Visually Guided Whole Body Interaction

Bram Bolder, Mark Dunn, Michael Gienger, Herbert Janßen, Hisashi Sugiura, Christian Goerick

2007

Preprint:

This is an accepted article published in IEEE International Conference on Robotics and Automation (ICRA 2007). The final authenticated version is available online at: https://doi.org/[DOI not available]

Visually Guided Whole Body Interaction

Bram Bolder, Mark Dunn, Michael Gienger, Herbert Janssen, Hisashi Sugiura, and Christian Goerick
Honda Research Institute Europe
Carl-Legien-Straße 30
D-63073 Offenbach am Main, Germany
{bram.bolder, mark.dunn, michael.gienger, herbert.janssen, hisashi.sugiura, christian.goerick}@honda-ri.de

***Abstract*— We describe a system for visual interaction developed for humanoid robots. It enables the robot to interact with its environment using a smooth whole body motion control driven by stabilized visual targets. Targets are defined as visually extracted "proto-objects" and behavior-relevant object hypotheses and are stabilized by means of a short-term sensory memory. Selection mechanisms are used to switch between behavior alternatives for searching or tracking objects as well as different whole body motion strategies for reaching. The decision between different motion strategies like reaching with right or left hand or with and without walking is made based on internal predictions that use copies of the whole-body control algorithm. The results show robust object tracking and a smooth interaction behavior that includes a large variety of whole-body postures.**

I. INTRODUCTION

Research on humanoid robots is increasingly focusing on interaction in complex environments, including autonomous decision making and complex coordinated behavior. Several interactive robot systems were already introduced. A complete architecture for a small humanoid (Sony QRIO) that uses a central action selection driven by so called *behavior values* provided by the individual behaviors is described in [1], [2]. This robot is equipped with some perceptual abilities and realizes impressive abilities including emotional/motivational control and learning as well as humanoid multi-degree of freedom control.

Kismet [3] also realizes a variety of interaction abilities and contains both a powerful vision and attention system and behavior selection. It also integrates low level vision representations as feature maps and higher level representations for faces. The main focus of this system is child-like interaction and developmental learning.

Proto-objects are a concept originating from psychophysical modeling [4], [5], [6]. They can be thought of as coherent regions or groups of features in the field of view that are trackable and can be pointed or referred to without identification. The term proto-objects is however used in many different ways. They are used to generate saccades, track multiple visual stimuli simultaneously, or to model attention, change blindness, or visual scene representation.

Orabona et al. [7], [8] developed a system that uses proto-objects — in their case colored blobs — to let a robot learn the notion of an object consisting of possibly multiple proto-objects using statistical means.

Here, the term *proto-objects* will be used in a manner similar to the psychophysical approaches. They refer to multiple elements in the visual scene that can be tracked simultaneously independent of belonging to an existing physical object or not. The representation is however extended to three dimensions to cope for ego-motion and for large changes in size due to depth changes. The proto-objects consist of a time series of sensory measurements (here 3d blobs) and a method to predict a future sensory measurement.

There exist many systems of tracking visually salient points or regions on humanoid robots [9], [10]. Here the stress is not to implement yet another method but to lay the foundation for a new concept powerful enough to be a flexible interface to behaviors.

The need for non-monolithic internal representation of the environment as well as non-monolithic control [11] inspired the distinction of proto-objects and object hypotheses. Object hypotheses interpret proto-objects as originating from physical objects according to some more or less specific models to allow different behaviors to interact. Some behaviors do not even need object hypotheses, they can operate only using proto-objects.

In the field of strategy selection, Wolpert et al. [12] propose a learning architecture that consists of several inverse and forward models. According to the prediction error, the most adequate strategy is selected.

Ude et al. [13] present a system that uses blobs for a visual interaction with a humanoid robot. Their system however does not use full 3d information for tracking, has restrictions on the ellipse radii and is not able to walk.

Regarding the challenge of realizing a fully autonomous interactive humanoid robot, an intermediate step is presented in this work: ASIMO is able to interact with the environment driven by visual perception using simple decision making and coordinated whole body motion.

Our approach is to build a system that — for now — uses a relatively simple definition of visual target objects, namely any elongated colored object. This system implements the fundamental elements of an architecture that is easily extendible to cope with more long term targets. Integrating a more elaborate computer vision system, for instance using object recognition [14], would be the next step, but in this paper the focus lies on closing the interaction loop.

Novel in the context of humanoid robots are the following key points:

