Experiments in self-calibration of an autonomous mobile manipulator

A. Sánchez, P. Núñez, L. Manso, P. Bustos

***Abstract*—Current autonomous mobile manipulators are very complex machines built with dozens of motors and sensors connected through feedback loops. On top of this first layer of real-time controllers, subsequent levels of software modules interact in many ways to create increasingly sophisticated behaviors. One of the initial requirements for all these elements to work properly, is that the robot be properly calibrated. The size of each link and the relative positions among them have to be estimated. Also, many sensors have intrinsic parameters that have also to be estimated in order to relate the incoming data to a common reference system. In this paper, this crucial problem has been studied for a new humanoid robot named Loki. Loki has been built in the Robotics and Artificial Vision (RoboLab) laboratory for research in social and service robotics. Two different optimization approaches have been explored, the *L*evenverg-Maquardt gradient-descent procedure and a *M*arkov Chain Monte Carlo Simulated Annealing algorithm. Both methods are compared under this high dimension self-calibration problem and the results are analyzed and compared. Finally, several strategies to continue research in this area and to achieve fully autonomous calibration and re-calibration procedures are described.**

***Index Terms*—Robotics, Mobile manipulators, self-calibration**

I. Introduction

Autonomous Mobile Manipulators (AMM's) are complex mechatronic devices composed of dozens of motors and sensors, controlled by networks of distributed processes running on multi-core hardware. The tasks performed by these robots usually require sensor fusion or vision-based manipulation and, when different parts of the robot are coordinated to achieve a certain goal, they must be calibrated. Calibration in this context means that their inter-relationships must be well-characterized. Sensors must be calibrated to the manipulators and, also, sensors must be calibrated to each other. In vision-guided manipulation tasks, for instance, the robot typically has to grasp a target that has been localized in the visual reference system. To succeed, a 3D point such as the tip of the right hand finger seen by the left camera as pixel $p(i, j)$, must be very close to the pixel value $q(i, j)$ obtained after computing, using direct kinematics, the coordinates of the finger tip in the left camera reference system and projecting them through the camera model. If both values are similar enough, the arm will be commanded to a location close to the target and the task will be completed. An analogous situation can be applied to the relationship between two sensors, like a camera and a laser, or between two cameras placed on different effectors of the body. Our long term goal is to develop self-calibration behaviors for AMMs that could be executed opportunistically causing a minimal interference with other running behaviors. In reaching this final objective, we have started by studying self-calibration procedures using the complete kinematic structure of the robot Loki. The basic idea is to perturb some initial robot configuration and to recover these initial values using optimization methods. To this end, the robot shows himself -to its sensors- a known object in different poses to generate a rich set of calibration data. Due to the non-linear nature of the optimization problem and to the physical constraints that the final solution must meet -i.e. limits in how much resizing is admitted for metal pieces-, both a gradient descent method and a stochastic method have been evaluated. The complexity of the robot Loki, see Section II, suggests to begin this self-calibration procedure with a wide set of simulation experiments These experiments will cover the comparison of stochastic versus gradient optimization methods, the number of parameters that can be simultaneously adjusted, the grouping of sets of parameters according to anatomical localization -i.e. hand, arm, head, cameras-, the nature of these parameters -i.e. mechanical offsets, gear reduction ratios, encoder uncertainty, optical parameters- and the initial knowledge about the uncertainties in the mechanical structure and sensor parameters. The rest of the paper is organized as follows: Section II describes the robot Loki. In Section III previous approaches to the problem will be introduced and compared to our own approach, in Section IV the forward kinematics of the robot will be briefly described to introduce the mathematical notation used in the other sections, Section V explains how the optimization problem is approached, in Section VI all the experiments are described and the results analyzed, and finally, the paper is closed with some conclusions and a description of future work.

A. Sánchez, P. Núñez, L. Manso and P. Bustos are with RoboLab, University of Extremadura.
E-mail: agustind@unex.es, pbustos@unex.es

II. Loki the robot

Loki is a robot built as a collaboration between the SIMD group at the Castilla-La Mancha University and RoboLab at Extremadura University. Figure 1 shows an early one-arm version of the robot. This robot has been built as an advanced social robotics research platform. The mobile base of Loki holds batteries, energy management electronics, local controllers and a high-end dual-socket Xeon board that will soon receive an NVIDIA GTX 690 featuring 3072 CUDA cores. Standing on it, a rigid column supports the torso which holds two arms and a expressive head. Each arm has 7 degrees-of-freedom (DOF) in anthropomorphic configuration built with a combination of Schunck motors in the upper arm and a

