Optics and Lasers in Engineering 54 (2014) 159–169

Contents lists available at ScienceDirect

Optics and Lasers in Engineering

journal homepage: www.elsevier.com/locate/optlaseng

# Optical 3D laser measurement system for navigation of autonomous mobile robot

Luis C. Básaca-Preciado [a], Oleg Yu. Sergiyenko [a,*], Julio C. Rodríguez-Quinonez [a], Xochitl García [a], Vera V. Tyrsa [d], Moises Rivas-Lopez [a], Daniel Hernandez-Balbuena [b], Paolo Mercorelli [c], Mikhail Podrygalo [d], Alexander Gurko [d], Irina Tabakova [e], Oleg Starostenko [f]

[a] Optoelectronics and Automated Measurement Laboratory, Engineering Institute of the Autonomous University of Baja California (UABC), Mexicali, B.C., Mexico
[b] Engineering Faculty of the Autonomous University of Baja California (UABC), Mexicali, B.C., Mexico
[c] Institute of Product and Process Innovation, Leuphana University of Lueneburg, Germany
[d] Kharkov National Highway and Automobile University, Ukraine
[e] Kharkov National University of Radioelectronics, Kharkov, Ukraine
[f] Kharkov Universidad de Las Americas, Puebla, Mexico

ARTICLE INFO

*Article history:*
Received 9 February 2013
Received in revised form
5 August 2013
Accepted 7 August 2013
Available online 8 September 2013

*Keywords:*
Autonomous navigation
Vision system
Mobile robot
3D laser scanner
Trajectory planning

ABSTRACT

In our current research, we are developing a practical autonomous mobile robot navigation system which is capable of performing obstacle avoiding task on an unknown environment. Therefore, in this paper, we propose a robot navigation system which works using a high accuracy localization scheme by dynamic triangulation. Our two main ideas are (1) integration of two principal systems, 3D laser scanning technical vision system (TVS) and mobile robot (MR) navigation system. (2) Novel MR navigation scheme, which allows benefiting from all advantages of precise triangulation localization of the obstacles, mostly over known camera oriented vision systems. For practical use, mobile robots are required to continue their tasks with safety and high accuracy on temporary occlusion condition. Presented in this work, prototype II of TVS is significantly improved over prototype I of our previous publications in the aspects of laser rays alignment, parasitic torque decrease and friction reduction of moving parts. The kinematic model of the MR used in this work is designed considering the optimal data acquisition from the TVS with the main goal of obtaining in real time, the necessary values for the kinematic model of the MR immediately during the calculation of obstacles based on the TVS data.



## 1. Introduction

Autonomous robot navigation and collision prevention tasks are becoming very important not only in the automotive industry but also in scientific research such as planet exploration e.g., the Mars Rover by NASA [1], exploration of unknown or dangerous terrain such as DEPTHX project by Carnegie Mellon University in Mexico [2], autonomous vehicles like Stanley in the DARPA Grand Challenge by Stanford University [3], among others.

Optical methods seem to be a very attractive option for such solution. However, its application is still of great demand due to its main weak point, which supposes that most of optical methods are for probabilistic estimation, not for measurement of real obstacle spatial position. Or those methods that can measure, they are still very sensitive to vibrations or any other kinds of mechanical noise. This paper is aiming to propose a robust and precise way to obtain a digitized map of robot's surrounding with metrological quality by means of optical measurements.

Autonomous vehicles such as Stanley from Stanford University (see Fig. 1) and most of the vehicles that have participated in the DARPA Grand or Urban Challenge [4] like Boss by Carnegie Mellon University [5], Odin by Virginia Tech [6], Talos from MIT [7], among others, use a combination or fusion of sensors to acquire the necessary information to detect obstacles and roads, required for safe traveling such as Light Detection and Raging (LIDAR), RADAR, Stereo Vision, GPS, Inertial navigation system. This is very important and a great advantage in comparison with autonomous

* Corresponding author. Tel.: +52 6861687170.
*E-mail addresses:* luis.basaca@uabc.edu.mx (L.C. Básaca-Preciado), srgnk@iing.mxl.uabc.mx, srgnk@mail.ru (O.Yu. Sergiyenko), julio.rodriguez37@uabc.edu.mx (J.C. Rodríguez-Quinonez), xomagac@hotmail.com (X. García), vera-tyrsa@mail.ru (V.V. Tyrsa), mrivas@uabc.edu.mx (M. Rivas-Lopez), mercorelli@uni.leuphana.de (P. Mercorelli), pmikh@rambler.ru (M. Podrygalo), agrk@mail.ru (A. Gurko).

0143-8166/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.optlaseng.2013.08.005


**Fig. 1.** Stanley—the Stanford Autonomous Vehicle that won the DARPA Grand Challenge.

vehicles using a single sensor to detect obstacles. However, the more sensors are included the higher the cost of the autonomous vehicle, this can be considered a great disadvantage if the available processing resources are not enough for full sensor fusion.

Nonetheless, the evident high cost of such systems is not its worst disadvantage; it still undermines another one. Such a versatility of the devices in use remains due to an uncertain situation in regards to which one of the multiple sensory system provides in a present moment the most proper information about the surrounding of the mobile robot.

In other words, such complex multisensory system always provides redundant information about the current situation on the robot's trajectory, and the most important task in this case is to properly filter this information and to correctly detect which one of the multisensory components is the most correct (less error) in a current moment. Such task is the main problem in dead reckoning and it consumes a huge capacity of computational resource.

Accordingly, as it is shown in all the mentioned publications about the autonomous vehicles that participated in the DARPA challenges [4], mobile robots equipped with such complex multisensory system still cannot achieve sufficient velocity, which is caused, first of all, by the necessity to solve in real time the difficult task of optimal data search.

Most of the relevant research, present autonomous robots using 2D laser range finders and servo motors for rotation [8,9]. Others propose stereo vision systems that utilizes two cameras to obtain two images of the same object from different points of view and reconstruct a 3D scene, such approach does not give true Cartesian coordinates of the object or obstacle, it just permits an evaluation of the shape and size of the obstacle, based on complex probabilistic algorithms of image processing [10–14]. Finally, it is very time consuming for a processor, it lacks precision, present solutions does not yet allow for a full 3D-reconstruction of the scene and it does not guarantee for errors or image misunderstanding.

Systems that use structured light methods for 3D shape measurement such as [37] present great advantages over stereo vision systems e.g., faster measurement performance and simpler processing algorithms but they also present a few disadvantages such as higher cost equipment, greater dimensions of the overall 3D system and the need of being connected to an AC power supply; these characteristics may limit this system to static applications as the author states in the conclusions of [37], such as computer graphics, medical diagnostics, plastic surgery, industrial inspection and reverse engineering instead of mobile applications such as robot navigation, mine exploration, rescue operations, among others. Although it is worth mentioning that this type of system may be modified to be applied in mobile applications but with some trade-offs.

Therefore, in this paper we present an autonomous mobile robot navigation system that utilizes a low cost high precision vision system with sufficient reliability of information about the surrounding of the mobile robot to allow the mobile robot's navigation system to make decisions regarding the path or trajectory it should follow to achieve its goal.

Our navigation system consists of a new method of motion planning and the mathematical model of a four wheel, four motor skid-steer mobile robot used to successfully navigate without collisions in an (indoor and outdoor) environment with obstacles.

This navigation system obtains the surrounding environment information by means of a 3D laser scanning vision system which is briefly explained in Section 2. The mechanical characteristics and specifications of the mobile robot are explained in Section 3. The mobile robot kinematics model is found in Section 4. In Section 5 our navigation system is presented. Section 6 presents the simulation results of our work with the TVS and the navigation system; followed by the statistical analysis of the data obtained through experimentation in Section 7; and at the end, the conclusions of this work are presented.

## 2. Technical vision system (TVS)

As mentioned before, the presented navigation system will obtain the environment information from our TVS. This system has been partially presented before in [15,16,35]; therefore the explanation will only contain the most important aspects of the TVS.

### 2.1. Dynamic triangulation method

The TVS is based on a method we developed which is called dynamic triangulation, it is called dynamic due to the rotation ability of the positioning laser and the scanning aperture, allowing us to have moving angles that can form laser light triangles with different shapes for a very short period of time and when a triangle is formed thanks to the reflected light of an obstacle surface, we obtain all the necessary information to calculate the 3D coordinates.