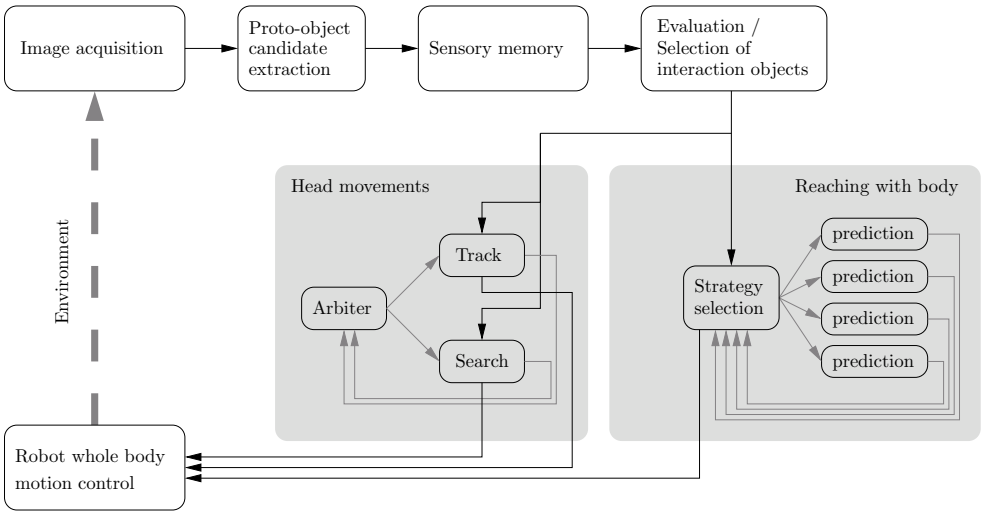

Fig. 1. Overview of the system design.

- The use of proto-objects that can be used both in raw form for e.g. visual tracking in 3d, or to form stable object hypotheses as they are needed for reaching and grasping. These proto-objects consist of relatively low level perceptual information (here 3d blobs) and are dynamically stored in a short-term sensory memory.
- Decision mechanisms that evaluate behavioral alternatives based on sensory information and internal prediction.
- A motion control system that is able to be driven by a wide range of possible target descriptions and that ensures smooth well coordinated whole body movements (using a set of cost functions as null space criteria).

The general design of the system is depicted in Fig. 1. The perception system uses visual features and stereo based 3d information to detect relevant visual stimuli. It keeps this information as proto-objects in a short-term sensory memory. This sensory memory is then used to derive targets for visual tracking and to form stable object hypotheses from which movement targets for reaching movements can be derived. A prediction based decision system selects the best movement strategy and executes it in real time. The internal prediction as well as the executed movements use an integrated control system that uses a flexible target description in task space in addition to cost-functions in null space to achieve well coordinated and smooth whole body movements. The system is implemented using the real time environment RTBOS [15].

II. Design of the System

In the following, several components of the system are discussed in detail.

A. Proto-object Candidate Extraction

To be able to generate proto-objects the image processing has to be able to find entities in the environment that are dynamically stable in position and extent. Thus the best candidates are segmentation algorithms such as color segmentation, texture segmentation, or feature extractors that find unique salient points. To obtain 3d information stereo disparity calculations or other stereo algorithms can be used.

The general idea is to extract 3d ellipsoids from the visual input — here referred to as blobs — that encode the position, size, and orientation of significant visual stimuli. As a proof of concept and since the visual preprocessing is not of significance, the only feature used here is the color similarity to a given reference color.

Pairs of color images are used, labeled with the time of acquisition. These images are processed in two parallel paths. One of them computes the stereo disparities for all pixels with sufficient texture information. In the other, pixels are evaluated as to whether they lie in a certain volume in HLS space. The result is subjected to morphological operations that eliminate small regions of one class of pixel. The resulting pixels that lie in the HLS volume are grouped into regions that are contiguous in the image plane. The largest resulting groups that exceed a minimum size are selected for further processing.

For each of these groups, the center of area in the image plane x_p, y_p and the median of the stereo disparities d of all its pixels are computed. Further it is detected whether the group region touches the image boundaries; in this case the data is labeled as inaccurate since parts of the real world object corresponding to the region are probably outside the field of view. The orientation of the principal axis ω_p and the standard deviations (σ_{p1}, σ_{p2}) of the pixels in the image plane are computed for each group using a principal component analysis of the correlation matrix of the pixel positions. The extracted ellipse and its axes of a region of specified color can be seen in Fig. 2. All images and postures are consistently time labeled at the time of their acquisition. Care is taken to synchronize these labels to allow the correct mapping of images to their respective kinematics in order to cope with ego-motion. Using the camera system geometry and the robot kinematics, the position (x_p, y_p, d), the sizes (σ_{p1} ,σ_{p2}) and the orientation ω_p are transformed to their

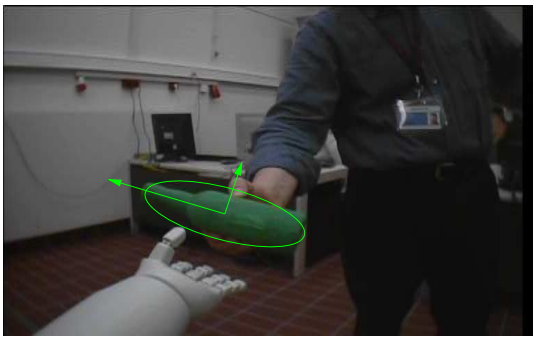

Fig. 2. Extracting a blob in form of an oriented ellipse.

respective metric world coordinates $\vec{r}_w$, (σ_{w1}, σ_{w2}) and $\vec{\omega}_w$. We define a blob as a set of data consisting of the time label, the position $\vec{r}_w$, the orientation $\vec{\omega}_w$, the standard deviations σ_{w1} and σ_{w2}, and the label whether the data is accurate or inaccurate as described above.

B. Proto-Objects in Sensory Memory

To be able to form stable object hypotheses, the sensory information needs to be buffered and organized consistently. This is done in form of proto-objects in sensory memory. Here the incoming blobs are mapped one-to-one to proto-objects in sensory memory which themselves contain the most recent blob data.