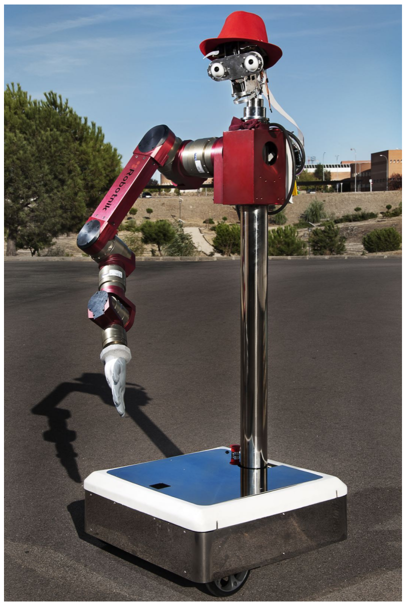

Fig. 1: Loki, a humanoid social robot build by RoboLab

custom-made 3 DOF forearm, a 6 DOF torque-force sensor in the wrist and a 4 DOF BH8 Barret hand with three fingers. The head is connected to the torso by a 4 DOF neck and has two orientable Point Grey Firewire cameras, an ASUS Xtion range sensor and 5MP Flea3 camera. It also features an articulated yaw and eyebrows for synthesis of facial expressions and microphones and a speaker for voice communication. The total number of DOF is 37. The control of all these elements is performed by a set of components created with the open source robotics framework *R*oboComp [1]. The robot currently can execute a basic repertoire of navigation, manipulation, and interaction behaviors that are under active development in other research projects [2].

III. Related work

Calibration of robot kinematics and its sensors has always been an active area of research and there are many contributions dealing with this topic. A crude division can be made attending to the object of interest. For example, in the camera calibration problem the goal is to compute the intrinsic and extrinsic parameters of the sensor using some knowledge of the environment [3] [4] [5] [6]. Stereo and multi-camera settings also fall within this category [7]. A second block can be assigned to calibration of kinematics parameters for robotics manipulators [8] [9] [10] [11] [12]. The combination of sensors attached to kinematic structures is a more complex problem. There are many works about the so called hand-eye calibration problem that deals with the computation of the relative position and orientation between the robot gripper and a camera mounted rigidly on the gripper [13] [14] [15] [16] [17] [18]. When the robot holds its own calibration object there are new restrictions that can be added to the original camera calibration problem. Moreover, the selection of more informative poses becomes an interesting problem by itself. The combination of optimization and pose selection leads to the exploration-exploitation trade-off. Finally, the most general setups are those in which multiple sensors are calibrated simultaneously [19] [20]. The work of Pradeep et al. [20] on a self-calibration method for the Willow Garage's PR2 robot [21] is a milestone in this specific view of the problem focused on robot autonomy. They propose a probabilistic formulation of a measurement model that allows to express measures from sensors with different uncertainty parameters and localized on distant kinematic structures in a common scale. This scale is obtained by propagating each sensor's uncertainty through its kinematic chain up to the common point where the comparison is made. At the cost of computing large Jacobians, measures are weighted by their accumulated uncertainty before entering the final error computation. However, the paper barely explores the problem of non-convergence situations and does not deal with strategies for on-line and opportunistic implementations. Building on these recent ideas and also on the background provided by the field, we have started a research line with the goal of building self-calibration behaviors for the Loki AMM, that can be both robust and efficient while running opportunistically with the rest of its control architecture. To achieve this goal an initial evaluation of global and gradient optimization methods will be performed with a set of experiments designed for this specific context.

IV. Kinematic model

The robot Loki has an anthropomorphic configuration that is schematically reproduced in Figure 2. Each of the nodes in the graph can represent more than one joint. For example, the shoulder complex is composed of three motors that define three DOF's. Each joint is assigned a reference system that

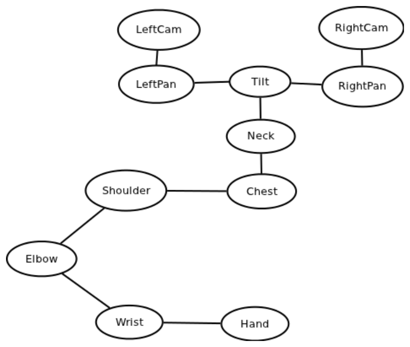

Fig. 2: Schematic kinematics of robot Loki

is represented as a 4x4 matrix encoding a 3D rotation and translation vector. A point in 3D space can be referenced with respect to two different systems, as is shown in Figure 3. The point P is represented in reference system B as the vector P_B, and in reference system A as the vector P_A. It is easy to change from one reference system to another if the last one is expressed as a rotation and translation from the former one. The resulting function has six parameters, three angles for the rotations and three coordinates for the translation vector:

$$H = f(T_x, T_y, T_z, \alpha, \beta, \gamma) \tag{1}$$

where T_x, T_y, T_z are the translation components and α, β, γ, are the rotation angles with respect to to axis x, y, z.

