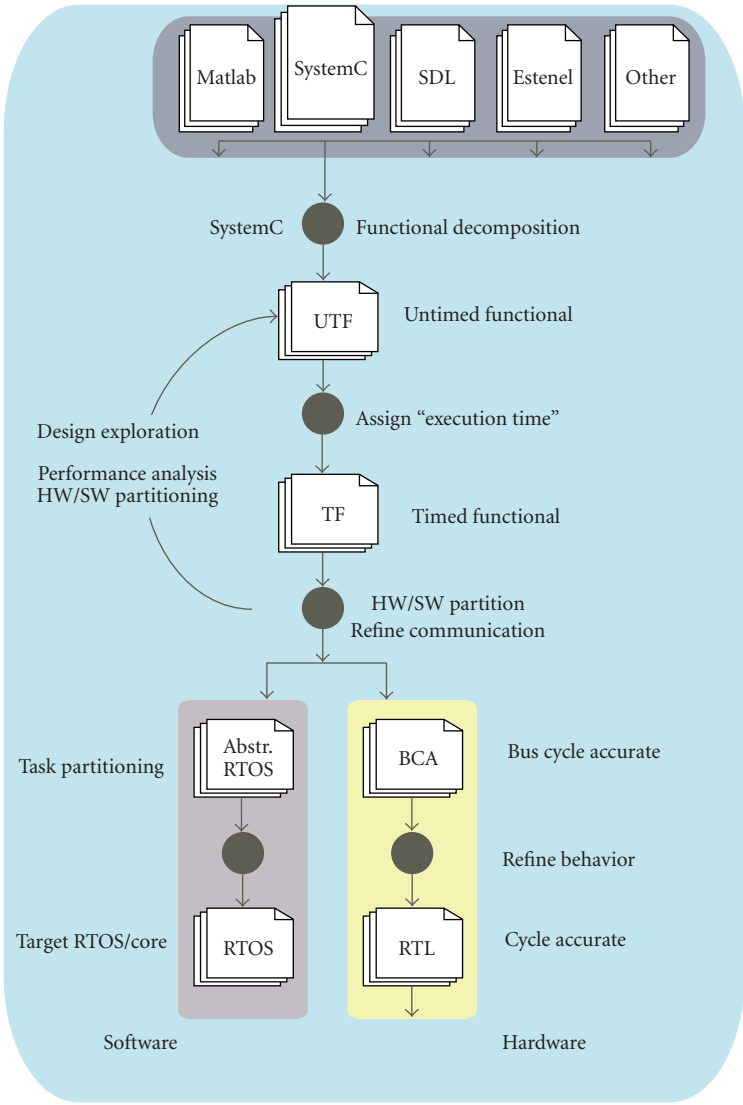


Figure 5: SystemC system design flow.

system simulation/verification capability with the following characteristics.

(i) Simulate real application software interacting with models for IP cores and the environment for full system functional and timing verification possibly under real-time constraints.

(ii) Verify that system supports enough bandwidth and concurrency for target applications.

(iii) Verify core interconnections and communications through buses and other channels.

(iv) Model the transactions occurring over communication channels with no restriction on communication type.

These objectives are achieved with additional practical aspects such as simulation performance must be enough to run a significant software application with an operating system booted on the system. In addition, the level of abstraction allows the following.

(i) Computation (inside a core) does not need to be modeled on a cycle-by-cycle basis, as long as the input-output delays are cycle-approximate which implies that for hardware accelerators both SystemC and C are allowed.

(ii) Intercore communication must be cycle-approximate, which implies cycle-approximate protocol modeling.

(iii) The processor model does not have to be a true architectural model; a software-based instruction set simulator (ISS) can be used, provided that the performance and timing accuracy are adequate.

In order to simulate real software, including the initialization and internal register programming, the models must be "bit-true" and register accurate, from an API point of view.

That is, the models must provide APIs to allow programming of registers as if the user were programming the real hardware device, including the proper number of bits and address offsets. Internal to the model, these "registers" may be coded in any way (e.g., variables, classes, structs, etc.) as long as their API programming makes them look like real registers to the users. Models need not be a precise architectural representation of the hardware. They may be behavioral models as long as they are cycle-approximate representations of the hardware for the transactions of interest (i.e., the actual transactions being modeled). There may be several clocks in the system (e.g., CPU, PLB, OPB). All models must be "macro synchronized" with one or more clocks. This means that for the atomic transactions being modeled, the transaction boundaries (begin and end) are synchronized with the appropriate clock. Inside an atomic transaction, there is no need to model it on a cycle-by-cycle basis. An atomic transaction is a set of actions implemented by a model, which once started, is finished, that is, it cannot be interrupted. Our system-design approach using IBM's PowerPC 405 evaluation kit (PEK) [32] allows designers to evaluate, build, and verify SoC designs using transaction-level modeling. However, PEK does not provide synthesis (area estimate) or energy consumption tools.

4.2.1. SW development, compilation, execution, debugging

In PEK, the PowerPC processors (PPC 405/PPC450) are modeled using an instruction-set simulator (ISS). The ISS is instantiated inside a SystemC wrapper module, which implements the interface between the ISS and the PLB bus model. The ISS runs synchronized with the PLB SystemC model (although the clock frequencies may be different). For running a software over this PowerPC processor, code should be written in ANSI C and it should be compiled using GNU cross compiler for PowerPC architecture.

The ISS works in tandem with a dedicated debugger called RiscWatch (RW) [33]. RW allows the user to debug the code running on the ISS while accessing all architectural registers and cache contents at any instance during the execution process.

4.2.2. HW development, compilation, execution, monitoring

Hardware modules should be modeled in SystemC using the IBM TLM APIs. Then these modules can be added to the platform by connecting them to the appropriate bus at certain addresses which were dedicated in software for these hardware modules. Both, synthesizable and nonsynthesizable SystemC can be used for modeling of hardware modules at this level but for getting area and energy estimates, it is important that SystemC code be part of standard SystemC synthesizable subset draft (currently under review by the OSCI synthesis working group) [34]. If we want to integrate already existing SystemC hardware modules, wrappers should be written that wrap the existing code for making it compatible with IBM TLM APIs. We have written generic interfaces which provide a generalized HW/SW interface hence reducing the modeling work required to generate different interfaces for every hardware module based on its control flow.

For simulation of SystemC, standard systemc functionality can be used for .vcd file generation, bus traffic monitoring and other parameters. We have also written the dedicated hardware modules which are connected with the appropriate components in the system and provide us with the exact timing and related information of various events taking place in the hardware environment of the system.

4.2.3. Creating and managing transactions

In a real system, tasks may execute concurrently or sequentially. A task that is executed sequentially, after another task, must wait till the first task has completed before starting. In this case, the first task is called a blocking task (transaction). A task that is executed concurrently with another need not wait for the first one to finish before starting. The first task, in this case, is called a nonblocking task (transaction).

Transactions may be blocking or nonblocking. For example, if a bus master issues a blocking transaction, then the transaction function call will have to complete before the master is allowed to initiate other transactions. Alternatively, if the bus master issues a nonblocking transaction, then the transaction function call will return immediately, allowing the master to do other work while the bus completes the requested transaction. In this case, the master is responsible for checking the status of the transaction before being able to use any result from it. Blocking or nonblocking transactions are not related to the amount of data being transferred or to the types of transfer supported by the bus protocols. Both multibyte burst transfers as well as single-byte transfers may be implemented as blocking or nonblocking transactions.

When building a platform, the designer has to specify the address ranges of memory and peripherals attached to the PLB/OPB busses. The ISS, upon encountering an instruction which does a load/store to/from a memory location on the bus, will call a function in the wrapper code which, in turn, issues the necessary transactions on the PLB bus. The address ranges of local memory, bus memory, cache sizes, cacheable regions, and so forth, can all be configured in the ISS and the SystemC models.

4.2.4. IP parameterization

Various parameters can be adjusted for the processor IPs and other IPs implemented in the system. For a processor IP, when the ISS is started, it loads a configuration file which contains all the configurable parameters for running the ISS. The configuration file name may be changed in the Tcl script invoking the simulation. The parameters in the file allow the setting of local memory regions, cache sizes, processor clock period, among other characteristics. For example, we can adjust the value of data and Instruction Cache sizes to be 0, 1024, 2048, 4096, 8192, 16384, 32768, and 65536 for the 405

processor. Besides setting the caches sizes, the cache regions need to be configured, that is, the user needs to specify which memory regions are cacheable or not. This is done by setting appropriate values into special purpose registers DCCR and ICCR. These are 32-bit registers, and each bit must be set to 1 if the corresponding memory region should be cacheable

The PowerPC uses two special-purpose registers (SPRs) for enabling and configuring interrupts. The first register is the machine state register (MSR) which controls processor core functions such as the enabling and disabling of interrupts and address translation. The second register is the exception vector prefix register (EVPR). The EVPR is a 32-bit register whose high-order 16 bits contain the prefix for the address of an interrupt handling routine. The 16-bit interrupt vector offsets are concatenated to the right of the high-order bits of the EVPR to form the 32-bit address of an interrupt handling routine. Using RiscWatch commands and manipulating startup files to be read from RiscWatch, we can enable/disable cachebility, interrupts, and vary the cache sizes. While on the other hand, CPU, bus, and hardware IP configuration-based parameters can be adjusted in top level file for hardware description where the hardware modules are being initialized.

Provision of these IPs and ease of modeling makes IBM TLM a suitable tool for platform generation and its performance analysis early in the system design cycle.

5. PROPOSED METHODOLOGY

It should be clear from Section 4 that IBM PEK provides almost all important aspects of system design. That is why we have based our methodology for HW/SW codesign on this tool. However, our methodology will be equally valid for all other tools having similar modeling and simulation functionality. Our HW/SW codesign approach has the following essential steps.

(a) Image processing chain development.
(b) Software profiling.
(c) Hardware modeling of image processing operators.
(d) Performance/cost comparison for HW/SW implementations.
(e) Platform generation, system design space exploration.

(a) Image processing chain development

Our system codesign approach starts from development of image processing chain (IPC). Roughly speaking, an image processing chain consists of various image processing operators placed in the form of directed graph according to the data flow patterns of the application. An image processing chain is shown in Figure 6.

This IPC describes the working of a Harris corner detector. IPC development process is very rapid as normally most of the operators are already available in the operator's library and they need only to be initialized in a top-level function to form an image processing chain and secondly it provides a very clean and modular way to optimize various parts of the application without the need of thorough testing and debugging. In our case, we have used coding guidelines as recommended by numerical recipes [35] which simplifies the IPC development process even further.

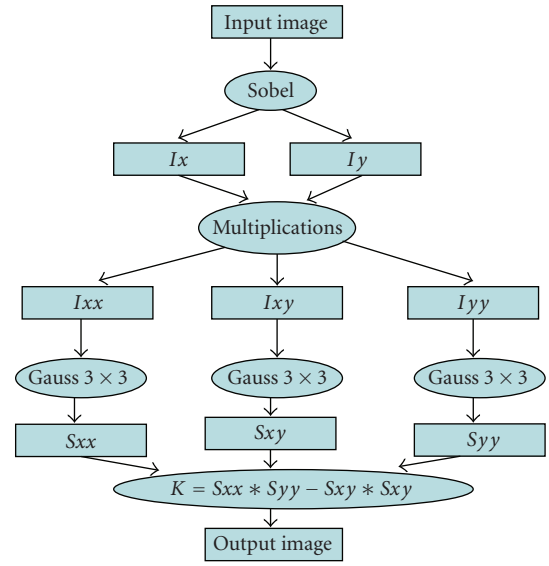


Figure 6: Harris corner detector chain.

(b) Software profiling

In this step, we execute the image processing chain over the PowerPC 405 IP provided with PowerPC evaluation kit. Using RisCWatch commands, we get the performance results of various software components in the system and detect the performance bottlenecks in the system. Software profiling is done for various data and instruction caches sizes and bus widths. This information helps the system designer take the partitioning decisions in later stages.

(c) Hardware modeling of image processing operators

In the next step of our system design approach, area and energy estimates are obtained for the operators implemented in the image processing chain. At SystemC behavioral level, the tools for estimating area and energy consumption have recently been showing their progress in the EDA industry. We use Celoxica's agility compiler [12] for area estimation in our case but our approach is valid for any behavioral-level synthesis tool in the market. As we advocate the fast chain development through libraries containing image processing operators, similar libraries can also be developed for equivalent SystemC image processing operators which will be reusable over a range of projects hence considerably shortening the hardware development times as well. At the end of this step, we have speed and area estimates for all the components of the image processing chain to be synthesized. This information is stored in a database and is used during HW/SW partitioning done in the next step.

Another important thing to be noted is that HW synthesis is also a multiobjective optimization problem. Previously,

[31] have worked over efficient HW synthesis from SystemC and shown that for a given SystemC description, various HW configurations can be generated varying in area, energy, and clock speeds. Then the most suitable configuration out of the set of pareto optimal configurations can be used in the rest of the synthesis methodology. Right now, we do not consider this HW design space exploration for optimal area/energy and speed constraints but in our future work, we plan to introduce this multiobjective optimization problem in our synthesis flow as well.

(d) Performance comparison for HW/SW implementations

At this stage of system codesign, system designer has profiling results of software as well as hardware implementation costs and the performance of the same operator in the hardware. So, in this stage performance of various individual operators is compared and further possibilities of system design are explored.

(e) Platform generation, system-design space exploration

Like traditional hardware/software codesign approaches, our target is to synthesize a system based on a general purpose processor (in our case, IBM PowerPC 405) and extended with the help of suitable hardware accelerators to significantly improve the system performance without too much increase in the hardware costs. We have chosen PowerPC 405 as a general purpose processor in our methodology because of its extensive usage in embedded systems and availability of its systemC models that provide ease of platform design based on its architecture. Our target platform is shown in Figure 7. Our target is to shift the functionality from image processing chain to the hardware accelerators such that system gets good performance improvements without too much hardware costs.

In this stage, we perform the system-level simulation. Based on the results of last step, we generate various configurations of the system putting different operators in hardware and then observing the system performance. Based on these results and application requirements, a suitable configuration is chosen and finalized as a solution to HW/SW codesign issue.

(f) Parameter tuning

In the last step of image processing chain synthesis flow, we perform the parameterization of the system. At this stage, our problem becomes equivalent to (application specific standard products) ASSP parameterization. In ASSP, hardware component of the system is fixed; hence only tuning of some soft parameters is performed for these platforms to improve the application performance and resource usage. Examples of such soft parameters include interrupt and arbitration priorities. Further parameters associated with more detailed aspects of the behavior of individual system IPs may also be available. We deal with the problem manually instead of relying on a design space exploration algorithm and our approach is to start tuning the system with the maximum resources available and keep on cutting down the resource availability until the system performance remains well within the limits and bringing down the value of a parameter does not dramatically affect system performance. However, in the future we plan to tackle this parameterization problem using automatic multiobjective optimization techniques.

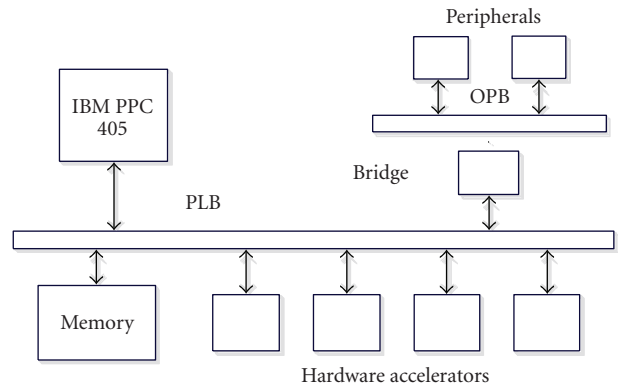


Figure 7: Target platform built using IBM TLM.

6. EVALUATION RESULTS

We have tested our approach of HW/SW codesign for Harris corner detector application described in Figure 6. Harris corner detector is frequently used for point-of-interest (PoI) detection in real-time embedded applications during data preprocessing phase.

The first step, according to our methodology, was to develop image processing chain (IPC). As mentioned in the previous section, we use numerical recipes guidelines for component-based software development and it enables us to develop/modify IPC in shorter times because of utilization of existing library elements and clarity of application flow. At this stage, we put all the components in software. Software is profiled for various image sizes and results are obtained. Next step is to implement hardware and estimate times taken for execution of an operator entirely implemented in hardware and compare it to the performance estimates of software. The results obtained from hardware synthesis and its performance as compared with software-based operations are shown in Table 1 and Figure 6.

Results in Table 1 show the synthesis results of behavioral SystemC modules for different operators computing different sizes of data. We can see that with the change in data size, memory requirements of the operator also change, while the part of the logic which is related to computation remains the same. Similarly, critical path of the system remains the same as it mainly depends on computational logic structure. Based on the synthesized frequencies and number of cycles required to perform each operation, last column shows the computation time for each hardware operator for a given size of data. It is again worth mentioning that synthesis of these operators depends largely on the intended design. For example, adding multiport memories can result in acceleration in read

Table 1: Synthesis results for Harris corner detector chain.

Module name	Area (computational logic and memory)			Critical path (ns)	Synth. freq. (MHz)	Total comp. time (μs)
	Size	Comp. logic slices	memory (bits)			
Sobel	8×8	218	18432	14.41	69.39	1.845
	16×16	220	18432	14.41	69.39	7.376
	32×32	222	36864	14.41	69.39	29.514
	64×64	224	131072	14.41	69.39	118.06
P2P Mul	8×8	151	36864	11.04	90.33	1.417
	16×16	151	36864	11.04	90.33	5.668
	32×32	152	73728	11.04	90.33	22.67
	64×64	152	262144	11.04	90.33	90.69
Gauss	8×8	184	18432	16.37	61.1	2.095
	16×16	186	18432	16.37	61.1	8.38
	32×32	188	36864	16.37	61.1	33.52
	64×64	190	131072	16.32	61.1	134.1
K = coarsity computation	8×8	351	36864	19.32	51.76	2.473
	16×16	352	73728	19.32	51.76	9.892
	32×32	353	147456	19.32	51.76	39.567
	64×64	354	294912	19.32	51.76	158.269

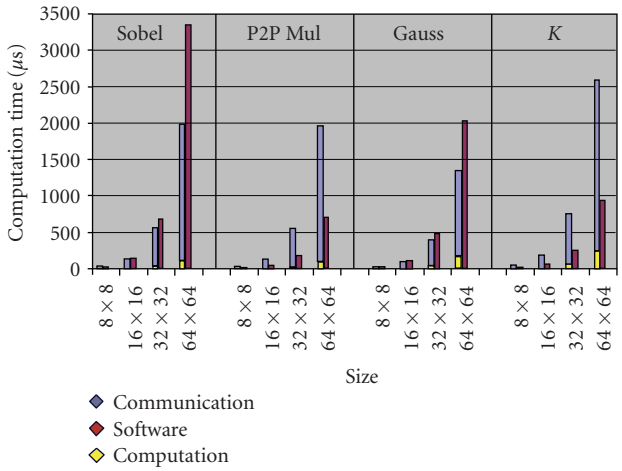


Figure 8: HW performance versus SW performance of operators.

operations from memory while unrolling the loops in SystemC code can result in performance improvement at a cost of an increase in area.

Figure 8 shows the comparison of execution times of an operator in its hardware and software implementations. There are two things to be noticed here. Firstly, operator computation time for hardware has been shown with two different parameters: computation and communication. Looking at Table 1, one might feel that all hardware implementations will be much faster than their software version but one needs to realize here that implementing a function in hardware requires the data to be communicated to the hardware module which requires changes in software design where computation functions are replaced by data transfer functions. Although image processing applications seem to be computation intensive, it should be noted that most of the time is taken up by communication while computation is only a fraction of total time taken by the hardware. An ideal function to be implemented in hardware will be the one which has lesser data to be transferred from/to the hardware to/from the general purpose processor. Secondly, in the example, we can see that Gaussian and Sobel operators seem to be better candidates to be put in hardware while coarsity computation in hardware lags in performance than its software version because of lesser computation and more communication requirements of the function.

After the performance comparison of operators in hardware and software, next step was to generate the platform and perform the system-level simulation for various configurations. For our system-level simulation, our general purpose processor (PowerPC 405) was running at 333 MHz while it had 16 Kbytes of data and instruction caches.

At first simulation run, we realized that due to data accesses, original software was spending a lot of time in memory access operations. We optimized the software which resulted in an optimized version of the software. After that, we started exploring HW/SW codesign options by generating various versions and getting the simulation results. Table 2 shows a few of the configurations generated and the CPU cycles taken by the system during the simulation. A quick look at the results shows that taking into consideration of hardware implementation cost, configuration 7 provides a good speedup where we have implemented Gaussian and Gradient functions in the hardware. Table 1 shows that adding these operators to hardware will result in a slight increase in computation logic while a bit more increase in memory and at that cost a speedup of more than 2.5 can be obtained.

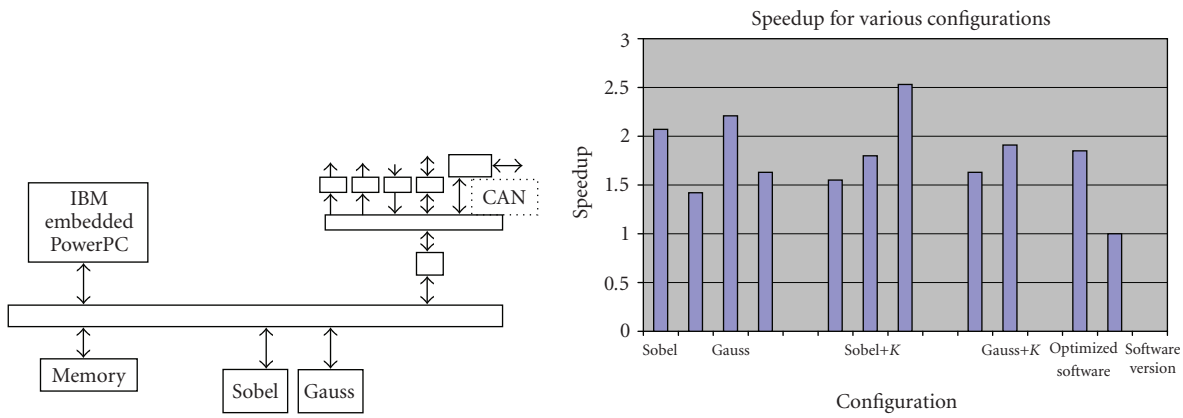


Figure 9: (a) Platform configuration 7. (b) Full HW/SW design space exploration results.

