Artificial Neural Networks For Spatial Perception
Towards Visual Object Localisation in Humanoid Robots

Jürgen Leitner, Simon Harding, Mikhail Frank, Alexander Förster, Jürgen Schmidhuber

Abstract— **In this paper, we present our on-going research to allow humanoid robots to learn spatial perception. We are using artificial neural networks (ANN) to estimate the location of objects in the robot's environment. The method is using only the visual inputs and the joint encoder readings, no camera calibration and information is necessary, nor is a kinematic model. We find that these ANNs can be trained to allow spatial perception in Cartesian (3D) coordinates. These lightweight networks are providing estimates that are comparable to current state of the art approaches and can easily be used together with existing operational space controllers.**

I. Introduction

THE field of robotics has advanced steadily over the last decades. Ever more complex robots with the ability to perform more complex actions have appeared in research labs have become available in industry and to the public. Yet the majority of robotic systems are still mainly performing pre-programmed automation tasks. While e.g. humanoid robots have been proposed to assist with activities of daily living, especially in household tasks and elderly-care, no robust and useful systems are currently available – neither in research nor in industry.

To make robots more robust and adaptable in real-world scenarios a big issue to overcome is the lack of autonomy. The recent progress in autonomous capabilities are needed to be further extended for the future use of robots in interesting settings. An important step to perform autonomous decisions and actions is to perceive the world. Perception though is still a hard problem in robotics.

The need for robust perception, in a variety of settings, led us to investigate machine learning approaches together with computer vision to tackle the impeding problems with humanoid robots. These systems, designed to be human-like, need to heavily rely on sensor systems that are biologically inspired. For example, most humanoid robots, have to use cameras to visually perceive the environment, whereas in mobile robots the use of time-of-flight sensors has become the norm.

The simplicity with which humans and animals interact with their environment is still far from reach for robots. To overcome this gap artificial systems with higher sensorimotor integration are being designed. These robots though have been pushing the limits of traditional control theoretic methods. A coordinated approach to motor planning, control, estimation, prediction and learning is needed.

Spatial perception, that is the detection and localisation of objects in the scene, has recently been of increased interest in robotics. This localisation is needed for on-line motion planning, obstacle avoidance, and object manipulation. While various new sensors have been proposed and used, we herein restrict ourselves to the use of visual information provided by two cameras ('eyes') in the head of the humanoid robot.

In this work, we present a machine learning setup that provides a humanoid robot with a method to estimate the position of objects relative to itself in 3D Cartesian space.

II. Problem Description and Related Work

The development of spatial perception in humans is an active research topic. In humans it develops over time from observation and interaction with the world. Research in the fields of brain- and neuro-science show clear trends on *what* changes during this development of spatial cognitive abilities, *how* these changes happen is not clear [1]. There is, for instance, evidence showing that the metric precision changes systematically over time (from 2-11 year old kids) [2].

Our developmental approach uses machine learning to create a purely vision-based spatial perception. We use Artificial Neural Networks (ANN) to provide our humanoids with the ability to estimate the location of objects perceived by the robot's cameras. There are various approaches of localising objects in use in the field of robotics. An important factor in deciding which techniques to use is the reference frame in which the objects position will be represented. Herein we use the operational space of the robot (3D Cartesian space).

The problem of localising objects in 3D given multiple images from cameras in different locations is widely known in the computer vision literature as 'Stereo Vision' [3]. In the following discussion, we try to explain the problems with stereo vision on humanoid robots, by using our experimental platform, the iCub humanoid robot [4], as an example. *CSL* and *CSR* refer to the local reference frames of the left and right cameras respectively, the reference frame of the body is *CSBody*, but as it is mounted at a fixed point this is also the reference frame chosen for the environment. Therefore *CSWorld* denotes the common environmental reference frame, in which we seek to express object locations. Cameras that photograph the same scene from two different locations provide different 2D projections of the 3D scene. If the 'intrinsic parameters' that specify each camera's projection from 3D to 2D, as well as the 'fundamental matrix' that is the rigid-body transformation between *CSL* and *CSR* are

J. Leitner, M. Frank, A. Förster and J. Schmidhuber are all with the Dalle Molle Institute for Artificial Intelligence (IDSIA), Galleria 2, 6928 Manno, Switzerland (email: juxi@idsia.ch).