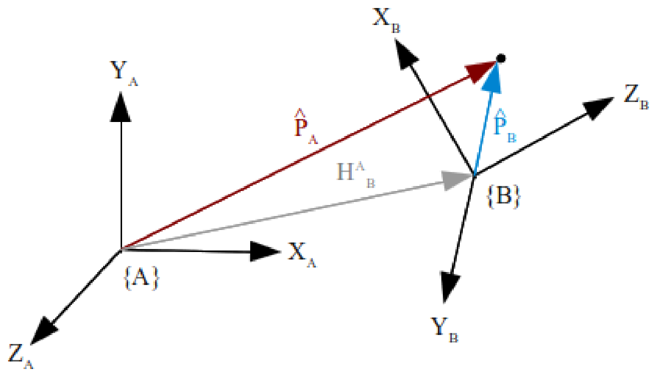

Fig. 3: Change of reference system

This function can be written more compactly in matrix form as:

$$H_i = \begin{bmatrix} c_{2i}c_{3i} & -c_{2i}s_{3i} & s_{2i} & T_{xi} \\ c_{1i}s_{3i} + s_{1i}s_{2i}c_{3i} & c_{1i}c_{3i} - s_{1i}s_{2i}s_{3i} & -s_{1i}c_{2i} & T_{yi} \\ s_{1i}s_{3i} - c_{1i}s_{2i}c_{3i} & s_{1i}c_{3i} + c_{1i}s_{2i}s_{3i} & c_{1i}c_{2i} & T_{zi} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where: $s_{1i} = sin(\alpha_i)$, $c_{1i} = cos(\alpha_i)$, $s_{2i} = sin(\beta_i)$, $c_{2i} = cos(\beta_i)$, $s_{3i} = sin(\gamma_i)$, $c_{3i} = cos(\gamma_i)$
This matrix allows us to transform a point in reference system B to reference system A. This is achieved multiplying the vector expressed in B times this matrix:

$$\begin{bmatrix} X_A \\ Y_A \\ Z_A \\ 1 \end{bmatrix} = \mathrm{H}_B^A \begin{bmatrix} X_B \\ Y_B \\ Z_B \\ 1 \end{bmatrix} \tag{3}$$

where X_B, Y_B, Z_B are the 3D coordinates of the point in B and X_A, Y_A, Z_A are the final coordinates in reference system A.
This operation can be chained to transform directly between two reference systems A and C linked by B. Figure 4 shows this situation. The resulting matrix is computed as:

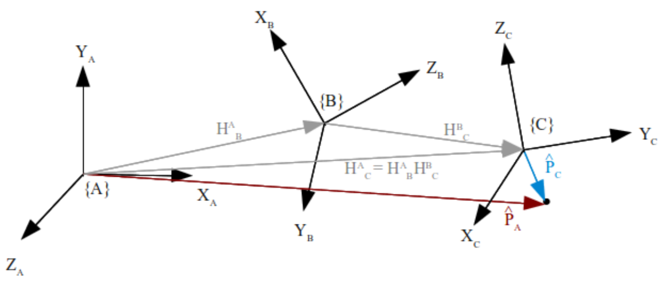

Fig. 4: Chainning transformations

$$\begin{bmatrix} X_A \\ Y_A \\ Z_A \\ 1 \end{bmatrix} = \mathrm{H}_B^A \mathrm{H}_C^B \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} \tag{4}$$

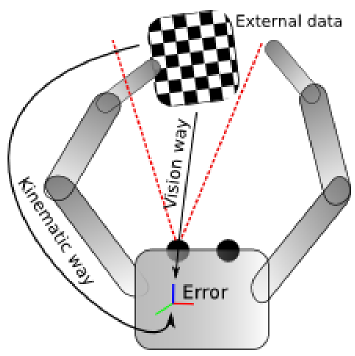

Fig. 5: Robot configuration for self-calibration

In the case of the robot Loki, to get from the hand to the eye it is necessary to go through 15 intermediate reference systems,

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} = \mathrm{H}_2^1 \mathrm{H}_3^2 ... \mathrm{H}_{14}^{13} \mathrm{H}_{15}^{14} \begin{bmatrix} X_{15} \\ Y_{15} \\ Z_{15} \\ 1 \end{bmatrix} \tag{5}$$

that makes a total of 90 parameters. To complete the arm-eyes kinematic chain, the cameras intrinsic parameters matrices have to be appended adding at least 6 more parameters, 2 focals and 2 optical centers. The translation vector of each transformation matrix encodes the length and positions of the *bones* making up the arms, torso and neck. These values are known from fabrication and assembly data but can carry a variable amount of uncertainty. Knowledge of this uncertainty will be very useful in initializing and bounding the optimization procedures.

