ACTA IMEKO
ISSN: 2221-870X
June 2015, Volume 4, Number 2, 90-99

A fast and low-cost vision-based line tracking measurement system for robotic vehicles

Daniele Fontanelli[1], David Macii[1], Tizar Rizano[2]

[1] *Department of Industrial Engineering, University of Trento, Via Sommarive 9, 38123 Trento, Italy*
[2] *Department of Information Engineering and Computer Science, University of Trento, Via Sommarive 5, 38123 Trento, Italy*

ABSTRACT
Localization and tracking systems are nowadays a quite common solution for automated guided vehicles (AGV) in industrial environments. The bunch of technological solutions developed in this sector is now being revitalized by their applications in service robots, e.g. for safe navigation in crowded public spaces or in autonomous cars. Related to this field are robotic competitions and races. In this particular scenario, the robot often has to track a line painted on the ground. Line tracking techniques typically rely on light dependent resistors (LDR), photo-diodes or photo-transistors detecting the light generated by normal or infrared Light Emitting Diodes (LEDs), arrays of electric inductance sensors or vision systems. Crucial issues for line tracking are: accuracy and reliability in estimating the direction of the moving vehicle with respect to the line and processing speed. Such problems are particularly critical when high-speed mobile robots are considered. To address this issue, in this paper a vision-based technique of moderate computational complexity is described. The proposed solution has been implemented on a low-cost embedded platform and relies on a high frame rate light contrast sensor, a tailored RANSAC-based algorithm and a Kalman filter. The reported experimental results prove that the proposed solution is able to track the direction of the vehicle in real-time even when the field of view of the camera is limited and the vehicle moves at high speed.

Section: RESEARCH PAPER

Keywords: Kalman filters; vehicular technologies; image processing; direction estimation; real-time; random sample and consensus

Citation: Daniele Fontanelli, David Macii, Tizar Rizano, A fast and low-cost vision-based line tracking measurement system for robotic vehicles, Acta IMEKO, vol. 4, no2, article 16, June 2015, identifier: IMEKO-ACTA-04 (2015)-02-16

Editor: Paolo Carbone, University of Perugia, Italy

Received February 9, 2015; **In final form** June 12, 2015; **Published** June 2015.

Corresponding author: David Macii, e-mail: david.macii@unitn.it

1. INTRODUCTION

Sensors and measurement science play a key role in the development of industrial automation and robotics [1]. One well-known fundamental and crucial problem in robotics is path planning-and-tracking [2]. In order to address this issue, a robot first has to identify the wanted path by sensing the environment (e.g. by recognizing suitable landmarks and visual cues). Secondly, it should be able to estimate steadily its position with respect to the planned trajectory, so that the robot controller can efficiently follow the desired path [3], [4]. One of the simplest and most widely adopted solution to this purpose, especially for Automated Guided Vehicles (AGVs) in industrial environments is line tracking [5], [6]. Line tracking is also often used to evaluate the ability of new robot prototypes to follow a given trajectory [7].

Typical sensors for this kind of applications include reflective infra-red Light Emitting Diodes (LEDs) coupled with suitable photo-diodes, photo-transistors or light dependent resistors (LDRs) [8], [9], arrays of electric inductance sensors [10], and vision-based systems [11]-[13]. While optical or electric inductance sensors are very cheap, they have also a very limited reading range. Moreover, at least two of such sensors are generally needed for correct line detection and to estimate whether the vehicle's position is left or right of the reference line. The vision-based line detection systems are instead advantageous because much information can be extracted from a sequence of collected pictures. Of course, in all cases sensor accuracy, range and speed are essential to ensure good performances and real-time behavior.

Stemming from industrial robotic applications, the technical solutions developed in this area of research have been also gradually applied to service robots, including autonomous guidance systems [14], [15], intelligent vehicles [16], [17], and road vehicles both for hazard detection [18], [19], and fully automatic guidance [20]. In this context, robotic vehicle competitions have considerably thrust the development of smart sensing systems. The main problem that still hinders the effectiveness of sensing solutions for position tracking in challenging scenarios (e.g. in robotic races) is the variability of the environment (especially in outdoor scenarios, where the speed of a robot can be quite higher than indoors) and the possible lack of known cues. In this respect, vision-based solutions have gained an undisputed leading role mainly due to their flexibility and ability to estimate multiple quantities with a single measurement system. Unfortunately, this increased flexibility comes at the price of an increased computational load as well. In this paper, we will focus on a novel vision-based measurement technique and on a system prototype that perfectly fits the industrial domain, as it is able to improve accuracy, robustness and speed in estimating both the direction and the position of a robotic vehicle with respect to a painted line, thus supporting efficient control, as preliminary reported in [21]. In general, the problem of line detection is closely related to the classic problem of line recognition in images [22], [23], although it is made more difficult by time-varying light conditions and by the robot's dynamics. In general, the performances of vision-based line detection systems are limited by robustness and speed issues, which in turn depend on camera frame rate, camera resolution and algorithm complexity. In this respect, one of the most famous and effective algorithms for line recognition is the so-called *Hough transform* [24]. However, this algorithm needs line clustering in the image space and it is quite heavy from the computational point of view, although several optimizations have been proposed in the last years, e.g. through randomization [25], or hierarchical image partitioning [26].

As a result of the availability of increasingly powerful embedded platforms, several other image processing algorithms have been proposed over the last few years, e.g. based on a customized image segmentation [27], fuzzy logic [28], or the Viterbi algorithm [29]. All of them rely on standard cameras and are characterized by a significant computational burden. Some of the solutions proposed in literature for path detection and tracking combine specifically conceived image features [30], [31], known path models [32], statistical methods [33] or a combination of particle filtering and artificial intelligence [34]. Another important field of research, which is somehow related to the problem at hand, but is more focused on lane and obstacle detection rather than on path tracking, is described, for instance, in [35].

The technique described in this paper is instead based on a simple algorithm, which works well also on binary low-resolution images, such as those collected by a special high-frame-rate light contrast sensor. This sensor ensures a straightforward detection of the line edges (provided that the colors of the line and of the background are different enough) with minimum bandwidth and latency requirements. It is worth emphasizing that the light contrast sensor is supposed to look at the road surface (i.e. with the camera image plane approximately parallel to the ground), in order to recognize the painted line only. In this way, the probability of recording unwanted objects that may perturb line detection is much smaller than using a front camera.

The proposed approach is explicitly conceived to support automated vehicle control over optimal paths [36], although the control problem is out of the scope of this paper. Line detection is performed through a RANdom Sample And Consensus (RANSAC) algorithm combined with a Kalman filter. RANSAC is a general guess–and–test method that randomly chooses a hypothesis in the measurement space and then scores its trustworthiness a posteriori [37]. RANSAC has been used in many contexts, including trajectory estimation [38] and, more recently, even in road applications [39], [40]. However, the solution described in this paper is faster and it is expected to be more accurate and more robust than the basic RANSAC algorithm preliminarily described and analyzed through simulations in [40]. In fact, the additional Kalman filter prevents sudden and unnatural jumps in the estimated vehicle direction. The algorithm has been implemented and tested on a simple embedded platform. Low cost is indeed a further important feature of the system developed.

The rest of the paper is structured as follows. At first in Section 2 the model underlying the theoretical problem is described. Then, Section 3 deals with the description of the estimation algorithm. Finally, in Section 4 the implementation details as well as several experimental results in different conditions are reported.

