

Off-Road Path and Obstacle Detection Using Decision Networks and Stereo Vision

Claudio Caraffi, Stefano Cattani, and Paolo Grisleri

Abstract—Autonomous driving in off-road environments requires an exceptionally capable sensor system, particularly given that the unstructured environment does not provide many of the cues available in on-road environments. This paper presents a complex vision system, which is able to provide the two basic sensorial capabilities needed by autonomous vehicle navigation in extreme environments: obstacle detection and path detection. A variable-width-baseline (up to 1.5 m) single-frame stereo system is used for pitch estimation and obstacle detection, whereas a decision-network approach is used to detect the drivable path by a monocular vision system. The system has been field tested on the TerraMax vehicle, which is one of the only five vehicles to complete the 2005 Defense Advanced Research Projects Agency (DARPA) Grand Challenge course.

Index Terms—Autonomous vehicle, decision networks, image stabilization, obstacle detection, path detection, stereo.

I. INTRODUCTION

ON October 8, 2005, 23 vehicles and no drivers gathered in the Mojave Desert to compete in the second edition of the Defense Advanced Research Projects Agency (DARPA) Grand Challenge [1]. The U.S. DARPA created this robotic vehicle competition as an open challenge intended to energize the engineering community to tackle the major issues confronting autonomous vehicle development. For the timed competition, DARPA designed a 132-mi off-road desert course that each vehicle had to negotiate. The course was defined by an ordered list of geographic waypoints, a maximum speed for each waypoint, and the boundaries that could not be crossed. Vehicles had to operate with full autonomy as they maneuvered around obstacles lining the desert course.

Performance in the inaugural 2004 Challenge gave a clear indication of how “Grand” a Challenge DARPA had proposed. No vehicle could travel even 10 mi of a 142-mi course. Autonomous operation is quite a challenge, but the unstructured off-road environment confounds the basic environmental perception. Color has no clearly defined meaning or interpretation. Slopes frequently change. Roads are not flat. No lane markings exist; in fact, boundaries can be defined by severe negative obstacles. All of these get coupled with high vehicle speeds (up

Manuscript received January 10, 2007; revised April 20, 2007 and June 21, 2007. This work was supported by Oshkosh Truck Corporation. The Associate Editor for this paper was U. Nunes.

The authors are with the Artificial Vision and Intelligent Systems Laboratory (VisLab), Dipartimento di Ingegneria dell’Informazione, Università di Parma, 43100 Parma, Italy (e-mail: caraffi@ce.unipr.it; cattani@ce.unipr.it; grisleri@ce.unipr.it).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TITS.2007.908583



Fig. 1. TerraMax vehicle.

to 45 mi/h): The challenge is not simply to complete the course but to do it faster than anyone else.

This paper presents an artificial vision system that is developed as part of the TerraMax vehicle, which is one of the five vehicles to complete the 2005 DARPA Grand Challenge course, and the only vehicle to have used vision as its primary sensor during the race. Oshkosh Truck Corporation, Rockwell Collins, and the University of Parma partnered together to form Team TerraMax [2]. The Team TerraMax robotic vehicle (Fig. 1) is a U.S. Marine Medium Tactical Vehicle Replacement truck which was fitted with electronic actuators for steering, brake, throttle, and transmission control. A computer network was installed to host the software applications necessary for autonomous navigation. The applications consisted of vehicle control, path planning, light-detection-and-ranging obstacle detection, and artificial vision obstacle detection and path following.

The Artificial Vision and Intelligent Systems Laboratory (VisLab; www.vislab.it) of the University of Parma developed the artificial vision systems that sensed the environment. Three color cameras captured the video images; a single computer processed the data. Obstacle detection used the stereo vision and V-disparity approach, whereas path detection used the monocular images. Vision systems were developed to meet the following requirements: supply valuable information about possible drivable surfaces, reliably detect obstacles and estimate their positions, avoid false detections, and minimize the risk related to wrong classifications.

As we will see in the next sections, the system developed is not just the bare sum of two separate vision algorithms: obstacle detection and path detection. In fact, it can be seen as a single complex system, where partial results of an algorithm can be

exploited by the other and characterized by a deep integration with other vehicle subsystems.

TerraMax used the laser scanners as well as the vision system; however, the limited testing time did not allow the development of algorithms to fuse vision and laser-scanner data at a low level. Each sensing system perceived the environment and built its own real-world coordinate map to be sent to the path planner system (like in [3]) on the basis of GPS and inertial-sensor information. Map databases rounded out the tools that TerraMax used to sense and understand its environment.

This paper presents and describes in detail the approach that is used to solve the problems of obstacle and path detection in off-road environments but keeps all the details, thresholds, and numeric parameters confidential as they are proprietary information. The paper is organized as follows. In Section II, we present the hardware architecture implemented on the TerraMax vision system. In Section III, we introduce the V-disparity image properties that are used to stabilize images. Section IV describes the path-detection algorithm, whereas Section V describes the core of our obstacle-detection algorithm. Conclusions are drawn in Section VI.

II. SYSTEM SETUP

A. Motivation

Human beings only partially rely on stereo vision while avoiding obstacles in driving cars; using motion analysis and accumulated experience, we can easily drive with a closed eye. This capability appears to be portable on a computer platform in environments where it is easier to find known image patterns to be tracked (as in [4] and [5]), but the same task seems more difficult in less structured environments. This is the reason why most of the vision-based obstacle detectors (ours included) rely on stereoscopic vision. However, there are other tasks, like path detection, that still remain as monoscopic problems even in unstructured environments. Path detection is basically a pattern-recognition problem, where it is necessary to classify each part of the ground either as road or off-road. Here, stereo vision can help in bounding the area of interest to those image portions that were not previously classified as obstacles, but the core classification is still made on the basis of various pixel properties that are commonly gathered by a single image.

B. Baseline Selection and Calibration

Without any *a priori* knowledge (apart from the calibration measurements), wide baselines¹ allow accurate distance estimation even for very distant obstacles. However, at close ranges, the wide baselines give very different viewing angles and make stereo matching difficult. The TerraMax used a three-camera variable-baseline system to allow for good precision and efficient computation at a wide range of viewing distances. The cameras are installed on a rigid bar over the vehicle hood (see Fig. 2).

Selecting two cameras at a time, the system can switch through the different baselines (0.5, 1, and 1.5 m). During the



Fig. 2. TerraMax camera setup. The three cameras feature a 640×480 -pixel Bayer-pattern output and mount low distortion 6-mm optics. Synchronization is directly supplied by the cameras through the firewire bus.

DARPA Grand Challenge, the TerraMax selected the viewing baseline based on the vehicle speed. Higher speeds required greater sensing distances and thus wider baselines. In fact, an error in stereo-reconstructed distance quadratically increases with distance and comes in inverse proportion with the baseline width. Anyway, stereo reconstruction is more difficult when objects are seen by too different angles; thus, short baselines are better for closer obstacle detection.

During stereo homologous-point search, the similarity measurements are made via a sum-of-absolute-difference computation.² This allowed code optimizations to exploit the processors MMX and SSE instruction set [8], as also seen in [9]–[11]. The right image is chosen as reference, and the homologous windows are searched in the left image. This way, disparities grow as the searched window shifts from left to right on the left image.

During camera installation, every effort is made to obtain a standard form, i.e., perfect alignment and coplanarity of the image planes. The goal is to reduce the amount of software transformation needed to rectify the images,³ avoiding problem complexity growth. In the work of Nedeveschi *et al.* [10], they comprehensively explain the importance of this step. Once the best achievable alignment is reached, the remaining calibration error (e.g., misalignment) can be measured. When present, small errors can be compensated by (small) image shift, rotation, and magnification.

Both algorithms run on the same Intel Pentium IV 2.8-GHz processor with 1 GB of RAM.

III. V-DISPARITY IMAGE ANALYSIS AND IMAGE STABILIZATION

A. Motivation

In off-road environments, vehicle oscillations caused by terrain bumps cannot be neglected. To overcome this problem, image stabilization is necessary. In [12], we introduced a system that relies on V-disparity image properties to stabilize images. Fig. 3 shows an example of V-disparity image [13], [14]. The V-disparity images are the 3-D graphical representations of the similarity measures between the left and right image rows (*v*-coordinate), depending on the disparities (*d*-coordinate) used to compare them. Brightness is used as the third dimension and is directly proportional to the measured similarity. In the example of Fig. 3, the slanted segment in the V-disparity image is caused by the linearly changing maximum similarity

² For the complete discussion about matching methods and similarity measurements; see [6] and [7].

³ In this context, “to rectify images” means to geometrically transform them as they were taken by cameras in a standard form.