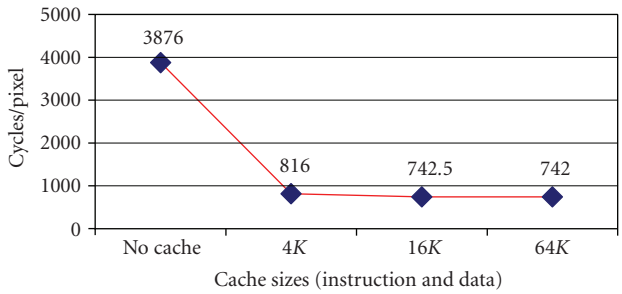


Figure 10: Various cache sizes and system performance.

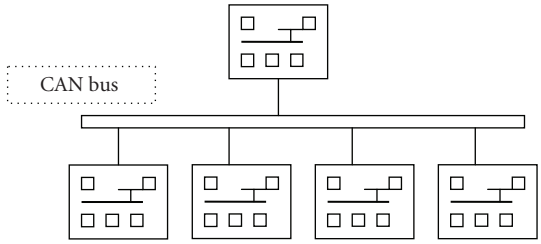


Figure 11: Platforms networked through CAN bus.

Figure 9 graphically represents Table 2. We can see that the configuration involving Sobel and Gaussian operators gives significant speedups while configurations involving point-to-point multiplication and coarsity computation (K) result in worse performance. Based on these results, a system designer might choose configuration 7 for an optimal solution. Or if he has strong area constraints, configurations 1 and 3 can be possible solutions for codesigned system.

When configuration 7 was chosen to be the suitable configuration for our system, next step was the parameterization of the system. Although parameterization involves bus width adjustment, arbitration scheme management and interrupt routine selection, for the sake of simplicity we show the results for optimal sizes of caches. Figure 10 shows the results for various cache sizes and corresponding performance improvement. We can see that cache results in significant performance improvements until 16K of data and instruction cache sizes. But after that, the performance improvements with respect to cache size changes reach a saturation point and there is almost no difference of performance for 16K and 64K caches in the system. Hence we choose 16K data and instruction caches sizes for our final system.

This approach allowed us to alleviate the problem of selecting inadequate microcontrollers for intelligent vehicle design such as those described Section 2. This process can be repeated with other applications in order to build a system based on networked platforms; see Figure 11.

Lastly, we will mention the limitations of the methodology. It should be noticed that we have chosen small image sizes for our system design. Although TLM-level simulation is much faster than RTL-level simulations, it still takes a lot of time for simulation of complex systems. Increasing the image sizes beyond 256×256 for the given example makes it increasingly difficult for exploring the design space thoroughly as it required multiple iterations of simulation for each configuration and one iteration itself takes hours or even days to complete. For larger image sizes where simulation time will dominates the system design time, RTL-level system prototyping and real-time execution over hardware prototyping boards seem to be a better idea where although system prototyping will take longer times but significant time savings can be made by preferring real-time execution over simulations. The approach of [36] can be used in this context.

7. FUTURE WORK: COMBINING UML-BASED SYSTEM-DESIGN FLOW WITH SYSTEMC TLM PLATFORM FOR INTELLIGENT VEHICLES DESIGN

The work presented so far described the potentials of SystemC TLM platform-based design for the system design of embedded applications through the customization of

Table 2: Various configurations and speedups for point-of-interest detection.

Config. no.	Hardware implement	Time (cycle)	Cycle/pixel	Speedup over software version
1	Sobel	3726350	909.75	2.07
2	P2P Mul	5419590	1323.14	1.42
3	Gauss	3490064	852.06	2.21
4	K = coarsity comp.	4725762	1153.75	1.63
5	Sobel + P2P Mul	4970836	1213.58	1.55
6	Sobel + K	4277108	1044.22	1.80
7	Sobel + Gauss	3041510	742.56	2.53
8	Gauss + P2P Mul	4734654	1155.92	1.63
9	Gauss + K	4040826	986.52	1.91
10	Optimized software	4175000	1019.29	1.85
11	Original software version	7717000	1884.03	1

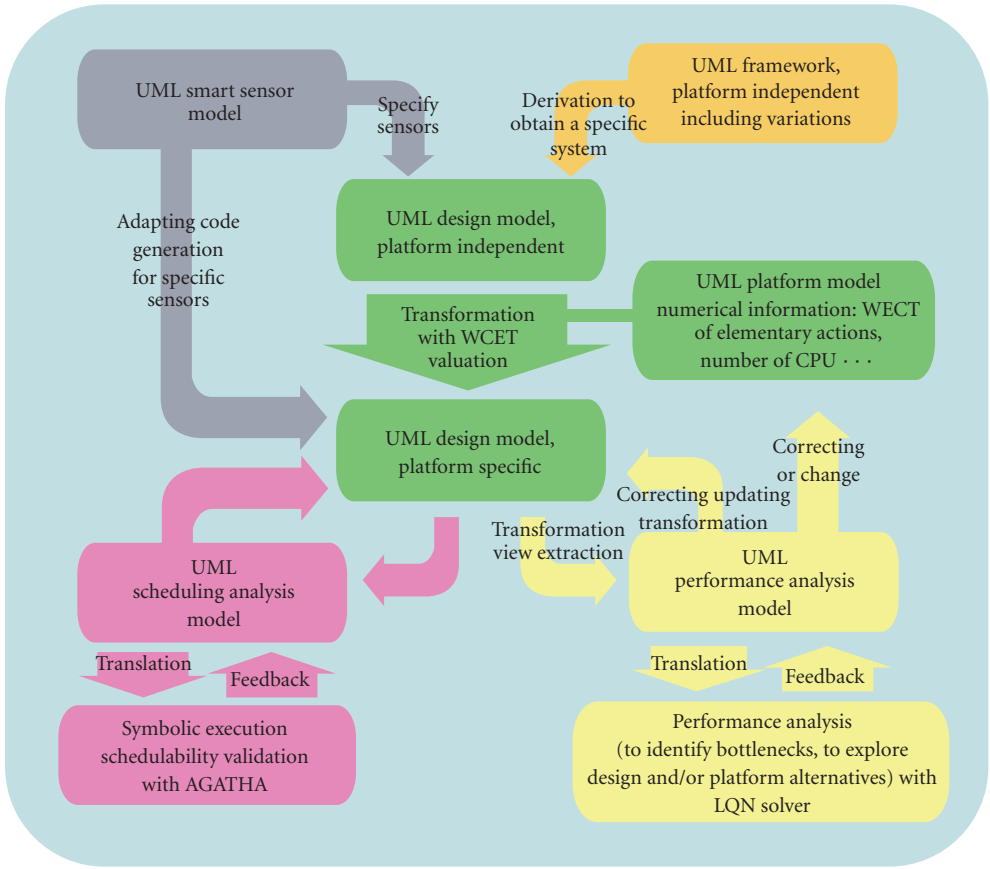


Figure 12: Accord/UML design methodology.

microcontrollers. Clearly important benefits come from this approach with the possibility to get access to implementation details (area, energy consumption) without lowering the design abstraction details down to implementation. This key point clearly contributes to the reduction of the design cycle and the ease of the design space exploration. On the other hand, several research projects have advocated the use of UML-based system design for real-time embedded systems [16–19]. The Accord/UML is a model-based methodology dedicated for the development of embedded real-time applications [16] (Figure 12). The main objectives of the methodology is to specify and prototype embedded real-time systems through three consistent and complementary models describing structure, interaction, and behavior. Examples of applications include smart transducer integration in real-time embedded systems [19].

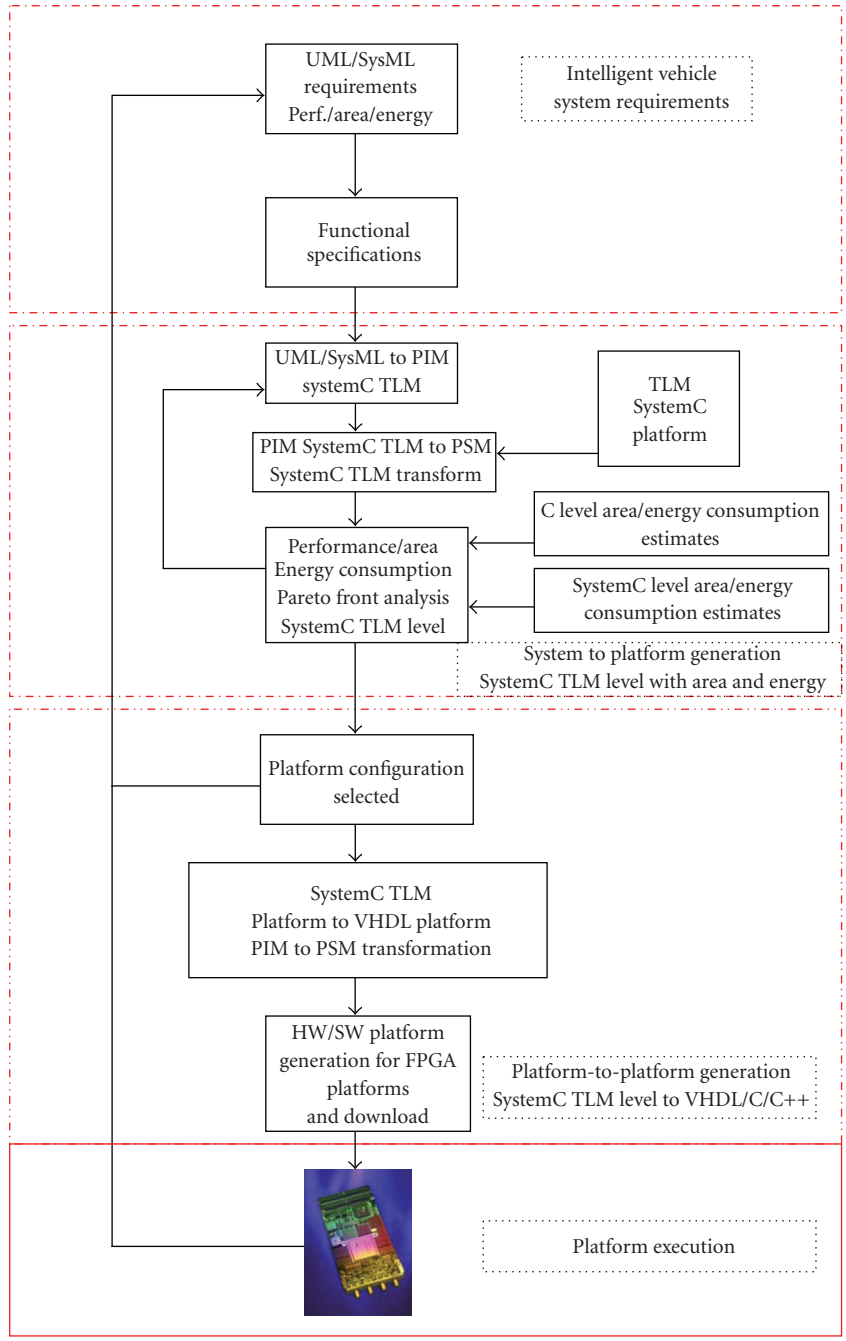


Figure 13: UML/SysML/TLM SystemC platform-based design methodology for intelligent vehicles.

One key step of the Accord/UML methodology is the model transformation from a UML design model platform independent to a UML design model platform specific. This is mainly accomplished through a transformation with a worst-case execution time (WCET) valuation. This PSM could be improved by iterating through a UML performance analysis model which would again influence the transformation. This performance analysis model could be conducted using SystemC TLM platform model and include additional analysis with area and energy consumption as we did in the previous section. The objectives of the ProMARTE working group is to define a UML profile for modeling and analysis of real-time and embedded systems (MARTE) that answers to the RFP for MARTE [17]. These examples of UML-based design methodologies of embedded real-time systems suggest that UML and platform SystemC TLM design methodologies may be combined for intelligent vehicles design. In this regard, the autosar organization have released its UML profile v1.0.1 as a metamodel to describe the system, software, and hardware of an automobile [37]. This profile is expected

to be used as well for intelligent vehicles design. However, translation from UML/SysML to SystemC have only recently been tackled. Work has been conducted on the description of executable platforms at the UML-level as well as the translation of UML-based application descriptions to SystemC [27]. However, this work is far from getting down to a SystemC level of the platform we used in this study. In [25] they present a UML2.0 profile of SystemC language exploiting MDA capabilities. No significant example of the methodologies is shown. In [23] a bi-directional UML-SystemC translation tool called UMLSC is described. According to the authors more work remains to be done to extend UML to make it better suited for hardware specification and improve the translation tool. In [26] translation from UML to SystemC for stream processing applications is presented. This work allows the translation of a stream processor, however, not a full-fledged processor. It is an implementation of the abstract model in UML 2.0. A very recent significant example of translation is provided in [38] using network on chip. However, all the works mentioned so far did not use (1) SystemC TLM platform-based design and (2) area and energy consumption of platform configurations.

We propose a UML/SysML to SystemC design flow methodology exclusively targeting platforms, that is, we are not interested to directly translate UML to hardware level nor we are interested to translate UML to SystemC. In a SystemC TLM, platform modules have SystemC interface but can be written with C. So UML structural parts are met with structural part of SystemC TLM platform while internal behavior of modules provided in C. This requires for area/energy consumption tradeoffs C-based synthesis and energy estimate tools such as [39]. Our proposed flow transforms UML to SystemC TLM platforms with design space exploration at SystemC TLM level for timing, area, and energy (Figure 13).

In a combined UML-SystemC design methodology, UML is used to capture the static system architecture and the high-level dynamic behavior while SystemC is used for design implementation.

The transformation of the SystemC TLM to VHDL platform is straightforward and will be described in a future publication [40]. The use of FPGA platforms allows faster prototyping especially if one considers actual intelligent vehicle driving conditions [41, 42]. This overall design flow will be the focus of future work [43].

8. CONCLUSIONS

In this paper, we have proposed a platform-based SystemC TLM system-level design methodology for embedded applications. This methodology emphasizes on components-based software design and high-level (TLM) modeling and simulation. Our proposed design flow facilitates the process of system design by higher leveling hardware modeling and behavioral synthesis of hardware modules. We have showed that using the methodology, complex image processing applications can be synthesized within very short time hence increasing the productivity and reducing overall time to market for an electronic system. The introduction of Autosar UML profile suggests the use of a combination of UML based and SystemC TLM platform-based joint methodologies. Microcontrollers customized with our approach could benefit from higher-level specification. Future work will extend to raising the design methodology abstraction level to combined UML/SysML/TLM SystemC platform design flow.

REFERENCES

[1] T. Bucher, C. Curio, J. Edelbrunner, et al., "Image processing and behavior planning for intelligent vehicles," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 1, pp. 62–75, 2003.

[2] L. Li, J. Song, F.-Y. Wang, W. Niehsen, and N.-N. Zheng, "IVS 05: new developments and research trends for intelligent vehicles," *IEEE Intelligent Systems*, vol. 20, no. 4, pp. 10–14, 2005.

[3] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 20–37, 2006.

[4] A. P. Girard, S. Spry, and J. K. Hedrick, "Intelligent cruise-control applications: real-time, embedded hybrid control software," *IEEE Robotics & Automation Magazine*, vol. 12, no. 1, pp. 22–28, 2005.

[5] W. van der Mark and D. M. Gavrila, "Real-time dense stereo for intelligent vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 38–50, 2006.

[6] K. D. Müller-Glaser, G. Frick, E. Sax, and M. Kühl, "Multiparadigm modeling in embedded systems design," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 2, pp. 279–292, 2004.

[7] Freescale Semiconductors, http://www.freescale.com/.

[8] F. Ghenassia, Ed., *Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems*, Springer, New York, NY, USA, 1st edition, 2006.

[9] L. Cai and D. Gajski, "Transaction level modeling: an overview," in *Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS '03)*, pp. 19–24, Newport Beach, Calif, USA, October 2003.

[10] N. Calazans, E. Moreno, F. Hessel, V. Rosa, F. Moraes, and E. Carara, "From VHDL register transfer level to SystemC transaction level modeling: a comparative case study," in *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design (SBCCI '03)*, pp. 355–360, Sao Paulo, Brazil, September 2003.

[11] Synopsys, "Behavioral Compiler User Guide," Version 2003.10, 2003.

[12] Agility, http://www.celoxica.com/products/agility/default.asp.

[13] O. Capdevielle and P. Dalle, "Image processing chain construction by interactive goal specification," in *Proceedings of the 1st IEEE International Conference Image Processing (ICIP '94)*, vol. 3, pp. 816–820, Austin, Tex, USA, November 1994.

[14] Y. Abchiche, P. Dalle, and Y. Magnien, "Adaptative Concept Building by Image Processing Entity Structuration," Institut de Recherche en Informatique de Toulouse IRIT, Université Paul Sabatier.

[15] R. B. France, S. Ghosh, T. Dinh-Trong, and A. Solberg, "Model-driven development using UML 2.0: promises and pitfalls," *Computer*, vol. 39, no. 2, pp. 59–66, 2006.

[16] Accord/UML, http://www-list.cea.fr/labos/fr/LLSP/accord_uml/AccordUML_presentation.htm.

[17] ProMARTE, http://www.promarte.org/.

[18] Protes project, http://www.carroll-research.org/.

[19] C. Jouvray, S. Gérard, F. Terrier, S. Bouaziz, and R. Reynaud, "UML methodology for smart transducer integration in real-time embedded systems," in *Proceedings of IEEE Intelligent Vehicles Symposium*, pp. 688–693, Las Vegas, Nev, USA, June 2005.

[20] S. Gérard, C. Mraidha, F. Terrier, and B. Baudry, "A UML-based concept for high concurrency: the real-time object," in *Proceedings of the 7th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC '04)*, pp. 64–67, Vienna, Austria, May 2004.

[21] H. Saíedian and S. Raguraman, "Using UML-based rate monotonic analysis to predict schedulability," *Computer*, vol. 37, no. 10, pp. 56–63, 2004.

[22] J.-L. Dekeyser, P. Boulet, P. Marquet, and S. Meftali, "Model driven engineering for SoC co-design," in *Proceedings of the 3rd International IEEE Northeast Workshop on Circuits and Systems Conference (NEWCAS '05)*, pp. 21–25, Quebec City, Canada, June 2005.

[23] C. Xi, L. J. Hua, Z. ZuCheng, and S. YaoHui, "Modeling SystemC design in UML and automatic code generation," in *Proceedings of the 11th Asia and South Pacific Design Automation Conference (ASP-DAC '05)*, vol. 2, pp. 932–935, Yokohama, Japan, January 2005.

[24] J Kreku, M. Eteläperä, and J.-P. Soininen, "Exploitation oF UML 2.0—based platform service model and systemC workload simulation in MPEG-4 partitioning," in *Proceedings of the International Symposium on System-on-Chip (SOC '05)*, pp. 167–170, Tampere, Finland, November 2005.

[25] E. Riccobene, P. Scandurra, A. Rosti, and S. Bocchio, "A SoC design methodology involving a UML 2.0 profile for SystemC," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE '05)*, vol. 2, pp. 704–709, Munich, Germany, March 2005.

[26] Y. Zhu, Z. Sun, W.-F. Wong, and A. Maxiaguine, "Using UML 2.0 for system level design of real time SoC platforms for stream processing," in *Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pp. 154–159, Hong Kong, August 2005.

[27] K. D. Nguyen, Z. Sun, P. S. Thiagarajan, and W.-F. Wong, "Model-driven SoC design via executable UML to SystemC," in *Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS '04)*, pp. 459–468, Lisbon, Portugal, December 2004.

[28] IBM CoreConnect, http://www.ibm.com/.

[29] *IEEE 1666 Standard SystemC Language Reference Manual*, http://standards.ieee.org/getieee/1666/download/1666-2005.pdf.

[30] MDA Guide Version 1.0.1 June 2003, OMG.

[31] S. Chtourou and O. Hammami, "SystemC space exploration of behavioral synthesis options on area, performance and power consumption," in *Proceedings of the 17th International Conference on Microelectronics (ICM '05)*, pp. 67–71, Islamabad, Pakistan, December 2005.

[32] IBM PEK v1.0, http://www-128.ibm.com/developerworks/power/library/pa-pek/.

[33] "RiscWatch Debuggers User Guide," 15th edition, IBM Number: 13H6964 000011, May 2003.

[34] OSCI SystemC Transaction-Level Modeling Working Group (TLMWG), http://www.systemc.org/web/sitedocs/technical_working_groups.html.

[35] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Cambridge, UK, 1989.

[36] R. Ben Mouhoub and O. Hammami, "MOCDEX: multiprocessor on chip multiobjective design space exploration with direct execution," *EURASIP Journal of Embedded Systems*, vol. 2006, Article ID 54074, 14 pages, 2006.

[37] UML Profile for Autosar v1.0.1, http://www.autosar.org/.

[38] E. Riccobene, P. Scandurra, A. Rosti, and S. Bocchio, "A model-driven design environment for embedded systems," in *Proceedings of the 43rd ACM/IEEE Design Automation Conference (DAC '06)*, pp. 915–918, San Francisco, Calif, USA, July 2006.

[39] Orinoco Dale, http://www.chipvision.com/company/index.php.

[40] O. Hammami and Z. Wang, "Automatic PIM to PSM Translation," submitted for publication.