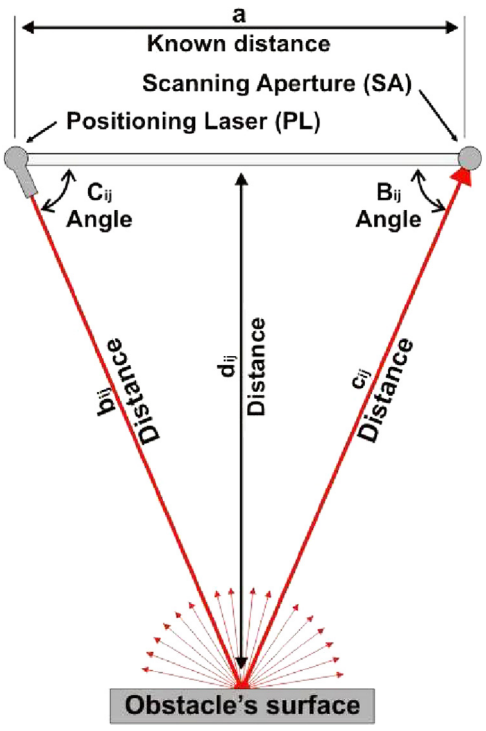


**Fig. 2.** Dynamic triangulation method.

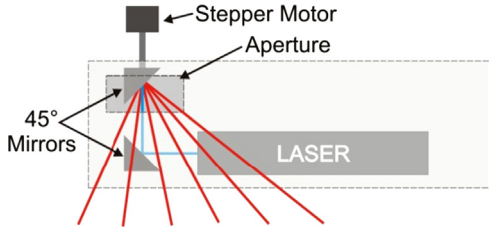


**Fig. 3.** Positioning laser principle.

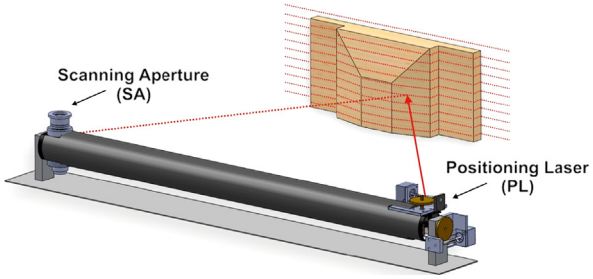


**Fig. 4.** TVS with tilt mechanism 3D model.

Fig. 2 shows the dynamic triangulation method used by the TVS. The main components of the TVS are the positioning laser (PL) and the scanning aperture (SA).

### 2.2. Positioning laser (PL)

The PL contains the next main elements which interact in the following way: a laser that emits its beam toward a fixed 45° mirror; this mirror redirects the beam orthogonally into a rotating 45° mirror which rotation is driven by a stepper motor. By controlling this motor we control the direction at which we want to direct the laser to verify the presence of obstacles; PL is driven by a stepper motor to control the laser direction. See Fig. 3.

The SA principle is more complex than the PL and it will be explained later, however the purpose of the SA is to receive the reflected laser rays indicating that an obstacle has been detected and to calculate the $B_{ij}$ angle which is needed for the 3D coordinates calculation.

The method shown in Fig. 2 functions for 2D coordinates measurement $x$ and $y$. However, we desire to measure 3D coordinates and in order to do that, we added to the TVS a tilt mechanism (see Fig. 4) which allows us to rotate the device in order to perform additional scans at different heights along $z$ axis, thus adding the third dimension coordinate $z$.

### 2.3. Scanning aperture (SA)

The SA is the most important component of the TVS (see Fig. 5) and is in charge of receiving the reflected laser rays through the aperture into a 45° rotating mirror which is rotated with constant speed by a DC motor. Then the captured laser rays are redirected orthogonally to a lens array to finally be received by the stop sensor (high speed photo-transistor) and when is activated we know an obstacle has been detected. The Zero sensor monitors the motor's position so we can know the precise time the mirror rotates 360°; this information will be used later to calculate the $B_{ij}$ angle.

The following equation is used to calculate angle $B_{ij}$:

$$B_{ij} = \frac{2\pi N_A}{N_{2\pi}} \tag{1}$$

where $N_A$ is the number of reference pulses when laser rays are detected by the stop sensor and $N_{2\pi}$ is the number of reference pulses when the 45° mirror completes a 360° turn detected by the zero sensor. See Fig. 5. For a more detailed explanation and figures see [16,18].

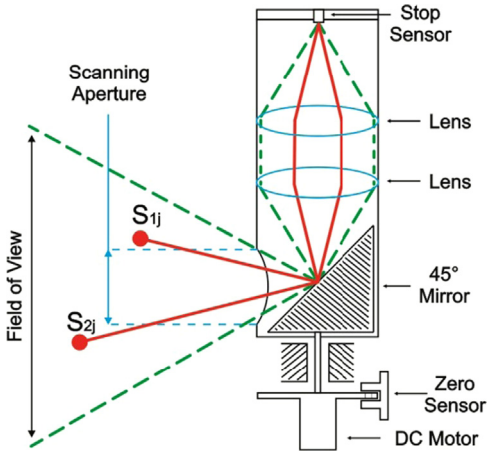


**Fig. 5.** Scanning aperture (SA).

### 2.4. 3D coordinates calculation

In order to calculate the $x$, $y$ and $z$ coordinates of any reflected point from an obstacle surface, we developed a set of equations derived from the law of sines (Eqs. (2)–(5)). These equations require the angles $B_{ij}$, $C_{ij}$, $\Sigma_{\beta j}$ and a (fixed distance between the center of PL and the center of SA) to calculate the 3D coordinates [15].

$$x_{ij} = a\frac{\sin B_{ij} \cdot \sin C_{ij} \cdot \cos \sum_{j=1}^{j} \beta_j}{\sin[180^\circ - (B_{ij} + C_{ij})]}, \tag{2}$$

$$y_{ij} = a\left(\frac{1}{2} - \frac{\sin B_{ij} \cdot \cos C_{ij}}{\sin[180^\circ - (B_{ij} + C_{ij})]}\right) \quad \text{at} \quad B_{ij} \le 90^\circ, \tag{3}$$

$$y_{ij} = -a\left(\frac{1}{2} + \frac{\sin B_{ij} \cdot \cos C_{ij}}{\sin[180^\circ - (B_{ij} + C_{ij})]}\right) \quad \text{at} \quad B_{ij} \ge 90^\circ, \tag{4}$$

$$z_{ij} = a \cdot \frac{\sin B_{ij} \cdot \sin C_{ij} \cdot \sin \sum_{j=1}^{j} \beta_j}{\sin[180^\circ - (B_{ij} + C_{ij})]} \tag{5}$$

During the TVS operation, the $x$, $y$, $z$ coordinates are calculated and stored in a multidimensional $20 \times 20 \times 4$ matrix. The first dimension contains a digital map of the field of view of the mobile robot; each matrix $ij$ position can have a value of 0 or 1. A 0 means no obstacle is found and a 1 means reflection was detected and an obstacle is found at that point.

The second dimension contains the $B_{ij}$ angle information, for each $ij$ position of the 1st dimension; a corresponding and calculated $B_{ij}$ angle is stored at the same position on the 2nd dimension. This also occurs for $C_{ij}$ angle on the 3rd dimension and the $\Sigma_{\beta j}$ on the 4th dimension. This information is then sent to the mobile robot navigation system to analyze and take navigation decisions e.g., obstacle avoidance and trajectory planning.

Experimental results have been obtained from TVS prototype I and some are published on [18–21]. However, we can assure that the TVS prototype I has a 95% confidence level on its coordinates measurements [15]. A prototype II has been fabricated with several mechanical improvements over prototype I [16,17], new experimental results are presented in Section 7.

## 3. Mobile robot specifications

As shown in our previous publications in this scope, until now we have been considering the proposed TVS as a pure theoretical task to resolve. However, on the present stage of research it is desirable connect our TVS with a certain industrially fabricated mobile robot in order to fully integrate them to each other by their working parameters (weight, size, operation speed, mobility skills, etc.)

Several known robot models were considered preliminary but we decided to base our work in the Pioneer 3-AT (Fig. 6) which is a small, four-wheel, four-motor skid-steer intelligent robotics platform, which in our opinion is well matched with our offered TVS. The Pioneer 3-AT can fulfill our needs of a mobile robot platform to implement our navigation and 3D vision systems due to its powerful microcontroller (44.2368 MHz Renesas SH2 32-bit RISC μC with 32KRAM and 128K FLASH; ARCOS firmware), the four wide pneumatic tires which in conjunction with the four DC motors equipped with encoders (135,465 ticks per revolution), provide high traversability and maneuverability, three long duration batteries (12 VDC at 7.2 Ah each) that last 4–6 continuous hours and the option to have an onboard laptop computer.