2. MODEL DESCRIPTION

2.1. Vision system model

As shortly explained in Section 1, the vision system is supposed to observe the road or floor surface just to detect a painted line. If ${}^{w}P = [{}^{w}x, {}^{w}y, {}^{w}z]^T$ is a point in the reference frame $\langle W \rangle$ defined by axes X_w, Y_w and Z_w (with plane $\pi_w = X_w \times Y_w$ lying on the ground as shown in Figure 1(a)), the equations of two parallel lines in space (namely the line edges) are given by:

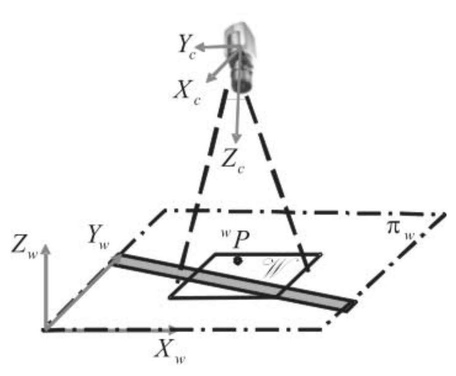
$$\begin{aligned} {}^{w}P_l(\lambda) &= {}^{w}P_a + \lambda\left({}^{w}P_b - {}^{w}P_a\right) \\ {}^{w}P_r(\lambda) &= \left({}^{w}P_a + {}^{w}o\right) + \lambda\left({}^{w}P_b - {}^{w}P_a\right) \end{aligned} \tag{1}$$

where $\lambda \in \mathrm{R}$, ${}^{w}P_a$ and ${}^{w}P_b$ are two given points belonging to the left edge and ${}^{w}o = [{}^{w}x_o, {}^{w}y_o, {}^{w}z_o]^T$ is the *offset* vector between the two lines. Let $\langle I \rangle$ be an additional reference frame defined by axes X_i, Y_i and Z_i so that $\pi_i = X_i \times Y_i$ includes the image plane and the origin of $\langle I \rangle$ coincides with the *principal point* of the camera, as shown in Figure 1(b). In such a frame, every point lying on the image plane has the Z_i coordinate equal to zero. If axes X_i and Y_i are parallel to X_w and Y_w, respectively, then the image coordinates ${}^{i}p$ of a generic point ${}^{w}P$ in the field of view of the camera are given by (see Appendix A for reference):

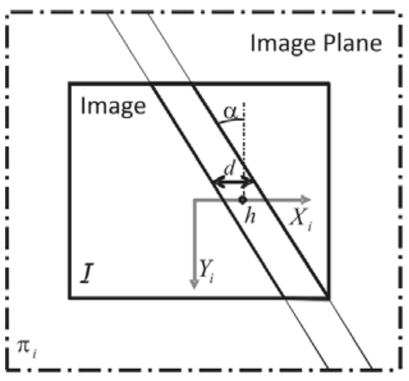
$${}^{i}p = \begin{bmatrix} {}^{i}x \\ {}^{i}y \end{bmatrix} = \begin{bmatrix} f_x \dfrac{{}^{w}y - t_y}{{}^{w}z - t_z} \\ f_y \dfrac{{}^{w}x - t_x}{{}^{w}z - t_z} \end{bmatrix} \tag{2}$$

where f_x and f_y are the focal lengths along axes X_i and Y_i, respectively, and $t_w = [t_x, t_y, t_z]^T$ is the *translation vector* expressing the coordinates of the camera pin-hole in the frame $\langle W \rangle$. Therefore, if the line equations (1) are plugged into (2), the coordinates of the points belonging to the line edges projected onto the image plane are:

$$\begin{aligned} {}^{i}p_l(\lambda) &= {}^{i}p_a + \lambda\left({}^{i}p_b - {}^{i}p_a\right) \\ {}^{i}p_r(\lambda) &= \left({}^{i}p_a + {}^{i}o\right) + \lambda\left({}^{i}p_b - {}^{i}p_a\right) \end{aligned} \tag{3}$$



(a)



(b)

Figure 1. Vision system model and reference frames: (a) perspective view and (b) top view.

where ${}^ip_a = [{}^ix_a, {}^iy_a]^T$ and ${}^ip_b = [{}^ix_b, {}^iy_b]^T$ are two points of the left edge mapped onto the image plane and ${}^io = [{}^ix_o, {}^iy_o]^T$ is the *image offset* vector between the left and the right edge. Since ${}^io$ can be chosen arbitrarily, in the following we assume that ${}^ix_o = d \neq 0$ and ${}^iy_o = 0$. As a consequence, d is the distance between the line edges along axis X_i, as depicted in Figure 1(b). Notice that if the image plane and ground are approximately parallel, three-dimensional parallel lines are mapped into two-dimensional parallel lines. The same considerations hold also for the orientation angles of each line assuming that $f_x = f_y$, as it often occurs in practice. Albeit these two assumptions are not perfectly true in a real scenario [41], a slight change of perspective can be easily addressed through *inverse perspective mapping* (IPM) [42]. Moreover, possible time-varying fluctuations of the image plane (e.g. due to vehicle vibrations or jolts), are generally negligible compared with the intrinsic fast variability of the collected images.

2.2. Problem formulation and estimation model

After detecting the line, the goal of the measurement system is to estimate the position and the orientation of the robotic vehicle with respect to the line itself in real-time. In particular, the quantities to be measured are:

- h, which is the horizontal coordinate of the intersection point between the line centroid and axis X_i, i.e.

$$h = \left({}^ix_a + \frac{d}{2}\right) - {}^iy_a\left(\frac{{}^ix_b - {}^ix_a}{{}^iy_b - {}^iy_a}\right) \tag{4}$$

- and $\alpha \in [0, \pi[$, namely the angle between the parallel line edges and axis Y_i, i.e.

$$\alpha = \mathbf{arctan}\left(\frac{{}^iy_b - {}^iy_a}{{}^ix_b - {}^ix_a}\right) \tag{5}$$

The meaning of these parameters is shown in Figure 1(b). Notice that $h \to \infty$ for lines that are parallel to X_i, axis, whereas $(h,\alpha) = (0,0)$ when the line is perfectly vertical and exactly in the center of the image. As explained in Section 3, the values of (h,α) associated with a given image are used as a prior for parallel line detection in a newly acquired image. In this way, the computation time is reduced by constraining the RANSAC-based line search in a specific subset of the pixels of every collected image. To this purpose, (4) and (5) must be inverted to recover (3). Unfortunately, the pair (h,α) alone is not sufficient to this end. Indeed, for any given pair, a very large (ideally infinite) number of parallel lines exist. In stricter theoretical terms, the position of the parallel line edges is *unobservable* using just h and α. In fact, observability is guaranteed only if d is available in (4). Therefore, also parameter d needs to be estimated by the algorithm. To this purpose, we can rely on a simple dynamic model, whose state is the vector $q=[h, \alpha, d]^T$. In general, the dynamic of q is a function of both the line to be tracked and the motion of the camera. However, if the field of view of the camera is much shorter than the radius of curvature of the wanted path, the effect of the line curvature in the image plane is negligible, thus greatly simplifying both the line recognition problem and the model. As a consequence, the system dynamic due only to the motion of the camera can be simply modelled as follows:

$$\begin{bmatrix} \dot{h} \\ \dot{\alpha} \\ \dot{d} \end{bmatrix} = \begin{bmatrix} v_h \\ v_\alpha \\ v_d \end{bmatrix} \Rightarrow \dot{q} = v \tag{6}$$

where $v = [v_h, v_\alpha, v_d]^T$, namely the input of the system, is the speed vector. Note that v_h depends mostly on the speed components of the camera along the plane of motion, v_α is related to the angular speed of the image plane of the camera, and v_d depends on the occasional motion of the camera along the axis orthogonal to the ground surface because of potholes, humps, or sporadic changes of the line width. In any case, the time evolution of the state variables of (6) is affected by the uncertainty associated with the measurement of $v = [v_h, v_\alpha, v_d]^T$. This in turn may require a dedicated inertial platform when involved scenarios are considered.