[41] S. Saponara, E. Petri, M. Tonarelli, I. del Corona, and L. Fanucci, "FPGA-based networking systems for high data-rate and reliable in-vehicle communications," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE '07)*, pp. 1–6, Nice, France, April 2007.

[42] C. Claus, J. Zeppenfeld, F. Müller, and W. Stechele, "Using partial-run-time reconfigurable hardware to accelerate video processing in driver assistance system," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE '07)*, pp. 1–6, Nice, France, April 2007.

[43] O. Hammami, "Automatic Design Space Exploration of Automotive Electronics: The Case of AUTOSAR," submitted for publication.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 46972, 8 pages
doi:10.1155/2007/46972

Research Article

Lane Tracking with Omnidirectional Cameras: Algorithms and Evaluation

Shinko Yuanhsien Cheng and Mohan Manubhai Trivedi

Laboratory for Intelligent and Safe Automobiles (LISA), University of California, San Diego, La Jolla, CA 92093-0434, USA

Received 13 November 2006; Accepted 29 May 2007

Recommended by Paolo Lombardi

With a panoramic view of the scene, a single omnidirectional camera can monitor the 360-degree surround of the vehicle or monitor the interior and exterior of the vehicle at the same time. We investigate problems associated with integrating driver assistance functionalities that have been designed for rectilinear cameras with a single omnidirectional camera instead. Specifically, omnidirectional cameras have been shown effective in determining head gaze orientation from within a vehicle. We examine the issues involved in integrating lane tracking functions using the same omnidirectional camera, which provide a view of both the driver and the road ahead of the vehicle. We present analysis on the impact of the omnidirectional camera's reduced image resolution on lane tracking accuracy, as a consequence of gaining the expansive view. And to do so, we present Omni-VioLET, a modified implementation of the vision-based lane estimation and tracking system (VioLET), and conduct a systematic performance evaluation of both lane-trackers operating on monocular rectilinear images and omnidirectional images. We are able to show a performance comparison of the lane tracking from Omni-VioLET and Recti-VioLET with ground truth using images captured along the same freeway road in a specified course. The results are surprising: with 1/10th the number of pixels representing the same space and about 1/3rd the horizontal image resolution as a rectilinear image of the same road, the omnidirectional camera implementation results in only three times the amount the mean absolute error in tracking the left lane boundary position.

1. INTRODUCTION: OMNIDIRECTIONAL CAMERA FOR LOOKING IN AND LOOKING OUT

Omnidirectional camera's main feature is its ability to capture an image of the scene 360 degrees around the camera. It has the potential to monitor many things in the scene at one time, illustrated in Figure 1. In an intelligent driver assistance system, this means a single sensor has the potential to monitor the front, rear, and side views of vehicle and even inside the vehicle simultaneously. This eliminates the need for multiple cameras and possibly complex calibration maintenance algorithms between multiple cameras. Due to the fact that reducing redundancy is one of the main goals in embedded systems, combining multiple functionalities into a single, simpler sensor reduces the cost associated with maintaining individual sensors for each sensing function.

There is also evidence that driver behavior should be an integral part of any effective driver assistance system [1], driving the need for a suite of sensors that extracts cues from both outside and inside the vehicle. With these motivations, we investigate problems associated with integrating driver assistance functionalities that have been designed for multiple rectilinear cameras on a single omnidirectional camera instead. For this paper, we examine issues involved in and suggest solutions to integrate lane tracking functions using the omnidirectional camera in this multifunction context.

Huang et al. [2] demonstrated that an omnidirectional camera can be used to estimate driver head pose to generate the driver's view of the road. Knowledge of the driver's gaze direction has of course many uses beyond driver-view synthesis. The driver head motion is the critical component that added one second of warning time to a lane departure warning system in [3]. This human-centered driver support system uses vehicle lane position from cameras looking out of the vehicle, vehicle speed, steering, yaw rate from the vehicle itself, as well as head motion from a camera looking in the vehicle to make predictions of when drivers make lane-change maneuvers. Estimates of driver head movement also improved intersection turn maneuver predictions [4]. There, each of these predictions can potentially describe the driver's

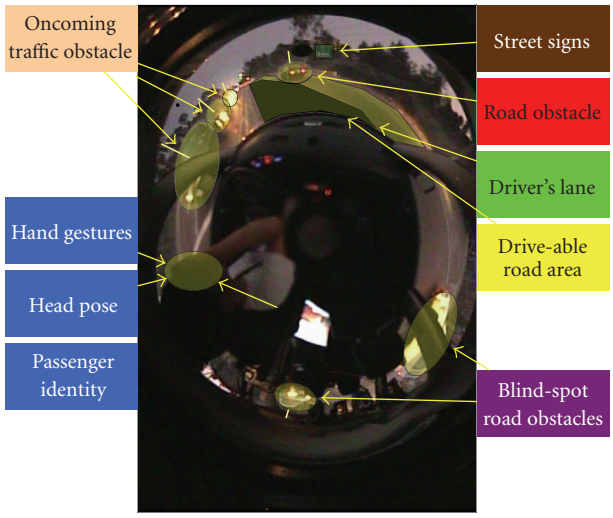


FIGURE 1: This figure shows an illustrative image captured by omnidirectional cameras and the panoramic field of view with a potential for holistic visual context analysis.

awareness of the driving situation. For example, given an obstacle in the vehicle's path and continued driver preparatory movements to perform the maneuver, the assistance system can then conclude that the driver is unaware of the danger and take appropriate action. Observing the driver also has applications in driver identity verification, vigilance monitoring, along with intention and awareness monitoring. It is clear that driver bodily movements are very significant cues in determining several driver behaviors. It is also clear that visual methods using omnidirectional cameras of extracting some of this driver information have been shown to be an effective approach.

In addition to head pose, lane tracking is also an important component in many intelligent driver assistance systems. Lane tracking has utility in lane departure warning, overspeed warnings for sharp curves, ahead vehicle proximity estimation for adaptive cruise control, collision avoidance, obstacle detection, and many others [5]. Just as observing drivers will enhance driving safety, lane tracking is an integral part in the same task.

This naturally leads to the following question: can efficiency be improved by utilizing a single omnidirectional camera rather than two rectilinear cameras to perform these same functions of observing the driver and the road? Since the only difference between rectlinear and omnidirectional images is the projection function, that is, the manner in which 3D points in space are projected onto the 2D image plane, the answer should be yes. The question is to what extent. We attempt to answer these questions by comparing the results between VioLET, a vision-based lane estimation and tracking system [6] operating on rectlinear images, and Omni-VioLET, a modified version operating on omnidirectional images. We compare the tracking results from both systems with ground truth. Our contributions can be listed in the following:

(1) we introduce a lane tracking system using an omnidirectional camera that utilizes a well-tested, robust lane tracking algorithm called Omni-VioLET. The omnidirectional camera also captures a view of the driver at the same time for driver monitoring applications;
(2) we discuss and undertake a systematic performance comparison of the lane tracking systems using a rectilinear camera and omnidirectional camera of the same road course with ground truth.

2. RELATED RESEARCH IN VISION-BASED LANE TRACKING

Most previously proposed vision-based lane tracking systems follow a processing structure consisting of these three steps: (1) extract road features from sensors, (2) suppress outliers from the extracted road features, and (3) estimate and track lane model parameters.

There are several notable lane tracking approaches using rectilinear cameras. The one by Bertozzi and Broggi [7] proposes to use stereo rectilinear camera for lane detection combined with obstacle detection. They employ a flat-plane transformation of the image onto a birds-eye view of the road, followed by a series of morphological filters to locate the lane markings. A recent contribution by Nedevschi et al. [8] augments the usual flat-plane assumption of the road to a 3D model based on clothoids and vehicle roll angles. That system relies on depth maps calculated from a stereo camera and edges in images to infer the orientation of the 3D road model. A detailed survey of lane position and tracking techniques using monocular rectilinear cameras is presented in [6].

Ishikawa et al. [9] proposes an approach using an omnidirectional camera to track lane position in an autonomous vehicle application. The approach first transforms the omni-image to flat-plane, followed by Hough transform to search for the left and right lane marking with a lane separation prior. With an autonomous vehicle application in mind, the scene captured by this omnidirectional camera saw lane markers ahead and behind the vehicle, both aiding in determining the vehicle's position in the lane and lane width. Furthermore, lines perpendicular to the vehicle could also be detected with this system since the sides are also monitored as well. This work demonstrates that effective lane tracking can be achieved to some extent with omnidirectional images.

Because the Ishikawa approach was designed in the context of autonomous vehicles, the operating environment, although the setting was outdoors, was idealized with solid white lines for lane markings with constant lane widths. The central component of the VioLET system is the use of steerable filters, which has been shown to be highly effective in extracting circular reflectors as well as line segments, prevalent lane markings in actual California freeways. Furthermore, the algorithm lacked a mechanism to incorporate temporal history of last observed lane locations with the current estimate. We will use the Kalman filtering framework to update interesting statistics of lane position, lane width, and so forth using lane position measurements from the current as well as previous moments in time. Lastly, we are interested in

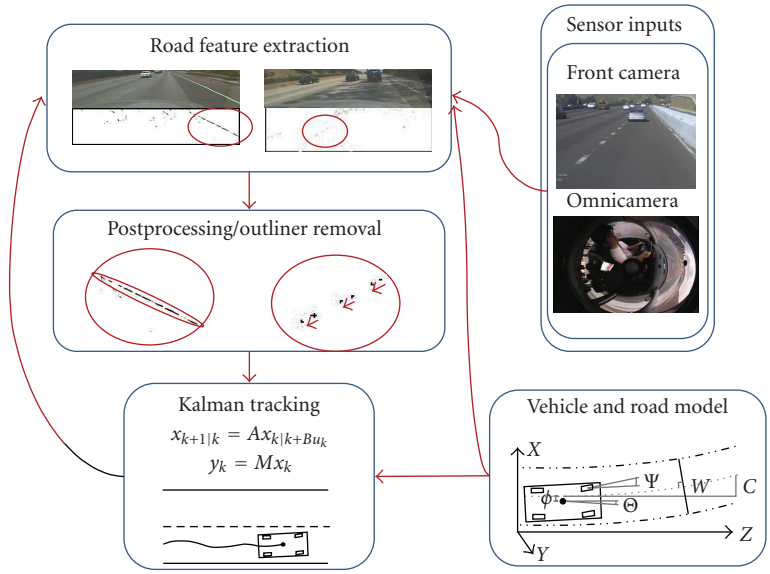


FIGURE 2: This illustrates the system flow diagram for VioLET, a driver-assistance focused lane position estimation and tracking system.

examining the extent to which lane tracking can be accurate by using only the view of the road ahead of the vehicle, deferring the other areas of the omnidirectional image for other applications that monitor the vehicle interior.

3. OMNI-VIOLET LANE TRACKER

In this section, we describe the modifications to the VioLET system for operation on omnidirectional images. An omnidirectional camera is positioned just below and behind the rear view mirror. From this vantage point, both the road ahead of the vehicle as well as the front passengers can be clearly seen. With this image, left lane marker position, right lane marker position, vehicle position within the lane, and lane width are estimated from the image of the road ahead. Figure 2 shows a block diagram of the Omni-VioLET system for use in this camera comparison.

The VioLET system operates on a flat-plane transformed image, also referred to as a birds-eye view of the road. This is generated from the original image captured from the camera, with knowledge of the camera's intrinsic and extrinsic parameters and orientation of the camera with respect to the ground. The intrinsic and extrinsic parameters of the camera describe the many-to-one relationship between the 3D points in the scene in real-world length units and its projected 2D location on the image in image pixel coordinates. The planar road assumption allows us to construct a one-to-one relationship between 3D points on the road surface and its projected 2D location on the image in image pixel coordinates. This is one of the critical assumptions that allow lane tracking algorithms to provide usable estimates of lane location and vehicle position, and is also the assumption utilized in the VioLET system.

The model and calibration of rectilinear cameras are very well studied, and many of the results translate to omnidirectional cameras. We can draw direct analogs between the omnidirectional camera model and the rectilinear camera model, namely its intrinsic and extrinsic parameters. Tools for estimating the model parameters have been also recently made available [10]. Table 1 summarizes the transformations from a 3D point in the scene $\mathbf{P} = (x, y, z)$ to the projected image point on the image $\mathbf{u} = (u, v)$.

Utilizing the camera parameters for both rectilinear and omnidirectional cameras, a flat-plane image can be generated given knowledge of the world coordinate origin and the region on the road surface we wish to project the image onto. The world origin is set at the center of the vehicle on the road surface with the y-axis pointing forward and z-axis pointing upward. Examples of the flat-plane transformation are shown in Figure 3. Pixel locations of points in the flat-plane image and the actual locations on the road are related by a scale factor and offset.

The next step is extracting road features by applying steerable filters based on the second derivatives of a two-dimensional Gaussian density function. Two types of road features are extracted: circular reflectors ("Bots dot") and lines. The circular reflectors are not directional so the filter responses are equally high in both the horizontal and vertical directions. The lines are highly directional and yield high responses for filters oriented along its length. The filtered images such as the one shown in Figure 3 are then thresholded and undergo connected component analysis to isolate the most likely candidate road features.

The locations of the road features are averaged to find the new measurement of the lane boundary location. The average is weighted on its proximity to the last estimated location of the lane boundary. The measurement for the other lane boundary is made the same way. The last estimated lane boundary location is estimated using a Kalman filter using lane boundary locations as observations, and vehicle position

Table 1: Projective transformation for rectilinear and omnidirectional cameras.

	Rectilinear camera model	Omnidirectional catadioptric camera model
World to camera coordinates	$\mathbf{P}_c = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \mathbf{RP} + \mathbf{t}$	$\mathbf{P}_c = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \mathbf{RP} + \mathbf{t}$
Camera to homogeneous camera plane (normalized camera) coordinates	$\mathbf{p}_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} X_c/Z_c \\ Y_c/Z_c \end{pmatrix}$	—
Undistorted to distorted camera plane coordinates	$\mathbf{p}_d = \begin{pmatrix} x_d \\ y_d \end{pmatrix} = \lambda \mathbf{p}_n + dx,$ $\lambda = 1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6,$ $dx = \begin{pmatrix} 2\rho_1 xy + \rho_2(r^2 + 2x^2) \\ \rho_1(r^2 + 2y^2) + 2\rho_2 xy \end{pmatrix},$ $r^2 = x_n^2 + y_n^2$	$\begin{pmatrix} x_d \\ y_d \\ f(x_d, y_d) \end{pmatrix} = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix},$ $f(x_d, y_d) = a_0 + a_1\rho + a_2\rho^2 + a_3\rho^3 + a_4\rho^4,$ $\rho^2 = x_d^2 + y_d^2$
Distorted camera plane to image coordinates	$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & \alpha f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix}$	$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & \alpha f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix}$

in the lane, left lane boundary location, right lane boundary location, and lane width as hidden states. For more details on these steps of extracting road features and tracking these features using the Kalman filter we refer the reader to the original paper [6]. The original implementation also takes advantage of vehicle speed, yaw-rate, and road texture to estimate road curvature and refine the estimates of the lane model. We chose to omit those measurements in our implementation, and to focus on estimating lane boundary position and vehicle position in the lane, for which ground truth can be collected, to illustrate the point that omnidirectional cameras have the potential to be used for lane tracking.

Altogether, the VioLET system assumes a planar road surface model, knowledge of camera parameters, road feature extraction using steerable filters, and the lane model parameters are tracked with a Kalman filter using road feature location measurements as observations. The outputs are lane boundary positions, lane width, and vehicle position in the lane.

4. EXPERIMENTAL PERFORMANCE EVALUATION AND COMPARISON

Omni-VioLET lane tracking system is evaluated with video data collected from three cameras in a specially equipped Laboratory for Intelligent and Safe Automobiles-Passat (LISA-P) test vehicle. A rectilinear camera placed on the roof of the car and the omnidirectional camera hung over the rear-view mirror capture the road ahead of the vehicle. A third camera is used to collect ground truth. All cameras are Hitachi DP-20A color NTSC cameras, and 720×480 RGB images are captured via a Firewire DV capture box to a PC at 29.97 Hz. The vehicle was driven along actual freeways and video was collected synchronously along with various vehicle parameters. Details of the test-bed can be found in [4].

For evaluation, we collected ground truth lane position data using the third calibrated camera. A flat-plane image from this camera is also generated such that the horizontal position of the transformed image represents the distance from the vehicle. A grid of points corresponding to known physical locations of the ground is used to adjust the orientation and position of the side camera. Figure 4 shows the result of manually correcting the pose of the camera, and thus the grid of points in the image from the side camera. With this grid of points and its associated location in the image, a flat-plane image is generated as shown in the same figure. From the flat-plane image, lane positions are manually annotated to generate ground truth. This ground truth is compared against lane tracking results of both the rectilinear and omnidirectional VioLET systems.

The lane tracking performance is analyzed on data collected from a test vehicle driven at dusk on a multilane freeway at freeway speeds for several minutes. It was shown that dusk was one of the many times during a 24-hour period when VioLET performed most optimally, because of the light traffic conditions [6]. This allows a comparison of the omni-image based lane tracking with the most optimal rectilinear image-based lane tracking results. The image resolution of the flat-plane transformed image derived from the omnicamera was set at 100×100, while the one derived from the rectilinear-image was set at 694×2000. For the omni-case, that resolution was chosen because the lateral resolution of the road is approximately 100 pixels wide. For the rectilinear case, the lateral resolution is slightly shorter than the

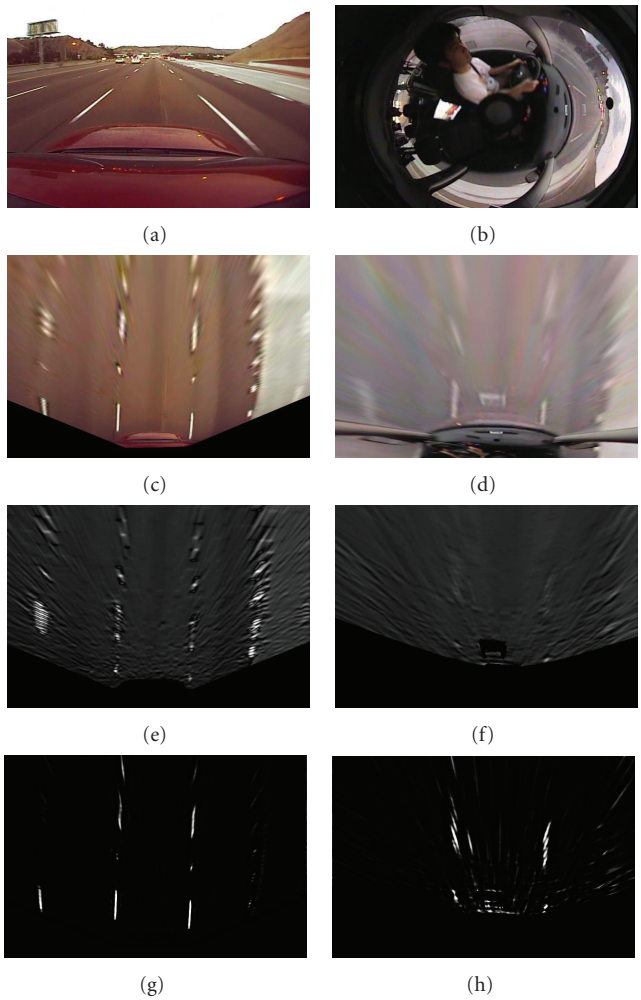


Figure 3: This illustration shows the original images, the flat-plane transformed images, and two filter response images from the flat-plane transformed images for circular reflectors and lane line markers.

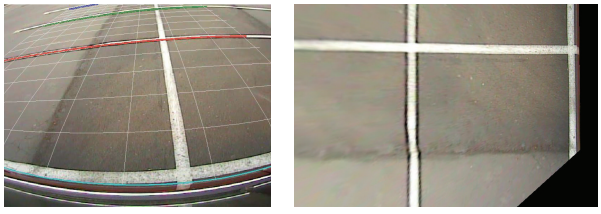


Figure 4: This illustrates the alignment of the ground-grid to the perceived ground in the ground-truth camera, and the resulting flat-plane transformed image. Radial distortion is taken into account as can be seen by the bowed lines.

width of the 720 pixel-wide horizontal resolution of the image. The vertical resolution was chosen by making road features square, up to 100 feet forward of the camera.

In aligning lane tracking results from the two systems with ground-truth, the ground-truth is kept unchanged for reference. The lane tracking estimates were manually scaled and offset to compensate for errors in camera calibration, camera placement, and error in lane-width estimation. This alignment consists of three operations on the lateral lane

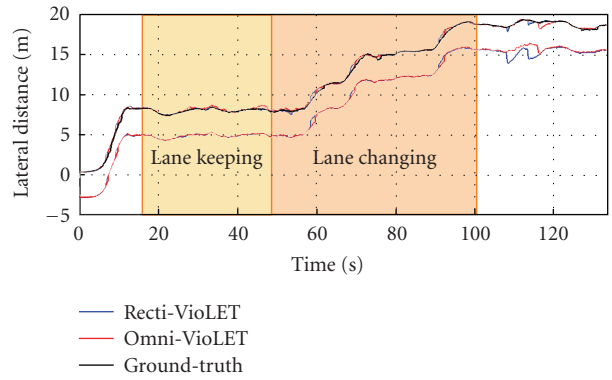


Figure 5: This figure depicts the progression of the lane-boundary estimates over time as found by the (full-resolution) rectilinear camera-based lane tracking, omnidirectional camera-based lane tracking and ground-truth. The shaded regions demarcate the lane-keeping and lane-changing segments of the road.

boundary and lane position estimates: (1) global offset, (2) global scale, and (3) unwrapping amount. The global offset puts all 3 cameras on the same lateral position of the car. The global scale changes the scale of the estimates which result from errors in camera calibration. Unwrapping amount is specified to compensate for errors in lane-width estimates, which impact left-lane position estimates when the left-lane location is more than half a lane-width away. These alignment parameters are set once for the entire experiment. The resulting performance characteristics are shown as error in centimeters.

