Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

Markerless Perspective Taking for Humanoid Robots in Unconstrained Environments

Tobias Fischer and Yiannis Demiris

***Abstract*— Perspective taking enables humans to imagine the world from another viewpoint. This allows reasoning about the state of other agents, which in turn is used to more accurately predict their behavior. In this paper, we equip an iCub humanoid robot with the ability to perform visuospatial perspective taking (PT) using a single depth camera mounted above the robot. Our approach has the distinct benefit that the robot can be used in unconstrained environments, as opposed to previous works which employ marker-based motion capture systems. Prior to and during the PT, the iCub learns the environment, recognizes objects within the environment, and estimates the gaze of surrounding humans. We propose a new head pose estimation algorithm which shows a performance boost by normalizing the depth data to be aligned with the human head. Inspired by psychological studies, we employ two separate mechanisms for the two different types of PT. We implement line of sight tracing to determine whether an object is visible to the humans (level 1 PT). For more complex PT tasks (level 2 PT), the acquired point cloud is mentally rotated, which allows algorithms to reason as if the input data was acquired from an egocentric perspective. We show that this can be used to better judge where object are in relation to the humans. The multifaceted improvements to the PT pipeline advance the state of the art, and move PT in robots to markerless, unconstrained environments.**

I. Introduction

Enabling robots to operate in the presence of humans and interact with them is a challenging problem [1]. The robots need to perceive the world from an egocentric perspective in order to adapt to a permanently changing environment, and in addition, they need to consider the state of the world from the others' perspective. For instance, this is particularly relevant in situations where the robot has to assist or cooperate with humans [2]. It is therefore desirable that robots have a mechanism to represent aspects of the world in a non-egocentric way, also called perspective taking (PT) [3–5].

Psychological studies suggest that humans rely on two different levels of PT depending on the task [3, 6]. For level 1 PT, *i.e.* to infer which object can be seen by another human, it is suggested that the line of sight between the other human and the object of interest is traced [6]. For level 2 PT, which is used to approximate the other humans view as well as to find the spatial location from the others perspective, mental rotation seems to be employed [6].

Several papers have shown that PT allows robots to interact with humans and other robots in a more natural way [5, 7]. More importantly, it is thought that PT is an essential element for successful cooperation and to ease communication [4]. For example, knowing that an object cannot be seen by another human has helped in resolving ambiguities in human-robot interaction [2].

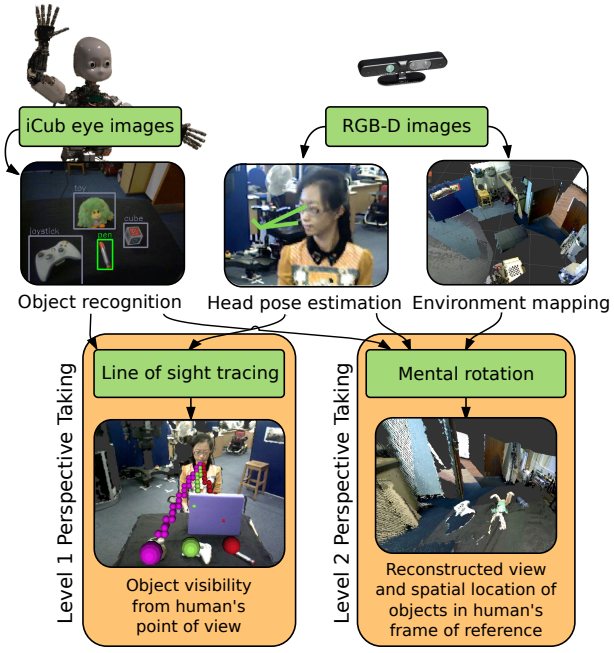

Fig. 1. Overall flow of the proposed method. The inputs to the perspective taking pipeline are images acquired from a RGB-D camera, and the iCub eyes. In the first step, the robot recognizes objects, estimates the head pose of surrounding humans, and maps the environment. Two separate processes are employed for level 1 and level 2 perspective taking; allowing the robot to infer which objects are seen by the human, what the spatial location of these objects are in the reference frame of the human, and how the world appears from the human viewpoint.

Previous works on PT in robotics [2, 7–9] either rely on motion capture systems and/or fiducial markers to retrieve the head pose of humans and position of objects, or need the objects perceivable by the human as prior information. This limits their usability to environments which are equipped with these tools, which is costly and requires a time-consuming setup. Furthermore, the environment needs to be known in advance, which requires manual modeling each time changes are made.

In this paper, we propose to overcome this limitation using a low-cost RGB-D camera mounted above an iCub humanoid robot [10] in conjunction with the iCub's eye cameras. The overall concept of the proposed framework is illustrated in Fig. 1, and Fig. 2 shows the setup used for our work.

The robots perception is split into three algorithms. Firstly, we use a state of the art visual simultaneous localization and mapping (SLAM) algorithm [11] to map the environment, so

The authors are with the Personal Robotics Lab, Department of Electrical and Electronic Engineering, Imperial College London, UK. This work was supported in part by the EU FP7 project WYSIWYD under Grant 612139. Email: *{t.fischer, y.demiris}@imperial.ac.uk*

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

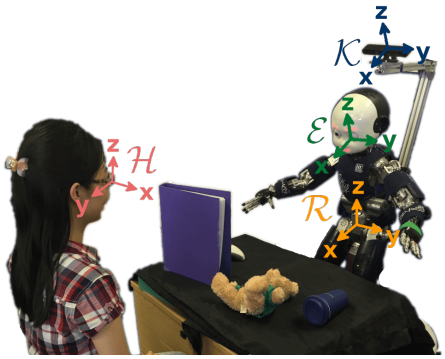

Fig. 2. Typical setup with the iCub humanoid robot, a human interacting with the robot, and various objects placed between them. Some of the objects are occluded to the human. We also show the different coordinate frames used throughout the paper.

no prior information of the environment needs to be known. Secondly, we employ a deep-learning based algorithm for real-time object recognition [12], such that no markers are needed. Thirdly, we propose a new head pose recognition algorithm to approximate the gaze of the human. For this, we extend a state of the art method based on random regression forests [13] using normalized depth images which results in higher robustness of the algorithm.

Finally, we also propose separating PT in two processes, grounded on the psychological studies introduced earlier [3, 6]. We implement a line of sight tracing algorithm for level 1 PT, and a mental rotation algorithm for level 2 PT.