V. Problem Statement

The initial self-calibration procedure starts by placing a calibrated checkerboard with 100 corners in the hand of the robot and making it change to different poses recording the image captured by the cameras. Being a simulation, for each pose the set of 3D corners in the board is projected through the cameras to obtain a list of pixel coordinates. Also, the 3D coordinates of the corners in the checkerboard reference system are propagated through the kinematic equations up to the camera. Once they are transformed to the camera reference system, they are projected to obtain the second set of pixel coordinates. Without additional perturbations, both sets are exactly the same. We now proceed to change a subset of the parameters in the kinematic chain and re-project the set of 3D corners to obtain a new set of misaligned pixel values. Figure 5 shows a schema of the experimental setup. A quadratic error expression can be readily computed as:

$$E = \sum_{n=1}^{N} (p_{c_n} - f(p_{k_n}, \mathrm{H}_2^1 \mathrm{H}_3^2 ... \mathrm{H}_{14}^{13} \mathrm{H}_{15}^{14}, P_c))^2 \tag{6}$$

where p_{c_n} is the point n seen by the camera and f is the kinematic function that transforms this point from the checkerboard reference system (p_{k_n}) to the camera. P_c is the camera projection matrix. We want to find the minimum of this error function with respect to a subset of the parameters H_i^j that define the robot kinematics. The complete set of parameters defines a non-linear minimization problem over 96 variables. We will approach the problem by selecting blocks of parameters. Note that this approach does not use inverse kinematics so there is no issues with ambiguous end-configurations. Instead, we sample the direct kinematics space searching form admissible solutions. Two different optimization methods will be used, the gradient descent Levenberg-Maquardt (LM) algorithm and the stochastic simulated annealing (SA) method.

The Levenberg-Marquardt method is a standard technique used to solve nonlinear least squares problems. It is actually a combination of two minimization methods: the gradient descent method and the Gauss-Newton method. In the gradient descent method, the sum of the squared errors is reduced by updating the parameters in the direction of the greatest reduction of the least squares objective. In the Gauss-Newton method, the sum of the squared errors is reduced by assuming that the least squares function is locally quadratic. The Levenberg-Marquardt method acts more like a gradient-descent method when the parameters are far from their optimal value, and acts more like the Gauss-Newton method when the parameters are close to their optimal value [22] [23] [24]. We use the available Matlab implementation.

Simulated Annealing is a variation of standard Markov Chain Monte Carlo algorithms designed to search for global minima in arbitrary surfaces, instead of approximating such surfaces. The basic idea of MCMC algorithms is to define a proposal function that provides samples from the function to minimize following a certain probability distribution and a admissibility criterion to accept or reject the sample. The difference with respect to the standard Metropolis-Hastings algorithm is that there is a so called *cooling schedule* that decreases the probability of accepting a solution with higher error than the current one. The following pseudo-code shows the idea:

```
1) Initialise $x^0$ and set $T_0 = 1$
2) For i=0 to N-1
    • Sample $u \sim \upsilon_{0,1}$
    • Sample $x^* \sim q(x^*|x^{(i)})$
    • If $u < A(x^{(i)}, x^*) = min\left\{1, \frac{p^{\frac{1}{T_i}}x^* q(x^{(i)}|x^*)}{p^{\frac{1}{T_i}}x^{(i)} q(x^*|x^{(i)})}\right\}$
          $x^{i+1} = x^*$
      else
          $x^{i+1} = x^{(i)}$
    • Set $T_{i+1}$ according to a chosen cooling schedule
```

where u is a sample from an uniform distribution, q is the proposal distribution that generates a new sample of the parameters of the function to be optimized, given the last accepted set of values, A is the acceptance function and $p(x)$ is the value of the optimized function for the set of parameters x [25][26][27]. The probability of accepting a new sample follows a decreasing cooling schedule with $lim_{i \to \infty} T_i = 0$. Levenberg-Marquardt is expected to obtain a much lower error for a well-formed convex error landscape. However, as Table I shows, there are situations in which no convergence is attained. Also, it is also possible to obtain low error solutions with physically unfeasible values for the parameters -i.e. elongated links or unrealistic offsets between joints. On the other hand, Simulated Annealing can find global minima but may show very low convergence rates in flat error regions. Table I shows this kind of situation for an experiment involving 32 free parameters and visual data obtained from 5 different robot configurations, although only *Pose 1* is shown in the Table for clarity. *Perturbation* is the deviation introduced to the initial value of the free parameters. Columns *LM* and *SA* show the final error for each method. The *SA+LM* shows the final error after applying both methods in sequence. Later, in Experiment 7, the results shown in the table are further discussed. The goal of the experimental work done here is to characterize the behavior of both methods in this specific calibration problem, as a prior step to the design of autonomous and opportunistic self-calibration procedures that will form part of Loki repertoire of behaviors.

TABLE I: Experiment showing poor convergence results of individual methods and low final error when both methods are combined.

