A Portable Ground-Truth System Based on a Laser Sensor

Román Marchant, Pablo Guerrero, and Javier Ruiz-del-Solar

Department of Electrical Engineering,
Universidad de Chile, Avenida Tupper 2007, Casilla 412-3, Santiago, Chile
{romarcha,pguerrer,jruizd}@ing.uchile.cl
http://www.die.uchile.cl/

Abstract. State estimation is of crucial importance to mobile robotics since it determines in a great measure its ability to model the world from noisy observations. In order to quantitatively evaluate state-estimation methods, the availability of ground-truth data is essential since it provides a target that the result of the state-estimation methods should approximate. Most of the reported ground-truth systems require a complex assembly which limit their applicability and make their set-up long and complicated. Furthermore, they often require a long calibration procedure. Additionally, they do not present measures of their accuracy. This paper proposes a portable laser-based ground-truth system. The proposed system can be easily ported from one environment to other and requires almost no calibration. Quantitative results are presented with the purpose of encouraging future comparisons among different ground-truth systems. The presented method has shown to be accurate enough to evaluate state-estimation methods and works in real time.

Keywords: Ground Truth, Laser.

1 Introduction

In the last decades, a vast work on state estimation and its applications to mobile robotics has been carried out. The existent solutions are able to estimate non-observable variables from limited and noisy observations. The applications of state estimation to mobile robotics include robot's self-localization and object tracking, which are essential when performing complex tasks such as navigation and interacting with mobile and static objects. In order to quantitatively evaluate the performance of a state-estimation method, a common practice is to compare the estimates obtained in an experiment by the evaluated method with a set of data, known as *ground-truth data* (GTD), assumed to be very accurate. We will call *ground-truth system* (GTS) to any system able to get the GTD in real time, i.e., a system that is able to give at every instant a very accurate estimate of the state. Applications of a GTS to the robotics field are on the following areas: (i) Method evaluation: The obtained GTD is used as a target data set to compare with. (ii) Environment monitor: The obtained GTD represents the current state of the environment, used for monitoring unattended robots.

(iii) External Control: The obtained GTD is used by an external robot controller for calculating the state error and the corresponding command. (iv) Training: Having a GTS would make easier some automatic training procedures such as gait training, automatic vision calibration, etc.

In most common situations, an environment with a global static origin, O, contains various types of objects, classified into two main classes: (i) Static Objects: Objects with fixed pose relative to O. These objects are usually used as landmarks. (ii) Mobile Objects: Objects with dynamic pose relative to O. In most robotics applications, the goal of a GTS is to achieve a precise estimation of the kinematic state of all the objects in the environment.

Most reported GTSs use information provided by a static camera located above the scene of interest, although some of them employ laser sensors or radars. Usually, camera-based GTSs require large complex structures, containing pillars and a ceiling mesh that holds the sensors. This fact makes the assembly procedure long and complicated and imposes a minimum to the ceiling height.

In this work, a GTS based on a laser sensor is presented. The main feature of the proposed system is its portability. The proposed GTS is said to be portable because its installation is quick and easy and it requires almost no extra space. A tripod, a laser sensor and a laptop are all the required hardware. This makes the system also have a low cost. Additionally, the system requires no extra calibration when changed to a different environment with a priori known map and can recover very fast from movements of the sensor. This feature also improves the portability of the system.

We propose that the different GTS alternatives should be compared from different viewpoints including, of course, their accuracy. To allow such a comparison, the presented GTS is characterized in terms of its accuracy and portability.

The proposed GTS can be applied in any situation, where an object of interest moves along a fixed plane and a single laser scan give enough information to classify them and determine their position (and orientation if needed). In the cases where the orientation is required and the symmetries of the object does not allow the determination of their orientation, a body extension may be added to the robots in order to eliminate the symmetries. The proposed GTS have been tested in a Robocup Standard Platform League (SPL) field and would probably work in a Humanoid League (HL) environment.

This document is structured as follows. Section 2 points out related work. Section 3 makes a brief review of the employed regression methods, while section 4 describes the proposed system in detail. Section 5 presents the experimental results in pose estimation. Finally, section 6 draws some conclusions and suggests some future work.