II. Related Work

Several approaches for perspective taking (PT) in robotics have been proposed over the past ten years. The system proposed by Trafton *et al.* [14] is able to handle ambiguous situations where the robot can see two similar objects, but one of them is occluded from the human. Kennedy *et al.* [9] show that left/right judgments can be solved by integrating this system within a cognitive framework.

Breazeal *et al.* [7] use such ambiguous situations in a learning scenario for humanoid robots. Their robot retains two sets of beliefs, one is for the self and one is for the others perspective. Using this strategy, the robot learns patterns such as turning all buttons which are mutually visible to ON from fewer demonstrations, and allows the human to clarify false beliefs of the robot. This is achieved using an attentional system, which reduces the problem space to a small subset.

In [8], it is shown that retaining a separate set of beliefs for the others perspective can also be used to model dynamic environments. That is, objects can be moved during the interaction. This has been used to take the perspective of the other even if the beliefs of the self and the other diverge.

In a similar fashion, Johnson and Demiris [15] use an internal simulation of possible motor commands to gain insight to the mental state of another robot. Interestingly, both the perspective taker as well as the target of the PT are robots. Using a list of coupled inverse and forward models, the authors show that PT can be used to determine the applicability of models from the others perspective. That is, the probabilities of models which cannot be applied by the other robot are reduced. This has been shown to increase the accuracy of action recognition [5].

Pandey *et al.* [4] focus on human-robot interactions and teach a robot what it means to make an object visible or accessible. This needs PT abilities as the reach-ability of an object has to be determined from the others perspective. Interestingly, their robot is able to detect the effort class of a human to reach an object, from no effort needed, to whole body effort. A recent summary of their groups work on PT is reported in [2], highlighting the need of PT in human aware robotic systems. The authors argue that one of the most important reasons for the need of PT in robots is that humans frequently change perspectives when describing locations.

The previously mentioned papers all demonstrate the importance and impact of PT in robotics. However, they are not suitable for our objective as the environment they can be used in is constrained. [2] and [4] use motion capture systems to detect humans and objects. While this provides accurate location information, motion capture systems are expensive and need precise calibration. In addition, as the environment must be known in advance, the applicability of these systems is limited. The object detection of [9] and [14] is relying on color blob segmentation, and thus can only detect few objects which have to be uniformly colored. Furthermore, their system only takes the body direction, but not the gaze of humans into account. In [5, 8, 15], optical markers are needed to compute the poses of the target robot and objects. In the next section, we present algorithms to overcome some of the limitations identified in these works.

III. Markerless Perception of Unconstrained Environments

As seen in Section II, previous works are constrained to environments equipped with markers. Our goal is to estimate the perceived world of humans and the surroundings of the robot, whilst not constraining the environment. To this end, we use only cameras mounted on the robot, namely the iCub eye cameras as well as a low-cost, calibration free RGB-D camera.

A. Notations

The point cloud $\mathbf{D}^{\mathcal{K}}$ acquired by the RGB-D camera is represented by the cloud points $p_c^{\mathcal{K}} \in \mathbf{D}^{\mathcal{K}}$. Superscripts are used throughout the paper to denote reference frames; here $\mathcal{K}$ denotes the reference frame of the RGB-D camera.

Let $\mathbf{O}^{\mathcal{R}} = \{(o_1^{\mathcal{R}}, n_1), \ldots, (o_N^{\mathcal{R}}, n_N)\}$ be a set of objects. $o_i^{\mathcal{R}} \in \mathbb{R}^3$ denotes the object location in the robot coordinate frame $\mathcal{R}$. N is the total number of objects perceived by the robot, and n_i contains the corresponding object class to object i.

We define the head set $\mathbf{H}^{\mathcal{K}} = \{h_1^{\mathcal{K}}, \ldots, h_M^{\mathcal{K}}\}$ containing the head poses $h_j^{\mathcal{K}} \in \mathbb{R}^6$ of the M humans interacting with the robot. Each $h_j^{\mathcal{K}}$ contains the position of the j^{th} head in the RGB-D camera coordinate frame, and the corresponding head orientation in yaw, pitch, and roll notation.

The elements p_{ij} of matrix $\mathbf{P}$ with size $N \times M$ store the perception of object i by agent j. Each $p_{ij} \in \mathbf{P}$ is a 3-tuple

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

$(o_i^{\mathcal{H}_j}, S_i^{\mathcal{H}_j}, L_i^{\mathcal{H}_j})$, where $o_i^{\mathcal{H}_j}$ is the i^{th} object in the frame of reference $\mathcal{H}_j$ of the j^{th} human, $S_i^{\mathcal{H}_j} \in \{\text{visible, occluded}\}$ describes whether the i^{th} object is in sight of the j^{th} human, and similarly $L_i^{\mathcal{H}_j} \in \{\text{left, central, right}\}$ encodes the spatial location, *i.e.* whether an object is left, central or right as seen from the human perspective.

For all coordinate frames used throughout the paper, we use the convention that positive values on 1) the x-axis point forwards, 2) the y-axis point to the left, and 3) the z-axis point upwards (see Fig. 2). Distances are described in meters, and angles in degrees.

B. Environment Mapping

We have chosen real-time appearance based mapping (RTAB-Map) [11] to map the environment as it has two advantages over typical visual Simultaneous Localization and Mapping (SLAM) methods. Firstly, RTAB-Map can meet real-time constraints even for large maps, which is important for robots to operate in complex environments. Secondly, RTAB-Map not only makes use of the RGB-D images, but optionally also of the odometry and laser information provided by the mobile base of the iCub. This prevents the loss of odometry in case of fast camera movements.

In Section V-A (visual PT level 2), we will show that the mapped environment can be used to approximate the view of the humans which interact with the robot. However, the environment mapping is optional for level 1 PT (Section IV), and spatial PT level 2 (Section V-B).

The algorithm is running online, and takes the most recently captured point cloud $\mathbf{D}^{\mathcal{K}}(t = t_{now})$ as input. A Bayesian filter is used to determine whether the current location was visited before, which increases robustness on long mapping sessions. Optionally, laser scans and the estimated odometry of the wheels can be provided, which allows the algorithm to recover in the case two consecutive point clouds do not have enough visual words in common.

The output of the algorithm is the 3D space $\mathbf{\Omega}^{\mathcal{K}}$ in the reference frame of the RGB-D camera. $\mathbf{\Omega}^{\mathcal{K}}$ contains the concatenated point clouds $\mathbf{D}^{\mathcal{K}}(t = t_0), \ldots, \mathbf{D}^{\mathcal{K}}(t = t_{now})$, after a) taking the robot movements into account, and b) removing duplicate points. Fig. 3 shows the resulting point cloud after moving the mobile base of the iCub around a table in a typical lab environment.