Param.	Pose 1	Toler.	Perturbation	LM	SA	SA+LM
1	-0,7804	0,02	0,005	1370,4	-0,0049	-0,0011
2	0,0050	0,02	0,005	-2,4	0,0003	-0,0050
3	0,0050	0,02	0,005	11,4	0,0040	-0,0050
4	1,5758	0,02	0,005	-17,6	-0,0013	-0,0050
5	0,0125	0,05	0,0125	3,3	0,0225	-0,0125
6	0,7500	3	0,75	4348,6	0,5346	-0,7500
7	1,0597	0,05	0,0125	6,5	-0,0243	-0,0125
8	0,0125	0,05	0,0125	0,3	0,0062	-0,0125
9	0,0125	0,05	0,0125	-2956,5	0,0153	-0,0236
10	0,7500	3	0,75	30726,4	1,7080	0,9795
11	0,0125	0,05	0,0125	-6105,1	-0,0100	-0,0224
12	0,0125	0,05	0,0125	98,3	-0,0037	-0,0126
13	0,5000	2	0,5	6657,2	0,1549	0,1588
14	0,2500	1	0,25	134231,5	-0,3654	-0,1380
15	0,2500	1	0,25	121379,8	0,4714	0,2012
16	1,0000	4	1	20523,8	0,6005	0,5942
17	0,2500	1	0,25	4336,4	-0,3131	-0,5424
18	0,2500	1	0,25	7010,4	0,1844	0,9129
19	0,2500	1	0,25	-14589,0	0,5305	0,5421
20	0,2500	1	0,25	0,0	0,1304	0,1304
21	0,2500	1	0,25	-1056,5	0,0901	-0,2500
22	0,2500	1	0,25	-18760,4	-0,0738	-0,5246
23	0,2500	1	0,25	-7381,2	0,0997	-0,2499
24	0,2500	1	0,25	21409,6	0,2144	0,3474
25	0,2500	1	0,25	2818,3	-0,3209	-0,5424
26	0,2500	1	0,25	-11007,1	-0,0339	-0,2500
27	0,2500	1	0,25	-13568,3	-0,0142	-0,2499
28	0,2500	1	0,25	4509,7	0,4805	0,3475
29	0,0050	0,02	0,005	1372,9	-0,0084	-0,0018
30	0,3750	1,5	0,375	32843,2	-0,1445	-0,8726
31	0,3750	1,5	0,375	725,5	0,1929	0,2059
32	0,3750	1,5	0,375	-11157,4	0,4155	-0,0359
Initial error				8,2031E+09	8,2031E+09	8,2031E+09
Final error				**7,2587E+09**	**1,8874E+06**	**2,0657E-12**
Relative error				8,8487E-01	2,3008E-04	2,5182E-22

VI. Experiments

In order to evaluate both optimization methods a set of experiments have been designed using Matlab. In these experiments the optimal values of the free parameters have been estimated. The absolute final error and a relative error have been used for comparing the outcome of each experiment. The relative error (R_e) is defined as:

$$R_e = \frac{E_c}{E_{bc}} \tag{7}$$

where E_c and E_{bc} represent the final error after calibrating the robot and the initial error before calibrating, respectively. Table II shows the subset of 32 parameters used in the experiments. In the next subsections a more detailed description about each experiments is provided.

TABLE II: List of the different calibration parameters used in the experiments.

Number of parameters	Description
Test 1	
2	shoulder rotation over the X axis shoulder rotation over the Y axis
Test 2	Test 1 plus
4	shoulder rotation over the Z axis elbow rotation over the Z axis
Test 3 8	Test 2 plus wrist rotation over the x-axis wrist rotation over the y-axis wrist rotation over the z-axis wrist translation over the z-axis
Test 4 16	Test 3 plus neck rotation over the x-axis neck rotation over the y-axis neck rotation over the x-axis neck translation over the z-axis Motor1 encoder (eye) Motor2 encoder (eye) Motor3 encoder (eye) radio (eye)
Test 5 32	Test 4 plus Motor1 (shoulder) translation over the x-axis Motor1 (shoulder) translation over the y-axis Motor1 (shoulder) translation over the z-axis Motor1 (shoulder) rotation over the x-axis Motor2 (shoulder) translation over the x-axis Motor2 (shoulder) translation over the y-axis Motor2 (shoulder) translation over the z-axis Motor3 (shoulder) translation over the x-axis Motor3 (shoulder) translation over the y-axis Motor3 (shoulder) translation over the z-axis Motor (eldow) translation over the x-axis Motor (eldow) translation over the y-axis Motor (eldow) translation over the z-axis Traslation (eye) over the x-axis Traslation (eye) over the y-axis Traslation (eye) over the z-axis

