24th ABCM International Congress of Mechanical Engineering
December 3-8, 2017, Curitiba, PR, Brazil

COBEM-2017-1195
EXPERIMENTAL CALIBRATION METHOD FOR THE NAO HUMANOID ROBOT

Humberto Cascardo Demolinari
José Maurício S T Motta
Universidade de Brasília, Department of Mechanical Engineering, Brasília, Brazil
humberto.demolinari@gmail.com; jmmotta@unb.br

Abstract. *This paper explores the design of a calibration technique for the kinematic model for the NAO robot. Mobile humanoid robots usually present complex kinematic chains and dull and time-consuming off-line calibration methods. For the purpose of this paper NAO's kinematic chain is separated into smaller end-to-end chain units and calibrated individually. By using Computer Vision techniques, we propose an experiment that combined with homogeneous transformations and forward kinematics analysis can lead to a reliable calibration technique. The main result expected is an accurate calibration for the robot kinematic chain.*

Keywords: *NAO, robot calibration, humanoid robotics, modelling*

1. INTRODUCTION

Robot calibration consists of a comparison of the mathematical model previsions and real measurements, based on the robot's kinematic parameters (Kastner, *et al*., 2015). A robust model is necessary to understand the kinematic behavior of any device precisely. Control engineering techniques and identification of systems are usually applied to estimate the relevant parameters of a mechatronic system (Ogata, 1985).

A manipulator robot has a fixed base that serves as reference for calibration models. However, humanoid robot calibration presents challenges, as they have no fixed references (Yamane, 2011). Parameter identification and equations solving are some of the problems in mobile robot modeling. In this paper, we address the modeling of a humanoid robot (NAO) and propose an experiment to calibrate its kinematic model.

SoftBank Robotics' (former Aldebaran Robotics) NAO is a humanoid robot designed to be functional, accessible, modular and open architecture (Gouaillier, *et al*., 2008). Aside its friendly look, NAO presents a complex open kinematic chain and 25 Degrees-of-Freedom (DoF) as shown in Fig. 1.

NAOqi is the NAO's embedded GNU/Linux Operating System (OS). It is based on Gentoo and runs all the robot programs and libraries (Softbank, 2017a). From the Python environment using the Software Development Kit (SDK) provided by SoftBank Robotics, it is possible to import the main functions and methods from NAOqi, allowing the program to access, e.g., the robot's sensors, actuators and memory. The NAOqi-Python connection is fundamental for the vision algorithm and provides a live feed of any (or both) of the robot's cameras.

For the purpose of this paper, NAO's kinematic chain is divided into five smaller kinematic chains starting in the robot´s torso and ending in each of its end effectors (feet, hands and head). The kinematic chains are separated by each of the robot limbs and further divided into its basic joint-link components and modeled using the Denavit-Hartenberg (DH) notation (Hartenberg and Denavit, 1964). DH is one of the most usual representations for kinematic models, whereas for each local coordinate frame of interest there is an associated homogeneous transformation that relates that frame to the next one (Craig, 2012). The DH parameters are: kinematic distance (l_n), link distance (r_n), joint angle (θ_n) and functional angle (α_n) as shown in Fig. 2.

Given a joint *j* with respect to its adjacent joint *i* the homogeneous transformation matrix (translation and orientation) that relates the next coordinate frame with the previous frame is described as a 4x4 matrix (Veitschegger and Wu, 1986) shown in Eq. (1). Each transformation is a multiplication of two rotations and two translations as shown in Eq. (2), where $R_k(a)$ represents a rotation around the *k* axis by an angle *a* and $A([i\ j\ k]^T)$ represents a translation defined by the vector $(i\ j\ k)^T$ from the previous to the next frame origins.

$$T_i^j = \begin{bmatrix} [X]_{3x3} & [\bar{y}]_{3x1} \\ 0 \dots 0 & 1 \end{bmatrix} \quad (1)$$