C. Object Recognition

The object recognition pipeline is based on the recent work by Pasquale *et al.* [12]. The authors port a deep learning framework on the iCub robot. It allows learning and classifying objects online, with one shot learning. The camera image of the left iCub eye camera is taken as input, and blobs representing objects are extracted based on the luminosity of the image. For each object, a vector representation of the image is computed using the output of the highest layer of the deep convolutional network. This representation is somewhat invariant to changes in scale, lightness, and orientation. The classification to one of the object classes is done using a support vector machine.

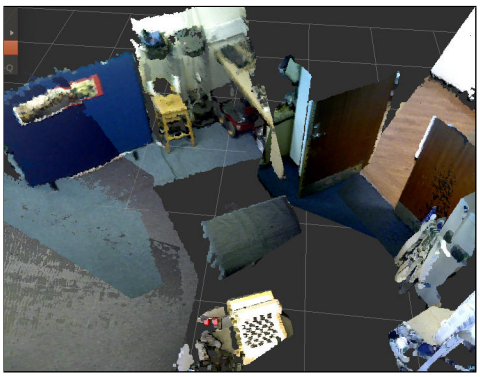

Fig. 3. Map of the environment after turning the iCub 360 degrees on the spot in a lab environment (view from above). The robot uses this information to reason about parts of the environment which cannot be perceived at the moment, but have been seen previously.

Once the object class is known, the stereo vision system [16] of the iCub is used to estimate the object location in the reference frame $\mathcal{E}$ of the left iCub eye. Then, using the robot kinematics the transformation $T_{\mathcal{E}\rightarrow\mathcal{R}}$ to the robot root reference frame $\mathcal{R}$ is computed, and the set $\mathbf{O}^{\mathcal{R}}$ is filled.

As the object recognition pipeline runs in real-time, objects can be moved during human-robot interactions by both the human as well as the robot while the object is still being recognized.

D. Head Pose Estimation

Accurate and robust estimation of the head poses $\mathbf{H}^{\mathcal{K}}$ of surrounding humans is crucial in order to take their perspective. If a head pose is not accurately estimated, the robot's judgments regarding the perception of the humans might be incorrect. Please note that we focus on the head pose as an approximation of the gaze, *i.e.* we do not track the eye movements.

Previous works [2, 4, 17] tackled the head pose estimation using motion capture systems. These provide accurate estimates, but have the disadvantage that humans need to wear a helmet (or similar) equipped with markers, which might partially occlude the field of view and requires a precise calibration. In contrast, our approach is based on camera images, which does not require humans to wear additional equipment, but comes at the cost of information with increased noise.

1) Camera Based Head Pose Estimation: Recent advancements in the area of computer vision allow the estimation of head poses using a single depth image acquired by a RGB-D camera in real-time [13]. On the dataset of the authors, the position error is around 15 ± 22mm, and the error of the angles around $8.5 \pm 13.5°$. The algorithm is based on discriminative random regression forests. The forest contains a number of trees, and the trees are learned in a way that within each node of the tree the variance of the head pose is reduced. The output of the forest is the mean of the predictions of the individual trees. To work with noisy depth data acquired by the RGB-D camera, the trees also classify whether a patch given as input belongs to a head. If that is the case, the output of the tree is considered for the

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

mean calculation.

To train the forest, a user specific 3D morphable model is needed. The ground truth data is generated using an iterative closest point algorithm. In [13], a dataset of 20 people is captured to train the random forest. The subjects were sitting in a distance of 1 meter straight in front of the camera. Note that no user specific model is required for the testing phase.

2) New Method Based on Normalization of Depth Data: As mentioned earlier, the accuracy of the algorithm works well on the testing set of the dataset. However, the input to the algorithm in our scenario differs largely from that of the dataset of [13]. The camera is not straight in front of the human, but comes with a large translational offset in z-direction and is tilted by an angle θ around the z-axis. Furthermore, the distance between the camera and the human(s) is larger than one meter. As we show in Section VI-A, these factors severely decrease the accuracy of the algorithm.

In our new approach, rather than building a new training set matching our environment, we normalize the depth image provided to the algorithm to match that of the expected input. Then, we achieve a performance similar to that of the testing set even if the human is located far away from the training position. Note that this normalization is independent of our presented PT system, and we expect performance improvements for any application where the setup largely differs from that used in [13].

An initial improvement is achieved by transforming the point cloud $\mathbf{D}^{\mathcal{K}}$ into a new reference frame $\mathcal{P}$, resulting in $\mathbf{D}^{\mathcal{P}} = T_{\mathcal{K}\to\mathcal{P}}\mathbf{D}^{\mathcal{K}}$. The axes of $\mathcal{P}$ are chosen to be aligned with that of $\mathcal{R}$, except of a translational offset such that the head of the human is one meter away from the origin of $\mathcal{P}$. Thus, the transformation matrix $T_{\mathcal{K}\to\mathcal{P}}$ is derived as follows[1]:

$$T_{\mathcal{K}\to\mathcal{P}} = T_{\mathcal{K}\to\mathcal{R}}T_{\mathcal{R}\to\mathcal{P}} = T_{\mathcal{K}\to\mathcal{R}}\begin{pmatrix} 1 & 0 & 0 & x_{\mathcal{R}\to\mathcal{P}} \\ 0 & 1 & 0 & y_{\mathcal{R}\to\mathcal{P}} \\ 0 & 0 & 1 & z_{\mathcal{R}\to\mathcal{P}} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1)$$

Then, the output $\mathbf{H}^{\mathcal{P}} = \{h_1^{\mathcal{P}}, \ldots, h_M^{\mathcal{P}}\}$ of the random forest is in the $\mathcal{P}$ reference frame, and needs to be transformed back in the initial frame of reference $\mathcal{K}$:

$$h_j^{\mathcal{K}} = {T_{\mathcal{K}\to\mathcal{P}}}^{-1} h_j^{\mathcal{P}} \qquad \forall j \in \{1, \ldots, M\}. \quad (2)$$

