A new paradigm of humanoid robot motion programming based on touch interpretation

Fabio Dalla Libera [a], Takashi Minato [b], Ian Fasel [d], Hiroshi Ishiguro [c,b], Enrico Pagello [a], Emanuele Menegatti [a]

[a] *Intelligent Autonomous Systems Laboratory, Department of Information Engineering (DEI), Faculty of Engineering, University of Padua, Via Gradenigo 6/a, I-35131 Padua, Italy*

[b] *ERATO, Japan Science and Technology Agency, Osaka University, Suita, Osaka, 565-0871, Japan*

[c] *Department of Adaptive Machine Systems, Osaka University, Suita, Osaka, 565-0871 Japan*

[d] *The University of Texas at Austin, Department of Computer Sciences, 1 University Station C0500, Taylor Hall 2.124, Austin, USA*

Abstract

Most humanoid soccer robot teams design the basic movements of their robots, like walking and kicking, off-line and manually. Once these motions are considered satisfactory, they are stored in the robot's memory and played according to a high level behavioral strategy. Much time is spent in the development of the movements, and despite the significant progress made in humanoid soccer robots, the interfaces employed for the development of motions are still quite primitive. In order to accelerate development, an intuitive instruction method is desired. We propose the development of robot motions through physical interaction. In this paper we propose a "teaching by touching" approach; the human operator teaches a motion by directly touching the robot's body parts like a dance instructor. Teaching by directly touching is intuitive for instructors. However, the robot needs to interpret the instructor's intention since tactile communication can be ambiguous. This paper presents a method to learn the interpretation of the touch meaning and investigates through experiments a general (shared among different users) and intuitive touch manner.

Key words: motion editor, human-robot interaction, touch
PACS:

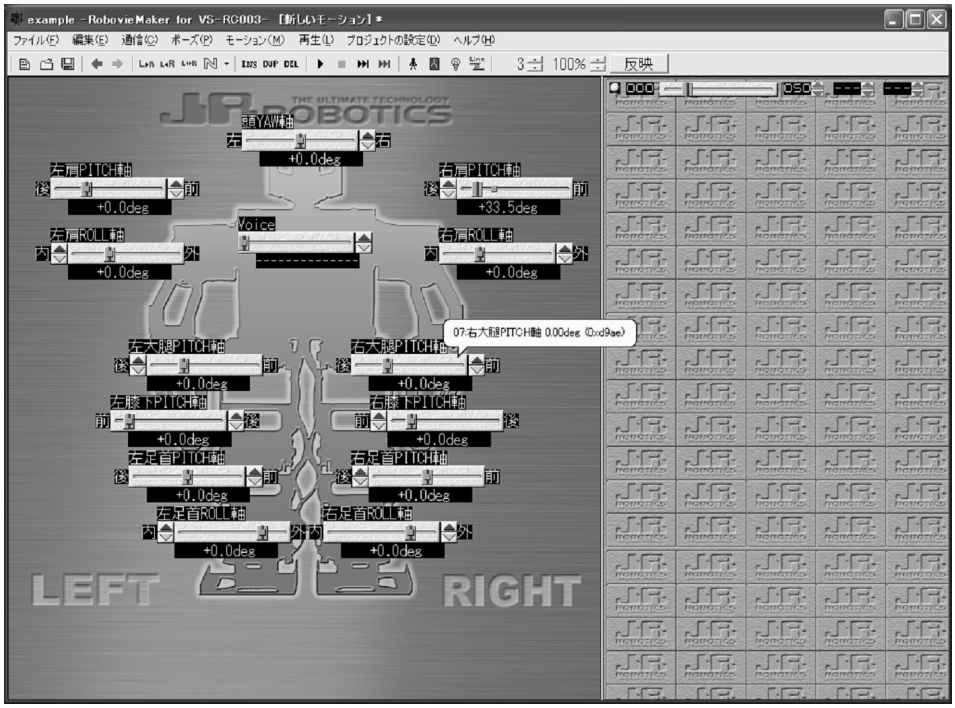


Fig. 1. Robovie Maker, a commercial motion development software developed by VStone.

1 Introduction

Since the introduction of humanoid robot soccer competitions, small humanoid robots have seen great progress in terms of both hardware and software. For instance, walking has been extensively studied, and several advanced methods for motion generation [1–6] have been proposed. Nonetheless, many teams still rely on handcrafted motions. The most widespread way of realizing robot movement is to specify the motion as a set of "frames"; that is, a set of instants in time for which the position of each and every joint is provided [7,8]. Those interfaces have one slider for each of the joints, and the user has to choose the angle assumed by the joint at the frame being edited. Fig. 1 shows a commercial motion editor based on this design principle. Developing motions with these low level interfaces requires much time and is not at all straightforward for novice users. The user would instead be able to naturally and intuitively teach motions to the robot if these "artificial" interfaces could be avoided.

Our goal is to develop a method by which humans can intuitively edit a robot motion without any special training. Observing how a dance instructor [9] or a sport coach teaches motions, we notice that, with simple touches, the instructor intuitively conveys plenty of information on how to modify the trainee's movement. Touch is particularly appealing as an intuitive method for humans to teach robots, and has been employed to program robot arms,

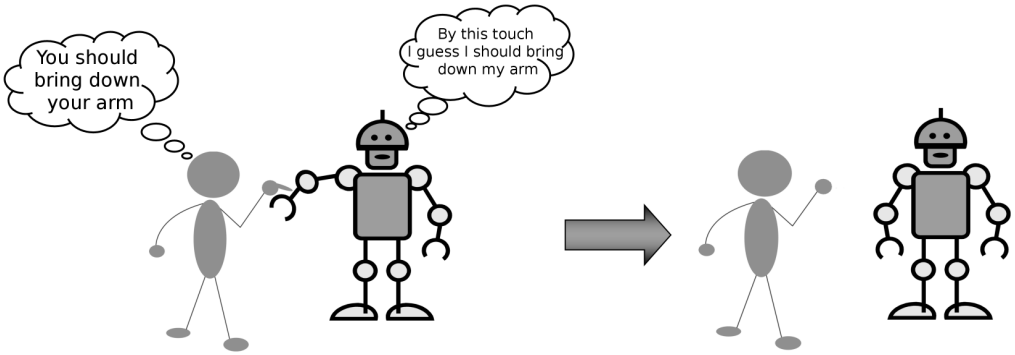


Fig. 2. The basic idea of teaching by touching: the user directly touch the robot to teach how to modify the motion and the robot interprets the meaning of the touch.

for example, by Voyles and Khosla in [10] and, more recently, by Grunwal et al. in [11] and by Yoshikai et al. in [12]. Taking inspiration from these facts, an interface was designed in which the human operator modifies the motion by touching the robot parts (Fig. 2). We intend to avoid the artificiality of classical interfaces by adopting "teaching by touching".

The teachers' touching is a method of encoding and transmitting their internal image of what the robot postures [1] should be. To make communication successful, the robot must then interpret the meaning of these touches in terms of adjusted body postures. However, for the robot this reconstruction process is not a trivial task. Not only can different touches have the same meaning – for instance, touching several different parts of the arm could all mean that the arm should move backwards – but similar touches could have different meanings depending on the context. For example if the robot is standing, touching the upper part of one leg could mean that the leg should bend further backwards. However if the robot is squatting, the same touch could mean that the robot should move lower to the ground by bending its knees (see Fig. 3).

This context dependency is not the only problem, since the style and method of touching could be (in part or totally) user-dependent. To avoid this ambiguity we could design a fixed protocol and force the user to use it. This, however, would strongly reduce the intuitiveness of the teaching. Fig. 4 depicts this fact. Suppose the user has an idea of the posture modification she/he would like to apply (for instance, raise the leg). If the protocol is fixed the user must decide which touch, according to the touch protocol, allows her/him to

[1] The method is not restricted to modification of static postures. It is assumed that an instructor touches the robot during its motion to tell when the modification should be applied. Clearly there will be practical problems when this technique is used with a real robot, instead of a 3D representation as was done here. Future works must analyze which kind of dynamic motions are realizable by the presented approach.

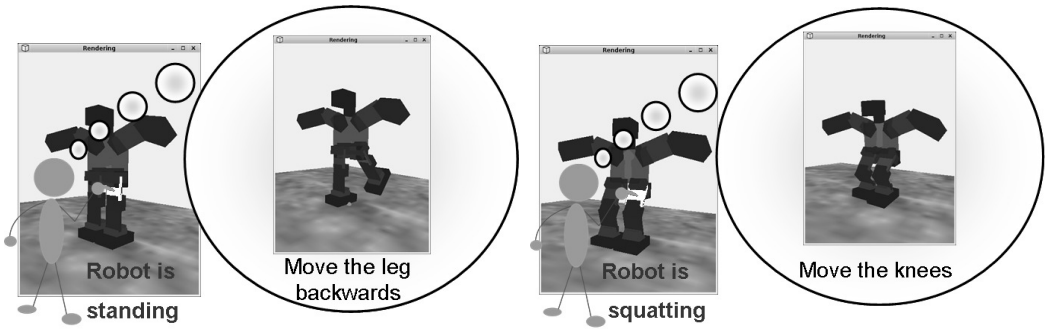
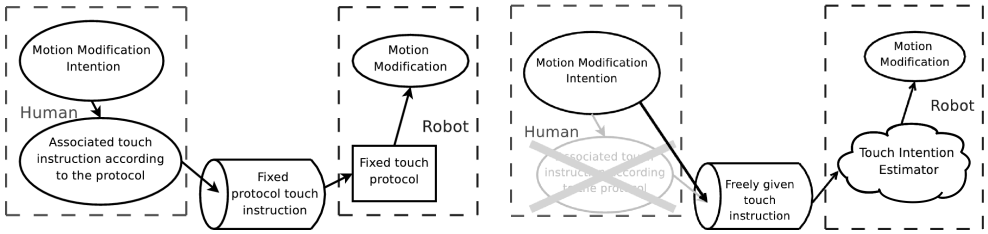


Fig. 3. An example on why the meaning of touch is context based. The user touches the robot in the same way, but the desired posture modification (bend the leg and bend the knees, respectively) is different because the robot posture is different.