¹ The baseline is defined as the distance between the two cameras that form the stereoscopic system.

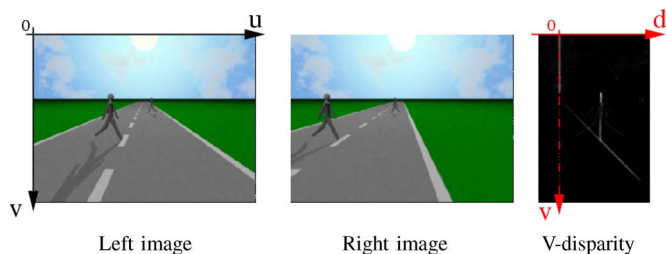


Fig. 3. V-disparity image example on synthetic images. The frame of reference is plotted. In this case, the similarity measures are computed on edges.

disparity among ground components. This segment, which is called the “ground correlation line” in this paper, contains the information about the position and the shape of the ground in the images, and its detection allows the stabilization of images.

B. Computation

The similarity measure of image rows can be computed in many ways. Labayrade *et al.* [13], [14] have computed a fast disparity space image (DSI); then, they obtained the similarity by collecting, for each image row, the number of matches for each disparity value. The V-disparity image is then used to obtain information about both ground and obstacles. In this paper, the V-disparity images are used to obtain information just about the ground. In [12], we introduced a slightly different approach to V-disparity image computation: We consider comparison windows that correspond to the image lines. That is, we measure similarity among windows of 1-pixel high and of the same width of the image. By making this choice, we considered that the ground has different disparities at different heights in the image. For this reason, it is preferable to maintain correlation windows as short as possible so that all elements belonging to the ground have similar disparity values within a correlation window. In this way, a coherent contribution is received from the ground elements during the stereo matching.

In off-road environments, a good method to highlight ground information is to apply a vertical Sobel mask to the image and to map the obtained values in a ternary domain (-1 , 0 , and $+1$), obtaining the so-called ternarized image. This approach is thresholdless and takes advantage of a property of the ground that is usually verified: It occupies the largest part of the image. Fig. 4 shows an example of the ternarized images and of the derived V-disparity image.

C. Pitch Evaluation

Considering a flat ground, the behavior of the ground correlation line in the case of vehicle-pitch variations is to oscillate parallel to itself. Knowing camera static calibration, the algorithm can compute a set of candidate ground correlation lines by varying the pitch angle. By collecting the V-disparity image values along these lines and choosing the line that collects the higher value, it is possible to estimate the pitch at time of acquisition (see Fig. 5).

The information obtained at this step is used in the subsequent path and obstacle detection to locate a region of interest in the image. It can also be forwarded to other sensors, like

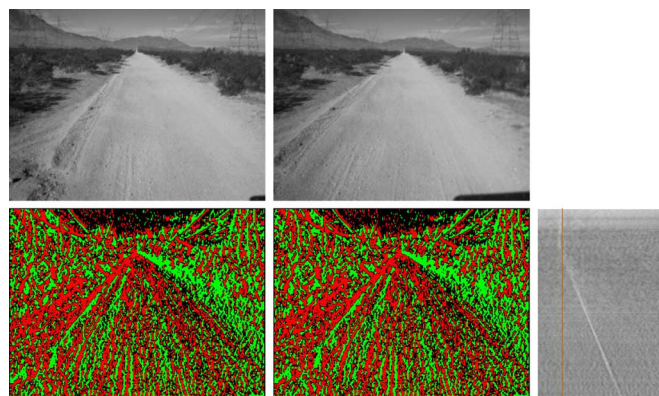


Fig. 4. Original stereo pair, ternarized images, and their V-disparity image. In the ternarized images, red means negative Sobel phase, whereas green means positive Sobel phase. In the latter, the red line lays on the zero disparity value.

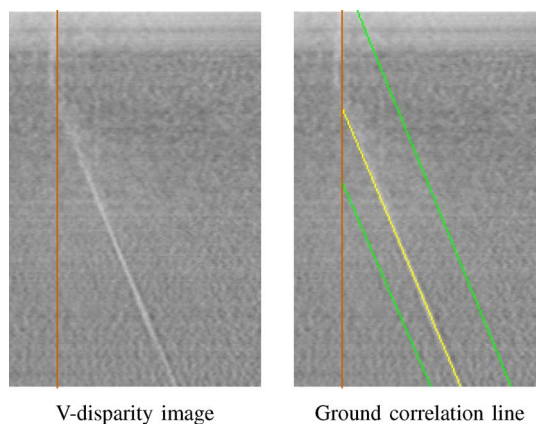


Fig. 5. V-disparity image analysis. In the right image, the detected ground correlation line is plotted in yellow. Green lines bound the search space. In this case, the ground correlation line is closer to the left bound, so that the horizon height in the image (which corresponds to the point where red and yellow lines intersect) is below its position when the vehicle is static. The algorithm can then state that the vehicle is pitching up.

the laser scanner [15], in order to improve the knowledge of their environment. Horizon estimation can also be useful to select an appropriate image area to be used to tune the camera’s automatic gain control.

D. Performance and Limits

Code optimization allows computation of a V-disparity image made of 160×240 points in less than 6 ms. Color information is not used at this step, and the green channel of the Bayer input is used to obtain the input 320×240 gray stereo images.

A few cases have yet to be addressed. In general, the approach fails when the terrain does not feature a prevailing slope and when the assumption that the ground should occupy the largest image part is not met (i.e., in front of the wide obstacles). A possible solution to this problem should be to compute the V-disparity image only, considering image regions that are classified as road by a former algorithm.

IV. PATH DETECTION

While on structured environments some successful approaches have been proposed for path detection, e.g., by VisLab

[16] and by others [17], on unstructured and unknown environments, it still remains as a hard problem. The main reason lies in the impossibility of relying on any *a priori* knowledge about the road structure, such as the existence of the lane markers. To overcome this lack of information, many different approaches have been proposed: learning the road properties by neural networks [18], selecting the actual road borders within the set of all possible curves on the image by evolutionary techniques [19], [20], and growing the regions believed to belong to the road, on the basis of some *a priori* assumptions or approximations [21]. All these methods look for a single homogeneous road surface in front of the vehicle, basing the search on brightness, color, texture, etc. The hypothesis of uniform homogeneity becomes a huge limitation as it bounds the set of detectable roads to the case of medium/well-structured environments.

To overcome the limitations of the aforementioned methods, our approach tries to generalize the problem: Roads can be made by heterogeneous surfaces. To find these potentially heterogeneous surfaces, the algorithm tries to build them up from a variable number of smaller homogeneous terrain portions. The homogeneous portion of terrain can represent any kind of natural or artificial environment elements, such as gravel, paved roads, grass, puddles of water, oil stains, drivable rocks, lane markers, shadows, etc. No previously learned and recorded features in a database [22] are used. It is possible to summarize the path-detection algorithm in the following two-step process.

- 1) *Segmentation of the image*: Here, a set of regions made of connected and homogeneous pixels is built. Image segmentation, sometimes called clustering, has been studied with success in computer vision. Using both evolutionary [23] and traditional [24], [25] approaches, it is possible to achieve good segmentation performance. Unfortunately, the real-time constraints of the Grand Challenge contest do not allow the adoption of such time-consuming algorithms; hence, we developed a simple but fast and easily tunable clustering algorithm, which is explained in Section IV-A as a good tradeoff between the performance and computational requirements.
- 2) *Decision*: Whichever combination of the obtained regions represents the road surface with the highest probability is selected here. This step falls in the class of decision problems. Decision theory [26] was originally developed to help decision processes in medical, transportation, political, or environmental fields, but now, it is widely used in artificial intelligence as a useful framework into which a variety of classical problems can be mapped. Decision networks [27] are an extension of Bayesian networks, which add actions and utilities to provide a general methodology for rational decision making. Section IV-B will describe how decision network fits the problem of deciding about the set of clusters that belong to the road surface. The presented decision process tries to minimize the risk of incorrect classifications, taking into account the current vehicle state given by speed and steering angle.

The basic idea is that the path-detection problem inevitably needs to be faced as a high-level problem. The wide range of

situations to be handled requires a reasoning-based approach. For example, there is no need to remove vehicle shadows at a medium/lower level on the basis of brightness considerations: Since vehicle shadows belong to the terrain surface, its corresponding cluster will be treated just as any other region, like any other potential part of the road. This is true in general: At a medium/low level, all the image regions have exactly the same likelihood to be part of the road. It is at a higher level, where it is possible to reliably decide about them by using road models and, when available, information about the current driving behavior.