A further increase of accuracy is achieved by employing the head position $\mathfrak{h}^{\mathcal{K}}_{\text{Skeleton}} \in \mathbb{R}^3$ returned by the RGB-D skeleton tracking[2] (note that the skeleton information only includes the head position, but not orientation). This step is optional, as it requires the users to go into a calibration pose to start the skeleton tracking. The same principle as above is used, but instead of using a fixed distance between robot root and the origin of frame $\mathcal{P}$ as above, the origin of frame $\mathcal{P}^*$ is chosen to be one meter away from the initial head position $\mathfrak{h}^{\mathcal{K}}_{\text{Skeleton}} \in \mathbb{R}^3$, and the rotation of the coordinate axes of $\mathcal{P}^*$ coincide with that of the robot frame $\mathcal{R}$. For this, $\mathfrak{h}^{\mathcal{K}}_{\text{Skeleton}}$ is first transformed to the robot frame $\mathcal{R}$: $\mathfrak{h}^{\mathcal{R}}_{\text{Skeleton}} = T_{\mathcal{K}\to\mathcal{R}}\mathfrak{h}^{\mathcal{K}}_{\text{Skeleton}}$. Then, $T_{\mathcal{R}\to\mathcal{P}^*}$ is computed as in Eq. 1, but with the following substitutions:

$$x_{\mathcal{R}\to\mathcal{P}}{}^* = \mathfrak{h}^{\mathcal{R}}_{\text{Skeleton},x} - 1.0$$
$$y_{\mathcal{R}\to\mathcal{P}}{}^* = \mathfrak{h}^{\mathcal{R}}_{\text{Skeleton},y} \qquad z_{\mathcal{R}\to\mathcal{P}}{}^* = \mathfrak{h}^{\mathcal{R}}_{\text{Skeleton},z}. \quad (3)$$

Again, the random forest algorithm is used to estimate the head pose in the new reference frame $\mathcal{P}^*$, and ${T^*_{\mathcal{K}\to\mathcal{P}}}^{-1}$ is used to transform the resulting pose back in the reference frame of the RGB-D camera $\mathcal{K}$:

$$h_j^{\mathcal{R}} = {T_{\mathcal{K}\to\mathcal{P}^*}}^{-1} h_j^{\mathcal{P}^*} \qquad \forall j \in \{1, \ldots, M\}. \quad (4)$$

If $\mathfrak{h}^{\mathcal{K}}_{\text{Skeleton}}$ is available, we use Eq. 4 to compute $h_j^{\mathcal{K}} \in \mathbf{H}^{\mathcal{K}}$, otherwise we fall back to the default normalization using $T_{\mathcal{K}\to\mathcal{P}}$ (Eq. 2). The performance improvement using the described extensions to the original head pose estimation algorithm are crucial for a markerless PT pipeline. The normalization step is shown in Fig. 4, and experimental results can be found in Section VI-A.

Fig. 4. Effect of the depth image normalization. The non-normalized input on the left differs largely from the training data, as the camera is mounted on an angle and too far from the subject. The normalized image after transformation on the right is more similar to the training data (subject is facing straight and is closer), which improves the performance of the head pose estimator as shown in Section VI-A.

E. Transform RGB-D Camera Frame to Robot Frame

The transformation $T_{\mathcal{K}\to\mathcal{R}}$ from the RGB-D camera coordinate frame $\mathcal{K}$ to the robot coordinate frame $\mathcal{R}$ is found as follows. First, $T_{\mathcal{K}\to\mathcal{R}}$ is initialized using the roughly known geometrical information between the RGB-D camera and the robot root. There is only one rotational component with angle θ around the z-axis, and three unknown translational components $x_{\mathcal{K}\to\mathcal{R}}$, $y_{\mathcal{K}\to\mathcal{R}}$, $z_{\mathcal{K}\to\mathcal{R}}$:

$$T_{\mathcal{K}\to\mathcal{R}} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 & x_{\mathcal{K}\to\mathcal{R}} \\ \sin(\theta) & \cos(\theta) & 0 & y_{\mathcal{K}\to\mathcal{R}} \\ 0 & 0 & 1 & z_{\mathcal{K}\to\mathcal{R}} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (5)$$

The final angle θ^* of $T_{\mathcal{K}\to\mathcal{R}}$ is found such that the x-axis of $\mathcal{R}$ aligns with the floor points in $\mathbf{D}^{\mathcal{K}}$ (except of a translational offset in z-direction). Then, using ${T_{\mathcal{K}\to\mathcal{R}}}^{-1}$, we transform the objects $\mathbf{O}^{\mathcal{R}}$ in the reference frame $\mathcal{K}$ to visualize them along the point cloud $\mathbf{D}^{\mathcal{K}}$ in the RGB-D camera reference frame $\mathcal{K}$:

$$\mathbf{O}^{\mathcal{K}} = {T_{\mathcal{K}\to\mathcal{R}}}^{-1}\mathbf{O}^{\mathcal{R}}. \quad (6)$$

The translational components are then changed stepwise such that the transformed object markers $\mathbf{O}^{\mathcal{K}}$ visually match with

[1] $T_{\mathcal{K}\to\mathcal{R}}$ is derived in Eq. 5

[2] PrimeSense NiTE (http://www.openni.ru/) is used for skeleton tracking

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

the corresponding objects in the point cloud, at which point the final values $x^*_{\mathcal{K}\to\mathcal{R}}$, $y^*_{\mathcal{K}\to\mathcal{R}}$ and $z^*_{\mathcal{K}\to\mathcal{R}}$ are found.

Instead of this manual procedure, an iterative closest point algorithm could be used to find this transformation automatically, which would also make a dynamic camera position feasible. However, as in our system $T_{\mathcal{K}\to\mathcal{R}}$ remains static, we found an automatic system not necessary.

IV. Level 1 Perspective Taking: Which Objects Can You See?

In the previous section, we introduced the markerless perception of the environment, including the estimation of the head pose of humans and the locations of objects. In this section, we introduce level 1 perspective taking (PT), which is the ability to know which objects are seen by others. Imagine a situation where the robot is asked to grasp "the toy", but the robot recognizes two toys, leading to an ambiguous situation. However, if the robot knows that only one of the toys can be seen by the human, the robot can infer which toy is meant.

In previous works, the robot mentally transformed its perspective to that of the other robot [5, 8, 15] or human [2, 4] to judge the visibility of objects from their perspective[3]. This approach is not desirable in our system, as it needs a very accurate representation of the objects from a non-egocentric viewpoint in order to recognize the objects. Previously, this was achieved using optical markers, however in this work we implement a markerless approach. We present a simpler and faster way of judging the visibility. Our approach is inspired by cognitive research, which suggest that a) there are two different processes for PT [3], and b) the process for level 1 PT is based on line of sight tracing [6]. Level 2 PT is shown in Section V.