It is important to note that these error measurements are subject to errors in the ground-truth-camera calibration accuracy. Indeed, an estimate that closely aligns with ground-truth can be claimed to be only that. In this particular case, a scaling error in ground-truth calibration could result in a scaling error in the lane tracking measurements. This would be the case for any sort of vision-based measurement system using another vision-based system to generate ground-truth. For the ground-truth camera used in our experiments, we approximate a deviation of ± 5 cm of the model to the actual location in the world by translating the model ground plane and visually inspecting the alignment with two 2.25 m parking slots; see Figure 4. With that said, we can however make conclusions about the relative accuracies between two lane-tracking systems, which we present next.

Several frames from the lane tracking experiment are shown in Figures 8 and 9. The top set of images show the even and odd fields of the digitized NTSC image, while the bottom set of images show the lane tracking result showing the lane feature detection in boxes and left and right lane boundary estimates in vertical lines. The original image was separated because each image was captured at 1/60 second from each other, which translates to images captured at positions 50 cm apart with the vehicle traveling 30 m/s (65 mph). Figure 5 shows results of estimates over time of left and right lane boundaries from both systems against left-lane boundary ground-truth during two segments of one test run. Two segments of lane-following and lane-changing manuevers are

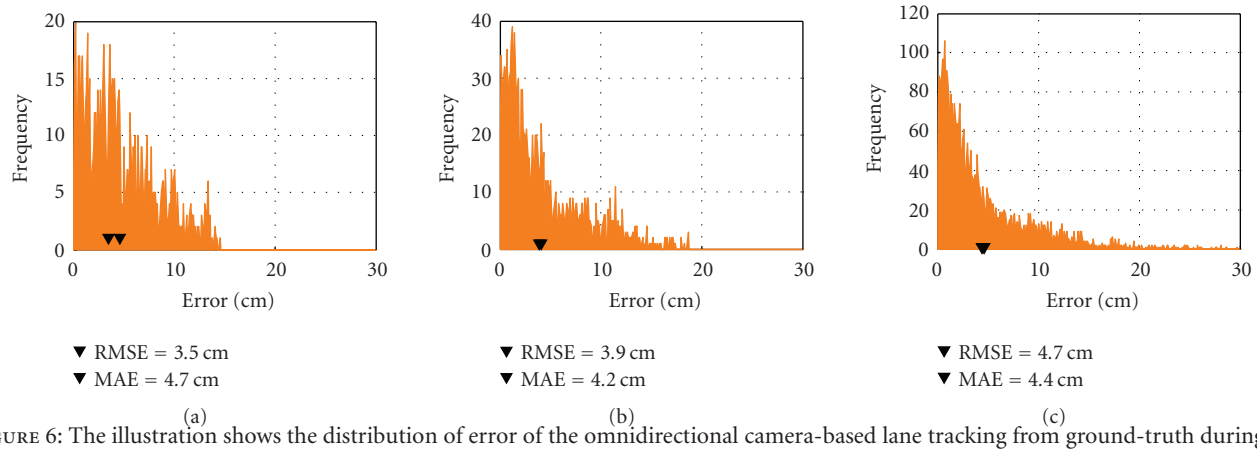


Figure 6: The illustration shows the distribution of error of the omnidirectional camera-based lane tracking from ground-truth during (a) the lane keeping segment, (b) the lane changing segment, and (c) the overall ground-truthed test sequence.

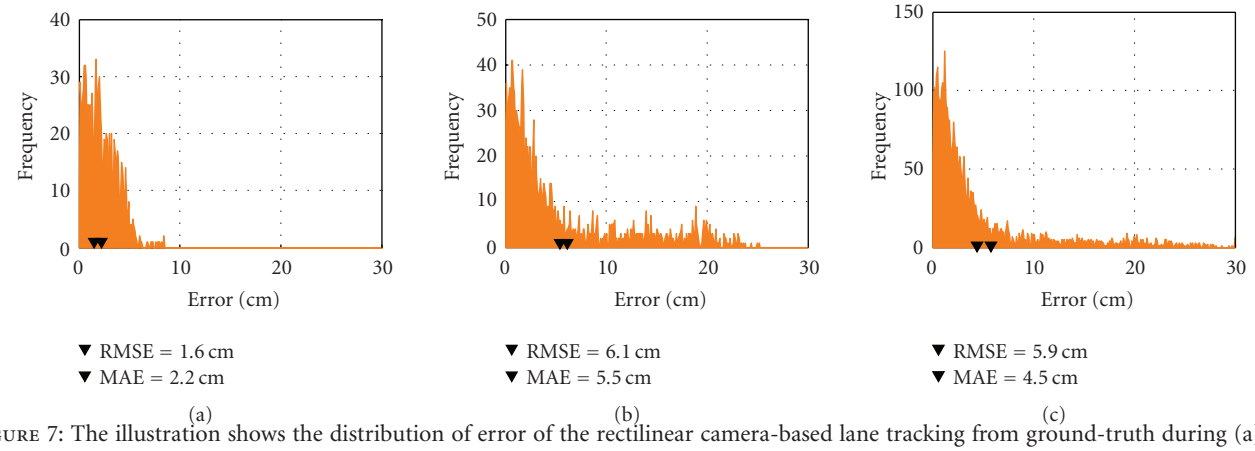


Figure 7: The illustration shows the distribution of error of the rectilinear camera-based lane tracking from ground-truth during (a) the lane keeping segment, (b) the lane changing segment, and (c) the overall ground-truthed test sequence.

analyzed separately. The error distributions are shown in Figures 6 and 7. In these two segments, we can see the strength of the rectilinear camera at approximately 1.5 cm mean absolute error (MAE) from ground-truth as compared to omnidirectional camera-based lane tracking performance of 4.2 cm MAE. During a lane change maneuver, the distinction is reversed. The two systems performed with 5.5 cm and 4.2 cm MAE, respectively; root mean square (RMS) error shows the same relationship. Over the entire sequence, the errors were 4.5 cm and 4.4 cm MAE and 5.9 cm and 4.7 cm RMS error for Recti-VioLET and Omni-VioLET, respectively. Errors are summarized in Table 2.

During the lane-change segment, a significant source of error in both systems is the lack of a vehicle heading estimate. Rather, the lanes are assumed to be parallel with the vehicle as can be seen from the lane position estimate overlaid on the flat-plane images in Figures 8 and 9, which is of course not always the case. For that reason, we gauge the relative accuracy between Recti-VioLET and Omni-VioLET along the lane following sequence. Despite the curves in the road, this segment could show that the diminished omni-image resolution resulted in a mere 3 times more MAE.

Additional runs of Recti-VioLET were conducted on half, quarter, and eighth resolution rectilinear images. The size of the flat-plane transformation is maintained at 694×2000. At a quarter of the original resolution (180×120), the lateral road resolution in the rectilinear image is approximately equal to that of the omni-image. The resulting accuracies are summarized in Table 2. Remarkable is their similar performance even at the lowest resolution. To determine lane boundary locations, several flat-plane marking candidates are selected and its weighted average along the lateral position serves as the lane boundary observation to the Kalman tracker. This averaging could explain the resulting subpixel accurate estimates. Only at the very lowest resolution (eighth) input image was the algorithm unable to maintain tracking of lanes across several lane changes. However, eighth-resolution lane tracking in lane following situation yielded similar accuracies as lane tracking at the other resolutions.

Resolution appears to play only a partial role in influencing accuracy of lane tracking. The lane-markings detection performance appears to suffer increased number of misdetections at low resolution. This error appears in the accuracy measurements in the form of lost tracks through lane changes. Accuracy seems to not be affected by the reduced resolution. At resolutions of 180×120, misdetections occurred infrequently enough to maintain tracking throughout

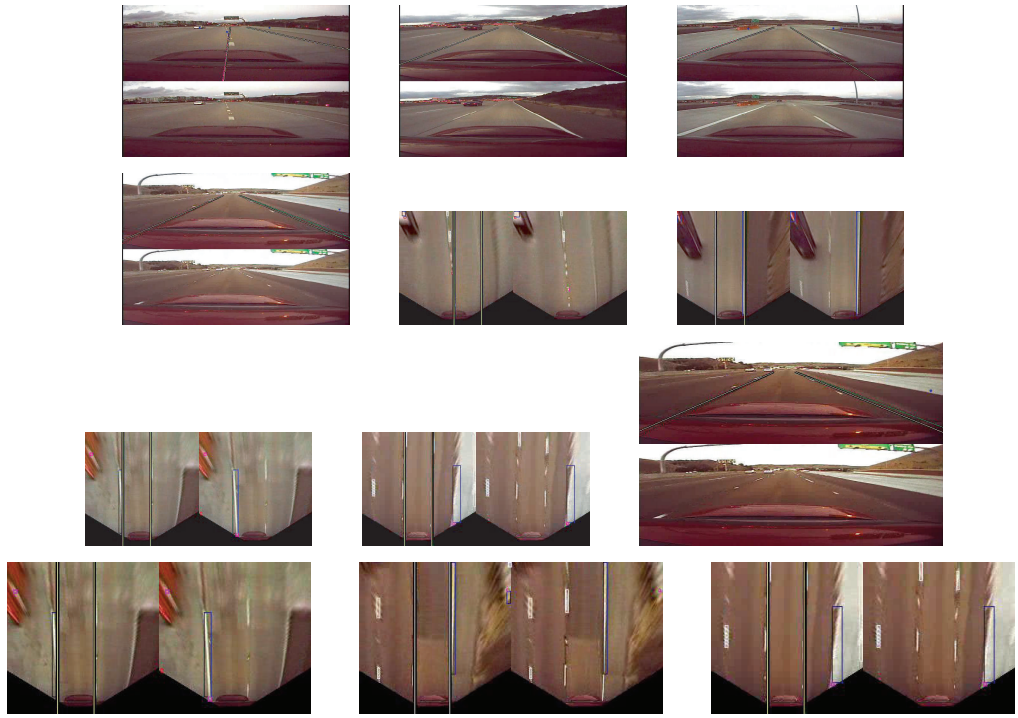


Figure 8: This illustrates the lane tracking results from rectilinear images. The top and middle images are the even and odd fields of the original NTSC image. The bottom two images are the flat-plane transformed images of the above two images. The two overlay lines depict the estimated left and right lane positions while the boxes represent detected road features.

Table 2: Lane tracking accuracy. All units are in cm.

	Lane following (RMSE/MAE)	Lane changing (RMSE/MAE)	Overall (RMSE/MAE)
Omni-VioLET	3.5/4.7	3.9/4.2	4.7/4.4
Recti-VioLET full-res. (720×480)	1.3/1.5	6.1/5.5	5.9/4.5
Recti-VioLET half-res. (360×240)	1.7/2.2	5.3/5.2	5.1/4.2
Recti-VioLET qtr-res. (180×120)	1.5/2.1	5.7/6.7	5.6/4.9
Recti-VioLET eighth-res. (90×60)	1.6/2.2	lost-track**	lost-track**

the test sequence. Lane-marking detection performance itself in terms of detection rate and false alarms at these various image resolutions and image types would give a better picture of the overall lane tracking performance; this is left for future work.

5. SUMMARY AND CONCLUDING REMARKS

We investigate problems associated with integrating driver assistance functionalities that have been designed for rectilinear cameras with a single omnidirectional camera instead. Specifically, omnidirectional cameras have been shown effective in determining head gaze orientation from within a car. We examined the issues involved in integrating lane-tracking functions using the same omnidirectional camera. Because the resolution is reduced and the image distorted to produce a 360-degree view of the scene through a catadioptric camera geometry as opposed to the traditional pin-hole camera geometry, the achievable accuracy of lane tracking is a question in need of an answer. To do so, we presented Omni-VioLET, a modified implementation of VioLET, and conducted a systematic performance evaluation of the vision-based lane estimation and tracking system operating on both monocular rectilinear images and omnidirectional images. We were able to show a performance comparison of the lane tracking from Omni-VioLET and Recti-VioLET with ground-truth using images captured along the same freeway. The results were surprising: with 1/10th the number of pixels representing the same space and about 1/3rd the horizontal image resolution as a rectilinear image of the same road, the omnidirectional camera implementation results in only twice the amount of the mean absolute error in tracking the left-lane boundary position.

Experimental tests showed that the input image resolution is not the sole factor affecting accuracy, but it does have an impact on lane marking detection and maintaining track. The nearly constant error for full, half, quarter, and eighth resolution input images implied that accuracy is not

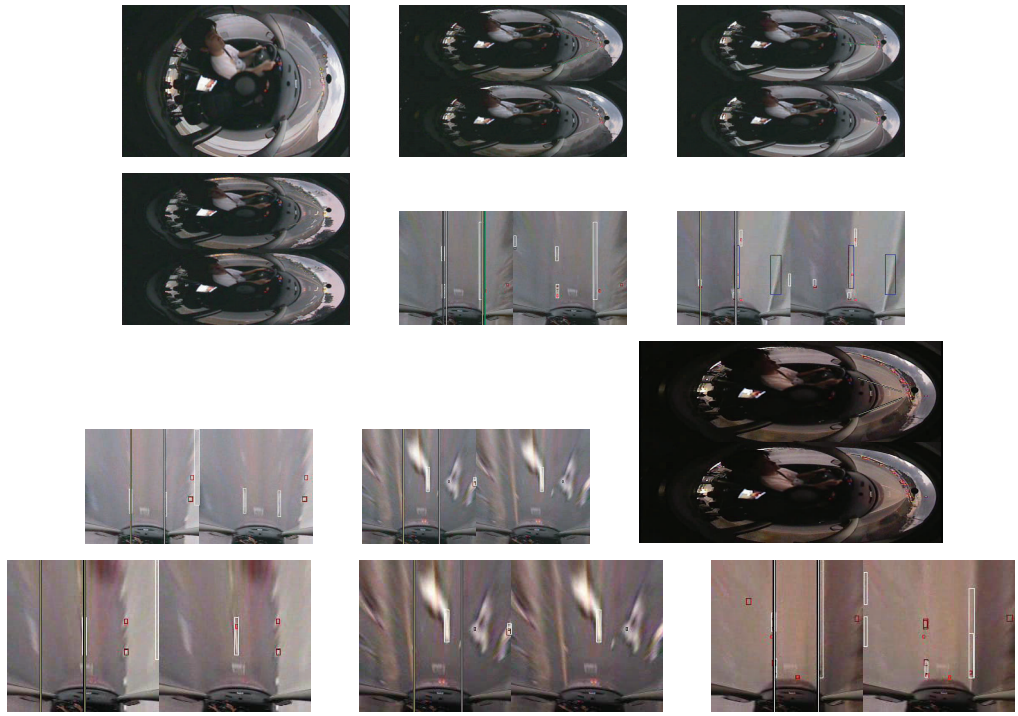


Figure 9: This illustrates the lane tracking results from omnidirectional images. The top and middle images are the even and odd fields of the original NTSC image. The bottom two images are the flat-plane transformed images of the above two images. The two overlay lines depict the estimated left and right lane positions while the boxes represent detected road features.

affected by resolution; we attributed the ability of the algorithm to maintain this accuracy to the temporal averaging from Kalman filtering and the large flat-plane image used for all Recti-VioLET tests. The experiments affirm the result that lane tracking with omnidirectional images is feasible, and is worth consideration when a system utilizing a minimal number of sensors is desired.

REFERENCES

[1] L. Petersson, L. Fletcher, A. Zelinsky, N. Barnes, and F. Arnell, "Towards safer roads by integration of road scene monitoring and vehicle control," *International Journal of Robotics Research*, vol. 25, no. 1, pp. 53–72, 2006.

[2] K. S. Huang, M. M. Trivedi, and T. Gandhi, "Driver's view and vehicle surround estimation using omnidirectional video stream," in *Proceedings of IEEE Intelligent Vehicles Symposium (IV '03)*, pp. 444–449, Columbus, Ohio, USA, June 2003.

[3] J. McCall, D. Wipf, M. M. Trivedi, and B. Rao, "Lane change intent analysis using robust operators and sparse Bayesian learning," to appear in *IEEE Transactions on Intelligent Transportation Systems*.

[4] S. Y. Cheng and M. M. Trivedi, "Turn-intent analysis using body pose for intelligent driver assistance," *IEEE Pervasive Computing*, vol. 5, no. 4, pp. 28–37, 2006.

[5] W. Enkelmann, "Video-based driver assistance—from basic functions to applications," *International Journal of Computer Vision*, vol. 45, no. 3, pp. 201–221, 2001.

[6] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 20–37, 2006.

[7] M. Bertozzi and A. Broggi, "GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 62–81, 1998.

[8] S. Nedevschi, R. Schmidt, T. Graf, et al., "3D lane detection system based on stereovision," in *Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems (ITSC '04)*, pp. 161–166, Washington, DC, USA, October 2004.

[9] K. Ishikawa, K. Kobayashi, and K. Watanabe, "A lane detection method for intelligent ground vehicle competition," in *SICE Annual Conference*, vol. 1, pp. 1086–1089, Fukui, Japan, August 2003.

[10] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A flexible technique for accurate omnidirectional camera calibration and structure from motion," in *Proceedings of the 4th IEEE International Conference on Computer Vision Systems (ICVS '06)*, p. 45, New York, NY, USA, January 2006.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 70256, 7 pages
doi:10.1155/2007/70256

Research Article

StereoBox: A Robust and Efficient Solution for Automotive Short-Range Obstacle Detection

Alberto Broggi, Paolo Medici, and Pier Paolo Porta

VisLab, Dipartimento Ingegreria Informazione, Università di Parma, 43100 Parma, Italy

Received 30 October 2006; Accepted 15 April 2007

Recommended by Gunasekaran S. Seetharaman

This paper presents a robust method for close-range obstacle detection with arbitrarily aligned stereo cameras. System calibration is performed by means of a dense grid to remove perspective and lens distortion after a direct mapping between image pixels and world points. Obstacle detection is based on the differences between left and right images after transformation phase and with a polar histogram, it is possible to detect vertical structures and to reject noise and small objects. Found objects' world coordinates are transmitted via CAN bus; the driver can also be warned through an audio interface. The proposed algorithm can be useful in different automotive applications, requiring real-time segmentation without any assumption on background. Experimental results proved the system to be robust in several envitonmental conditions. In particular, the system has been tested to investigate presence of obstacles in blind spot areas around heavy goods vehicles (HGVs) and has been mounted on three different prototypes at different heights.

1. INTRODUCTION

Problems concerning traffic mobility, safety, and energy consumption have become more serious in most developed countries in recent years. The endeavors to solve these problems have triggered the interest towards new fields of research and applications, such as automatic vehicle driving. New techniques are investigated for the entire or partial automation of driving tasks. A recently defined comprehensive and integrated system approach, referred to as intelligent transportation systems (ITSs), links the vehicle, the infrastructure, and the driver to make it possible to achieve more mobile and safer traffic conditions by using state-of-the-art electronic communication and computer-controlled technology.

In fact, ITS technologies may provide vehicles with different types and levels of "intelligence" to complement the driver. Information systems expand the driver's knowledge of routes and locations. Warning systems, such as collision-avoidance technologies, enhance the driver's ability to sense the surrounding environment. Driver assistance and automation technologies simulate the driver's sensor-motor system to operate a vehicle temporarily during emergencies or for prolonged periods.

Human-centered intelligent vehicles hold a major potential for industry. Since 1980, major car manufacturers and other firms have been developing computer-based in-vehicle navigation systems. Today, most developed/developing systems around the world have included more complex functions to help people to drive their vehicles safely and efficiently. New information and control technologies that make vehicles smarter are now arriving on the market either as optional equipment or as specialty after-market components. These technologies are being developed and marketed to increase driver safety, performance, and convenience. However, these disparate individual components have yet to be integrated to create a coherent intelligent vehicle that complements the human driver, fully considering his requirements, capabilities, and limitations.

In particular, concerning heavy goods vehicles (HGVs), many accidents involving trucks are related to the limited field of view of the driver: there are large blind spots all around the vehicle (see Figure 1). Some of these blind areas can be at least partly covered by additional mirrors. However, this is not always an optimal solution considering the aerodynamic effects and also the resulting complex driver interface.

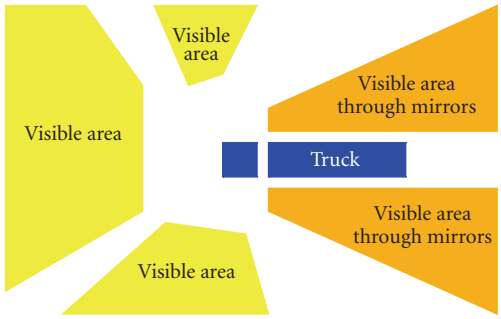


Figure 1: Field of view of a truck driver.

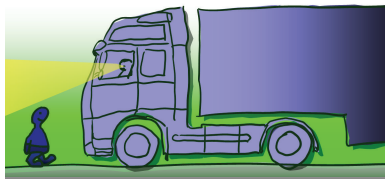


Figure 2: Typical dangerous situation.

Examples of traffic situations where the limited field of view can result in conflicts are

(i) starting from stationary at crosswalks or other places where a person or an object can be close in front of the vehicle,
(ii) lane change and turn situations to the passenger side,
(iii) situations with cross-road traffic sideways,
(iv) backup situations especially when ranging up to a loading dock.

This type of accidents accounts for approximately 10% of all accidents between trucks and unprotected road users and about 20% of all fatal accidents between trucks and unprotected road users.

The most effective single measure would be to improve the forward vision from HGV cabs so that an average size pedestrian could be seen even when standing right up against the front of the vehicle, see Figure 2. This would have been likely to save the lives of 12% of the pedestrians killed by HGVs. Changing the design of the front of a truck in this way is not an easy task. Similar benefits can be achieved by using sensors to detect the presence of a pedestrian or an obstacle and to warn the driver and also to prevent the vehicle from taking off when something is present in the forward blind spot: this is called start-inhibit, see Figure 3.