S. Harding is with Machine Intelligence Ltd, South Zeal, United Kingdom (email: simon@machineintelligence.co.uk). He was previously at IDSIA.

This work was partly supported by EU contract FP7-IST-IP-231722.

known, and if there are some features of the scene that can be identified in both images, then the 3D locations of those features can be triangulated. Hartley and Zisserman [3] give a thorough review of approaches based on this principle, we refer the interested reader to their book.

While traditional stereo vision approaches, based on projective geometry, have been proven effective under carefully controlled experimental circumstances, they are not ideally suited to most robotics applications. Intrinsic camera parameters and the fundamental matrix may be unknown or time varying, and this requires the frequent repetition of lengthly calibration procedures, wherein known, structured objects are viewed by the stereo vision system, and the required parameters are estimated by numerical algorithms. Assuming a solution to the standard stereo vision problem, applying it to a real physical robot to facilitate object manipulation remains a challenge. In many robotics applications, it is somewhat inconvenient to express the environment with respect to a camera. For example, from a planning and control standpoint, the most logical choice of coordinate system is *CSWorld*, the reference frame at the base of the manipulator, which does not move with respect to the environment. In order to transform coordinates from *CSL* or *CSR* to *CSWorld*, such that we can model objects and control the robot in the same frame of reference, an accurate kinematic model of the robot is required. If such a model is available, it must be carefully calibrated against the actual hardware, and even then its accuracy may be limited by un-modelled nonlinearities.

The precision of these approaches depends upon an accurate kinematic model of the iCub. A very accurate model, or estimation of the model, is therefore necessary. There exists currently no module estimating the kinematics of the iCub, for other robotic systems this has been done: Gloye et al. used visual feedback to learn the model of a holonomic wheeled robot [5] and Bongard et al. used sensory feedback to learn the model of a legged robot [6], but their method uses no high-dimensional sensory information (such as images).

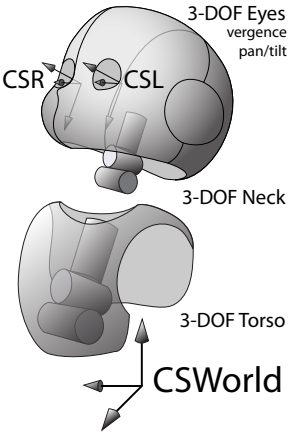

Fig. 1. The coordinate frames relevant for object localisation on the *iCub*. Cameras located at the origin of *CSL*/*CSR* are used to express the position of objects with respect to the *CSWorld*.

Current approaches of localising objects on the iCub are limited to the 'Cartesian controller module', providing a basic 3D position estimation functionality [7]. This module works well on the simulated robot, however its performance on the hardware platform is weak[1], this is because of inaccuracies in the robot model and camera parameters (even with regular camera calibration).

In robot learning, especially imitation learning, various approaches have been investigated to tackle localisation problems. Sauser & Billard have investigated the problem of reference frame transformations from a neuroscience perspective [8], with a special focus on human-robot-interaction less so on accuracy. They were able to imitate gestures from a teacher on a Hoap-2 humanoid robot with external fixed cameras. Though promising their approach has so far not been extended to systems with non-stationary cameras. Machine Learning has earlier been used for spatial perception in simpler scenarios, e.g. Leitner et al. used Artificial Neural Networks (ANNs) to estimate positions of objects to be manipulated by a robot on a 2D plane (a table) [9].

We show that localising objects from vision can be learned on a humanoid robot. The learning does not require explicit knowledge of the camera parameters or the kinematic model.