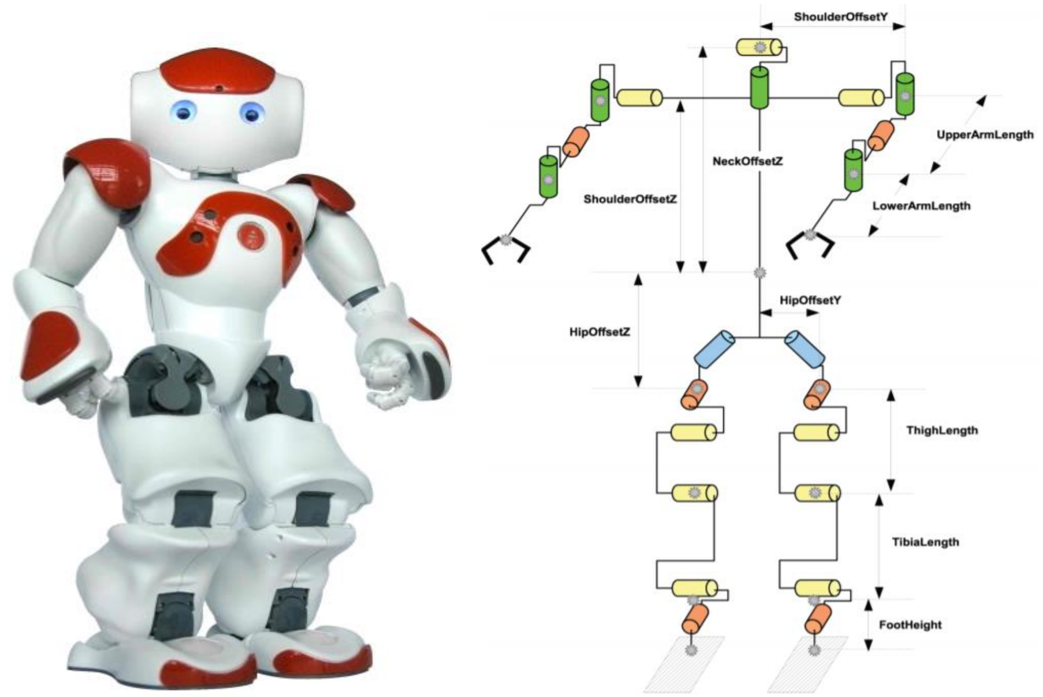

Figure 1. NAO Robot and its kinematic representation (Gouaillier, *et al*., 2008). Wrist joints are not represented

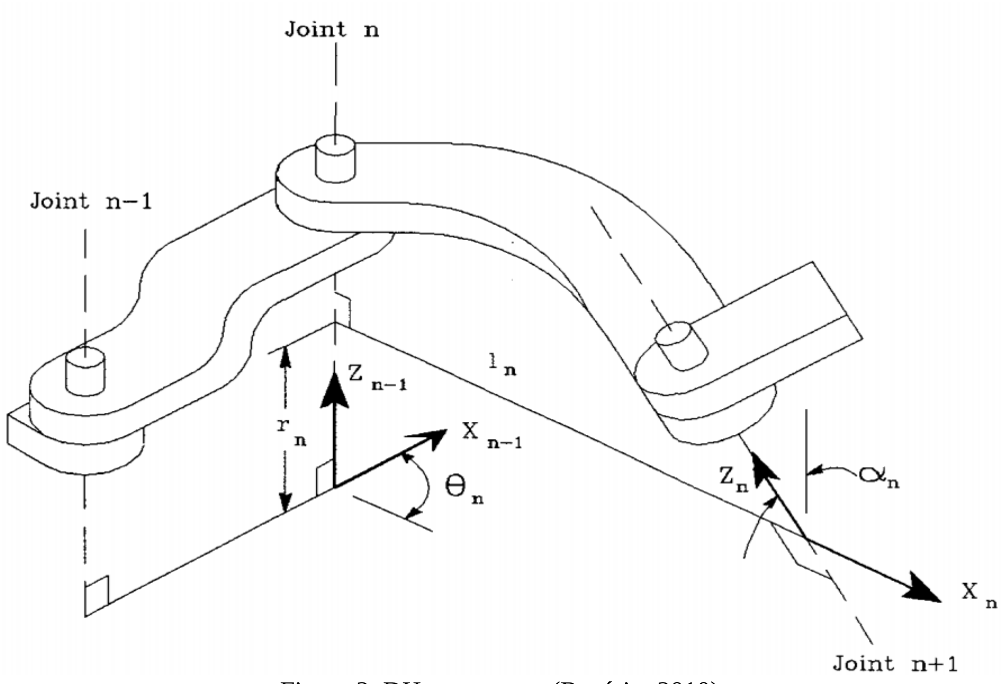

Figure 2. DH parameters (Rosário, 2010).