A. Visual Perspective Taking Level 1

We base our level 1 PT algorithm on a line of sight tracing approach presented by Amanatides and Woo [18], which remains the foundation for most works on line of sight tracing to date [19]. We first map the points $p_c^{\mathcal{K}} \in \mathbf{D}^{\mathcal{K}}$ captured by the depth camera in a grid of voxels. The voxel grid $\mathbf{V}^{\mathcal{K}}$ approximates the 3D space spanned by the points $p_c^{\mathcal{K}}$ to volume items $v_c^{\mathcal{K}}$ of equal shape (here: cubes of dimension ξ^3), such that:

$$\mathbf{V}^{\mathcal{K}} = \cup_{c=1}^{V} v_c^{\mathcal{K}} \qquad v_i^{\mathcal{K}} \cap v_j^{\mathcal{K}} = \emptyset \qquad \forall i,j,\ i \neq j. \quad (7)$$

The coordinates of a point $p_c^{\mathcal{K}} \in \mathbb{R}^3$ in 3D space are mapped to its equivalent voxel $v_c^{\mathcal{K}} \in \mathbb{Z}^3$ as follows:

$$v_c^{\mathcal{K}} = \lfloor p_c^{\mathcal{K}} / \xi \rfloor. \quad (8)$$

Using Eq. 8, we compute the coordinates of the head poses $\mathbf{H}^{\mathbf{V},\mathcal{K}}$ and object locations $\mathbf{O}^{\mathbf{V},\mathcal{K}}$ in the voxel grid.

The line of sight is traced between each human $j = 1,\ldots,M$ and each object $i = 1,\ldots,N$, leading to $N \times M$ traces. The tracing is performed as follows. A trace $T_{j\to i}$ starts at the nose tip $h_j^{\mathbf{V},\mathcal{K}}$ of the j^{th} human, and its target is the object at location $o_i^{\mathbf{V},\mathcal{K}}$. At each step, depending on the offset of the current voxel and the target voxel, a decision is made whether the next voxel to be traversed is one step towards the x, y, or z direction. If the traversed voxel contains a point, the algorithm returns with the result that the object is hidden. If the traversed voxel is closer to the target voxel than a given threshold δ, the algorithm returns with the result that the object is visible to the human. The pseudocode is described in Algorithm 1.

Algorithm 1: Level 1 Perspective Taking

Input : Voxel grid $\mathbf{V}^{\mathcal{K}}$ with leaf size ξ
Origin point $\mathbf{O}^{\mathcal{K}}$ and target point $\mathbf{T}^{\mathcal{K}}$
Output: Visibility of $\mathbf{T}^{\mathcal{K}}$ from $\mathbf{O}^{\mathcal{K}}$

direction $\leftarrow (\mathbf{T}^{\mathcal{K}} - \mathbf{O}^{\mathcal{K}})/\|\mathbf{T}^{\mathcal{K}} - \mathbf{O}^{\mathcal{K}}\|$
$\mathbf{O}^{\mathcal{K}}_{\mathbf{V}}$ $\leftarrow$ `3DtoVoxel(`$\mathbf{O}^{\mathcal{K}}$`)`
voxel$_{max}$ $\leftarrow$ `VoxelTo3D(`$\mathbf{O}^{\mathcal{K}}_{\mathbf{V}}$`)`
for $i \in \{x, y, z\}$ **do**
  if *direction*$[i] < 0.0$ **then**
    step$[i] \leftarrow -1$; voxel$_{max}[i] \leftarrow$ voxel$_{max}[i] - \xi/2$
  else
    step$[i] \leftarrow +1$; voxel$_{max}[i] \leftarrow$ voxel$_{max}[i] + \xi/2$
  $t_\delta[i]$ $\leftarrow \xi/|$direction$[i]|$
  $t_{max}[i] \leftarrow$ (voxel$_{max}[i] - \mathbf{O}^{\mathcal{K}}$)/direction$[i]$
while $\mathbf{O}^{\mathcal{K}}_{\mathbf{V}} \in \mathbf{V}^{\mathcal{K}}$ **do**
  if $\|\mathbf{O}^{\mathcal{K}}_{\mathbf{V}} - \mathbf{T}^{\mathcal{K}}\| < \delta$ **then**
    return *visible*
  if `isOccluded(`$\mathbf{O}^{\mathcal{K}}_{\mathbf{V}}$`)` **then**
    return *occluded*
  if $t_{max}[x] \leq t_{max}[y]$ *and* $t_{max}[x] \leq t_{max}[z]$ **then**
    $t_{max}[x] \leftarrow t_{max}[x] + t_\delta[x]$
    $\mathbf{O}_{\mathbf{V}}[x]$ $\leftarrow \mathbf{O}_{\mathbf{V}}[x]$ + step$[x]$
  else if $t_{max}[y] \leq t_{max}[x]$ *and* $t_{max}[y] \leq t_{max}[z]$ **then**
    $t_{max}[y] \leftarrow t_{max}[y] + t_\delta[y]$
    $\mathbf{O}^{\mathcal{K}}_{\mathbf{V}}[y]$ $\leftarrow \mathbf{O}^{\mathcal{K}}_{\mathbf{V}}[y]$ + step$[y]$
  else
    $t_{max}[z] \leftarrow t_{max}[z] + t_\delta[z]$
    $\mathbf{O}^{\mathcal{K}}_{\mathbf{V}}[z]$ $\leftarrow \mathbf{O}^{\mathcal{K}}_{\mathbf{V}}[z]$ + step$[z]$

The result of the tracing, being either "visible" or "occluded", is stored in the variables $S_i^{\mathcal{H}_j}$ of matrix $\mathbf{P}$ (see Section III-A). A visualization of the tracing is shown in Fig. 6. Interestingly, the execution time of our approach follows qualitatively the reaction times found in humans [6].

V. Level 2 Perspective Taking: What Does the World Look Like to You?

In this section, we describe how level 2 perspective tasks are solved in our system. We differentiate two tasks: 1) estimating how the world is visually perceived by a human (visual PT), and 2) judging whether objects are to the left, right or in front of the human (spatial PT). We solve the visual PT task by transforming the concatenated point cloud $\mathbf{\Omega}^{\mathcal{K}}$ (see Section III-B) from the camera reference frame $\mathcal{K}$ in

[3] A notable exception is [17], where a line of sight tracing similar to ours is used. However, [17] employs a motion capture system.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

the reference frame of the j^{th} human $\mathcal{H}_j$, whose visualization leads to a reconstructed view from the human's perspective. Once the point cloud is in the reference frame of the human, a simple case differentiation is used to solve the spatial PT task.