(a) Fixed protocol (b) Intention Estimation

Fig. 4. Fixing a protocol (left figure) requires the user to think which instructions would allow him/her to perform the intended modification. If the robot is able to adapt to the user, instead, it will be able to decode touch instructions freely given by the user (right figure), and the user mental effort will be reduced.

make the desired modification (Fig. 4(a)). Conversely if the system is able to adapt to the user and estimate his/her intention, the user just needs to touch the robot spontaneously with no mental effort (Fig. 4(b)). In other words, by making the robot's instruction interpretation powerful enough we can let the user express her/his intention without the need to learn and adapt to an artificially designed protocol. We can therefore expect that such kind of interface could reduce the time spent by inexperienced users.

As soon as we start to consider the development of such adaptive interfaces we can hypothesize that while some touches will have the same meaning for most users, others will have different meanings when provided by different people. The results here presented provide a first verification of this initial guess. Understanding to what extent the touch meaning is user dependent and in general what is user dependent and what is not will allow us to decide which elements of the touch instructions can be assumed as "generally recognized" and therefore be built-in in future works. In other words, if a common touch manner exists, we can use this knowledge to make the robot quickly adapt to the users. Section 2 describes the developed interface, while section 3 describes

the experiments performed to verify our hypotheses. The results are reported in section 4 and an overview of the findings is provided in section 5. Section 6 summarizes the content of this paper and presents possible future directions.

2 Teaching by touching

2.1 Concepts

As briefly reported in the introduction, touch is very intuitive and effective for interpersonal communication. However, often no direct mapping from touched part to modified joint angle is perceived as natural, intuitive. In fact, this one-to-one mapping would be quite similar to the one used in classic slider-based interfaces.

The robot must be able to decode the user's intention conveyed by the touch. As explained in the previous section the system must be able to adapt its touch instruction interpretation to the user. In our method, the robot at first cannot interpret the user's intention at all. When the user thinks the robot cannot understand the intention of the touch instruction, the user teaches the correct interpretation through another low-level protocol which is shared between the user and the robot. The robot then comes to interpret the touch instruction by supervised learning. Fig. 9 schematizes this approach. In detail a classical slider-based interface was implemented to allow the user to provide the meaning of touch instructions.

The examples can be collected on-line, during the development of robot motions. This brings two advantages. First of all no special session where the user is required to provide how she/he would touch the robot to express certain predefined modifications is required. Secondly the human operator can identify when the system fails to predict her/his intention expressed through touch, and can provide, by the shared protocol, the intended joint modification, so that the mapping between touch instructions and estimated modification intentions can be refined where it needs to be. Ideally the system therefore keeps improving its knowledge base during the motion development and users need to teach the meaning of the touch instructions less and less frequently.

The interface was designed following the classical "observe and correct" motion-development approach, similar to a human dance or sports learning section. In each teaching episode, the human teacher watches the robot performing a motion, observes what is wrong or could be improved, and touches the robot's body parts to instruct the robot how to refine the motion. Since humanoid soccer robots usually lack of touch sensors, we introduced virtual tactile sensors,

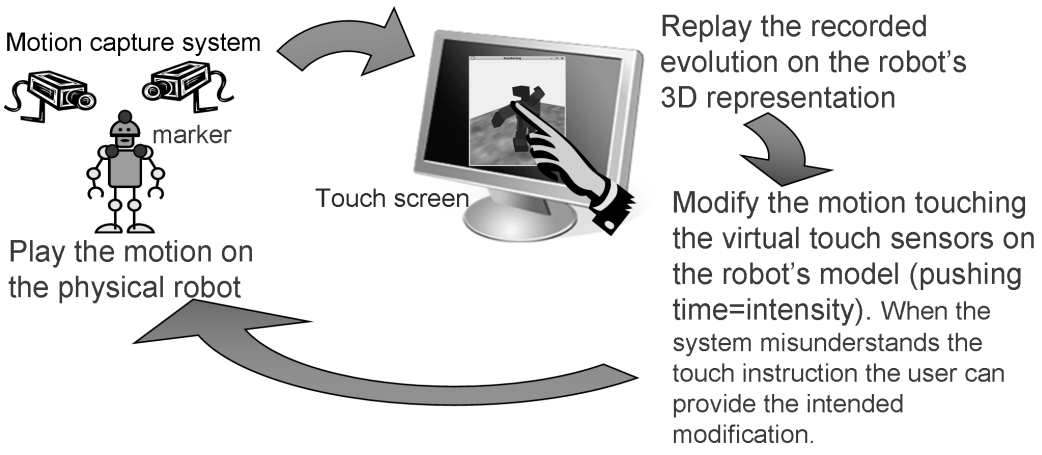


Fig. 5. Motion development cycle.

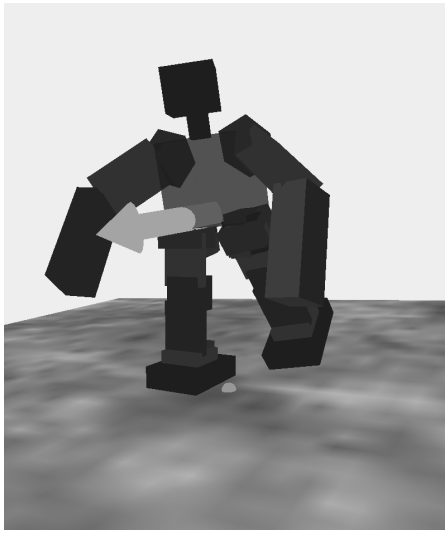


Fig. 6. The simplified 3D model of Vstone's Vision4G. The 3D model rendering also shows the projection of the center of gravity onto the ground (represented by a sphere on the ground) and its velocity (represented by an arrow).

i.e. the user interacts with a 3D representation of the robot, but obviously the presented approach could be applied to direct physical interaction between the robot and the user. Fig. 5 illustrates this development process.

The developed interface presents a time-line slider and the user, after selecting the instant in which the motion should be modified, touches the robot's 3D representation by employing a touch screen or a mouse (Fig. 6).

Given a touch instruction the touch-interpreter module needs to convert it to a modification of the current (selected on the time-line) posture. The modified posture becomes a key-frame at the selected time, and the complete modified motion is generated by linear interpolation between consecutive frames. Since

the touch instruction meaning is context dependent, the learning algorithm obtains as input not only the touch duration of each of the touch sensors but also the contextual information. Currently the context consists of the angle of each of the joints, the orientation (roll, pitch and yaw) of the robot, and the velocity of the center of gravity. The joint angles are needed because the meaning of touches may depend on the posture, as in the provided example in which touching the lap means different things depending on whether the robot is standing or squatting. Likewise, the meaning of a touch may also depend on the orientation, for instance whether the robot is standing or lying down. Finally, touches may also depend on the velocity, especially if the robot is moving fast (for instance if it is falling down). It is possible that the velocity of each single link should also be included, however for the moment we put off investigating this and other features for future work. Employing a supervised learning algorithm and requiring the user to supply examples of touch instructions and their desired interpretation allows the direct study of the input-output couples, giving insights on the way humans intuitively use touch to express their intention.

2.2 Learning algorithm

As described in the previous section, the role of the learning algorithm is to realize a mapping between the tuple $(touch information, context)$ and expected intended modification of joint angles. Various approaches could be used to realize this mapping, for instance neural and in particular RBF (Radial basis functions) networks, but one solution that appears quite straightforward is to employ the k-Nearest Neighbor algorithm. Despite its simplicity, this algorithm performs very well and has been effectively employed in many fields [13] like speech or character recognition, for instance in [14] and in [15]. Unlike some other classification algorithms the output of the k-Nearest Neighbor can be continuous, if the example classes can be expressed with continuous numerical values. In such a case the output of the algorithm can be calculated as an average (which can also be weighted by some function of the distance) of the k nearest neighbor classes. In our case each output class is a vector, containing the modification to apply to each joint and to have a mapping without discontinuities we chose $k = \infty$.