Embedded systems have to be compact and well designed for integration, but at the same time easy to use and to configure. In particular for a market ready product, there are some production aspects that get a central importance, for example, calibration procedure.

In all vision systems, calibration is one of the main topics, because it deeply affects algorithms performance; with our method the system is hardware-independent. In fact in case of accident or generic camera misalignment, the system can be restored after a recalibration (that could be done with automatic procedure with the vehicle parked in front of a grid). Even in case of cameras substitution for damage or commercial, reason system restoring would be done in the same way.

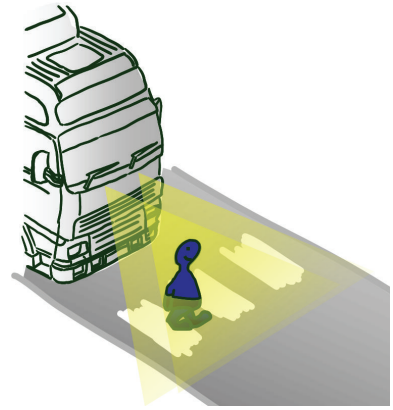


Figure 3: Start-inhibit protection system.

This is a strong point of StereoBox because it allows an easy installation and maintenance.

The system is composed of two cameras with sferic lenses to get a wide field of view, but introducing a strong distortion on images. They are placed in front of the truck and are arbitrary aligned, as will be discussed in Section 5. In particular, only the frontal driver blind spot area is framed by the cameras.

Two well-known approaches for stereo obstacle detection have been considered:

(i) the computation of the disparity of each pixel [1],
(ii) the use of stereo inverse perspective mapping [2].

An obstacle detection algorithm for offroad autonomous driving is presented in [1]. The dominant surface (e.g., the ground) is found through a v-disparity image [3] computation, while the obstacles come from a disparity space image (DSI) analysis. In this case, the cameras axes of the stereo system are almost parallel to the ground. Unfortunately, this approach is not suitable for start-inhibit, because one of the most important design issues is not to force a specific cameras alignment. In fact, the approach described in [1] requires a perfect camera alignment and precise constraints on cameras orientation.

Therefore a stereo inverse perspective mapping-based approach has been considered. The whole processing is performed by means of two main steps:

(i) lens distortion and perspective removal from both stereo images,
(ii) obstacle detection.

Concerning the first step, the problems of distortion removal and inverse perspective mapping without the knowledge of the intrinsic and extrinsic parameters of cameras

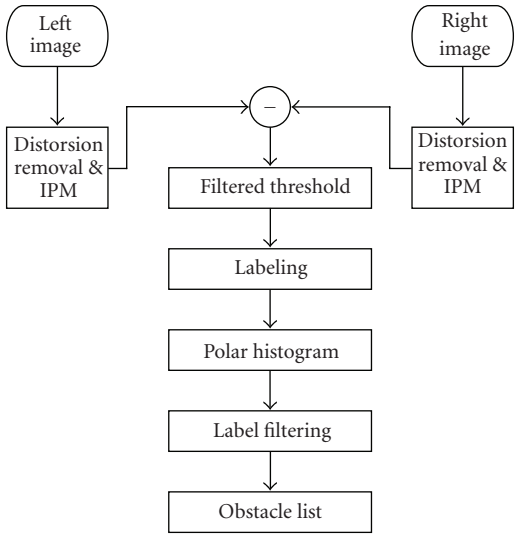


Figure 4: Algorithm's block diagram.

have to be solved. Lens distortion is usually modeled as polynomial radial distortion [4, 5] and it is removed by estimating the coefficients of this polynomial. After the distortion removal phase, extrinsic parameters are obtained [6], nevertheless, the highly complex mathematical model of the sferic lens may affect the computational time.

Therefore, a graphic interface to remove lens distortion has been designed to manually associate the grid points of the source image to their homologous points on a square grid on the *IPM* image [2] as explained in Section 2. This preprocessing is performed offline and the result are stored in a lookup table for a quicker online use.

In order to detect obstacles, two different approaches have been tested.

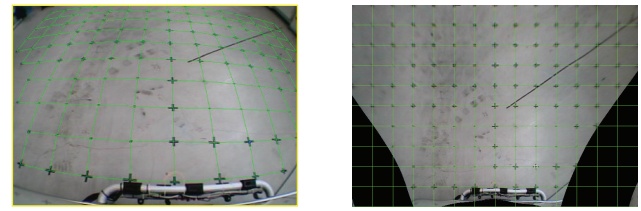
(1) The first searches for connected blocks on the thresholded image generated from the difference between left and right images after distortion removal and inverse perspective mapping (see [7]).
(2) The other one is based on the use of a polar histogram (see [8, 9]).

These two approaches have been fused into one algorithm to get the best from both. The whole algorithm flowchart is described in Figure 4 and is discussed in the following.

2. CALIBRATION

Camera calibration is one of the most important topics for vision systems especially when fielding systems that must be installed on real vehicles which have to operate in real scenarios.

In our case, highly distorting cameras are used without any knowledge about the intrinsic and extrinsic camera parameters. An analytic approach to calibration would be computationally prohibitive: the equations that are normally used to model sferic lenses become too complex when wide-angle lenses are used.



(a) (b)

Figure 5: Original and undistorted images of the grid.

Therefore, an empiric strategy has been used: during an offline preprocessing, a lookup table that allows a fast pixel remapping is generated; namely each pixel of the distorted image is associated to its corresponding pixel on the undistorted image. Images of a grid, painted on a stretch of flat road in front of the truck, are used to compute the lookup table (see Figure 5). A manual system to pinpoint all the crossing points on the source image is used.

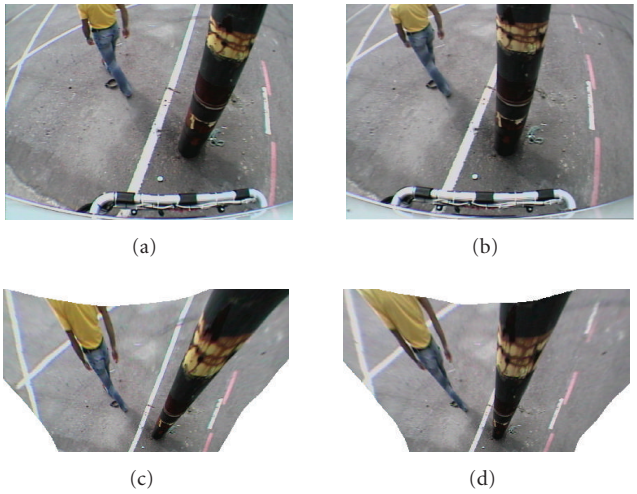
Thanks to the knowledge of the relative position of the truck with respect to the grid itself and to the assumption that the road can be considered nearly flat in the proximity of the vehicle, it is possible to compute a new image (the *IPM* image) removing both the perspective effect and camera distortion at once. A nonlinear interpolation function is used to remap the pixels of the source image that are not cross-points.

The process to determine coordinates (x, y) of the source image from the (i, j) pixels of the IPM image is divided into two steps.

Let us assume to have a grid with N vertical lines and M horizontal lines. For each vertical line of the grid, a function f_n is defined, where $n \in [1, N]$ is the line number. The spline creation is constrained by the correspondences between the crossing points of each line in the source image and in the IPM image; see (2) as an example, assuming x_1, y_1, x_2, y_2, and so forth, as the coordinates of the cross-points on the source image:

$$f_n(j) : \mathbb{R} \longrightarrow \mathbb{R}^2, \qquad f_n(j) = \begin{cases} f_n^x(j) \longrightarrow x, \\ f_n^y(j) \longrightarrow y, \end{cases} \tag{1}$$

$$\begin{aligned} f_1^x(0) &= x_1, \\ f_1^y(0) &= y_1, \\ f_1^x(1) &= x_2, \\ f_1^y(1) &= y_2, \\ &\vdots \\ f_1^x(N) &= x_N, \\ f_1^y(N) &= y_N. \end{aligned} \tag{2}$$



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

Figure 6: Perspective and distortion removal: (a) left source image; (b) right source image; (c) left IPM image; (d) right IPM image.

Using functions $f_1(j), f_2(j), \ldots, f_N(j)$, another class of functions can be built, called $g_j(i)$ and defined as described in (3) with (4) as constraint:

$$g_j(i) : \mathbb{R} \longrightarrow \mathbb{R}^2, \qquad g_j(i) = \begin{cases} g_j^x(i) \longrightarrow x, \\ g_j^y(i) \longrightarrow y, \end{cases} \tag{3}$$

$$\begin{aligned} g_j(1) &= f_1(j), \\ g_j(2) &= f_2(j), \\ &\vdots \\ g_j(N) &= f_N(j). \end{aligned} \tag{4}$$

In this way, all the pixels of the IPM image have a correspondence to a pixel of the source image and the cubic spline interpolation method allows to get the best match between the two sets of pixels. An example of the resulting images obtained using these equations is shown in Figure 6.

Being the system based on stereo vision, two tables, one for each camera and both fixed under the same reference frame, are computed with this procedure. The lookup table generation is a time-consuming step, but it is computed only once when the cameras are installed or when their position is changed.

3. ALGORITHM

Starting from the *IPM* images, a difference image D is generated by comparing every pixel i of left image (L) to its homologous pixel of the right one (R) and computing their absolute distance:

$$D_i = |L_i - R_i|. \tag{5}$$

In particular, working on RGB color images, the distance used is the average of absolute differences of each color channel.

(a) (b)

Figure 7: Difference image between Figures 6(c) and 6(d) and result of labeling.

Then a particular threshold filter is applied on the resulting image D. In particular, for each pixel we define a square area A centered on it; the average value m of all the pixels in that area is computed and a threshold γ is applied on m. The resulting value is assigned to the pixel T_i as shown in the following equation:

$$\forall i \in D, \quad m = \frac{\sum_{\forall j \in A} D_j}{N_A}, \quad T_i = \begin{cases} 0 & \text{if } m < \gamma, \\ 1 & \text{if } m > \gamma, \end{cases} \tag{6}$$

where N_A represents the number of pixels in A.

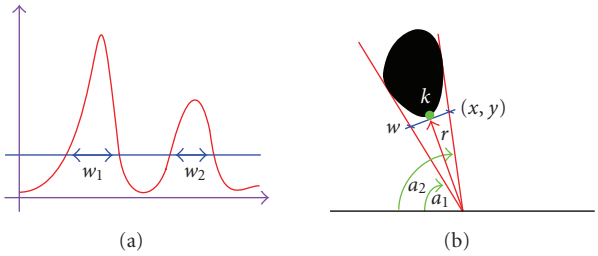
This is a kind of lowpass filtering and is useful to find the most significant differences in these images. Compared to similar methods like a thresholding followed by a morphological opening, it is faster because it is easy to be optimized and, nevertheless, works on the whole range of values of grey images.

Connected areas appearing in the resulting image are localized for and labeled: a progressive number is assigned to each label for further identification (as shown in Figure 7). A polar histogram is computed for each region. The focus used to compute the polar histogram is the projection of the mid point between the two cameras onto the road plane. These regions produce peaks on the polar histogram. Thus, the presence of strong peaks can be used to detect obstacles.

Some specific configurations of this histogram have to be considered, due to regions that are weakly connected or too thin to be a real obstacle. Therefore, it is necessary to further filter the polar histograms to remove regions that cannot be considered as obstacles.

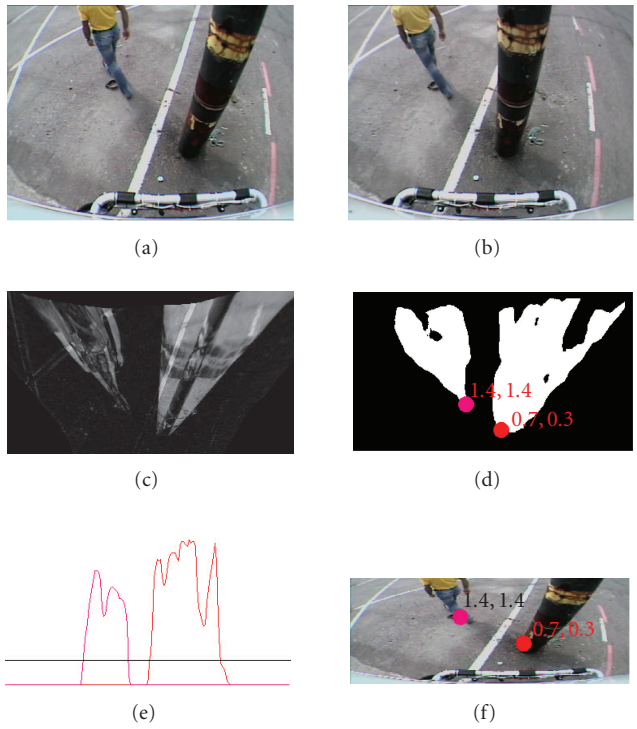
This filtering is performed considering the width of the histogram for the region of interest. The width of the histogram is computed in correspondence to a given threshold. When a polar histogram features several peaks, different values of width (w_1, w_2, etc.) are generated (see Figure 8(a)). If $\max\{w_1, w_2, \ldots, w_n\} > w_{\min}$ (where $w_{\min}$ is a width threshold), then the region previously labeled is maintained, otherwise it is discarded.

For each resulting region, the point k closest to the origin of the polar reference system and the angles of view (a_1, a_2) under which the region is seen are computed (see Figure 8(b)).



(a) (b)

FIGURE 8: (a) Polar histogram thresholding and filtering and (b) information extracted from the detected obstacle.



(a) (b) (c) (d) (e) (f)

FIGURE 9: (a) left source image, (b) right source image, (c) difference image, (d) connected components labeling, (e) polar histograms, (f) resulting image.

A rough width (w) of the detected object is computed as well, applying the following equation and considering r as the distance of k from the focus:

$$w = 2r \cdot \tan\left(\frac{a2 - a1}{2}\right). \tag{7}$$

Working on the *IPM* image, the location of point k in world coordinates can be estimated through the same lookup table previously used.

Figure 9 shows the complete set of intermediate results starting from the left and right original images; the difference and labeled images; the polar histogram whose filtering allows detecting one obstacle and discarding the small road curb; and finally the left original image with a red marker indicating the obstacle.

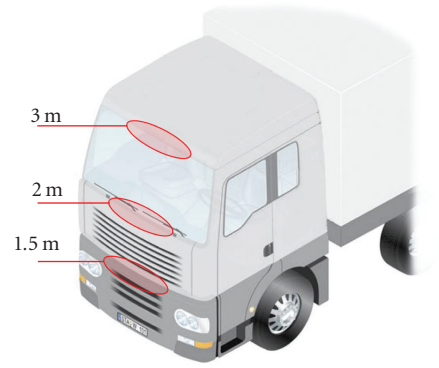


FIGURE 10: Possible position of the stereo pair.

(a) (b)

FIGURE 11: Two examples of StereoBox hardware.

4. COMPUTATIONAL REQUIREMENTS

The system presented in this paper was tested in several situations and with different architectures.

The algorithm can be applied to both progressive and interlaced images, widen the range of possible applications and hardware. Applied to a pair of 768×576 pixels interlaced color images, it takes approximately 30 milliseconds to be executed on an off-the-shelf Pentium4 running at 3.2 GHz. On the same architecture, working on stereo 640×480 progressive image retrieved from Bayer Pattern CCD sensor, the algorithm takes only 20 milliseconds to be executed on each frame.

Due to the small amount of resources required, the system was ported also on cheaper architectures. On Via EPIA EN15000 running at 1.5 GHz, analyzing stereo 640×480 progressive images, algorithm takes about 80 miiliseconds and it is thus capable to run up to 10 Hz.

5. SYSTEM SETUP

The stereo pair is placed right above the region of interest: in particular in all the different set-ups tested so far the cameras have been fixed in the front side of the vehicle.

The system was tested with cameras installed at several different heights: 3 m, 2 m, and 1.5 m, as shown in Figure 10. Stereo baseline and camera lenses must be changed accordingly. Values for baseline and focal length shown in Table 1 were chosen in order to view a given area.

Another important degree of freedom is cameras convergence: especially in case of large baselines or low heights, it is hard to view the whole region of interest with both cameras when their optical axes are parallel. Since images are

Figure 12: Result images showing typical algorithm output. A red dot shows the closest point of contact of each obstacle with the ground.

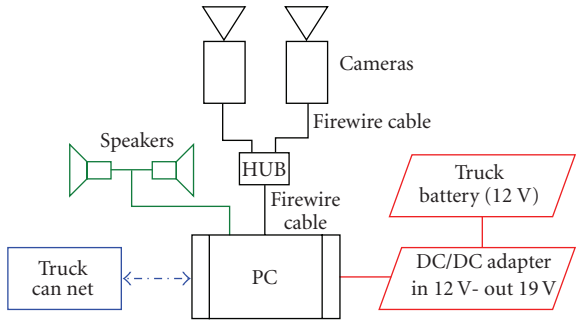


Figure 13: Block schema of the system.

preprocessed with a lookup table (as explained in Section 2) every effect introduced by freely placing of cameras is removed together with distortion and perspective.

In Figure 11 are shown different systems developed for two different projects.

The system is able to provide several types of output on several peripherals (typical application layout is shown in Figure 13).

Table 1: System specifications for different cameras heights.

Height (m)	Baseline (m)	Focal length (mm)
3	0.8	2.3
2	0.5	2.3
1.5	0.5	2.2

(i) The system can provide a visual output (e.g., on a display). This output consists in dedistorted image with mark on detected obstacles. A blinking red frame notify to driver danger condition.

(ii) An audio output: an intermittent sound is modulated according to distance and position of obstacles.

(iii) Through CAN bus, detected object's world coordinates are sent and a system can use this information to perform a high-level fusion with others' sensors.

(iv) Using CAN (or serial/ethernet) interface, the system can drive directly others' warning device (e.g., load torque on throttle command).

6. CONCLUSION AND FUTURE WORK

This paper presents an easy, fast, and reliable stereo obstacle detection technique for a start-inhibit system. Cameras mounted on a vehicle are arbitrary aligned, meaning that no special alignment is required by specialists or IT professionals. The choice of using a stereo vision system instead of radar or ultrasonic devices stems from the fact that the driver can see directly the image and can understand what caused the alarm.

Tests were made in several environmental conditions considering different kinds of road and obstacles, even with different illumination conditions. Low illumination conditions do not affect the system behavior because headlamps light up only the interesting part of vertical obstacles, easing the detection. To avoid light reflection, polarizing filters could be mounted in front of cameras.

Figure 12 shows some examples of the algorithm output remapped onto the original image. Red circles are used to mark obstacles positions. On the long tests performed, no false negatives were found: every single pedestrian and every tall enough obstacle were detected. Some false positives were generated by reflective road surfaces (water, e.g.).

Taking advantage of the stereo approach, the road texture, road markings, and shadows are successfully filtered out. Moreover, the algorithm easily detects large obstacles, rejecting most of smaller ones, like sidewalk borders. In general, due to the particular configuration of the system, vertical objects are correctly detected, thus the use of image tracking or temporal comparisons seems not mandatory.

Future developments will be centered on providing an automated algorithm to calibrate the system. A standard grid with easily recognizable markers will be placed in front of vehicle and an automated calibration procedure will be engaged by an operator. This procedure will become necessary only after major vehicle changes and/or maintenance.

ACKNOWLEDGMENT

The work described in this paper has been developed in the framework of the Integrated Project APALACI-PReVENT, a research activity funded by the European Commission to contribute to road safety by developing and demonstrating preventive safety technologies and applications.

REFERENCES

[1] A. Broggi, C. Caraffi, R. I. Fedriga, and P. Grisleri, "Obstacle detection with stereo vision for off-road vehicle navigation," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, p. 65, San Diego, Calif, USA, June 2005.

[2] M. Bertozzi, A. Broggi, and A. Fascioli, "Stereo inverse perspective mapping: theory and applications," *Image and Vision Computing*, vol. 16, no. 8, pp. 585–590, 1998.

[3] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection on non flat road geometry through "v-disparity" representation," in *Proceedings of IEEE Intelligent Vehicles Symposium*, vol. 2, pp. 646–651, Versailles, France, June 2002.

[4] D. Claus and A. W. Fitzgibbon, "A rational function lens distortion model for general cameras," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 213–219, San Diego, Calif, USA, June 2005.

[5] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.

[6] R. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[7] M. Bertozzi, A. Broggi, P. Medici, P. P. Porta, and A. Sjögren, "Stereo vision-based start-inhibit for heavy goods vehicles," in *Proceedings of IEEE Intelligent Vehicles Symposium (IVS '06)*, pp. 350–355, Tokyo, Japan, June 2006.

[8] M. Bertozzi and A. Broggi, "GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 62–81, 1998.

[9] K. Lee and J. Lee, "Generic obstacle detection on roads by dynamic programming for remapped stereo images to an overhead view," in *Proceedings of IEEE International Conference on Networking, Sensing and Control (ICNSC '04)*, vol. 2, pp. 897–902, Taipei, Taiwan, March 2004.

Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 74706, 16 pages
doi:10.1155/2007/74706

Research Article

State of the Art: Embedding Security in Vehicles

Marko Wolf,[1] André Weimerskirch,[2] and Thomas Wollinger[2]

[1] *Horst-Görtz-Institute for IT Security, Ruhr-University Bochum, Universitätsstraße, 44780 Bochum, Germany*
[2] *escrypt-Embedded Security GmbH, Lise-Meitner-Allee 4, 44801 Bochum, Germany*

Received 19 October 2006; Accepted 13 April 2007

Recommended by Paolo Lombardi

For new automotive applications and services, information technology (IT) has gained central importance. IT-related costs in car manufacturing are already high and they will increase dramatically in the future. Yet whereas safety and reliability have become a relatively well-established field, the protection of vehicular IT systems against systematic manipulation or intrusion has only recently started to emerge. Nevertheless, IT security is already the base of some vehicular applications such as immobilizers or digital tachographs. To securely enable future automotive applications and business models, IT security will be one of the central technologies for the next generation of vehicles. After a state-of-the-art overview of IT security in vehicles, we give a short introduction into cryptographic terminology and functionality. This contribution will then identify the need for automotive IT security while presenting typical attacks, resulting security objectives, and characteristic constraints within the automotive area. We will introduce core security technologies and relevant security mechanisms followed by a detailed description of critical vehicular applications, business models, and components relying on IT security. We conclude our contribution with a detailed statement about challenges and opportunities for the automotive IT community for embedding IT security in vehicles.