$$T_i^j = R_x(\alpha_j).A\left(\left[a_j\ 0\ 0\right]^T\right)R_z(\theta_j)A\left(\left[0\ 0\ d_j\right]^T\right) \tag{2}$$

Any point described in a given frame j, $\overline{p}_j = [p_x\ \ p_y\ \ p_z\ \ 1]^T$ can be described in another frame i as a product of all the intermediate transformations frame by frame (Kofinas, 2012) as shown in Eq. (3).

$$T_i^j = T_i^{i+1}.T_{i+1}^{i+2}\ldots T_{j-1}^j \quad , \quad \overline{p}_i = T_i^j.\overline{p}_j \tag{3}$$

By using the kinematic model proposed by Kofinas (2012), it is proposed here an approach for measuring 3D coordinates and frame locations by using a calibration board with feature points or patterns extracted from the images provided by the robot's head cameras (top and bottom). The main algorithm is developed in Python using the OpenCV library, Numpy and Pickle extensions.

The algorithm is divided into sections as follows: initialization and variable setup, image acquisition and feature detection, chessboard pose estimation, robot encoder values reading, forward kinematics calculations, error estimation and correction of the robot´s joint angles. According to Wiest (2001) and Elatta *et. al.* (2004) in Kastner (2014), the proposed algorithm fulfils the four steps of robot calibration: *modeling*, *measurement*, *identification* and *compensation*.

2. CAMERA CALIBRATION AND 3D POSITION ESTIMATION

Camera calibration is a necessary step the robot pose calculation, providing an estimation of the camera calibration matrix (intrinsic parameters including the radial and tangential lens distortion coefficients k1, k2, k3,p1 and p2). The OpenCV camera calibration uses a pin hole camera model shown in Eq. (4) and its expanded form in Eq (5):

$$s[m'] = A.[R|\ t].[M'] \tag{4}$$

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} . \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{5}$$

where *s* is a scale factor, *u* and *v* are image projected coordinates, *f* is the focal length (x and y directions), *c* is the camera optical center coordinates. The [R| t] matrix is composed by the extrinsic parameters that transform the [X,Y,Z] object position in world coordinates to the camera coordinate frame (OpenCV, 2017). The calibration routine takes into account some distortion coefficients as shown in the lens distortion model equations in Eqs. (6) and (7), assuming that coefficients of fourth or higher order are usually negligible. The equations can be simplified to Eqs. (8) and (9) respectively.

$$x'' = x' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x' y' + p_2(r^2 + x'^2) \tag{6}$$

$$y'' = y' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(r^2 + 2y'^2) + 2p_2 x' y' \tag{7}$$

$$x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x' y' + p_2(r^2 + x'^2) \tag{8}$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y'^2) + 2p_2 x' y' \tag{9}$$

The automatic calibration process consists of acquiring several images with a fully visible chessboard planar pattern and next the coordinates of the extracted feature points from the image are output into an iterative solvePnP algorithm to estimate the camera pose. After this initial guess the program runs a Levemberg-Marquadt optimization routine to minimize the reprojection error, comparing the distances between the feature points in image coordinates with the reprojected ones by using the current camera parameters and poses (OpenCV, 2017).

For the purpose of the camera calibration, this initial step makes use of three sets of calibration image sets (containing 40, 50 and 53 images) and the resulting camera matrices (mtx) are shown in Tab. 1. The expected value for cx is 160 pixles and for cy is 120 pixels, as the image is 320 pixels wide and 240 pixels in height.

Table 1. Experimental results for the estimation of the matrices of the camera intrinsic parameters.

N. Img.	**40**			**50**			**53**		
Matrix	180,079	0	148,874	256,922	0	160,862	198,944	0	166,102
	0	182,126	116,678	0	256,206	119,149	0	202,909	116,476
	0	0	1	0	0	1	0	0	1

The respective distortion coefficients are shown in Tab. 2 together with the estimation of the reprojection error as a Root Mean Square error (RMS) provided by the OpenCV camera calibration algorithm.

Table 2. Experimental results for camera distortion coefficients and RMS error estimates.