If the memory is empty, a proto-object is generated from blob data by simply assigning a unique identifier to the new proto-object and inserting the incoming blob data into it.

If the sensory memory already contains one or more proto-objects, a prediction for each proto-object is generated as a blob data. This predicted blob data is based on all blob data that is contained in the proto-object and is generated for the current time. The current implementation assumes noisy input and slowly moving objects. Therefore the prediction uses a weighted low-pass filter for the positions, orientations, and standard deviations (σ_{w1}, σ_{w2}).

Each incoming blob is either inserted into an existing or newly generated proto-object. This is based on a minimum distance between incoming and predicted blob so that unique identifiers are assigned to all incoming blobs. The metric for the distance computation is based on both euclidean distance and rotation angle. The blob orientation description is ambiguous with respect to 180 degree flips since it is derived from the principal axis of a 2d distribution. To be able to track the blob orientation, the orientation of newly inserted blob data is modified so that the orientation distance of the new blob is always less than or equal to 90 degrees.

Every time new incoming blob data is generated by the processing, its time label is compared to the time labels of the blob data inside all proto-objects and all blob data that are older than a certain threshold are deleted. In order to keep the system in a consistent state, this deletion is done even if the image processing does not find any blobs in the image pairs. If a proto-object does not contain any blob data it is also deleted from sensory memory.

C. Evaluation / Selection of Interaction Objects

The interaction system needs to evaluate the proto-objects in sensory memory using different criteria. These evaluations are also based on the blob data predictions of all proto-objects. The label of this prediction is set to "memorized" if the latest blob data in the proto-object is older than the prediction time. Otherwise, it is set to the label of the latest blob data in the proto-object.

A minimum criterion that is already sufficient for the behavior of fixation and tracking is a blob labeled as inaccurate. If more severe criteria such as stable values σ_{w1} and σ_{w2} and a maximum distance to avoid relying on insufficient vision data are considered, stable object hypotheses can be extracted. To implement manipulation behaviors like "poke balloon", additional constraints can be put on the stable object hypotheses. These could be roughly spherical shape $((\sigma_{w1}-\sigma_{w2})/\sigma_{w1} < \text{threshold})$ and easiest execution of the behavior (minimum distance to a behavior specific reference point, e.g. for poking in front of the body). A behavior like "power grasp object" could require a minimum elongation for grasp stability $((\sigma_{w1} - \sigma_{w2})/\sigma_{w1} > \text{threshold})$ and a suitable diameter $(\text{threshold} < \sigma_{w2} < \text{threshold})$.

The output from the sensory memory is therefore evaluated with respect to the object criteria (in this case distance, size, and minimum elongation). The raw output together with the evaluation is sent to the 3 behaviors (search, track, reach) in a compact form. Each behavior can then easily extract the relevant information, since e.g. inaccurate blobs that do not match the object criteria can be tracked, but only stable elongated objects will be reached for.

D. Behavior Selection: Tracking and Searching

The output of the sensory memory is used to drive two different head behaviors: 1) searching for objects and 2) gazing at or tracking objects or blobs. Separate from these behaviors is a decision instance or *arbiter* that decides which behavior should be active at any time. The decision of the arbiter is solely based on a scalar value that the behaviors provide, which we call a *fitness value*, but which other authors [1], [2] refer to as *behavior value*. This fitness value describes how well a behavior can be executed at any time. In this concrete case tracking needs at least an inaccurate blob position to point the gaze direction at, but of course can also use a full object hypothesis. Thus the tracking behavior will output a fitness of 1 if any blob or object is present and a 0 otherwise. The search behavior has no prerequisites at all and thus its fitness is fixed to 1. For the case of this very simple behavior setup, the arbitration is of course trivial. However we implemented it as a competitive dynamical system similar to the one described in [16] for extensibility. Thus the arbiter uses the vector of fitness values from all behaviors, as an input to a competition dynamics that calculates an activation value for each behavior. The competition dynamics uses a pre-specified inhibition matrix that can be used to encode directed inhibition — behavior A inhibits behavior B but not vice versa — to specify behavior

prioritization and even behavior cycles. In this case tracking is prioritized to searching by such a directed inhibition.

Our method is comparable to e.g. Nicolescu and Mataric [17] in that a non-distributed selection mechanism is employed which avoids limited scalability as e.g. Tani and Nolfi [18] experienced. We are currently extending the system to about a dozen different behaviors.

The search behavior is realized by means of a very low resolution (5 by 7) inhibition of return map with a simple relaxation dynamics. If the search behavior is active and new vision data is available it will increase the value of the current gaze direction in the map and select the lowest value in the map as the new gaze target. Additionally the whole map is subject to a relaxation to 0 and a small additive noise. This generates a visual search pattern with a random sequence of fixations that takes into account all visual information immediately and results in an efficient and fast finding of relevant objects. The size of the inhibition of return map is derived from the field of view of the cameras relative to the pan/tilt movement range. Higher resolutions will not change the searching significantly. The relaxation time constant is set in the second range so that motion of the robot, which will effectively invalidate the inhibition map, is not a problem.