A. Image Segmentation

During the clustering phase, the image's pixels are grouped together in the connected and homogeneous regions, which will later be analyzed by the decision network to classify the space in front of the vehicle. To achieve this, our clustering method first decomposes the image in $d \cdot d$ -pixel cells and then tries to put them together to create homogeneous groups. The searched homogeneity is intended with respect to a set of local-pixel properties, including the average color and brightness values and information about the texture, like the variance. In the next paragraph, a function that is used to measure the similarity of a pair of cells will be discussed in detail.

1) *Pseudodistance Function*: To measure the similarity of cells, we defined a pseudodistance function that combines distances from cell to cell, from cluster to cluster, and from cell to cluster. Before introducing the distance function, it is necessary to define the following entities.

A set C of c_k , where $\langle x_{k1}, x_{k2}, \dots, x_{kn} \rangle = c_k \in \mathbb{R}^n$, is the property vector of the k th cell, which is placed in the position $(i, j)^4$ on the interest area of the image. The interest area is made of cells that correspond to the position not farther than 50 m from the vehicle (see Fig. 7). This information is obtained by exploiting the methods described in Section III-C. This implies that the path-detection algorithm always has to use images captured by one of the two cameras on the currently selected baseline.

A set V of v_k , where $v_k = \langle y_{k1}, y_{k2}, \dots, y_{kn} \rangle \in \mathbb{R}^n$, is the property vector of the k th cluster of cells.⁵

The partial comparison functions are defined as follows.

- 1) The cell-to-cell comparison function is

$$c2c(c_k, c_l) = \sum_{i=0}^n D_{i-c2c}(x_{ki}, x_{li}) \cdot \alpha_i \quad \forall c_k, c_l \in C. \quad (1)$$

- 2) The cluster-to-cluster comparison function is

$$v2v(v_k, v_l) = \sum_{i=0}^n D_{i-v2v}(y_{ki}, y_{li}) \cdot \alpha_i \quad \forall v_k, v_l \in V. \quad (2)$$

⁴ This is the classical line-row pixel position notation, where $k = j \cdot 320 + i$.

⁵ Hereinafter, we will use v_k , meaning both the set of cells contained in the k th cluster and the corresponding property vector.

3) The cell-to-cluster comparison function is

$$c2v(c_k, v_l) = \sum_{i=0}^n D_{i-c2v}(x_{ki}, y_{li}) \cdot \alpha_i \quad \forall c_k \in C \text{ and } v_l \in V \quad (3)$$

where $\alpha_i \in \mathfrak{R}^+$ is the fixed weight assigned to the i th cluster and cell properties, and $D_i(\cdot, \cdot)$ is the corresponding property comparison function.

The property comparison functions must always be greater or equal to zero, but no other constraints are required.

Finally, we can define the overall cell-to-cell pseudodistance function $\text{Dist}(\cdot, \cdot)$ as follows.

1) If c_k and c_l belong, respectively, to the clusters v_k and v_l

$$\text{Dist}(c_k, c_l) = c2c(c_k, c_l) \cdot (1 - \beta_c) + v2v(v_k, v_l) \cdot \beta_c. \quad (4)$$

2) If only c_l belongs to a cluster (v_l) while c_k is still unclassified

$$\text{Dist}(c_k, c_l) = c2c(c_k, c_l) \cdot (1 - \beta_c) + c2v(c_k, v_l) \cdot \beta_c. \quad (5)$$

3) If only c_k belongs to a cluster (v_k) while c_l is still unclassified

$$\text{Dist}(c_k, c_l) = c2c(c_k, c_l) \cdot (1 - \beta_c) + c2v(c_l, v_k) \cdot \beta_c \quad (6)$$

where $\beta_c \in \mathfrak{R}^+$ is the fixed weight given to the cluster distance when computing the distance between two cells.

The two cells c_k and c_l are similar if and only if

$$\text{Dist}(c_k, c_l) \leq q_0 \quad (7)$$

where q_0 is a fixed threshold. Note that $\text{Dist}(\cdot, \cdot) \geq 0$, but in general, we cannot say anything about $\text{Dist}(a, b) = 0$ and the commutative and associative properties. Consequently, the overall process can also be considered as neither commutative nor associative. Moreover, the function $\text{Dist}(\cdot, \cdot)$ also takes into account the cluster to which the cells belong; hence, when comparing the two cells with the same characteristic vector, the distance could be greater than zero, depending on the respective clusters. For this reason, it is very important to adopt a particular cell comparison order, which is able to minimize the possible negative effects of the used noncommutative pseudodistance function. In Section IV-A2, more details are given about the comparison order.

The partial comparison functions showed in (1)–(3) are built on the hue, luminance, and saturation (HLS) color space. The images are originally acquired in Bayer pattern and then converted into HLS just before the elaboration. No other preprocessing is applied on the images.

The property vector of a given cell c_k is built from the respective HLS pixel values. Since each cell is a squared group of $d \cdot d = 8 \cdot 8$ pixels, the cell property vector can be written as follows: $\langle (H_{k1}, \dots, H_{k64}), (L_{k1}, \dots, L_{k64}), (S_{k1}, \dots, S_{k64}) \rangle = \langle (x_{k1}^{[1]}, \dots, x_{k1}^{[64]}), (x_{k2}^{[1]}, \dots, x_{k2}^{[64]}), (x_{k3}^{[1]}, \dots, x_{k3}^{[64]}) \rangle = \langle x_{k1}, x_{k2}, x_{k3} \rangle$. The property vector of a given cluster v_k is made of the corresponding cells' average HLS values: $\langle (\bar{H}_k, \sigma_{Hk}), (\bar{L}_k, \sigma_{Lk}), (\bar{S}_k, \sigma_{Sk}) \rangle = \langle (\mu_{k1}, \sigma_{k1}), (\mu_{k2}, \sigma_{k2}), (\mu_{k3}, \sigma_{k3}) \rangle = \langle y_{k1}, y_{k2}, y_{k3} \rangle$.

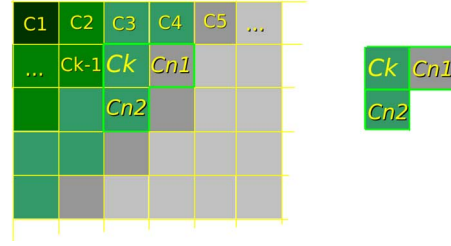


Fig. 6. Two neighbor cells c_{n1} and c_{n2} .

Cell-to-cell (1) partial comparison functions are based on the Fisher distance function. Given the two vectors $a = \langle a_1, \dots, a_n \rangle$ and $b = \langle b_1, \dots, b_n \rangle$, the Fisher distance is defined as follows:

$$D(a, b) = \frac{\sum_{j=1}^n |a_j - b_j|}{\sqrt{\sigma_a^2 + \sigma_b^2}}. \quad (8)$$

All the comparison functions obtained are greater than or equal to zero and are commutative when comparing cluster-to-cluster and cell-to-cell functions. In general, they are not associative.

2) *Segmentation Algorithm*: Now, we are able to describe in detail the clustering processing, at the end of which, each cell belongs to one, and only one, cluster in V . Collectively, the clusters will cover the entire interest area of processing. For each pseudorandomly chosen $c_k \in C$, do the following.

- 1) If c_k does not belong to any cluster, a new cluster $v_k = c_k$ is created. This cluster actually corresponds to the c_k cell only.
- 2) The distances D_{n1} and D_{n2} to the two neighbor (see Fig. 6) cells c_{n1} and c_{n2} are computed by (7). Suppose that $D_{n1} < D_{n2}$.
- 3) When c_{n1} belongs to a cluster $v_{n1} \in V$, the following rules, in the order of priority, apply.
 - a) $v2v(v_k, v_{n1}) < q_{c0}$: The clusters v_k and v_{n1} will be merged, averaging the vector properties. q_{c0} is a fixed threshold.
 - b) $(c2v(c_k, v_{n1}) < c2v(c_k, v_k)) \wedge \neg(c2v(c_{n1}, v_k) < c2v(c_{n1}, v_{n1}))$: The cell c_k will be moved from v_k to v_{n1} , and the vector properties from v_k to v_{n1} will be updated.
 - c) $\neg(c2v(c_k, v_{n1}) < c2v(c_k, v_k)) \wedge (c2v(c_{n1}, v_k) < c2v(c_{n1}, v_{n1}))$: The cell c_{n1} will be moved from v_{n1} to v_k , and the vector properties from v_k to v_{n1} will be updated.
 - d) $(c2v(c_k, v_{n1}) < c2v(c_k, v_k)) \wedge (c2v(c_{n1}, v_k) < c2v(c_{n1}, v_{n1}))$: The cells c_k and c_{n1} will be switched, and the vector properties from v_k to v_{n1} will be updated.
 - e) In the case of $D_{n1} \geq D_{n2}$, point 3 applies, replacing c_{n1} with c_{n2} .

