3D Stereo Estimation and Fully Automated Learning of Eye-Hand Coordination in Humanoid Robots

S.R. Fanello\textsuperscript{1}, U. Pattacini\textsuperscript{2}, I. Gori\textsuperscript{2}, V. Tikhanoff\textsuperscript{2}, M. Randazzo\textsuperscript{2}, A. Roncone\textsuperscript{2}, F. Odone\textsuperscript{4} and G. Metta\textsuperscript{2}

Abstract—This paper deals with the problem of 3D stereo estimation and eye-hand calibration in humanoid robots. We first show how to implement a complete 3D stereo vision pipeline, enabling online and real-time eye calibration. We then introduce a new formulation for the problem of eye-hand coordination. We developed a fully automated procedure that does not require human supervision. The end-effector of the humanoid robot is automatically detected in the stereo images, providing large amounts of training data for learning the vision-to-kinematics mapping. We report exhaustive experiments using different machine learning techniques; we show that a mixture of linear transformations can achieve the highest accuracy in the shortest amount of time, while guaranteeing real-time performance. We demonstrate the application of the proposed system in two typical robotic scenarios: (1) object grasping and tool use; (2) 3D scene reconstruction. The platform of choice is the iCub humanoid robot.

I. INTRODUCTION

Three dimensional information is fundamental to control robots that interact physically with objects and people. The recent availability of cheap 3D sensors such as the Kinect has certainly revolutionized our ability to measure the environment in three dimensions. This advancement, however, may not be sufficient under all conditions. For instance, the Kinect sensor is of little use outdoors since the projected infrared pattern is virtually invisible in direct solar light. In addition, high-end humanoid robots often sport moving cameras, mounted in the eyes, meant to imitate the efficiency of the human oculo-motor system. If we find ourselves in one of these situations, then it is likely that we need to calibrate the robot cameras and to refer the calibration to the robot kinematics (i.e. to construct a body-centered calibration).

Calibration is typically a “solved problem” for static cameras, where calibration patterns can be shown off-line to the system, parameters estimated and checked (even manually) before using the cameras for measurement. In these cases, the remaining problem is to calibrate the robot kinematics to the camera positions, which is again simple for typical industrial robots (fixed base, rigid, etc.). The case of humanoids is somewhat special. On the one hand, often humanoid robots are highly complicated devices, where the guarantee of rigidity is difficult to maintain. On the other hand, a static cameras is too limiting when we like to achieve high accuracy, to observe a large field of view, etc.

In this sense the iCub robot \cite{14} is paradigmatic. It possesses a three degree of freedom oculomotor system with independent control of vergence and a common tilt. The cameras can move fast up to the velocity range of human saccadic movements. The head can also move independently, adding to the difficulty of maintaining an accurate hand-eye calibration. Furthermore, the robot is mostly tendon-actuated, therefore the calibration of the camera with respect to the hand requires an additional stage of modeling and learning.

It cannot be given for granted that an \textit{a priori} model of the robot kinematics would save the day. Empirically, at least for the iCub, the simple approach of using the CAD model of the kinematics would lead to errors in the 3D point estimation in the range from a few centimeters up to 6 cm. The error varies as a function of the workspace due e.g. to gravity, to the mechanical interplay of the various links, to friction, and so forth making a fully parametric approach practically unfeasible.

In the following we address these issues. We aim at an automated calibration procedure that does not require a special rig for the 3D stereo estimation. In practice we would like to re-calibrate frequently, ideally continuously (incrementally) and in real time (online and fast). Therefore small glitches in the robot’s mechanics, drifts, and other inaccuracies would be irrelevant. Figure \ref{fig:1} illustrates the problem formulation for the case of the iCub. Our goal is to compensate for those unmodeled factors by estimating the relation between the two indicated roto-translations $H_A$ and $H_B$ that are described by the known kinematics.