2 Related Work

The global perception of an interesting scene has been addressed by many authors (e.g. [12,13,7,5,4,3,2]), and it is currently relevant for several applications. The estimation of the state of the environment may be used for monitoring autonomous robots ([1,11]), evaluation of localization methods ([9,10]) and ground-truth database generation ([7]).

Various sensing strategies have been applied, even though most of the existing work on the area uses vision-based systems, placing one or more cameras held by a complex structure on top of the scene ([12,13,5,4,3,2,7]). To the best of our knowledge, a laser-based strategy for a GTS has not been addressed in the literature, except for [9] where no description of the global perception methodology is presented.

At the moment, there are GTSs applied to some Robocup leagues. For instance, the standardized method used at Robocup Small Size League (SSL) is [13]. This work implements an image processing algorithm based on color segmentation that allows pattern recognition with the purpose of estimating each robot's id and pose. In addition, it considers camera-pose calibration, defining the relationship between the field geometry and the image plane. Although no occlusions are possible, this work does not include tracking of detected objects and does not present quantitative results on estimation error, preventing any quantitative comparison with any other methodology. Furthermore, the system requires that each robot has a flat surface in its top that does not rotate with respect to the robot which makes it not suitable for applications with headed robots, which is the case of the Robocup SPL and HL, for instance.

The GTS described on [7] uses several reflectors located in the body of the robot and fifteen infrared cameras to detect them. Due to the redundancy of the sensors, the system accuracy is in the order of millimeters for each reflected point, and occlusions are unlikely to affect. Another positive feature is that it can independently track the movement of the head, allowing it to calculate the orientation of the robot's cameras. An important fact is that this work uses a tracking system and matches space state information with local information on the interesting robot, generating a GTD database with global and local information. However, that GTS requires a very complex hardware structure, hardly reproducible. Furthermore, [7] does not contain any detail on the system algorithms to estimate the robot's pose, neither presents quantitative results on pose estimation error.

The main drawback of the existing GTSs is that they make use of large structures, minimizing portability and disabling its utilization on multiple scenarios. Furthermore, they do not present error statistics thus preventing numerical comparisons among different GTSs. The proposed system differs from other reported GTSs because of its portability. Additionally, an estimation of the measurement errors in position and angle of the presented GTS for fixed robot positions on a Robocup SPL field is presented. As mentioned before, we believe that comparing different methodologies is fundamental, and we expect to encourage other authors to report the error statistics of their system.

3 Employed Regression Techniques

Since several parts of the presented methodology employ some regression techniques, we will briefly review them. Linear regression and *Gaussian Process* (GP) regression are the utilized techniques and thus will be reviewed.

The regression problem may be formulated in the following terms. There is a training set of N_t samples $\{\mathbf{x}_i\}$ in an N_x-dimensional input space $\mathcal{X}$ and a correspondent set of targets $\{y_i\}$.[1] We want to learn a function, f, from the training data and be able to evaluate it at any test input $\mathbf{x}_* \in \mathcal{X}$. For convenience we will define the matrix $X = [\mathbf{x}_0, \ldots, \mathbf{x}_{N_t-1}]^T$ and the vector $\mathbf{y} = [y_0, \ldots, y_{N_t-1}]^T$. The here-described regression techniques assume that the set of targets $\{y_i\}$ have zero mean. If that is not the case, the mean of $\{y_i\}$ must be subtracted to each y_i and then added to any test output.

a. Linear Regression

The linear regression consists in predicting any test output as a linear function of the test input:

$$f(x_*) = Ax_* \tag{1}$$

The here-employed linear regression mechanism is known as *ordinary least squares* (OLS) which is the most used choice because of its great simplicity. OLS assumes also the training input samples $\{\mathbf{x}_i\}$ to have zero-mean. Again, if that is not the case, the mean of $\{\mathbf{x}_i\}$ must be subtracted from each $\{\mathbf{x}_i\}$ and from any test input. OLS estimates the linear transformation matrix, A, as:

$$A = \left(X^T X\right)^{-1} X^T \mathbf{y} \tag{2}$$

b. Gaussian Process Regression

Gaussian Processes are a non-parametric tool for non-linear regression and classification. For space considerations, we will make a extremely summarized review of GPs for regression from a practical viewpoint. An excellent review that includes both theoretical and practical aspects and references to deeper theoretical insights can be found in [8].

