Intelligent extended floating car data collection

Stefano Messelodi\textsuperscript{a}, Carla M. Modena\textsuperscript{a,}\textsuperscript{*}, Michele Zanina\textsuperscript{a}, Francesco G.B. De Natale\textsuperscript{b}, Fabrizio Granelli\textsuperscript{b}, Enrico Betterle\textsuperscript{c}, Andrea Guarise\textsuperscript{c}

\textsuperscript{a} FBK-irst, Via Sommarive 18, I-38100 Povo, Trento, Italy
\textsuperscript{b} Università degli Studi di Trento, Via Sommarive 14, I-38100 Povo, Trento, Italy
\textsuperscript{c} Centro Ricerche FIAT, Via dei Solteri 38, I-38100, Trento, Italy

\begin{abstract}
The elaboration of data collected by vehicles moving on road network is relevant for traffic management and for private service providers, which can bundle updated traffic information with navigation services. Floating data, in its extended acceptation, contains not only time and location provided by a positioning system, but also information coming from various vehicle sensors. In this paper we describe our extended data collection system, in which vehicles are able to collect data about their local environment, namely the presence of roadworks and traffic slowdowns, by analyzing visual data taken by a looking forward camera and data from the on-board Electronic Control Unit. Upon detection of such events, a packet is set up containing time, position, vehicle data, results of on-board elaboration, one or more images of the road ahead and an estimation of the local traffic level. Otherwise, the transmitted packet containing only the minimal data, making its size adaptive to the environment surrounding the vehicle.
\end{abstract}

\section{Introduction}

Acquisition of road traffic data is a crucial and necessary activity for a traffic management information system. The term floating car data (FCD) refers to the data being collected (continuously) by a fleet of vehicles, which can be considered as a distributed network of sensors, i.e. FCD is an embedded traffic measurement system. The method was introduced to determine the average travel time in certain links of the road network. In its basic form it relies on the collection of position of vehicles at constant time intervals. The early methodology (Turner, Eisele, Benz, & Holdener, 1998), in late 1920s, was performed by personnel within the test vehicles, called \textit{probes}, that manually recorded travel times at designated checkpoints or intervals.

The technological progress in positioning and communication systems (GPS and GSM or GPRS), made possible the extension of the method to fleets of appropriately equipped vehicles acting as moving sensors for the road network. They provide basic data to a traffic control center where they are analyzed and merged to identify traffic congestion, calculate travel times and generate reports on traffic (Turksma, 2000).

Nowadays, technology permits to collect, transmit and elaborate enhanced additional information acquired by specific devices a modern vehicle is endowed with. Such data can be exploited to obtain hints on traffic status (travel time, queue) and environmental condition (road, weather) at certain locations (Huber, Lädke, & Ogger, 1999). The integration of data coming from the different vehicles supports the creation of updated traffic status maps (Kern-er & Rehborn, 2005; Sarvi et al., 2003), and the realization of high quality services for the users, for example to determine an alternative route in case of traffic congestion on the planned one. The enhanced FCD is now considered as a central element in extending the \textit{information horizon} for improved telematic services and it is recognized as a reliable methodology to update traffic related map, as witnessed by many research projects in this field (Escher, 2005; refer to Bishop (2005) for a list of activities till 2004).

We present the extended floating car data (xFCD) service which has been realized within the DIPLODOC\textsuperscript{1} project. The main result of the project has been the design and implementation of a distributed architecture composed by a set of moving platforms, endowed with sensors and processing units, connected through a wireless channel to a Main Center. The distributed architecture paradigm permits to balance the on-board computational load for management of reactive behaviors, the usage of the communication channel and the load for tasks performed in the Main Center.

The architecture of the xFCD service has been realized through software modules living on the moving platforms which take input from on-board sensors, and are governed and coordinated by a

\footnotesize* Corresponding author. Tel.: +39 0461 314 516; fax: +39 0461 314 591. E-mail address: modena@fbk.eu (C.M. Modena).

\footnotesize\textsuperscript{1} Distributed Processing of Local Data for On-Line Car Services, PAT FU Project 2001–2005, funded by Provincia Autonoma di Trento, Italy.
service manager. The on-board modules sense certain events in the local environment and, when needed, send enriched data to the Main Center. Here, larger computational and storage resources support a more in-depth elaboration of the data and the merging process of information coming from multiple platforms and other sources in order to gather a global knowledge of the road environment. This information is delivered, upon request, to the moving platforms which can benefit of the global knowledge they themselves contribute to build.

Each platform is endowed with “intelligence”, i.e. it is able to detect some critical situations, namely roadworks presence and high traffic level. In particular, a vision module receives its input from a color camera (shared with other on-board services) and reacts on temporary danger warning signs, typically placed at roadworks sites. When those signs are detected an alarm triggers off and some frames are stored in a xFCD packet, along with vehicle data, time and position. The same happens when a second on-board module, a traffic level estimator, detects a traffic level exceeding a certain threshold. Its estimates are based on the analysis of data coming from the vehicle ECU (Electronic Control Unit) and, when slow traffic is detected, verified by a vision based module which computes the distance from the preceding vehicle. If no critical situation occurs, the packets sent to the Main Center contain only the minimal data, to avoid the transmission of unneeded information. The packet composition, and therefore its dimension, is adaptive to the status of the environment surrounding the vehicle. The major novelty of our system relies in the use of visual information to improve the automatic detection of potentially interesting events and to document them by sending extended data (containing images and results of on-board processing) to the Main Center.

The paper is organized as follows. In the next section we briefly refer to other related systems and projects, highlighting the novelty of our system. Section 3 describes the different data collection policies upon which the behavior of the on-board service manager relies. Section 4 presents the on-board service architecture, introducing the involved modules and their connections. The algorithm for a quick detection of temporary danger warning signs is described in Section 5. The traffic level estimator based on the analysis of vehicle parameters and on the test on the presence of a vehicle ahead is described in Section 6. Architecture and techniques for data transmission are referred to in Section 7. Section 8 reports the experiments for evaluating the algorithms for the automatic event detection in terms of correct alarms, as well as processing time. Finally, Section 9 concludes the document.

2. Comparison with related works

In the case of basic FCD collection, on-board equipment records time and vehicle location provided by a GPS receiver, and transmits them to the central system along with a vehicle identifier. Provided a sufficient number of probe vehicles, sent data, this can be used to detect actual congestion and estimate the traffic flow speed (Fastenrath, 1997). A description of several FCD projects can be found in Bishop (2005). We are interested in focusing on the so called second generation FCD system, i.e. designed for the collection of extended data.