II. RELATED WORK

In short, the calibration problem requires recovering the position and orientation of the cameras with respect to a common reference frame. A second transformation is required to account for the end-effector position in the camera coordinate systems. Tsai et al. \cite{26}, \cite{25} and Shiu et al. \cite{20} carried out pioneering work in the field of eye-hand coordination. They tackled the problem by considering fitting rotations first and then recovering translations. Later approaches attempted to simplify the notation and compute closed form solutions.
As we mentioned, humanoid robots are typically a different “beast” altogether. They are built with the goal of being mechanically and perceptually competent, by paying the price of high complexity in the joints, sensors, interconnections. Complexity unfortunately introduces defects that accumulate and eventually lowers the overall accuracy. In this sense the assumption of a highly accurate kinematic model simply derived from the CAD drawings cannot be satisfied. We have therefore to design methods that do not rely on the accurate knowledge of the end-effector trajectory. To recapitulate, the main contributions of this paper are:

- An eye-hand calibration algorithm (for humanoid robots), where unmodeled elasticity or wear and tear make the CAD kinematic models unreliable. We cannot use information about the camera/end-effector motion.
- Ability to calibrate complex stereo rigs - not limited to in-hand devices - where the relative configuration of the cameras can significantly vary in actual operating conditions.
- Use of visual features to recover 3D structure without depth sensors. The entire system is based on a standard pair of RGB cameras.
- Use of visual features to match the end-effector frame of reference to that of the camera in different configurations; the processing pipeline is fully automated and does not require human supervision.

III. The System

The general formulation for eye-hand calibration builds on the following equation:

$$AX = XB,$$

where $A \in \mathbb{R}^{4 \times 4}$ is a rototranslation matrix that represents the camera motion; it can be decomposed into a rotational part $R_A \in \mathbb{R}^{3 \times 3}$ and a translation vector $t_A \in \mathbb{R}^3$. The matrix $B \in \mathbb{R}^{4 \times 4}$ accounts for the end-effector motion, whereas $X \in \mathbb{R}^{4 \times 4}$ is the unknown transformation relating the two reference systems.

Fig. 2. Left: A typical scene as seen by the robot point of view. Middle: The depth map retrieved by the vision system based on pure kinematics calibration (producing poor results). Right: The improved depth map obtained after the calibration described in Section III-A.

State-of-the-art methods usually make assumptions regarding the knowledge of $A$ and $B$, or foresee the use of dedicated calibration patterns. Differently, we exploit optimization techniques to model the function $f(A) = B$, along a single axis (e.g. error on the $X$ axis around 0.8 cm); when considering all the axes simultaneously, the overall error is around 3.2 cm [11].
avoiding the explicit computation of the matrix $X$, and providing an estimate of the unknown matrices $A$ and $B$.

Generally, in order to achieve precise open-loop reaching knowing the coordinates of the 3D point to reach, the following conditions must be satisfied: (1) the target needs to be retrieved very accurately in the cameras reference frames; (2) the coordinates transformation from the camera to the end-effector frame must be known with precision.

An example of a scenario wherein the first condition is not satisfied is illustrated in Fig. 2: inaccurate kinematic model leads to poor disparity map and thus to incorrect 3D points. Similarly, as it can be noted in Fig. 4, if the second condition is not met, the end-effector position retrieved from stereo vision (red dot) does not match the expected position calculated from the robot kinematics (green dot).

Therefore, we propose an algorithm to solve the two following sub-problems:

- **3D Structure Estimation.** We calibrate on-line and in real-time the iCub camera relative positions dealing with varying eyes configuration. This procedure generates 3D points with respect to the camera reference system providing an estimate of matrix $A$.

- **Eye-Hand Calibration.** We collect pairs of 3D points representing the end-effector in the stereocamera (i.e. $A$) and kinematics (i.e. $B$) reference frames. Then we employ optimization techniques to model the mapping $f(A) = B$. We further show that a fast linear mapping is sufficient to achieve good results.

Solving these sub-problems in a sequence allows obtaining highly accurate 3D points to be used for reaching tasks thus registering the stereo cameras with the robot kinematics.