III. Machine Learning Approach

In this paper we present a biologically inspired machine learning approach by using a feed-forward artificial neural network (ANN). An ANN is an information processing technique inspired by the way the human nervous systems processes information. Inspired by the neurons in human brains, which receive signals — electric impulses transmitted by chemical processes — from other neurons, modify the signals and forward them to their connections. An artificial neuron in comparison is a computational unit with many inputs and one output. The neuron 'fires' based on the input pattern received (Fig. 2). The key difference of artificial neural networks is in the structure of the system. They are a network of interconnected firing neurons, with the important possibility to be trained. Neural networks, especially because of the power to derive meaning from complicated or imprecise data, have been used in a variety of applications.

[1]There errors tend to be in the 3-4 cm range, when localising an object in a typical object manipulation position (in front of the robot, in reachable distance, e.g. on a table).

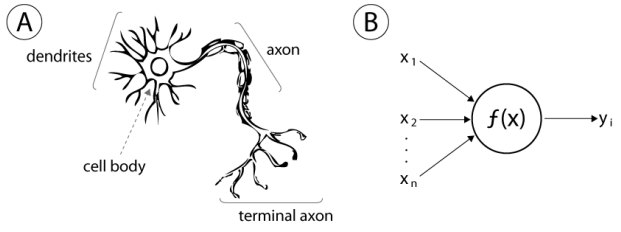

Fig. 2. Model of a human brain cell (neuron, A) and an artificial neuron (B) used in Artificial Neural Networks (ANNs). Image by [10].

TABLE I

A TYPICAL ENTRY FROM THE DATASET AND THE LIMITS USED TO SCALE THE FEATURES AND LOCATIONS FOR THE NEURAL NETWORK.

	ImageL						**ImageR**					
	X	Y	X	Y	width	height	X	Y	X	Y	width	height
Vector	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}
Example	479	411	429	401	99	25	503	437	477	429	80	27
max	640	480	640	480	640	480	640	480	640	480	640	480
min	0	0	0	0	0	0	0	0	0	0	0	0

	Neck			**Eyes**			**Torso**				**Location**		
	0	1	2	3	4	5	0	1	2		X	Y	Z
Vector	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}	v_{17}	v_{18}	v_{19}	v_{20}		p_0	p_1	p_2
Example	-10.0	0.0	0.0	-19.9	-19.9	0.0	-0.1	-9.9	10.1		0.42	0.27	-0.12
max	25	25	10	20	15	5	20	20	50		0.6	0.3	-0.2
min	-25	-25	-10	-20	-15	0	-20	-20	0		0.1	-0.7	-1.2

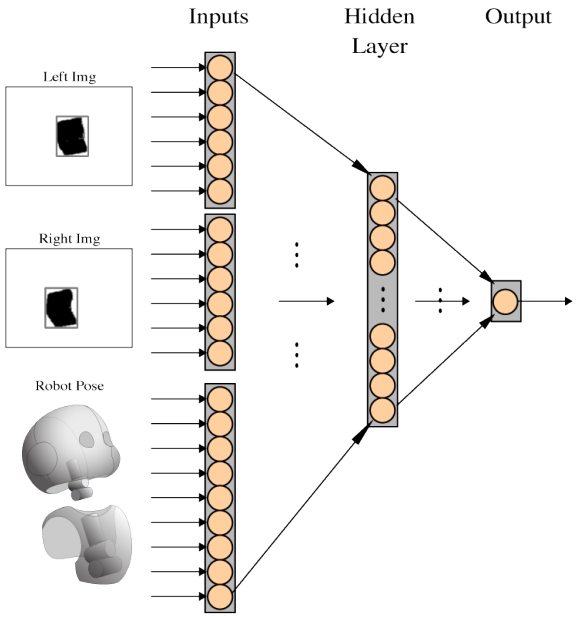

Fig. 3. The model of the feed-forward ANN used for location prediction herein. The inputs are based on computer vision and the state of the robot, depicted on the left. These input neurons are connected to a hidden layer of neurons, which in turn are connected to the single output unit. This unit predicts the position of the object in one axis (of the 3D Cartesian space). (Note: Bias nodes and connections are omitted.)