A random selection of the cells helps to minimize the risk of obtaining a final customization that depends on the specific cell comparison order. At the end of the clustering processing, each cell will belong to one, and only one, cluster.

The results of the clustering processing are shown in Fig. 7.

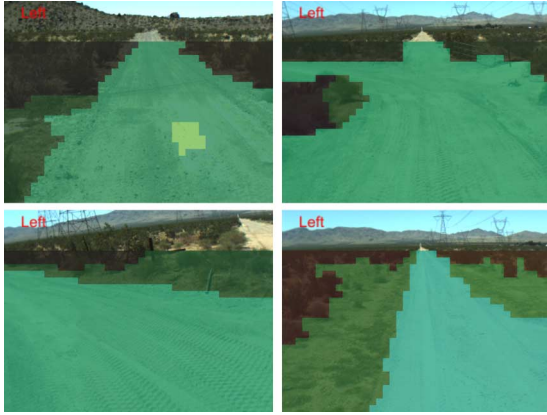


Fig. 7. Clustered images: Cells superimposed with the same color belong to the same cluster, appearing like one single homogeneous area. The color used depends on the cluster's property vector. The upper elaboration limit on the image always corresponds to 50 m and is obtained frame-by-frame by a stereo-based stabilization algorithm.

B. Decision

The decision phase tries to understand whether a cluster belongs to the road surface, using two different kinds of information:

- 1) cluster properties (as a set of pixels): homogeneity, size, and shape factors;
- 2) vehicle state: linear speed and steering angle.

The underlying idea is the following: Each cluster belongs to the road with a given probability, depending only on its own intrinsic properties: homogeneity (the higher the better), size (the higher the better), and shape factors (the closer to road shape the better) and covered free-space area (the wider the better). The probability is computed at each frame with no memory of the previous frame classifications. However, having a high probability of being a road is not sufficient, and sometimes even necessary, to be finally classified as a road. Suppose that the vehicle is running at the highest speed that is allowed by DARPA in a particular race segment and is neither steering nor decelerating but is actually running on the off-road surface. In this situation, telling to the path planner that the road is not where the vehicle is driving could be extremely risky. This is why, sometimes, even if a cluster has a low probability of being a road, the algorithm will classify it as a road, and vice versa. Total risk must be minimized, and the risk associated with an incorrect classification depends on the current vehicle state. If the vehicle is smoothly running at high speeds, one should only deliver information about the location of the road which could alter vehicle behavior if one is very sure about the information. In other words, in the decision about a possible classification, the following rule applies: Classifications with higher risks require higher probabilities (of being correct) before they can be assigned.

A maximum expected utility technique includes these risk considerations in the decision process. $U(S)$ denotes the utility of reaching state S as the consequence of decision D . This utility function assigns a number to express the desirability of state S . Instead of S , $R_i(D)$ can denote one of the possible outcomes of decision D . Each outcome can occur with a prob-

ability $P(R_i(D)|D, E)$, where E summarizes the environment status in which the decision is taken. The expected utility of decision D has the following value:

$$E(D) = \sum_{\text{each } R_i(D)} (U(R_i(D)) \cdot P(R_i(D)|D, E)). \quad (9)$$

In other words, the expected value of taking a decision is the weighted sum of the possible utility, where the weights are the respective probabilities. A rational decider would take the action with the maximum expected utility.

In the context of the road classification decision, the terms in (9) have the following interpretations. The possible decisions D are: Classify a cluster as road or off-road; the outcomes R_i are the classification was correct or incorrect, the environment E is determined by the cluster properties and vehicle state, and the utility function U depends on the consequences of the corresponding outcome on the vehicle.

Consequently, the expected utility value can be expressed as follows:

$$E(\text{road}) = P(\text{road}) \cdot U_1 + (1 - P(\text{road})) \cdot U_2 \quad (10)$$

$$E(\text{off-road}) = (1 - P(\text{road})) \cdot U_3 + P(\text{road}) \cdot U_4 \quad (11)$$

where U_1 is the utility of correctly classifying a cluster as road, U_2 is the utility of incorrectly classifying a cluster as road, U_3 is the utility of correctly classifying a cluster as off-road, and U_4 is the utility of incorrectly classifying a cluster as off-road.

1) *Cluster Growing*: The decision algorithm is implemented as a cluster-growing process.

- 1) A cluster v_k from V is randomly chosen.
- 2) If v_k is classified as off-road, another cluster is chosen, and the decision process returns to 1).
- 3) If v_k is classified as road with an expected utility E_k , we define the current maximum expected utility $E_{\max} = E_k$. Then, we perform the following.
 - a) We define $v_{k\text{grown}}$ as the union of V_k and one of its neighbors that has yet to be classified.
 - b) If $v_{k\text{grown}}$ is classified as off-road, another neighbor of v_k is chosen, and the process returns to a).
 - c) If $v_{k\text{grown}}$ is classified as road and if $E_{\max} < E_{k\text{grown}}$, then we merge V_k and its neighbor, and the process restarts from a).
- 4) When it is no longer possible to keep increasing the road surface, adding new clusters, the process ends.

The mechanism of cluster growing implements a one-shot decision process, where any decision taken at time t does not influence the decisions taken at time $t + 1$ and where decisions are taken only on the basis of the current environment and current possible outcomes (i.e., no tracking stage is adopted). The following paragraphs discuss the computation of probability and utility functions.

2) *Computing Probability Functions*: To compute $P(R_i(D)|D, E)$, a complete causal model of the world is required. However, having this causal model means having the solution to the path-detection problem, in general, because we were able to accurately decide when a cluster is actually a road or not simply by image processing. Since we have to deal with a completely

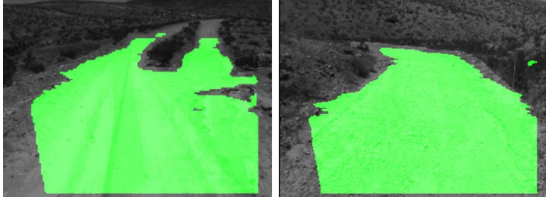


Fig. 8. Free space obtained by the stereo vision: Note that the system detects the whole drivable space in front of the vehicle.

unknown environment, we can only use an estimation of this probability distribution.

The probability function has the following form:

$$P(R_i(D)|D, E) = \sum_{j=0}^n (p_j \cdot h_j) \quad (12)$$

where p_j is the probability that would be assigned if one only knew the j th cluster property (this j th cluster property is called y_j), with h_j being the respective heuristic fixed weight. The underlying hypothesis is that the events $R_i(D)$, conditioned to the occurrence of D, y_j , are disjoint events when varying y_j . This allows the probability function to be expressed as a linear combination of simpler weighted probability functions. Moreover, the probability function might not involve the same sets of cluster properties when evaluating a cluster alone and when evaluating a set of regions already classified as road.

The cluster properties y_j that are used to compute the probability functions p_j in (12) are the following:

Name	Property
y_1	Average variance of cells' HLS values.
y_2	Number of cells (%).
y_3	Percentage of cells contained in the <i>freespace</i> .
y_4	Position and form factors 1.
y_5	Position and form factors 2.

Each property value is normalized to $[0, 1] \in \mathbb{R}$. The free space is made of the portions of the image assumed to be free of obstacles (Fig. 8). This elaboration is based on the warped image provided by the stereo algorithm. However, comparing Fig. 8 with Fig. 15 in Section V-C should be clear with how the free space is not just the obstacle complementary space but is something closer to the definition of ground surface. As will be extensively explained in the next paragraphs, the obstacle detector was designed not to detect objects like bushes, curbs, and short stones but to improve the reliability and to avoid false detections. On the contrary, path detection takes advantage in bounding its search to the ground surface. Hence, a different algorithm was developed for the path detector, which was still based on the V-disparity analysis but with a different scope.

When searching for the first road cluster, the probability function depends only on the first four properties. The fifth property gets included after the initial road cluster gets classified. The refined position and form characterizations in the fifth property can help the cluster-growing process to classify small complementary clusters.