The probe car information system developed by JARI, Japan Automotive Research Institute, receives instructions from a data center and transmits relevant probe data gathered from car sensors and stored on-board. The message contains information on position, wiper activity, speed, fuel consumption, engine revolutions per minute, turn signals. The system has been tested using taxi fleets (Horiguchi & Wada, 2004). The research is getting on inside WIDE (Widely Integrated Distributed Environment) framework with the iCAR (Internet CAR) project (Sato, 2005), BMW Group (BMW AG, 2005; Breitenberger, Grüber, Neuhertz, & Kates, 2004; Hauschild, 2005) has developed an extended floating car data system (XFCD), where information about the local traffic status and the weather condition are collected by on-board modules. During the journey, the software module processes various vehicle information, such as speed, brake activity, data from the stability control system, headlight and wiper status, and data from the navigation system. The on-board program uses this data to estimate the local traffic and weather situation and, if it is the case, warns other cars on the same route and the traffic center about traffic jams or aquaplaning danger, for example. Furthermore, collected information is made available to all road users through the Internet.

Global System for Telematics (GST) is an EU-funded Integrated Project (2004–2007) that is creating an open and standardized end-to-end architecture for automotive telematics services. Three service-oriented telematic subprojects are considered in the GST project: RSQ about emergency situations, SAF-CHAN for dynamic communication relevant to traffic, road and weather conditions, and EFCD for the collection of enhanced floating car data (Escher, 2005).

The xFCD telematic service we have implemented on the general DIPLODOC architecture relies, in many aspects, on the same concepts used in JARI, BMW and GST projects. The main difference is that DIPLODOC includes a vision-based driver assistance service which makes available results of the visual analysis of the scene ahead to the xFCD service too.

The benefit brought to our xFCD service by the usage of on-board cameras and image analysis modules is manifold: it permits to detect specific road situations in the scene ahead which can trigger off the data collection; it makes possible the transmission of images, or short videos, to document the detected events; it allows to verify the alarms generated by an on-board traffic module, by detecting, for example, if a vehicle stop is due to a red traffic lamp or a grade crossing, or if a low speed is due to a tractor ahead. In our implementation we have chosen to exploit the visual information in two on-board modules: a local traffic level estimator and a roadworks detector.

The traffic module is based on the analysis of the temporal distributions over the last minutes of GPS data (position, time, direction) and ECU data (speed, gear, breaks, revolutions per minute, and so on). When the analysis suggests, with a certain confidence level, a traffic slow down, the on-board xFCD manager checks if the slow down is intentional or not, by asking to an image analysis module if there is a vehicle ahead and its distance. If the traffic slow down hypothesis is confirmed, a xFCD packet is built containing GPS and ECU data, along with some still images that will be analyzed in the Main Center to determine possible causes of the slow down.

The on-board roadworks detector is a vision module that analyzes the images coming from a color camera in order to detect if a temporary danger warning sign is present in the scene. The occurrence of a minimum number of consecutive detections, causes the creation of a xFCD packet containing the start and end time of the detection sequence along with some documenting images. In the Main Center these images are processed in a more in-depth way (reflective elaboration), in order to confirm the presence of the danger sign and possibly to search for other typical patterns in roadworks sites, like barriers and fences.

3. Policy for xFCD collection

In the Main Center the operator, or the automatic system devoted to traffic monitoring and control, establishes the most appropriate collection strategy and communicates it to the floating vehicles.
Specific strategies will be chosen for different zones and in different time, aiming to:

– Guarantee a good coverage of the road network.
– Maintain an up-to-date picture of the traffic conditions.
– Monitor the evolution of critical sections (previously detected).
– Resolve conflicting signals coming from different vehicles.
– Minimize the amount of data each vehicles has to transmit to the Main Center.

Our xFCD service implements three collection modalities: temporal, regional, and on-event.

**Temporal modality (T):** requires the collection of a data packet every \( \delta_T \) seconds, where \( \delta_T \) is a parameter of the policy.

**On-Event modality (E):** the packets are requested only when certain events are detected by the on-board modules. At present, two kind of events have been taken into account: traffic and roadworks. The on-event policy requires a parameter \( \delta_E \) specifying the time interval between two successive collections.

**Regional modality (R):** the packets are requested only when the vehicle position is inside a specified region (a circle defined by the center position \( x, y \) and the radius \( r \)). A parameter specifying the time interval \( \delta_R \) is required.

The policy specifies not only the collection modality but also the amount of data to be stored in the xFCD packets. Three different levels of data content are provided: basic information about time and spatial location (B), information about vehicle status (V), and visual information (images) or results of on-board processing (X), as illustrated in Table 1.

<table>
<thead>
<tr>
<th>Type</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>B – Base information</td>
<td>Vehicle ID, time and GPS data: position, speed, direction</td>
</tr>
<tr>
<td>V – Vehicle information</td>
<td>Base information + ECU data: gear, brake, clutch, sidelight, etc.</td>
</tr>
<tr>
<td>X – Extended information</td>
<td>Vehicle information + one or more video frames related to latest seconds; estimated traffic level</td>
</tr>
</tbody>
</table>

The type of requested data should be carefully selected in order to avoid the transmission of unneeded information.

The collection strategy is encoded in a sequence of strings which can have one of the following forms:

\[
\begin{align*}
T & \delta_T X/V/B/Id \quad (1) \\
E & \text{roadworks}/\text{traffic} \delta_E X/V/B/Id \quad (2) \\
R & x \ y \ r \ \delta_R X/V/B/Id \quad (3)
\end{align*}
\]

The identifier Id specifies the policy and is used to deactivate a previously specified policy by sending the string Id. Moreover, it is inserted in each xFCD packet to indicate which policy caused the transmission of the packet itself.

### 4. On-board xFCD service architecture

In this section we describe the on-board architecture of the xFCD service we have designed and implemented. The core is a Service Manager that manages data collection and transmission according with the policy established by the Main Center, through the activation of the proper modules and the coordination of their actions. It aims at optimizing the usage of processing and communication resources, for example the on-board vision module for the roadworks detection is executed only when the collection modality is ‘on-event’ and the specified event parameter is ‘roadworks’.

In the temporal collection mode, the Service Manager triggers off data collection at the arranged time. In the case of an on-event strategy, it waits for alarms from the traffic level estimator or the roadworks detector to determine when to trigger off the data collection.

The main modules of the on-board architecture and their interactions are depicted in Fig. 1 and are briefly described in the following.

**Car-Data Server** comprises two modules that provide information coming from the GPS and from the vehicle control unit, respectively. They store more recent data in circular buffers and deliver them upon request, after a filtering step.

**Image Server** manages the acquisition of images from a stereo color camera. It acquires and stores the image pairs into a circular buffer. When a client module needs images, it send a request to the Image Server specifying which image it requires (left, right or both), the desired resolution and frame rate.

**xFCD Service Manager** manages the xFCD collection. It receives the policy from the Main Center through the Communication Module. It controls the activation and the configuration, at proper time, of the on-board modules needed to implement the selected collection strategy. When the creation of a new data packet is required by one of the active policies, it manages the collection, compression (when needed) and packing of data and signals the Communication Module that a compressed xFCD packet is ready for transmission.

**Compression Module** receives from the Service Manager the location of the images, compresses them and returns to the Service Manager the location of the resulting file. The adopted compression technology is JPEG-2000 for its SNR-scalability and robustness features which permits to adapt the compressed bit-stream to the available bandwidth.