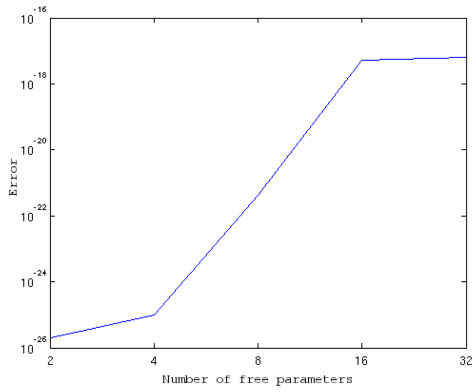

Fig. 6: Gradient descent with varying number of free parameters

A. *Experiment 1. Gradient descent with varying number of free parameters*

In this experiment we test the behavior of the LM method when the number of free parameters is increased. The values used are 2, 4, 8, 16 and 32, out of the 92 total number of parameters, see Table II. The rest of the parameters where kept to their initial value. Figure 6 shows the results with the error in each case. The curve shows an expected increase in the error with the number of free parameters.

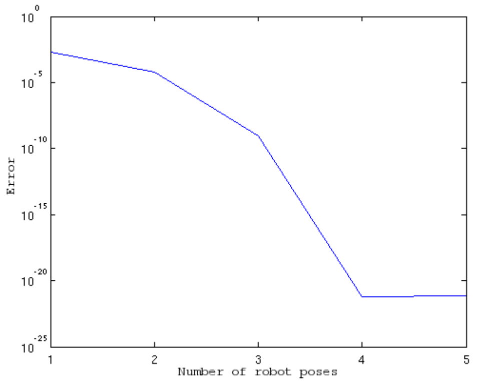

Fig. 7: Gradient descent with varying number of robot poses

B. *Experiment 2. Gradient descent with varying number of robot poses*

This next set of experiments with the LM method show the variation of the error when the number of robot poses increases. Each pose is a different set of initial values for the joints of the robot and also a different set of pixels corresponding to the checkerboard corners. The error function is the sum of errors for each pose:

$$E_T = \sum_{p=1}^{P}(E_p) \tag{8}$$

where E_p is the error in pose p given by the equation 6. Five different poses have been used with 32 free parameters in each one. To find out how the number of poses affects the final error we start with one pose and obtain a set of values. Then, these values are injected in the remaining poses to compute the error in each pose due to the calibration obtained initially. Afterward, the same procedure is repeated using two poses for calibration and so on until completing the five poses. Figure 7 shows the evolution of the relative error. The error decreases with the number of poses, showing that a better sampling of the error space leads allows the method to find an optimum with very low final error.

C. *Experiment 3. Simulated Annealing varying the sampling distributions of the free parameters*

This experiment shows the behavior of the SA method in relation to the sampling distribution of the free parameters. An important advantage of the sampling approaches is that the sampling distribution can include prior knowledge about the variation range of the free parameters. As proposal distribution we use a set of independent normal gaussians centered at the last accepted point. Their variances are computed from previous knowledge of the mechanical structure and of the total admitted tolerance. For each parameter the proposal distribution q is defined as,

$$q(x_i^*|x^{(i)}) \sim N\left(p_i, \kappa t_i\right) \tag{9}$$

where p_i is the last accepted value of parameter i, κ is a factor and t_i is the tolerance of parameter i. For each value of

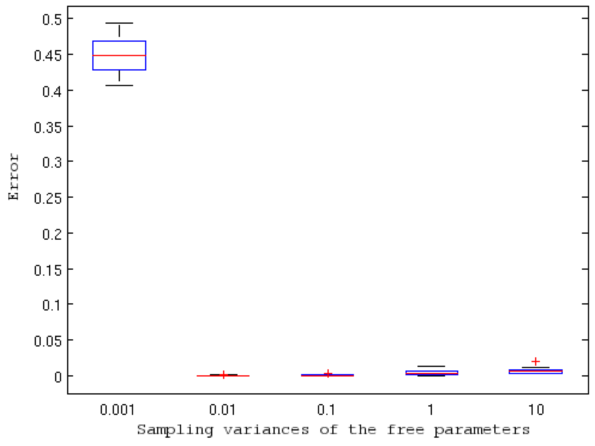

Fig. 8: Error in Simulated Annealing for different sampling variances of the free parameters

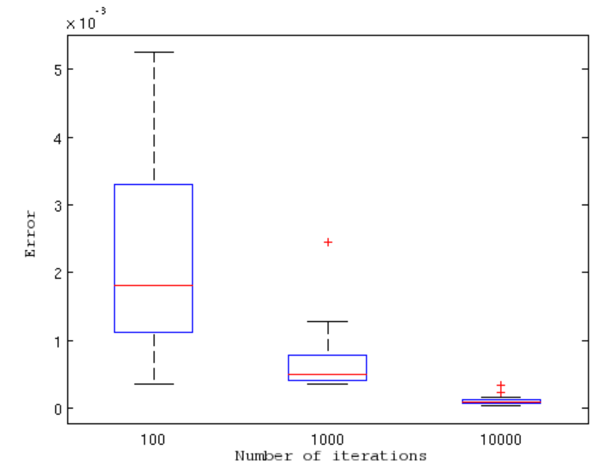

Fig. 9: Error in Simulated Annealing for different number of iterations

the variance, 20 different runs have been done to compute the mean and variance of the error. There were 32 free parameters and only one pose. The variance range goes from $\kappa = 0.001$ to $\kappa = 10$. As Figure 8 shows, the error increases with the variance value but if the variance is very small the error increases very quickly. The reason is that with too small perturbations the method can't reach the global minimum.