A. Artificial Neural Network (ANN) Model

We are using a feed-forward ANN, consisting of three layers: one input layer, a hidden layer, and an output layer. This structure of an ANN is sometimes also referred to as a multi-layer perceptron [11]. Each input, in our case image coordinates and robot pose information, arrives at an input node in the input layer of our ANN. As our network is fully connected this input is then provided to every node at the hidden layer. Each connection has a weight attached to it. In addition to all inputs, each hidden node also has a connection to a bias node (i.e. a node with constant value 1). Similarly the output value of every hidden node is then provided to the output node. These connections are again weighted. The output node is as well connected to a bias node. In mathematical terms each node in the hidden and output layer therefore performs the following operation:

$$u = w_0 + \sum_{1}^{n} w_i x_i \tag{1}$$

where u is the output of the node, n is the number of the neurons in the previous layer, w_i the weight related to the i-th connection and x_i the output value of the i-th node in the previous layer. w_0 is the weight related to the bias node.

We herein use neurons in the hidden and output layer containing a sigmoidal activation function of the following form to calculate their respective outputs:

$$\sigma(u) = \frac{1}{1 + e^{-u}} \tag{2}$$

where u is the output calculated to according to Eq. 1. The output value of the whole ANN is the 'calculated' value at the output node.

We chose each ANN's output layer to be of only one single neuron. The output of this node is the estimated position along one Cartesian space axis. Therefore to estimated the full 3D position three separate networks are required.

B. Training

An ANN is usually trained for a specific application through a learning process. To train various methods are available. Herein we use a supervised learning technique. This method requires a collected dataset including both inputs and outputs, i.e. the ground truth. We want the ANN to be trained for prediction of an object's position along one particular axis.

More formally, the task is to estimate the position of an object $p \in \mathbb{R}^3$ in the robot's reference frame (*CSWorld*) given an input, also called feature vector, v. Here we defined $v \in \mathbb{R}^{21}$ containing the state of the robot as described by 9 joint encoder values (ie. the 9 controlled DOF) and the observed position of an object in both camera images. The position is described by the centre of the detected object in the image plane (X and Y coordinate) and additionally by the bounding box of the detected object (upper-left corner, width and height), Fig. 3.

The neural network approach requires a pre-processing step, in which the dataset (input vector) is scaled using the limits given in Table I to get values in the range $[-1, +1]$. The limits are based on to the retrieved image size for the 6 values in each image, and the joint limits (i.e. range of motion) of the robot, for the encoder values.

For training the network the (scaled) dataset was first randomly split into a training (80% of the data) and test set (20%). The test set allows to verify that the results obtained via learning are not over-fitting.

The ANNs were trained using the standard error back-propagation algorithm [12] method on the dataset collected. This method is a generalization of the delta-rule and requires the activation function of the neurons to be differentiable. Back-propagation can be described in two phases: (i) a forward propagation and (ii) a weight update phase. In phase one inputs from the training dataset are propagated through the network using the current weights for each connection. The output is then compared to the ground truth, the known result for the provided inputs. The error between the correct and predicted output is then used to calculate a gradient at each connection, which defines how the weights will be updated.

IV. Collecting the Dataset

To learn about more than one state and hence get the ability to generalise, the dataset needs to contain points, where the robot is in various configurations looking at objects at various locations. A dataset of reference points (RPs) was collected on the real hardware to train the ANNs. Our research platform is the *iCub* humanoid robot [4], an open robotic system, providing a 41 degree-of-freedom (DOF) upper-body, comprising two arms, a head and a torso. Its visual system is a pair of cameras mounted in the head in a human-like fashion (see Fig. 1), providing passive, binocular images.

First we moved the iCub to a randomly selected pose, then the robot's state and the location of the object in the camera frame, provided by an object detection algorithm [13], were registered. The robot would then move about and another data point, with the same RP but from a different point of view, was recorded. In [9] these positions were hand measured and every few movements the object of interest was moved to another (hand-measured) RP.

TABLE II

The collected dataset herein for 3D in comparison to related work in 2D [9].