**Communication Module** manages the data exchange between the Main Center and the vehicle. It gathers xFCD packets ready to be transmitted and, following a strategy depending on configurable parameters, activates the communication channel.
and transmits the packets to the Main Center. This strategy should guarantee that the delay between the collection of an xFCD packet and its transmission to the Main Center, and the total size of the gathered packet do not overcome prefixed values.

Traffic Level Analyzer receives input from Car-Data Server and estimates the traffic level by analyzing the temporal distribution of GPS and ECU data. If the measured level is above a certain threshold, it alerts the Service Manager by sending it the estimated traffic level.

Vehicle ahead Checker receives input from the Image Server. By analyzing stereo images coming from the Image Server, it checks if there is a vehicle ahead and its distance and provides this information to the Service Manager upon request.

Roadworks Detector is a vision module which analyzes image sequences coming from the Image Server, in order to detect the possible presence of temporary warning signs. Its output, produced on a frame-by-frame basis, is filtered and an alarm is sent to the xFCD Manager only if the sign detection persists over a number of frames. Every detection causes two alarms: the first at the beginning of the detection interval, the second at the end.

4.1. Data collection triggering

Different techniques have been implemented in the Service Manager to determine when a data collection is needed according to the provided policy.

In the case of T modality (1) a timer is set up to the multiples of \( \delta_T \) and when the timer expires data is collected according to the defined type B, V or X. Data type B and V is requested to the Car-Data Server, while X data is requested to the Image Server and to the Traffic Level Analyzer.

In case of R modality the xFCD Service Manager monitors the location of the vehicle and when it enters the region specified by the policy (3) it triggers off a collection. Next, a timer is set up to multiples of \( \delta_R \) and only when the timer expires a new collection takes place. When the vehicle leaves the specified region the timer is switched off.

If the collection modality is on-event (2) we distinguish two cases: roadwork event and traffic event. In the former case, the Service Manager waits for a start signal from the Roadworks Detector: when it arrives the Manager asks for data to the Car-Data Server and/or Image Server, according to the selected data type, putting them into the xFCD packet and repeats the request every \( \delta_k \) seconds until the end signal arrives from the Roadworks Detector. In the case of traffic event, when the Traffic Level Analyzer signals a slow traffic condition, the Manager asks the Vehicle ahead Checker for information about the space ahead the vehicle: if a close vehicle is present the alarm is confirmed and the collection takes place, otherwise the alarm is ignored. Next, to avoid too frequent packet collection, a timer is set up to the value \( \delta_k \) and only when it expires a new alarm can be accepted.

4.2. xFCD packet form

The identifier of policy that provoked the collection, along with the vehicle identifier are added to each packet so that the Main Center can recognize the sender and the reason it has sent the packet. If packets contain images, the manager asks the Compression Module to compress them before informing the Communication Module that a new packet is ready to be sent. Data contained in the xFCD packets is represented in two modalities: all information except images are stored in a xml formatted string, while images are stored in separate compressed files.

5. Temporary danger warning signs on-board detection

In this section we describe the on-board vision module for roadworks event detection.

The presence of roadworks is characterized by specific signals designed to be clearly visible to the drivers under different illumination conditions, such as temporary danger warning signs, barriers, cones, fences. Nevertheless, analyzing images at frame rate with the aim to detect some of these typical roadworks elements, which appear under different illumination, positions and scales, could be computationally impractical on the limited on-board resource, especially if other services require an intensive CPU usage, like the driver assistance system.

For this reason we choose to develop a module specialized for the detection of temporary danger warning signs that works on low resolution color images at frame-rate. We focus on these signs because they are mandatory in each roadworks site and because they are characterized by regular shape and color. Temporary danger warning signs (TDWS) have triangular shape with a corner upwards, red border, yellow background, and a black icon inside, according to the Italian Highway code. Starting from the above considerations, we decide to use color and shape features not only to detect the specific item, but also to focus the search through the image.

Even in this limited domain the problem remains quite difficult as real traffic signs are very different from those reported in driving manuals: typically they appear with different sizes and rotations, are embedded in cluttered background, and often are stained or damaged. Examples of these situations are reported in Fig. 2.

The basic idea of the algorithm consists in labeling the image pixels analyzing their color, determine some candidate positions where to focus the matching with templates depicting a yellow corner surrounded by a red border and, finally, to check the presence of a sign along a sufficiently long frame sequence. In the following the single steps are detailed.

5.1. Pixel color labeling

This step aims to classify image pixels based on their color. We use a rough definition of red and yellow that is simply based on the hue of the pixel and is, in principle, invariant to the illumination intensity.

Hue is an angular measure around the achromatic axis with respect to an origin at pure red. Generally speaking, a color image segmentation based only on hue is not robust, because this value is undefined on the gray level axis and is unstable for pixels with low saturation. Since we are interested only in classifying the pixels in the color set \([Y, R, O]\) where Y stands for yellow, R for red, and O for other, we firstly compute the saturation and if it is close
to zero (gray values) we assign the pixel directly to the class O, avoiding the hue computation. Otherwise, hue is fast computed using a Kender’s like algorithm (Kender, 1976; Schwarzbacher, Foley, Brutscheck, & Schwingel, 2000). The standard angular range $[-36^\circ, 36^\circ]$ is used to label a pixel as R and the angular range $[36^\circ, 60^\circ]$ for Y, which contains the gradations from orange (pure orange hue is $38^\circ$) to yellow (pure yellow hue is $60^\circ$). All the other pixels are labeled as O.

Contemporaneously, we consider the R–Y transitions in the labels and select the salient transitions, i.e. those for which one of the conditions in Fig. 3 is valid.

Analogously the salient Y–R transitions are those for which one of the conditions in Fig. 4 is valid.

In order to exploit the knowledge of the TDWS structure we search for salient R–Y transitions followed, within a fixed number of pixels, by a Y–R salient transition and, if no pixel between them is red, those are all labeled Y.

The described labeling process, performed through a single row-by-row scan of the input image, produces a three-label map, a transition map, i.e. a binary map storing the locations of the salient R–Y and Y–R transitions, and the total number of salient transitions. Fig. 5 shows an example of this step.

A focused template matching procedure to possibly detect TDWS in the three-label map is now described.

5.2. Templates definition

In order to implement a fast algorithm able to cope with targets having different size, we choose to focus the search on the top corner of the sign, as suggested by observing a relevant number of temporary signs in real images. In fact, the following points came out:

1. The bottom-right corner is usually the less visible in the images, because it is the first to leave the field-of-view when the camera approaches the sign.
2. The bottom-left corner is the most frequently damaged because it is subject to be hit by vehicles (particularly, heavy vehicles moving in roadworks zones).
3. Other kinds of occlusions evenly occur for the three corners: the top corner is sometimes occluded by special lamps, the bottom corners by vegetation, vehicles or other signs.