3. ALGORITHM DESCRIPTION

As stated in Section 1, the algorithm for line detection and state variable estimation relies on the combination of a tailored RANSAC algorithm and a Kalman filter. The rationale behind the combination of such techniques is related to the different nature of the uncertainty sources affecting the model parameters. On one hand, the unknown probability density function related to the estimation of q can be considered as multimodal for the presence of both outliers and noise in every grabbed image. For instance, if no accurate speed values are available, the components of v in (6) can be described just stochastically (e.g. using the variance of available data). This assumption justifies the choice of a randomized, multi-hypothesis algorithm like RANSAC. On the other hand, as a camera cannot move instantaneously in multiple different directions, the distribution of the fluctuations due to its motion must be definitely unimodal. Therefore, the multi-modal

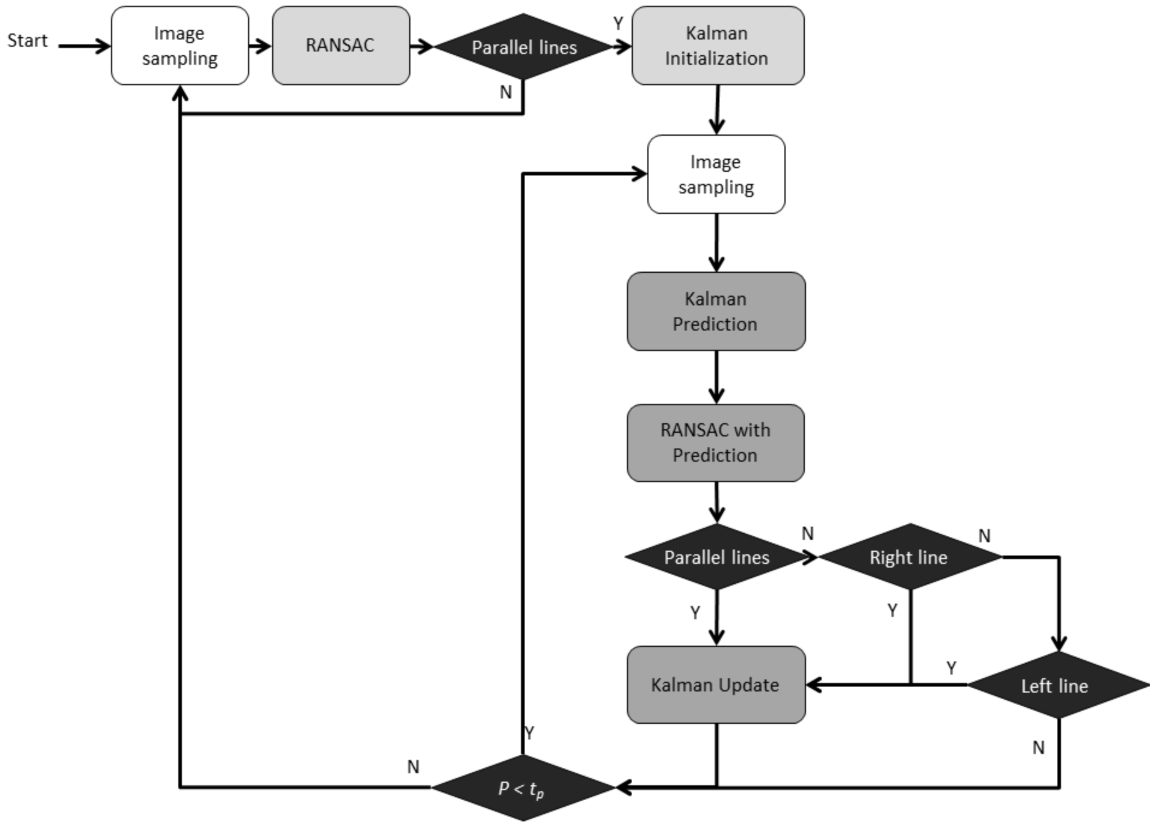


Figure 2. Flow-chart of the algorithm. The RANSAC algorithm runs in two different modes, i.e. either using the prior information returned by the Kalman filter or by processing the whole image for initialization.

RANSAC approach takes advantage of the Kalman filter principal mode defined by (6), since the Kalman filter is able to reduce such fluctuations. When the system starts, the Kalman filter is initialized as soon as the line is clearly detected in the image plane using RANSAC without any prior. In this preliminary phase, a longer execution time is tolerated in order to have an accurate first guess. Then, the estimation algorithm performs iteratively the following three steps: *Kalman-based prediction*, *RANSAC-based road line recognition* and *Kalman-based update*. The flow chart of the proposed algorithm is shown in Figure 2. Note that RANSAC operates between the prediction and the update steps of the Kalman filter. Since the RANSAC algorithm estimates the parallel lines in the image space, on one hand it benefits from Kalman prediction results as a prior; on the other it is used to generate the measurement data for the following update step. In the next subsections, the three steps mentioned above as well as the initialization criteria of the Kalman filter are described in detail.

3.1. Kalman filter initialization

When the algorithm starts, two quantities are initialized in the Kalman filter, i.e.

- the initial system state $q(0)$, which is set equal to the values returned by the first iteration of the RANSAC algorithm on a full image;
- the initial value of the state covariance matrix $P(0)$, which is set equal to 1/10 of the input covariance matrix Q.

The covariance matrix Q is a constant diagonal matrix, whose elements are the estimated variances of v_b, v_a and v_d. Observe that, unlike how it is commonly done in most Kalman filters, in this case the initial state covariance matrix is one order of magnitude smaller than Q. This is due to the fact that the initial guess $q(0)$ has usually a high accuracy since it results from the application of the RANSAC algorithm to a full image.

3.2. Kalman filter prediction step

The Kalman filter is executed anytime a new image is available. Hence, the model reported in (6) is discretized with a sampling period Δt equal to the inverse of the frame rate. The discretized system model is then simply given by $\bar{q}(t+\Delta t) = q(t) + v(t)\Delta t$, where $\bar{q}(t+\Delta t)$ is the predicted state of system (6) at time $t + \Delta t$. Since no knowledge about the motion of the camera is assumed, the state prediction equation is $\bar{q}(t+\Delta t) = q(t)$. Similarly, the predicted covariance matrix is given by $\bar{P}(t+\Delta t) = P(t) + Q$, where $P(t)$ is the covariance matrix of the state variables in (6) and Q is the model input covariance matrix defined in Section 3.1.

3.3. RANSAC-based line recognition

In general, the purpose of RANSAC is to find a subset $\mathcal{S}^*$ of a set $\mathcal{S}$ containing N data, such that its elements fit an instance $\mathcal{M}^*$ of a model $\mathcal{M}$ (depending on $n \le N$ parameters) within a user-defined tolerance threshold s_r. In the case considered, $\mathcal{M}$ refers to the parallel line equations defined in (3), and $\mathcal{S}$ is the set of camera pixels where the probability of finding the road line edges is maximum. In the worst case, $\mathcal{S}$ coincides with the whole pixel matrix and N coincides with the number of pixels. However, even if $\mathcal{S}$ changes anytime a new frame is collected, it can be properly reduced to the region of interest by using the prior information obtained from the Kalman filter prediction step. Of course, such a reduction does not take place during the initialization of the Kalman filter (see Figure 2 for reference).

The RANSAC-based line detection relies on (3), which depends on n=3 independent parameters, i.e. ${}^{i}p_a$, ${}^{i}p_b$ and ${}^{i}o = [d, 0]^T$. On the basis of these assumptions, the main steps of the RANSAC algorithm conceived for the intended application are briefly summarized in the following.

- Starting from k=1, n=3 pixels are randomly extracted from $\mathcal{S}$ to create a subset $\mathcal{S}_k$ composed by ${}^{i}p_{ak}$, ${}^{i}p_{bk}$ and ${}^{i}p_{ak}$ $+{}^{i}o_k$. If some prior information is available (i.e. the output $\overline{q}(t+\Delta t)$ of the prediction step) the choice of the subset $\mathcal{S}_k$ is constrained in the region determined by $\overline{q}(t+\Delta t)$. The selected points are then used to create a model instance $\mathcal{M}_k$, namely two parallel lines based on (3).
- Afterwards, $\mathcal{M}_k$ is used to determine the subset of pixels $\mathcal{S}_k^* \subset \mathcal{S}$ that are likely to belong to the line edges within a tolerance interval $\pm s_t$. $\mathcal{S}_k^*$ is usually referred to as the *consensus set* or the set of *inliers*. In practice, this set consists of two disjoint subsets $\mathcal{S}_{kl}^*$ and $\mathcal{S}_{kr}^*$ corresponding to the left and right line edges, respectively. If the prior information provided in the prediction step of the Kalman filter is available, threshold s_t can be modified according to the diagonal elements of the predicted covariance matrix $\overline{P}(t+\Delta t)$;
- Finally, this procedure starts over and a new set $\mathcal{S}_{k+1}$ of n random points is chosen from $\mathcal{S}$.