More formally, each example (a point in a high dimensional space) consists of an input I_i and an associated intended joint modification vector M_i. Given an input I_*, the system output vector M_* can be obtained by weighting the joint modifications present in the collected examples M_i, with weights ω_i calculated employing the distance (in the high dimensional space) between the system input I_* and each example coordinates I_i. Concretely, indicating with E the

number of collected examples

$$M_* = \sum_{i=1}^{E} \omega_i M_i \tag{1}$$

A straightforward use of the k-Nearest Neighbor algorithm with a Euclidean or Mahalanobis distance function presents two problems in the realization of the mapping between touch information and their context to a joint modification. First of all "priority" should be given to the touch information over the context. This is to avoid the output being determined mainly by the context instead of by the pushed parts, as happens if the touch information is given no priority over the other features of the input. As a trivial example, suppose a user is focusing on a leg motion and therefore only provides examples involving a leg, then when she/he will push on an arm this will cause the leg to move, while for most users in such cases of no available knowledge the most reasonable choice would probably be the null modification (no joint movement). This kind of behavior is likely to happen since the context space dimension is very large - one dimension for each degree of freedom plus six dimensions, three for the robot orientation and three for the center of gravity velocity - so it is very difficult to have, for a particular kind of touch pattern, enough examples with different contexts, such that the touch pattern can be influential in most of the contexts. This is what is usually required for the most basic features of the mapping; for instance, pushing the right cheek should translate into turning the head toward the left, regardless of the position of the legs or arms. To solve this problem, given an input vector I_* and in particular the touching information T_*, the output M_* is calculated considering only the examples having a set of pressed sensors (i.e., sensor having a pushing duration greater than zero) the same set of sensors pressed in T_* or a subset of them. In other words, indicating with n the number of sensors and with the notation $T_*[s]$ and $T_i[s]$ as the pushing duration of the s-th sensor in the system input T_* and in the i-th example touch information vector respectively, the i-th example is considered if and only if

$$\bigvee_{s=1}^{n} (T_i[s] > 0)) \wedge (T_*[s] = 0) \tag{2}$$

is false. This is analogous to setting ω_i equal to 0 in equation 1 for the examples in which equation 2 holds. Some examples are shown in Fig. 7.

With this approach most of the counterintuitive behaviors are avoided and, although as a drawback the generality of the mapping is limited and discontinuities are introduced when the set of pressed sensors varies, a great speed-up of the learning algorithm is achieved (in preliminary tests the measured speed-up was slightly over 2000%). This allows the calculation and display of the predicted joint modification in real-time, i.e., visual feedback of the interpreted touch meaning while pushing a sensor.

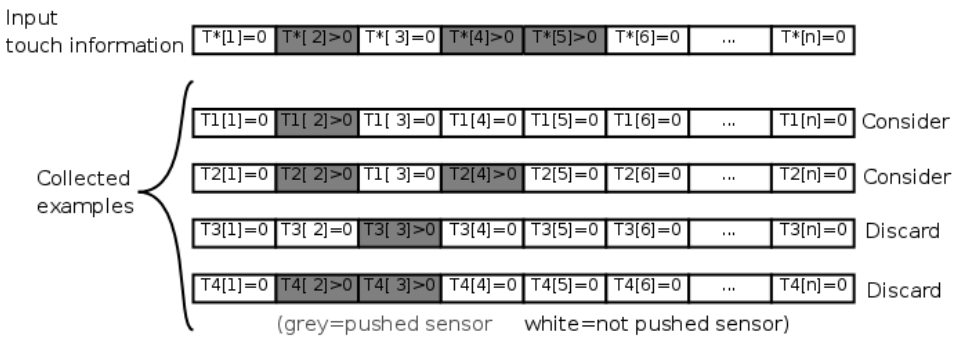


Fig. 7. Examples of considered and discarded examples applying the described rule.

The second problem arises from the fact that every distance is a symmetric function. Suppose to have just one training example, where a sensor was pushed for 300 milliseconds, and this corresponded to a single motor joint change of 40 degrees. A user might naturally expect that pushing for less time will cause a smaller change in that joint, while a longer press should produce a larger joint angle change. Conversely the system behavior with a distance based weighting would be that any touch on that sensor with duration different from 300ms, both longer and shorter, would result in a smaller angle change. For instance, using as a weighting function

$$\omega_i = \frac{1}{1 + \|I_* - I_i\|}$$

pressing the sensor for 200ms or for 400 ms would give the same modification. Fig. 8 illustrates this problem.

To overcome this counterintuitive behavior, the weight ω_i, by which the i-th example output M_i is weighted, is calculated as a product of two factors α_i and β_i, that is, $\omega_i = \alpha_i \beta_i$. Given T_*, the touching information of the input vector, and the various example touch information vectors T_i, α_i is calculated as

$$\alpha_i = \prod_{s:T_i[s]>0} T_*[s] / T_i[s]$$

This value keeps increasing (linearly) as the pushing time increases; thanks to the condition $s : T_i[s] > 0$ if multiple sensors are touched and several examples are considered in the calculation, increasing the pushing time of one sensor will increase just the weight of the examples in which such sensor is pushed, and will not vary the weights of other examples; The second factor β_i accounts for all information not used in the calculations of α_i

- the sensor information $T_*[s]$ and $T_i[s]$ for the sensors s such that $T_i[s] = 0$;
- the joint angles of the robot in the system input (P_*) and the angles recorded in the i-th example P_i;
- the orientation present in the system input O_* and the one of the i-th

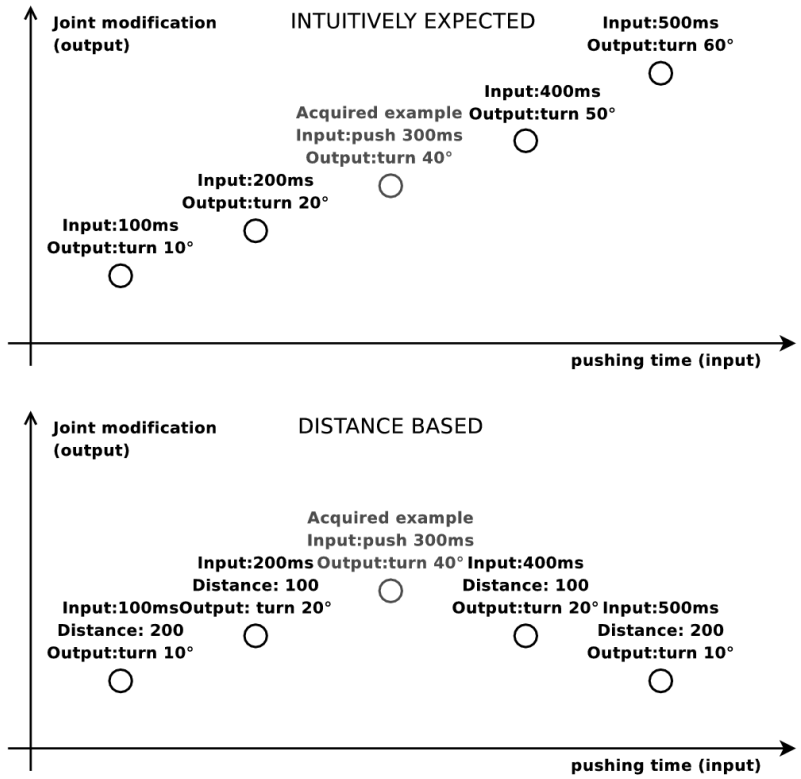


Fig. 8. Expected behavior versus the behavior obtained weighting the output by a decreasing function of the distance

example O_i;

- the center of gravity velocity vectors V_* and V_i, relative to the system input and to the i-th example respectively.

It was chosen to calculate each β_i as

$$\beta_i = \frac{1}{1 + d_i} \tag{3}$$

where d_i is, roughly speaking, a measure of the diversity of the current input I_* and the i-th example input I_i.

Denoting the Euclidean norm by "$\|\|$", d_i is given by

$$d_i = \sqrt{\sum_{s:T_i[s]=0} (T^*[s] - 0)^2 + \|P - p_i\|^2 + \|O - o_i\|^2 + \|V - v_i\|^2}$$

where each vector component is scaled by its variance in the example data set since the units are heterogeneous.

The meaning of the way d_i is calculated is clearly understandable, since, apart from the touching information, it corresponds to a Euclidean norm. The struc-

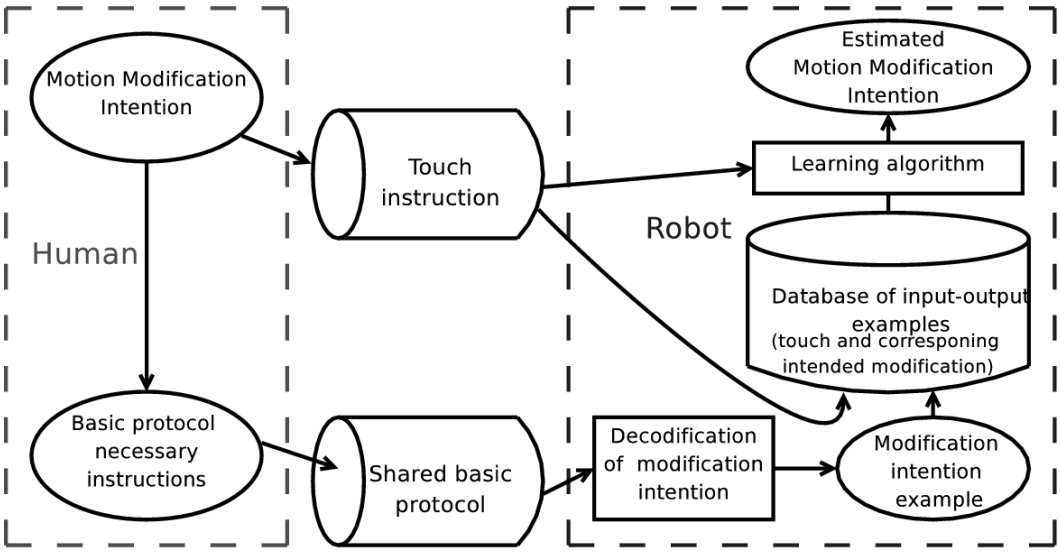


Fig. 9. The developed interface employing examples of input (touch pattern) and corresponding output (intended motion modification) converts touch instructions in estimated intended motion modification.

ture of equation 3, instead, emerges from practical experiments: several decreasing functions were tested and the one which appeared to give the most intuitive behavior, $f(x) = 1/(1+x)$ was chosen. Obviously, a deeper and more formal analysis should be conducted.