Even if we limit the search to a single corner, we need multiple templates to cover its different appearances in the images. We select three templates to describe the shape of the top corner in case the sign is acquired from a nearly frontal position, or from a lateral position (left and right), and a fourth template to capture complete but far signs, characterized by a small size. Two examples are shown in Fig. 6.

Template pixels can assume four different values: Y, R, O and D. The first three values match, respectively, Y, R and O pixels in the three-label map, while D is a Don’t care value indicating that corresponding pixels will not be considered in the matching.

5.3. Focus of the search

Computational complexity represents the major drawback of the template matching technique, in its straightforward application. The capability of limiting the search space by focusing on regions of interest could significantly reduce the processing time.

In order to speed up the matching, we use the information about salient transitions collected during the labeling step. Firstly,
we apply the template matching procedure only if the total number of transitions in the image is greater than the minimum number of transitions in the template set. The second heuristic is to compute the matching distance only in a reduced set of candidate positions, that is where a transition in the image matches a transition in the template. This set of candidate positions can be efficiently computed from the salient transitions in the image and the templates (Fig. 7).

5.4. Matching measure

In order to evaluate the similarity between a template and a region of the label map, we introduce a measure \( s() \) defined over set of values \( \{Y, R, O\} \times \{Y, R, O\} \), while template pixels having value \( D \) are excluded from the computation. The measure \( s() \) is defined as follows:

\[
s(Y, Y) = s(R, R) = s(O, O) = 2
\]

\[
s(Y, O) = 1
\]

\[
s() = 0 \quad \text{otherwise.}
\]

The template is split vertically into two parts, left and right, in correspondence of the sign vertex; the similarity score, i.e. the sum of the similarity in each location, is computed separately on the left and right parts. If both the similarity scores are greater than a fraction of the maximum score corresponding to a perfect matching (twice the number of non-\( D \) pixels in the template), then a match is found and the searching procedure stops returning a TDWS detected result. If no candidate position produces a good match the procedure returns a TDWS not found result.

5.5. Temporal persistence

The described procedure is applied on a frame by frame basis, making it prone to generate false alarms or multiple close alarms. To deal with this problem and to send to the xFCD Manager only a limited number of reliable alarms, we introduced a sort of temporal smoothing performed through the finite state automaton shown in Fig. 8. We require a minimum number of TDWS detections to establish the start of a “Roadworks” alarm, and a minimum number of consecutive frames without detections to identify the end.

6. On-board traffic level estimator

The slow traffic condition is detected on-board by two modules coordinated by the Service Manager:

Traffic Level Analyzer a module which estimates the local traffic by analyzing the temporal distribution of vehicular data.

Vehicle ahead Checker a vision module which analyzes the free space in front of the vehicle in order to estimate the distance of the preceding vehicle, if any.

When a critical condition is inferred by the former, the latter is consulted to verify the presence of a vehicle ahead as possible cause of the slowdown. This last check ensures a high reliability level to our xFCD system.

In this section, an overview of the Traffic Level Analyzer is provided, followed by a brief description of the Vehicle ahead Checker (Zanin, 2007).

6.1. Traffic Level Analyzer module

The Traffic Level Analyzer aims to estimate traffic condition in real time, both in urban and extra-urban environments. In the first case the task is particularly challenging because of the low average speeds typical of the city centers, especially at rush hours.

The implemented module is based on the analysis of various vehicle information, including data from the navigation system. The most relevant data analyzed by the system are: latitude, longitude, direction (from GPS receiver) and speed, brake and transmission clutch activity, revolutions per minute, acceleration, steering wheel movements, turning lights status, etc. (from the vehicle ECU). The temporal evolution of this data is analyzed by dividing the observation period into short time intervals which are long enough to characterize the condition of the traffic. Within each time interval, 3 min long in the current version, the rough data are filtered and integrated to obtain a sequence of more complex data, like e.g. gear changes and inter braking space.

The values of these parameters, both directly measured and derived parameters, are put into connection to the traffic level by defining, for each of them, an adaptive weighting factor which has been determined on an experimental basis. The weighting factors depend on external parameters like the surrounding conditions or the users’ driving style. Moreover, they can be manually modified for the fine tuning of the algorithm.

At the end of each time interval, the module produces a score, in the range from 0 to 100, that can be related to the possible presence of a traffic slow down. The score can be easily mapped into an indicator of the real traffic situation, for example by associating different traffic conditions, e.g. no traffic, slow traffic, heavy traffic and stationary traffic, to different score ranges.

![Fig. 8. The finite state machine for the Start/End detection of the roadworks zone. Two thresholds MinAl (minimum number of alarms) and MinNoAl (minimum number of consecutive frames without alarms) determine the Start and End time of the detection.](image-url)
All the parameters involved in the Traffic Level Analyzer, the length of the observation periods, the data aggregation sub-periods, the selected parameters, the weighting factors, and the score mapping, have been optimized using field tests.

6.2. Vehicle ahead Checker

The goal of this module is to confirm or cancel the traffic alarms generated by the Traffic Level Analyzer. The computation manages firstly a single frame at a time, looking for vehicles directly ahead and close to the host platform. Then it integrates temporally the single frame measures to obtain a more robust detection of ahead vehicles, along with a confidence level.

In the current system, ahead vehicle detection is entirely based on stereo vision, i.e. obstacle hypotheses are derived from the disparity map only. Stereo computation is performed using SVS (Small Vision System) (Konolige, 2007), a commercial software that is an efficient implementation of an area correlation algorithm. A disparity map example is shown in Fig. 9b.

6.2.1. Ground plane fitting

The first step of the algorithm computes ground plane equation. The approach is based on on-line least-squares estimation and assumes an almost flat road in the near proximity of the vehicle (about 15 m ahead). Knowledge of the ground plane is useful not only to exclude road points from further analysis, but also to estimate relevant scene parameters like horizon line, camera roll and pitch (Zanin, 2006).

The input data can be seen as a set of points \((h_i, v_i, d_i)\), where \(h_i\) and \(v_i\) are coordinates of left image pixels and \(d_i\) is the corresponding disparity. Suppose we are able to select a certain number \(N\) of points that have a position on the image compatible with being part of the road. Our aim is to find a linear function \(d(h, v) = ah + bv + c\) (a plane in the \(h, v, d\) space) that can explain the data points, i.e.

\[
d(h_i, v_i) \approx d_i, \quad i = 0, \ldots, N - 1.
\]  

We estimate the fitting function defining an error measure and finding the parameters that minimize it. A common error measure is the sum of the squares of the residuals

\[
R = \sum_{i=0}^{N-1} (d(h_i, v_i) - d_i)^2.
\]  