All of these characteristics make the Pioneer 3-AT a very suitable research development platform to implement our navigation and 3D vision system for autonomous mobile robots. On the next section the kinematics analysis for the mobile robot is presented.

## 4. Mobile robot kinematics

This section presents a kinematic analysis and the necessary transforms to control our mobile robot in a three dimensional space.

**Fig. 6.** Pioneer 3-AT mobile robotic platform.

The axis convention used for the frame assignment is that the $z$ axis points up (aligned with local gravity), the $y$ axis points to the right of the mobile robot (east) and the $x$ axis points forward (north).

The frame assignment is shown in Fig. 7a–c. Frame 0 is the reference frame for navigation, the frame r stands for robot frame and is located at the center of gravity of the mobile robot, w1–w4 is for wheels 1 to 4 (these frames are located in the rotation axis at the center of the wheels and they rotate with the wheels), frames e and s are located in the vision system (mounted on a tilt mechanism that rotates around the $x$ axis) and they belong to the emitter and sensor respectively.

In order to represent the position and orientation in space we need six pieces of information, three for position and three for orientation. For position we use a homogeneous transform matrix to translate the mobile robot to the desired position and then we use RPY angles (roll, pitch, yaw) to adjust the robot to the desired orientation, RPY angles are a sequence of three rotations about the current frame $\vec{a}$, $\vec{o}$, $\vec{n}$ moving axes (approach, orientation and normal) respectively, i.e., each rotation about $\vec{a}$, $\vec{o}$, $\vec{n}$ axes is represented with a $4\times4$ matrix and the multiplication of these three matrices describes the desired orientation of an object in space using RPY angles, this is shown in Eq. (8). The $\vec{a}$, $\vec{o}$, $\vec{n}$ axes (from the current frame) are parallel to $x$, $y$, $z$ (reference frame) at the origin position (before any movement) [22]. The sequence of transformations stated on Eq. (7), defines the forward kinematics model for this robot which represents the current position and orientation of the robot relative to a fixed reference frame.

The following equations were defined to control the position and orientation of the mobile robot in a tridimensional space relative to a fixed reference frame:

$$ {}^{0}T_{r} = Trans(x,y,z)\times RPY(\varphi,\theta,\psi) \tag{6} $$

$$ {}^{0}T_{r} = Trans(x,y,z)\times Rot(\vec{a},\varphi)\times Rot(\vec{o},\theta)\times Rot(\vec{n},\psi) \tag{7} $$

$$ {}^{0}T_{r} = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} c\varphi & -s\varphi & 0 & 0 \\ s\varphi & c\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} c\theta & 0 & s\theta & 0 \\ 0 & 1 & 0 & 0 \\ -s\theta & 0 & c\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\psi & -s\psi & 0 \\ 0 & s\psi & c\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8} $$

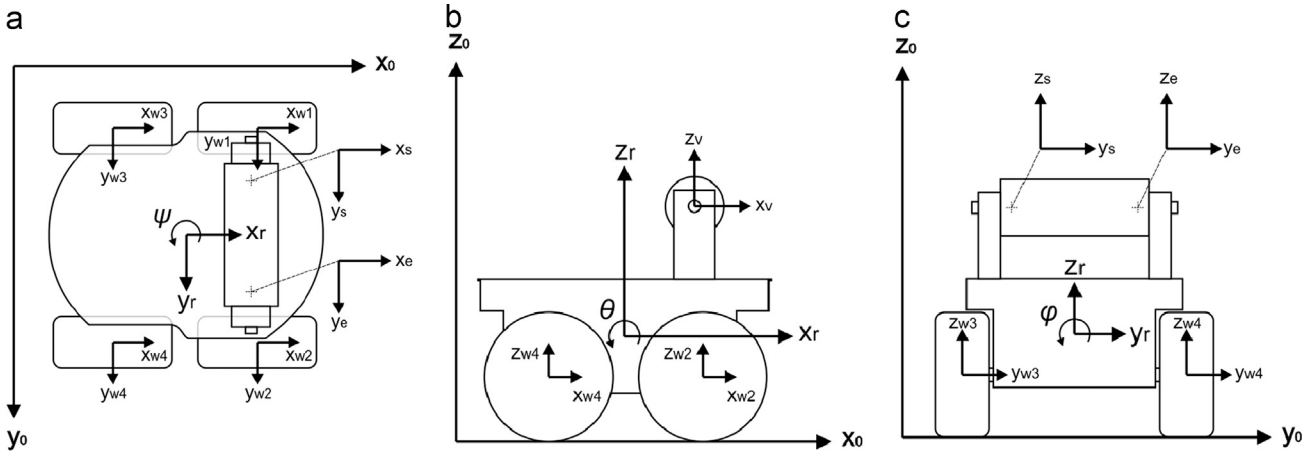


**Fig. 7.** Frames assignment: (a) top view; (b) side view; and (c) back view.

$$ {}^{0}T_{r} = \begin{bmatrix} c\varphi c\theta & c\varphi s\theta s\psi - s\varphi c\psi & c\varphi s\theta c\psi + s\varphi s\psi & x \\ s\varphi c\theta & s\varphi s\theta s\psi + c\varphi c\psi & s\varphi s\theta c\psi - c\varphi s\psi & y \\ -s\theta & c\theta s\psi & c\theta c\psi & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9} $$

$$ {}^{0}T_{r} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & c\psi & -s\psi & b \\ 0 & s\psi & c\psi & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10} $$

where the first three columns of Eq. (9) represents the unit vectors for orientation ($\vec{n}, \vec{o}, \vec{a}$), the last column represents the position vector in Cartesian coordinates ($x$, $y$, $z$), ${}^{0}T_{r}$ stands for transformation matrix of frame $r$ relative to frame 0 (or reference frame), $c\varphi$ is used as an abbreviation of $\cos\varphi$ and $s\varphi$ for $\sin\varphi$ (the same for angles $\theta$ and $\psi$).

The RPY angles (roll, pitch, yaw) sequence consist of the following transformations: a rotation of φ about the robot's $\vec{a}$ axis, called roll; a rotation of $\theta$ about the robot's $\vec{o}$ axis, called pitch; and a rotation of $\psi$ about the robot's $\vec{n}$ axis, called yaw. Angles are measured counterclockwise positive according to the right hand rule. By doing this forward kinematics analysis (Eq. (9)); We can calculate the position and orientation of the robot at any instant and we can use this equation to find the inverse kinematics equations that will enable us to calculate the variables needed to control and move the mobile robot to a desired position and orientation. RPY angles we can get using the advanced accelerometers interrogation system with LMS adaptive filter already presented in [32,33]

It is important to state that rotations values in Eq. (7) (or elements $x_{11}$–$x_{33}$ of matrix in Eq. (9)) coming from accelerometers using formalism of [32,33] and fourth column in all matrixes of Eq. (8) (or last column in general matrix of Eq. (9)) are defining the spatial position of our robot in Cartesian coordinates fixed to robot body; at the same time our TVS System (Section 2, Figs. 3–5) represent the Cartesian coordinates of the obstacles located in the same spatial volume. The Cartesian coordinates of obstacles robot can be easily transformed into the robot Cartesian system.

Our kinematic model does not have such a general nature as Dynamic Bayesian Network in due to simplification of our task conditions, but taking into account specific conditions of our robot operation it is sufficient for our practical application, and can be even more simplified for selected pair "TVS–navigation system". Taking into consideration some assumptions we can write a frame parameters table for the mobile robot, that will allow us to represent the position and orientation between every frame presented in the frame assignment (Fig. 7).

Table 1 specifies six degrees of freedom $x$, $y$, $z$ which are three linear movements for position and $\psi$, $\theta$, $\varphi$ which are the RPY angles for orientation. These parameters can be variables or known values. The assumptions we made are that our mobile robot will operate in ideal floor conditions and in an indoor environment on the first stage of laboratory experimentation (in order to prove TVS functionality for this task), therefore $\varphi$ and $\theta$ are set to 0 in ${}^{0}T_{r}$, leaving less variables to work with, simplifying the forward and inverse kinematics model (see Eq. (10)) and reducing the calculation time for the equations. Further experimentation will be carried out in a condition of non-zero state of $\varphi$ and $\theta$ parameters.