We can notice that the weighted average given as estimate of the desired posture modification depends on the number of the samples. For instance if I_* has distance 2δ from an example I_A (with joint modification M_A) and distance 3δ from two examples I_{B1} and I_{B2} with $I_{B1} \approx I_{B2}$ and $M_{B1} \approx M_{B2}$ then the output will be more similar to $M_{B1} \approx M_{B2}$ than to M_A. This has as advantage that the effect of wrong examples can be canceled by providing new examples. However, even if in practical experiments we never faced this problem, if the density of the provided examples is highly heterogeneous this behavior could be undesired. Modifications of the algorithm to make the output independent of the density of the examples will be tackled in future works.

3 Experiment

3.1 Realization of physical robot motions

To perform our experiment we employed VStone's [2] VisiON 4G (see Fig. 10(a)), a humanoid robot with 22 degrees of freedom used in the Robocup competition. A walking and a jumping motion were developed (actually since the servo torque is not sufficient for liftoff given the robot's weight, the jumping motion was realized with the help of a rubber band pulling the robot up).

As already explained the robot is not equipped with touch sensors, so we developed a simplified 3D model of the robot that was touched by the user employing a touch screen(Fig. 6). This simplified representation models each robot's part by a box and each of the box faces simulates a single touch sensor (so when one sensor is pushed we can assume that the user is applying a force normal to the face). Multiple sensors can be clicked one after the other, allowing to simulate a simultaneous pressure of multiple sensors. Each sensor gradually changes its color from green to red while being pushed, so the users have a visual feedback of the touch instructions they provide.

As specified in Section 2.1 the motion development consists in alternating

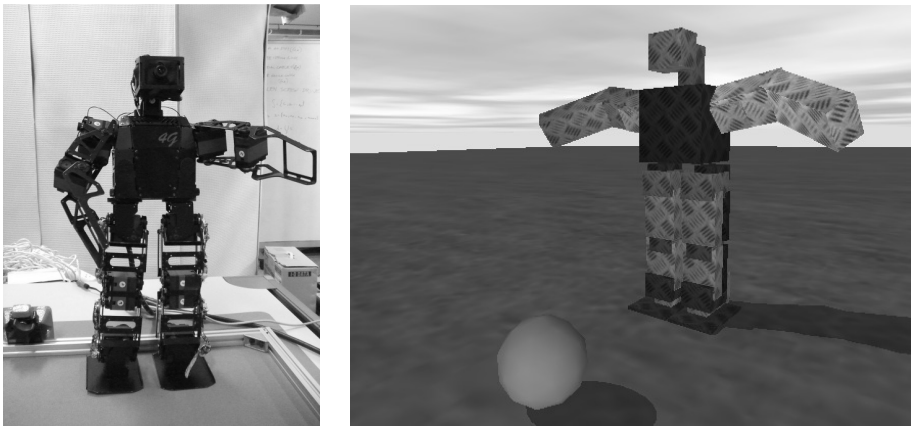
(1) motion editing, performed by touching the 3D model,
(2) execution of the motion on the real robot

To allow reviewing of the evolution of the motion each time the motion was executed by the real robot the robot's position and orientation were recorded using a capture system from Motion Analysis Corp [3].

3.2 Analysis of user dependency of the touch instructions

Once we verified the feasibility of the proposed method by practical realization of motions on a real robot we conducted an experiment to obtain insights into which features of the touch instructions are shared by human operators and which strongly depend on the user. We asked six subjects (hereafter denoted by A, B, C, D, E and F) to develop a walking and a kicking motion, two fundamental capabilities required by a humanoid soccer robot. The subjects are all Italian male computer science students, and their age is in the range

[2] `http://www.vstone.co.jp`
[3] For details see `http://www.motionanalysis.com/applications/movement/neuro/eaglesystem.html`



(a) (b)

Fig. 10. Vstone's Vision4G and its simulated counterpart.

23-27 (mean 24.5, standard deviation 1.87). To assure all the users to be in exactly the same condition we asked them to develop motions using a simulated robot, instead of using the real robot as done in the previous experiments. We simulated VisiON 4G in a custom made simulator based on ODE[4]. Actually to allow faster execution times the simulator approximates the robot's parts as boxes, as visible in Fig. 10(b).

We asked the subjects to freely touch the robot to give commands and to teach their meaning using the sliders when the robot did not understand them. Given the test subjects' similar background (all male computer science students) we expected them to have provided very similar touch instructions. However, as will be shown in the next session, the touching manner greatly varied, even in this restricted experiment.

4 Results

4.1 Complexity of the touch instruction interpretation

As reported in the previous session a first validation of the proposed approach was realized by a single non experienced user developing a jumping (see Fig. 11) and a walking motion. To have a preliminary comparison of the motion development time required by the use of a classical interface and by the proposed method we realized the same motion (jumping) with both the interfaces: while developing the motion employing only the sliders took over 40 minutes developing the motion using touch required just 17 minutes.

[4] ODE is an open source library for simulating rigid body dynamics. See http://www.ode.org/ for details.

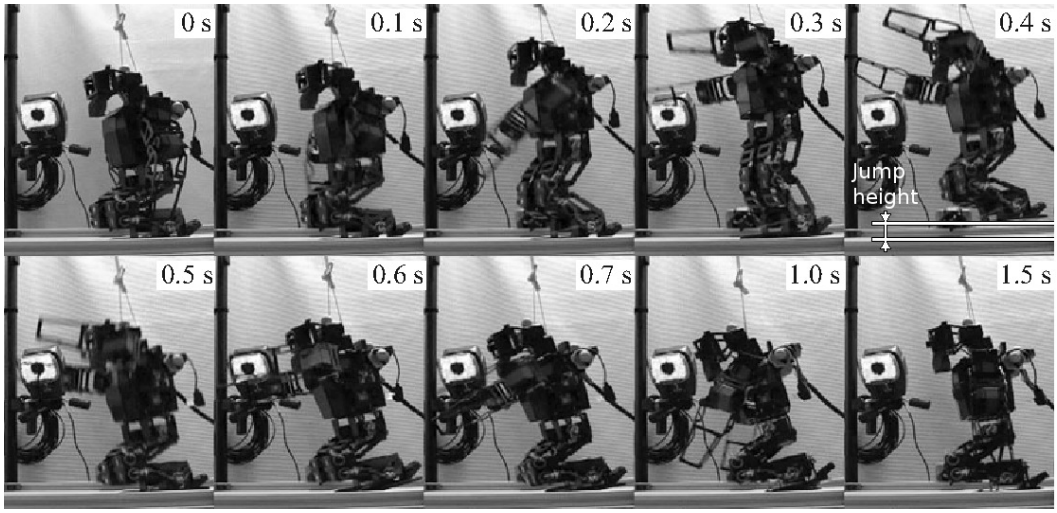
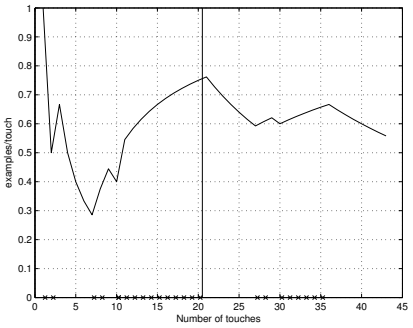


Fig. 11. Image sequence of the jump motion.

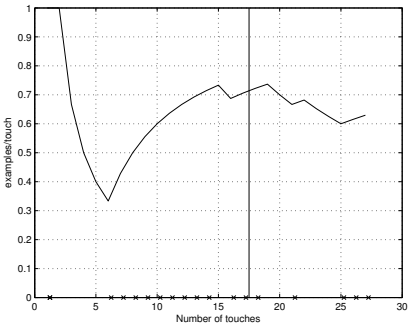
Although a single experiment is not sufficient to assure that the motion development time is always much shorter using the proposed method we can say that these preliminary results are encouraging. A statistically significant test will be provided in future works. We also noticed that, as expected, the system keeps improving its understanding of the touch meaning so the user needs to provide the examples less and less frequently. Fig. 12 shows quantitative results obtained in the successive experiment, in which six subjects were required to realize a walking and a kick motion. The X axis represents the number of touches [5] while the Y axis is the ratio between the number of examples provided by the user (when the system fails to estimate the user intention) and the number of touches provided. As expected, for most of the users, the need to provide the correct interpretation of the touch decreases over time (the graphs in Fig. 12 on average correspond to decreasing slopes) proving that the learning algorithm can correctly estimate the user intention, at least partially. In fact if the system had always failed to understand the touch meaning the graphs would have been horizontal lines, since the ratio between examples and touches would have always been equal to one. We can notice some fast increases of the ratio due to the pressure of sets of sensors previously not used. For instance if the user teaches how to deal with the arm sensor information she/he can modify the arm posture nearly exclusively by touch but if then she/he starts to press the leg sensors she/he needs to provide examples on how to modify the leg joints when such sensors are pressed.