### A. 3D Structure Estimation

We consider couples of images acquired by the iCub stereo vision system. The main problem is the estimation of the two-view geometry used in the image rectification process. Subsequently, any state-of-the-art disparity algorithm can be used. In general, a 3D point $X = [x, y, z, 1]$ is projected (up to a scale factor $s$) into the image plane $x = [u, v, 1]$ using a perspective transformation:

$$sx = PX^T,$$

where $P = K[R|t]$ is the camera matrix composed of the intrinsic parameters $K \in \mathbb{R}^{3 \times 4}$ and the extrinsic parameters $[R|t]$, with $R \in \mathbb{R}^{3 \times 3}$ accounting for rotation, whereas $t \in \mathbb{R}^{3 \times 1}$ is the translation vector. The intrinsic parameters are to be determined only once; this can be done using standard calibration methods as in [8]. On the contrary, the extrinsic parameters need to be estimated each time the robot eyes change their relative configuration. To effectively meet this goal, while complying with real-time performance constraints, we proceed as described in the following sections.

1) **Undistorted Images:** The first step consists in removing the image distortion. Knowing the intrinsic parameters and the distortion coefficients, the images can be remapped into two new undistorted frames.

2) **Feature Matching & Fundamental Matrix:** the second step is the estimation of the Fundamental Matrix $F \in \mathbb{R}^{3 \times 3}$ relating corresponding points in two images (i.e. matches). Given a match $x, x'$, it holds:

$$x' \cdot F \cdot x = 0.$$  \hspace{1cm} (3)

In this work we used SIFT detectors and descriptors [13] to compute keypoints from the undistorted images to then execute the matching step (see Fig. 3). Being rank 2 and up to scale, the matrix $F$ can be estimated using at least 7 points. Alternatively, $F$ can be calculated employing the camera matrices $P_L$ and $P_R$ [8]. In this respect we can profitably make use of the known kinematics as well as the camera intrinsics parameters to retrieve an initial estimate of $P_L$ and $P_R$ so as filter out the false positive matches and successively refine the prediction. We combine the two methods. First, we compute the Fundamental Matrix $F_K$ that relates the cameras planes by using the kinematics structure as priors, and then we use $F_K$ to validate the correspondences. Second, we employ the good matches to estimate the real Fundamental Matrix $F$.

Given $i = 1, 2, 3$, $j = (i + 1) \text{ mod } 3$ and $k = (i + 2) \text{ mod } 3$, we define:

$$X_i = \begin{pmatrix} P_L(j, 1) & P_L(j, 2) & P_L(j, 3) & P_L(j, 4) \\ P_L(k, 1) & P_L(k, 2) & P_L(k, 3) & P_L(k, 4) \end{pmatrix}$$ \hspace{1cm} (4)

$$Y_i = \begin{pmatrix} P_R(j, 1) & P_R(j, 2) & P_R(j, 3) & P_R(j, 4) \\ P_R(k, 1) & P_R(k, 2) & P_R(k, 3) & P_R(k, 4) \end{pmatrix}$$ \hspace{1cm} (5)

The matrix $F_K$ is derived as follow:

$$F_K = \begin{pmatrix} \text{det}([X_1; Y_1]) & \text{det}([X_2; Y_1]) & \text{det}([X_3; Y_1]) \\ \text{det}([X_1; Y_2]) & \text{det}([X_2; Y_2]) & \text{det}([X_3; Y_2]) \\ \text{det}([X_1; Y_3]) & \text{det}([X_2; Y_3]) & \text{det}([X_3; Y_3]) \end{pmatrix}$$ \hspace{1cm} (6)

At this point we validate the correspondences, discarding the matches where $x' \cdot F_K \cdot x > 0.01$. From the remaining correspondences we run the normalized 8-points algorithm [8] to compute the final Fundamental Matrix $F$, using a traditional RANSAC scheme for outliers rejection [5]. At the end of this process we obtain a Fundamental Matrix $F$ describing the epipolar geometry of the current eyes configuration.