**Table 1**
Frame parameter table for mobile robot at ideal conditions.

| Transform | $x$ (mm) | $y$ (mm) | $z$ (mm) | $\psi$ (deg) | $\theta$ (deg) | $\phi$ (deg) |
|---|---|---|---|---|---|---|
| ${}^{r}T_{s}$ | −111.7 | 120.6 | 264.1 | 0 | var | 0 |
| ${}^{r}T_{w1}$ | 198.9 | −135.8 | 28.5 | 0 | var | 0 |
| ${}^{r}T_{w2}$ | −198.9 | −135.8 | 28.5 | 0 | var | 0 |
| ${}^{r}T_{w3}$ | 198.9 | 135.8 | 28.5 | 0 | var | 0 |
| ${}^{r}T_{w4}$ | −198.9 | 135.8 | 28.5 | 0 | var | 0 |

The following nomenclature is used in Table 1, ${}^{0}T_{r}$ represents the transformation matrix from the frame r to the frame 0 and the same applies for the rest of the transforms; The values on the first three columns are distances in $x$, $y$, $z$ coordinates from one frame to another and the last three columns show the angle values or rotation from one frame to another.

Assuming that the desired position and orientation achieved by Cartesian coordinates and RPY angles are known, three equations can be found from the forward kinematics model (Eqs. (12)–(14)).

$$ {}^{0}T_{r} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & px \\ r_{21} & r_{22} & r_{23} & py \\ r_{31} & r_{32} & r_{33} & pz \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c\varphi c\theta & c\varphi s\theta s\psi - s\varphi c\psi & c\varphi s\theta c\psi + s\varphi s\psi & a \\ s\varphi c\theta & s\varphi s\theta s\psi + c\varphi c\psi & s\varphi s\theta c\psi - c\varphi s\psi & b \\ -s\theta & c\theta s\psi & c\theta c\psi & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11} $$

$$ \varphi = ATAN2(r_{21}, r_{11}) \tag{12} $$

$$ \theta = ATAN2(-r_{31}, r_{11}c\varphi + r_{21}s\varphi) \tag{13} $$

$$ \psi = ATAN2(-r_{23}c\varphi + r_{13}s\varphi, r_{22}c\varphi - r_{12}s\varphi) \tag{14} $$

where Eq. (11) represents the known forward kinematic model and $\varphi, \theta, \psi$ are the unknown RPY angles. These equations form the inverse kinematic model of the mobile robot and are used to control the location of the mobile robot i.e., with this model we will be able to determine the value of each angle in order to place the mobile robot at a desired position and orientation.

## 5. Navigation system

As it was mentioned before, we are implementing our navigation system on the Pioneer 3-AT Robotics Platform which is a four-wheel, four-motor skid-steer mobile robot. Therefore some basic standards for mobility and operation have to be defined in order to implement the proposed TVS and navigation system in this platform.

### 5.1. Navigation system basic standards

The first is that due to the skid-steer characteristic of the mobile robot, the method used to steer the mobile robot to the left will be to activate the motor of wheel 2 in forward direction along with the motor of wheel 3 in reverse direction. To steer the mobile robot to the right, we will activate the motor of wheel 1 in forward direction and the motor of wheel 4 in reverse direction. During the steer operation the other two motors of the wheels that are not being used (motors 1 and 4 for left and 2 and 3 for right) will be left de-energized and free to skid in order to go along the motion. It is well known and noted that using a skid-steer method [23,24] may produce a source for uncertainty; therefore we will return to this topic in future research and compare it to other types of steering such as differential drive assisted steering [25], but for the time being we will focus on the TVS and navigation system integration.

Secondly, the main objective of our mobile robot will be to reach a final goal by avoiding any obstacles found in its trajectory. The initial trajectory will be set to the shortest secure path between the starting point and the final goal and it will be adjusted as the mobile robot detects and avoids obstacles. In the case that the mobile robot (while following a plotted trajectory) does not detect any obstacle, that trajectory will lead the mobile robot to the goal.

The mobile robot will be using optical encoders on the four wheels to keep track of the traveled distance and a 3-axis accelerometer to monitor the orientation (RPY angles) in every new current position. It will also be equipped with a set of IR sensors. These sensors will be set to detect in a fixed distance and they will be used as a means to inform the TVS that an obstacle is inbound and a scan needs to be performed.

In other words we will be using the IR sensors to perform fast obstacle detection (presence/absence) and then the TVS will scan the field of view to calculate the 3D coordinates of the obstacle (s) visible surface to establish the exact position of the detected obstacle.

### 5.2. Trajectory planning

After the mobile robot detects an obstacle in its field of view, the next logical step is to plot a new trajectory for the mobile robot to reach its goal.

In order to plot a new trajectory we first take into consideration the mobile robot size and safe margins, then we calculate a certain quantity of points in space that evade the obstacle and which will lead the mobile robot to the goal by following linear movements, then we calculate a curve that will pass through those points in order to smooth the trajectory and hence the mobile robot movement.

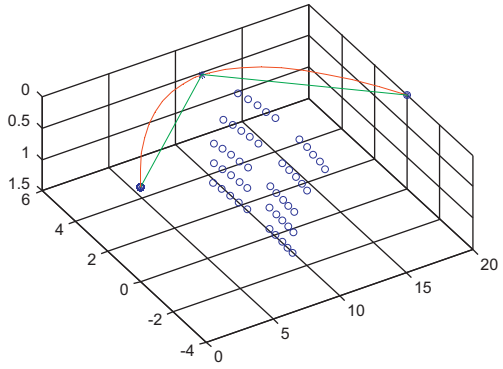


**Fig. 8.** Mobile robot trajectory from a Matlab simulation (scale in m), using a 3 points curve.

Our first approach was to use the minimum quantity of points needed to form a curve which is three, the result is a curve that successfully evades the obstacles and reaches the goal but in the other hand the traveled distance in form of a semi-circle results more than what is necessary (see Fig. 8).

Therefore we simulated another approach which uses 10 points instead of three to plot the trajectory and form the curve, the result (shown in Fig. 9b) is that the curve not only evades the obstacles and reaches the goal but it also reduces the traveled distance of the trajectory by 12–15% while the calculation time with more discrete points grows up only 1.2 ms. This trade-off permits us save robot resources significantly, meanwhile the operating time of control system increases insignificantly.

The first of the 10 points is the current location of the mobile robot, the second point is calculated to be on the left or right of the obstacle (depending on which end is closer to the mobile robot frame) and it is added a 50% safe margin and the robot's width to allow the mobile robot to go around the obstacle without colliding. Finally, from the third to the last one, which is the goal, a straight line is formed (this line is also smoothened as part of the curve, see Fig. 9).

To calculate the points of the curve, the robot can use the spline formalism. To simulate these we use an internal Matlab function called “spline”, this function given the coordinates of the 10 point mentioned before, uses a cubic spline interpolation to find the new y, which is the values of the underlying function $Y$ (range of the 10 points) at the values of the interpolant $x$ (which has the new domain values for the curve).

The results are two vectors ($x$, $y$) with the values for the new trajectory from the current location of the mobile robot to the goal (solid red line on Fig. 9). Currently the new trajectory consists of 50 calculated points, this has proven to be enough for curve smoothness but this parameter can be adjusted as needed.

On Fig. 9a, a plotted view of a new trajectory is shown. The red solid line is the calculated trajectory for the mobile robot, the lines made of blue x's indicate the physical limits of the mobile robot (width); the blue circles are the detected points of an obstacle (each point contains $x$, $y$, $z$ coordinates calculated with the TVS). Each group of blue circles represent a detected obstacle (two groups in this figure meaning two obstacles were found on this simulation), and the blue asterisks are the points used to form the trajectory curve; this is clearly seen in Fig. 9b. It is also important to note in this figure that the obstacles are being avoided by