The tracking behavior is realized as a multi-tracking of 3-dimensional points. The behavior takes all relevant proto-objects and object hypotheses into account and calculates the pan/tilt angles for centering them in the field of view. Then a cost function with a trapezoidal shape in pan/tilt coordinates is used to find the pan/tilt angle that will keep the maximum number of objects in the effective field of view of the cameras and this is sent as the pan/tilt command. Since the tracking behavior always uses the stabilized output of the sensory memory the robot will still look at a certain position even if a blob disappears for a short time. This significantly improves the performance of the overall system.

The two visual interaction behaviors together with the arbiter switching mechanism show very short reaction times and have proven efficient to quickly find and track objects.

E. Strategy selection: Reaching

Similarly to the search and track behaviors, the reaching behavior is driven by the sensory memory. As shown in Fig. 1, the proto-object information is evaluated, and the position and orientation of the target points is sent to the reaching behavior. This behavior is composed of a set of internal predictors and a strategy selection instance. Each predictor includes a whole body motion controller and a cost function evaluation.

The underlying whole body control model is depicted in Fig. 3. The geometry and kinematic topology matches the humanoid robot ASIMO [19]. The first link corresponds to the heel coordinate system comprising three degrees of freedom. Its degrees of freedom are translations in forward and lateral direction as well as a rotation about the vertical axis. The consecutive links correspond to the body segments of the robot. The pelvis is undergoing three translations and rotations with respect to the heel frame. The head is

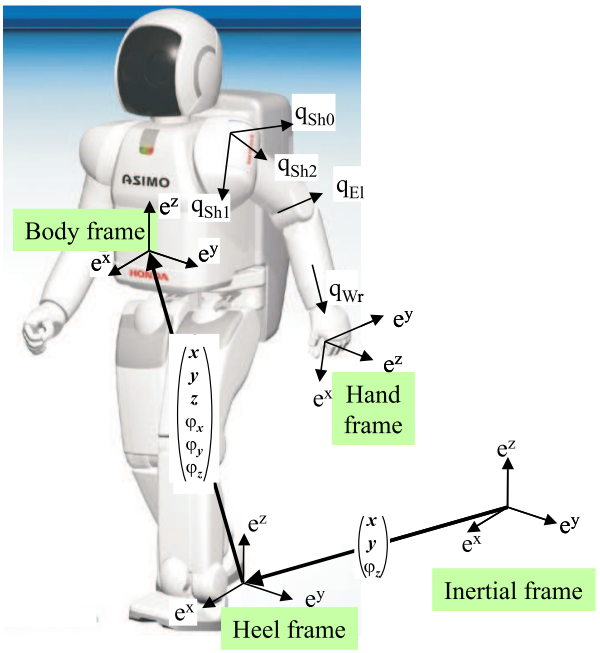

Fig. 3. Kinematic model for whole body motion.

connected to the upper body with pan and tilt joints. Further, the two arms comprise 5 dof each. An additional coordinate system with some offset to the hand origin defines a hand reference point. All together, the model comprises 21 dof.

The robot motion is generated with the "redundancy resolution" framework by Liégeois [20], [21], [22] for redundant systems. To map the task space trajectories into joint space, a resolved motion rate control algorithm is employed. It computes joint speeds by using a weighted generalized pseudo-inverse of the task Jacobian. Redundancies are resolved by mapping the gradient of an optimization criterion into the null space of the motion. In this work a joint limit avoidance criterion is used. Details on the whole body control algorithm are given in [23], [24].

The control system allows to give commands in a highly flexible way. For this, a vector $\boldsymbol{l}$ with the number of task variables is defined. Its elements define if a respective task variable is active ("1") or inactive ("0"). In the current implementation, the task vector is

$$\boldsymbol{x}_{task} = \left(\boldsymbol{x}_{ht,l}\ \boldsymbol{\varphi}_{ht,l}\ \boldsymbol{x}_{ht,r}\ \boldsymbol{\varphi}_{ht,r}\ \boldsymbol{\varphi}_{head}\right)^T \qquad (1)$$

where $\boldsymbol{x}_{ht}$ denotes the hand tip reference position of the respective hand and $\boldsymbol{\varphi}$ the orientation of the hands and the head. Based on vector $\boldsymbol{l}$, the equation system is set up according to the following procedure: Starting with the task element 0, the respective task Jacobian is added to the overall task Jacobian if the corresponding element of $\boldsymbol{l}$ equals "1", otherwise it is skipped. Vector $\boldsymbol{l}$ is updated every sampling interval through a command interface, so that the task vector can be changed dynamically at run-time.

The whole body controller is coupled with a walking

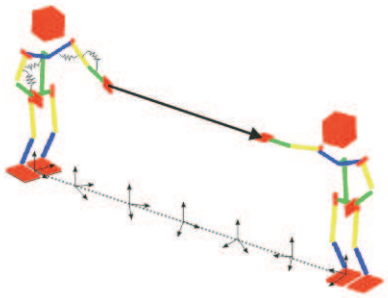

Fig. 4. "Floating" heel frame: The stance position of the feet is the result of the null space motion. It is a local optimum regarding the given task.