The minimization of \(R\) is a convex cost minimization problem, i.e. it presents a single global minimum. The global minimum is directly computable in one step. A drawback is that \(R\) is very sensible to outlier points. There are error measures that are more robust, but that are much more computationally expensive. Our approach is to implement a number of heuristic strategies in order to avoid as much as possible the presence of outlier points in the fitting set. The minimization is performed zeroing the partial derivatives of \(R\) with respect to the parameters \(a, b,\) and \(c\). In the case of noisy data or in presence of disparity points that belong to an undetected obstacle, the plane estimation becomes very unstable and can produce results that are very improbable, like quasi-vertical or very skewed planes. One possible way to limit this effect is to constrain the plane to pass through a fixed anchor point \((h_a, v_a, d_a)\). With the anchor point constraint the minimization problem becomes:

\[
\begin{aligned}
\min_{a,b,c} \sum_i (ah_i + bv_i + c - d_i)^2, \\
n &= d_a - ah_a - bv_a.
\end{aligned}
\]  

Two problems need to be addressed: how to select the \(N\) fitting points; and how to select the anchor point. The disparity map includes points from every object in the scene that is big or contrasted enough to be caught by the stereo algorithm. Our interest is to feed the least square fitting with points having high probability of being part of the road. Our strategy is to consider only valid disparity points that fall inside an predefined interest region and that are near to the ground plane of the previous frame, if available. The fitting region is shown in Fig. 9c and is determined projecting a straight virtual road on the image, using the camera calibration parameters at rest.

Let us define the real world point \(P_a\) as the projection of the camera optical center on the ground. A reasonable choice for an anchor point could be the projection of \(P_a\) onto the image plane. Unfortunately, \(P_a\) usually projects out of the image. In this case, we select as anchor point the first intersection of the camera frustum with the line that passes through \(P_a\) and has the same direction of the

![Image](image-url)
horizontal component of the camera optical axis. Coordinates $h_i$ and $v_i$ are easily derived from camera parameters, while $d_i$ is estimated sampling a large number of disparity maps. $d_i$ is bootstrapped from a standard reasonable value and is automatically updated when a sufficient number of disparity maps are sampled.

Once the road plane equation is available, we exclude from the subsequent obstacle detection steps each $(h_i, v_i, d_i)$ point that satisfies

$$d_i < ah_i + bv_i + c + r(d_i),$$

where $r(d)$ is a relaxation of the planar constraint that depends on the measured disparity for that pixel and accounts for measurement and model errors. In the current implementation we use either a constant value or the following linear relaxation

$$r(d) = \frac{d_{\text{M}}}{d_{\text{ME}}},$$

where $d_{\text{ME}}$ and $d_{\text{M}}$ are the maximum possible disparity and the maximum desired value of $r(d)$, respectively. Fig. 9d presents an example, showing in dark the pixels selected by (7). A rule based strategy is then applied to the remaining disparity points.

6.2.2. Pre-clustering and background regions

Pre-clustering step extracts connected regions of constant disparity, and then applies the following absorption rule. Absorption of a region $r$ into a larger region $R$ happens if all the following conditions are satisfied:

1. The absolute difference of disparity values is less than $d$.
2. The distance between $R$ and $r$ is less than $r$ pixels; here, the distance between two regions is defined as the minimum distance between any point of the first one and any point of the second one.
3. Let $A(\cdot)$ denote the area of a region in pixels, one of the following two conditions is true: $A(r)/A(R) \leq z_1$; or $A(r)/A(R) < z_2$ (with $z_2 > z_1$) and a portion $\beta$ of the two perimeters overlaps.

Default values for pre-clustering parameters are: $d = 2$ disparity levels; $r = 1$ pixel; $z_1 = 0.15$; $z_2 = 0.4$; and $\beta = 0.2$.

After pre-clustering, small isolated regions located entirely over the horizon line are eliminated. The horizon line, along with camera roll and pitch angle can be derived from ground plane equation (details are available in Zanin (2006)). Then, background regions are labeled. This helps excluding from further computation evidently far and large regions. Knowledge of camera parameters and average vehicle size, make possible to derive an a priori relation between disparity values and maximum expected height and width of a generic road obstacle. Regions are labeled as background if the ratio of their height with respect to the maximum expected height is higher than a threshold. In our implementation the ratio threshold is 1.

6.2.3. Non-background regions

The remaining regions are subjected to the following elaboration steps.

Firstly, regions displaying a perspective effect are searched for and merged. The algorithm uses the following definition of side object: “an object composed of at least three aligned regions, with similar shape, and with adjacent disparity values”. Alignment is referred to rows: shape similarity between two regions is measured in terms of their equivalent ellipses, considering the directions of the main axes and the eccentricities. The regions merged in this way are classified as side objects. Prospective side objects are checked for small areas, in order to not include into this class large lateral objects, like trees.

Secondly, regions that satisfy one of the following three rules are merged: small regions are merged with those regions that have bounding boxes mostly overlapping with the original region; other regions are merged if their bounding boxes overlap for more than a percentage $\eta$ (default $\eta = 60\%$); finally, close regions with identical disparity are merged.

Next, a check on remaining object heights is performed, with the aim to label regions lower that a certain threshold as flat objects. It could be the case, for instance, of a piece of asphalt. The processing is analogous to the background objects labeling technique.

6.2.4. Ahead obstacles

The still unlabeled regions are obstacle hypotheses. They are processed by a region growing algorithm. It tries to recover object parts that could have been erroneously removed because considered as part of the road. The region expansion is an iterative process, controlled by a stopping rule. The elaboration considers the $N$ lower pixels of the region at $t$th iteration. Pixels directly below each of them are checked for disparity compatibility with the object. Let $n_i$ be the number of successfully checked pixels, $B$ and $b$ thresholds that determine the stop condition. If $n_i \geq B$ and $n_i/N > \beta$, then the $n_i$ pixels are added to the region, becoming part of the lower boundary at the following iteration. $B$ plays the role of minimum percentage of expandable pixels and relates minimum $n_i$ with object width, while threshold $b$ puts a lower bound on acceptable $n_i$, in order to avoid the growth of thin peaks. Default values are $B = 20\%$ and $b = 3$ pixels.

In this task we are interested in detecting ahead vehicles, so vehicles moving in lateral road lanes must be excluded. An example is shown in Fig. 9e; the black car on the left is not in the same lane, so it should be discarded. In order to do so, we define an ahead region of interest $E$ (see Fig. 9e). An obstacle candidate is discarded if its intersection with $E$ is null or small. The shape of $E$ depends on camera parameters. In the current implementation, it is the image projection of a stripe about 3.5 m wide, starting in the near vicinity of the vehicle at ground level and ending 20 m ahead with a slope of $4^\circ$ in order to take into account possible variations of road slope. Each region is characterized by a minimum and a maximum disparity. Through camera calibration it is possible to assign a real world distance to each disparity level. Objects whose minimum distance is greater than 20 m are discarded from the candidate ahead vehicle region list.

The input frame is classified as busy if at least one ahead vehicle candidate survives the selection; it is classified as free otherwise.

6.2.5. Temporal analysis

When a traffic alarm is generated we want to check its validity and knowing the ahead status of the current frame only is of little help. Much more interesting is to consider globally the ahead status of a sequence of frames. So, we aim at classifying a sequence of frames as busy or free.