A key component of GPs are the so-called *covariance functions.* Covariance functions encode the information of the kind of functions that can be learned by a GP. They also give restrictions to the possible measures of proximity that are necessary for the regression mechanism to operate. Usually covariance functions have parameters, called *hyperparameters* and the GPs theory bring a method to calculate them from data. We use a *squared-exponential covariance function* for the regression:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}') W (\mathbf{x} - \mathbf{x}')^T\right) \tag{3}$$

Where W is a diagonal matrix with scaling factors in its diagonal and σ_f^2 is the so-called *signal variance.* Then, for the squared exponential covariance function, the hyperparameters vector is $\mathbf{h} = \left(W_{(0,0)}, \ldots, W_{(N_x,N_x)}, \sigma_f^2, \sigma_n^2\right)$,

[1] Note that, for simplicity, every y_i is a scalar but there is a very simple procedure for generalizing the reviewed methods to multidimensional outputs that consists of learning an independent one-dimensional function for each dimension of the output

where σ_f^2 is the so-called *signal variance*. The optimal hyperparameters, $\mathbf{h}^*$, are learned by maximizing the *log marginal likelihood*:

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \left(\log p\left(\mathbf{y}\left|X\right.\right)\right) \tag{4}$$

$$\log p\left(\mathbf{y}\left|X\right.\right) = -\frac{1}{2}\mathbf{y}^T\left(K_X + \sigma_n^2 I\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left|K_X + \sigma_n^2 I\right| - \frac{N_t}{2}\log 2\pi \tag{5}$$

Where $K_X = K(X, X)$ and the matrix function K can be defined in a component-wise fashion:

$$K(X', X'')_{(i,j)} = k\left(\mathbf{x}'_i, \mathbf{x}''_j\right) \tag{6}$$

With $\mathbf{x}'_i$ the i^{th} column of X' and $\mathbf{x}''_j$ the j^{th} column of X''. Finally, the output for any testing point is predicted using the GP predictive equation:

$$f\left(\mathbf{x}_*\right) = K_*^T\left(K_X + \sigma_n^2 I\right)^{-1}\mathbf{y}, K_* = K\left(X, \mathbf{x}_*\right) \tag{7}$$

4 Methodology

The presented system consists of a precise laser sensor that provides data input, and a software module that processes information and delivers GTD output, represented as a 2D object map. The system's main goal is to detect objects in the environment and to estimate their pose relative to the environments global origin.

A laser sensor, with field of view with size 2α, is manually placed as close as possible to an initial theoretical pose $\mathbf{s} = (s_x, s_y, s_\theta)$ with respect to O. To prevent reference-system transformation errors due to the initial error and/or changes in the 3D pose (6dof) of the laser, the system uses at every instant the perceived relative landmark poses to estimate a linear transformation that moves the raw data to a reference system with known pose in the environment. The linear transformation parameters are initialized with their theoretical values when the sensor is exactly on $\mathbf{s}$ in order to deal with possible environment symmetries. This procedure enables the system to work in a new environment without needing any calibration to determine the laser's pose.

Every time step, the laser sensor performs a scan consisting of an array of measurements, $\mathbf{m} = (\mathbf{m}_1, \ldots, \mathbf{m}_N)$. The N measurements of each step are equally spaced in angle by $(2\alpha)/N$. For the i^{th} measurement, $\mathbf{m}_i = (r_i, \theta_i, I_i)$, the laser acquires distance (r_i), angle (θ_i) and intensity (I_i) values.