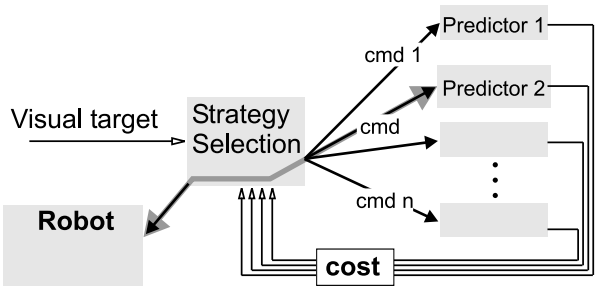

Fig. 5. Parallel simulation architecture.

and balancing controller, which stabilizes the motion. This scheme allows to perform even fast dynamic whole body motions in a stable way.

Internal prediction architecture

In the following, a step from a single whole body controller towards a parallel simulation architecture that consists of several controllers will be made. The idea is to evaluate many strategies that solve the task in different ways. In the remainder, the task of reaching towards an object and aligning the robot's palm with the objects longitudinal axis will be regarded.

In a first step, the visual target is split up into different motion commands, with which the task can be achieved. Four commands are chosen: Reaching towards the target with the left and right hand, both while standing and walking. In the following, a motion control simulation relating to one of those commands will be called a "strategy". Other interesting sets of commands such as kneeling down or crawling (see e. g. [25] are not possible due to the robot's physical capabilities.

In the strategies that reach from a fixed stance, the degrees of freedom describing the heel coordinate system are constrained. For the strategies that involve walking, the kinematic constraints on the heel degrees of freedom are released. This leads to a "floating" heel frame that will converge to a position and orientation that is a local optimum with respect to the null space criterion (see Fig. 4). This leads to a very interesting property of the control scheme: the control algorithm will automatically compute the optimal stance position and orientation with respect to a given target. On the robot, the floating frame is set as the target for a step pattern generator, which generates appropriate steps to reach the computed heel position and orientation.

Now each strategy computes the motion and an associated cost according to its specific command. The cost describes the suitability of the strategy in the current context. It is composed of a set of penalties that will be described later. The costs are evaluated by the strategy selection process, and the strategy with the lowest cost is identified. The corresponding command is redirected to the physical robot (See Fig. 5). The robot is controlled with the identical whole body motion controller that is employed for the internal simulations.

An interesting characteristic of the system is the temporal decoupling of real robot control and simulation. The strategies are sped up by a factor of 10 with respect to the real-time control, so that each strategy has converged to the target while the physical robot still moves. Therefore, the strategies can be regarded as prediction instances, since they look some time ahead of the real robot. Nevertheless, the control algorithms running within the strategies and on the robot are identical.

From a classical point of view, the predictions could be seen as alternative results of a planning algorithm. A major difference is their incremental character. We use a set of predictors as continuously acting robots that each execute the task in a different way. The most appropriately acting virtual robot is mapped to the physical instance.

Selection mechanism

The selection of the most appropriate strategy is based on the evaluation of a multi-criteria cost function. The cost function encodes the following heuristics: Standing will be preferred over walking, walking over doing nothing. For this, the following criteria are incorporated in the cost function:

Reachability of the target: In the control algorithm, two state vectors are computed. One regards the system as "ideal", ignoring joint limits. The state vector that is commanded to the physical robot is clipped so that the joint limits are not violated. In normal operation, both vectors are identical. If however, one ore more joint limits are violated, a more or less large error between commanded and actual task will emerge. This is illustrated in Fig. 6. This error is used as a measure of reachability.

To prevent the strategy selection mechanism to oscillate between two adjacent solutions, the target is surrounded by two interval regions ξ_1 and ξ_2, where $\xi_1 < \xi_2$. Now, the two following cases are regarded:

1) The hand has not reached the target: The target is considered as reached, if the effector gets inside ξ_1
2) The hand has reached the target: The target is considered as not reached, if the effector gets outside ξ_2

Postural discomfort: The weighted least squares distance of the joints to their center positions defines a "discomfort" penalty. Whenever the target is moved from the one side to the other, this penalty will make the system switch the reaching arm. The hysteresis ensures that there is no

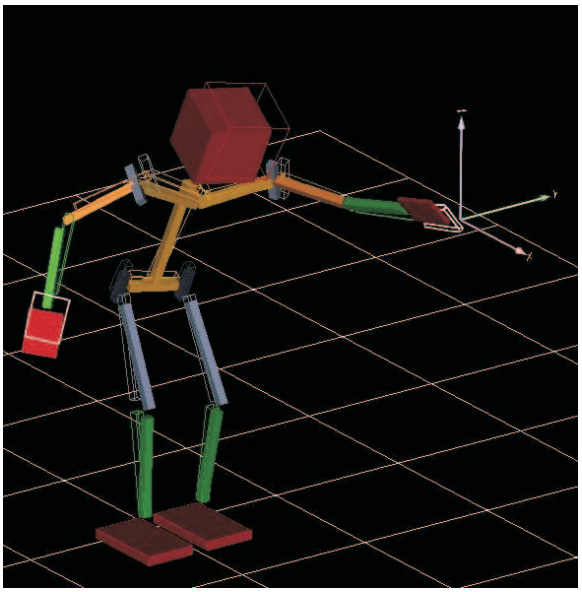

Fig. 6. "Ideal" (wireframe) and "clipped" (solid) state vectors lead to a task error that is used as a measure of reachability.

oscillation between left and right arm when both penalties have the same value.

"Laziness": Both walking strategies receive a constant penalty, so that standing will be preferred over walking.