In a typical RANSAC algorithm, the iterative approach ends as soon as the number of elements in $\mathcal{S}_k^*$ exceeds a given threshold. However, due to the time-varying uncertainty contributions affecting the problem at hand, no a priori outlier stochastic description is available. As a consequence, in this case the iterative process stops as soon as a maximum number of iterations K is reached. The value of K in presence of outliers is given by [37]

$$K = \frac{\mathbf{log}(1-p)}{\mathbf{log}\left(1-w^n\right)} \tag{7}$$

where p is the wanted probability to select the correct model if the set $\mathcal{S}$ is sampled K times and w is the probability that one of the chosen points is actually an inlier. In the presented solution the value of K is computed and updated in real-time.

Let $\mathcal{S}_l^*$ and $\mathcal{S}_r^*$ be the sets with the largest number of inliers for the left and right edges, respectively, after K iterations, with $\mathcal{S}_l^* \cap \mathcal{S}_r^* = \varnothing$ and $\mathcal{S}_l^* \cup \mathcal{S}_r^* = \mathcal{S}^*$. If $L^* = ({}^{i}p_a^*, {}^{i}p_b^*, {}^{i}o^*)$ is the corresponding set of parameters to be replaced into (3), the optimal set of values finally results from

$$\widetilde{L} = \underset{L}{\mathbf{argmin}}\left(\sum_{{}^{i}p\in\mathcal{S}_l^*}\underset{\lambda}{\mathbf{min}}\left\|{}^{i}p-{}^{i}p_l^*(\lambda)\right\|^2+\sum_{{}^{i}p\in\mathcal{S}_r^*}\underset{\lambda}{\mathbf{min}}\left\|{}^{i}p-{}^{i}p_r^*(\lambda)\right\|^2\right) \tag{8}$$

where $\widetilde{L} = \left({}^{i}\widetilde{p}_a, {}^{i}\widetilde{p}_b, {}^{i}\widetilde{o}\right)$. Given that $\widetilde{h}$ and $\widetilde{\alpha}$ are computed from $\widetilde{L}$ using (4) and (5), respectively, and recalling that ${}^{i}\widetilde{o} = \left[\widetilde{d}, 0\right]^T$, the output of the RANSAC-based line recognition algorithm is $\widetilde{q} = \left[\widetilde{h}, \widetilde{\alpha}, \widetilde{d}\right]^T$.

3.4. Kalman filter update step

The elements of $\widetilde{q}$ represent the measurement values to be injected into the Kalman filter during the update step.

In particular, the Kalman gain K_g is given by

$$K_g = \overline{P}(t+\Delta t)\left(\overline{P}(t+\Delta t)+\mathrm{R}\right)^{-1} \tag{9}$$

where R is the covariance matrix of $\widetilde{q}$ and results from the residuals of (8). Since the motion of the camera is not included in the model, the Kalman filter strongly relies on the available measurement data. In particular, the updated values of both the state and the system covariance matrix result from

$$\begin{aligned} q(t+\Delta t) &= \overline{q}(t+\Delta t) + K_g\left[\widetilde{q} + \overline{q}(t+\Delta t)\right] \\ P(t+\Delta t) &= \left(I_3 - K_g\right)\overline{P}(t+\Delta t)\left(I_3 - K_g\right)^T + K_g R K_g^T \end{aligned} \tag{10}$$

where the Joseph form has been used to express the updated covariance matrix $P(t + \Delta t)$, thus preventing numerical instability. Note that the low computational burden of the Kalman filter update step is very suitable for an embedded implementation.

It may happen that, if an image is heavily corrupted by structured or unstructured outliers (e.g. illumination problems, shadows, small potholes, faded paint), the RANSAC algorithm fails in finding a good estimate of the parallel line edges. Such situations are tolerated to a certain extent thanks to the information retained by the Kalman filter. In such cases, the Kalman filter update step simply becomes

$$\begin{aligned} q(t+\Delta t) &= \overline{q}(t+\Delta t) \\ P(t+\Delta t) &= \overline{P}(t+\Delta t) \end{aligned} \tag{11}$$

Clearly, expression (11) provides the open-loop dynamic of the Kalman estimator. If, for some reason, RANSAC does not work properly, the covariance matrix $P(t + \Delta t)$ tends to grow indefinitely. For this reason, the trace of $P(t + \Delta t)$ is checked at the end of each update step. If the value of the trace exceeds a user-defined threshold t_p (e.g. when the total uncertainty is larger than the image size), the state estimated by the Kalman filter is discarded at all and the Kalman filter is reinitialized, as soon as the next image is grabbed. It is worth noticing that the case of incomplete measures (e.g. when just one of the line edges is in the field view of the camera) can be handled by the algorithm. Indeed, the prior information given by the Kalman filter during the prediction step allows RANSAC to detect correctly even a single line edge, while waiting for the other one reappearing in the image. In this case, just the uncertainty associated with d tends to increase monotonically in the state covariance matrix.

4. EXPERIMENTAL RESULTS

4.1. Experimental setup description

The proposed algorithm has been implemented in a BeagleBoard XM embedded platform. This platform is equipped with a 1-GHz ARM DM3730 by Texas Instruments, 512 MB of RAM, a 4 GB micro Secure Digital (SD) memory and a Linux Angstrom distribution. The BeagleBoard XM provides expansion headers to connect other peripherals to the platform. The adopted vision system relies on a custom light contrast imager driven by an XC2C512 Xilinx CoolRunner-II Complex Programmable Logic Device (CPLD). Both board and camera are shown in Figure 3. Since the pins of the Beagleboard expansion headers are rated at 1.8 V while the camera requires 3.3 V, a piggy-back voltage level translator has been used to interface the embedded platform with the camera. The light contrast sensor is a 35-μm CMOS imager with a

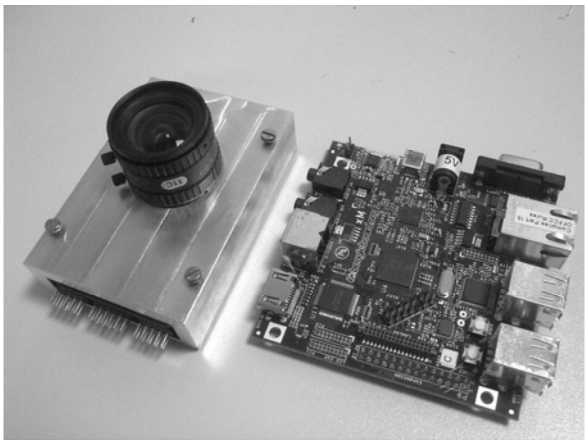


Figure 3. Vision system for light contrast detection (on the left) and BeagleBoard xM embedded platform (on the right).

resolution of 128 × 64 pixels made at the "Fondazione Bruno Kessler" (FBK), Trento, Italy [43]. Note that the low resolution of the camera is helpful for the application considered, as it greatly reduces the number of iterations of the RANSAC algorithm at no cost in terms of accuracy, as it will be shown in Section 4.2. An important feature of the adopted vision system is that it requires a single bit for each pixel. Each pixel generates a voltage value that is proportional to the maximum relative variation of the average light intensity over triples of adjacent pixels during a given integration interval. All output voltage values are quantized by a 1-bit comparator. The resulting bits can be stored into a local on-chip memory or they can be immediately read out and buffered into the CPLD. This feature (along with the low resolution of the imager) makes frame acquisition faster than in regular cameras. As a consequence, the camera frame rate can be larger than 100 frame/s, but with low bandwidth requirements for data transfer. In practice, the frame rate is limited by the light integration time of each pixel. This can be as low as 1 ms, but it should be at least 10 ms not to affect the sensitivity of the imager [44].