4.1 System Structure

Each laser scan is processed by the GTS using the following stages (See Fig. 1):

a. Segmentation Process
In this stage, raw measurements are grouped into object candidates. A criterion based on the Euclidean distance between two consecutive points is used

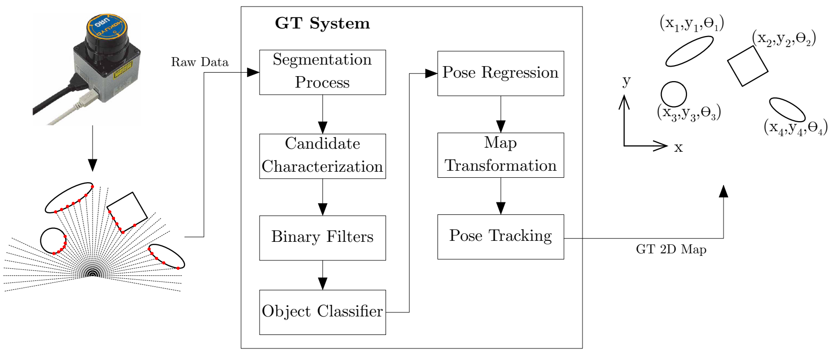


Fig. 1. System Structure Block Diagram

to determine whether these points belong to the same object or not. The output of the process consists of an array $\{S_j\}$ of object candidates, each of them containing a group $\{\mathbf{m}_{j,i}\}$ of M_j measurements. Fig. 2 shows an example of the result of the described process.

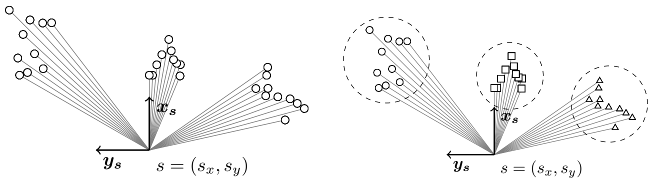


Fig. 2. Measured Data Segmentation

For each object candidate, an *object reference system* (ORS) is established. The origin, $\mathbf{o}'_j = (x'_j, y'_j, \theta'_j)$, of the ORS of the j^{th} object candidate with respect to $\mathbf{s}$ is located in $(\overline{r}_j \cos \overline{\theta}_j, \overline{r}_j \sin \overline{\theta}_j, \overline{\theta}_j)$. With,

$$\overline{r}_j = \frac{\max r_{j,i} - \min r_{j,i}}{2}, \overline{\theta}_j = \frac{\max \theta_{j,i} - \min \theta_{j,i}}{2} \tag{8}$$

Additionally, the ORS-relative position of each measurement is calculated:

$$\begin{pmatrix} x'_{j,i} \\ y'_{j,i} \end{pmatrix} = Rot\left(-\overline{\theta}\right) \begin{pmatrix} r_{j,i}\text{cos}\theta_{j,i} - \overline{r}\text{cos}\overline{\theta} \\ r_{j,i}\text{sin}\theta_{j,i} - \overline{r}\text{sin}\overline{\theta} \end{pmatrix}, Rot(\theta) = \begin{pmatrix} \text{cos}\theta & -\text{sin}\theta \\ \text{sin}\theta & \text{cos}\theta \end{pmatrix} \tag{9}$$

Finally, the origin $\hat{\mathbf{o}}_j = (\hat{x}_j, \hat{y}_j, \hat{\theta}_j)$, of the ORS with respect to O is calculated using the linear transformation:

$$(\hat{x}_j, \hat{y}_j)^T = \mu_o + A\left(\left(x'_j, y'_j\right) - \mu_s\right)^T, \hat{\theta}_j = atan2\left(A_{1,0}, A_{0,0}\right) \tag{10}$$

In the first iteration, the parameters of the linear transformation are set to: $A = Rot(s'_\theta)$, $\mu'_s = (0,0)^T$ and $\mu_o = (s'_x, s'_y)$, with $\mathbf{s}' = (s'_x, s'_y, s'_\theta)$ an initial not-necessarily-accurate guess of **s** provided by the user of the system. For the following iterations, these parameters are calculated in advance in the Map Transformation stage of the previous iteration. An object candidate and its ORS, (x', y'), are shown in Fig. 3.