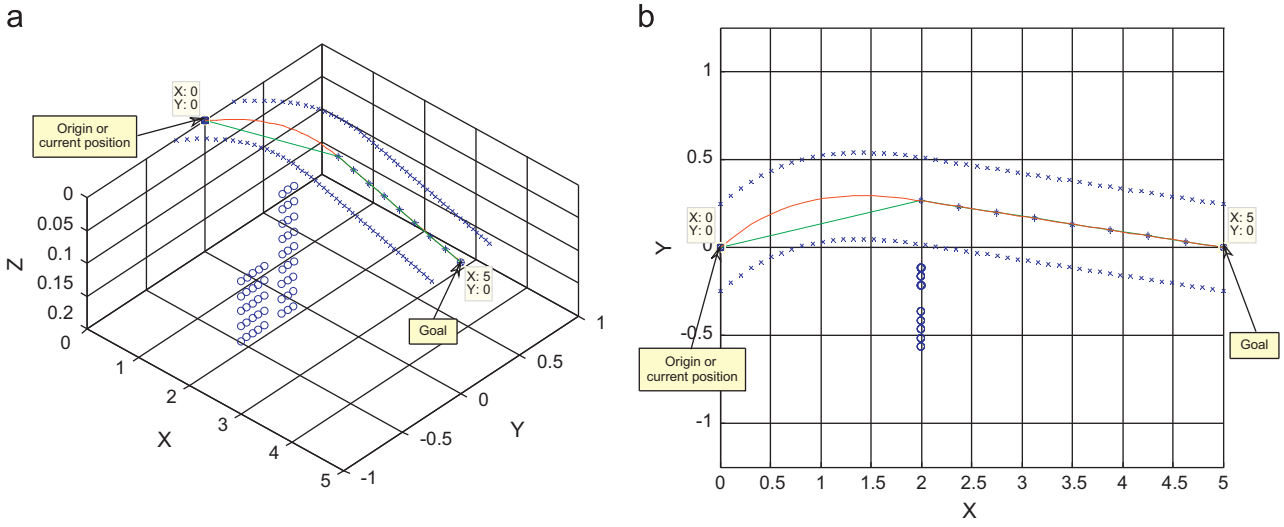


**Fig. 9.** Mobile robot trajectory from a Matlab simulation (scale in m), using a 10 points curve: (a) isometric view and (b) top view. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

the left side of the mobile robot because our algorithm estimates the preliminary level of curvature of both possible trajectories (to the left and to the right) and selects the lower curvature i.e., the shorter trajectory.

### 5.3. Navigation system operation and TVS integration

Trajectory planning is a very important element of the navigation system, but it is more important to know when to calculate it or when to call the TVS to perform a scan, and what to do before and after these processes, therefore a general operations flow chart is presented. The flowchart in Fig. 10 is for the most part self-explanatory; nevertheless a few things require further explanation.

After programming the goal of the mobile robot, all following tasks are performed autonomously. The first step is to calculate and initial trajectory, this will be a straight path between the initial position (mobile robot frame origin) and the goal. After that, the mobile robot will move forward, following the calculated trajectory and it will stop as soon as an obstacle is detected by the IR sensors or if the goal has been reached.

If an obstacle has been detected, the mobile robot will stop and will call the TVS to perform a scan of the mobile robot field of view and then it will continue to calculate the 3D coordinates of any obstacle found. After calculations are done, the data will be analyzed to obtain the main characteristics and useful information of the obstacle(s), e.g., height, width, depth, distance between obstacles (if applicable), distance from the mobile robot to the obstacle ($x$) and to the left or right limits ($y$), etc.

This information will be used to calculate a new trajectory for the mobile robot to reach its goal. After the new trajectory is calculated as explained in Section 5.2, the mobile robot will follow this trajectory until either an obstacle is found or the goal is reached.

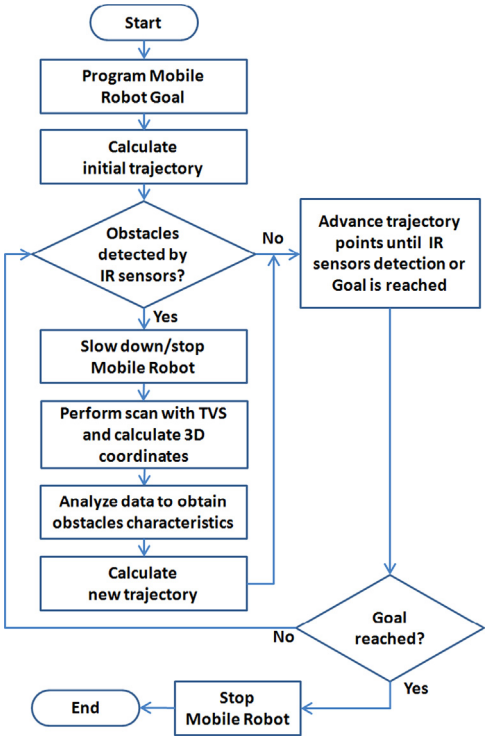


**Fig. 10.** Navigation system general operations flowchart.

If another obstacle is found, the same process will be repeated until the mobile robot reaches its goal, after that, the mobile robot will stop and enter a "waiting for new goal" mode.

## 6. Navigation system and technical vision system (TVS) simulation results

The computational experiments presented in this paper have many purposes. One of the most important is to prove the compatibility between the navigation system and TVS, in order to use the data provided by the TVS in the navigation system, this data has to be transformed and analyzed, e.g., axes rotation and adjustment, manipulation of data stored in arrays, data filtering, etc. [29,30].

As mentioned in Section 2, the data provided by the TVS comes in the form of a multidimensional $20 \times 20 \times 4$ matrix. The 1st dimension contains a digital map of the field of view of the mobile robot; each matrix $ij$ position can have a value of 0 or 1. The 2nd dimension has the $B_{ij}$ angles, the 3rd has the $C_{ij}$ angles and the $\Sigma_{\beta j}$ are on the 4th dimension.

Fig. 11 shows an example in which $i=3$ and $j=3$ therefore $B_{ij}=71^\circ$ (laser angle), $C_{ij}=58^\circ$ (aperture angle) and $\Sigma_{\beta j}=4^\circ$ (vertical tilt angle).

MATLAB® (R2012a) was used to perform the simulations. The program named BASACA NAVIGATION SYSTEM (BANS) is designed to calculate robot's desired trajectory basing on 3D data from TVS to avoid collision with detected and located obstacles and at the same time avoiding sharp movements of robot's steering. Part of the code (see Appendix A) was written to calculate the 3D coordinates of detected obstacles on a scanned field of view of the mobile robot, using Eqs. (2)–(5), presented in Section 2.

The code mentioned before is just a fragment of the main Matlab program, what it does is that using two nested FOR cycles, the program runs through every position of the $20 \times 20 \times 1$ matrix and (the digital map) and if the contents of an $ij$ position is 1, it calls for the $B_{ij}$, $C_{ij}$, and $\Sigma_{\beta j}$ values of that specific position (on the other dimensions). While it calculates the 3D coordinates, the program creates a $20 \times 20 \times 3$ matrix to store the results, the 1st dimension stores $x$ values, the 2nd $y$ values and the 3rd, $z$ values.

This program also calculates the elapsed time on the coordinate's calculation, the obstacles characteristics and important distances to name a few; a fragment of the output information of BANS used for tests monitoring is shown in Appendix A.

On this simulation, after the new trajectory is calculated, the program uses $x$, $y$, $z$ (3D coordinates calculated before) to form a 3D plot of the obstacle(s), then the origin or the mobile robot current location and the goal are added to the graph and after that, it plots in that same graph the $x$ and $y$ values of the new trajectory to demonstrate that the path evades the obstacle and reaches the goal. For proper autonomous robot operation, graphs will not be

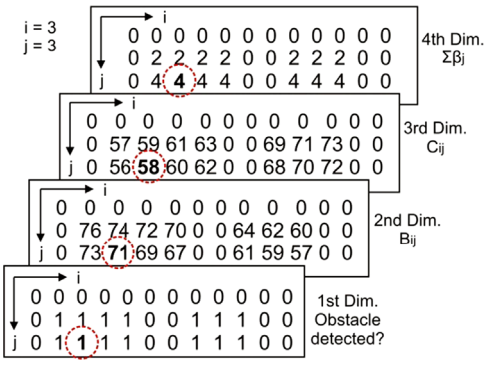


**Fig. 11.** Multidimensional matrix for TVS data storage and manipulation.

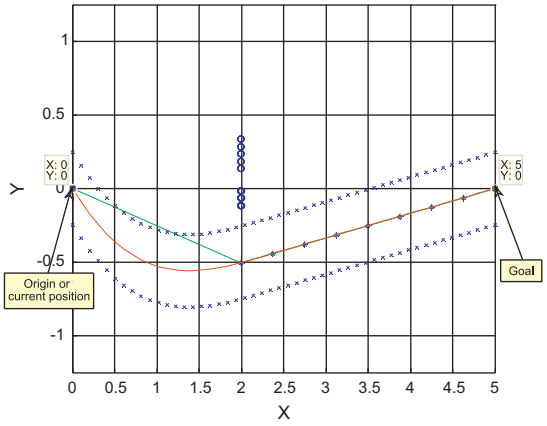


**Fig. 12.** Top view of the trajectory plotted to avoid obstacle by the right side of the mobile robot.

**Table 2**
TVS and navigation system calculation time distribution.

| Operation | Calculation time (ms) | (%). |
|---|---|---|
| Obstacle's 3D coordinates | 20.1590 | 20.5 |
| Obstacle's characteristics | 38.1835 | 38.7 |
| Decision making | 10.3025 | 10.5 |
| New trajectory | 6.8921 | 7.0 |
| Inverse kinematics | 12.2402 | 12.4 |
| Total time | 98.5592 | 100 |

processed or displayed to save processing time. Part of the code used to calculate the new trajectory is shown in Appendix A.