3) **Essential Matrix, Camera Geometry & Rectification:** at this point we estimate the Essential Matrix $E$ [12]. $E$ relates...
the right and left views given the camera intrinsic parameters. Starting from the Fundamental Matrix, we compute:

\[ E = K_R^T F K_L, \]  

where \( K_R, K_L \) are the 3 \times 3 matrices of the intrinsic parameters. Alternatively, \( E \) can be also derived in terms of a rototranslation transformation between the two camera views, leading to the final estimate of the extrinsic camera parameters. We follow the classic procedure described in [8], determining four rototranslation matrices and disambiguating them using the chirality test [7]. A further check is done considering the kinematics prior of the robot: if the retrieved matrices are in line with the kinematic predictions, given a certain tolerance, the solution is accepted, otherwise is discarded. We hence obtain a couple of camera matrices \( P_L = K_L[I|0] \) and \( P_R = K_R[R|t] \), with which we perform the rectification process by applying the Bouguet algorithm [30] to rotate the cameras so that they share the same \( X \) axis.

4) Structure Estimation: dealing with rectified images allows us to apply standard algorithms for 3D structure estimation as the procedure described in [9]. Thus, given the disparity \( d \) computed at the pixel \((u, v)\), we can readily reproject points in the 3D space using the formula:

\[ \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \frac{(u - c_x)b/d}{(v - c_y)b/d} \begin{pmatrix} f_x \\ f_y \end{pmatrix}. \]  

where \( b \) is the baseline of the two cameras (i.e. the norm of the translation vector \( t \)), \( f \) the focal length and \((c_x, c_y)^T \) is the principal point of the stereo camera system. In Fig. 2-right we show an example of the resulting disparity maps.

Fig. 4. RGB image and disparity map with the expected (green dot) and the real (red dot) end-effector. The former position is retrieved directly using the known kinematics and projecting the 3D point into the image plane while the latter position is computed automatically by means of the depth map allowing for fast and easy segmentation of the fingertip.

B. Eye-Hand Calibration

Provided with a perfect model describing how the vision system gets coupled with the kinematics, any 3D location of the end-effector can be flawlessly mapped in the cameras image planes with no mismatch. In reality, as visible in Fig. 4, the expected position of the end-effector and the actual one differ by some unknown offsets due to the mechanical issues discussed in Section I. We therefore tackle the calibration problem from a different perspective: instead of looking for \( A \) and \( B \), we consider a set of \( N \) points belonging to the two different reference systems: \((X_A^i, X_B^i) \) \( \forall i = 1, \ldots, N \), with \( X_A^i, X_B^i \in \mathbb{R}^3 \). Given this set of points, our goal is to learn the function \( f(X_A) = X_B \). The main advantage with respect to other approaches is twofold: (1) when a large set of examples is available, the function can be learned accurately and can generalize to new positions in 3D space; (2) we do not need a calibration pattern to collect the training data.

![Fig. 5. Data generated for the hand-eye mapping: red dots are 3D points retrieved from the stereo vision system after the fingertip detection; blue dots are the end-effector positions detected via kinematics. Left and right plots show two different views of the same point clouds highlighting the relative offsets. Units are given in meters.](image)

We consider the 3D end-effector position in homogeneous coordinates \( \vec{X}_B^i \in \mathbb{R}^4 \) and we use the forward kinematics \( \vec{H}_B \in \mathbb{R}^{4 \times 4} \) to map it to \( \vec{X}_B^i = \vec{H}_B \vec{X}_B^i \), which gives the coordinates of the end-effector in the ROOT reference system (see Fig. 1). The acquisition of \( X_A^i \) samples employs the stereo vision system of the iCub. In particular, we use the Otsu method [15] applied to the depth map in order to segment out the background within a bounding box around the expected end-effector position \( X_B^i \). Then, considering the configuration of the hand showed in Fig. 4 with the index finger pointing upward, we can detect the fingertip by retrieving the first non-null pixel in the region of interest starting from the top-left corner. This 2D point is reprojected in 3D space using Eq. 8, obtaining \( X_A^i \). The homogeneous point \( \vec{X}_A^i \in \mathbb{R}^4 \) is then mapped to the ROOT frame \( \vec{X}_B^i = \vec{H}_A \vec{X}_A^i \) by means of the known forward transformation \( \vec{H}_A \). Fig. 4 depicts an example of the described procedure: the end-effector point is reprojected in the image plane for visualization purposes (green dot); in red we show the detected fingertip.