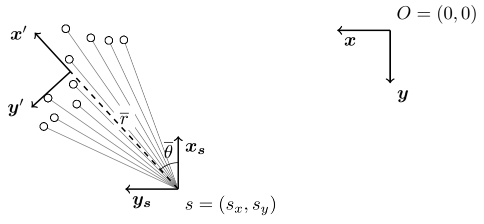


Fig. 3. (x', y') Candidate's Object Reference System

b. Candidate Characterization

This stage generates characteristics for each object candidate. For every object candidate, a characteristic vector, $\mathbf{v}_i$, is built using data from three sources:

(i) The bounding box width, w, of the points $(x'_{j,i}, y'_{j,i})$. The bounding box is a rectangular area, with center located on $(0, 0)$ relative to the ORS. Its width is $(\max y'_{j,i} - \min y'_{j,i})$ and its height is $(\max x'_{j,i} - \min x'_{j,i})$.

(ii) The distance profile vector, $d_{j,p} = \{d_{j,k}\}$. This vector has a fixed size of N_d values. Each vector component corresponds to a profile cell, and it is calculated with the following equation:

$$d_{j,k} = \frac{\sum_i^{N_c} x'_{j,i} e^{dist(k,i)}}{\sum_i^{N_c} e^{dist(k,i)}} \quad \forall k \in \{0, N_d - 1\} \tag{11}$$

(iii) The intensity profile vector, $I_p = \{I_k\}$. It consists of N_I cells, which values are obtained using the following equation:

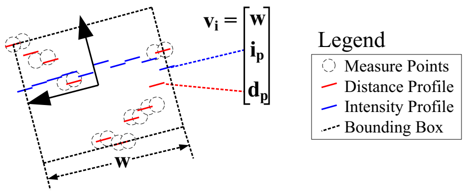
$$I_{j,k} = \frac{\sum_i^{N_c} I_{j,i} e^{dist(j,i)}}{\sum_i^{N_c} e^{dist(j,i)}} \quad \forall k \in \{0, N_I - 1\} \tag{12}$$

The expression $dist(j, i)$ is a one dimensional distance over the y' axis between profile cell j and measurement i.

The array of characteristic vectors $\{\mathbf{v}_j\}$ is the output of the stage. The characteristic vector (with $N_d = N_I = 10$) of an example object candidate is graphically shown on Fig. 4.

c. Binary Filters

Every object that does not fulfill a minimum heuristic condition to be considered as an object candidate is filtered. A threshold on the bounding box width is applied on every candidate. As a result, the characteristic vectors of the filtered objects are deleted from $\{\mathbf{v}_j\}$.



$\mathbf{v_i} = \begin{bmatrix} \mathbf{w} \\ \mathbf{i_p} \\ \mathbf{d_p} \end{bmatrix}$
Legend
Measure Points
Distance Profile
Intensity Profile
Bounding Box
w

Fig. 4. Characteristic Vector Generation: Points are direct measurements, red lines are distance profile values d_p and blue lines are intensity profile values i_p

d. Object Classifier

This stage classifies each object candidate into an object class. As a result, an object class, c_j, is associated to each candidate characteristic vector $\mathbf{v}_j$. For every characteristic vector in $\{\mathbf{v}_j\}$ and for every object class, c_k, the probability, $p_{j,k}$, of the jth object candidate belonging to c_k is estimated. This estimation is performed by means of the GP regression prediction (see equation 7):

$$p_{j,k} = f_k^{class}(\mathbf{v}_j) \tag{13}$$

Then the class, $c_j = \arg\max_k(p_{j,k})$, with the maximum probability is assigned to the jth candidate. If p_{j,k^*} is below a threshold, the candidate is filtered, i.e., it is deleted from both $\{\mathbf{v}_j\}$ and $\{c_j\}$.

In a prior off-line process, for every object class, k, the training samples $\left\{\left(\mathbf{x}_{k,l}^{class}, y_{k,l}^{class}\right)\right\}$ are obtained by calculating the characteristic vectors $\left\{\mathbf{v}_{k,l}^{class}\right\}$ of several known objects and making:

$$\mathbf{x}_{k,l}^{class} = \mathbf{v}_{k,l}^{class}, y_{k,l}^{class} = \begin{cases} 1 \text{ if the } l\text{th object belongs to class } k \\ 0 \text{ else} \end{cases} \tag{14}$$

Then, the GP, GP_k^{class}, associated to f_k^{class}, is trained, i.e., its hyperparameters, $\mathbf{h}_k^{class}$, are learned using equations 4 and 5.