To have insights on the complexity of the mapping from touch instructions to joint modifications we checked whether a simple linear model could be sufficient. We applied linear regression to the examples collected during the development of the jumping and walking motions on the real robot. Hereafter let us identify by $JUMP$ and $WALK$ the sets of examples acquired during the

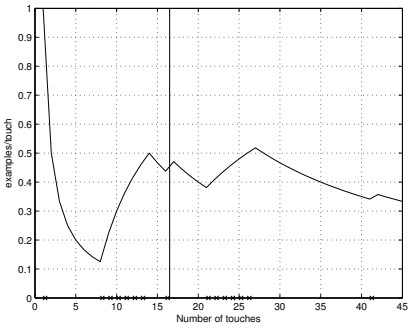
[5] A single touch can consist in multiple sensors being pushed at the same time.



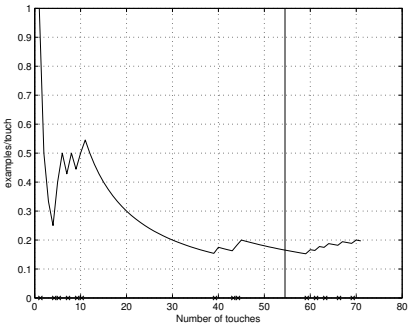
(a) Subject A



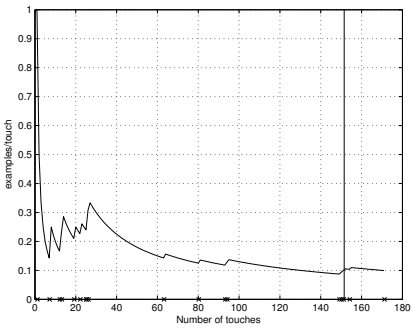
(b) Subject B



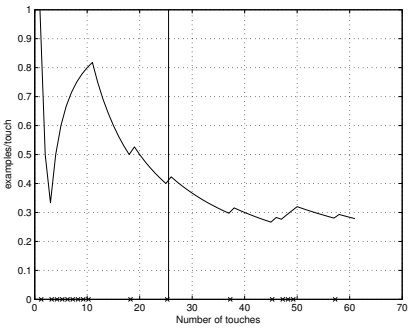
(c) Subject C



(d) Subject D



(e) Subject E



(f) Subject F

Fig. 12. Ratio between the examples and the touches over time for the six subjects. The X axis is the number of touches provided by the user. The graph indicates the ratio between the examples provided by the user (when the robot mistook to interpret the touch) and the number of touches. The vertical line present in each graph indicates when the development of the walking motion finished and the realization of the kicking motion began.

development of the two motions. Given the data set D, let us denote by X_D the matrix having as the i-th row the i-th example input (touch information and context) followed by a 1 (to include the possibility of a constant term in the mapping) and similarly let us indicate by Y_D a matrix having in the i-th row the i-th example output (the joint modification provided by the user). Then, $Y_D = X_D L_D + \epsilon$ (where ϵ is a matrix expressing the error due to the linear

approximation) or, in other terms, $Y_D \approx X_D L_D$. The matrices L_D (where D can be $JUMP$, $WALK$) were calculated by the least square method, that is

$$L_D = (X_D^T X_D + \alpha I)^{-1} X_D^T Y_D \tag{4}$$

where the T superscript indicates the transpose operation, I is the identity matrix, and α is a small number which can be interpreted as an estimate of the standard deviation of a Gaussian noise affecting the data [16]. All the results reported here were obtained setting it to 0.1.

The importance of the input features (given by the absolute value of the coefficient in L_D) in determining a joint modification calculated in this way, at least for the sensors, has a strong relationship with common sense, since, for instance, the sensors identified as important for determining the head orientation are mainly the ones on the head. Nonetheless, the linear model seems not articulated enough to capture the structure of the touching data and often overfit them. A comparison with the k-Nearest Neighbor algorithm showed that the latter performs better on validation data not used for the training. More specifically:

(1) the matrices L_{JUMP}, L_{WALK} were calculated using the two "training sets" by using (4);
(2) each of the matrices L_D were used to predict the output for each of the datasets (used as "test sets"), that is, $\hat{Y}_{C,D} = X_C L_D$ were calculated for each possible combination of C and D (C and D can be $JUMP$ or $WALK$);
(3) the differences $E_{C,D} = \hat{Y}_{C,D} - Y_C$ were calculated. The i-th row of $E_{C,D}$ provides the error in the prediction of the i-th example of the test dataset C when the linear model is constructed based on the dataset D;
(4) for each training dataset D and each test dataset C the average error magnitude over all the examples of the test dataset was determined.

Similarly, the k-Nearest Neighbor algorithm (with the described weighting schema) was tested giving each time as training set (set of examples used to calculate the output) just one of the data sets. Table 1 provides a comparison of errors made by the two algorithms; as can be seen especially observing the second row, the k-Nearest Neighbor algorithm strongly outperformed the linear model in the test.

These results supports our hypothesis that the mapping between touch instructions and desired joint modification is not trivial and cannot be explained by a linear model. In detail, we can hypothesize that the mapping depends on the context in a non linear way.

Table 1
Average errors in the output prediction by linear regression and by k-Nearest Neighbor algorithm.

Training Dataset	Test Dataset	LR error	KNN error
JUMP	JUMP	0.2846	0.1985
JUMP	WALK	13.717	0.5938
WALK	JUMP	2.0567	1.0198
WALK	WALK	0.0314	0.0588

4.2 Analysis of user dependency of the touch instructions

After verifying that the proposed learning algorithm performs sufficiently well in estimating the user intention, we carried on an experiment that aims at identifying features of the teaching instructions that are shared between most of the users. The knowledge of these common features could then be employed in future works to speed-up the learning of the instruction meaning.

As a first step of this analysis we identified for each user which joints he moved when he touched one sensor (possibly in combination with other sensors). This allows us to identify if there are direct mappings between sensors and joints that are intuitive for most of the users. In Fig. 13 (as well as in all the subsequent figures [6]) each row represents a sensor and each column represents a joint. The rows are divided in groups by the position of the sensor:

(1) body
(2) head
(3) right arm
(4) left arm
(5) leg
(6) left leg

Similarly the columns are divided into four groups according to the part that is moved by the corresponding joint:

(1) head
(2) right arm
(3) left arm
(4) leg
(5) left leg

[6] Figures are available with colors and full resolution at `http://robotics.dei.unipd.it/~fabiodl/papers/material/ras08/`.

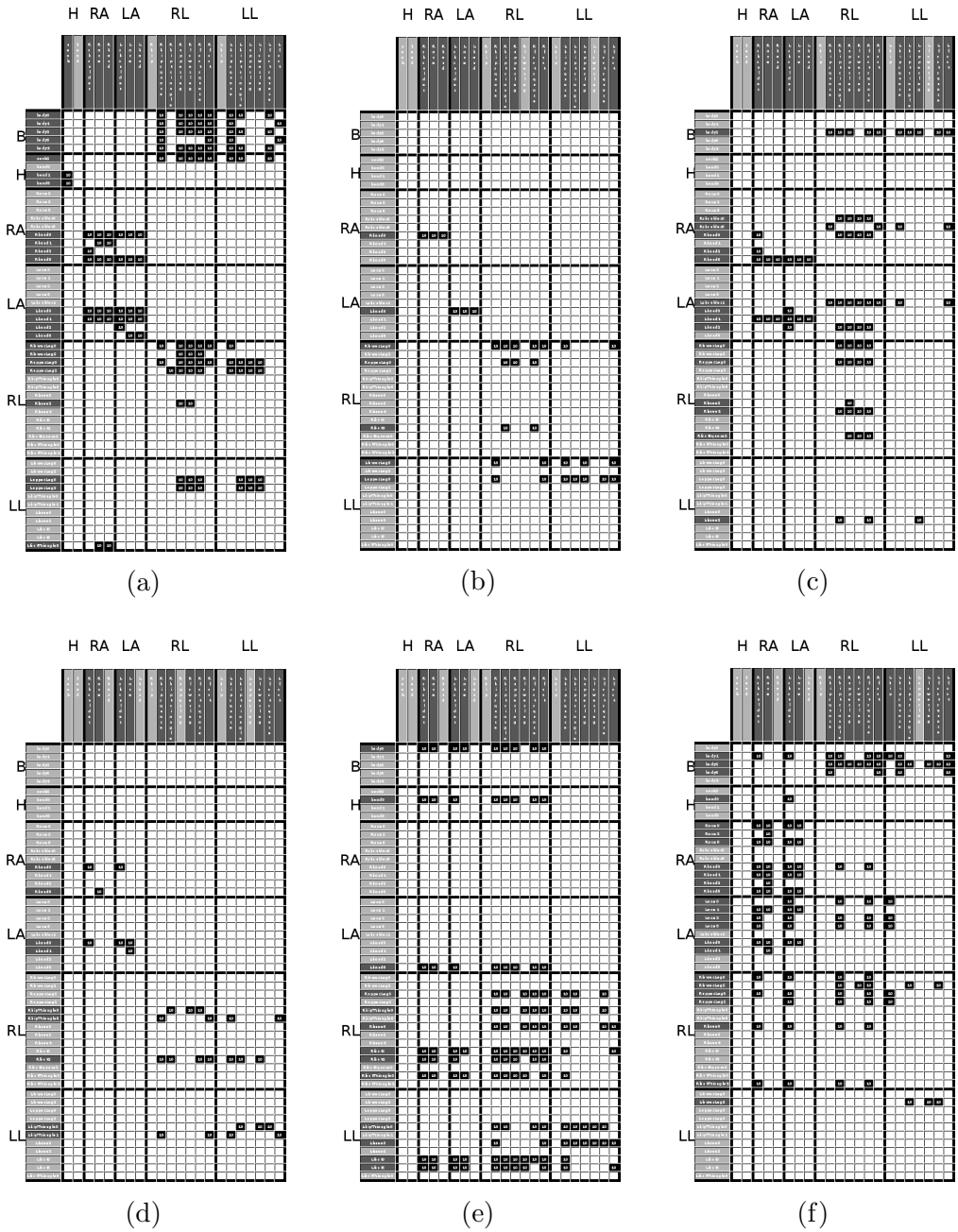


Fig. 13. Relationship between sensors (rows) and joints (columns) for each of the subjects. The thick lines divide the sensors (rows) and joints (columns) by the part of the robot in which they are placed. B, H, RA, LA, RL, LL stand respectively for body, head, right arm, left arm, right leg and left leg.

