Interactive Demonstration of Pointing Gestures for Virtual Trainers

Yazhou Huang and Marcelo Kallmann

University of California, Merced
{yhuang6,mkallmann}@ucmerced.edu

Abstract. While interactive virtual humans are becoming widely used in education, training and delivery of instructions, building the animations required for such interactive characters in a given scenario remains a complex and time consuming work. One of the key problems is that most of the systems controlling virtual humans are mainly based on pre-defined animations which have to be re-built by skilled animators specifically for each scenario. In order to improve this situation this paper proposes a framework based on the direct demonstration of motions via a simplified and easy to wear set of motion capture sensors. The proposed system integrates motion segmentation, clustering and interactive motion blending in order to enable a seamless interface for programming motions by demonstration.

Keywords: virtual humans, motion capture, interactive demonstration.

1 Introduction

The motivation of this work is to develop new algorithms and techniques to achieve effective virtual assistants that can interact, learn, train, assist and help people to execute tasks, with the potential to provide assistance, deliver training and education, etc. In general, there are two main approaches for synthesizing gestures for interactive virtual humans: 1) the first approach produces high quality gestures with pre-defined keyframe animation either hand-crafted or by motion capture (*mocap*) [1] [2]. These systems are then able to reproduce the available motions according to the given goals, context and speech. The final result is usually very realistic; however motions cannot be easily reused in new scenarios. 2) The main alternative is to algorithmically synthesize the needed motions, such as in the VHP [3] and MAX [4] systems, however achieving less realistic results.

This paper proposes a framework that attempts to capture the advantages of both approaches. We employ parameterized motion blending techniques [5] in order to achieve realistic results which can be adapted and parameterized, and we also take into account on-line motion demonstrations for the interactive modeling of new gesture motions. To test our framework we present in this paper our prototype system for demonstration of pointing gestures. Pointing was chosen because of its importance: besides being one of the earliest forms of communication used by humans it serves several purposes: it guides the attention of the listeners or viewers, assigns meanings and labels to locations and objects, etc [6]. Our framework can be easily adapted to

J.A. Jacko (Ed.): Human-Computer Interaction, Part II, HCII 2009, LNCS 5611, pp. 178–187, 2009.

other demonstrative gestures for instance for describing characteristics (weight, size, etc), for showing how actions should be performed, or for expressing qualitative characteristics needed to perform actions. The main characteristic explored in this work is the fact that the end-effector trajectory and/or final location are the most important aspects in demonstrative gestures.

The proposed framework also presents a new gesture modeling paradigm based on interactive motion demonstrations. Our framework can be seen as an imitation-based approach [7] [8] and is especially relevant for controlling and programming tasks for humanoid agents [9] [10] [11] [12] [13]. It also represents a natural approach to human-computer interaction by analogy to the way humans naturally interact with each other. This paper describes our first results towards achieving an intuitive interface based on low cost wearable motion sensors for programming and customizing demonstrative gestures.

2 System Overview

Our framework models each demonstrative gesture with a cluster of example gestures of the same type but with variations for the different locations being demonstrated. In the pointing gestures modeled in this work the different locations are the locations being pointed at. For example a gesture cluster for a certain way of pointing consists of several examples of similar pointing gestures but each pointing to a different location. Gestures in a cluster are time-aligned in order to allow correct blendings and Inverse Kinematics corrections to be performed for computing a final gesture motion able to achieve given specific target pointing locations. This process is referred here as *Inverse Blending*.

We have also developed for our system a *gesture vest*, for achieving a mocap-based human-computer interface for on-line motion demonstrations. Example motions can be demonstrated interactively and converted to new examples to be added in the database, in order to enable the interactive customization of gestures. This interactive interface allows seamless user interactions for programming gestures. Our overall system is depicted in Figure 1.

The system maintains a gesture database on-the-fly by storing selected demonstrations from the gesture vest. The database can also be initialized with basic gestures. The virtual character can then reuse the demonstrated motions in order to point to any

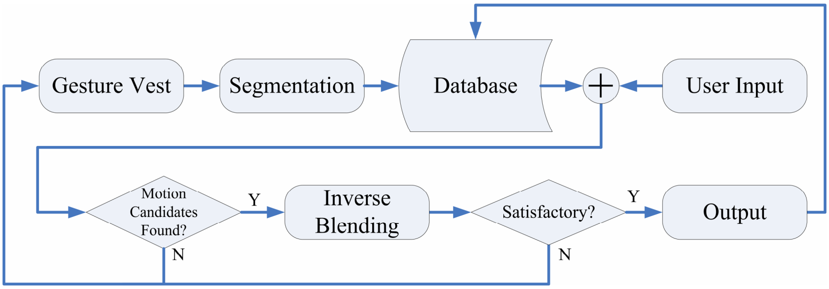

Fig. 1. System overview

desired location and the resulted motion will exhibit the same characteristics of the demonstrated examples. If suitable candidate example motions are not available in the database, or if the user is not satisfied with the aspect of the output motion, new demonstrations can be interactively given by the user via the gesture vest. The system can therefore learn user-specific (pointing) tasks in different scenarios through on-line demonstrations, and at the same time preserve the characteristics of the demonstrated motions as much as possible.

The next section describes our inverse blending algorithm for pointing gestures, and section 4 presents our *gesture vest* interface built specifically for this work.