A. Visual Perspective Taking Level 2

The view of the j^{th} human is estimated by first converting $h_j^{\mathcal{K}} \in \mathbb{R}^6$ from Euler angles in the RGB-D coordinate frame $\mathcal{K}$ (see Section III-A) to a transformation matrix, and afterwards using this transformation matrix to convert $\mathbf{\Omega}^{\mathcal{K}}$ in the new reference frame $\mathcal{H}_j$. We decompose $h_j^{\mathcal{K}}$ into its components:

$$h_j^{\mathcal{K}} = (h_{j,x}^{\mathcal{K}}, h_{j,y}^{\mathcal{K}}, h_{j,z}^{\mathcal{K}}, h_{j,\alpha}^{\mathcal{K}}, h_{j,\beta}^{\mathcal{K}}, h_{j,\gamma}^{\mathcal{K}}). \tag{9}$$

The angles $h_{j,\alpha}^{\mathcal{K}}$, $h_{j,\beta}^{\mathcal{K}}$, and $h_{j,\gamma}^{\mathcal{K}}$ are yaw, pitch and roll angles, *i.e.* the first rotation is described by $h_{j,\gamma}^{\mathcal{K}}$ about the x-axis ($R_x(\gamma)$), the second rotation by $h_{j,\beta}^{\mathcal{K}}$ about the y-axis ($R_y(\beta)$), and the third rotation by $h_{j,\alpha}^{\mathcal{K}}$ about the z-axis ($R_z(\alpha)$) [20, p. 99]. The 3x3 rotation matrix $R_{\mathcal{K}\to\mathcal{H}_j}$ is obtained as follows:

$$R_{\mathcal{K}\to\mathcal{H}_j} = R_z(\alpha)R_y(\beta)R_x(\gamma). \tag{10}$$

Finally, the homogeneous transformation $T_{\mathcal{K}\to\mathcal{H}_j^{\mathcal{K}}}$ with rotational component $R_{\mathcal{K}\to\mathcal{H}_j}$ and translational component $t_j^{\mathcal{K}} = (h_{j,x}^{\mathcal{K}}, h_{j,y}^{\mathcal{K}}, h_{j,z}^{\mathcal{K}})^\top$ is calculated as:

$$T_{\mathcal{K}\to\mathcal{H}_j^{\mathcal{K}}} = \begin{pmatrix} R_{\mathcal{K}\to\mathcal{H}_j^{\mathcal{K}}} & t_j^{\mathcal{K}} \\ 0 & 1 \end{pmatrix}. \tag{11}$$

The transformation matrix is then used to obtain a point cloud whose origin coincides with that of the nose tip of the j^{th} human:

$$\mathbf{\Omega}^{\mathcal{H}_j} = T_{\mathcal{K}\to\mathcal{H}_j^{\mathcal{K}}}\mathbf{\Omega}^{\mathcal{K}} \tag{12}$$

The resulting point cloud $\mathbf{\Omega}^{\mathcal{H}_j}$ can then be visualized, and contains an approximated view of what the human is seeing. An example of a reconstructed view is shown in Fig. 7, and more discussion can be found in Section VI-C. In comparison to earlier works [2, 4, 5, 8, 15], the transformed view is solely based on the images acquired from the RGB-D camera during the environment mapping (see Section III-B), rather than an *a priori* known virtual environment. In the next section, we will show that the transformed cloud can also be used for spatial reasoning.

B. Spatial Perspective Taking Level 2

Spatial PT is the ability of judging the spatial location of an object from another frame of reference. Here, we outline how the transformed point cloud $\mathbf{\Omega}^{\mathcal{H}_j}$ is used for these judgments. Importantly, the judgments are universal, *i.e.* they do not depend on the frame of reference. This means that the iCub is able to transfer the knowledge acquired from an egocentric viewpoint ("the toy is to my left") to that of another viewpoint ("the toy is to his right") without changes in the underlying algorithm. This is not limited to spatial PT, but might also be used for other tasks such as learning by imitation [21], which is beyond the scope of this paper.

We first show a simple spatial reasoning approach which is used by the iCub to judge the object locations from the iCub's viewpoint, and then apply the same algorithm to the transformed view, allowing the robot to reason about the human's visual perception. Remember that $o_i^{\mathcal{R}} \in \mathbb{R}^3$ denotes the object location in the robot reference frame $\mathcal{R}$; and $L_i^{\mathcal{R}} \in \{\text{left, central, right}\}$ describes the spatial relation between object i and the robot. The left/right judgments are made as follows, with $\alpha_i^{\mathcal{R}} = \arctan(o_{i,x}^{\mathcal{R}}/o_{i,y}^{\mathcal{R}})$ and $\theta = 7.5°$:

$$L_i^{\mathcal{R}} = \begin{cases} \text{left} & \text{if } \alpha_i^{\mathcal{R}} > +\theta \\ \text{right} & \text{if } \alpha_i^{\mathcal{R}} < -\theta \\ \text{central} & \text{otherwise.} \end{cases} \tag{13}$$

As mentioned before, spatial PT is done in the same way, with the only difference being the angle which is given as the input. Using Equations 5 and 11, the object location $o_i^{\mathcal{R}}$ is transformed in the reference frame of human j:

$$o_i^{\mathcal{H}_j} = {T_{\mathcal{K}\to\mathcal{R}}}^{-1} T_{\mathcal{K}\to\mathcal{H}_j} o_i^{\mathcal{R}}. \tag{14}$$

Then, we provide the angles $\alpha_i^{\mathcal{H}_j} = \arctan(o_{i,x}^{\mathcal{H}_j}/o_i^{\mathcal{H}_j})$ to the algorithm in Eq. 13, and use the return values to fill $L_i^{\mathcal{H}_j}$ of the perception matrix $\mathbf{P}$. Albeit simple, this is a powerful approach to allow the iCub to take the spatial perspective of humans. The knowledge is then used by the iCub to refer to objects as seen by the human, *e.g.* "I want to play with the toy on your left".

VI. Experimental Evaluation

We utilize an iCub humanoid robot mounted on an iKart mobile base, with a RGB-D camera attached $z_{\mathcal{K}\to\mathcal{R}} = 57.5$ cm centrally above the robot root frame $\mathcal{R}$, at an angle of $\theta = 26°$ downward facing (see Fig. 2). As the RGB-D camera, we use an ASUS Xtion Live, which provides RGB and depth images with a resolution of 640×480 pixels. This allows the iCub to observe objects placed in front of it on a table, as well as to observe one or two humans sitting at the other end of the table. You may want to watch the accompanying video for a demonstration of our experimental results.