1) Data Acquisition: to automatically collect ground truth data, iCub moves its end-effector along multiple elliptic trajectories with different centres, sizes and orientations in the Cartesian space. The data acquisition procedure is very simple, reliable and requires on average 1 minute for each trajectory. During data acquisition, the robot actively fixes the fingertip position, in order to track the target exploiting a large number of eye configurations. Examples of the acquired data is depicted in Fig. 5: in blue we show the set of expected end-effector positions \( X_B \), while in red the 3D vision points \( X_A \) transformed with respect to the root frame are drawn.

2) Map Calibration: our goal is to learn the spatial relationship between the acquired two data points. To this end, we propose to rely on a fast linear mapping \( \vec{H}^* \in \mathbb{R}^{4 \times 4} \).
such that \( \overline{X}_{R,B} = H^* \cdot \overline{X}_{R,A} \), which can be determined by solving the following minimization problem:

\[
H^* = \arg \min_{H} \frac{1}{N} \sum_{i} \| \overline{X}_{R,B}^i - H \overline{X}_{R,A}^i \|^2 
\]

\[
\text{s.t. } H \in SE(3),
\]

where \( SE(3) \) is the space of the admissible rototranslation matrices. To solve \( 9 \), we use Ipopt [29], a public domain software package designed for large-scale nonlinear optimization. Notably, to calibrate data sets composed of hundreds of points, Ipopt takes less than one second, a small time interval when compared to the duration of the data acquisition step, which is instead in the range of minutes. Once \( H^* \) is found, it is used at run time to correct the target \( \overline{X}_A \), as measured by the vision system, in the new 3D point \( \overline{X}_R = H^* \cdot \overline{X}_{R,A} \), expressed in the ROOT frame. The latter point \( \overline{X}_R \) serves to drive the robotic effectors to reach a point in space.

3) Mixture of Experts: in real scenarios it is likely that a single linear transformation might not generalize as expected over the whole robot workspace. To get around this inconvenience, we extend the model by introducing a mixture of linear transformations that we term experts, whose spatial competences can be easily retrieved from the corresponding training sets. As result, any point \( \overline{X}_i \) is remapped into the ROOT coordinates system using the linear combination \( \overline{X}_R = \sum_i w_i H_i \overline{X}_{R,A} \), where each \( H_i \) is obtained by locally minimizing the Eq. \( 9 \). The weights \( w_i \) depend on the distances of the point \( \overline{X}_{R,A} \) from the centroids \( c_i \in \mathbb{R}^3 \) of the training sets, so as on the corresponding covariance matrices \( S_i \in \mathbb{R}^{3 \times 3}. \) The parameters \( c_i, S_i \) describe indeed the spatial occupation of the 3D points used to train the i-th expert in terms of the resulting minimum ellipsoid [24].

The weights are thus assigned through radial basis functions computed with resort to the Mahalanobis distance and then normalized:

\[
w_i = \frac{\exp((-c_i - \overline{X}_{R,A})^T S_i^{-1} (c_i - \overline{X}_{R,A}))}{\sum_j \exp((-c_j - \overline{X}_{R,A})^T S_j^{-1} (c_j - \overline{X}_{R,A}))}, \tag{10}
\]

Remarkably, we tested in our experiments how the mixture of experts runs fast, providing real-time performances, and, accurately, achieving with only 4 experts very low reaching errors in the iCub workspace (see Section IV-A).