	2D	**3D**
Data Points	965	1034
Reference Points (RPs)	32	45

To automate and speed up the data collection, a high precision robotic arm, in our case a 5 DOF Katana arm by Neuronics [14], is used to position an object in the shared workspace to provide the humanoid with the information to learn from. Here the ground truth is provided by using the precise inverse kinematics of the industrial robotic arm (mm accuracy) to build the dataset. More than 1000 points were collected in a fraction of the time it took collect a similar dataset by hand (see Table II).

As now two robots are sharing the same workspace, while acting independently, special precautions had to be taken to avoid collisions. We used MoBeE [15], [16], an open-source software system, which provides real-time collision avoidance on the iCub based on geometric calculations.

V. Experiments and Results

A. Size of Hidden Layer

To find the appropriate number of neurons for the hidden layer experiments with various hidden neurons were performed. The number of hidden neurons achieving the lowest estimation errors was then selected. Fig. 4 shows that there is an minimal predictions errors when using around 10 neurons in the hidden layer. For this we used only a reduced dataset with one axis. The networks were trained on the dataset and the errors reported are on the 20% validation set. The errors were averaged over 10 runs, i.e. with 10 trained networks all having the same number of hidden neurons. For prediction along one axis the lowest average error was found with 10 hidden neurons and is about *0.8* cm.

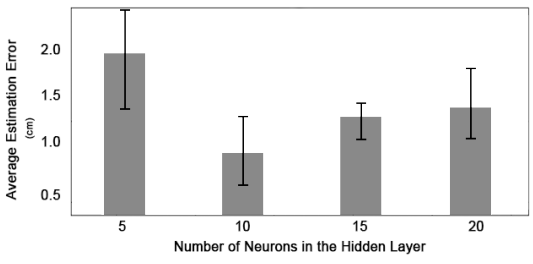

Fig. 4. The estimation errors for one specific axis using ANNs with varying number of hidden neurons. The error bars show min and max errors in 10 trials.

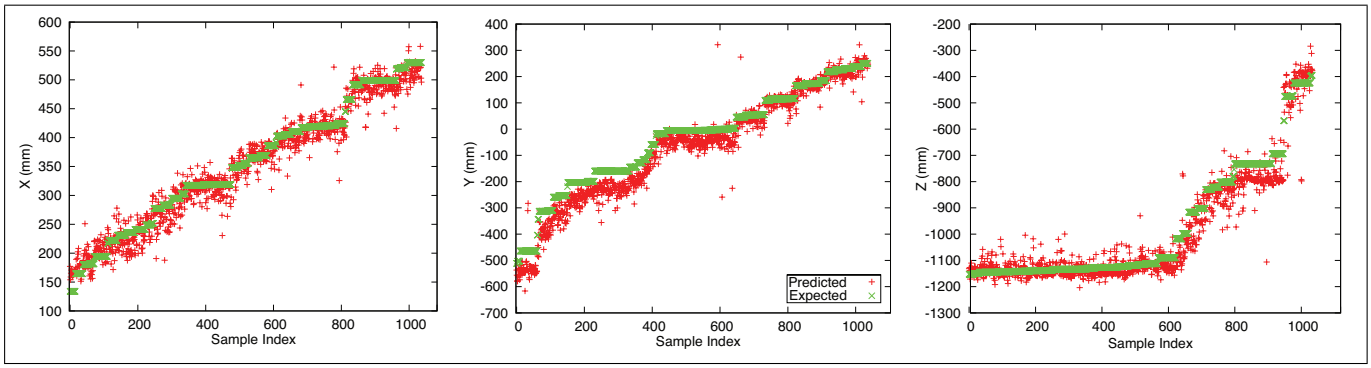

Fig. 5. The prediction errors by three separate ANNs. One each for the X, Y, and Z axis.

TABLE III
COMPARING THE ERRORS MEASURED WITH THE ANN MACHINE LEARNING METHOD HEREIN TO THE PREVIOUS 2D CASE (FROM [9]).