D. Experiment 4. Simulated Annealing varying the number of running iterations

The other important parameter that we want to test for the SA algorithm in the self-calibration problem is the number of iterations. Three experiments were performed with 100, 1.000 and 10.000 epochs. As expected, the error decreases when the number of iterations increases. Figure 9 shows the results of the experiment.

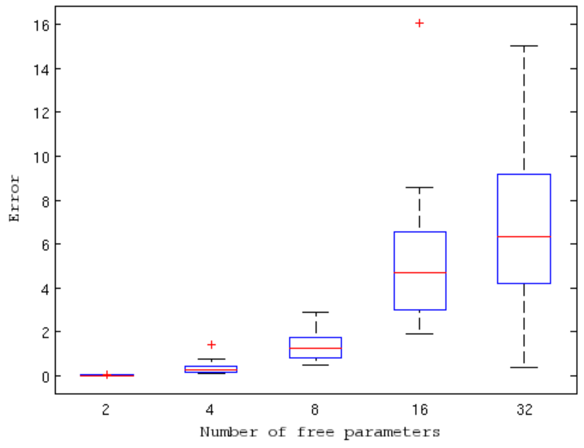

Fig. 10: Simulated Annealing varying the number of free parameters

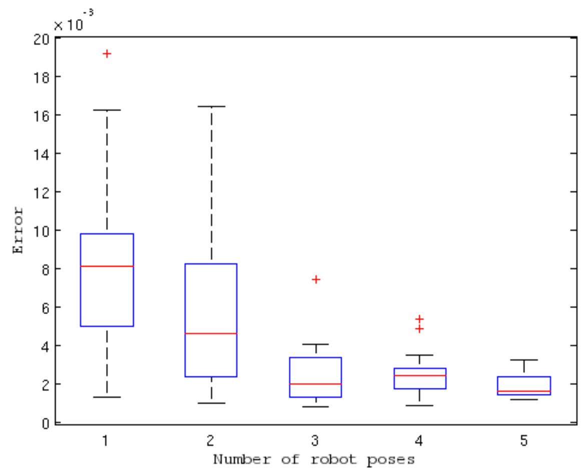

Fig. 11: Error in Simulated Annealing with different number of robot poses

E. Experiment 5. Simulated Annealing varying the number of free parameters

We also have tested the SA method with respect to the number of free parameters, with values 2, 4, 8, 16 and 32. Figure 10 shows the evolution of the error in this context. The results are similar to the LM method.

F. Experiment 6. Simulated Annealing varying the number of poses

This experiment tests the behavior of SA with a varying number of poses, each one generating a set of calibration data. Like in the Experiment 2, 32 free parameters have been used in a number of poses going from 1 to 5 poses. Figure 11, shows how the relative error decreases with the number of poses, showing a similar behavior to the LM method.

G. Experiment 7. Combining SA and LM to avoid local minima

As the former experiments have shown the LM method achieves lower error values in many situations. However, there are cases in which LM does not converge to an usable state. In some cases, the parameter values are way out of the admissible physical range and, in others, the method gets stuck in a local minimum far from the global one. The results or this experiment were shown in Table I and indicate how a combination of both methods, based on a simple observation of the evolution of the error, can provide excellent results - 2.06E-12- in a difficult, but not unusual, calibration situation. The experiment uses 32 free parameters and 5 robot poses. Table I shows the initial value of each parameter in the first pose, the maximum tolerance admitted for each parameter, the perturbation introduced to the first pose, the values obtained with LM, with SA and, finally, the values obtained running first SA and then LM. The last row shows the error values before and after calibrating, and the relative error in both situations.

VII. Conclusions and Future Work

After this set of experiments it is clear that with good initial conditions the LM method achieves more accurate results. The problem for a self-calibrating robot is that this method may fail in some situations, returning values for the parameters out of the physically admissible range or not being able to find a low minimum of the error function. When this happens, the best action is to use the SA method to find a position close to the global minimum and restart LM from that position. SA always returns values inside the admitted tolerance and always achieves a lower error than the initial one, so this procedure can be iterated several times until convergence. This initial experiences with the optimization methods will allow us to design efficient procedures for self-calibration during real-world operation of the robot. Much work remains to be done in order to build an opportunistic behavior that can observe and wait until a propitious situation presents. This behavior should periodically evaluate the calibration state of the robot and detect situations in which known size objects are manipulated by the robot, thus capturing valuable data that can be used to re-calibrate the kinematic and sensor parameters. A new component is already being developed under RoboComp to implement and test several opportunistic self-calibration strategies. Also, we plan to extend the experiments to include additional sensors and the base odometry.