The chosen approach is straightforward. Firstly, we compute the busy/free status for each of the last $N_w$ frames. Secondly, we compute the busy confidence $c$ defined as the ratio between the number of frames labeled busy and $N_w$. Lastly, we classify the sequence as busy if $c$ is greater than a threshold $c_3$, free otherwise.

7. Data communication

The communication system in DIPLODOC is based upon two Local Area Networks (LAN), within the vehicle and in the Main Service Center, connected by means of a GSM or GPRS data connection. It is mainly based on the geographic coverage provided by cellular networks, even if it is easily adaptable to any other wireless communication architecture.
The connection between the two LANs is managed in such a way that applications on both networks can transparently communicate among them without any prior knowledge about the nature of the physical and logical link that is enabling the connection itself. In fact, the implemented telematic routing system can route Internet Protocol (IP) packets between the two sub-networks masking the nature, complexity and management procedure of the link.

It is possible to identify two different levels of information exchange between the Main Center and the on-board xFCD Service Manager:

- A high-level protocol for the exchange of control communications or commands, like the collection policy from the Center to the vehicle or an end of xFCD packet transmission notification from the vehicle to the center. This protocol is based on XML and the communication is based on a socket connection.
- A protocol for the transmission of xFCD packets from the vehicle to the Center. Information in the packets is grouped in one or more files depending on the type of the requested data: images are stored in compressed binary files, other data are stored in a text file (in XML format). In this case a FTP connection is established and used.

The following paragraphs outline the communication architecture designed for meeting the xFCD needs and detail the data exchanges involved in the xFCD service.

7.1. Data communication architecture

Data communications are enabled through wireless connection of the vehicle to the cellular network. If a GPRS/UMTS (Universal Mobile Telecommunications System) data connection is available, a bi-directional tunnel is established within the operator’s network to connect the vehicle LAN and the Main Center LAN. Moreover, a Virtual Private Network (VPN) is established.

Protocols used to achieve full IP-level communication are:

- Point-to-Point Protocol (PPP) between the vehicle gateway and the GPRS network.
- PPP between VPN Gateway and an Internet Point Of Presence (POP).
- IP over PPP, both between VPN Gateway and the Internet POP and between the vehicle and the GPRS network.
- The vehicle then sets up a bi-directional TCP connection with the VPN Gateway interface on the public Internet.
- A Secure Shell (SSH) tunnel is set up over such TCP socket.
- Within this tunnel (connecting the two end-points of the system: the vehicle and the VPN Gateway) a PPP connection is established.
- Over this PPP connection regular IP traffic can then be routed.

In case GPRS connectivity is not available, GSM network is employed. Protocols used to achieve a full IP-level communication in this scenario are:

- V110 between the GSM module and an ISDN modem at the Main Center.
- PPP between Remote Access Service (RAS) Server and the vehicle IP over PPP.
- IP over PPP.

At the time the project was deployed, the available equipment only allowed digital transmission over GPRS networks using 2 slots for upload (from the mobile device to the base-station) and 5 slots for download (from base station to the mobile device). Since UMTS and other convenient technologies (HSCSD, EDGE) are available nowadays, they could be considered for integration within the overall architecture.

7.2. Information exchange

When the vehicle is switched on a socket connection to the Main Center is opened and, if the xFCD service is enabled on the vehicle side, the Main Center transmits the data collection policy to the xFCD Manager. The policy consists of a set of string of the form in (1)–(3) embedded in a properly defined XML Schema. According to the policy, the Manager activates on-board all the needed modules and manages the collection of xFCD packets.

Whenever a new packet is set up, it is sent to the Communication module that stores it in a temporary area. According to a sending strategy that aims at minimizing the number of data transmissions while ensuring the arrival of xFCD packets within a maximum delay, the Communication manager opens a FTP connection to the FTP server in the Main Center and starts to send the files that compose the packet. The files, each one containing an identifier of the packet, are stored in a dedicated area in the Main Center. At the end of the files transmission, a signal is sent to the Main Center over the socket connection, to notify that a new packet is available.

Any time a connection exists, the Main Center can send an update of the collection policy.

8. Experiments

A set of experiments were planned and executed in order to evaluate the main on-board modules of the xFCD system: Traffic Level Analyzer, Vehicle ahead Checker, and Roadworks Detector.

8.1. Performance of on-board Traffic Level Analyzer

The evaluation of this module is a complex task due to the difficulty to compare its output to a ground-truthed real traffic sequence. We choose two different ways to obtain an automatic classification of the traffic conditions. The first one uses the standard classification proposed by the Radio Data System Traffic Message Channel (RDS-TMC) which provides a set of rules to associate a numeric label (from 0 to 3) to different traffic conditions. A label is computed for each consecutive time slot taking into account the distance traveled by the vehicle. The second way estimates the traffic condition by measuring an objective parameter that has been historically used for this purpose: the average time vehicles take to cover a pre-defined route.

A set of tests have been performed along urban routes in Trento, a one hundred thousand people town in northern Italy. Test routes took place at different day time in order to evaluate the system in different traffic conditions. The following paragraphs compare the output of the Traffic Level Analyzer with the traffic conditions estimated in these two ways.

8.1.1. Comparison with RDS-TMC classification

The RDS-TMC standard reference has been used to associate numeric label to different traffic conditions: 0: No traffic; 1, Slow traffic; 2, Tailback; 3, Stationary Traffic.

Each value refers to a time slot of 3 min, in which the distance covered by the vehicle in urban scenario is likely to range from 0 to approximately 3 km. If the urban average speed is considered (25 km/h), the distance covered in 3 min in case of no traffic is 1.25 km. According to this consideration, the RDS-TMC traffic values were assigned with the following criteria:

- label 0 the covered distance in the considered time slot goes over 1 km suggesting absence of traffic;
label 1 the covered distance in the time slot goes from 100 m to 1 km due to traffic conditions, slow traffic is considered; much attention was paid to distinguish stops due to ordinary traffic light phases from stops due to congestion; label 2 the covered distance in the 3 min slot is below 100 m; the condition corresponds to a tailback; label 3 if at least two consecutive time slots show traffic conditions of tailback, the condition actually corresponds to stationary traffic (therefore two consecutive time slots with label 2 are changed to 3).

As the Traffic Level Analyzer generates, for each disjoint time slot, a value in the range 0–100, we experimentally selected a threshold value (50 in the experiments) for establishing if an alarm has to be signaled or not. We consider it a false alarm if the corresponding RDS-TMC label is 0. Table 2 shows the results related to a urban route covered in 1 h 51’. For each time slot, it reports the result of our traffic estimation (a value in the range 0–100), the traffic class computed according to the RDS-TMC classification, the traffic alarm status and the agreement between the alarm status and the RDS-TMC class.

It can be seen that there is a good correspondence between the output of the Traffic Level Analyzer and the one estimated using the RDS-TMC standard. There is only one false alarm detection, and it happened after the situation of stationary traffic (time slot 21). This may be caused by a time quantization problem, but in practice it does not represent a critical error, because it simply extends to the next time slot the traffic condition which was previously detected. No missing alarms were found in the collected data.