N. Img.	40	50	53
k1	-0,1705	-0,0825	-0,2805
k2	0,1412	-0,0814	0,3303
p1	0,0209	0,0031	-0,0027
p2	-0,0056	0,0008	-0,0075
k3	-0,0677	0,3032	-0,1824
RMS	**0,490**	**0,100**	**0,289**

Most of the parameters was found to be larger than expected and further intrinsic matrix adjustments will be performed with the OpenCV optimization process called "getOptimalNewCAmeraMatrix" for future works. From the results obtained, the parameters chosen were those achieved by the 50 image calibration set of images. This particular experiment returned the smallest RMS reprojection error and also smaller optical center deviations. The resulting camera matrix was used for the next algorithm processes.

The next step of the program is to locate the chessboard on images. For this, the program uses the OpenCV "findChessboardCorners" function that uses a similar solvePnP program but using a RANSAC algorithm to make the function respond better to noise and outliers (OpenCV, 2017). With the image points of the chessboard acquired in mage coordinates, the program can calculate the 3D pose of each feature point, located on the colored square vertex. These 3D data are to be compared to the forward kinematics data calculated from the robot's encoders.

3. KINEMATIC MODEL

The DH parameters used to build the kinematic model were chosen according to the work of Kofinas (2012) with few small variations to adapt the model from the 3.3 (Academic Edition) to the H25 V4 version of the robot used on this work. The Left Leg and Top Camera parameters are shown in Tables (3) and (4), according to Fig. 1 and Fig. 2:

Table 3. Left Leg DH parameters.

Frame(joint)	a	alpha	d	theta
Base	A(0,HipOffsetY,-HipOffsetZ)			
LHipYawPitch	0	-3π/4	0	θ_1 - (π/2)
LHipRoll	0	-π/2	0	θ_1 - (π/2)
LHipPitch	0	π/2	0	θ_3
LKneePitch	-ThighLength	0	0	θ_4
LAnklePitch	-TibiaLength	0	0	θ_5
LAnkleRoll	0	-π/2	0	θ_6
Rotations	$R_z(\pi)R_y(-\pi/2)$			
End Effector	A(PatternOffsetX,0,-FootHeight)			

Table 4. Top Camera DH parameters.

Frame(joint)	a	alpha	d	theta
Base	A(0,0,NeckOffsetZ)			
Head Yaw	0	0	0	θ_1
Head Pitch	0	-(π/2)	0	θ_2-(π/2)
Rotations	$R_x(\pi/2)R_y(\pi/2)$			
Top Camera	A(topCameraX,0,topCameraZ)			

The local transformations from Base to TopCam and from Base to LeftLeg are shown in Eq. (10) and Eq. (11).

$$T_{Base}^{TopCam} = A_{Base}^{0} T_0^1 T_1^2 R_x\left(\frac{\pi}{2}\right) R_y\left(\frac{\pi}{2}\right) A_2^{TopCam} \quad (10)$$

$$T_{Base}^{LeftLeg} = A_{Base}^{0} T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_5^6 R_z(\pi) R_y\left(-\frac{\pi}{2}\right) A_6^{TopCam} \tag{11}$$

The global transformation from the left leg chessboard planar pattern (attached to the robot's foot) to the top camera coordinate system is defined by the Eq. (12). The setup is shown in Fig. 3.

$$T_{LefLeg}^{TopCam} = \left(T_{Base}^{LefLeg}\right)^{-1} T_{Base}^{TopCam} \tag{12}$$