3) *Computing Utility Functions*: The utility functions U_1, U_2, U_3 , and U_4 give a measure, in $[0, 1] \in \mathbb{R}$, of the desirabil-

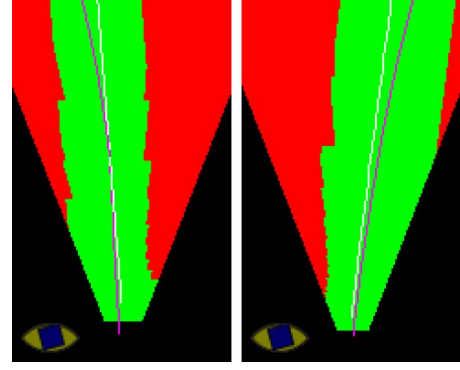


Fig. 9. Bird's eye view map of the space in front of the vehicle, as classified by the path detector: Green means road and red means off-road, whereas black means unknown (out of the field of view). The white line represents the center of the detected road, whereas the blue line is the expected truck trajectory.

ity of the actions' outcomes. Typically, correct classifications (U_1, U_3) have high utilities, whereas incorrect classifications (U_2, U_4) have low utilities. However, in some circumstances, safe operation might dictate the use of low values of U_3 and high values of U_2 . These alternate weights increase the likelihood of classifying a given section as a road—even when the estimated probability that the section is, in fact, a road is low. High values of U_2 (and U_1) make the utility of classifying the section as a road large. Low values of U_3 (and U_4) make the utility of classifying the section as off-road small.

Each outcome has an associated set of attributes vs_i , which determines its utility in terms of effect on the vehicle state. The set of attributes is assumed to be mutually utility-independent (MUI) [28]: A subset Σ of attributes is utility-independent from another subset Γ if the preference between two outcomes characterized by Σ_1, Σ_2 does not depend on the values of the corresponding Γ_1, Γ_2 . In the case of MUI, the utility functions can be expressed in the following terms:

$$U = k_1 \cdot U_1 + k_2 \cdot U_2 + k_3 \cdot U_3 + \dots + k_1 \cdot k_2 \cdot U_1 \cdot U_2 \\ + k_2 \cdot k_3 \cdot U_2 \cdot U_3 + k_1 \cdot k_3 \cdot U_1 \cdot U_3 \\ + \dots + k_1 \cdot k_2 \cdot k_3 \cdot U_1 \cdot U_2 \cdot U_3 + \dots$$

where U_i is the utility function of the i th outcome attribute, and k_i is the corresponding assigned weight.

The following vehicle-state parameters are used to compute the utility values and to represent the outcome attributes:

Name	Property
vs_1	Current truck speed.
vs_2	Current steering angle.

The higher the speed, the higher is the risk related to change the current path. The farther the computed path to the current trajectory, the higher the risk related to its classification (see Fig. 9). Hence, the equation in Section IV-B3 becomes

$$U = k_1 \cdot U_{vs_1} + k_2 \cdot U_{vs_2} + k_1 \cdot k_2 \cdot U_{vs_1} \cdot U_{vs_2}.$$

4) *Decision-Network Diagram*: Fig. 10 shows the decision-network diagram of the framework.

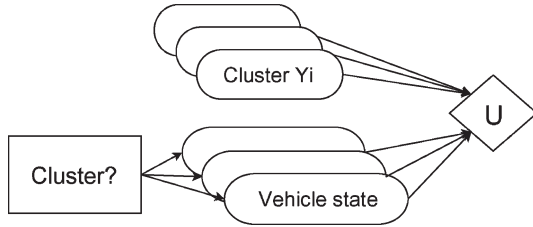


Fig. 10. Decision-network diagram. The ovals represent chance nodes: The random properties over which the decider has no influence (cluster properties). The rectangles represent decision nodes: Properties influenced by the decider's choice (properties of the vehicle state). The diamonds represent the utility functions.

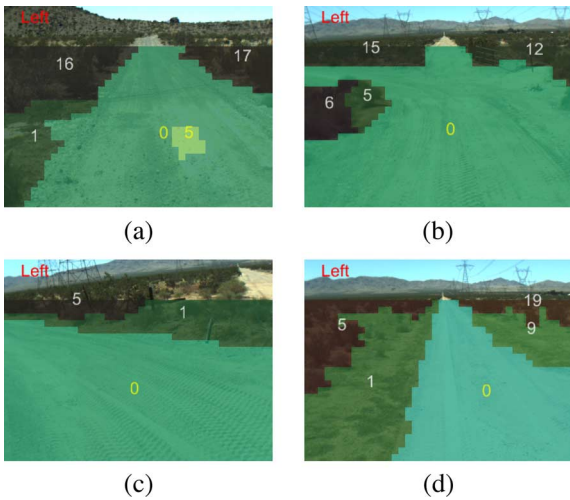


Fig. 11. Classification result. (a) The road is made of two clusters; (b) the truck is facing a junction; (c) a sharp curve; (d) a straight road. The color of the number superimposed to clusters represents the classification outcome: White means off-road, whereas yellow means road. White numbers denote the clusters classified as off-road, whereas yellow ones denote those clusters classified as road.

C. Path Detection Results

The overall path-detection-algorithm elaboration of a single frame takes about 30 ms using 320×240 -pixel color images. Some classification results are shown in Fig. 11.

V. OBSTACLE DETECTION

A. Algorithm Overview

The hardware setup of our system requires the camera pairs to have a horizontal baseline in order to perceive vertical obstacles. In these conditions, as Nedeveschi *et al.* [10] state, it is reasonable to perform a localization of homologous points on the vertical edges obtained by means of a Sobel filter. Furthermore, a Sobel-filtered image does not suffer from stereo-matching problems caused by brightness differences between cameras. Unfortunately, the image areas with constant appearance (low contrast or lack of texture) do not have many (vertical) edges and, thus, do not provide sufficient information for stereo reconstruction via a simple DSI computation. In such cases, image segmentation can still provide information for stereo reconstruction, but it does so at a high computational cost (e.g., in [29], it takes between 1 and 7 s per image). This problem is simply one specific instance of the general

phenomena noted in [3]: It is difficult to detect obstacles of any kind with a single approach. The solution to the general problem is to use more than one sensor system. In this specific example, the laser scanners mounted on the TerraMax can easily detect wide textureless obstacles; thus, the vision system design need not consider obstacles without edges. Exploiting the complementary strengths of vision and laser scanners can provide a complete real-time sensing solution.

Full 3-D stereo reconstruction is currently receiving a great deal of attention by the scientific community; nevertheless, computational time aspects are not always addressed (for example, the algorithm described in [30], which reaches very good performance in terms of 3-D reconstruction, takes 40 s to process a single pair of images). Obviously, this clashes with our real-time requirements.

As a real-time alternative, this paper proposes a two-step approach to full 3-D stereo reconstruction. The first step was shown in Section III. During the V-disparity image analysis, information about position and shape of the ground is collected. This information provides some structure that off-road environments would otherwise lack. The second step detects the obstacles with a DSI-based algorithm that uses information from the first step to reduce the region of interest prior to calculation and to filter the results after the calculation. The found obstacles, which are mapped in real-world coordinates, are then grouped following a neighbor rule.

B. DSI Computation

Before the DSI computation, the algorithm proceeds by computing a region of interest using the pitch estimation. That is, in order to reduce the computational cost, real-world regions too close to the vehicle for a successful stereo matching and the image areas well above the estimated horizon are not processed for obstacle detection.

The DSI is computed using a four-row by three-column confrontation window. The chosen window is narrow in order to allow the detection of thin obstacles and short to ease matching of ground features (as stated in Section III-B). The DSI is not computed in textureless regions.

The disparity search range is centered, considering the expected ground disparity depending on the v -coordinate in the image. Thanks to the code optimization, the DSI is computed within 15 ms with a resolution of 320×240 pixels. In the meantime, the 3-D world coordinates are computed via stereo triangulation so that the DSI also encodes range information.⁶ The wide stereo baselines used here provide a sufficiently accurate depth estimation, thus, not requiring subpixel accuracy.

C. Obstacle Search in the DSI

Many methods have been studied to extract obstacles from a DSI/range image. In road environments, Nedeveschi *et al.* [10] and Lemonde and Devy [31] exploit their knowledge about ground surface in order to delete ground matches in the range image: The remaining points are grouped into obstacles.

⁶ The set of computed 3-D world coordinates is called the range image.

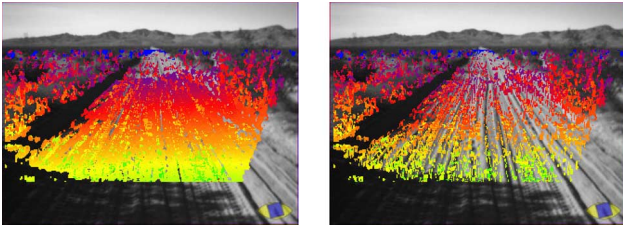


Fig. 12. DSI image before and after ground filter in a highly textured ground situation.