8.1.2. Comparison based on travel time

In this second approach to evaluate the reliability of the traffic algorithm, the time the vehicle takes to cover a pre-defined route is measured. This method is considered relatively objective, as it is not influenced by any “human” factor, apart from the particular user’s driving style. This last aspect is mitigated (if not completely eliminated) by the urban speed limit.

First, the route was covered in no-traffic condition and the measured time is used as benchmark. The duration of the total route was about 12 min, traveling at the highest speed allowed by the limits. For a more accurate comparison, the route was split into two segments covered in the same time interval by the vehicle (about 6 min), to ensure that at least one 3 min time slot is completely contained in the time interval. Next, the same route has been covered several times in rush hours. The idea is to validate the algorithm output by comparing the actual travel time to the benchmark time. Table 3 shows the time taken to cover, several times, the two half routes and the corresponding traffic level estimated by our algorithm. As it can be seen, relevant levels of traffic measured by the algorithm correspond to significant increases in travel time.

From these figures comes out that an increase in the travel time less than 20% with respect to the benchmark (6 min) is typically not caused by the traffic conditions. Above this percentage, the increase is likely due to road congestion, although a rigorous relation between travel time and traffic condition cannot be inferred using only these data.

8.2. Vision-based Vehicle ahead Checker

The Vehicle ahead Checker module is tested on two real world sequences, presenting different environmental conditions and camera configurations. Both sequences were acquired with the onboard color stereo camera configured to work at resolution 320 × 240 at 15 Hz.

The first sequence is 12,384 frames long (13’ 46”). It was taken during a sunny Summer morning (July 16, 2004) in northern Trento, Italy. It covers both urban and highway scenes, with different traffic and illumination conditions. The stereo camera optical axis is almost parallel to the ground plane, i.e. pitch angle at rest is almost null. The second sequence is 14,845 frames long (16’ 29”). It was taken around midday of a cloudy Winter day (February 23, 2006). Roads are clean, but patches of snow are sometimes present. The vehicle path covers different situations: a downhill road to

<table>
<thead>
<tr>
<th>Slot no.</th>
<th>TLA</th>
<th>RDS-TMC</th>
<th>Alarm</th>
<th>Agree Y/N</th>
<th>Slot no.</th>
<th>TLA</th>
<th>RDS-TMC</th>
<th>Alarm</th>
<th>Agree Y/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>Y</td>
<td>3</td>
<td>100</td>
<td>3</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>21</td>
<td>63</td>
<td>0</td>
<td>Y</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>2</td>
<td>Y</td>
<td>23</td>
<td>100</td>
<td>2</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>53</td>
<td>1</td>
<td>Y</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>25</td>
<td>93</td>
<td>2</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>26</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>28</td>
<td>57</td>
<td>1</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>29</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>2</td>
<td>Y</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>31</td>
<td>90</td>
<td>2</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>87</td>
<td>2</td>
<td>Y</td>
<td>33</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>34</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>35</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>36</td>
<td>60</td>
<td>1</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>37</td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>82</td>
<td>3</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For each time slot the estimated traffic level, the RDS-TMC class, and their comparison is reported.
Trento center, a high traffic situation close to the train station, and a highway road in the South direction. In this case the stereo camera is more pitched (about $8^\circ$ at rest).

8.2.1. Ground truth generation

Each available frame is manually labeled with respect to the presence of other vehicles directly ahead of the host platform. We label a frame as busy if the following conditions are true:

1. There is at least one ahead vehicle in the same road lane.
2. The ahead vehicle is less than 20 m away.

If these conditions are not met, than the frame is labeled as free.

As the latter condition requires to provide a precise distance estimation, we developed an aiding tool for manual measuring of distances. It shows the stereo image pair and an inverse perspective mapping (IPM) transformed image (Bertozzi, Broggi, & Fascioli, 1998) and then projects a standard wire-frame vehicle model, placed on the road plane, on the three images. The operator can move around the virtual vehicle until it overlaps with the real vehicle, obtaining its position in real world units. Current pitch angle has a strong influence on this procedure, so the user should manually estimate it. The best way to do it is provided by the IPM view, that is extremely sensitive to pitch angle. A wrong value transforms parallel features into converging or diverging ones, providing an immediate visual clue to the operator (Fig. 10). Another indication of wrong pitch angle is that model size does not match the real vehicle appearance. The precision of this manual procedure was tested on a sequence with an ahead vehicle positioned at several known distances from the camera, obtaining a distance estimation error lower than 15 cm at 20 m. Both the sequences were manually labeled: 3234 frames were busy, while the remaining 23,995 were free.

8.2.2. Single frame results

A direct comparison of the frame-by-frame output of the algorithm with ground truth data provides the following results: 2417 correct busy classifications (true positives TP); 22,026 correct free classifications; 817 wrong free classifications (false negatives FN); and 1969 wrong busy classifications (false positives FP).

![Fig. 10. Manual distance measurement: (a) original image; (b) IPM transform with the correct pitch angle. Parallel features remain parallel; (c) IPM transform with a wrongly lower pitch angle; (d) IPM transform with a wrongly higher pitch angle. The world position of the wire-frame model shown in every image is always the same; it matches the real vehicle only if pitch angle is correct.](image)

![Fig. 11. Example images from the July 16, 2004 sequence. The first three show correct detections, while the last three are error examples.](image)
A large portion of the errors are temporally isolated and are caused by spurious detections (see Fig. 11). Other sources of errors are vehicles positioned in front of the cameras that cannot be considered as busy, e.g., vehicles passing perpendicularly when the host vehicle is stopped at a crossing, or parked cars that appears directly ahead during a curve.

8.2.3. Temporal analysis results

As explained in Section 6.2, the output of this module is not the ahead status on the single frame, but on a time interval. We choose to consider the previous $N_w = 200$ frames, corresponding to a temporal windows of about 13 s. The same $N_w$ is used also on the ground truth.

Fig. 12 presents the temporal evolution of the traffic estimation computed on the 2004 sequence. All major events are correctly detected with a high confidence. False detections are generally characterized by a low confidence value. Fig. 13 presents the same graph for the 2006 sequence.

Using $N_w = 200$ and a confidence threshold $c_t = 45\%$ on all the available 26,831 frames, we have the following results: 2761 TP,
22,921 TN, 496 FN, and 653 FP. In terms of completeness and correctness, defined respectively as TP/(TP + FN) and TP/(TP + FP), the final ahead traffic presence performances are:

\[
\text{completeness} = 84.8\% \\
\text{correctness} = 80.9\%.
\]

Considering a different temporal window or confidence threshold leads to different results. Fig. 14 presents completeness and correctness measures for different \(N_w\) and \(c_t\) values. The main variability is along the completeness axis and is mainly influenced by the confidence threshold: a higher \(c_t\) increases the correctness, but it decreases completeness. The length of the temporal window has low influence on performances.