1. INTRODUCTION

Information technology—we broadly define as being systems based on digital hardware and software—has gained central importance for many new automotive applications and services. The costs for software and electronics are estimated to approach the 50% margin in car manufacturing in 2015 [1]. Perhaps more importantly, there are estimates that already today more than 90% of all vehicle innovations are centered on IT software and hardware [2]. These applications are realized as embedded systems and range from simple control units to infotainment systems equipped with high-end processors whose computing power approaches that of current PCs. In premium cars, one can find up to 70 processors that are connected by several bus types and up to several hundred megabytes of embedded codes.

Not surprisingly, many classical IT and software technologies are already well established within the automotive industry, for instance hardware-software codesign, software engineering, software component reuse, and software safety. However, one aspect of modern IT systems has little attention in the context of automotive applications: IT security. Security is concerned with protection against malicious manipulation of IT systems [3, 4]. The difference between IT safety and IT security is depicted in Figure 1. Nevertheless, IT safety and IT security are interleaved fields, that is, some technical failure (safety issue) can be used to realize some malicious threat (security issue) and vice versa.

However, there are today niche applications in the automotive domain (e.g., immobilizers) that particularly rely on IT security technologies. Nevertheless, the majority of software and hardware systems in current cars is *not* protected against manipulations. The reason being that past car IT systems did not need security functions because there was only little incentive for malicious manipulation. Secondly, security tends to be an afterthought in any IT system, because achieving of the core function is often the main focus when designing a system. As can be seen for instance by the Internet development, implementing IT security afterwards, is normally doomed to failure.

The situation has changed dramatically, as we will state in this contribution with respect to the arguments given above. More and more vehicular systems need security functionality in order to protect the driver, the manufacturer, and the component supplier. Secure software update of electronic control units (ECUs), preventing chip tuning, preventing the unauthorized change of the mileage, or assembling nonoriginal parts are only some examples. Future cars will become even

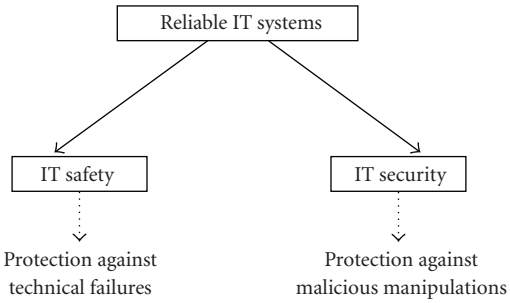


Figure 1: The relationship between IT safety and IT security.

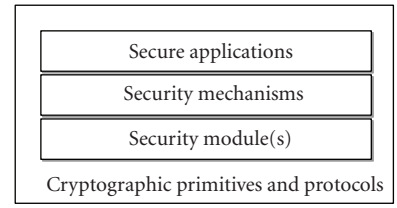


Figure 2: Layered security architecture to enable security-critical vehicular applications based on cryptographic primitives and protocols.

more dependent on IT security due to the following developments.

(i) An increasing number of ECUs will be reprogrammable and have to be protected.
(ii) Electronic antitheft measures will go beyond current immobilizers, for example, by protecting individual components.
(iii) An increasing number of legislative requirements (e.g., secure emergency call functions).
(iv) New business models (e.g., time-limited car functions or pay-per-use infotainment content) will be established.
(v) Vehicles will communicate with the environment in a wireless fashion that requires protected car-to-infrastructure communication.
(vi) Increasing networking of cars enables car-to-car communication that has to be protected against abuse and violation of privacy.

IT security will play an important role for several future automotive technologies and will even be an enabling technology for some future applications. The target platforms within cars that incorporate security functions are embedded systems, rather than classical PC-style computers. Some obvious differences in comparison to common PC-based environments are listed below.

(i) Embedded devices have small processors (often 8-bit or 16-bit microcontrollers) which are limited with respect to computational capabilities, memory, and power consumption. Hence, the usage of cryptographic primitives and protocols is limited.
(ii) Embedded devices mostly have only limited possibilities and limited bandwidth for external communication. Hence, the extent and frequency of external communication, for example, for internal updates, are limited.
(iii) Attackers of embedded systems have often physical access to the target device itself.
(iv) Embedded systems are often relatively cheap and cost-sensitive because they often involve high-volume products. Thus, adding complex and costly security solutions is not acceptable.
(v) It is costly to establish the necessary organizational aspects for security products, for example, one needs to adopt the production and life-cycle chain.

Hence, the technologies needed for securing vehicular applications mainly belong to the field of embedded security that differs from general IT security.

Outline

The topics discussed in this contribution give a state-of-the-art overview of IT security in vehicles. We start with an introduction of basic cryptographic functionality in Section 2, providing the theoretical framework for most security mechanisms in cars. We then point out the necessity of vehicular IT security while presenting attacks and attackers, deduce relevant security objectives, and indicate characteristic constraints within the automotive area in Section 3. As depicted in Figure 2, we subsequently introduce and explain each essential layer to enable security-critical applications within vehicles. Therefore, we first discuss the necessary security module in Section 4, followed by an overview of security mechanisms in Section 5 that are based on the security module. In Section 6, we present various current and future security-critical vehicular applications that rely on available vehicular IT security. We conclude our contribution with a detailed statement about challenges and opportunities for the automotive IT community for embedding IT security in vehicles.

IT security, however, comprises both technical and organizational measures. IT security systems always include security relevant organizational processes, and in many cases an IT security system is compromised due to organizational weaknesses. To enable secure automotive IT applications, complex and reliable organizational structures are required. Thus, organizational security has to be considered individually and additionally to all technical measures treated in our contribution.

2. CRYPTOGRAPHIC BACKGROUND

Besides security enhancing technologies such as filtering (e.g., firewalls), anomaly detection (e.g., intrusion detection systems), or vulnerability scanning (e.g., antivirus software), cryptographic primitives for data encryption and decryption, signature generation, and verification including the

necessary cryptographic protocols are the core of virtually all security-critical IT systems.

Understanding the basic functionality is essential for designing, analyzing, implementing, and assessing an IT security system. In this section, we therefore identify the basic security services that can be provided by cryptography followed by short introductions of symmetric- and public-key cryptographies, cryptographic hash functions, and cryptographic protocols relevant for vehicular security applications.

2.1. Security properties

Even though security depends on much more than cryptographic algorithms—a robust overall security design including secure protocols and organizational measures is needed as well—cryptographic primitives and schemes are in most cases the atomic building blocks of a security solution. In the following, we specify the security properties that properly combined cryptographic primitives and schemes are required to enable. Further reading can be found in [5, 6].

(i) *Confidentiality* (or *privacy*) is a service ensuring that information is kept secret from all but authorized parties.

(ii) *Integrity* is a service ensuring that unauthorized parties cannot modify system assets and transmitted information. Modification includes writing, changing, changing the status, deleting, and creating the transmitted messages. It is important to point out that integrity relates to active attacks as well as technical errors, and therefore it is concerned with detection rather than prevention. Moreover, integrity can be provided with or without recovery.

(iii) *Authentication* (more precisely *message origin authentication*) is a service concerned with assuring that the origin of a message is correctly identified. Note that origin authentication implies integrity; the opposite is not true.

(iv) *Identification* (more precisely *entity authentication*) is a service establishing the identity of an entity (e.g., a person, computer, credit card).

(v) *Nonrepudiation* is a service that prevents the sender of a message from denying commitments or actions.

(vi) *Access control* is a service restricting access to resources to privileged entities.

Security services can be achieved by employing the two most important cryptographic schemes: symmetric and asymmetric cryptographies. Symmetric cryptography provides the ability to securely exchange messages between two parties. This is especially important if the data should not be revealed to any third party. Authentication without nonrepudiation can also be achieved if the secret key is known only to the two parties. The second family of schemes, asymmetric or public-key algorithms, provides advanced functions such as digital signatures and key distribution over insecure channels. For common automotive applications, both symmetric- and public-key algorithms are used.

2.2. Symmetric-key cryptography

Symmetric-key cryptographic algorithms are the basic building blocks of any secure system that requires at least confidentiality. They are used to encrypt messages in bulk and to provide secure storage of data. In this kind of cryptographic algorithms, the keys used for encryption and decryption are the same for both communicating entities, and hence called a *symmetric cipher*. It can be considered as a locked box with the messages inside that is sent to the other party. If the other party has the right key to the lock, then the party can open and read all the messages in the box. The security of the symmetric cipher depends on the key (the algorithm is assumed to be public). The exchange of these keys between the parties should be done using a secure channel, for example, provided by a public-key cryptosystem.

Symmetric-key algorithms are mainly divided into two categories: *block ciphers* and *stream ciphers*. Block ciphers encrypt the messages in data blocks of fixed length, mostly 64 bits or 128 bits. Most well-known block ciphers are the data encryption standard (DES) [7], and the advanced encryption standard (AES) [8]. DES was the first commercially standardized block cipher with 64-bit data block size and 56-bit key size. The algorithm has been widely used because it was the only standardized and openly available algorithm extensively studied by the cryptanalytic community. There have been no major weaknesses found in the algorithm to date to practically break it other than the relatively small size of the key. This allows a brute force attack running through all the keys. DES finally expired as an US standard in 1999 and the National Institute of Standards (NIST) selected the Rijndael algorithm as the advanced encryption standard (AES) in October 2000. In the transition phase, triple-DES was approved as an FIPS standard [7]. The Rijndael algorithm [9] developed by Daemen and Rijmen was selected in an open challenge from a large set of algorithms submitted. AES [8] supports variable block and key sizes of 128 bits, 192 bits, and 256 bits to give a choice of different security levels based on its application. AES has been optimized for efficient software and hardware implementations.

Unlike block ciphers, stream ciphers encrypt a plain text bit by bit. The most famous example is the one-time pad (OTP) [10] encryption (also called Vernam cipher) which is the only known cipher which can be proven to be unbreakable [11]. The OTP works by bitwise XOR of the plain text with a one-time key, which is of the same length. The problem of having a secret key of the same length as the message to be transmitted over a secure channel makes OTP encryption inconvenient in practice. This shortcoming is overcome by using a pseudorandom generator as source for the secret key (but the unconditional security holds no more). Today's stream ciphers operate on a single bit of plain text (or a few bytes of data) being XORed to a pseudorandom key stream generated based on a master key and an initialization vector. Stream ciphers are especially useful in situations where transmission errors are highly probable because they do not have error propagation. They can be used when the data must be processed one symbol at a time because of lack

of device memory or limited buffering. Furthermore, stream ciphers mostly provide a higher throughput in comparison with block ciphers.

2.3. Public-key cryptography

The main function of symmetric algorithms is the encryption of information, often at high speeds. However, there are two problems with symmetric-key schemes.

(1) It requires secure transmission of a secret key, before being able to exchange messages.
(2) If in a network environment, each pair of users shares a different key, this will result in many keys.[1] Hence, this fact may result in problems handling the key management.
(3) After secure reception of a secret key, each party has to store its key securely for reuse.

The idea behind public-key (PK) cryptography can be visualized by making a slot into the locked box so that everyone can deposit a message (like a letter box). However, only the receiver can unlock the box and read the messages inside. This concept was first proposed by Diffie and Hellman [12] in 1976.

Public-key cryptography is based on the idea of separating the key used to encrypt a message from the one used to decrypt it. Anyone who wants to send a message to another party, for example, to Bob, can encrypt that message using Bob's *public key*. However, only Bob can decrypt the message using his *private key*. It is understood that the private key should be kept secret at all times, whereas the public key is publicly available to everyone. Furthermore, it is impossible for anyone, except Bob, to derive the private key from the public key (or at least to do so in a reasonable amount of time).

One can realize three basic mechanisms with public-key algorithms:

(i) key establishment and key exchange;
(ii) digital signatures;
(iii) data encryption.

In general, one can divide practical public-key algorithms into three major families.

(i) Algorithms based on the *integer factorization problem*: given a positive integer n, it is computationally hard to find its prime factorization, for example, RSA [13].
(ii) Algorithms based on the *discrete logarithm problem* (DLP): given α and β, it is computationally hard to find x such that $\beta = \alpha^x \bmod p$, for example, the Diffie-Hellman key exchange and the digital signature algorithm (DSA).
(iii) Algorithms based on *elliptic curves* rest upon the DLP on the algebraic structure of elliptic curves over finite fields. Elliptic curve cryptosystems [14, 15] are the most recent family of practical public-key algorithms, which have gained acceptance including standardization [16].

There are many other public-key schemes, such as NTRU or systems based on hidden field equations, which are not in widespread use. The scientific community is only at the very beginning of understanding the security of such algorithms. Despite the differences between their underlying mathematical problems, all three algorithm families have something in common: they all perform complex operations on very large numbers, typically 1024–4096 bits in length for the integer factorization and discrete logarithm systems, and 160–256 bits in length for elliptic curve systems (see also Table 1). This results in a poor throughput performance in comparison with symmetric ciphers. Nevertheless, public-key algorithms solve the key distribution problem in an elegant way, since the public part of the key can be distributed via an unsecured channel. Hence, one can establish a secure link between two parties without the need for an ulteriorly, previously exchanged secret. Thus, PK encryption is normally used for transmitting only small amount of data, like symmetric keys (see Section 2.6). Public-key algorithms are not only used for the exchange of secret keys, but also for the authentication by using digital signatures. Digital signatures are analogous to handwritten signatures. They enable communication parties to prove to a third party that one party has actually generated the message, also called nonrepudiation. The idea of the digital signature is appending a digital data block to the message that can be generated according to the message only by the person who signs it (like conventional signatures). Since the digital signature is a function of the message content and the private key, only the holder of the private key can sign the corresponding message. In practical terms, we use the private key for signing (thus only the holder of the nonpublic private key can sign a document) and the public key for the verification (thus everyone can verify the signature using the openly available public key). For practical implementations, using the RSA algorithm for digital signatures, a significant smaller public key[2] can be chosen to make the verification of an RSA signature a very fast and facile operation. Hence, RSA should be used in applications where the verification is done on the embedded platform and the signing on a personal computer or server. Instead, ECC should be used for applications where the embedded device performs encryption and signature generation as well as decryption and signature verification, since ECC is more efficient considering an application where the embedded device has to cover the complete public-key functionality.

2.4. Recommended key length

Table 1 puts the public-key and symmetric-key bit lengths in perspective. This recommendation assumes that in the near

[1] For a network with n users, $n \cdot (n-1)/2$ individual keys have to be shared afore.

[2] However, the private RSA key needs to have full length, for security reasons.

Table 1: Recommended key length for public-key and symmetric-key cryptographies.

Security	AES/DES	ECC	RSA
Short-term	64 bits	128 bits	700 bits
Middle-term	80 bits	160 bits	1024 bits
Long-term	128 bits	256 bits	4096 bits

future, there will be no unexpected (mathematical) attacks. PK systems need much longer keys, because of the attacks known today, which are more powerful than in the case of private-key primitives. However, choosing the appropriate key length depends much on the kind and security targets of the respective application. Highly security-critical vehicular applications such as digital tachographs, motor control units, or immobilizers have to provide *at least* middle-term security, whereas less security-critical applications such as personalized presets or customer information services could apply even short-term security. Although OEMs hardly provide any public information about applied security standards, we will provide at least two useful references providing key length recommendations for flash security [17] and for wireless car access [18].

2.5. Hash functions

In cryptography, hash functions are used in many applications, for example digital signatures, pseudorandom number generators, one-way functions, message authentication codes (MAC), and others. Hash functions compress a message of any length to a (nearly) unique string of fixed length, the so-called hash value or digital fingerprint. Hash functions are *one-way functions*, that is, for (almost) all given outputs y, it is impossible to find any input x such that $h(x) = y$. Hence, with a given input, a hash value can be computed, but it is computationally infeasible to compute the input if only the hash value is known. A *collision-free* function is a function where an attacker cannot find two inputs that compute the same hash value. Since hash functions map more than one value to the same hash, a collision cannot be prevented, but it has to be hard for the attacker to find a collision.

Nowadays, there are several families of hash functions. The MD family [19] and SHA family [20] are the ones mostly used. The MD family generates hash values up to 128 bits but suffer from serious flaws[3] making further use of the algorithm for security purposes questionable. The SHA family was developed by the NSA in 1995 (updated last in 2004) and generates hash values up to 512 bits. Attacks have been conducted also within the SHA family particularly for the widely used SHA-1 (160-bit hash value). No attacks have yet been reported on the higher SHA variants (256 bits and 512 bits), but since they are similar to SHA-1, researchers are worried, and are currently developing candidates for a new hashing standard.

[3] There exist algorithms to find a collision within minutes using a standard computer.

2.6. Cryptographic protocols

Two cryptographic protocols used for many automotive applications are *hybrid encryption* and the *challenge-response* protocol.

The major disadvantage of public-key primitives, when compared to symmetric-key schemes, is the arithmetic intensive operations that need to be performed. Hence, this can lead to a poor system performance. Even when properly implemented, all PK schemes proposed to date are several orders of magnitude slower than the most efficient symmetric-key schemes. Hence, in practice, cryptographic systems are applied as a mixture of symmetric-key and public-key cryptosystems in a hybrid fashion. Usually, a public-key algorithm is chosen for key establishment and then a symmetric-key algorithm is chosen to encrypt the communications, achieving in this way high throughput rates. As shown in Figure 3, the sender (Alice) first encrypts a symmetric-key K with the public key $\mathrm{PK}_{\mathrm{Bob}}$ of the receiver (Bob). Bob then decrypts K using his secret private key $\mathrm{SK}_{\mathrm{Bob}}$. Afterwards, both proceed their communication using a symmetric cipher with the previously shared K.

The challenge-response protocol provides entity authentication also called identification, that is, one communication party identifies itself to a second party. The identification can be provided by using knowledge, possession, or individual properties. The basic idea of the protocol is that one party challenges the second party, for example, by sending a random number. The challenged party then has to answer with the correct response. This correct response can be generated only if the second party has for instance some kind of knowledge, for example, the key for a cryptographic primitive. The party can use the key to encrypt the given random number and returns it to the challenger, thus proving the possession of knowledge without revealing it.

Figure 4 presents the challenge-and-response protocol using a symmetric-key algorithm. However, the protocol can also be implemented using public-key primitives. In Figure 4, Alice challenges Bob by sending a random number c. Bob encrypts c together with the identity of Alice and returns the response r. Bob is authenticated once the identity of Alice and the random number are correctly verified. Note that only Bob can response to the challenge correctly, since only Bob possesses the knowledge of the appropriate secret key K.

3. AUTOMOTIVE ATTACKS, SECURITY OBJECTIVES, AND CHARACTERISTIC CONSTRAINTS

In the following, we first provide an overview of specific attacks and attackers in the automotive environment that differ from common PC-based IT systems. We then deduce overall automotive security objectives along with the

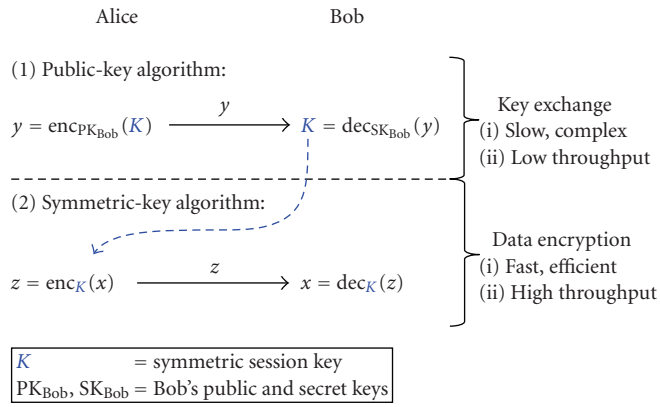


Figure 3: Hybrid encryption protocol.

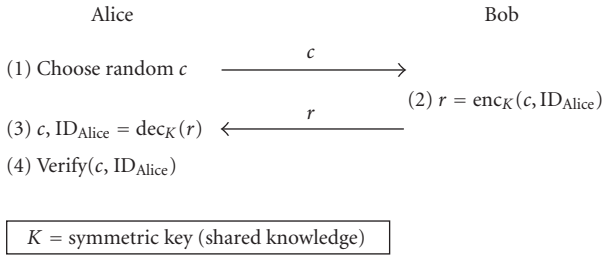


Figure 4: Challenge-response protocol.

characteristic automotive technical and organizational constraints.

3.1. Attackers in the automotive area

Today attackers within the automotive area usually either want to steal a vehicle or a certain valuable component (e.g., the navigation system) or—at owner's disposition—want to modify certain critical components. These modifications include for instance manipulation of the mileage for a higher resale value (reduced mileage) or a higher tax return (increased mileage), manipulation of the motor control unit (chip tuning) for unauthorized driving parameters, or manipulations of the tachograph to circumvent legal driving restrictions or to conceal potential previous infringements. With future electronic applications (cf. Section 6) such as electronic license plates, event data recorders, car communication, and copyrighted infotainment, misuse potentialities will obviously increase further. Finally, there exist nonnegligible, partially quite extensive, efforts to steal competitors' expertise and intellectual property in order to advance own developments or, more likely, to illegally produce marketable counterfeits.