	ANN-2D	**ANN-3D**
Average Error X (mm)	5.40	15.9
Average Error Y (mm)	5.43	43.1
Average Error Z (mm)	-	37.3

B. Learning to Estimate Object Positions

Three separate ANNs were trained, each containing 10 neurons in the hidden layer, to predict the position along one axis each. This independent training allows for parallel evaluation and reduces the complexity of each ANN. On average 1800 epochs were needed for the prediction error of the ANNs to converge. The back-propagation technique was using a learning rate of 0.35 and a momentum of 0.1.

Fig. 5 shows three plots, one per axis, visualising the location and prediction error per sample in the dataset. The dataset was sorted by location in this figure, to highlight that multiple samples per location were collected. Overall the ANNs seem to quite nicely predict the position of the object. Table III shows the average errors per axis and compares this with previous work, which predicted positions in the simple

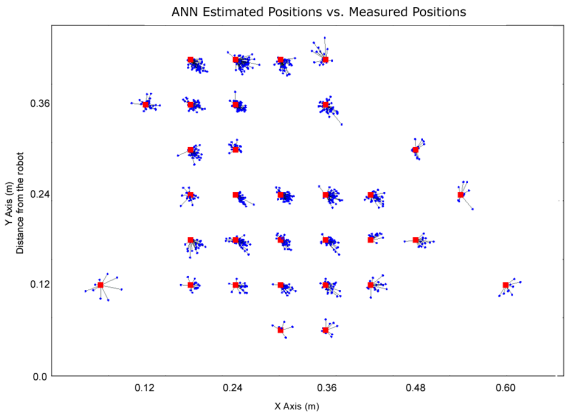

Fig. 6. The estimated object position (blue dots) vs. the measured object position (red blocks) for the machine learning approach. This 2D result are from [9].

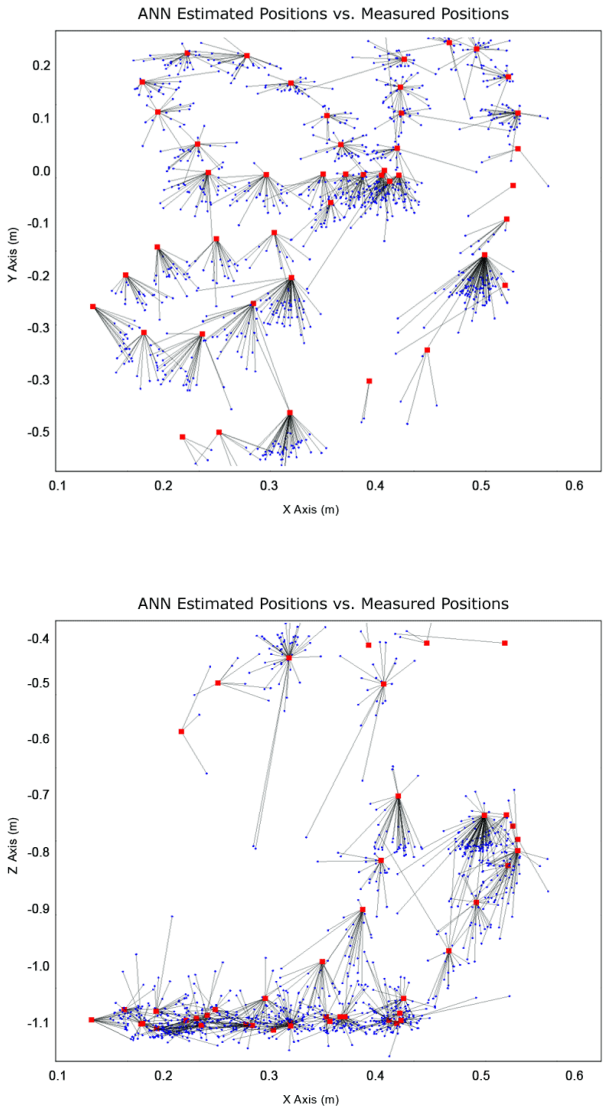

Fig. 7. The estimated object position (blue dots) vs. the measured object position (red blocks) for the 3D machine learning approach. *Note: The values here are in the reference frame of the Katana robot.*

2D case. The trained neural networks allow to estimate the position of the object in 3D space, with a high enough accuracy to allow for grasping experiments. The average error on the dataset is for the X-axis 15.9 mm, for the Y-axis 43.1 mm and for the Z-axis 37.3 mm. This is also in the error range of current localisation methods available for the *iCub*. The errors reported are the average of 10 runs.