The on-board CPLD is provided with a Serial Peripheral Interface (SPI) to read the image frames captured by the camera. The embedded platform communicates with the logic circuitry of the vision system through General Purpose Input-Output (GPIO) and SPI pins. The GPIO pins are used to set the camera operating parameters (e.g. operation modes and integration time) and to exchange interrupts and control signals with the vision system. The SPI pins are instead used to transfer the image frames. The transmission rate is controlled by the SPI clock. In the current implementation the SPI clock frequency is 48 MHz. The CPLD is designed to hold only one image row at a time. Therefore, one SPI read per row is needed to read a full image frame. In order to collect such data as quickly as possible, a highly optimized software driver was developed for Linux.

4.2. Performance evaluation

In order to evaluate the performances of the overall system, several image records of several minutes each have been collected at about 100 frame/s with a pixel integration time of 10 ms. The camera was fastened to the right external rear view mirror of a real vehicle, about 1.2 m above the ground to detect one of the side lines of the road. Such lines are typically white and about 15 cm wide. The results of all experiments reported in the following were conducted on urban roads, in a peripheral district of the city of Trento. The speed of the vehicle was kept between about 30 km/h and 70 km/h (i.e. 50 km/h on average) depending on the road and traffic conditions. It is worth emphasizing that such testing conditions are more challenging than those adopted in robotic vehicle competitions. In fact, during the experiments, line recognition was hindered and occasionally perturbed by shadows, faded or interrupted lines, gravel and road imperfections (e.g. asphalt patches). Usually, the light contrast between the road surface and the lines painted on the street is large enough to enable an easy detection of the line edges. In particular, such edges are depicted as linear clusters of active pixels on a grey background (in the following referred to as *data points*). Some significant examples of collected images are shown in Figures 4(a)-(i). Figure 4(a) refers to ideal conditions, i.e. a freshly painted white line on a uniform dark grey road background. In this case, the line edges are easily recognizable to the naked eye. In Figure 4(b) the right side of the line is partially faded. In Figure 4(c) the whole line is faded. In Figure 4(d) and 4(e) the line edges are sharp, but some shadows due to nearby objects create some additional contrast points (additional dark pixels). In Figure 4(f) one of the line edges is out of the field of view of the camera because the vehicle has slightly departed from the wanted trajectory. In Figure 4(g) the road line is covered by some gravel. In Figure 4(h) the road is both dusty and in shadow. Finally, in Figure 4(i) line recognition is perturbed by a manhole cover. In the examples above, the thin, white straight lines plotted in each picture are reconstructed using the values of $q = [h,\alpha,d]^T$ estimated in real-time. In all experiments, the threshold value s_t for the RANSAC algorithm is a function of the state covariance matrix P, and it is never smaller than 2 pixels.

Estimation accuracy and robustness to visual artefacts have been evaluated offline. To this purpose, the images collected in about 15 tests of several minutes each have been grouped into three data sets depending on the amount of disturbances. In the following, *data set 1* refers to the experiments in which the percentage of images affected by disturbances, such as those shown in Figure 4, is moderate (i.e. between about 15 % and 25 %). *Data set 2* includes the best experiments, namely those where no more than about 15 % of the collected images is perturbed. Finally, *data set 3* comprises the worst experiments, i.e. those where the line was faded, partially interrupted or perturbed in up to 40 % of images.

In all cases, the actual values of the state variables (namely the *ground truth* parameters denoted as $h_{gt}(t)$, $a_{gt}(t)$ and $d_{gt}(t)$) were obtained with the following off-line procedure:

1. at first, the so-called *projection pursuit algorithm* is used to select the coarse-grained histograms with the highest peaks [45]. Each set of features generating the histograms' peaks (one peak for each line) represents a hypothesis;
2. among all possible hypotheses, the one with the highest number of features is chosen;
3. the projection pursuit algorithm is executed again, but with a fine-grained resolution, on the remaining features to detect possible clusters. The cluster with the highest ratio between the number of features and its standard deviation is chosen as the winning hypothesis;
4. the features belonging to such a hypothesis are determined using a voting scheme similar to RANSAC;
5. finally, a Least Squares Quadratic (LSQ) optimization algorithm is used to compute the values of $h_{gt}(t)$, $a_{gt}(t)$ and $d_{gt}(t)$.

The results of this ground truth reconstruction procedure were carefully checked a posteriori by visual inspection record

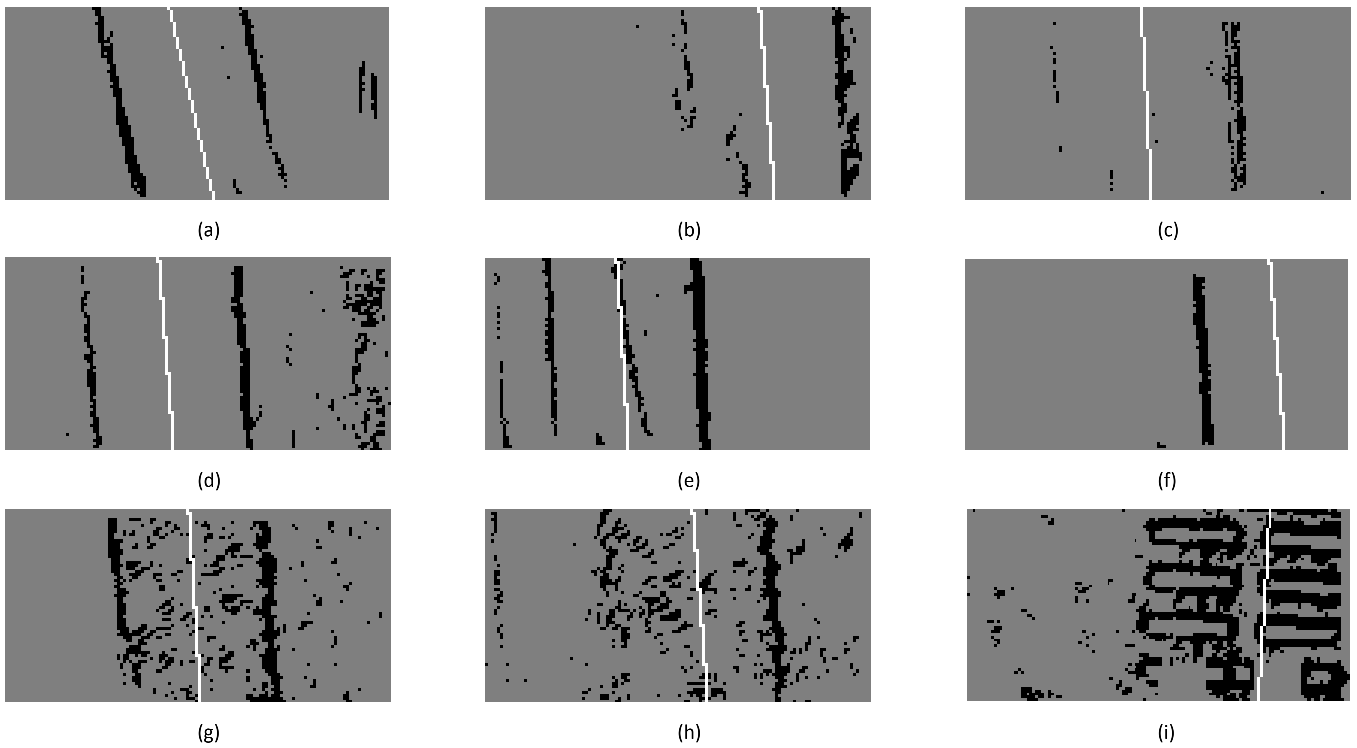


Figure 4. Examples of images collected from the light contrast sensor: (a) ideal situation; (b) partially faded line; (c) considerably faded line; (d)-(e) effect of shadows; (f) right edge out of the field of view of the camera; (g) road partially covered by gravel; (h) dusty shady line; (i) line painted over a manhole cover (i). In each image, the direction of the vehicle estimated by the algorithm with respect to the road line is represented by a white line.

by record to remove or to correct possible outliers. A few images whose ground truth parameters could not be clearly estimated and that could not be corrected manually were simply removed from the data sets.