The sensors reported are just the ones used by at least one of the subjects. In each figure the headers of the sensors and joints (respectively the row and column headers) are colored darker if they were used by a user.

In Fig. 13 each intersection between the s-th sensor and the j-th joint is

colored black if in at least one touch the s-th sensor was pressed and then it was taught to move the i-th joint (and possibly other joints). We can observe that even if we fixed the task and used subjects with similar background, the mappings between sensors and joints are very different. To better grasp the similarities and dissimilarities of these relationships between the mappings we plotted Fig. 14(a). In this table the color of each entry indicates for how many users the sensor corresponding to the row was pressed to move the joint identified by the column. The color ranges from white (for the minimum value, 0) to black (for the maximum value, in the experimental data 5).

From Fig. 14(a) we notice that the combinations that appear more frequently are the ones which map sensors to joints of the same limb (that is, the sensor-joint combinations for which the row group and the column group). It is in fact very reasonable that most of the users touch what they wanted to move. We can see a correlation between the body sensors and the joints of the legs. The reason is that the users pushed the robot's body to bend the trunk, and this is done by moving the hip joints. At first glance the relatively high correlation between the sensors of one arm and the joints of the other (third row group and third column group, fourth row group and second column group) is surprising. A deeper analysis of the data showed that this is because most of the users pushed both hands at the beginning to close the arms, which are opened in the initial position (see Fig. 10(b)). It should be noted that if, after providing these kinds of examples, the user pushes just one arm the other one is not moved, because when inferring the meaning of a touch instruction the learning algorithm ignores the examples that have pushed sensors which are not being pressed in the provided touch instruction(see equation 2).

Observing the lower part of Fig. 14(a) we notice that just one subject touched sensors on the left leg and moved the right one. By Fig. 13 we can see that this was done by subject E. Unlike other people, E's table shows a correlation between the body and the arms. A direct analysis of the touch data showed that user E pushed the robot's belly and made it close the arms. He then touched the left knee and taught the robot to move the left leg forward (and not backward, as one could imagine). Since we never observed such kind of touch instruction we contacted E again and learned that this subject gave a very high-level meaning to the touch instructions. For instance with a single touch on the belly he wanted to say "go to the normal posture", considering as normal to lie the arms along the body sides. He then tried to teach with a single touch on a knee to make one step with that leg, i.e., to bring that leg forward (swing leg) and adjust the other one to keep equilibrium.

Paying attention to the sensors of the body, which are quite particular because they do not directly correspond to any joint, we can observe that subject A is the one who more extensively used them. More specifically he pushed the robot sides to make it swing, imagining the joints to be "elastic" and supposing that

applying a force the robot will move accordingly (given also the constraints posed by the ground). Actually this is the approach we expected for most users, and it strongly resembles the Yamane and Nakamura's "pin and drag" model [18]. Also subjects B and D used a similar approach, though they seemed to treat the interaction more abstractly. In detail, they identified one sensor which could be associated with the joints they wanted to move and pressed that part any time they needed to use that joint(s). This fact is easily observed in Fig. 15 that reports the sensors used by each of the subjects: subjects B and D employed very few sensors to develop all the motion. This strict correspondence between the joints to move and the pushed sensor was not maintained by subjects C and F, which used more sensors to express similar joint modifications.

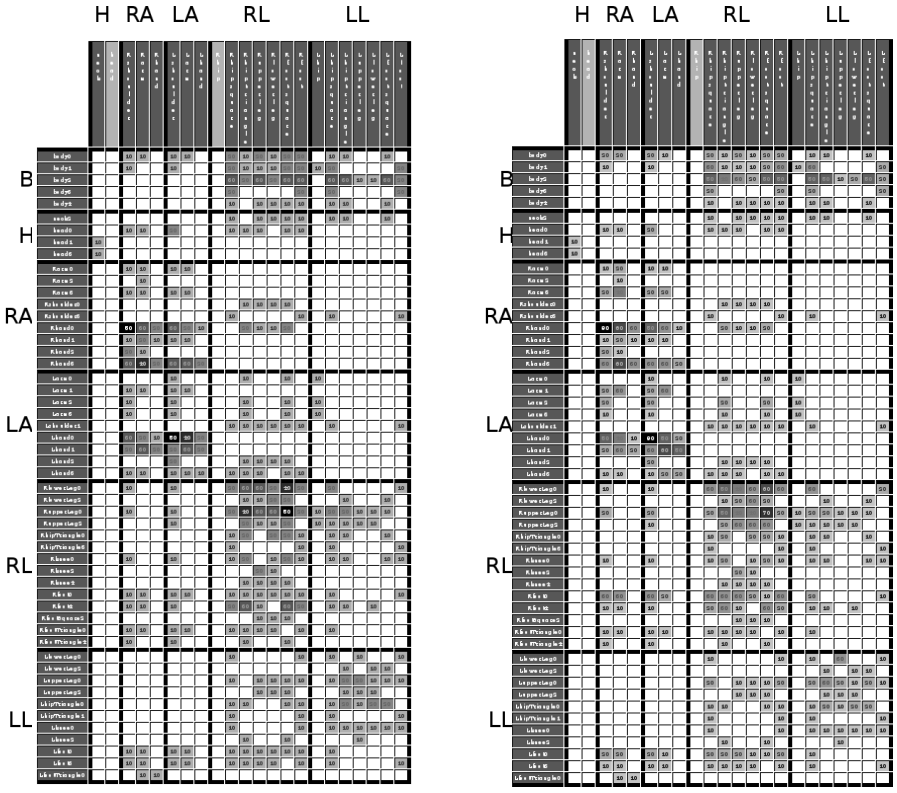
Analysis of the system behavior after it had been trained by each of the users gives support to these ideas intuitively derived from direct observation of the mappings. First of all we created a test set containing the touch instruction and relative context of all the examples provided by the users. For each I_i, consisting of a touch instruction and its context, we calculated the corresponding predicted joint modification using the estimators trained by each of the users. Let us denote by $M_{i,u}$ the output of the estimator trained by user u when I_i is given as input. We calculated the correlation between the mapping of each couples of users as

$$corr(u_1, u_2) = \begin{cases} 1 & if\ u_1 = u_2 \\ \underset{i}{avg}\left(\frac{M_{i,u_1} \cdot M_{i,u_2}}{|M_{i,u_1}||M_{i,u_2}|}\right) & if\ u_1 \neq u_2 \end{cases}$$

i.e. when the two users are not the same person the correlation is given by the average of the dot products of the outputs for the same input, otherwise it is trivially one. Each output is normalized, so that scaling of the magnitude of the joint rotations does not influence the coordination value. We derived a "distance" between the user applying the monotonically decreasing function to the coordination:

$$dist(u_1, u_2) = -\log(\|corr(u_1, u_2)\|)$$

The choice of this function is arbitrary, and comes from the results of practical experiments conducted in [20]. Finally we applied multidimensional scaling [19] and obtained Fig. 16, a 2D representation of the differences between the users' mappings. We can here observe how B and D are near, E is far from all the others and C and F are placed in another area. This quantitative analysis seems to confirm our intuitions based on the direct analysis of the tables representing the relationships between sensors and joints. In detail the horizontal axis, along which the distance is much bigger, has a role similar to what we identified as complexity of the mapping, while the vertical axis



(a) (b)

Fig. 14. The number of users that used each sensor-joint combination (figure a) and the number of times each combination was used by the users (figure b). The value of the entries is represented by the darkness of the color (white for the minimum value, black for the maximum one).B, H, RA, LA, RL, LL stand respectively for body, head, right arm, left arm, right leg and left leg.

could represent, for the same complexity, variations in the mapping. We can suppose that A appears near B and D in the low dimensional plot because for many sensors users B and D decided a fixed mapping that is very similar to the application of force to the robot, i.e. the consideration done by subject A.

Given the clear partition in at least three groups of users we can assume the existence of strongly differing touching manners. Identifying them would allow us to improve the system's ability to interpret the touch instructions. In future works the number of subjects will be increased to assure statistical significance. Similar tests will be conducted also using real hardware as users may behave differently interacting with a real 3D multi-DOF system.