e. Pose Regression

The ORS-relative pose, $\tilde{\mathbf{p}}_j$, of each object with characteristic vector $\mathbf{v}_j$ and object class c_j is estimated using an independent GP regression for each of the three pose components:

$$\tilde{\mathbf{p}}_j = \left(\tilde{x}_j, \tilde{y}_j, \tilde{\theta}_j\right)^T = \left(f_{c_j}^x(\mathbf{v}_j), f_{c_j}^y(\mathbf{v}_j), f_{c_j}^\theta(\mathbf{v}_j)\right)^T \tag{15}$$

Then, the obtained pose estimate is moved from the ORS to O:

$$\mathbf{p}_j = \begin{pmatrix} x_j \\ y_j \\ \theta_j \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} \hat{x}_j \\ \hat{y}_j \end{pmatrix} + \begin{pmatrix} \cos\hat{\theta}_j & -\sin\hat{\theta}_j \\ \sin\hat{\theta}_j & \cos\hat{\theta}_j \end{pmatrix} \begin{pmatrix} \tilde{x}_j \\ \tilde{y}_j \end{pmatrix} \\ \hat{\theta}_j + \tilde{\theta}_j \end{pmatrix} \tag{16}$$

Just like in the previous stage, some training procedure must be performed in a prior off-line process. For every object class, k, several training samples

$\left\{\left(\mathbf{x}^{\mathbf{P}}_{k,l}, y^{x}_{k,l}, y^{y}_{k,l}, y^{\theta}_{k,l}\right)\right\}$ are obtained by calculating the characteristic vectors $\left\{\mathbf{v}^{\mathbf{P}}_{k,l}\right\}$ of an example object put in several known poses and making:

$$\mathbf{x}^{\mathbf{P}}_{k,l} = \mathbf{v}^{\mathbf{P}}_{k,l}, y^{x}_{k,l} = \hat{p}^{x}_{k,l}, y^{y}_{k,l} = \hat{p}^{y}_{k,l}, y^{\theta}_{k,l} = \hat{p}^{\theta}_{k,l} \tag{17}$$

With $\left(\hat{p}^{x}_{k,l}, \hat{p}^{y}_{k,l}, \hat{p}^{\theta}_{k,l}\right)$ the known pose of the object when $\mathbf{x}^{\mathbf{P}}_{k,l}$ was acquired. Then, the GPs, GP^{x}_{k}, GP^{y}_{k} and GP^{θ}_{k}, associated to respectively f^{x}_{k}, f^{y}_{k} and f^{θ}_{k} are trained, i.e., its hyperparameters, $\mathbf{h}^{x}_{k}$, $\mathbf{h}^{y}_{k}$, and $\mathbf{h}^{\theta}_{k}$, are learned using equations 4 and 5.

f. Map Transformation

The detected 2D positions $\{\mathbf{l}^{s}_{i}\}$ of the N_l static landmarks and their respective 2D positions $\{\mathbf{l}^{o}_{i}\}$ in the a-priori-known map of the environment are used to recalculate the linear regression transformation for the next instant. A linear regression can cope with the effects produced by changes in the 3D pose (6dof) of the laser sensor. The means of the position sets are calculated:

$$\mu_s = \frac{\sum_i \mathbf{l}^{s}_{i}}{N_l}, \mu_o = \frac{\sum_i \mathbf{l}^{o}_{i}}{N_l} \tag{18}$$

Then $X = \left[\mathbf{l}^{s}_{0} - \mu_s, \ldots, \mathbf{l}^{s}_{N_l-1} - \mu_s\right]$ and $\mathbf{y} = \left[\mathbf{l}^{o}_{0} - \mu_o, \ldots, \mathbf{l}^{o}_{N_l-1} - \mu_o\right]^{T}$ are calculated to finally calculate A using equation 2.

g. Pose Tracking

The pose of each mobile object is tracked using a Bayesian filter that use the pose observations of each object for the corrective stage. The predictive stage consists only of an increase of the covariance matrix. When there are several objects of the same class, the minimum Mahalanobis distance is used as a criterion for matching the observations with the tracked objects. The Bayesian filter currently implemented for pose tracking is an *extended Kalman filter* (EKF). The output of this stage is the GTD provided by the presented system.