Anyway, as discussed earlier, the definition of a “ground plane” is not straightforward in off-road environments.

An exhaustive method, suitable for off-road environments, has been introduced in [32] and also applied in [11] and [33]. This approach evaluates connectivity of points in the range image, builds candidate obstacles, and estimates if they present a sufficiently high angle with the ground plane (i.e., they are vertical). A tree search allows then to connect close candidate obstacles and to obtain 3-D point clouds, whose densities have to be measured in order to classify them as incorrect matches or as obstacles. Unfortunately, the applications of this method are still below 1.5-Hz throughput. Furthermore, this method seems to be inadequate for thin-obstacle detection, which is one of the main targets of this paper.

Dang and Hoffman [34] use a standard flood-filling algorithm to find the connected regions (clusters) of similar disparity in the DSI, and they evaluate cluster size to decide if they represent obstacles.

In the intent of keeping the computational load under control, in our approach, we developed a series of fast filters to be applied to the DSI in order to detect disparity concentrations that are eligible to be obstacles. A full example of the filter series is shown in Fig. 15 and is explained step by step in the following lines.

Ground matches deletion: If, moving up along a DSI column, a continuously decreasing disparity is found, this is considered to be caused by the ground elements. Pixels satisfying this property are deleted. Of course, not all the terrain contribution is removed at this step because of the possible former DSI false matches. Anyway, good results are achieved on highly textured ground regions (Fig. 12), decreasing the probability of a false obstacle detection that is otherwise common in these situations.

Correlation and distance thresholding: Poor matchings due to low correlation or relative to too far away areas (low resolution) are then removed [Fig. 15(b)].

Column filter: Focusing on thin vertical obstacle search, in the next step, only the most common disparity value for each DSI column is preserved [Fig. 15(c)], turning off the pixels with a different disparity. Even if this step is very effective and significantly draws up the final detection of obstacles, in further development, it has to be considered that too much information is lost in applying this filter, and other more general approaches have to be contemplated. For example, two obstacles at different distances that are aligned on the same column of the right image are both eligible, which may interfere with each other (see Fig. 13). Generally, the farther one (being smaller in the

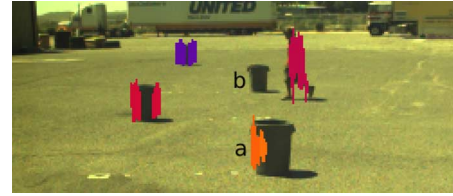


Fig. 13. Example of obstacle detection in the case of incorrect column-filter behavior. Barrel *b* is not detected because of the interference with the left edge of barrel *a*. The right edge of barrel *a* is not detected because a pedestrian lays in the same columns of the image: Even if the pedestrian is farther than the barrel, the pedestrian—being taller—attracts the focus onto himself.

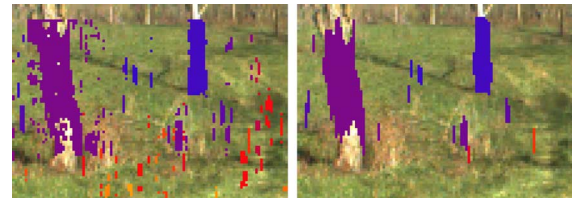


Fig. 14. Processed image of Fig. 15 before and after compactness filter (detail). In the second, the spots are represented and centered around their barycenters, and their sizes are proportional to their scores.

image) is not going to be detected, but it can happen that the missed obstacle is the closer one or even that both obstacles are missed because of the prevailing alternate columns. This system is also unsuitable for the detection of slanted thin obstacles.

The remaining pixels turned on are now divided in spots, i.e., the set of pixels still on for each column of the image. These spots are characterized by a disparity and a 3-D world coordinate in the ground plane. This coordinate is obtained by averaging the 3-D world xy -coordinates of the points forming the spot. As previously mentioned, these coordinates are contained in the range image obtained during the DSI computation.

Compactness filter: The barycenters of the spots are computed in the DSI. Then, each pixel moves toward the barycenter of the spot to which it belongs; every time it encounters a pixel on (that is part of the spot), it receives a bonus, whereas a penalty is given if it has to go through the turned-off pixels. In this way, spots obtain a score proportional to their size and compactness.

The compactness filter is very effective against false positive caused by repetitive patterns that usually cause discontinuous spots, which are formed by (correct) ground matchings and, some lines above, by incorrect matches at the same disparity (see Fig. 14). The drawback is that partially hidden obstacles may cause discontinuous spots as well and, consequently, may get lower scores.

Spots scoring: Spots receive an extra bonus if the neighbor columns are ruled by the spots featuring the same disparity. The number of columns in the neighborhood is determined in inverse proportion with the distance characterizing the spot.

Thresholding: Farther obstacles have smaller sizes in the image. To locate obstacles, Nedeveschi *et al.* [10] map the range-image data in a bird’s eye view map and looks for point concentration. To address the fact that a 3-D information is more and more sparse as the distance grows, it proportionally

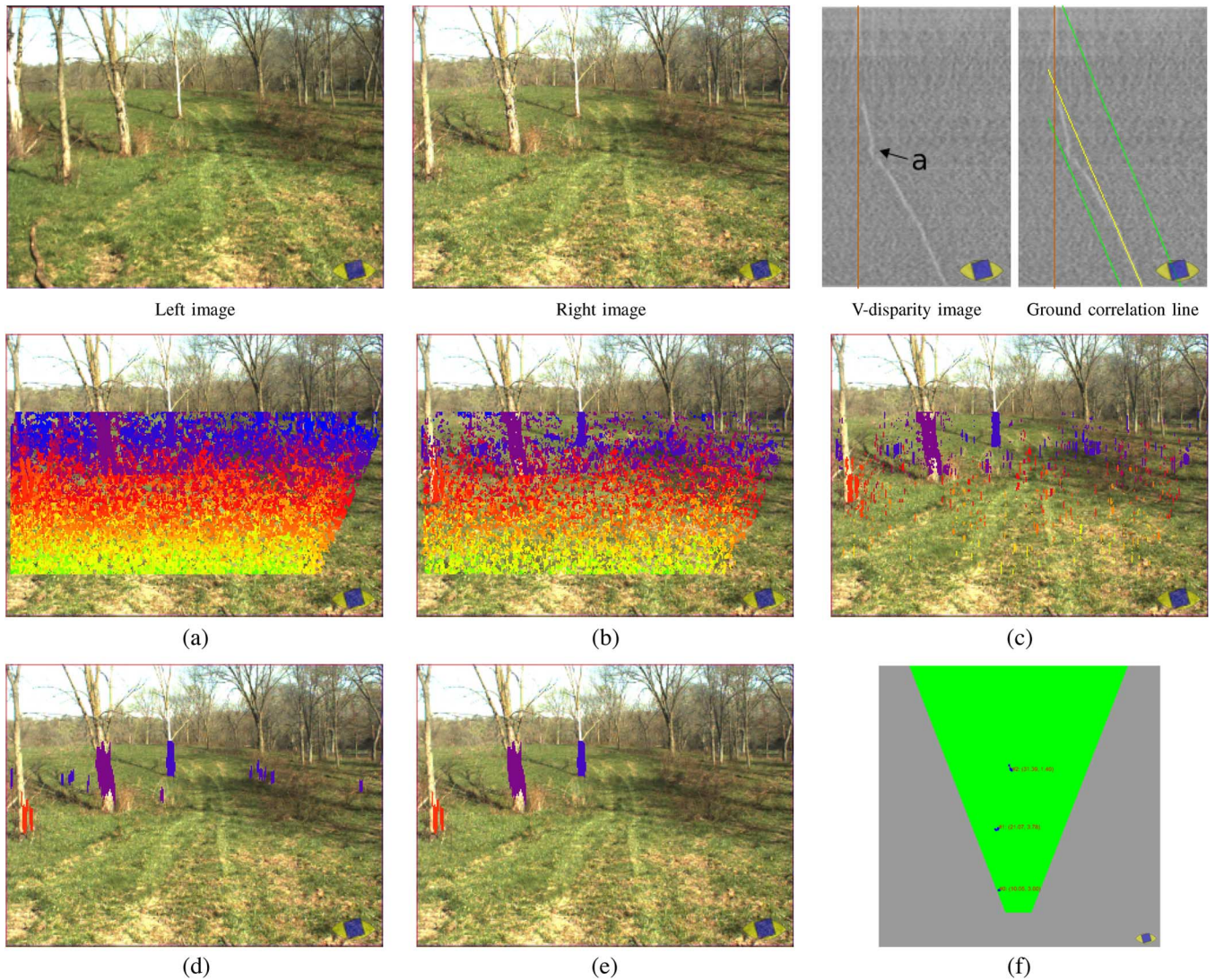


Fig. 15. Example of frame processing in a situation in which the vehicle was facing a short downhill. After the V-disparity analysis, obstacles are found via a series of filters applied to the DSI. The slope filter helps to remove the false positives that are mainly present in the region featuring a slope change. Note that, in this case, the ground filter is not effective because of a discontinuously textured terrain. In the bird's eye view map, the field of view is shown in green. (a) Original DSI. (b) After ground, correlation, and distance threshold. (c) After column filter. (d) After compactness and size filter. (e) After slope filter. (f) Obstacle bird's eye view map.

compresses the space to the distance. Since we directly look for obstacles in the image (and not in a 3-D-world representation), we keep the threshold on obstacle size (measured in pixels) in inverse proportion to the distance. Fig. 15(d) shows the results at this point of the processing.