A few outliers, with higher prediction errors can also be seen in Fig. 5. These might be results from errors when data was collected while a collision was avoided, e.g. when a data point was collected while the robots were reset to a safe pose and the synchronisation for the dataset between the two robots and the vision was lost.

To better visualise where the prediction errors are largest the difference between actual and predicted locations are plotted. Fig. 6 shows the difference for each test case in the 2D plane from previous work [9]. To compare we plotted the results from the 3D prediction in Fig. 7. The two plots show the errors for the XY and XZ plane respectively. It can be seen that the dataset is not evenly distribute over the 3D space. This is due to the independent and random movements of the Katana and the iCub robot. A dataset containing more points and distributed better throughout the workspace, we expect, will allow for improved prediction.

They were then implemented on the *iCub* to perform real time distance estimation of objects and to allow for further verification. The object position in the images (provided by an object detection filter from a separate iCub vision system [13]) and joint encoder values were provided to the trained ANNs in real-time. The output of the ANNs was then used to steer the robot's left arm, with the provided operational space controller, to the position estimated. Fig. 8 shows a still-picture of the iCub reaching for the red-block at the end of the Katana robot.

Although the system was trained with only one specific object, arbitrary objects can be localised by the deployed system. Tests performed with a moving object, but the error is harder to measure when both the object and the robot are moving, yet no large errors were found by visual inspection.[2]

[2]A video showing localisation while the iCub and the object is moving can be found at `http://Juxi.net/projects/icVision/`.

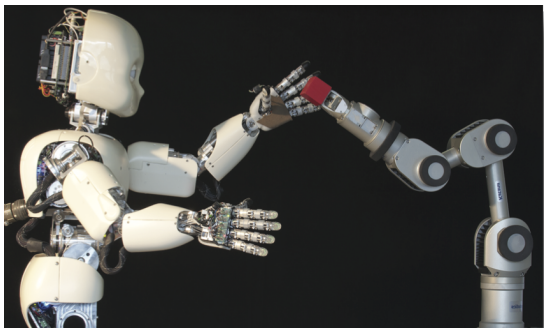

Fig. 8. Testing the localisation in combination with the operational space controller available for the iCub humanoid. The red cube was located from vision and the left arm is then reaching for the estimated position.

C. Learning Spatial Perception

Assuming that the maximum reachable space is limited by the robot's arm, we can try to train an ANN to predict whether an object in the scene is reachable given the current robot pose. To do so we modify the training set to contain, instead of the 3D coordinates in p, only a binary output, whether the point is reachable or not. For simplicity we defined every point that is farther than $60cm$ away unreachable and every point closer to be reachable. In this simplified problem we do not predict if there is a feasible joint configuration for this point. The ANN for this task, which has the same model as each of the ANNs before, can be trained using the same back-propagation method.

VI. Conclusions

In this paper we investigate artificial neural networks (ANN) and their applicability to the spatial perception problem in robots. We show that by training ANNs the skill of estimating object locations based on visual perception can be learned. A series of experiments were performed on the iCub humanoid robot.

Although our approach does not need explicit knowledge of a precise kinematic model or camera calibration, it learned to localise objects with comparable accuracy to currently available systems. Furthermore our approach when deployed is computationally less expensive and less dependent on other concurrently running modules and techniques. As the learnt models are 'light weight' they could easily be incorporated into embedded systems and other robotic platforms. We also presented a method to generate a model representing whether an object is reachable or not can based on the same method.