The GTS is designed for tracking multiple objects, although occlusions between moving objects may lead to estimation errors. Extra laser sensors must be placed for dealing with occlusions in a multiple robot environment. The experimental section evaluates GTS estimation error for one mobile object.

5 Results

The system was implemented on a Robocup SPL environment, using goal posts as landmarks and Nao Robots as mobile objects. The used Laser sensor was a Hokuyo URG-04LX, this sensor has $4[m]$ distance range and 240° of FOV.

The object classifier was trained using characteristic vector examples of goal posts and robot, with their corresponding targets. The tripod height was set to allow the laser to aim at the body extension, two centimeters below the shoulders of the robot. At least three landmarks are always detected by the system allowing the map transformation stage to be calculated on every method iteration. The GP pose regression training data base was obtained using the experimental setup on Fig. 5.

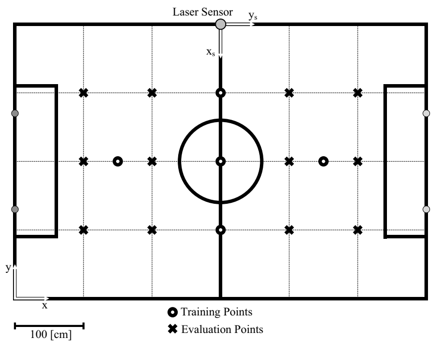


Fig. 5. Experiment setup

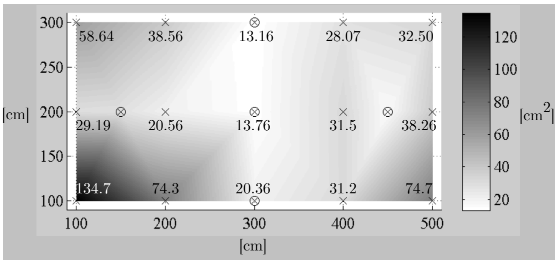
For obtaining a quantitative evaluation of the proposed GTS, a static robot was placed on several arbitrarily defined positions, which are specified on Fig. 5 as circles on the field. For each position, all the orientations spaced by 45° where tested. The pose estimation error was calculated for several samples on each testing pose.

The mean square error in the estimation of the position and angle are respectively plotted in Fig 6(a) and 6(b), showing the values of the mean square error on each evaluated position. The results show a low error on the position and orientation estimation near training positions. Nevertheless, the evaluated positions $(100, 100)$ and $(500, 100)$ show the biggest error because they have the longest distance to the laser sensor located on $(300, 400)$. The error increases at a larger distance because less sampling points are available to calculate the distance and the intensity profiles, worsening the performance of the regressor.

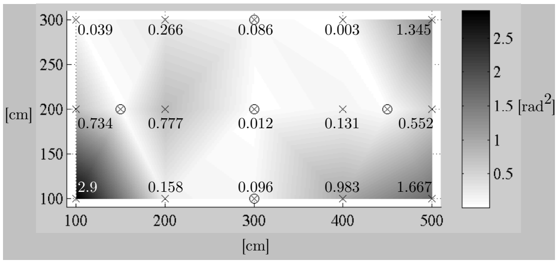
Although no quantitative results are presented for a dynamic case, the system ability to work in real time and in a dynamic situation was tested by using the provided GTD as a feedback to an external pose controller that allows the robot to follow an arbitrarily defined trajectory on the environment. Fig. 7 shows schematically the experimental setup for this test. In all the performed tests, the robot was able to follow any arbitrary trajectory that was required. A video of the robot following a desired trajectory can be found at http://www.robocup.cl/PapersMedia/GTS.mov

6 Conclusions

A method for generating a laser-based ground-truth map has been established. The results show that a laser sensor in conjunction with the proposed methodology are able to detect an object and then correctly classify it and characterize



(a)



(b)

Fig. 6. (a) Position Mean Squared Error (b) Angle Mean Squared Error

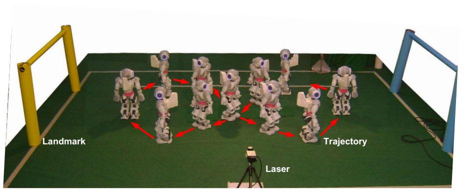


Fig. 7. Trajectory Experiment on Field

its pose in a known environment. Like in camera-based GTSs, the size of the environment is an issue. It has been shown that a laser-based GTS is able to generate controlling signals to handle a blind robot movement allowing it to execute trajectories, eliminating accumulative odometry errors. Standard deviation of the GTS is much smaller than usual levels of error in odometry and perceptions of an autonomous mobile robot.