Figure 5(a)-(c) shows the 0.95-level confidence intervals associated with $\varepsilon_h = h - h_{gt}$ (a), $\varepsilon_a = a - a_{gt}$ (b) and $\varepsilon_d = d - d_{gt}$ (c), for the three data sets described above. In each case the results of two different estimation techniques are shown and compared, i.e. the one described in this paper (which relies on both RANSAC and Kalman filtering) and the solution based on RANSAC only presented in [40].

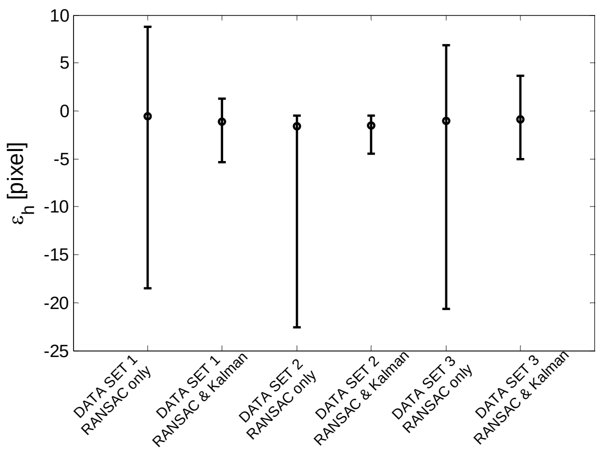
We decided to evaluate measurement uncertainty in terms of confidence intervals rather than reporting the standard uncertainty computed with a *Type-A* evaluation approach [46], because the probability density functions of ε_h, ε_a and ε_d are strongly unimodal, but the normal probability plots of the collected data show that they are not normally distributed. An example of such distributions in the case of variable ε_a for *data set 3* is shown in Figure 6. The histograms of the other variables are quite similar and are not particularly significant; so they are not reported for the sake of brevity.

The use of confidence intervals in Figure 5(a)-(c) emphasizes more clearly the benefits of using the Kalman filter to improve accuracy, precision and robustness in line detection. This is especially true as far as the *a* parameter is concerned, which is also the most important for vehicle control purposes. Observe that ε_a ranges approximately between [-1,+1] degree with 95 % probability in all cases. Of course, the results related to *data set 2* are the best ones. However, even in the presence of more frequent disturbances, the proposed estimation technique is quite robust. This improvement is due to two reasons. First of all, the Kalman filter decreases the probability that RANSAC detects wrong line edges in consecutive images, thus greatly reducing the probability of unnatural "jumps" in the estimation process. Secondly, the Kalman filter makes the algorithm track angle *a* even when one of the two line edges is occasionally lost (e.g. because the line is too faded or because it is out of the field of view of the camera, as it is shown in Figure 4(f)).

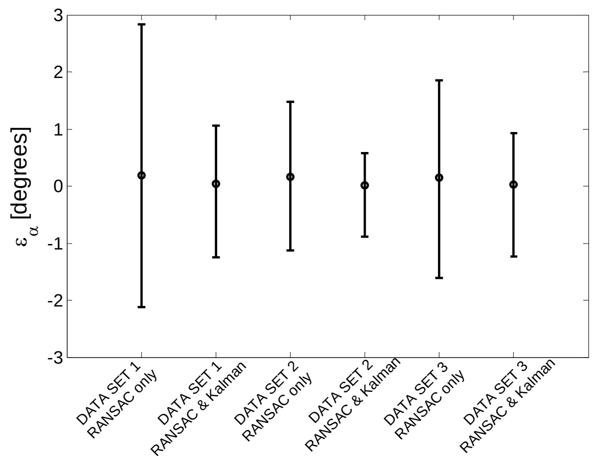
The analysis of ε_d and ε_h is a bit more complex. First of all, the distribution of such estimation errors in the RANSAC-only case is asymmetric. This asymmetry is due to the fact that when one of the two line edges is out of view, the "orphan" one is not always detected on the correct side of the line. Therefore, the right edge could be recognized as the left one and vice versa. However, the probability of these events is not the same. Thus, the estimation errors exhibit an asymmetric distribution, which, in the RANSAC-only case, can sometimes exceed 15 pixels. Given that in the current setup 1 pixel (namely with the vision system about 1.2 m above the ground) corresponds to about 3 cm, the position error can be larger than 45 cm.

By using the proposed algorithm, symmetry is usually re-established because the Kalman filter keeps memory of the previous positions of the line edges. In this way, the estimation errors are considerably reduced. Indeed, they are generally within ± 5 pixels (i.e. about 15 cm), with the only exception of ε_d in *data set 3*. This is due to the fact that, when the line is heavily faded, its width can be hardly estimated with good accuracy. Nonetheless, the accuracy in estimating vehicle direction is still preserved. It is important to remind that the memory effect of the Kalman filter is lost in the case of filter re-initialization, which occurs when the trace of matrix *P* exceeds the threshold t_p defined in Section 3.4.

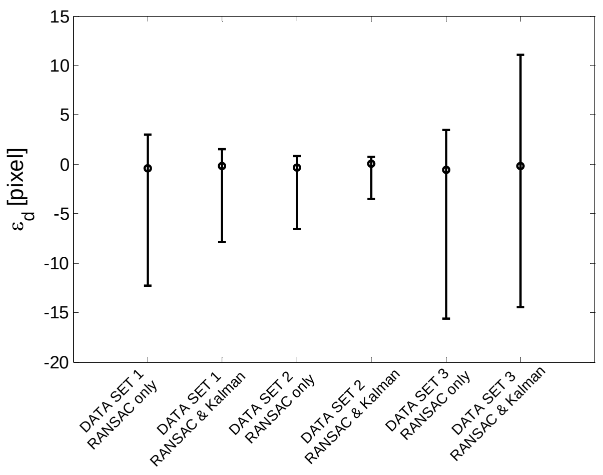
A further important aspect that has to be carefully evaluated to appreciate the performances of the system prototype is related to computation time. Figure 7 shows the cumulative distribution curves of the execution times of the estimation algorithm running on the chosen low-cost embedded platform. The dashed line refers to the algorithm based on RANSAC



(a)



(b)



(c)

Figure 5. 0.95-level confidence intervals associated to ε_h (a) and ε_α (b) and ε_d (c) for three different data sets. Pairs of adjacent intervals refer to the estimation errors resulting from the proposed RANSAC algorithm with and without Kalman filtering, respectively.

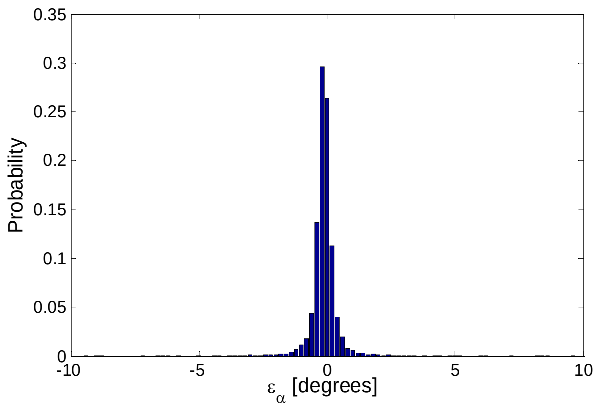


Figure 6. Histogram of ε_α obtained when the proposed algorithm is applied to the images of *data set 3*.