The results of locating objects are easily usable with current operational space control on the iCub and provide sufficient accuracy for real world reaching scenarios. The results on the first 3D dataset show that the method can in fact perform full 3D position estimation. The results show that our approach can learn simpler ways to perceive the location of objects than the human engineered methods. More thorough experimental testing on the iCub will need to be conducted especially to on how to improve our system over time.

VII. Future Work

In the future we are planning to investigate methods of learning spatial perception that will not require the use of a second robot to generate the ground truth data. A proprioceptive model together with some haptic feedback could provide the necessary inputs. We would also like to overcome the necessity of using 3D Cartesian space and directly learn about joint space or joint space regions.

Another interesting question we aim to investigate is the importance of the chosen reference frame, e.g. it could be better to use a frame representing the robot's ego-sphere [17] or even find a direct vision to joint-space mapping, as proposed in [18]. Both of these would probably allow for a better sensorimotor integration.

REFERENCES

[1] J. Plumert and J. Spencer, *The emerging spatial mind*. Oxford University Press, USA, 2007.

[2] A. Schutte, J. Spencer, and G. Schöner, "Testing the dynamic field theory: Working memory for locations becomes more spatially precise over development," *Child Development*, vol. 74, no. 5, pp. 1393–1417, 2003.

[3] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, 2nd ed. Cambridge University Press, 2000.

[4] N. G. Tsagarakis *et al.*, "iCub: the design and realization of an open humanoid platform for cognitive and neuroscience research," *Advanced Robotics*, vol. 21, pp. 1151–1175, 2007.

[5] A. Gloye, F. Wiesel, O. Tenchio, and M. Simon, "Reinforcing the Driving Quality of Soccer Playing Robots by Anticipation," *IT - Information Technology*, vol. 47, no. 5, 2005.

[6] J. Bongard and V. Zykov, "Resilient machines through continuous self-modeling," *Science*, vol. 314, no. 5802, pp. 1118–1121, 2006.

[7] U. Pattacini, "Modular Cartesian Controllers for Humanoid Robots: Design and Implementation on the iCub," Ph.D. dissertation, RBCS, Italian Institute of Technology, Genova, 2011.

[8] E. Sauser and A. Billard, "View sensitive cells as a neural basis for the representation of others in a self-centered frame of reference," in *Int'l. Symposium on Imitation in Animals and Artifacts*, 2005.

[9] J. Leitner, S. Harding, M. Frank, A. Förster, and J. Schmidhuber, "Towards spatial perception: Learning to locate objects from vision," in *Postgraduate Conference on Robotics and Development of Cognition (RobotDoC)*, 2012.

[10] V. G. Maltarollo, K. M. Honório, and A. B. F. da Silva, "Applications of artificial neural networks in chemical problems," in *Artificial Neural Networks - Architectures and Applications*, K. Suzuki, Ed., 2013.

[11] L. V. Fausett, *Fundamentals of neural networks: architectures, algorithms, and applications*. Prentice-Hall Englewood Cliffs, NJ, 1994.

[12] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2010.

[13] J. Leitner, S. Harding, M. Frank, A. Förster, and J. Schmidhuber, "icVision: A Modular Vision System for Cognitive Robotics Research," in *Proc. of the Int'l Conference on Cognitive Systems (CogSys)*, 2012.

[14] Neuronics AG, "Katana user manual and technical description."

[15] M. Frank, J. Leitner, M. Stollenga, S. Harding, A. Förster, and J. Schmidhuber, "The Modular Behavioral Environment for Humanoids and Other Robots (MoBeE)," in *Proc. of the International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2012.

[16] J. Leitner, S. Harding, M. Frank, A. Förster, and J. Schmidhuber, "Transferring spatial perception between robots operating in a shared workspace," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[17] J. Ruesch, M. Lopes, A. Bernardino, J. Hornstein, J. Santos-Victor, and R. Pfeifer, "Multimodal saliency-based bottom-up attention a framework for the humanoid robot iCub," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 962–967.

[18] J. Law, P. Shaw, and M. Lee, "Development of eye-head gaze control on the icub robot," in *Proc. of the Int'l Conference on Cognitive Systems (CogSys)*, 2012.