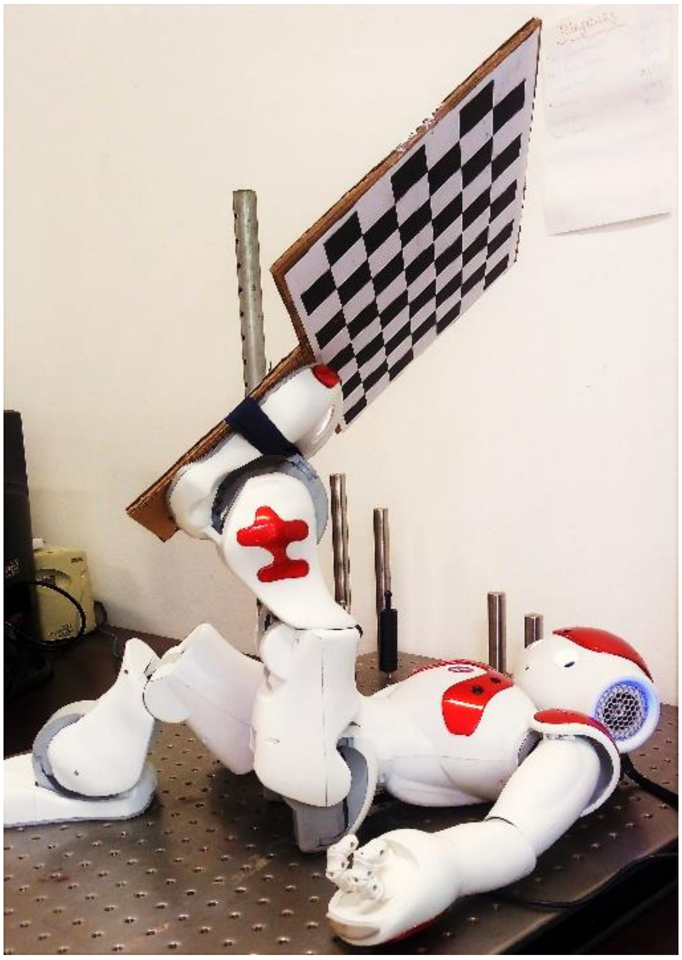
Figure 3. Robot laying position with planar pattern attached to its left foot.

4. RESULTS

To calculate the deviations between the measured data and the estimated 3D coordinates by using the Forward Kinematic model, the robot was put in laying position (Fig. 3) and several images of the calibration board were recorded synchronized with the joint encoder values. The results are presented as: 3D point estimates (points in the camera coordinates and calculated by the forward kinematic model), robot joint encoder values and distance estimates (for qualitative comparison purposes only). The data is shown in Tabs. 5, 6 and 7.

Table 5. Robot's Joint Encoder Values (Degrees).

	Camera Angles		Left Leg Angles					
	HeadYaw	HeadPitch	LhipYawPitch	LhipRoll	LHPitch	LkneePitch	LanklePitch	LankleRoll
f0	0,07666	0,51845	-0,04291	-0,11194	-1,61066	0,59515	0,25460	-0,34664
f1	0,07819	0,51845	-0,04291	-0,10887	-1,41431	0,59515	-0,17338	-0,34664
f2	0,08126	0,51845	-0,11654	-0,35585	-1,44192	0,08893	0,89888	-0,03371
f3	0,07666	0,51845	-0,05058	-0,03677	-1,09677	0,06899	0,25614	-0,01837
f4	0,07819	0,51845	0,06907	-0,22085	-1,23790	0,06899	0,71940	-0,15336
f5	0,07666	0,51845	0,00618	0,27003	-1,17193	-0,12890	0,32977	0,46945
f6	0,07666	0,51845	-0,00303	0,22554	-1,17193	-0,11816	0,59822	0,40042
f7	0,07819	0,51845	-0,04291	-0,10887	-1,41584	0,59515	-0,17185	-0,34664
f8	0,07819	0,51845	0,01845	-0,36965	-1,33914	0,04598	0,89428	-0,19478
f9	0,07819	0,51845	0,01845	-0,36965	-1,33914	0,04598	0,89428	-0,19478
f10	0,07819	0,51845	0,01231	-0,34051	-1,17807	0,02450	0,24080	-0,39266
f11	0,07666	0,51845	-0,00609	0,22861	-1,22409	0,07206	0,42948	-0,01990
f12	0,07666	0,51845	-0,00916	0,23014	-1,17347	0,06899	0,08279	-0,01837
f13	0,07666	0,51845	-0,02297	0,22554	-1,55850	0,08279	0,91422	-0,01683
f14	0,07666	0,51845	-0,02297	0,22554	-1,55850	0,08279	0,91422	-0,01683

Table 6. Camera points and Forward Kinematic Points (mm).