Time to target: This penalty is added to the walking strategies. It is a measure for the estimated "steps to target". This penalty makes the robot select the strategy that brings it towards stance with the minimum number of steps.

If a walking strategy has been selected, the real robot will start making steps toward the computed heel coordinate system. In this case, the standing strategies will not be considered, until the step pattern generator of the real robot has reached the converged target heel position.

F. Collision Detection

In order to ensure safety of the robot during operation, a real time collision detection algorithm is used. The collision detection uses an internal hierarchical description of ASIMO's body in terms of spheres and sphere-swept lines that is used together with the kinematic information to calculate the distances between the segments (limbs and body parts) of the robot. If any of these distances falls below a threshold the high-level motion control will be disabled, so that only the dynamic stabilization of the bipedal walking is active. The collision detection acts as a last safety measure and is not triggered during normal operation of the robot.

Additionally a simple collision avoidance limits the position of all movement targets so that e.g. wrist target positions inside or very close to the body are never generated.

III. Implementation

The vision and control processing is divided up into several smaller modules that interact in a data driven way in the real time environment RTBOS [15]. This component based subdivision of processing is similar to that of other RTBOS based systems, for instance the Brain-like Active Sensing System BASS described in [14], [26] that combines active vision with object recognition. Since several components are shared between both systems, an additional object recognition can be easily added to the system described here.

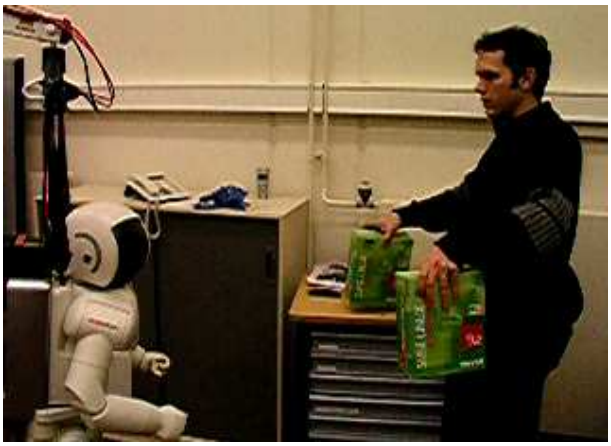

Fig. 7. Snapshot while visually tracking multiple non-ellipsoid objects.

For the implementation the system and work load was distributed over four different computers interconnected using fast ethernet connections. Onboard of the robot, the vision host is used to acquire images (half-frame NTSC) and send them to the offboard vision PC. The control host receives whole body motion targets, generates postures taking the collision detection into account, and sends the current posture every 5 ms to the offboard control PC.

All vision processing was done on images of 384 by 240 pixels, i.e. a horizontally scaled version of the NTSC images. This allowed faster stereo calculations for which the Small Vision System SVS software of SRI was used.

Two offboard PCs were used to split the computationally expensive vision calculations (from the raw images to the proto-objects in sensory memory) from the control part running many parallel components with small cycle times (processing from the proto-object evaluation to the motion targets).

IV. Results

The system as described above was tested many times with different people interacting with ASIMO with a variety of target objects. The scenario was always to have a human interaction partner who had an elongated object that was shown or hidden in various ways to ASIMO.

The system is not restricted to only one object, as can be seen in Fig. 7. If a number of objects are close to each other, the system will try to keep all objects in the field of view. If they are further apart, the objects leaving the field of view will be neglected after a short while and the system will track the remaining object(s).

The system shows a very good interaction performance. Objects are quickly found and reliably tracked even when moved quickly. The reaching behavior will reach for any elongated object of appropriate size that is presented within

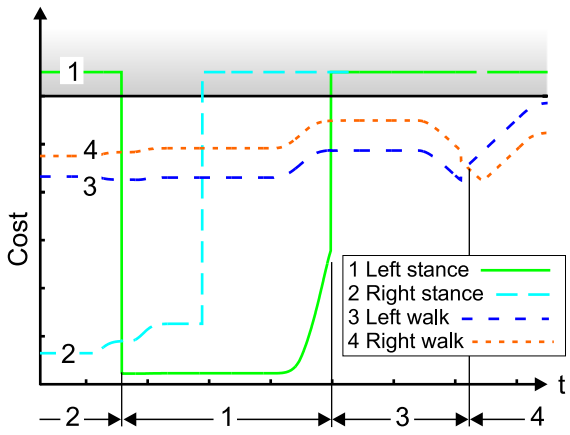

Fig. 8. Progression of fitness values over time.

a certain distance — from about 20cm to about 3m. ASIMO will also track the object orientation correctly, e.g. if a bottle is presented first with the bottleneck up, the robot will reach with the proper hand orientation following the object orientation in real time even if the bottle is rotated to a horizontal or inverted position.

Fig. 8 shows the cost functions of the four moving strategies over time. The cost function with the lowest value is considered the best. The periods within which the respective strategy is active, is shown under the time axis. When moving the object from the left to the right, the robot first (period 1) tries to reach it with the right hand and then dynamically switches to the left hand. When the object gets out of reach, it starts walking and follows the object (period 3). In period 4, the robot walks and tries, at the same time, to reach the object with the right hand. If all costs are above the gray region, no command is given to the robot.