Since automotive IT systems, in comparison to common (PC-based) IT systems, imply specific characteristics, attacks on vehicular IT systems differ from attacks on usual computer systems. Attackers of a computer system seldom have physical access to the target system, whereas attackers in the automotive sector mostly have physical access to all built-in electronics. If there are no further protection measures integrated, attackers can manipulate or replace all built-in components. Moreover, afterwards discovered vulnerabilities are much harder to fix once hundreds and thousands of vehicles are sold already. Finally, automotive attacks are usually "offline attacks," where attackers have almost unlimited time and unlimited trials to succeed.

According to the two different attacking objectives—theft and modification—we identify four different groups of attackers as depicted in Table 2. In case of theft, the thief may have considerable technical expertise and some appropriate tools. However, a thief usually has only limited physical access and limited time. Within the attacking group having systematic modifications in mind, three typical kinds of attackers can be identified. The first group includes individuals such as the car owner. They normally have only little technical expertise, a few appropriate devices, as well as restricted financial resources for applying an attack. Skilled (OEM) garage employees are the second group of attackers. They have appropriate tools, have the necessary technical expertise, and are mostly endued with insider information. They even would invest money if an attack promises appropriate revenues, that is, if the attack can be scaled to many automobiles easily. The third group of attackers includes concurrent manufacturers, counterfeiters, and organized crime that may have immense technical and financial resources limited only be the potential economic gain. The motivation of this group is to gain competitors intellectual property (IP) or to exploit the outcome of an attack commercially, for example, by selling counterfeits or providing tools and expertise in the Internet.

Since the group of counterfeiters and organized crime is the most powerful and dangerous one, all actions to protect automotive IT should try to resist in particular attacks from these in such a way that the costs of a successful attack will exceed the potential gain. More concrete, a single successful attack on an automotive device must not scale to break also all other devices, for example, by revealing a global identical secret.

Table 2: Attackers in the automotive area.

Target	Systematic modification			Theft
Attacker	Individual, owner	Mechanic, garage personnel	Organized crime, competitor, faker	Thief
Technical resources	Varied (Generally low)	High	Very high	Varied
Financial resources	Low	Medium	Very high	Low
Physical access	Full	Full	Full	Limited
Risk	Low	Medium	Very high	Medium

3.2. Automotive attacks

This section provides an overview of specific hardware and software attacks in the automotive environment that typically differ from attacks on common PC-based IT systems.

3.2.1. Attacks on automotive hardware

Attacks on automotive hardware comprise attacks to replace critical components with unauthorized components or to illegally modify existing components. Usually, most hardware components provide, beyond some more or less sophisticated tags, no further protection mechanisms. They can be easily cloned, modified, or replaced by unauthorized components. However, a few critical components such as the tachograph, the speedometer, or airbags provide some basic (cryptographic) mechanisms to prevent or at least to detect unauthorized modifications, replacements, or misuse.[4] In such cases, hardware attacks aim at the circumvention or breaking of these protections by readout of secret keys, deactivation of alarm channels, or wiretapping their operation or communication.

3.2.2. Attacks on automotive software

Today's vehicles hold several dozen electronic control units (ECU) that control almost anything such as air conditioning, electric windows, engine, and break system. Several of these ECUs allow downloading updated program and data code to apply bug fixes, to improve existing functionality, to renew underlying data, or to install/activate new software features. The software update might be performed over a diagnosis channel, other available communication channels such as Bluetooth and GSM, or by using a storage medium such as a CD-ROM or a USB device.

However, present automotive IT systems are mostly unprotected against malicious software attacks. Often, for example, ECUs memory can be accessed without any further restrictions using their regular interface. Other may be compromised employing unprotected diagnosis or communication interfaces. At last, all ECUs without further measures for tamper resistance can be dismounted and analyzed offline using sophisticated analysis equipment. Obfuscation techniques[5] and pure software encryption (without hardware support) provide only minimal additional protection, since all programs have to be decrypted during runtime, and hence will be stored and decrypted at one point. The program code can then be read out and analyzed by attackers with only moderate technical understanding. Moreover, encryption keys are mostly stored somewhere unprotected or can be guessed easily. Disabling even sophisticated software protection measures by reengineering the "decisive validation branch" within the binary enables circumvention of almost all available software protection mechanisms [22].

Important (software) security vulnerabilities could also originate from inadequate OEM-internal software protection management. Thus, employees should not be able to reveal software to competitors or other unauthorized persons (unconsciously or maliciously) if adequate organizational security precautions are established and executed.

3.3. Overall security objectives

To guarantee road safety and operational reliability of vehicles and to sufficiently protect business models based on the security of the vehicular platform, we define the following overall automotive security objectives.

(i) *Confidentiality of data*: unauthorized access to protected data must be infeasible.

(ii) *Integrity of data*: unauthorized modification of data must be infeasible or at least detectable.

(iii) *Hardware and software integrity*: unauthorized modifications to vehicular hardware and software must be infeasible or at least detectable (by the vehicle).

(iv) *Availability*: authorized hardware and software components must have proper access to their dedicated data and services.

(v) *Uniqueness*: unauthorized cloning of a hardware components must be infeasible or at least detectable as nonauthentic.

3.4. Technical constraints

The application of complex IT systems in automotive environments is subject to some characteristic technical constraints.

[4] Night vision devices for instance, already available for premium class vehicles, are mandatory [21] protected against unauthorized, nonautomotive usage, for example, as military equipment, by terrorists or guerilla forces.

[5] Still used to "protect" for instance mileage information.

Automotive computing resources are—in comparison to usual computer systems—rather limited. Nevertheless, automotive applications are often required to provide (hard) real-time capabilities. This leads to severe requirements on complexity, memory size, and runtime efficiency for automotive implementations that moreover often have to cope with lots of specific architectural restrictions.

Vehicular IT systems are often subject to specific physical constraints such as high variations in temperature, moisture, or particular mechanical loads. They have to cope with these conditions usually over a product life cycle of up to 20 years in which only minimal maintenance efforts are acceptable.

Moreover, vehicular IT systems usually have only limited communication resources to, for example, exchange cryptographic keys or updating software. Thus, virtually all vehicular functionalities have to work properly even with an external communication functionality severely limited in capacity and frequency.

Since typical computer users can mostly employ ergonomic input and output devices, users within the automotive environment are restricted to only little ergonomically designed peripheral devices. To demand only a minimum of user interactions, virtually all vehicular applications are required to run almost completely autonomously.

3.5. Nontechnical constraints

Beyond the technical constraints, automotive IT systems are also subject to some particular organizational and legal constraints that may substantially differ from legal constraints for usual computer systems.

A possible public key and certificates infrastructure (PKI) for instance requires complex and costly organizational structures, particularly within the automotive context with a multitude of involved parties (e.g., manufacturer, supplier, OEM, garage personnel, content provider, etc.) and only limited (end-user) security understanding.

Another important key factor is interoperability to existing infrastructures and devices to enable end-users to integrate their existing devices (e.g., mobile navigation systems, smart phones, multimedia players, etc.) as simple and holistic as possible.

Since vehicular IT systems—in comparison to, for example, usual operating system software—have only limited possibilities for maintenance, compatibility, stability, safety, and reliability of deployed hardware and software are obligatory requirements. In particular, the corresponding support infrastructure must be available during the complete typical life cycle of the vehicle, that is, up to two decades.

Finally, as vehicular IT systems are often involved in highly safety-critical modules (e.g., steering lock, drive-by-wire systems), they cannot be released "without any warranty" and "exclusion of any damages" as most PC software usually does. For providing operating safety and legal security, legally binding warranties are mandatory. However, warranty statements can usually only be given based on complex and expensive internal and external certification procedures [23]. Thus corresponding documentation, models, tests, and assessments, as well as the development process itself have to be prepared for possible certifications already at the beginning of any development process.

4. SECURITY MODULE

security module, which is also called a security anchor, provides necessary security relevant methods such as encryption and decryption, generation and verification of signatures, hashing, and secure storage of cryptographic keys. Such a module might be implemented in software or hardware. Clearly, a hardware solution provides higher performance and a far higher security level.[6] It is possible to deploy a single central security module in a vehicle (e.g., at a central control unit) or to implement it in each control unit that has a need for security. In the first case, a hardware implementation is appropriate to securely protect numerous critical assets; whereas in the latter case sometimes a software implementation could be adequate.

A security module must fulfil the following requirements.

(i) *Unclonable*: a security module must be unclonable. It is desirable to bind the identity of a vehicle to the security module in such a way that it can neither be faked, or manipulated, nor cloned. In addition, it must be impossible to install the security module in another car in order to change its identity.

(ii) *Secure key storage*: a security module must be able to store keys in a secret and protected way. It must protect secret keys from being read and public keys from being altered.

(iii) *Secure computations*: the security module must be able to securely (and efficiently) perform cryptographic operations to prevent leakage of cryptographic secrets into unprotected areas.

(iv) *Alarm channel*: in case of a security breach, the security module must be able to give notice. For instance, such an alarm channel might be provided at diagnosis.

A security module can be based on a customized security controller, a trusted platform module (TPM), or an FPGA. A TPM provides a compatible standard interface more suited to the PC office world, whereas a customized security controller approach can be adapted in a flexible way. Both approaches provide a highly secure computing environment as well as secure key storage. An approach based on FPGAs provides a very flexible way at higher cost. Table 3 summarizes the assets and drawbacks of different hardware solutions.

Using a security module purely based on software, runtime attacks exploiting available software interfaces can be usually avoided, if an implementation as small as possible

[6] Note that pure software security mechanisms can often be broken very easily [22], and thus provide only little protection of the corresponding control unit.

Table 3: Hardware security module.

	Trusted platform module (TPM)	Customized security controller	Field-programmable gate array (FPGA)
Standardized	Yes	No	No
Flexibility	Very limited	Yes, until release	Yes, even after release
Cost	Medium	Low (high volumes)	High
Security level	High	Adaptable	Medium-high

and secure[7] is used. Runtime or online attacks are limited to use given software interfaces and, for instance, try to inject malicious code. However, hardware modifications based on manipulation, exchange, and addition of hardware components probing communication lines cannot be prevented (or even detected) by pure software security modules. Applying solutions based on a hardware security module and plausibility checks, most attacks can be at least detected.[8] Hence, the main achievements of a security module are as follows.

(i) A single security module might save code size, and hence even cost.

(ii) A solution based on a software security module is able to prevent at least runtime software attacks (such as injecting malicious code).

(iii) A solution based on a hardware security module is able to prevent software attacks and detect hardware-based attacks (such as hardware manipulation).

5. SECURITY MECHANISMS

In the following, we present mechanisms based on cryptographic methods and the security module that enable securing components and business models described in the subsequent section. We start by presenting mechanisms to ensure hardware and software integrities as well as to secure communication channels.

5.1. Hardware protection

A facile way of providing a basic protection against hardware manipulations can be achieved by mechanical countermeasures deploying special component constructions. Such special constructions could be proprietary constructions that fit only into cars of a single manufacturer or constructions that require proprietary (not publicly available) tools and equipment. However, that solution is uncomfortably and provides only minimal hardware security.

More reliable approaches [24] for detection of faked or bogus vehicle components use small computing tags attached to each crucial component in order to logically bind security and safety related parts to a specially protected central security module. Such component identification schemes rely on the tamper-evidence of the computing tags that are tightly (nonremovable) integrated into critical components that can communicate with each other and on the tamper resistance of the central security module. The component identification protocol works even without the need of a central tamper resistance security module by distributing its task to the, of course more powerful, computing tags.

To protect hardware (and particularly hardware IP) effectively, all critical hardware cores have to be integrated completely into a single protected chip. Although there are (physical and chemical) methods to comprise even such a system on a chip (SoC), these are highly sophisticated and expensive methods. Thus, today attacks on SoC hardware can comprise only small amount of data and are not applicable to large amount of hardware. However, if the outcome is worthwhile enough, for example, if an SoC contains a globally similar secret key that easily enables fraudulent manipulation on a large scale, even sophisticated and expensive attacks are feasible.

5.2. Software protection

In order to provide effective software protection,

(1) only original software must be accepted by the vehicle: no manipulated or malicious software must be downloaded to the car. In particular, no software must be successfully downloaded to the ECU that alters the defined behavior of the vehicle (e.g., due to software version conflicts);

(2) only authenticated parties are able to alter data, for example, parameters, stored in the vehicle.

Furthermore, the following is desired for an actual security design:

(i) the compromise of a single control unit does not affect the entire system, that is, a successful attack does not scale;

(ii) the required computational performance on the side of the control unit will be minimal.

A solution for this problem in general is quite simple. Based on digital signatures, the issuer of the software signs the program code and the control unit in the vehicle verifies

[7] The ideal case would be a software security module that could be verified formally.

[8] Assuming sufficient time and money, even hardware attacks cannot be prevented at reasonable cost in a vehicle. Thus, a single successful attack on a security module must not scale to also break all other security modules.

Table 4: RSA signature verification on ARM7TDMI at 40 MHz.

	Code size	Time
SHA-1 hashing	1132	680 kB/s
RSA exponentiation w/small public key	2368	11 ms
RSA verification (16 kB code)	3500	34 ms

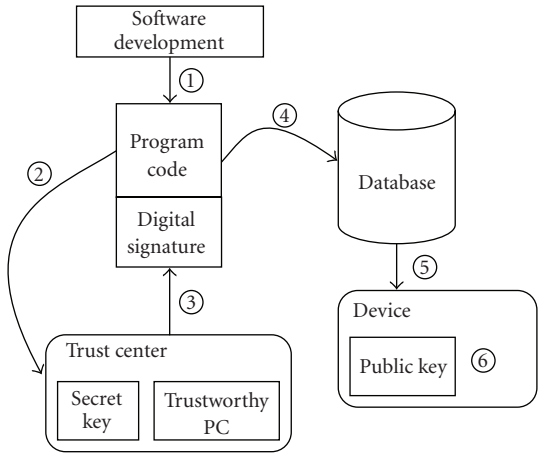


Figure 5: Secure software download.

it. Hence, the issuer holds a secret key for signing the program code and the control units hold the corresponding public key for verifying it. This is depictured in Figure 5 in more detail. First, the software is developed. Once it is finished (Step 1), the program object code is passed to a trust center (Step 2) that signs the object code using its secret key. The signature is then passed back and attached to the program object code (Step 3). The package of code and signature is now stored in a database (Step 4) that might hold versions for different control units. Finally, the appropriate program code is downloaded to a control unit (Step 5) and verified by means of the public key stored in the ECU (Step 6).

One can see that security objective (1) is clearly fulfilled: Only a legitimate authority can issue an appropriate signature for program code that will be accepted by the vehicle, that is, only authentic software will be accepted by the vehicle. Most of today's cars already provide a mechanism for downloading software. Hence, only a mechanism for verifying the signature is additionally needed in the vehicle. For signature verification, RSA is an appropriate fit since it allows very fast signature verification. This can be implemented in software. Some performance values of our implementation are displayed in Table 4. Note that for the signature verification, first the program code needs to be hashed and then an RSA exponentiation is performed. The last column displays the overall time for the signature verification at the example of 16 kB of program code, which is a typical size for a vehicle control unit.

On the server side, the key management and the organizational security must be thoroughly organized. Latter aspect includes organization of who has access to sign program code and how the process of signing is performed and recorded. However, there is no full-sized public-key infrastructure (PKI) necessary. It is sufficient to issue a single private/public key pair such that the private key is stored in the trust center and the public key in the control unit. The trust center might simply be a PC that is disconnected from any computer network and a secure smart card that holds the secret key. If a finer-grained approach is desired, a key pair for each control unit type, for each production year, or each production location might be applied. No certificates that induce overhead are required, though. The ECU only needs to store a public key such that no secret information is stored here. However, this public key must be protected from manipulation. Otherwise, an adversary could replace this key in the ECU and then induce any manipulated software.

Security objective (2) can be fulfilled by a simple challenge-and-response mechanism as presented in Section 2.6. The vehicle and an external party (e.g., a standard PC) share a secret key. The parties then run a challenge-response scheme in order to prove that the external party knows the secret key. After a successful run, the external party can access the vehicle's data. However, it is crucial that a well-defined interface is specified. For instance, it is reasonable to protocol all changes in a log file and give access only to nonsafety critical data. Using a symmetric-key management is reasonable—each ECU knows an individual symmetric secret key shared with the third party. This third party might be the car manufacturer storing all keys in a protected database.

Protecting the key of the ECU is crucial. If an adversary is able to read out or replace the key, he might be able to manipulate program or data code. Thus, virtual software protection can be achieved only by applying hardware-assisted approaches employing a security module as described in Section 4.

Nevertheless, there also exist mechanisms that try to complicate utilization or at least try to help identifying the origin in case a software could be successfully read out by an attacker. In order to make decompiling and reengineering of program binaries more difficult, programs known as "obfuscators" convert source code, object code, or both, into obfuscated code, making the result overcomplicated, and thus far less readable and almost impossible to understand by a human being. However, obfuscation [19, 25] only increases the difficulty for reverse engineering, limits portability, and is regarded as "security through obscurity." Digital watermarking [26] or fingerprinting are techniques which embed (visible or invisible) information into a digital content (software or data) that cannot or only hardly can be removed or modified. Original owners then can use tools to extract the embedded information to detect, for example, the origin of an illegitimate copy or tampering. However, these already exist technologies to abolish respective restrictions for both mechanisms. Thus, such mechanisms cannot replace a proper hardware-based software protection.

5.3. Secure communication

Until now, vehicles did communicate to the outside world only rarely. The communication channels were mainly provided for diagnosis purposes by proprietary methods. Now mobile devices, in particular cell phones, are connected to the vehicle's systems by a cradle and recently also by a wireless Bluetooth channel. The software download described above can also be seen as a communication channel. The vehicle manufacturers start to open more and more communication channels to the car such that vehicles are about to become more open. Hence, sophisticated strategies for secure communication are necessary. There are different general communication facets to consider here.

(i) *In-vehicle communication*: communication inside of a vehicle, for example, to link a mobile device to the vehicle's head unit or to allow communication between head unit and anti-lock braking system (ABS).

(ii) *Vehicle-to-vehicle communication*: communication between vehicles, for example, to exchange data about road conditions or traffic jams.

(iii) *Vehicle-to-infrastructure communication*: communication between vehicles and the infrastructure. For instance, sensors embedded in the road report icy streets and traffic lights forward their light phase.

The vehicle-to-vehicle as well as vehicle-to-infrastructure communications make possible a variety of new fascinating scenarios. For instance, cars communicate to each other in order to transmit information about imminent dangers (traffic jams, accidents, or sudden weather changes), traffic controls, or free parking spaces. In addition, it would be possible to group together cars on a highway and make driving more comfortable and safe. Digital contents such as multimedia files could be downloaded to a car when connected to the Internet (e.g., at a gas station) and navigation data could be updated at the same time. The license plate can be replaced by an electronic license plate that transmits the license number such that it can be used for various purposes, for example, wireless payments or automatic gateway access control.

There are various requirements to implement such scenarios. These comprise both technical and organizational aspects as follows.

(i) *Message integrity*: altered messages must be detected by the vehicle.

(ii) *Message authenticity*: origin of a message from a valid source must be verifiable by the vehicle.

(iii) *Privacy*: vehicles must not endanger privacy of the driver and owner. For instance, a GPS receiver or an electronic plate must not be used to track a car. On the other side, it is often desirable to allow authorities to access this data in a well-defined way.

(iv) *Efficiency*: if cryptographic algorithms are involved, these must often run extremely fast to allow real-time behavior. For instance, a warning message about an imminent danger must be processed immediately.

Secure communication is mainly based on encryption and authentication in order to provide confidentiality and authenticity of exchanged data. While authentication is necessary in most scenarios in order to make sure that there are no malicious messages induced to the network, confidentiality might often be less important. For instance, a warning message should be authentic but not confidential. Clearly, a vehicle should implement a balanced security policy in such a way that it reacts in a reasonable way based on external input, internal input (such as internal sensors), and most important the drivers decision. Based on the external infrastructure, protocols that are more sophisticated can be implemented. For instance, location-based protocols [27] as well as time-based protocols can be implemented.

There is a variety of literature available about secure communication. See [28] for in-vehicle communication and [27] for car-to-car and for car-to infrastructure communications. Furthermore, there are a variety of standardizations and on going projects dealing with such topic such as Car-2-Car Communication Consortium [29], Network on Wheels (NOW) [30], CIVS [31], SAFESOPT [32], and IEEE 1609.2 [18].

6. SECURITY-CRITICAL VEHICULAR APPLICATIONS

Several vehicular applications provide security features or are security relevant. These applications are usually based on a security module and security mechanisms as described before. In order to properly realize such a security-critical vehicular application the following steps have to be done:

(1) analysis of valuable attacking targets and all relevant attackers to create the corresponding attacker model (cf. Section 3.2);

(2) establishing the corresponding security objectives (cf. Section 3.3);

(3) design of a proper security module(s) capable to successfully fend off the attacker model and fulfill the security objectives derived before (cf. Section 4);

(4) implementation of the required security mechanisms, based on the afore-designed security module(s), the security-critical application builds on (cf. Section 5).

We would like to point out that once a security module and the corresponding security mechanisms have been implemented properly, this base can be easily reused to protect also other future security-critical applications[9] with almost no additional cost.

In the following, we give an overview about current and future security-critical vehicular applications that can be protected in this way.

[9] Provided that the added application has the same attacker model.

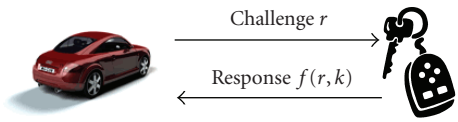


Figure 6: Electronic immobilizer.

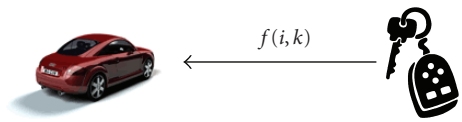


Figure 7: Remote key.