To calculate the new trajectory, first we need to know the current location of the mobile robot and the goal coordinates, the information needed is obtained from the homogeneous transform matrix and then as explained in Section 5.2, the trajectory is calculated and plotted. Fig. 9a and b shows the result of the obstacle(s) 3D coordinates and new trajectory calculation obtained with the MATLAB® code presented in this section.

In Fig. 12, a different set of inputs is being used in the simulation; therefore we have different obstacle locations and obstacle avoidance by the right side of the mobile robot.

The different algorithms and operations calculated during simulations were categorized and monitored to obtain their average duration time. For every category, 30 time samples were measured and averaged to obtain Table 2 and Fig. 13, this figure reflects the relations in time between different stages in robot's navigation, such as: data adjustment from TVS to the navigation system, obstacle's detection, location and characteristics, decision making, new trajectory planning and inverse kinematics.

The characteristics of the computer used to perform the calculations and simulations are

- Intel® Core™ i7 CPU Q720@1.6 GHz
- DDR3 8 GB RAM
- Windows 7 64-bit Operating System

## 7. Experimentation results and statistical analysis of the obtained data

Experimentation was performed in a controlled laboratory environment. A calibrated optical table was used to support the TVS and run 2D measurement tests with static obstacles at known locations, the third coordinate was not considered because of the same electric motor and other similar conditions of measurement, but it will be included in near future tests.

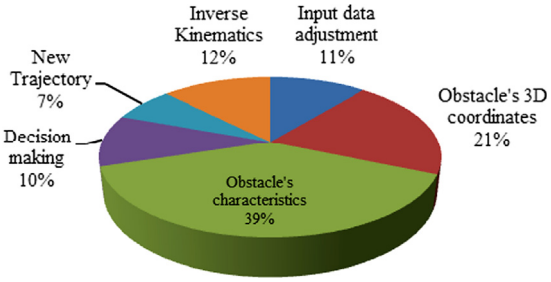


**Fig. 13.** Pie chart of time distribution in different stages of calculations in TVS and navigation system.

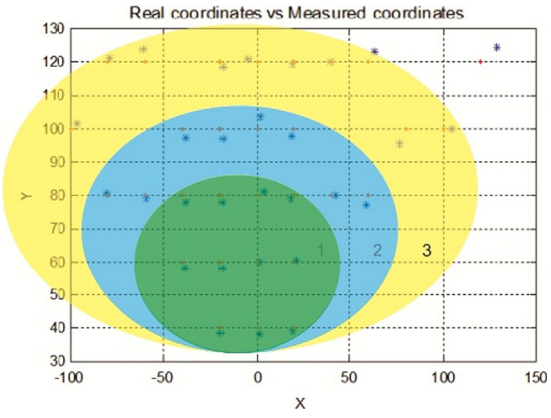


**Fig. 14.** Comparison graph of obstacle's real coordinates and measured coordinates with the TVS prototype I (scale in cm). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

The tests performed proved that the second TVS prototype has a higher confidence level than its predecessor. According to our previous publications [15,16,35]; the first TVS prototype demonstrated 95% confidence level for coordinates measurements at the center of its field of view and it decreased for the angles at the ends. We identified three different regions inside the field of view of the TVS prototype I and each one presented different accuracy or confidence level.

The three regions are shown in Fig. 14; the first and smaller region (green) proves to be the best measurement space of the TVS with 95% average confidence level. The second (blue) and third (yellow) regions, demonstrated lower measurement confidence level. The finding of these areas allowed us to establish the working space of the TVS which is a very important piece of information to the navigation system of the mobile robot platform.

However, the first tests with TVS prototype II showed a wide range of confidence level, from 85% to 98% without any kind of data manipulation; this increase is mostly due to the mechanical improvements explained in [16]. In order to increase and stabilize the measurements confidence level a neural network was developed using the Levenberg–Marquardt method. This method is used primarily in the least square curve fitting problem: given a set of m empirical datum pairs of independent and dependent variables, $(x_i, y_i)$ optimize the parameters b of the model curve $f(x, \beta)$ so that the sum of the squares of the deviations becomes minimal [31,34]

$$S(\beta) = \sum_{i=1}^{m} [y_i - f(x_i, \beta)]^2 \quad (15)$$

**Table 3**
Average confidence level and error of TVS prototype II coordinates measurements.

| | | |
|---|---|---|
| Average confidence level of calculated coordinates (%) | $x$<br>$y$ | 93.14%<br>93.93% |
| Average confidence level of corrected coordinates (%) | $x$<br>$y$ | 98.58% (+5.44% increase)<br>97.54% (+3.61% increase) |
| Average calculated error | $\Delta B$ (deg)<br>$\Delta x$ (m)<br>$\Delta y$ (m) | 3.9466<br>0.0686<br>−0.0592 |
| Average corrected error | $\Delta B$ (deg)<br>$\Delta x$ (m)<br>$\Delta y$ (m) | 0.2998<br>0.0095<br>−0.0146 |

**Table 4**
TVS prototype II angle measurements and coordinates calculation (simplified table).

| Real values (deg, m) | | | | Measured angles (deg) | | Calculated coordinates (m) | | Calculated errors | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ***C* angle** | ***B* angle** | ***X*** | ***Y*** | ***C* angle** | ***B* angle** | ***X*** | ***Y*** | **$\Delta B$ (degrees)** | **$\Delta X$ (m)** | **$\Delta Y$ (m)** |
| 64.7 | 62.3 | 1.0 | −0.027 | 64.9687 | 67.5996 | 1.1375 | −0.0312 | 5.2996 | 0.13746 | −0.0042 |
| 62.6 | 64.25 | 1.0 | −0.016 | 62.9625 | 67.1964 | 1.0744 | −0.0483 | 2.9464 | 0.0744 | −0.0323 |
| 61.3 | 65.5 | 1.0 | −0.046 | 61.9687 | 68.9497 | 1.0902 | −0.0804 | 3.4497 | 0.0902 | −0.0344 |
| 59.7 | 67.3 | 1.0 | −0.0850 | 59.9625 | 73.4938 | 1.1434 | −0.1612 | 6.1938 | 0.1434 | −0.0762 |
| … | … | … | … | … | … | … | … | … | … | … |
| 50 | 40.8 | 0.5 | 0.078 | 49.9312 | 45.3699 | 0.5470 | 0.0399 | 4.5699 | 0.0470 | −0.0381 |
| 46.8 | 43.6 | 0.5 | 0.026 | 46.9500 | 45.7943 | 0.5244 | 0.0101 | 2.1943 | 0.0244 | −0.0159 |
| 45 | 45.2 | 0.5 | −0.002 | 44.9438 | 46.4971 | 0.5126 | −0.0136 | 1.2971 | 0.0126 | −0.0116 |
| 43 | 47.3 | 0.5 | −0.038 | 42.9562 | 49.0435 | 0.5149 | −0.0531 | 1.7435 | 0.0149 | −0.0151 |