ASIMO switches between reaching with the right and left hand according to the relative object position with some hysteresis. Also walking is used only when necessary and the robot can be driven to show a large variety of postures when an object is presented accordingly. Fig. 9 shows a series of snapshots taken from an experiment. From second 1-7, ASIMO is reaching for the green bottle with its right hand. This corresponds to the first phase in Fig. 8. At second 8, the object gets out of reach of the right hand, and the strategy selection mechanism selects the left hand reaching strategy, still while the robot is standing (Second phase in Fig. 8). At second 12, the object can neither be reached with the left hand while standing. The strategy selection mechanism now selects to reach for the object with the left hand while walking towards it (Third phase in Fig. 8. The whole body motion control generates smooth motions and is able to handle even extreme postures which gives a very natural and human-like impression even to the casual observer.

The visual system is capable of running at 12.5 fps (80 ms loop time), its main limitations are the stereo disparity calculations (50 ms) and the transmission time of the color images via the fast ethernet connection (maximum 15 fps). The data connection between the onboard and offboard control hosts exchanges postures every 5 ms. The total reaction is mainly given by three parts: the latency between frame acquisition and the arrival of data on the offboard vision PC, the latency between its arrival and the start of the vision loop, and the computation side of the vision loop. The latency of the control loop and the sending to the onboard host can be neglected. The total latency between time of acquisition and arrival of proto-objects on the offboard control PC was measured to be between 150 and 210 ms, i.e. roughly 2 frames.

Due to this latency and the fact that the field of view is not so large, motion speeds of the interaction object have to be limited, especially in the range where the robot is able to reach the object. Furthermore, the interaction object is restricted to have a narrow color distribution and no possible distractors should be too close and have similar colors.

Some limitations of the current system are that the reaching is done with a vertical offset of about 10 cm to avoid occlusions of the object by ASIMO's hand, the limited field of view of ASIMO's current cameras, and the limited pan/tilt range which in this scenario limit the interaction range especially in vertical direction.

Obviously, since the object segmentation used is based on color, only certain objects can be used for interaction. Green objects were used mostly, but also tested with different colors like yellow and red - adjusting the color segmentation accordingly. These limitations can be reduced when incorporating a system like BASS.

We had mixed results when using distractingly colored backgrounds, as most of the time the background was ignored since it was outside the interaction range, but in some cases the disparity calculation failed and produced spurious targets which confused the robot.

Overall we are satisfied with the performance and hope to build on this system in our future research.

V. OUTLOOK

We plan to extend the collision detection scheme by a collision avoidance system that will be integrated into the whole body interaction mechanism.

Vision capabilities will be substantially enhanced and extended, building on the current concept of proto-objects. Furthermore we are interested in including a visual detection mechanism and object hypotheses generation for support surfaces in order to enrich the interaction possibilities of the system.

The presented selection mechanism is based on a predefined heuristics. Future work will focus on learning such selection criteria, and on making a step towards learning behaviors from demonstration.

ACKNOWLEDGMENT

The authors would like to thank the members of Honda's ASIMO and robot research teams in Japan for their support.