Acknowledgment

This work has been partially supported by the Spanish Ministerio de Ciencia e Innovación TIN2011-27512-C05-04, IPT-430000-2010-002 and AIB2010PT-00149 and the Junta de Extremadura projects IB10062.

References

[1] *RoboComp* http://robocomp.sourceforge.net

[2] L.J. Manso, P. Bachiller, P. Bustos, P. Nuñez, R. Cintas and L. Calderita. *RoboComp: a Tool-based Robotics Framework*. Simulation, Modeling and Programming for Autonomous Robots (SIMPAR). Pages 251-262. 2010.

[3] R. Y. Tsai, *A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses*. IEEE Journal on Robotics Automation, vol. 3, no. 4, 1987.

[4] Z. Zhang, *Flexible camera calibration by viewing a plane from unknown orientations*. International Conference on Computer Vision, 1999.

[5] J. Heikkila and O. Silven, *A four-step camera calibration procedure with implicit image correction*. IEEE Conference on Computer Vision and Pattern Recognition, 1997.

[6] J. Bouguet, *Caltech camera calibration toolbox for MATLAB*. Caltech, Tech. Rep., 2008.

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, March 2004.

[8] Charles W. Wampler, John M. Hollerbach, Tatsuo Arai, *An Implicit Loop Method For Kinematic Calibration and Its Application to Closed-Chain Mechanisms*, IEEE Transactions on Robotics and Automation, Vol 11, N^o 5, pp. 710-724. October 1995. R. Bernhardt and S. Albright, Robot calibration. Kluwer, 1993.

[9] H. Zhuang, J. Yan, and O. Masory, *Calibration of stewart platforms and other parallel manipulators by minimizing inverse kinematic residuals*. Journal of Robotics Systems, 1998.

[10] H. Zhuang, S. Motaghedi, and Z. Roth, *Robot calibration with planar constraints*. International Conference on Robotics and Automation, 1999.

[11] S. Besnard and W. Khalil, *Identifiable parameters for parallel robots kinematic calibration*. International Conference on Robotics and Automation, 2001.

[12] L. Beyer and J. Wulfsberg, *Practical robot calibration with ROSY*. Robotica, vol. 22, pp. 505-512, 2004

[13] Gin-Shu Young, Tsai-Hong Hong, Martin Herman, and Jackson C. S. Yang, *Kinematic Calibration of an Active Camera System*. IEEE, pp. 748-751. 1992.

[14] R. Horaud and F. Dornaika, *Hand-eye calibration*. International Journal of Robotics Research, vol. 14, no. 3, pp. 195–210, 1995.

[15] Mengxiang Li, *Kinematic Calibration of an Active Head-Eye System*. IEEE Transactions on Robotics and Automation, Vol 14, N^o 1, pp. 153-158, February 1998.

[16] G-Q Wei, K Arbter, and G Hirzinger. *Active self-calibration of robotic eyes and hand-eye relationships with model identication*. IEEE Trans. on Robotics and Automation, pp:158,165, 1998.

[17] K. Daniilidis, *Hand-eye calibration using dual quaternions*. International Journal of Robotics Research, 1999.

[18] N. Andreff, R. Horaud, and B. Espiau. *Robot Hand-Eye Calibration Using Structure from Motion*. Journal of Robotics Research, 20:228-248, 2001.

[19] Quoc V. Le and Andrew Y. Ng, *Joint calibration of multiple sensors*. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3651–3658. 2009.

[20] Vijay Pradeep, Kurt Konolige, Eric Berger, *Calibrating a multi-arm multi-sensor robot*. International Symposium on Experimental Robotics (ISER). 2010.

[21] PR2. http://www.willowgarage.com/pages/pr2/overview

[22] K. Levenberg. *A Method for the Solution of Certain Non-Linear Problems in Least Squares*,The Quarterly of Applied Mathematics, 2: 164-168. 1944.

[23] D.W. Marquardt. *An algorithm for least-squares estimation of nonlinear parameters*, Journal of the Society for Industrial and Applied Mathematics, 11(2):431-441, 1963.

[24] Henri Gavin. *The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems*, Department of Civil and Environmental Engineering Duke University, 2011

[25] Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. *Optimization by simulated annealing*, Science 220, 671-680. 1983

[26] Cerný, V. *Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm*, Journal of Optimization Theory and Applications 45: 41-51. 1985.

[27] C. Andrieu, N.De Freitas, A. Doucet, and M. I. Jordan. *An Introduction to MCMC for Machine Learning* Science: 5-43. 2003.