### 8.2.4. Elaboration time

Elaboration times are measured using the profiling software TAU (Performance Research Lab, 2007) on a PC equipped with a Dual Core Pentium®D 3 GHz, 2 Gb RAM. Times are averaged on 10,000 frames taken from both the described test sequences (320 × 240, 15 Hz, color). We consider negligible the acquisition time, i.e. the time required to have the current image stored in the memory. Table 4 presents a detailed analysis of each computation step. The average time required to elaborate a frame is 47.36 ms, which is compatible with the real time constraint of 15 frames per second (66.67 ms per frame). Some partial results are shared with other modules of the DIPLODOC system (road recognition, obstacle detection and classification), so they need to be computed only once.

### 8.3. Vision-based temporary danger warning sign detection

To develop and validate the proposed on-board vision algorithm we used a database of about 10,000 still images acquired from the

---

**Table 4**

<table>
<thead>
<tr>
<th>Task</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disparity computation (SVS)</td>
<td>9.03</td>
</tr>
<tr>
<td>Ground plane fitting</td>
<td>0.59</td>
</tr>
<tr>
<td>Non-road point selection, Eq. (7)</td>
<td>0.23</td>
</tr>
<tr>
<td>Pre-clustering</td>
<td>15.54</td>
</tr>
<tr>
<td>Background and far away region labeling</td>
<td>14.76</td>
</tr>
<tr>
<td>Region merging</td>
<td>1.40</td>
</tr>
<tr>
<td>Flat objects detection</td>
<td>3.69</td>
</tr>
<tr>
<td>Expansion phase</td>
<td>1.34</td>
</tr>
<tr>
<td>Ahead obstacles selection</td>
<td>0.34</td>
</tr>
<tr>
<td>Sequence classification (busy/free)</td>
<td>Negligible</td>
</tr>
<tr>
<td>Other</td>
<td>0.44</td>
</tr>
</tbody>
</table>

**Table 5**

<table>
<thead>
<tr>
<th>Output</th>
<th>Rate (%)</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct output</td>
<td>74.8</td>
<td>260 No when no-sign and 143 yes when sign</td>
</tr>
<tr>
<td>False alarm rate</td>
<td>1.9</td>
<td>10 Yes when no-sign</td>
</tr>
<tr>
<td>Miss alarm rate</td>
<td>23.4</td>
<td>126 No when sign (the small targets are here)</td>
</tr>
</tbody>
</table>

**Table 6**

<table>
<thead>
<tr>
<th>Test no.</th>
<th>Time (s)</th>
<th>Road type</th>
<th>Road works</th>
<th>Frames</th>
<th>False alarms</th>
<th>Miss detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>180</td>
<td>Mountain</td>
<td>None</td>
<td>900</td>
<td>No</td>
<td>–</td>
</tr>
<tr>
<td>2</td>
<td>90</td>
<td>Mountain</td>
<td>None</td>
<td>450</td>
<td>No</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td>90</td>
<td>Mountain</td>
<td>Yes</td>
<td>450</td>
<td>No</td>
<td>Partial</td>
</tr>
<tr>
<td>4</td>
<td>90</td>
<td>Suburban</td>
<td>Yes</td>
<td>450</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>90</td>
<td>Suburban</td>
<td>None</td>
<td>450</td>
<td>No</td>
<td>–</td>
</tr>
<tr>
<td>6</td>
<td>120</td>
<td>Mountain</td>
<td>Yes</td>
<td>600</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>120</td>
<td>Mountain</td>
<td>Yes</td>
<td>600</td>
<td>No</td>
<td>Partial</td>
</tr>
</tbody>
</table>
prototypical vehicle. The collected images depict scenes both with and without road works. 883 images containing typical roadworks items have been manually segmented and labeled. The database has been used to gather statistics about the presence of the various signs, their dimensions and positions within the images. The most frequent items in roadworks images are fences and temporary danger warning signs (49% and 35% of the labeled images, respectively). Temporary danger warning signs are usually at the beginning of a work area.

A small subset of the images has been taken to develop the algorithm and to define the set of templates. The first experiment was led on 539 images taken from the database, excluding the development set. In 270 of them no TDWS was present. In the remaining 269 at least one TDWS was present. We notice that in about 40 of them the target was visible but smaller, but the signs are captured later when the vehicle approaches the target. Few miss detections depend on the template shape, particularly because small targets can be captured by only one template. Signs in which the upper-left corner is occluded, stained, very dark, blurred or with a highlight, are not detected because the pattern does not appear in the three-label map.

Other experiments have been performed on seven fragments of a long sequence, randomly selected. The sequence was acquired in a trip on suburban and mountain road. The results of the analysis performed at 5 frames per second, are summarized in Table 5. There are some miss detections in the tests. In tests No. 3 and No. 7 they are considered partial: early instances of signs are lost because small, but the signs are captured later when the vehicle approaches to them. In test No. 4 one sign gets lost due to the high image blurring (only one instance is detected, too few to trigger off an alarm) while in the test No. 6 a complete miss detection of a sign is due to the bad illumination condition [see Fig. 15]. In test No. 5 a false instance was detected at frame level, due to an accidental configuration of a yellow corner in the segmentation map; being a spot, it has been filtered by the state machine. We conjecture that false alarms are more probable in urban environments, where more man-made objects are present, which are characterized by strong colors and edges, hence likely to be confused with the target.

The execution time depends on the number of red–yellow and yellow–red transitions, i.e. the candidate positions where the template matching is applied. The average processing time is 20 ms per frame, making the algorithm suitable for the task.

9. Conclusions

In the context of the extended floating car data (xFCD) service, inside a more general telematic architecture, we have described a system able to analyze the environment surrounding the vehicle. On-board functionalities that are commonly referred to as intelligent are implemented by means of algorithms for the detection of critical situations, such as traffic jams and roadworks presence, that can be useful for a global vision of traffic variations on the road network.

In particular, an on-board car-data analyzer hypothesizes continuously the traffic status along the trip. If the case, the hypothesis of traffic queue or jam is discarded by a vision-based module that looks ahead for the presence of vehicles: if there is no vehicle ahead, the low speed of the host vehicle is not due to traffic. A second on-board vision module is able to react on temporary danger warning signs, typically placed on roadworks sites. This real-time algorithm labels images as containing, or not, the specific roadworks pattern. All these modules are activated on-board when the collection policy established by the Main Center and sent to the vehicle is on-event or the required data is of type extended, i.e. with information derived from on-board processing.

When a critical situation is discovered an alarm triggers off. An extended FCD packet is prepared containing vehicle identifier, time, position, vehicle data, estimated traffic level and images of the scene ahead. The xFCD packets collected by the moving platforms are transmitted to the Main Center where they are merged by proper algorithms to maintain an updated map of the traffic conditions and where image content can be analyzed by specialized algorithms.

The presented experiments are promising both in terms of detection performance as well as execution time. The loop of policy communication, simple and extended data collection, and packet transmission have been successfully tested in several condition, making the xFCD service interesting for a larger scale application.

References