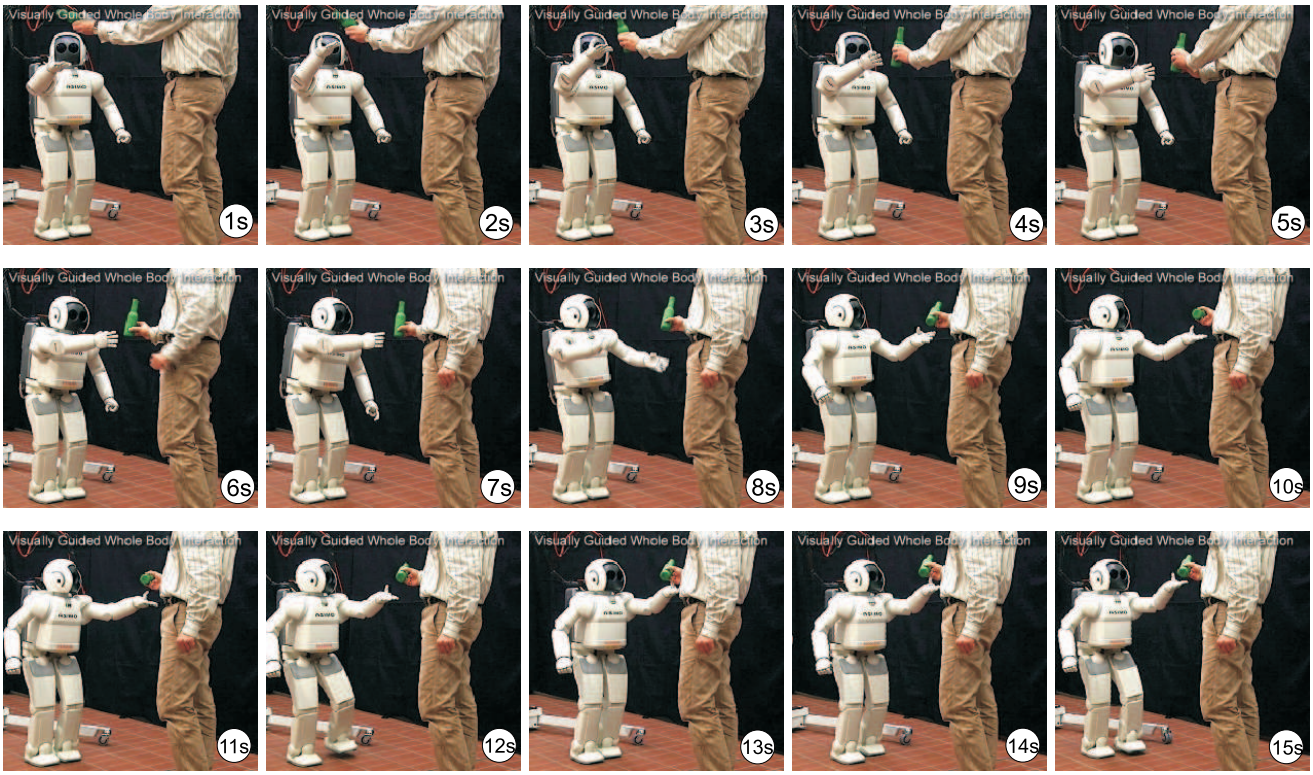

Fig. 9. Snapshot series from an experiment.

REFERENCES

[1] Fujita, Takagi, and Hasegawa, "Ethological modeling and architecture for an entertainment robot," in *ICRA*, 2001.

[2] ——, "An ethological and emotional basis for human-robot interaction," *Robotics and Autonomous Systems*, no. 3-4, 2003.

[3] C. Breazeal and B. Scassellati, "How to build robots that make friends and influence people," in *IROS*, 1999.

[4] R. A. Rensink, "Seeing, sensing, and scrutinizing," *Vision Research*, vol. 40, pp. 1469–1487, 2000.

[5] A. Clark, "Feature-placing and proto-objects," *Philosophical Psychology*, no. 4, pp. 443–469, December 2004.

[6] Z. W. Pylyshyn, "Visual indexes, preconceptual objects, and situated vision," *Cognition*, no. 1, pp. 127–158, June 2001.

[7] F. Orabona, G. Metta, and G. Sandini, "Object-based visual attention: a model for a behaving robot," in *CVPR*, 2005.

[8] L. Natale, F. Orabona, F. Berton, G. Metta, and G. Sandini, "From sensorimotor development to object perception," in *Humanoids*, 2005.

[9] J. Driscoll, R. A. Peters, and K. R. Cave, "A visual attention network for a humanoid robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998.

[10] S. Vijayakumar, J. Conradt, T. Shibata, and S. Schaal, "Overt visual attention for a humanoid robot," in *International Conference on Intelligent Robots and Systems IROS*, vol. 4, 2001, pp. 2332–2337.

[11] R. A. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, and M. M. Williamson, "The cog project: Building a humanoid robot," *Lecture Notes in Computer Science*, vol. 1562, pp. 52–87, 1999.

[12] D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, vol. 11, no. 7-8, pp. 1317–1329, 1998.

[13] A. Ude, C. G. Atkeson, and G. Cheng, "Combining peripheral and foveal humanoid vision to detect, pursue, recognize and act," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2003, pp. 2173–2178.

[14] C. Goerick, H. Wersing, I. Mikhailova, and M. Dunn, "Peripersonal space and object recognition for humanoids," in *Proceedings of the IEEE/RSJ International Conference on Humanoid Robots (Humanois 2005), Tsukuba, Japan*, 2005.

[15] A. Ceravola, F. Joublin, M. Dunn, J. Eggert, and C. Goerick, "Integrated research and development environment for real-time distributed embodied intelligent systems," in *International Conference on Intelligent Robots and Systems IROS*, 2006.

[16] T. Bergener, C. Bruckhoff, P. Dahm, H. Janssen, F. Joublin, R. Menzner, A. Steinhage, and W. von Seelen, "Complex behavior by means of dynamical systems for an anthropomorphic robot," *Neural Networks*, 1999.

[17] M. Nicolescu and M. J. Mataric, "Learning and interacting in human-robot domains," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 31, pp. 419–430, 2001.

[18] J. Tani and S. Nolfi, "Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems," in *Neural Networks*, vol. 12, 1999, pp. 1131–1141.

[19] M. Hirose, Y. Haikawa, T. Takenaka, and K. Hirai, "Development of humanoid robot asimo," in *IEEE/RSJ International Conference on Intelligent Robots and Systems – Workshop 2, Hawaii, USA*, 2001.

[20] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 12, December 1977.

[21] J. D. English and A. A. Maciejewski, "On the implementation of velocity control for kinematically redundant manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, pp. 233–237, May 2000.

[22] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley Publishing, 1991.

[23] M. Gienger, H. Janssen, and C. Goerick, "Task oriented whole body motion for humanoid robots," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 2005.

[24] ——, "Exploiting task intervals for whole body robot control," in *IEEE/RSJ*, 2006.

[25] K. Nishiwaki, M. Kuga, S. Kagami, M. Inaba, and H. Inoue, "Whole-body cooperative balanced motion generation for reaching," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 2004.

[26] H. Wersing, S. Kirstein, M. Götting, H. Brandl, M. Dunn, I. Mikhailova, C. Goerick, J. Steil, H. Ritter, and E. Körner, "A biologically motivated system for unconstrained online learning of visual objects," in *Proc. Int. Conf. Art. Neur. Netw. ICANN*, 2006.