A. Head Pose Estimation with Applied Normalization

Our intention is to apply a pre-trained state of the art head pose estimator in a scenario which differs largely from the training input. As discussed previously, we extended the algorithm by normalizing the input data, and we predicted that it becomes more viewpoint invariant. To investigate, we compared the original algorithm with the extended algorithm on six subjects with each five different poses, capturing a wide range of horizontal angles. Fig. 5 shows our method, compared with the original method. All other parameters were kept constant, and both methods were applied to the same input image. Without normalizing the input, the resulting head poses were center-biased for all subjects. For the first and third subject, the original method was not able to estimate a head pose for large angles, whereas our improved method returned accurate estimates. Normalizing the input data also leads to a better estimate of the vertical angle, *i.e.* whether the subject looks up or down (not shown for brevity). For the sixth subject, the way the subject's hair had fallen across his face resulted in incorrect estimates for both the original, and our method.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

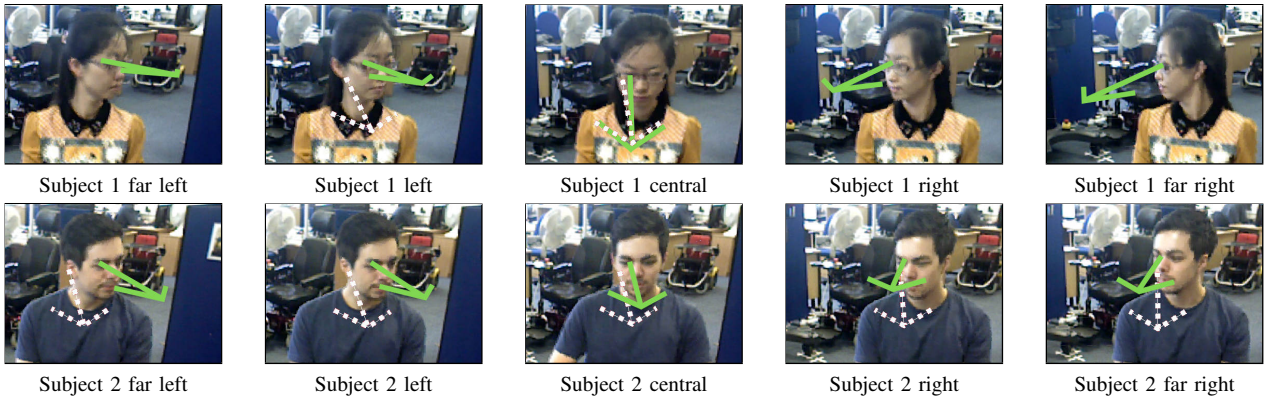

Fig. 5. Comparison of the normalized head pose algorithm (green arrows) and original method (white, dotted arrows). The normalized method leads to much higher accuracy of the head pose estimation for all head rotations. For three rotations of subject 1, the normalized method resulted in correct pose estimates whereas the original method failed to detect the head.

B. Level 1 Perspective Taking Performance

In order to judge which objects are seen by the humans, we proposed a line of sight tracing in Section IV. We reasoned that this is a suitable algorithm in a markerless scenario, where object recognition in the transformed view of the human is a yet to be solved problem. Here we show the effectiveness of our proposal in three different scenarios. In all scenarios, $N = 3$ different objects belonging to one of $C = 7$ object classes are placed on a table which is situated between the iCub and the subject (a clock, a joystick and a pen). In Fig. 6, small spheres are used to visualize the traced line of sight, and big spheres are used to denote the object locations. In the first case, all objects can be seen by the subject. In the second case, one object is hidden from the subject, and the tracing stops at the barrier. Similarly, the third case shows two occluded objects. For a better comparison, the actual view of the subject in this case is shown in Fig. 6(d). In all scenarios, the line of sight tracing algorithm successfully determined which objects can be seen, demonstrating the robustness of our approach.

C. Level 2 Perspective Taking Performance

We proposed to mentally rotate the point cloud acquired by the environment mapping in the frame of reference of the human to estimate what the world looks like to the human. Furthermore, we showed that rotating the point cloud allows using the same spatial reasoning algorithm as from the robots perspective. Here we validate this proposal in a similar setup as the level 1 PT experiments.

In the first experiment, we place three objects on a table between the iCub and the subject: a joystick to the left of the subject, a toy in the center, and a cup to the right of the subject. First we mentally transform the point cloud in the reference frame of the human, which is shown in Fig. 7. Albeit being a low resolution approximation due to the RGB-D camera, the gist of the scene is comprehensible. Importantly, the iCub uses the mapped environment to reason about areas of the scene which cannot currently be perceived by the robot.

Fig. 7. Approximated view of the human using a mental transformation. The robot, while facing the human, correctly estimates that the human is looking at a table with three objects. Also, as the robot has a map of the environment, it can reason about areas of the environment which are currently perceived by the human, but not by the robot.

In a second experiment, we ask the subject to look at a specific object, and apply the spatial reasoning algorithm. For example, using the setup in Fig. 6(a), we asked the subject to look at the clock. The output of the spatial reasoning was as follows: $L_{\text{pen}}^{\mathcal{H}_j} = \text{left}$, $L_{\text{joystick}}^{\mathcal{H}_j} = \text{center}$ and $L_{\text{clock}}^{\mathcal{H}_j} = \text{right}$. Similarly, in the other scenarios, the spatial reasoning determined the object location from the human's view correctly.

We conducted a third experiment to determine the accuracy of our spatial reasoning algorithm in situations where the objects are further away from the subject, and at wider angles. We asked six subjects to focus on five points with varying distance and horizontal angle from the subject (far

(a) All objects visible (b) One object occluded (c) Two objects occluded (d) View from subject for c)

Fig. 6. Level 1 perspective taking in different scenarios. The visible objects are correctly inferred in all scenarios.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE International Conference on Robotics and Automation (2016)
Published by: IEEE

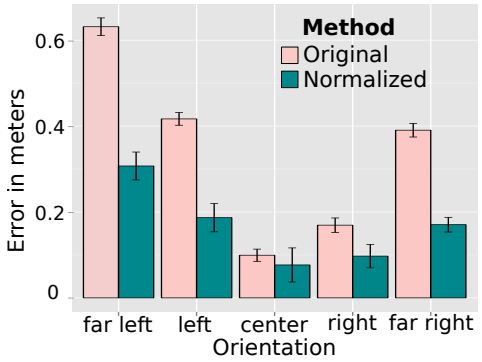

Fig. 8. Horizontal error for spatial perspective taking, using the original (red) and normalized (green) head pose estimation algorithms.

left / far right focal points: 1.19m from the subject at $\pm 33^\circ$ angle; left / right focal points: 1.07m at $\pm 20^\circ$ angle; center focal point: 1.0m distance). All focal points were level with the heads of the subjects. In Fig. 8, we show that spatial PT using the normalized algorithm is significantly more accurate ($p < 0.01$), allowing the robot to determine which object is focused at a reasonably high accuracy even at large angles and distances.