6.1. Electronic immobilizer

The electronic immobilizers as well as the keyless entry to a vehicle are probably the oldest applications of cryptography in vehicles. The electronic immobilizer usually works in the following way. The vehicle sends a challenge to a passive batteryless transponder integrated in the vehicle key, which then answers by a response. Transponder and vehicle share a secret key. Only if the transponder knows the secret key, then the vehicle will start. Hence, a vehicle's key that has the appropriate physical properties (i.e., that is an exact physical copy of the original key) but does not know the secret cryptographic key will not make the vehicle starting. This is depictured in Figure 6. Here, f is a cryptographic function such as a keyed hash function that takes as input the challenge r as well as a key k and returns the response. A general approach for an electronic immobilizer was presented in [33].

For the electronic immobilizer, attacks at the hardware layer must be considered. Such a hardware attack can never be prevented at reasonable cost. The goal is to make such an attack impossible for a rational attacker, that is, the cost of an attack will exceed the gain of the stolen car on the black market. Hence, the goal is to achieve an economic security. A hardware security module is an appropriate platform to provide such security goals. It is able to securely store the secret keys and to bind the key's transponder to the vehicle by means of the security module. The immobilizer binds a crucial control unit (usually the engine control unit) to the vehicle's key. Hence, the engine control unit is only activated if the proper key is presented. There are several weaknesses though. The crucial control unit can simply be replaced by one that is always activated, that is, by one that implements exactly the same functions but without the key verification. Hence, avoiding malicious software updates of the firmware is absolutely necessary. Furthermore, a so-called Mafia attack is possible. Here, the vehicle's signal is forwarded over an external channel (say, a wireless LAN) to the vehicle's key. This is in particular dangerous in combination with a keyless entry system where an adversary establishes a channel between the victim's vehicle and the victim such that the vehicle's doors unlock and the engine starts. Usually, once the vehicle starts, the engine will not turn off even if the key's signal is lost.

The later attack can hardly be avoided on a cryptographic layer. A countermeasure is to use the so-called distance bounding (see, e.g., [34]). Here, the protocol can make sure that the vehicle's key is inside of a well-defined geographical area. However, due to timing problems and wavelength of the transponder, this might be too imprecise. Further countermeasures can be provided on the physical level. For instance, multifrequency hopping is already applied today for such. Clearly, each electronic immobilizer can be compromised. However, the objective is not to set up a perfectly secure system, but an economical secure system—breaking a single vehicle should be more expensive than the gain of the attacker.

6.2. Keyless entry

The remote key entry works in a similar way as the electronic immobilizer. Here the key is equipped with an active battery-powered transponder. When pushing the button of the vehicle's remote control key, the vehicle will unlock the doors. Therefore, the so-called rolling codes are used. Both vehicle and key share a secret. Each time the key's button is pushed, a new secret is derived by the key and sent to the vehicle. The vehicle can compute the same transition and compare the two values. If they are equal, the vehicle unlocks its doors. Certainly, the key's button might be pushed several times. The vehicle then repeatedly computes the internal value and compares it. It repeats this, say, a hundred times before giving up. This is shown in Figure 7. The remote control as well as the car hold a counter i that is increased by one after each application.

Modern cars are equipped with a keyless entry system. Here, the driver carries a key-card in his pocket. Once he approaches the vehicle, the doors are unlocked. This system usually works as follows. The protocol starts when the door handle is touched. The vehicle then transmits a signal. Once the key card approaches the car, it will be detected by the vehicle and then responds to the car. Then a cryptographic challenge-response method is carried out. Note that the key card usually is comparable to a passive or battery-powered radio frequency identification (RFID) tag.

Attacks on the remote key entry and keyless entry system are located at the protocol and physical transmission layers. An attacker might try to compromise the secret key or to replay a message in such a way that it unlocks the doors. However, it can be assumed that there is no physical breach—an adversary could otherwise just smash a window. Hence, such systems can be designed in a secure way using traditional cryptographic schemes. However, attacks on the physical transmission layer such as the Mafia attack have to be considered carefully since they are inherent for any such scheme and can hardly be prevented by cryptographic means.

6.3. Digital tachograph and event data recorder

Digital tachographs and the so-called event data recorders are a well-considered security relevant component in

vehicles. Manipulation of tachographs has a serious safety and economic impact. Today, it is assumed that about a third of all used cars was manipulated regarding the tachograph counter, for example, in order to achieve higher prizes on the used markets. In the case of trucks, the attacker usually is the truck driver or the owner, who tries to circumvent rest periods, speed limits, and law regulations. However, recently several law regulations were introduced to stop such misbehavior. For truck tachographs, there was a European law introduced [35] concerning the required security level according to the ITSEC security standard and specification of the involved processes. This leads to several security certified truck tachographs. Furthermore, for vehicles in Germany, a law was introduced making any change of the tachograph counter liable to penalty.

The attacker of this system is usually the owner (company) or the driver of the vehicle. Hence, he has full physical access to any component and unlimited time for an attack. An attack on the hardware level is usually performed and the goal is to achieve an economic gain. For standard personal vehicles, it is almost impossible to prevent a manipulation at reasonable cost, whereas for high-cost trucks it is possible up to a certain point. In both cases, the first objective is to detect a manipulation though. Hence, some kind of security module is required. This might even involve a security controller such as a smart card controller as described by the European law regulation for truck tachographs.

Each truck is equipped with a motion sensor that receives input of the gearbox and transmits signals to the tachograph on an encrypted channel. The main objective is to record the truck drivers' behavior and working hours. The European law enforces that every truck driver uses a personalized smart card when driving the truck by inserting it to the digital tachograph. Clearly, privacy is a crucial aspect here. Hence, there are four kinds of smart cards provided. Besides the truck driver's card, there is a card for each company that owns trucks, for workshops that maintain the trucks, as well as for police authority. The smart cards are issued with keys that are organized by European wide key management hierarchy.

The security of the system can only be provided by a combination of technical and organizational means. For instance, it is possible to manipulate the gearbox such that the motion sensor receives false input. Therefore, integrity of the motion sensor and the gearbox must regularly be verified by police authorities. Once again, attacks that manipulate hardware components cannot be avoided by technical means, but they can be detected by a combination of technical and organizational means.

6.4. Counterfeit and expertise protection

Today, large amounts of OEM's capital investments are spent on software and electronic development [1] that—without further protection—can be simply copied, analyzed, and reused by simply buying the corresponding components or vehicles. Thus, reliable counterfeit and intellectual property (IP) protection should prevent copyright infringement or expertise theft by potential competitors and particularly to prevent mass production of unauthorized counterfeits of vehicle components.

(i) *Counterfeit protection*: illegal produced replacement parts cause a worldwide loss of about 3 billion dollars [36] per month. The professional organized manipulation of automotive electronics [37] causes considerable damage to the manufacturers and to the economics by unwarranted claims, brand damage, and undermined business models. Moreover, counterfeits endanger the safety of all motorists and cyclists. Traditional methods to prevent counterfeits use tags, for example, holographic stickers that are supposed to be unforgeable. However, there exist illegal businesses that create boxes, labels, and other significant trademark logos and emblems to let counterfeits look like real parts [38].

(ii) *IP and Expertise protection*: automotive OEMs and suppliers always have a comprehensible interest to find out valuable expertise from their potential competitors. Moreover, even though intellectual property rights are legally effective in most countries in the world, there exist large domestic markets, such as China, where IP thefts and infringements are virtually nontriable. Therefore, expertise leakage and IP theft are a serious problem. Today mostly for software and firmware, but even complex hardware, can be copied when it is profitable enough. Expertise leakage and IP theft have to be tackled primarily applying organizational security measures such as scrutinizing potential partners and preventing employees from unintentional (or intentional) exposures. However, there exist also (cryptographic) technologies that can help protecting IP and expertise or making a theft or leakage at least detectable.

6.5. After-sale business applications

Embedding security in vehicles enables various new and interesting business models previously not possible. Particularly, it enables business models where all involved parties (OEMs, suppliers, and customers) can benefit from. In the following, we present three exemplary business models made possible by progress in vehicular security.

6.5.1. Feature activation

The production of vehicular components moves from various small charges of different individually adjusted components towards large-scale production of only a small number of uniform standard components. Thus, today many of the various vehicle versions internally consist mostly of the same components. On the other hand, providing manifold individual vehicle configurations is crucial even now. To solve

this opposing requirements, car manufacturer could build parts identical in construction cost-efficiently with most features already built-in, but individually activated. Moreover, it is possible to individually activate (or deactivate) built-in hardware components or software after sales for an additional charge that would furthermore bind the customer long term to the OEM. Features that would be capable for after-market activation could be, for instance, special setups for engine, gear or chassis control, enhanced board computer and comfort diagnosis functions, additional driving assistance and infotainment capabilities, or certain personalization and individualization features. However, capable security measures are required to prevent unauthorized feature activation that may undermine the underlying business model.

6.5.2. Infotainment

Maybe the most exciting new applications in the automotive are driven by new infotainment business models distributing digital content. The area ranges from individual software upgrade packages, OEM premium content, newscasts up to various multimedia files including music, video, or games. Today, already most medium-sized cars are equipped with multimedia capable on-board computers and radio systems. Upcoming integrated wireless broadband communication promises a brisk market for automotive-related on-demand sales. Embedding a reliable digital rights management (DRM) enables business models for usage-metered and on-demand utilization of digital contents, software, and even hardware beyond the classical lump-sum model. Some possible examples are provided below.

(i) *Time-limited utilization*: up-to-date navigation and traffic data may be available on demand for any place in the world (e.g., only for a two-week vacation trip in the respective area).

(ii) *Quantity-limited utilization*: movies, music tracks, or games can be bought for an n-times repeated utilization.

(iii) *Device-bound utilization*: extra software can be installed on a particular device or a particular vehicle only. Certain car functions are performed only via a certain authentication device such as a driver's key, dealer token, or personal cellular.

(iv) *Usage-metered utilization*: navigation routes can be charged for their actually used length. Movies or music tracks can be charged for the actual viewing time.

(v) *Subscription services*: audio, video, or information broadcast services can be received as long as a valid subscription to the corresponding service exists.

Furthermore, almost arbitrary combinations are possible. For instance, an afterwards activated enhanced comfort sensor (e.g., tire air pressure sensor) may be enabled as free sample for 4 weeks. Business models using digital content that has usage or access restrictions are only possible with a secure and reliably implemented DRM system. As it could be seen in various (nonautomotive) DRM scenarios such as pay-TV, online music stores, or video game consoles, having no such secure module, the business model will certainly fail.

6.6. Location-based services

Offering services based on the vehicle's current location provided by a built-in GPS or GSM receiver together with a wireless communication device enables various safety, management, and business applications.

(i) *Automatic emergency call (eCall)*: the first popular location-based service would probably be the automatic emergency call, mandatory for all new vehicles within the European Union from 2009. As proposed by the eCall Driving Group [39], in case of emergency the eCall system establishes a voice connection directly to a call center initiated either manually by vehicle occupants or automatically via activation of in-vehicle sensors. At the same time, actual location, available incident, or medical data will be sent to the eCall operator receiving the voice call. To address privacy issues and prevent potential misuse, an eCall system requires mechanisms for secure authorization and confidential transmission.

(ii) *Location-based information*: location-based information services might for example allow the driver to find the nearest business of a certain type, for example, the next fuelling station, the next ATM, or accommodations and restaurants available in the immediate vicinity. Optionally, the driver might allow certain location-based incoming information such as traffic news, local objects of interest, or localized advertisements. To prevent potential misuse, we need a secure authentication and authorization for all incoming messages. Outgoing queries, however, require adequate protection of the driver's privacy.

(iii) *Location-based billing*: having the current vehicle position would also enable certain automatic vehicular billing applications, for example, for toll roads or parking. Then drivers could securely pay by a simple acknowledgment within their car while the operating company or authority would not need to maintain an expensive billing infrastructure. This scenario, however, again needs efficient and reliable cryptographic mechanisms to mutually ensure payments while protecting the driver's privacy. Furthermore, only secure positioning could reliably enable advanced applications such as restricted areas of operation or upcoming pay-as-you-drive insurances.

(iv) *Fleet management*: modern fleet management systems enable, in addition to vehicle tracking, advanced functionality such as centrally managed routing and efficient dispatch, driver authentication, remote diagnosis while gathering details on current driver's status, mileage, fuel consumption, or container status. Therefore, a fleet management system demands for mechanisms to establish secure connections to the

vehicle's onboard computer and requires appropriate mechanisms to provide in-vehicle driver authentication and authorization.

6.7. Legal authority support

Support for various vehicular legal authority applications could become a crucial impulse for automotive electronics. Since official applications very often involve sensitive personal information, they often demand appropriate IT security measures to become enabled. In the following, some feasible applications for legal authority support in the automotive area are listed.

(i) *Electronic road signs*: there already exist developments [40] for dashboards with integrated traffic sign recognition systems that will warn the driver whether he is driving too fast. Since present traffic sign recognition systems still process digital images of their environment, future approaches will also integrate electronic road signs that wirelessly transmit their (variable) information about actual speed limits, road works, traffic jams, or road conditions. However, to detect bogus or faked information, the vehicle requires an appropriate IT security architecture to reliably verify incoming traffic sign information for validity.

(ii) *Electronic license plate*: integrating a wireless transponder into a vehicle that broadcasts an (unique) identification string will be another promising automotive development. Such an electronic license plate could for instance help to easily implement tolling and payment systems, or particularly help police forces and public authorities to identity a vehicle in case of accident or law violation. However, an adversary could modify or just steal an electronic license plate for misinformation or impersonation. Drivers in turn, require that toll road stations or an arbitrary road user cannot acquire the same amount of information as for instance qualified police forces. Thus, the application of electronic license obviously requires an adequate vehicular IT security architecture that regards attackers from both outside and inside.

(iii) *Electronic log books*: providing evidence for accomplished trips or critical maintenance operations can be very important for legal restraints, commuting accounts, or warranty claims. Having an integrated electronic service check book and/or driver's log would clearly ease bookkeeping and provide reliable information. However, both demand appropriate manipulation and privacy protection.

7. CONCLUSION: CHALLENGES AND OPPORTUNITIES FOR THE AUTOMOTIVE IT COMMUNITY

In this contribution, we presented a state-of-the-art overview of IT security in vehicles. After a short introduction to cryptographic terminology and functionality, we identified the need for automotive IT security while presenting the specific attackers and attacks within the automotive area. We introduced core security technologies and relevant security mechanisms required to protect current and future vehicular applications, business models, and components that rely on IT security. In summary, it can be stated that embedding IT security in vehicles.

(1) protects against manipulations by outsiders, owners, and maintenance personnel;
(2) increases the safety and reliability of a vehicular system;
(3) enables new IT-based automotive applications and business models.

As sketched above, there are several difficulties to overcome in order to develop strong embedded security solutions. We would like to give an outlook on the future of IT security in cars in the form of the following recommendations and conclusions.

(i) IT security will be a necessary requirement for many future automotive applications.
(ii) IT security will allow a multitude of new IT-based business models, for example, location-based services or fee-based flashing. For such systems, security will be an enabling technology.
(iii) IT security will be integrated invisibly in embedded devices. Embedded security technologies will be a field in which manufacturers and part suppliers need to develop expertise.
(iv) IT security solutions have to be designed extremely carefully. A single "minor" flaw in the system design can render the entire solution insecure. This is quite different from engineering most other technical systems: a single nonoptimum component usually does not invalidate the entire system. An example is the content scrambling system (CSS) for DVD content protection, which was broken easily once it was reverse-engineered.
(v) Embedded security in vehicles has to deal with very specific boundary conditions: computationally and memory-constrained processors, tight cost requirements, and physical security.
(vi) The multitier manufacturing chain for modern vehicles (OEM and possibly several layers of suppliers) can have implications for the security design. It is, for instance, relevant who designs a security architecture and, most importantly, who has control over the cryptographic keys.
(vii) Merging the automotive IT and the embedded security community will allow many new applications. However, there are also several challenges: security and cryptography have historically been a field dominated by theoreticians, whereas the automotive IT is usually done by engineers. The culture in those two communities is quite different at times, and both sides have to put effort into understanding each other's way of thinking and communicating.

REFERENCES

[1] A. Saad and U. Weinmann, "Automotive software engineering and concepts," in *GI Jahrestagung*, pp. 318–319, Frankfurt, Germany, September-October 2003.

[2] E. Nickel, "IBM automotive software foundry," in *Press Conference on Computer Science in Automotive Industry*, Frankfurt University, Frankfurt, Germany, September 2003.

[3] ISO/IEC, "Information technology—guidelines for the management of IT security—part 1: concepts and models for IT security," Tech. Rep. TR 13335-1, ISO/IEC, Genf, Switzerland, 1996.

[4] R. Shirley, "Internet security glossary," Tech. Rep. RFC 2828, GTE/BBN Technologies, Cambridge, Mass, USA, May 2000, http://www.rfc-editor.org/rfc/rfc2828.txt.

[5] M. Bishop, *Computer Security: Art and Science*, Addison-Wesley, Reading, Mass, USA, 2003.

[6] W. Stallings, *Cryptography and Network Security*, Prentice-Hall, Englewood Cliffs, NJ, USA, 4th edition, 2005.

[7] National Institute of Standards & Technology, "FIPS-46-3: Data Encryption Standard (DES)," October 1977, reaffirmed in October 1999.

[8] National Institute of Standards & Technology, "FIPS-197: Specification for the Advanced Encryption Standard (AES)," November 2001.

[9] J. Daemen and V. Rijmen, "AES proposal: rijndael," in *Proceedings of the 1st Advanced Encryption Standard (AES) Candidate Conference*, Ventura, Calif, USA, August 1998.

[10] G. S. Vernam, "Cipher printing telegraph systems for secret wire and radio telegraphic communications," *Journal of the American Institute of Electrical Engineers*, vol. 55, pp. 109–115, 1926.

[11] C. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[12] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[13] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[14] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[15] V. Miller, "Uses of elliptic curves in cryptography," in *Advances in Cryptology (Crypto '85)*, H. C. Williams, Ed., vol. 218 of *Lecture Notes in Computer Scienc*, pp. 417–426, Springer, Berlin, Germany, 1986.

[16] IEEE P1363-2000, "Standard Specifications for Public Key Cryptography," http://standards.ieee.org/catalog/olis/busarch.html.

[17] T. Miehling, B. Kuhls, H. Kober, H. Chodura, and M. Heitmann, "Security module specification," Tech. Rep., HIS-Herstellerinitiative Software, Bochum, Germany, July 2006, Version 1.1.

[18] IEEE 1609.2-2006, "Trial-Use Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages," http://ieeexplore.ieee.org/servlet/opac?punumber=11000.

[19] R. Rivest, "RFC 1321: the MD5 message-digest algorithm," April 1992, http://www.ietf.org/rfc/rfc1321.txt.

[20] National Institute of Standards & Technology, "FIPS-180-2: secure hash standard (SHS)," August 2002.

[21] U.S. Department of State, International traffic in arms regulations (ITAR), code of federal regulations, title 22, parts 120–130.

[22] P. van Oorschot, "Revisiting software protection," in *Proceedings of the 6th International Conference on Information Security (ISC '03)*, vol. 2851 of *Lecture Notes in Computer Science*, pp. 1–13, Bristol, UK, October 2003.

[23] S. Amendola, "Improving automotive security by evaluation—from security health check to common criteria," Tech. Rep., Security Research & Consulting GmbH, Bochum, Germany, 2004.

[24] A. Weimerskirch, C. Paar, and M. Wolf, "Cryptographic component identification: enabler for secure vehicles," in *Proceedings of the 62nd IEEE Vehicular Technology Conference (VTC '05)*, pp. 1227–1231, Dallas, Tex, USA, September 2005.

[25] C. Linn and S. Debray, "Obfuscation of executable code to improve resistance to static disassembly," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pp. 290–299, Washington, DC, USA, October 2003.

[26] C. S. Collberg and C. Thomborson, "Watermarking, tamper-proofing, and obfuscation—tools for software protection," *IEEE Transactions on Software Engineering*, vol. 28, no. 8, pp. 735–746, 2002.

[27] J.-P. Hubaux, S. Čapkun, and J. Luo, "The security and privacy of smart vehicles," *IEEE Security & Privacy Magazine*, vol. 2, no. 3, pp. 49–55, 2004.

[28] M. Wolf, A. Weimerskirch, and C. Paar, "Security in automotive bus systems," in *Proceedings of Embedded Security in Cars Workshop (ESCAR '04)*, Bochum, Germany, November 2004.

[29] Car-2-Car Communication Consortium. http://www.car-2-car.org/.

[30] Network on Wheels, http://www.network-on-wheels.de/.

[31] CVIS—Cooperative Vehicle-Infrastructure Systems. http://www.cvisproject.org/.

[32] Safespot, Cooperative vehicles and road infrastructure for road safety, http://www.safespot-eu.org/.

[33] K. Lemke, A.-R. Sadeghi, and C. Stüble, "An open approach for designing secure electronic immobilizers," in *Proceedings of the 1st International Conference on Information Security Practice and Experience (ISPEC '05)*, pp. 230–242, Singapore, April 2005.

[34] S. Čapkun and J.-P. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 3, pp. 1917–1928, Miami, Fla, USA, March 2005.

[35] European Commission. EU NO 1360/2002, June 2002, Corrigendum to commission regulation adapting for the seventh time to technical progress council regulation (EEC) no 3821/85 on recording equipment in road transport.

[36] Gieschen Consultancy, Report: IP theft up 22%, massive $3 trillion counterfeits, May 2005, http://www.bascap.com/.

[37] R. J. Anderson, "On the security of digital tachographs," in *Proceedings of the 5th European Symposium on Research in Computer Security (ESORICS '98)*, pp. 111–125, Springer, Louvain-la-Neuve, Belgium, September 1998.

[38] S. Ross, "Parts counterfeiting," October 2004, http://www.aftermarketbusiness.com/aftermarketbusiness/article/articleDetail.jsp?id=125346.

[39] eCall Driving Group, http://ec.europa.eu/information_society/activities/esafety/forum/ecall/index_en.htm.

[40] Siemens VDO, Traffic sign recognition. http://www.siemensvdo.com/products_solutions/cars/propilot/.