IV. EXPERIMENTS

In this section we evaluate the system with exhaustive quantitative experiments and comparisons.

A. Linear Experts vs. Machine Learning Techniques

Given the availability of (theoretically) infinite ground-truth data, also more general machine learning (ML) techniques seem to be good candidates to learn the visuo-kinematic mapping. For this reason, we compare the performance of 4 different methods: the mixture of linear experts (SE3) described in Section III-B, the Gaussian Processes (GP) [18], the Grand Unified Regularized Least Squares (GURLS) [22], and the Support Vector Machines (SVM) [27]. While SE3 locally minimizes a set of linear transformations, GP, GURLS and SVM are applied to learn the more generic relation \( f(\overline{X}_A) = \overline{X}_B \). The rationale underlying this choice has a pragmatic foundation: the latter ML frameworks are available in the form of off-the-shelf libraries, hence the intention is to make use of them as they come without any attempt to inject an \textit{a priori} knowledge to facilitate the search. Conversely, the \textit{ad hoc} design of SE3 that shapes the structure of the solution of Eq. \( 9 \) requires a very small implementation effort, given the simplicity of the geometric model. Importantly, the significant outcome of such an approach is that SE3 does outperform the other ML techniques as demonstrated hereinafter: in fact, SE3 reaches the same level of accuracy and generalization of ML methods in determining the map with less number of samples and, thus, within the shortest exploration time.

During data acquisition, we let the iCub follow with its end-effector (i.e. the index fingertip) some predefined ellipsoidal paths in the Cartesian space. Each path is sampled with 100 points representing two ellipses lying on co-orthogonal planes. Overall, we collect about 2000 points (i.e. 20 ellipsoids). During the testing phase we generate new ellipsoidal paths. Empirically, we found out that 4 experts suffice to cope with the iCub workspace relevant for the envisaged tasks. The hyper-parameters of ML methods have been estimated using standard cross-validation. In the first experiment we aim to evaluate the robustness of the algorithms with respect to the number of training data (Fig. 6). Notably, SE3 reaches precisions up to 1 cm after a few examples (i.e. around 50). All the other methods require more examples, around 150, before reaching the same accuracy.

The significantly faster convergence of SE3 has a favourable impact in operative contexts where uncertainty might take place varying the robot body schema. When such a situation is detected, the eye-hand calibration needs to be triggered again; hence, the procedure requiring the fewest training samples with the lowest prediction error must be preferred.

Further, in order to assess generalization, given a relatively small number of training points, we train on a single ellipsoid and test on the other 19. Results for both training and test sets are showed in Fig. 7. The initial error between vision
and kinematics is 4 cm on average. In this experiment, SE3 outperforms the remaining methods, achieving 1.1 cm of error. GURLS is second in rank with 1.3 cm. Notably, on the training data, the ML methods behave better than SE3 (i.e. errors around 0.2 cm against 0.6 cm). This clearly shows how, with respect to the pure geometric model SE3, ML techniques suffer more from data overfitting, requiring a larger samples set in the exploration stage.

TABLE I
RESULTS OF THE REACHING EVALUATION USING THE Vicon system.

<table>
<thead>
<tr>
<th>Method</th>
<th>NO CALIBRATION</th>
<th>CALIBRATION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\sigma_{eff}$ [cm]</td>
<td>$</td>
</tr>
<tr>
<td>P1</td>
<td>0.27</td>
<td>4.29 ± 0.56</td>
</tr>
<tr>
<td>P2</td>
<td>0.01</td>
<td>7.83 ± 0.05</td>
</tr>
<tr>
<td>P3</td>
<td>0.33</td>
<td>6.84 ± 0.52</td>
</tr>
<tr>
<td>P4</td>
<td>0.49</td>
<td>5.89 ± 0.82</td>
</tr>
<tr>
<td>P5</td>
<td>0.13</td>
<td>4.69 ± 0.68</td>
</tr>
<tr>
<td>P6</td>
<td>0.51</td>
<td>5.79 ± 0.50</td>
</tr>
<tr>
<td>P7</td>
<td>0.20</td>
<td>5.67 ± 0.20</td>
</tr>
<tr>
<td>P8</td>
<td>0.16</td>
<td>2.97 ± 0.02</td>
</tr>
</tbody>
</table>

V. APPLICATIONS

In order to qualitatively validate the proposed methodology, we present two real applications implemented on the iCub, which significantly benefit from the improved coordination of perception and motor capabilities.