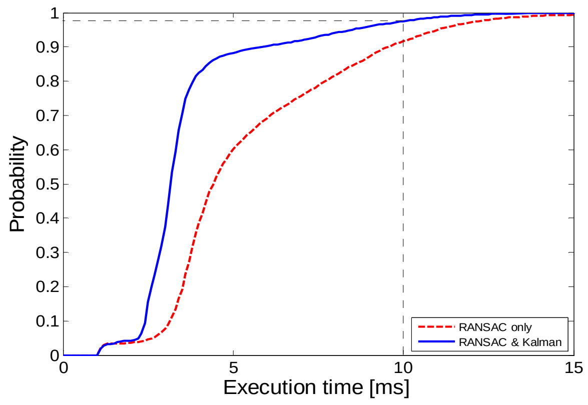


Figure 7. Execution time cumulative distribution curves of the RANSAC-only (dashed line) and RANSAC-with-Kalman-filter (solid line) algorithms running on the BeagleBoard XM platform.

only, whereas the solid line corresponds to the proposed solution based on both RANSAC and Kalman filtering. Observe that the execution time of the RANSAC-with-Kalman-filter approach is faster than the solution based on RANSAC only. This is due to the fact that generally a smaller number of iterations is needed to detect the line edges due to the prior retained by the Kalman filter. In particular, in this case the median execution time of the proposed algorithm is just 4 ms and it is lower than 10 ms with 97 % probability. This means that with a camera frame rate equal to 100 frame/s (so that a new image can be processed every 10 ms), even if the vehicle travels at 100 km/h, the measurement system is able to estimate direction changes with a space resolution of about 30 cm, which is compatible with automated high-speed vehicle control.

5. CONCLUSIONS

Increasingly sophisticated, high-speed automated guided vehicles (AGVs) require fast sensors able to ensure accurate and reliable real-time path-following techniques. In this context, this paper presents a simple vision-based technique and a low-cost system able to estimate the direction and the relative position of a vehicle with respect to a line painted on the ground. Line-tracking techniques are indeed commonly used in robots for industrial applications. The proposed solution relies on a special camera and on a tailored RANSAC algorithm enhanced by a Kalman filter. Even if the system does not address other crucial safety problems, such as collision avoidance for instance, it is robust to manifold uncertainty sources and it is cheaper and faster than other vision-based solutions. This is due not only to the proposed algorithm per se, but also to the low resolution of the adopted light contrast imager, which greatly reduces image processing burden and data transfer latency. In future, the current prototype could be further improved by using a more sensitive imager and an inertial platform measuring vehicle speed.

APPENDIX - DERIVATION OF EXPRESSION (2)

Let $\langle W \rangle$ be a reference frame, with axes X_c, Y_c and Z_c defined in such a way that:

- X_c is directed as the *x*-axis of the image plane;
- Y_c is oriented as the *y*-axis of the image plane (according to a right-handed reference frame);
- Z_c is oriented towards the observed scene till intersecting the image plane in the *principal point*, as shown in Figure 1.

The origin of $\langle C \rangle$ coincides with the camera pin-hole. Therefore, $\langle C \rangle$ is simply translated with respect to $\langle I \rangle$ without any rotation. On the contrary, the orientation of $\langle C \rangle$ with respect to $\langle W \rangle$ can be arbitrary. Therefore, if ${}^{c}R_{w}$ is the rotation matrix of $\langle C \rangle$ and $t_w = [t_x, t_y, t_z]^T$ is the same *translation vector*, as defined in Section 2.1, the coordinates of a generic point ${}^{w}P$ in the camera frame are given by [41]:

$$ {}^{c}P = \left[{}^{c}R_{w} | t_{w} \right] \left[{}^{w}P^{T}, 1 \right]^{T} \tag{A.1}$$

where ${}^{c}P = [{}^{c}x, {}^{c}y, {}^{c}z]^T$ and $[{}^{c}R_{w} | t_{w}]$ is a 3 x 4 matrix which defines the rigid transformation between $\langle C \rangle$ and $\langle W \rangle$. In particular, if the axes of $\langle C \rangle$ are completely rotated with respect to $\langle W \rangle$ as shown in Figure 1, (A.1) can be rewritten as

$$ {}^{c}P = \begin{bmatrix} -{}^{w}x + t_x \\ -{}^{w}y + t_y \\ -{}^{w}z + t_z \end{bmatrix} \tag{A.2}$$

From the basic theory of vision systems, it is known that the projection of any point in the field of view of the camera onto the image plane results from [41]:

$$ \begin{bmatrix} {}^{c}z\,{}^{i}x \\ {}^{c}z\,{}^{i}y \\ {}^{c}z \end{bmatrix} = G \begin{bmatrix} {}^{c}x \\ {}^{c}y \\ {}^{c}z \end{bmatrix} \tag{A.3}$$

where

$$ G = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.4}$$

is the so-called *camera calibration matrix*, f_x and f_y represent the focal lengths along axes X_i and Y_i, respectively, s models the radial distortion of the image, and the coordinates c_x and c_y of the principal point of the camera are both equal to 0 in the frame $\langle I \rangle$. In practice, the terms of the calibration matrix can be estimated using widely known numerical tools [47]. Thus, if the radial distortion coefficient is negligible (as it commonly occurs in practice), by replacing (A.2) into (A.3), after a few mathematical steps, (2) finally results.

ACKNOWLEDGEMENTS

The authors would like to thank Marco De Nicola, Dr. Massimo Gottardi and Dr. Leonardo Gasparini from FBK, Trento, for their advice and support in configuring the experimental light contrast imager used for the system prototype described in this paper.

REFERENCES

[1] S.S. Carlisle, "The role of measurement in the development of industrial automation," ACTA IMEKO 3 (2013), pp. 4-9.

[2] T.H. Lee, H.K. Lam, F.H.F. Leung, P.K.S. Tam, "A fast path planning-and-tracking control for wheeled mobile robots," Proc. IEEE International Conference on Robotics and Automation, (ICRA), May 22-26, 2001, Seoul, Korea, pp. 1736-1741.

[3] C. Samson and K. Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in Cartesian space," in Proc. IEEE International Conference on Robotics and Automation, Apr. 9-11, 1991, Sacramento, CA, USA, pp. 1136–1141.

[4] A. Aguiar and J. Hespanha, "Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty," IEEE Transactions on Automation and Control 52 (2007), pp. 1362–1379.

[5] R. D'Souza, "Designing an autonomous robot vehicle," IEEE Potentials 17 (1998), pp. 40-43.

[6] Xiuzhi Li, Songmin Jia, Jinhui Fan, Liwen Gao, Bing Guo, "Autonomous mobile robot guidance based on ground line mark," Proc. of SICE Annual Conference (SICE), Akita, Japan, 20-23 Aug. 2012, pp. 1091-1095.

[7] Yangsheng Xu, S. Kwok-Way Au, "Stabilization and path following of a single wheel robot," IEEE/ASME Transactions on Mechatronics 9 (2004), pp. 407-419.

[8] L. Xiafu, C. Yong, "A design of autonomous tracing in intelligent vehicle based on infrared photoelectric sensor," Proc. International Conference on Information Engineering & Computer Science (ICIES), Dec. 19-20, 2009, Wuhan, China, pp. 1-4.

[9] N.M. Arshad, M.F. Misnan, N.A. Razak, "Single infra-red sensor technique for line-tracking autonomous mobile vehicle," Proc. IEEE 7th International Colloquium on Signal Processing and its Applications (CSPA), Mar. 4-6, 2011, Penang, Malaysia, pp.159-162.

[10] Jeong-Yean Yang; Dong-Soo Kwon, "Electric inductance sensor-based path recognition for the highly configurable path tracking of service robot," Proc. 17th IEEE International Symposium on Robot and Human Interactive Communication (ROMAN), Aug. 1-3, 2008, Munich, Germany, pp. 419-424.

[11] A. Bonarini, P. Aliverti, M. Lucioni, "An omnidirectional vision sensor for fast tracking for mobile robots," IEEE Transactions on Instrumentation and Measurement 49 (2000), pp. 509-512.