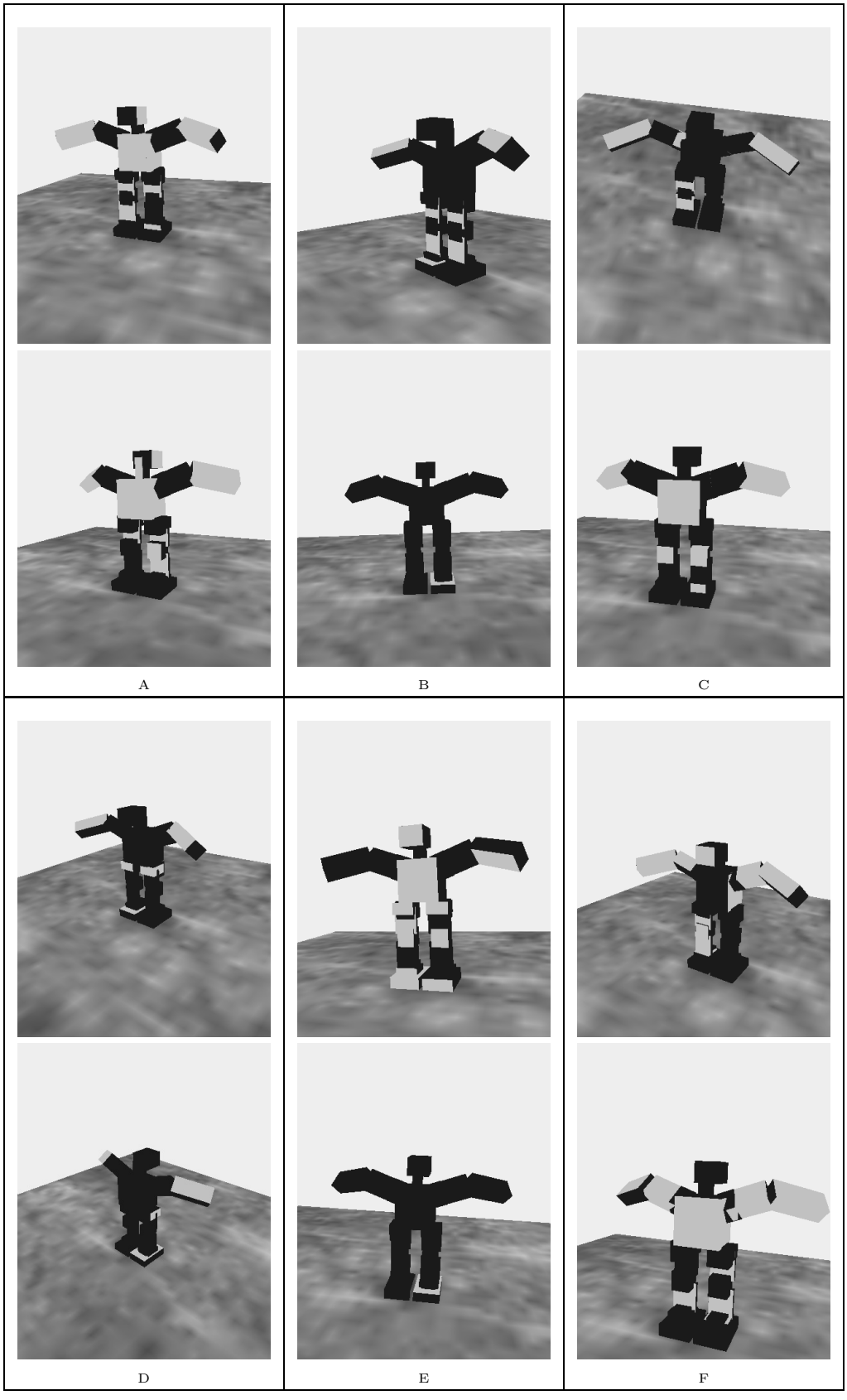


Fig. 15. For each subject the sensors touched are displayed coloring them white. The robot is shown both from the front and from the back.

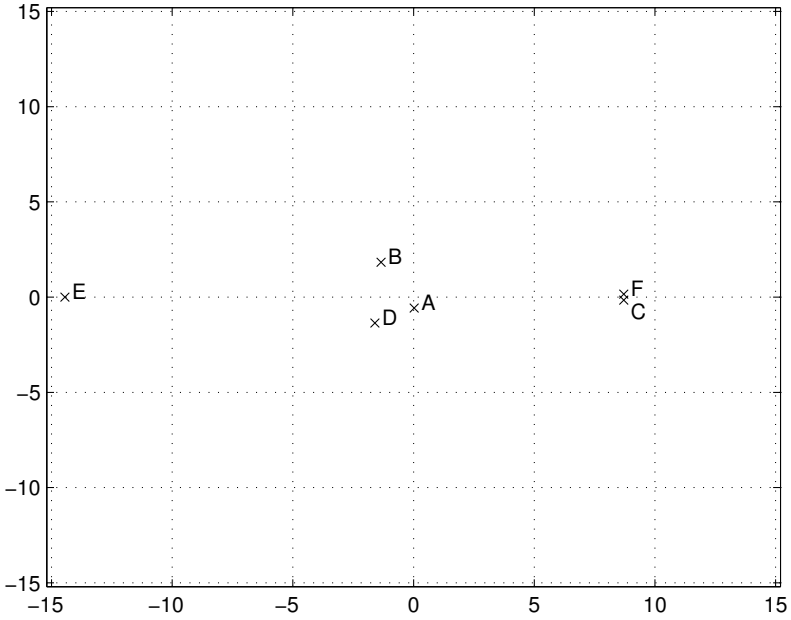


Fig. 16. Representation of the difference between the user's mapping obtained employing multidimensional scaling.

5 Discussion

As predicted, some parts of the mapping from touch instruction to joint modification are shared between the users, for instance for most of the users when the hand is pressed it is intuitive to bring the arm down. Nonetheless analyzing the touch data we can see that it would probably be more proper not to fix any part of the mapping. Instead, the interface should be able to understand the "level" of the instructions the user is providing. For instance in the experimental data we can observe three levels, from the lower to the higher:

- a nearly fixed mapping from a small set of sensors to the joints (subjects B and D) on which the context has little or no influence
- a mapping based on physical considerations (subject A); in this case, the context, for instance the position of the ground, becomes crucial
- a very high level representation of the motion, where for instance just the limb that should be moved is indicated by touching; at this level of abstraction a single touch corresponds to a motion primitive.

We can easily imagine that the implementation of these strategies is more complex the higher the level of the interaction is, and that the role of the context becomes more and more important. We can guess that to reach the level suggested by subject E we will need to introduce task knowledge to fulfill

such high expectations from the estimator. It is also possible to notice that our approach is able to be trained to work as a fixed protocol as in Fig. 4(a), but this protocol can be chosen runtime by the user (ex. users B and D) and does not require the programmer to design it beforehand.

One interesting point that emerged in the analysis of the developed motions is that all users realized the kick motion using the robot's right leg, while we gave no indications on which leg to use. Currently, we can provide two hypothesis:

(1) since all the students are right leg dominant, they taught the kicking motion thinking at what they usually do. It is in fact likely that persons transfer their own motion image to the learner when teaching a motion.
(2) when the robot is watched from the front, the leg that appears on the left is the robot's right leg; since all students are used to read from the left to the right, they could have picked up the "first" leg and developed the motion using that one.

Since in the development of the walking motion not all the users executed the first step with the same leg (four with the right leg and two with the left one) we believe the correct hypothesis is the first one, but further investigations, conducted including left handed subjects, are required. If this hypothesis is verified in future works we could employ the knowledge of the human operator's dominant hand or leg to improve the interpretation of the touching instructions. Furthermore if a person subconsciously teaches a motion as she/he does, it could be said that the instruction is done without mental load. Comparing the teaching by touching with the existing motion editor system on this point would be an interesting future work.

6 Conclusions and future work

In this work we presented a system for manually developing robot motions using touch instructions instead of a classical slider-based motion editor. We made this system completely adaptive, so that the user is not forced to employ a pre-determined protocol. More specifically the user touches the robot, sees how it moves and if the robot mistakes the interpretation of the touch instruction the human operator provides an example of the "correct" (the one that was desired) interpretation. In this work analyzing the touch examples of six different subjects we could g insights into the meaning of the touching instructions provided by the users. We could see that, as easily predictable, some basic motions (for instance closing the arms) are converted by most of the users into similar instructions and that usually users touch the limb they want to move. We also noticed that different users gave different levels of ab-

straction to the touch meanings, ranging from "when I press sensor s turn joint j" (subject B) to "when I touch your a knee, execute a step with that leg" (subject E).

Future work will need to address this finding. We plan to perform analysis of many more subjects, asking them to perform more complex motions, where most of the robot's parts must be used in many different contexts, and identify, for instance, whether there is a continuum of abstraction levels or, as could be suggested by these preliminary results, if it is possible to identify few well defined levels and associate the users to one of them. Given the experimental data analyzed in this work we strongly believe that if the touch-interpreter module will be able to identify the abstraction level adopted by the user then a great improvement in the speed of its adaptation to the user can be achieved. This would therefore allow the system to become very intuitive without requiring each user providing many examples. A further possible extension of this work is the introduction of more details on the touch instructions, for instance the pressure and the direction of the force applied to the sensors. We conclude stressing that the main purpose of this work is to study how touch can be employed for robot motion programming by unexperienced users without any emphasis on the maximization of the performances like the walking speed, for which specifically devised algorithms can surely obtain better results.