VII. Conclusion and Future Works

In this paper we have introduced a novel framework which allows a robot to take the perspective of surrounding humans. The combined improvements in key parts of the visuospatial perspective taking pipeline have led to a system that works in markerless setups. The system was validated in several experiments using an iCub humanoid robot. To estimate the head pose, we propose a new method which normalizes the input so it becomes more similar to the training data. This improves the performance in scenarios where the pre-normalized input data is dissimilar to the training data, which extends the application scenarios of the head pose estimator.

We employ line of sight tracing for level 1 perspective taking, to determine whether an object is visible to the human, and highlighted that previous methods are not suitable for a markerless environment. For level 2 perspective taking, a mental perspective transformation is used to reconstruct the world from another viewpoint, whereby the robot does not have any prior information about the world and is learning the environment online. We demonstrated that the robot can judge whether objects are to the left, right or in front of a human using the same algorithm as from an egocentric perspective.

Previous works need artificial markers and/or motion capture systems, which constrains their usability. In contrast, our system can be applied to any environment, *e.g.* care homes where a robot might aid elderly people by describing object locations in their frame of reference ("the remote control is to your left").

Our framework would benefit from more accurate gaze estimates by taking the eye movements of the human into account. Also, the human field of view should be taken into consideration for level 1 PT.

Our proposed method overcomes limitations in the perceptional part of perspective taking, and we plan to integrate it with a higher-level cognitive system [22]. This will allow an insight to the developmental process of perspective taking [23], as well as to solve more complex perspective taking tasks. Furthermore, we plan to investigate the relationship between joint attention [24] and perspective taking.

References

[1] M. A. Goodrich and A. C. Schultz, "Human-Robot Interaction: A Survey," *Found. Trends Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.
[2] S. Lemaignan, M. Warnier, E. A. Sisbot, and R. Alami, "Human-Robot Interaction: Tackling the AI Challenges," *Artif. Intell.*, 2014.
[3] J. H. Flavell, B. A. Everett, K. Croft, and E. R. Flavell, "Young Children's Knowledge About Visual Perception: Further Evidence for the Level 1-Level 2 Distinction," *Dev. Psychol.*, vol. 17, no. 1, pp. 99–103, 1981.
[4] A. K. Pandey, M. Ali, and R. Alami, "Towards a Task-Aware Proactive Sociable Robot Based on Multi-state Perspective-Taking," *Int. J. Soc. Robot.*, vol. 5, no. 2, pp. 215–236, 2013.
[5] M. Johnson and Y. Demiris, "Perceptual Perspective Taking and Action Recognition," *Int. J. Adv. Robot. Syst.*, vol. 2, no. 4, pp. 301–308, 2005.
[6] P. Michelon and J. M. Zacks, "Two kinds of visual perspective taking," *Perception & Psychophysics*, vol. 68, no. 2, pp. 327–337, 2006.
[7] C. Breazeal *et al.*, "Using perspective taking to learn from ambiguous demonstrations," *Rob. Auton. Syst.*, vol. 54, no. 5, pp. 385–393, 2006.
[8] M. Johnson and Y. Demiris, "Visuo-Cognitive Perspective Taking for Action Recognition," in *Int. Symp. Imitation Animimals Artifacts*, 2007, pp. 262–269.
[9] W. G. Kennedy, M. D. Bugajska, A. M. Harrison, and J. G. Trafton, ""Like-Me" Simulation as an Effective and Cognitively Plausible Basis for Social Robotics," *Int. J. Soc. Robot.*, vol. 1, no. 2, pp. 181–194, 2009.
[10] G. Metta *et al.*, "The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural Networks*, vol. 23, no. 8-9, pp. 1125–1134, 2010.
[11] M. Labbe and F. Michaud, "Appearance-Based Loop Closure Detection for Online Large-Scale and Long-Term Operation," *IEEE Trans. Robot.*, vol. 29, no. 3, pp. 734–745, 2013.
[12] G. Pasquale *et al.*, "Teaching iCub to recognize objects using deep Convolutional Neural Networks," in *Proc. Work. Mach. Learning Interactive Syst.*, 2015, pp. 21–25.
[13] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real Time Head Pose Estimation from Consumer Depth Cameras," in *Annu. Symp. German Association Pattern Recognition*, 2011, pp. 101–110.
[14] J. G. Trafton *et al.*, "Enabling Effective Human-Robot Interaction Using Perspective-Taking in Robots," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 35, no. 4, pp. 460–470, 2005.
[15] M. Johnson and Y. Demiris, "Perspective Taking Through Simulation," in *Proc. Towards Autonomous Robotic Syst.*, 2005, pp. 119–126.
[16] S. Fanello *et al.*, "3D Stereo Estimation and Fully Automated Learning of Eye-Hand Coordination in Humanoid Robots," in *IEEE-RAS Int. Conf. Humanoid Robotics*, 2014, pp. 1028–1035.
[17] A. K. Pandey and R. Alami, "Mightability Maps: A Perceptual Level Decisional Framework for Co-operative and Competitive Human-Robot Interaction," in *Int. Conf. Intell. Robot. and Syst.*, 2010, pp. 5842–5848.
[18] J. Amanatides and A. Woo, "A Fast Voxel Traversal Algorithm for Ray Tracing," *Eurographics*, vol. 87, no. 3, pp. 3–10, 1987.
[19] S. Laine and T. Karras, "Efficient Sparse Voxel Octrees," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 8, pp. 1048–1059, 2011.
[20] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.
[21] Y. Demiris and M. Johnson, "Distributed, predictive perception of actions: a biologically inspired robotics architecture for imitation and learning," *Connection Sci.*, vol. 15, no. 4, pp. 231–243, 2003.
[22] M. Petit, T. Fischer, and Y. Demiris, "Lifelong Augmentation of Multi-Modal Streaming Autobiographical Memories," *IEEE Trans. Cogn. Develop. Syst.*, 2016, to appear.
[23] N. Newcombe and J. Huttenlocher, "Children's Early Ability to Solve Perspective-Taking Problems," *Developmental Psychology*, vol. 28, no. 4, pp. 635–643, 1992.
[24] S. Boucenna, P. Gaussier, and L. Hafemeister, "Development of joint attention and social referencing," in *IEEE Int. Conf. Develop. and Learning*, 2011.