Slope filter: As Fig. 15(d) shows, in sloped ground conditions (more precisely, when facing a concave ground), a number of false positives still endure. They are caused by ground regions that present a geometry comparable to obstacle geometry, i.e., an orientation with respect to the cameras similar to a vertical surface, resulting in a nearly constant ground disparity on adjacent lines of the images. However, this reflects on the V-disparity image as well. In the V-disparity image of the example, the letter *a* highlights the nearly constant disparity of ground in the region of images where the ground is oriented almost perpendicularly with respect to the camera optical axes. Therefore, as long as the ground occupies a prevailing area in

all the image rows, the V-disparity image contains information that allows the detection of these kinds of false positives.

For each V-disparity image row, a maximum value is found, and the disparity that caused this best match collects one point. After all rows and maxima are considered, it is possible to plot the collected statistics in a histogram (Fig. 16). This histogram is used to raise the applied obstacle threshold, specifically on each disparity, depending on its score in the histogram. This way, false positives are eliminated, obtaining the final result shown in Fig. 15(e).

This approach fails when facing an obstacle that occupies a prominent part of the image because it can modify the statistic on the V-disparity image masking the ground information. Wide obstacles are, in fact, considered as slope changes and are not detected as obstacles. This problem has yet to be addressed because other sensors mounted on the TerraMax can easily detect wide obstacles.

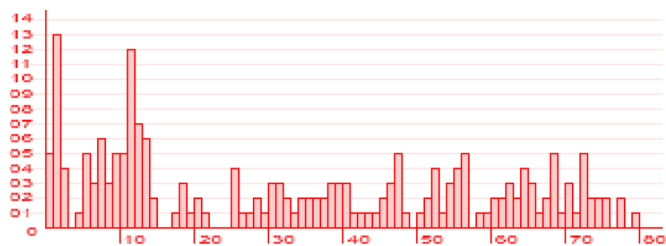


Fig. 16. Number of row maxima caused by each disparity in the V-disparity image of Fig. 15 example. Only the rows below the estimated horizon are considered.

D. Real-World Coordinate Mapping

The obstacles found in the image are not framed with bounding boxes because the algorithm does not need another obstacle classification step. As mentioned, the spots feature a 3-D world coordinate, and obstacles are obtained, grouping spots close to each other (e.g., closer than the vehicle width), using a convex hull algorithm. Fig. 15(f) shows the bird's eye view map of the obstacles obtained in Fig. 15(e). The entire filtering and real-world coordinate mapping are accomplished within 5 ms. Other output examples in different scenarios are shown in Fig. 17.

E. Obstacle-Detection Results

The algorithm thresholds were tuned in order to avoid false detections. Preparatory testing in the desert near Barstow, CA, showed the vision obstacle-detection system to be very effectively complemented by the other sensor systems. The vision could detect obstacles which the other sensors missed—namely, tall thin ones—and ran with a guaranteed 15-Hz throughput. The obstacle-detection system can perform the entire computation within 30 ms.

During the National Qualification Event prior to the 2005 DARPA Grand Challenge, the system detected all the bounding cones with only one false detection in four runs. On October 8 and 9, 2005, the TerraMax vehicle was one of the only five vehicles to complete the entire DARPA Grand Challenge course and was the only finisher exploiting vision for obstacle detection.

The data from the high-precision laser scanner helped tune the distance estimates from the vision system to meet all the requirements.

VI. CONCLUSION

This paper has presented a complex trinocular vision system for obstacle and path detection, which was specifically developed for the 2005 DARPA Grand Challenge and tested on the TerraMax autonomous vehicle. Due to the competitiveness scenario that the 2005 Grand Challenge presented, the general approach used while developing the TerraMax's sensors was to have high reliability: a minimum number of incorrect classifications and minimization of incorrect classification risks.

With accurate calibration, the obstacle detector proved to be robust to false detections, even in complex environments, with slopes, shadows, and different quality of textures. It also provided features (such as vehicle pitch) to other subsystems, like the path detector or higher level object detectors.

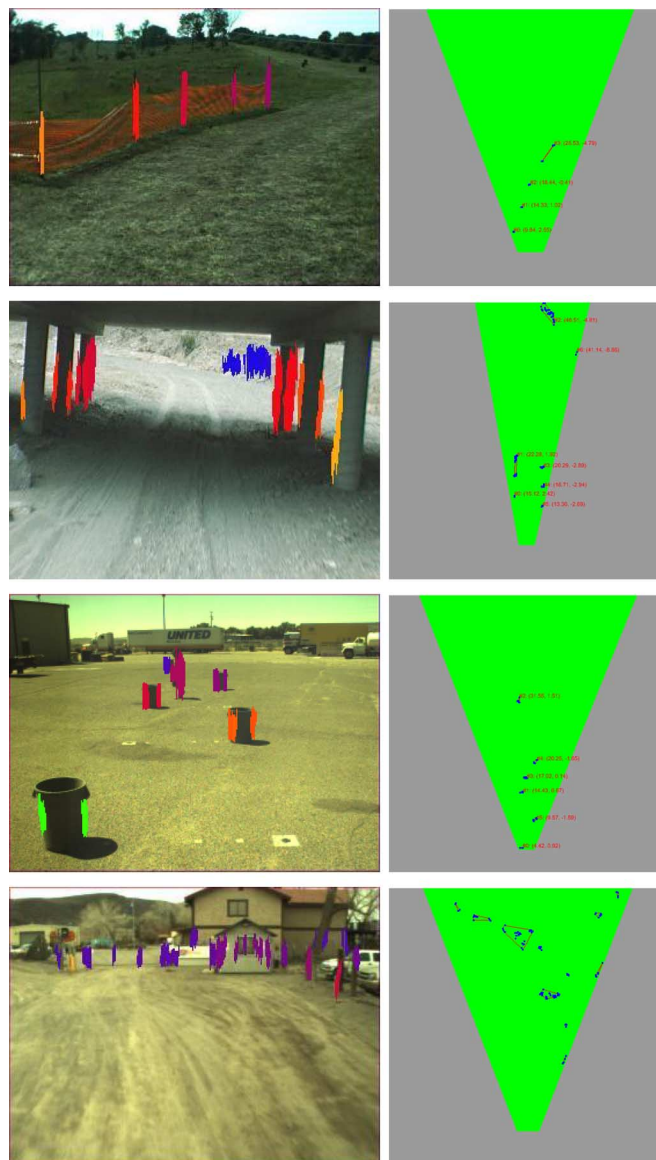


Fig. 17. Obstacle detection for different scenarios. The color of the obstacles varies with distance. In the bird's eye view map, the field of view varies because of the different baselines and camera setup. In the bottom image, a far gate in the left side is detected (in purple).

To reach the required reliability, the path-detector algorithm first divides the whole problem into a number of smaller tasks, i.e., clustering and decision. Next, accounting for the current vehicle state, it introduces some degree of high-level elaboration into the path-detection problem, allowing minimization of the risk associated with incorrect path classifications.

The overall processing of a single shot takes about 60 ms using 320×240 -pixel color images, which is fast enough to provide a reliable 10-Hz throughput and to keep room for possible additional elaborations, such as fusion with laser scanners.

ACKNOWLEDGMENT

The authors would like to thank Prof. A. Broggi for his direction and supervision of the whole project, all the members of Team TerraMax for professional support, and the friendly

atmosphere demonstrated during the project development and vehicle test. In particular, the authors would also like to thank all of the Oshkosh Truck Corporation personnel for the invaluable help they received.