**Table 5**
TVS corrected angle measurements, coordinates and error calculation (simplified table).

| Real values (deg, m) | | | | Measured angles (deg) | | Corrected angles and new calculated coordinates | | | New calculated error | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ***C* angle** | ***B* angle** | ***X*** | ***Y*** | ***C* angle** | ***B* angle** | ***B* angle** | ***X*** | ***Y*** | **$\Delta B$ (deg)** | **$\Delta X$ (m)** | **$\Delta Y$ (m)** |
| 64.7 | 62.3 | 1.0 | −0.027 | 64.9687 | 67.5996 | 62.6312 | 1.0156 | 0.0257 | 0.3312 | 0.0156 | 0.0527 |
| 62.6 | 64.25 | 1.0 | −0.016 | 62.9625 | 67.1964 | 64.3507 | 1.0096 | −0.0152 | 0.1007 | 0.0096 | 0.0008 |
| 61.3 | 65.5 | 1.0 | −0.046 | 61.9687 | 68.9497 | 65.4971 | 1.0119 | −0.0388 | −0.0029 | 0.0119 | 0.0072 |
| 59.7 | 67.3 | 1.0 | −0.0850 | 59.9625 | 73.4938 | 68.5265 | 1.0292 | −0.0951 | 1.2265 | 0.0292 | −0.0101 |
| … | … | … | … | … | … | … | … | … | … | … | … |
| 50 | 40.8 | 0.5 | 0.078 | 49.9312 | 45.3699 | 41.8864 | 0.5112 | 0.0700 | 1.0864 | 0.0112 | −0.0080 |
| 46.8 | 43.6 | 0.5 | 0.026 | 46.9500 | 45.7943 | 42.8568 | 0.4970 | 0.0357 | −0.7432 | −0.0030 | 0.0097 |
| 45 | 45.2 | 0.5 | −0.002 | 44.9438 | 46.4971 | 43.9339 | 0.4902 | 0.0088 | −1.2661 | −0.0098 | 0.0108 |
| 43 | 47.3 | 0.5 | −0.038 | 42.9562 | 49.0435 | 46.9810 | 0.4982 | −0.0351 | −0.3190 | −0.0018 | 0.0029 |

Levenberg algorithm had some disadvantages [36] and Marquardt contributed to the problem which resulted in the Levenberg–Marquardt algorithm, which is the one we applied to our neural network [31]

$$(J^T J+\lambda diag(J^T J))\delta = J^T[y-f(\beta)] \tag{16}$$

The neural network was trained with the raw data from the TVS and the result was a 5.44% confidence level increase on $x$ coordinate measurements and 3.61% increase on $y$, i.e., the neural network effectively corrected the $x$ and $y$ coordinates measurements, minimizing the average error in $x$ to less than 1 cm and in $y$ to 1.5 cm.

The neural network was integrated in the TVS test measurements software to correct the error in the $B_{ij}$ angle measurement in real time, this resulted in the before mentioned accuracy increase. See Table 3 for the summary of the analysis of the experimentation data and Tables 4 and 5 for a sample of angle measurements and coordinate calculations and errors.

After correcting the TVS prototype II measurements, the results demonstrated a different behavior in comparison to its

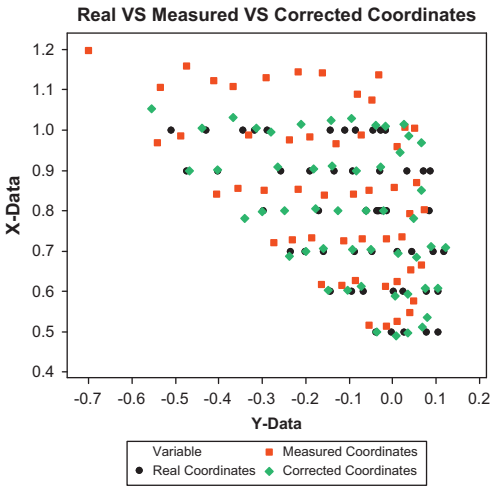


**Fig. 15.** Comparison graph of obstacle's real, measured and corrected coordinates with the TVS prototype II (scale in m).

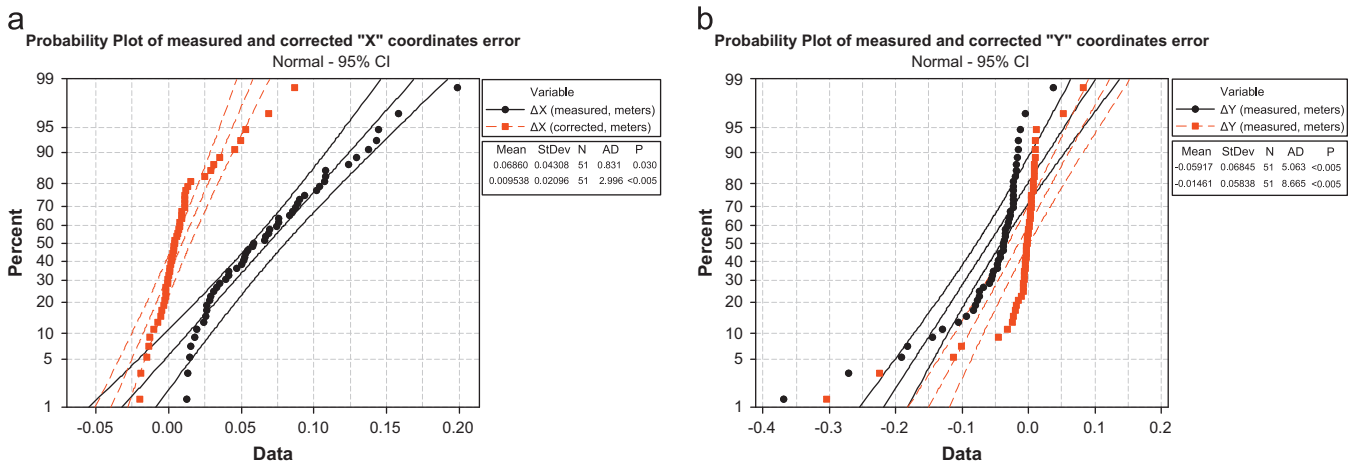


**Fig. 16.** Probability plots of $x$ and $y$ measured and corrected coordinates errors: (a) $x$ error tends to 0 m; and (b) $y$ error tends to 0 m.

predecessor, instead of having a divided field of view with different accuracy regions, we observed that the error was behaving in the same way for the most part of the field of view, effectively leaving only one region with a stable confidence level range which is 95–99% for both $x$ and $y$ coordinates. This can be observed in Fig. 15, while the measured coordinates error changes according to the distance and angle to the center of the TVS ($x=0$, $y=0$), the error in the corrected coordinates remains low and stable.

Using the experimentation data from the TVS prototype II we generated probability plots with the errors from the measured and corrected coordinates ($x$ and y). In these plots, Fig. 16a and b, it can be observed that the error is consistent at 0 cm for the most part of the field of view, but it can vary by 1–1.5 cm in the ends.

## 8. Conclusions

In the present paper we introduced the use of our novel original TVS based on dynamic triangulation [15,16,26,27] for practical navigation of a mobile robot. The data obtained by TVS was used for the control of the robot under simulation (Pioneer 3-AT, as most popular among low-cost robots in applied laboratory tasks) considering robot's kinematics under mathematical model as Eqs. (6)–(14)).

BANS was firstly designed to permit calculations of the mobile robot's desired trajectory as a continuous line built on n discrete points (where $n$ is a real number reasonably constrained by 500 in real time for the considered TVS and mobile robot), basing on 3D data from TVS to avoid collision with detected and located obstacles and at the same time avoiding sharp movements of robot's steering. Thus, two principal advantages are assumed over other known robot navigators [26–28] (1) smoother trajectory to facilitate the work of the mechanical part of robot's steering, and (2) significantly shorter trajectory (optimized by variation of $n$) up to 12–16% length of the robot's path on each obstacle, which is a crucial point for any navigation principle from the point of view of resources saving (fuel and electric energy).

One of the most attractive impacts of BANS is that it gives in real time (exactly during the calculations) the values of variables in Eqs. (11)–(14), in each of $n$ points, which are necessary to perform robot's kinematic control.

Also it is notable that the use of 3D laser TVS as sensory part of robot's navigation is a unique system among known, which gives a metrological quality of the obstacle location and shape. It permits to consider the robot and the obstacle as parts of the same notation system, e.g., Cartesian coordinates, without post-processing or probabilistic solutions.

It is important to note that our system in comparison to others has significantly wide opening angle of field of view (theoretically up to $\approx 160^{\circ}$), while omnidirectional vision (fish-eye like) claim to provide 180°, actually due to image curvature and excessive postprocessing time, those systems are still at a disadvantage; and our system is a perfect match for operation in completely dark environment (e.g., mines, caves exploration, rescue operations, etc.), which is completely impossible for camera systems.

Although 3D scene reconstruction is not a requirement for mobile robot navigation, short term future investigation includes tests and detailed 3D measurements data for surface reconstruction, obtained with the presented TVS to experiment in different applications such as computer graphics, medical diagnostics [31] and structural health monitoring [19], which we will demonstrate that are within our TVS capabilities.

Finally, the use of the chosen among three others method of Levenberg–Marquardt [31] helps us to increase the metrological accuracy of 3D points coordinates up to 5%, comparing to our own previous results.

## Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.optlaseng.2013.08.005.