[12] L. Armesto, J. Tornero, "Automation of industrial vehicles: A vision-based line tracking application," in Proc. IEEE Conference on Emerging Technologies & Factory Automation (ETFA), Sep. 22-25, 2009, Mallorca, Spain, pp. 1-7.

[13] N.M. Arshad, N.A. Razak, "Vision-based detection technique for effective line-tracking autonomous vehicle," Proc. IEEE 8th International Colloquium on Signal Processing and its Applications (CSPA), Mar. 23-25, 2012, Melaka, Malaysia, pp. 441-445.

[14] H. Gross, H.J. Boehme, T. Wilhelm, "Contribution to vision-based localization, tracking and navigation methods for an interactive mobile service-robot," Proc. IEEE International Conference on Systems, Man, and Cybernetics, Oct. 7-10, 2001, Tucson, AZ, USA, pp. 672-677.

[15] G. Antonelli, S. Chiaverini, G. Fusco, "A Fuzzy-Logic-Based Approach for Mobile Robot Path Tracking," IEEE Transactions on Fuzzy Systems 15 (2007), pp. 211-221.

[16] F. Jimenez, "Improvements in road geometry measurement using inertial measurement systems in datalog vehicles," Measurement, 44 (2011), pp. 102-112.

[17] R. Satzoda, S. Sathyanarayana, T. Srikanthan, and S. Sathyanarayana, "Hierarchical additive Hough transform for lane

detection," IEEE Embedded Systems Letters 2, (2010), pp. 23–26.

[18] A. Troiano, E. Pasero, and L. Mesin, "New system for detecting road ice formation," IEEE Transactions on Instrumentation and Measurement 60 (2011), pp. 1091-1101.

[19] F. Espinosa, "Design and implementation of a portable electronic system for vehicle-driver-route activity measurement," Measurement 44 (2011), pp. 326-337.

[20] M. Buehler, K. Iagnemma, and S. Singh, The 2005 DARPA Grand Challenge - The Great Robot Race (Springer Tracts in Advanced Robotics), vol. 36, Springer-Verlag, Berlin/Heidelberg, Germany, 2007.

[21] D. Fontanelli, L. Palopoli, T. Rizano, "High speed robotics with low cost hardware," Proc. IEEE 17th Conference on Emerging Technologies & Factory Automation (ETFA), Sep. 2012, 17-21, Kralow, Poland, pp. 1-8.

[22] G. Vanderbrug, "Line detection in satellite imagery," IEEE Transactions on Geoscience Electronics 14 (1976), pp. 37–44.

[23] D. Guru, B. Shekar, P. Nagabhushan, "A simple and robust line detection algorithm based on small eigenvalue analysis," Pattern Recognition Letters 25 (2004), pp. 1–13.

[24] N. Aggarwal, W. Karl, "Line detection in images through regularized Hough transform," IEEE Transactions on Image Processing 15 (2006), pp. 582–591.

[25] L. Xu, E. Oja, "Randomized Hough transform (RHT): basic mechanisms, algorithms, and computational complexities," CVGIP: Image Understanding 57 (1993), pp. 131–131.

[26] R. Satzoda, S. Sathyanarayana, T. Srikanthan, S. Sathyanarayana, "Hierarchical additive hough transform for lane detection, IEEE Embedded Systems Letters 2 (2010), pp. 23–26.

[27] A.H. Ismail, H.R. Ramli, M.H. Ahmad, M.H. Marhaban, "Vision-based system for line following mobile robot," Proc. IEEE Symposium on Industrial Electronics & Applications (ISIEA), Oct. 4-6 2009, Kuala Lumpur, Malaysia, pp. 642-645.

[28] A. Chatterjee, A. Rakshit, N. Nirmal Singh, "Vision Based Mobile Robot Path/Line Tracking," in Vision Based Autonomous Robot Navigation, Springer, Berlin-Heidelberg, Germany, 2013, ISBN: 978-3-642-33964-6, pp. 143-166.

[29] P. Mazurek, "Line estimation using the Viterbi algorithm and track-before-detect approach for line following mobile robots," Proc. 19th International Conference on Methods and Models in Automation and Robotics (MMAR), Sep. 2-5, 2014, Miedzyzdroje, Poland, pp. 788-793.

[30] C. Guo, S. Mita, D. McAllester, "Lane detection and tracking in challenging environments based on a weighted graph and integrated cues," Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct. 18-22, 2010, Taiwan, pp. 5543 - 5550.

[31] A. López, J. Serrat, C. Canero, F. Lumbreras, "Robust lane lines detection and quantitative assessment," in Pattern Recognition and Image Analysis. Springer-Verlag, New York, NY, 2007, ISBN: 978-3-540-72846-7, pp. 274–281.

[32] B.-F. Wu, C.-T. Lin, and Y.-L. Chen, "Dynamic calibration and occlusion handling algorithms for lane tracking," IEEE Transactions on Industrial Electronics 56 (2009), pp. 1757–1773.

[33] Y. Wang, N. Dahnoun, A. Achim, "A novel system for robust lane detection and tracking," Signal Processing (2012), pp. 319-334.

[34] Z. Kim, "Robust lane detection and tracking in challenging scenarios," IEEE Transactions on Intelligent Transportation Systems 9 (2008), pp. 16-26.

[35] M. Bertozzi, A. Broggi, "Gold: A parallel real-time stereo vision system for generic obstacle and lane detection," IEEE Transactions on Image Processing 7 (1998), pp. 62–81.

[36] T. Rizano, D. Fontanelli, L. Palopoli, L. Pallottino, P. Salaris, "Global path planning for competitive robotic cars," Proc. IEEE 52nd Annual Conference on Decision and Control (CDC), Dec. 10-13, 2013, Florence, Italy, pp. 4510-4516.

[37] M. Fischler, R. Bolles, "RANdom SAmpling Consensus: a paradigm for model fitting with application to image analysis and automated cartography," Communications of the ACM 24 (1981) pp. 381–395.

[38] D. Fontanelli, L. Ricciato, S. Soatto, "A fast RANSAC–based registration algorithm for accurate localization in unknown environments using LIDAR measurements," Proc. IEEE International Conference on Automation Science and Engineering, Sep. 22-25, 2007, Scottsdale, AZ, USA, pp. 597–602.

[39] G. Mastorakis, E. Davies, "Improved line detection algorithm for locating road lane markings," Electronics Letters 47 (2011), pp. 183–184.

[40] D. Fontanelli, M. Cappelletti, D. Macii, "A RANSAC-based fast road line detection system for high-speed wheeled vehicles," Proc. of IEEE Int. Instrumentation and Measurement Technology Conference (I2MTC), May 10-12, 2011, Hang Zhou, China, pp. 186–191.

[41] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2003, ISBN: 0-52-154-051-8.

[42] H. Mallot, H. Blthoff, J. Little, and S. Bohrer, "Inverse perspective mapping simplifies optical flow computation and obstacle detection," Biological Cybernetics 64 (1991), pp. 177–185.

[43] M. Gottardi, N. Massari, S. Jawed, "A 100-μW 128×64 pixels contrastbased asynchronous binary vision sensor for sensor networks applications," IEEE Journal of Solid-State Circuits 44 (2009), pp. 1765–1770.

[44] L. Gasparini, D. Macii, M. Gottardi, D. Fontanelli, "A low-power data acquisition system for image contrast detection," in Proc. IMEKO TC-4 International Workshop on ADC Modelling and Testing (IWADC), Jun. 30- Jul. 1, 2011, Orvieto, Italy, pp. 1-6.

[45] P. Huber, "Projection pursuit," The annals of Statistics 13 (1985) 435–475.

[46] BIPM, and IEC, and IFC, and ISO, and IUPAC, OIML, Guide to the Expression of Uncertainty in Measurement, Geneva, Switzerland, 2008.

[47] Z. Zhang, "A flexible new technique for camera calibration," IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2002), pp. 1330–1334.