REFERENCES

- [1] D. A. R. P. Administration, *Grand Challenge 2005*. [Online]. Available: <http://www.grandchallenge.org>
- [2] Oshkosh Trucks Corporation, Rockwell Collins, Univ. Parma, *Team TerraMax 2005*. [Online]. Available: <http://www.terramax.com>
- [3] R. Arturo, A. Huertas, and L. Matthies, "Evaluation of stereo vision obstacle detection algorithms for off-road autonomous navigation," in *Proc. 32nd AUVSI Symp. Unmanned Syst.*, Jun. 2005.
- [4] G. Alessandretti, A. Broggi, and P. Cerri, "Vehicle and guard rail detection using radar and vision data fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 1, pp. 95–105, Mar. 2007.
- [5] T. Kato, Y. Ninomiya, and I. Masaki, "An obstacle detection method by fusion of radar and motion stereo," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 3, pp. 182–188, Sep. 2002.
- [6] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *Proc. IEEE Workshop Stereo Multi-Baseline Vis.*, Kauai, HI, Dec. 2001, pp. 131–140.
- [7] K. Konolige, "Small vision systems: Hardware and implementation," in *Proc. 8th Int. Symp. Robot. Res.*, Hayama, Japan, Oct. 1997, pp. 111–116.
- [8] *IA32-Intel Architecture Optimization Reference Manual*, Intel, Santa Clara, CA, Jun. 2005. Order 248966-012. [Online]. Available: <http://developer.intel.com>
- [9] Z. Hu, F. Lamosa, and K. Uchimura, "A complete U-V-disparity study for stereovision based 3D driving environment analysis," in *Proc. 3DIM*, 2005, pp. 204–211.
- [10] S. Nedeveschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, T. Graf, and R. Schmidt, "High accuracy stereovision approach for obstacle detection on non-planar roads," in *Proc. IEEE INES*, Cluj Napoca, Romania, 2004, pp. 211–216.
- [11] J. van den Hauvel, J. Kleijweg, W. van der Mark, M. Lievers, and L. Kester, "Obstacle detection for people movers using vision and radar," in *Proc. 10th World Conf. Intell. Transp. Syst. Services*, Madrid, Spain, 2003.
- [12] A. Broggi, C. Caraffi, R. I. Fedriga, and P. Grisleri, "Obstacle detection with stereo vision for off-road vehicle navigation," in *Proc. Int. IEEE Workshop Mach. Vis. Intell. Vehicles*, San Diego, CA, Jun. 2005, p. 65.
- [13] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection on non flat road geometry through V-disparity representation," in *Proc. IEEE Intell. Vehicles Symp.*, Versailles, France, Jun. 2002, pp. 646–651.
- [14] R. Labayrade and D. Aubert, "A single framework for vehicle roll, pitch, yaw estimation and obstacles detection by stereovision," in *Proc. IEEE Intell. Vehicles Symp.*, Columbus, OH, Jun. 2003, pp. 31–36.
- [15] A. Broggi, S. Cattani, P. P. Porta, and P. Zani, "A laser scanner-vision fusion system implemented on the TerraMax autonomous vehicle," in *Proc. IEEE Int. Conf. IROS*, Beijing, China, Oct. 2006, pp. 111–116.
- [16] M. Bertozzi and A. Broggi, "GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Trans. Image Process.*, vol. 7, no. 1, pp. 62–81, Jan. 1998.
- [17] R. Labayrade, J. Douret, and D. Aubert, "A multi-model lane detector that handles road singularities," in *Proc. IEEE ITSC*, Toronto, ON, Canada, Sep. 2006, pp. 1143–1148.
- [18] C. Rasmussen, "Combining laser range, color, and texture cues for autonomous road following," in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, May 2002, pp. 4320–4325.
- [19] A. Broggi and S. Cattani, "An agent based evolutionary approach to path detection for off-road vehicle guidance," *Pattern Recognit. Lett.—Special Issue on Evolutionary Computer Vision and Image Understanding*, vol. 27, no. 11, pp. 1164–1173, Aug. 2006.
- [20] A. Broggi, M. Cellario, P. Lombardi, and M. Porta, "An evolutionary approach to visual sensing for vehicle navigation," *IEEE Trans. Ind. Electron.*, vol. 50, no. 1, pp. 18–29, Feb. 2003.
- [21] C. Tan, H. Tsai, C. Tommy, and S. Michael, "A multi-model lane detector that handles road singularities," in *Proc. IEEE ITSC*, Toronto, ON, Canada, Sep. 2006, pp. 1143–1148.
- [22] D. Mateus, G. Avina, and M. Devy, "Robot visual navigation in semi-structured outdoor environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, Barcelona, Spain, Apr. 2005, vol. 1, pp. 4691–4696.
- [23] N. B. Karayiannis and M. M. Randolph-Gips, "Soft learning vector quantization and clustering algorithms based on non-Euclidean norms: Single norm algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 423–435, Mar. 2005.
- [24] P. H. Batavia and S. Singh, "Obstacle detection using adaptive color segmentation and color stereo homography," in *Proc. IEEE Int. Conf. Robot. Autom.*, Seoul, Korea, May 2001, vol. 1, pp. 705–710.
- [25] T.-K. Kim and J. Kittler, "Locally linear discriminant analysis for multimodally distributed classes for face recognition with a single model image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 318–327, Mar. 2005.
- [26] J. Smith, *Decisions Analysis*. London, U.K.: Chapman & Hall, 1988.
- [27] I. Matzkevich and B. Abramson, "Decision analytic networks in artificial intelligence," *Manage. Sci.*, vol. 41, no. 1, pp. 1–22, Jan. 1995.
- [28] R. Keeney, J. Hammond, and H. Raiffa, *Smart Choices: A Guide to Making Better Decisions*. Boston, MA: Harvard Univ. Press, 1999.
- [29] F. Estrada and A. Jepson, "Quantitative evaluation of a novel image segmentation algorithm," in *Proc. CVPR*, San Diego, CA, Jun. 2005, vol. 2, pp. 1132–1139.
- [30] J. Sun, Y. Li, S.-B. Kang, and H.-Y. Shum, "Symmetric stereo matching for occlusion handling," in *Proc. CVPR*, San Diego, CA, Jun. 2005, vol. 2, pp. 399–406.
- [31] V. Lecomte and M. Devy, "Obstacle detection with stereovision," in *Proc. MECHROB*, Aachen, Germany, Sep. 2004, vol. 3, pp. 919–924.
- [32] A. Talukder, R. Manduchi, L. Matthies, and A. Rankin, "Fast and reliable obstacle detection and segmentation for cross-country navigation," in *Proc. IEEE Intell. Vehicles Symp.*, Versailles, France, 2002, pp. 610–618.
- [33] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," *Auton. Robots*, vol. 18, no. 1, pp. 81–102, Jan. 2003.
- [34] T. Dang and C. Hoffmann, "Fast object hypotheses generation using 3D position and 3D motion," in *Proc. Int. IEEE Workshop Mach. Vis. Intell. Vehicles*, San Diego, CA, Jun. 2005, p. 56.



Claudio Caraffi received the M.Sc. degree in computer engineering from the Università di Parma, Parma, Italy, in 2004, where he is currently working toward the Ph.D. degree in information technology with the Artificial Vision and Intelligent Systems Laboratory (VisLab), Dipartimento di Ingegneria dell'Informazione, Università di Parma.

He is a member of Team TerraMax. He participated in the 2004 and 2005 DARPA Grand Challenge and is involved in the 2007 DARPA Urban Challenge. He has been working on problems related to system calibration, image stabilization, real-time obstacle detection, lane detection, and traffic-sign recognition. His research interests are mainly in the area of computer vision applied in intelligent vehicles.



Stefano Cattani received the M.Sc. degree in computer engineering from the Università di Parma, Parma, Italy, in 2004, where he is currently working toward the Ph.D. degree in information technology with the Artificial Vision and Intelligent Systems Laboratory (VisLab), Dipartimento di Ingegneria dell'Informazione, Università di Parma.

He participated in the 2004 and 2005 DARPA Grand Challenge and is involved in the 2007 DARPA Urban Challenge as a member of Team TerraMax. His research works are mainly focused on path and vehicle detection based on monocular vision in the automotive field.



Paolo Grisleri received the Dr.Eng. and Ph.D. degrees in computer engineering from the Università di Parma, Parma, Italy, in 2002 and 2006, respectively.

Since 2002, he has been a Temporary Researcher with the Artificial Vision and Intelligent Systems Laboratory (VisLab), Dipartimento di Ingegneria dell'Informazione, Università di Parma. His research is mainly focused on computer vision, data-acquisition techniques, and system architectures for advanced driver-assistance systems.