## References

[1] Cheng Y, Maimone MW, Matthies L. Visual odometry on the Mars exploration rovers—a tool to ensure accurate driving and science imaging. IEEE Robotics Autom Mag 2006;13(2):54–62.
[2] Fairfield N, Kantor G, Wettergreen D. Segmented SLAM in three-dimensional environments. J Field Robotics 2010;27(1):85–103.
[3] Thrun Sebastian, Montemerlo Mike, Dahlkamp Hendrik, Stavens David, Aron Andrei, Diebel James, et al. Stanley: the robot that won the DARPA Grand Challenge.
J Robotic Syst 2006;23:9.
[4] Buehler M, Iagnemma K, Singh S, editors. The 2005 DARPA Grand Challenge: the great robot race. Berlin: Springer; 2006.
[5] Urmson C, Anhalt J, Bagnell Drew, Baker Christopher, Bittner Robert, Clark MN, et al. Autonomous driving in urban environments: Boss and the Urban Challenge. J Field Robotics 2008;25:425–66, http://dx.doi.org/10.1002/rob.20255.
[6] Bacha A, Bauman R, Faruque A, Fleming M, Terwelp C, et al. Odin: Team ictorTango's entry in the DARPA Urban Challenge. J Field Robotics, 25; 467–92.
[7] Leonard J, Barrett D, How J, Teller S, et al. Team MIT Urban Challenge Technical Report. CSAIL Technical Reports. Available from: 〈http://hdl.handle.net/1721.1/39822. MIT-CSAIL-TR-2007-058〉 ; 2007 [accessed 11.06.11].

[8] Surmann H, Lingemann K, Nüchter A, Hertzberg J. A 3D laser range finder for autonomous mobile robots. In: Proceedings of the 32nd international symposium on robotics; 2001: p. 153–8.
[9] Andersen JC, Blas MR, Ravn O, Andersen NA, Blanke M. Traversable terrain classification for outdoor autonomous robots using single 2D laser scans. Integrated Comput-Aided Eng 2006;13(3):223–32.
[10] Genovesea K, Casalettoa L, Rayasb JA, Floresb V, Martinez Amalia. Stereo-Digital Image Correlation (DIC) measurements with a single camera using a biprism. Opt Lasers Eng 2013;51(3):278–85.
[11] Zhou Yihao, Chen Yan Qiu. Feature matching for automated and reliable initialization in three-dimensional digital image correlation. Opt Lasers Eng 2013;51(3):213–23.
[12] Pintoa Tiago, Kohlerb Christian, Albertazzi Armando. Regular mesh measurement of large free form surfaces using stereo vision and fringe projection. Opt Lasers Eng 2012;50(7):910–6.
[13] Penga Zhao, Guo-Qiangb Ni. Simultaneous perimeter measurement for 3D object with a binocular stereo vision measurement system. Opt Lasers Eng 2010;48(4):505–11.
[14] Lipnickas A, Knyš A. A stereovision system for 3-D perception. Electron Electr Eng 2009;91:99–102.
[15] Basaca LC, Rodriguez JC, Sergiyenko O. 3D laser scanning vision system for autonomous robot navigation. In: Proceedings of IEEE international symposium on industrial electronics ISIE-2010, Bari, Italy; 2010: p. 1773–8.
[16] Luis C. Básaca, Julio C. Rodríguez, Oleg Sergiyenko, Vera V. Tyrsa, Wilmar Hernández, Juan I. Nieto Hipólito, et al., Resolution improvement of dynamic triangulation method for 3D vision system in robot navigation task. In: Proceedings of IEEE—36th annual conference of IEEE industrial electronics IECON'10; 2010: p. 2886–91.
[17] Oleg Sergiyenko, Vira Tyrsa, Luis Carlos Basaca Preciado, Julio Cesar Rodriguez Quinonez, Wilmar Hernandez, Juan Ivan Nieto Hipolito, et al., Optoelectronic devices and properties. Electromechanical 3D optoelectronic scanners: resolution constraints and possible ways of improvement. Austria: IN-TECH; 2010 (p. 549–82).
[18] Rivas Lopez M, Sergiyenko O, Tyrsa V. Computer vision. Machine vision: approaches and limitations. Austria: I-TECH; 2008 (p. 395–428).
[19] Moisés Rivas, Oleg Sergiyenko, Mario Aguirre, Luis Devia, Vera Tyrsa, Ismael Rendón, Spatial data acquisition by laser scanning for robot or SHM task. In: IEEE–IES Proceedings of the international symposium on industrial electronics ISIE-2008, Cambridge, United Kingdom; 2008: p. 1458–63.
[20] Sergiyenko O, Tyrsa V, Hernandez-Balbuena D, Rivas López M, Rendón López I, Devia Cruz L. Precise optical scanning for practical multiapplications. In: Proceedings of the IEEE 34th Annual IEEE Conference IECON'08, Florida, USA; 2008: p. 1656–61.
[21] Sergiyenko OYu. Optoelectronic system for mobile robot navigation. Optoelectron, Instrum Data Process 2010;46(5):414–28.
[22] Niku SB. Introduction to robotics: analysis, systems, applications. Indianapolis, IN: Wiley; 2010.
[23] Kozłowski K, Pazderski D. Modeling and control of a 4-wheel skid-steering mobile robot. Int J Appl Math Comput Sci 2004;14(4):477–96.
[24] Kang J, Kim W, Lee J, Yi K. Design, implementation, and test of skid steering-based autonomous driving controller for a robotic vehicle with articulated suspension. J Mech Sci Technol 2010;24(3):793–800.
[25] Wang J, Wang Q, Jin L, Song C. Independent wheel torque control of 4WD electric vehicle for differential drive assisted steering. Mechatronics 2011;21 (1):63–76.
[26] Sergiyenko O, Hernandez W, Tyrsa V, Devia Cruz L, Starostenko O, Pena-Cabrera M. Remote sensor for spatial measurements by using optical scanning. Sensors 2009;9(7):5477–92.
[27] Oleg Y. Sergiyenko, Vera V. Tyrsa, Luis F. Devia, Wilmar Hernandez, Oleg Starostenko, Moisés Rivas Lopez, Dynamic laser scanning method for mobile robot navigation. In: Proceedings of the ICCAS–SICE 2009 ICROS–SICE international joint conference, Japan; 2009: p. 4884–9.
[28] Du Y, Gao X, Liu Z, Sun M. The new navigation system for automatic guided vehicle. In: Proceedings of the Chinese control and decision conference; 2008: p. 4653–8.
[29] Byczkowski T, Lang J. A stereo-based system with inertial navigation for outdoor 3D scanning. In: Proceedings of the Canadian conference on computer and robot vision, CRV'09; 2009: p. 221–8.
[30] Hwang CL, Shih CY. A distributed active-vision network-space approach for the navigation of a car-like wheeled robot. IEEE Trans Ind Electron 2009;56 (3):846–55.
[31] Rodriguez-Quiñones JC, Sergiyenko O, González-Navarro FF, Basaca-Preciado LC, Tyrsa V. Surface recognition improvement in 3D medical laser scanner using Levenberg–Marquardt method. Signal Process (Elsevier) February 2013;93(2):378–86 (ISSN 0165-1684).
[32] Hernandez W, de Vicente J, Sergiyenko O, Fernández E. Improving the response of accelerometers for automotive applications by using LMS adaptive filters, MDPI, sensors1424-8220. Switzerland: Basel; http://dx.doi.org/10.3390/s100100313 (p. 313–29).
[33] Hernandez W, de Vicente J, Sergiyenko O, Fernández Eduardo. Improving the response of accelerometers for automotive applications by using LMS adaptive filters: Part II, MDPI, sensors1424-8220. Switzerland: Basel; http://dx.doi.org/10.3390/s100100952 (p. 952–62).
[34] Abba A, Caponio F, Geraci A, Ripamontii G. Non-linear leastsquares in FPGA devices for digital spectroscopy. Nuclear Science Symposium Conference Record (NSS/MIC), Orlando; 2009: p. 563–8.
[35] Basaca-Preciado LC, Sergiyenko O, Rodriguez-Quinonez JC, Rivas-Lopez M. Optoelectronic 3D laser scanning technical vision system based on dynamic triangulation. In: Proceedings of the 2012 IEEE photonics conference (IPC); 2012: p. 648–9.
[36] Marquardt D. An algorithm for least-squares estimation of nonlinear parameters. SIAM J Appl Math 1963;11(2):431–41.
[37] Zhang Song, Huang Peisen S. High-resolution, real-time three-dimensional shape measurement. Opt Eng 2006;45(12).