A. Power Grasp & Tool Use

Our first task is about power grasp. The procedure described in [6] is applied on several objects. In that case, we had to fix the eye configurations and then estimate empirically the offset between vision and kinematics. In particular, for the sake of grasp reliability, we considered conditions where we manually established the offset between the 3D point perceived by the stereo vision and the one predicted by the kinematics. Clearly, that procedure was not automatic, and it had to be carried out every time the eyes configuration of the robot changed. Notably, in [6] we recorded a grasp success rate of 91.25%.

In the present work, we let the robot execute 20 grasps on 4 objects (the ones showed in Fig. 2, same used in [6]) based on the visual information retrieved by the stereo camera calibrated as in Section III-A. Then, we evaluated the performance of our system in two configurations: (1) without and (2) with the eye-hand registration step described in Section III-B. We finally measured a success rate of 23% and 97.5%, respectively. These results definitely demonstrate...
the dramatic improvement provided by the automatic eye-hand calibration, so as the remarkably increased precision achieved with respect to the original grasping experiment where offsets were fine tuned manually for each object and position in the space.

We then put to test the system in a second scenario that extends our previous work on the use of tools for exploring affordances [23]. The iCub was able to explore hand held tools, learn how to use them and eventually employ the learned skill in order to accomplish his task. We placed the tools on a rack within the reach of the iCub (see the setup in Fig. 9) and used the proposed calibration procedure to successfully reach and grasp the required tool. To demonstrate that the system is robust and sufficiently precise, we performed 20 reach and grasp actions on tools placed on the rack arranged in four different orientations with respect to the gravity direction: 0, −45 and 45 degrees obtaining 95%, 90% and 90% of successful grasps respectively.

B. 3D Scene Reconstruction

The estimated depth map can be also exploited to reconstruct the 3D space surrounding the robot, integrating data belonging to different views of the environment into a single 3D scene. This is a typical application also for mobile robotics, in which 3D cameras or laser systems are commonly employed to perform simultaneous localization and mapping tasks. In our specific case we attempt to evaluate the quality of depth data by reconstructing scenes, such as the workspace in front of the robot. For the experiment we used the CCNY Visual Odometry package that is publicly available. The algorithm [3] works by tracking relevant RGBD features between camera frames and aligning them against a unique 3D model of the world, obtaining an estimated camera pose. This pose is then used to expand the model of the world, by inserting new landmarks for each feature which is not associated to the model set. An example of a 3D scene reconstructed using robot’s cameras is shown in Fig. 10. It is worth noting that the reconstruction exhibits many non valid regions, which correspond to the areas where the depth map algorithm [9] fails, usually because of uniform color and lack of features. Nevertheless, the overall performance is good and data acquired from different views are successfully merged into a single coherent spatial model.

VI. DISCUSSION

In this paper we tackled the problem of learning the vision-kinematics mapping in humanoid robots. We showed the importance of having a reliable coordination between the vision system and the end-effector for high level applications. We proposed a fully automated procedure to calibrate the eye-hand coordination based on the stereo vision system of the iCub robot. The method accounts for the mechanics inaccuracies of the robot and works for non in-hand camera setups; notably, it does not require any supervision. Furthermore, the procedure is very fast and it can be performed on the fly. We also showed different applications which benefit from the proposed method. As future work we foresee to evaluate different algorithms for depth map estimation as well as to integrate the method with visual servoing approaches.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude to Dr. Ambra Bisio for her invaluable help and expertise provided during the Vicon experiments.

REFERENCES