PointCam	**X**	-22,231	-33,511	-4,219	-65,949	-1,215	-105,625	-118,068	-33,514	34,691	35,812	12,007	-103,639	-102,474	-86,644	-86,647
	Y	-130,846	-132,578	-140,435	-76,534	-45,431	-24,143	-23,075	-132,540	-102,973	-103,465	-164,777	-22,114	-36,442	-37,004	-37,009
	Z	326,470	269,981	384,583	342,363	479,691	246,534	324,579	269,967	464,967	465,162	313,539	346,768	267,703	345,663	345,621

Pt For. Kin.	**X**	-14,687	81,474	-124,305	33,159	-109,743	56,612	-21,503	81,179	-142,038	-142,038	29,988	-4,110	90,527	-100,184	-100,184
	Y	-159,774	-149,999	-136,264	-60,821	-139,614	138,884	129,131	-149,987	-178,353	-178,353	-210,870	4,779	6,344	4,317	4,317
	Z	274,793	250,167	294,997	343,024	348,041	270,826	323,579	250,133	298,792	298,792	240,648	352,414	301,182	320,332	320,332

Table 7. Distance estimations: Camera (Dcam) and Forward Kinematics (Dfk) (mm).

Dcam	352,417	302,638	409,443	356,958	481,839	269,292	346,157	302,609	477,495	477,874	354,404	362,599	288,953	358,273	358,234
Dfk	307,756	242,813	368,055	267,978	355,377	244,558	298,755	243,005	377,655	377,655	285,966	287,591	218,151	346,008	346,008

From the data collected it is possible to observe that the Z coordinate estimates presented larger absolute errors due to the difficulties to reproject 2D to 3D points using a single camera in a single pose. This type of estimation is largely accepted as non-functional but it was able to reproduce consistent results in the X and Y axis, even in highly inclined positions of the pattern in relation to the robot's camera. All data is represented in the Top Camera frame, the 3D points and the distances are represented in the same axis.

5. CONCLUSIONS

In this article a robot calibration methodology for the humanoid robot NAO was proposed, using proprioceptive cameras and joint encoders of the robot. The modeling and measurement step are described and some results assessment was carried out so far. For the future ongoing work, it is necessary to ensure that the vision system provide more accurate measurements and an accurate kinematic model can be fit.

This research is under development, and so far, high order lens distortion coefficients and a non-linear least squares optimization based on the Levemberg-Marquadt algorithm have been tested. The methodology is expected to provide global minima for the robot joint errors and link lengths such that the new corrected kinematic model can improve the stability of the robot dynamic control.

6. ACKNOWLEDGEMENTS

The authors would like to thank CAPES – Coordination for the Improvement of Higher Education Personnel – for their financial support.

7. REFERENCES

Craig, J., J., 2012, “Robótica”, Ed. Pearson, São Paulo, Brasil, pp 1-125.
Gouaillier, D. et al. 2008, “The NAO humanoid: a combination of performance and affordability”, CoRR abs/08073223.

Elatta, A. Y., Gen, L. P., Zhi, F. L., Daoyuan, Y., Fei, L., 2004, "An overview of robot Calibration" IEEE Journal on Robotics and Automation 3, pp.74-78.
Hartenberg, R.S., Denavit, J., 1964. "Kinematic Synthesis of Linkages", McGraw-Hill Book Company, New York, USA.
Kastner, T., Röfer, T., Laue, T., 2015, "Automatic Robot Calibration for the NAO", RoboCup 2014: Robot World Cup XVIII, d. Springer International Publishing, Lecture Notes in Computer Science, Vol. 8992, pp. 233-244.
Kofinas, N., 2012, "Forward and Inverse Kinematics for the NAO Humanoid Robot", Diploma Thesis, Technical University of Crete, Greece.
Ogata, K., 1985, "Engenharia de Controle Moderno", São Paulo, Brazil: Prentice/Hall do Brasil.
OpenCV, 2017 "OpenCV Documentation", 10 Sep. <http://docs.opencv.org/3.3.0/>
Rosário, J., M., 2010, "Robótica Industrial I", Ed. Baraúna, São Paulo, Brasil, pp. 49-169.
SoftBank Robotics, 2017b "NAO Documentation", 7 Aug. <http://doc.aldebaran.com/2-1/home_nao>
SoftBank Robotics, 2017a "NAOqi Documentation", 14 Aug. <http://doc.aldebaran.com/>
Veitschegger,W., K., Wu,C., 1986, "Robot accuracy analysis based on kinematics", IEEE Journal of Robotics and Automation, RA-2(3), pp.171-179.
Weist, U., 2001, "Kinemtische Kalibrierung von Industrierobotern. Berichte aus der Automatisirunggstechnik" Shaker Verlag.
Yamane, K., 2011, "Practical Kinematic and Dynamic Calibration Methods for Force-Controlled Humanoid Robots", Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots, Bled, Slovenia.

8. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.