As a future work, we expect to test the scalability of the system to larger environments by adding extra laser sensors. Additionally, it could be interesting to analyze a possible multi-sensor (camera and laser) data fusion for applications where the accuracy is very important. In order to eliminate geometric plane movement constrain, the use of a 3D camera instead of a laser sensor could be studied. In order to be able to track a robot soccer match with several robots in the field the system should be enhanced by adding extra sensors and probably finding a way of identifying the robots, such as reflective materials. Finally, testing a particle filter in the Bayesian filter stage may reduce error due to impulsive noise on pose estimation.

References

1. Arenas, M., Ruiz-del-Solar, J., Norambuena, S., Cubillos, S.: A Robot Referee for Robot Soccer. In: Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C. (eds.) RoboCup 2008. LNCS (LNAI), vol. 5399, pp. 426–438. Springer, Heidelberg (2009)
2. Baltes, J., Anderson, J.: Interpolation Methods for Global Vision Systems. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 434–442. Springer, Heidelberg (2005)
3. Egorova, A., Simon, M., Wiesel, F., Gloye, A., Rojas, R.: Plug and Play: Fast Automatic Geometry and Color Calibration for Cameras Tracking Robots. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 394–401. Springer, Heidelberg (2005)
4. Hayashi, Y., Tohyama, S., Fujiyoshi, H.: Mosaic-Based Global Vision System for Small Size Robot League. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, pp. 593–601. Springer, Heidelberg (2006)
5. Masutani, Y., Tanaka, Y., Shigeta, T., Miyazaki, F.: Pseudo-local Vision System Using Ceiling Camera for Small Multi-robot Platforms. In: Polani, D., Browning, B., Bonarini, A., Yoshida, K. (eds.) RoboCup 2003. LNCS (LNAI), vol. 3020, pp. 510–517. Springer, Heidelberg (2004)
6. Naruse, T., Masutani, Y., Mitsunaga, N., Nagasaka, Y., Fujii, T., Watanabe, M., Nakagawa, Y., Naito, O.: SSL-Humanoid RoboCup Soccer Using Humanoid Robots under the Global Vision. In: Ruiz-del-Solar, J., Chown, E., Ploeger, P.G. (eds.) RoboCup 2010. LNCS, vol. 6556, pp. 60–71. Springer, Heidelberg (2010)
7. Niemüller, T., Ferrein, A., Eckel, G., Pirro, D., Podbregar, P., Kellner, T., Rath, C., Steinbauer, G.: Providing Ground-Truth Data for the Nao Robot Platform. In: Ruiz-del-Solar, J., Chown, E., Ploeger, P.G. (eds.) RoboCup 2010. LNCS (LNAI), vol. 6556, pp. 133–144. Springer, Heidelberg (2010)
8. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press (2005)
9. Röfer, T., Jüngel, M.: Fast and robust edge-based localization in the Sony four-legged robot league (2004)
10. Sheh, R., West, G.: Visual Tracking and Localization of a Small Domestic Robot. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 410–417. Springer, Heidelberg (2005)
11. Ruiz-del-Solar, J., Loncomilla, P., Vallejos, P.A.: An Automated Refereeing and Analysis Tool for the Four-Legged League. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 206–218. Springer, Heidelberg (2007)
12. Umemura, S., Murakami, K., Naruse, T.: Orientation Extraction and Identification of the Opponent Robots in RoboCup Small-Size League. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 395–401. Springer, Heidelberg (2007)
13. Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: SSL-Vision: The Shared Vision System for the RoboCup Small Size League. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S. (eds.) RoboCup 2009. LNCS (LNAI), vol. 5949, pp. 425–436. Springer, Heidelberg (2010)