7 Acknowledgements

The authors would like to thank Marco Antonelli for his effort in setting up the experiment session and instructing the subjects, Tomoyuki Noda for the fruitful discussions on the data analysis and Dale Thomas for the comments on the draft of this paper. We also thank the anonymous reviewers for the useful observations provided.

References

[1] S. Nakaoka, A. Nakazawa, K. Yokoi, H.Hirukawa and K. Ikeuchi, Generating whole body motions for a biped humanoid robot from captured human dances, IEEE International Conference on Robotics and Automation, (ICRA 2003),2003, vol. 3, pp. 2905-3910.

[2] Y. Okumura, T. Tawara, K. Endo, T. Furuta and M. Shimizu, Realtime ZMP compensation for biped walking robot using adaptive inertia force control, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), 2003, vol. 1, pp. 335-339.

[3] T. Furuta, H. Yamato and K. Tomiyama, Biped Walking Using Multiple-Link Virtual Inverted Pendulum Models, Journal of Robotics and Mechatronics, 1999, vol.11, issue 4, pp. 304-309.

[4] S. Iida, S. Kato, K. Kuwayama, T. Kunitachi, M. Kanoh and H. Itoh, Humanoid robot control based on reinforcement learning, Proceedings of the 2004 International Symposium on Micro-Nanomechatronics and Human Science, 2004 and The Fourth Symposium Micro-Nanomechatronics for Information-Based Society, 2004, pp. 353-358.

[5] J. Stckler and Sven Behnke, On-line Optimization of Dynamic Walking Using Stochastic Policy Gradient Ascent, Proc. of Dynamic Walking 2006, pp. 29.

[6] F. Yamasaki, K. Endo, H. Kitano and M. Asada, Acquisition of humanoid walking motion using genetic algorithm-Considering characteristics of servo modules, Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA 2002) May 11-15, 2002, Washington, DC, USA.

[7] T. Wama, M. Higuchi,H. Sakamoto and R. Nakatsu, Realization of Tai-chi Motion Using a Humanoid Robot, Entertainment Computing, LNCS, 2004, vol. 3166, pp. 14-19.

[8] J. Baltes, S. McCann and J. Anderson, Humanoid Robots: Abarenbou and DaoDan, RoboCup 2006 - Humanoid League Team Description, 14-20 June 2006, Bremen, Germany.

[9] T. Takeda,Y. Hirata and K. Kosuge, Dance partner robot cooperative motion generation with adjustable length of dance step stride based on physical interaction , IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007 (IROS 2007), pp.3258-3263.

[10] R. Voyles and P. Khosla - Tactile gestures for human/robot interaction, Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems, Pittsburgh, PA, August 1995, vol. 3, pp. 7-13.

[11] G. Grunwald, G.Schreiber, A. Albu-Schaffer and G. Hirzinger, Touch: The Direct Type of Human Interaction with a Redundant Service Robot, Proceedings of the 10th IEEE International Workshop on Robot and Human Interactive Communication, 2001, pp. 347-352.

[12] T. Yoshikai, M. Hayashi, Y. Ishizaka, T. Sagisaka and M. Inaba, Behavior Integration for Whole-body Close Interactions by a Humanoid with Soft Sensor Flesh, IEEE-RAS International Conference on Humanoid Robots (Humanoids 2007), December 2007, Pittsburg, USA.

[13] D. Michie, D.J. Spiegelhalter and C.C. Taylor, Machine Learning, Neural and Statistical Classification, Ellis Horwood, 1994.

[14] V. Gupta, J. Bryan and J. Gowdy, A speaker-independent speech-recognition system based on linear prediction, IEEE Transactions on Acoustics, Speech, and Signal Processing, 1978, vol. 26, issue 1, pp. 27-33.

[15] J.L. Henry, A k-nearest neighbour method for managing the evolution of a learning base, Proceedings of the Fourth International Conference on Computational Intelligence and Multimedia Applications, 2001 (ICMA 2001), Yokusika City, Japan, pp. 357-361.

[16] A.E. Hoerl and R.W.I. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics, February 1970, vol. 12, issue 1, pp. 55-67.

[17] J.R. Quinlan, C4.5 Programs for Machine Learning, Morgan Kaufmann, 1993.

[18] K. Yamane and Y. Nakamura, Natural Motion Animation through Constraining and Deconstraining at Will, IEEE Transactions on Visualization and Computer Graphics, July-September 2003, vol. 9, issue 3, pp. 352-360.

[19] I. Borg and P.J.F. Groenen, Modern Multidimensional Scaling, 2nd edition. 2005, New York, Springer.

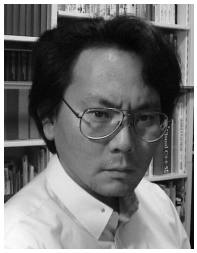
[20] T. Noda, T. Miyashita, H. Ishiguro, N. Hagita, Map Acquisition and Classification of Haptic Interaction using Cross Correlation between Distributed Tactile Sensors on the Whole Body Surface, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007 (IROS 2007), pp. 1099-1105.

8 Bibiographies

Fabio Dalla Libera received his Master degree in Computer Science from Padua University, Padua, Italy, in 2007. He is now a Ph.D. student at Padua University and he is studying touch as a means to communicate with humanoid robots. For the time being he visited Professor Ishiguro's Intelligent Robotics Laboratory at Osaka University three times.

Takashi Minato received his B.E. and M.E. degrees in mechanical engineering from Osaka University in 1996 and 1998, respectively. He was a researcher of CREST, JST since December 2001. He has been a Research Associate of the Department of Adaptive Machine Systems, Osaka University since September 2002. He is a member of The Robotics Society of Japan.

Ian Fasel is a postdoctoral researcher at the University of Texas at Austin, in the lab of Peter Stone. He received his Ph.D. in Cognitive Science from UC San Diego in 2006, in the Machine Perception Laboratory under Javier Movellan. His research focuses on applying machine learning and probability theory to understanding and solving real-world problems in machine perception and robotics. He is particularly interested in cognitive development, and his dissertation was on probabilistic generative models for learning real-time object detectors with little or no external supervision. This led to a new machine learning technique called Segmental Boltzmann Fields (SBFs).



Hiroshi Ishiguro (Member, IEEE) received the D.Eng. degree from Osaka University, Osaka, Japan, in 1991. In 1991, he started working as a Research Assistant of the Department of Electrical Engineering and Computer Science, Yamanashi University, Yamanashi, Japan. Then he moved to the Department of Systems Engineering, Osaka University, as a Research Assistant in 1992. In 1994, he was an Associate Professor, Department of Information Science, Kyoto University, Kyoto, Japan, and started research on distributed vision using omnidirectional cameras. From 1998 to 1999, he worked in the Department of Electrical and Computer Engineering, University of California, San Diego, as a Visiting Scholar. In 2000, he moved to the Department of Computer and Communication Sciences, University, Wakayama, Japan, as an Associate Professor and became a Professor in 2001. He is now a Professor of the Department of Adaptive Machine Systems, Osaka University, and a Group Leader at ATR Intelligent Robotics and Communication Laboratories, Kyoto. Since 1999, he has also been a Visiting Researcher at ATR Media Information Science Laboratories, Kyoto.

Enrico Pagello received the "Laurea" degree in Electronic Engineering from the Univ. of Padua, Padua, Italy, in 1972. From 1976 to 1983, he was a Research Associate at the Inst. of Biomedical Eng. of the Nat. Research Council of Italy, where he is now a part-time Senior Associated Researcher. Since 1983 he has been with the Faculty of Engineering, Univ. of Padua, where he is a Professor of Computer Science. During 1977 and 1978, he was a Visiting Scholar at the Lab. of A.I. of Stanford Univ., CA. Since 1994, he has regularly visited the Dept of Precision Engineering, Univ. of Tokyo, Japan, where he has been a JSPS Fellow. On 2005, and 2008 he has been an Invited Professor at Keio Univ., Japan. His current research interests include the application of A.I. to Robotics with particular regards to the Multi-robot Systems domain. He was a General Chair of the Sixth IAS Conference in July 2000, and a General Chair of RoboCup-2003. He has been a member of the Ed. Board of the IEEE/Trans. on Rob. and Aut., and he is currently a member of the Ed. Board of RAS Int. Journal. He is President of the Intelligent Autonomous Systems International Society.

Emanuele Menegatti graduated in Physics in 1998 at the University of Padua, Italy. From June 2002 to September 2002, he was Visiting Researcher at the Intelligent Robotics Laboratory of Prof. Hiroshi Ishiguro in the University of Wakayama. He received the Ph.D. in Informatics and Industrial Electronics in 2003 at the Dept. of Information Engineering of the University of Padua. From May 2003 to Dec.2004, he had a Post-Doc Position at the Dept. of Information Engineering of the University of Padua. In January 2004, he was a visiting researcher at the BORG Lab (Georgia Institute of Technology , Atlanta U.S.A.) working with prof. Frank Dellaert. In February-March 2004, he was visiting researcher at the Intelligent Robotics Laboratory of Prof. Hiroshi Ishiguro at Osaka University. In 2006-2007 he became adjunct Professor of Robotics at Padua University.