3 Inverse Blending of Pointing Gestures

After a collection of similar pointing motion segments is acquired, they will form a gesture cluster G stored in the database. Let m_i be one motion segment in G. Each motion is represented as a sequence of frames and each frame is represented with the position of the root joint and the rotations of each other joint in the character skeleton. The system then computes the stroke time t_i of m_i and determines the position p_i and orientation q_i of the character's hand in global coordinates at time t_i by Forward Kinematics, with the orientation q_i being represented in quaternion format. Each segment m_i is then time-warped such that all motions have their stroke times and durations the same. The stroke time is in particular addressed here due its main importance for parameterizing demonstrative gestures, and in particular pointing gestures.

A motion blending mechanism is then employed for computing new gesture motions for achieving new targets at the stroke point. We use a traditional blending scheme of the type $m_b(w_1, \ldots, w_n) = \sum B_i(w_i)m_i$ to interpolate examples from G, where B_i are normalized kernel functions whose values depend on the weight coefficients w_i. The challenge is to determine the weight values which will lead to stroke locations as close as possible to the (p,q) target. A typical approach is to adjust the kernel functions B_i in order to best address the parameterization and spatial constraints [14] [5]. However pre-computation of the weights is required making this approach less suitable for interactive systems and in any case it is not possible to guarantee precise placements. Therefore, in order to achieve hand placements with precise control, we also employ Inverse Kinematics (IK) corrections.

We first determine the blending weights according to the distances between the wrist locations p_i and the center axis of a cone delimiting feasible pointings to the target location p. Figure 2 illustrates such a cone in yellow wire frame. Example motions which are far away from the cone will get a zero weight and will not contribute to the blending result. Once the weights are computed, a first blended motion m_b is then determined.

Motion m_b will produce a pointing motion pointing to a location nearby the target but most likely it will not be precisely pointing to the target. Correction iterations are then performed. Different iterative correction methods are being evaluated in our prototype system and we are currently using a simple correction mechanism as follows. Let (p_b,q_b) be the wrist joint position and orientation at the stroke time of motion m_b. We then adjust the hand position p_b in order to achieve the fingertip precisely pinpointing the goal, obtaining a new position p'_b. Next, we use the position and orientation (p_b',q_b) as a IK goal for producing a new arm posture precisely pointing the target. We use a fast analytical IK solver [15] which allows specifying the swivel angle of the final arm posture to be the same as the swivel angle observed in the stroke posture of m_b. The solved IK posture will therefore precisely point the target and at the same time will be very similar to the blended posture due to the enforcement of the original q_b and the swivel angle. The blending weights are then adjusted during a few iterations in order to get new a blended motion with a stroke posture closer to the solved IK posture. When finished, a final correction with IK is performed in all the frames of the motion to compute the final precisely correct pointing motion. First the stroke posture is corrected by the posture computed by the IK and then decreasing correction values are propagated along all the previous and subsequent frames of the motion in order to achieve the final pointing motion. See Figure 3 for some obtained examples. In this way, blending is guiding IK to be more "human-like", while IK is guiding blending to get closer to the precise pointing. As the algorithm iterates, the blended posture approaches the IK solution, given that enough example motions are available.

Note that a suitable solution may not be found in several cases: when positions are not reachable within the desirable error, when there are no sufficient example gestures in the database, etc. The user will then need to interactively demonstrate new gestures

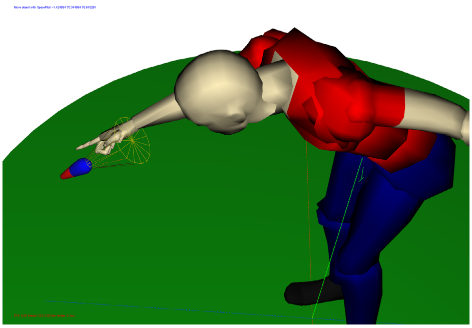

Fig. 2. A cone is used to delimit the region of feasible pointings. The shown posture represents one example pointing posture that will receive a high blending weight as it is close to the cone.

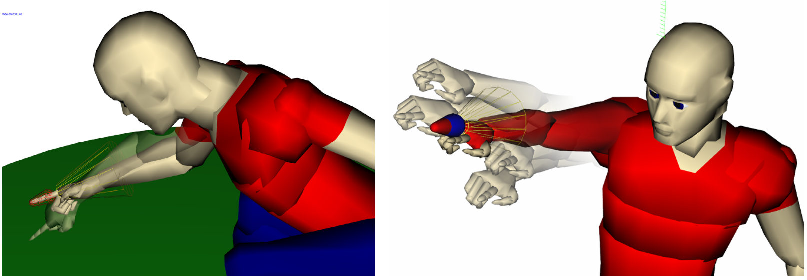

Fig. 3. Left: comparison between a posture obtained with blending (in transparence) and the same posture after IK correction. Right: precise pointings are achieved after the IK correction.

to be included in the cluster until an acceptable result is obtained. This overall combined approach will result in an effective system for gesture modeling and synthesis.

4 Wearable Motion Capture Vest

We have built a wearable motion capture vest for achieving a suitable human-computer interface for our system. The interface uses five InnaLabs AHRS sensors [16] to capture the orientation changes of performer's spine, head and a full arm. In addition, a 5DT data glove is used to capture hand shapes, and a Nintendo Wii remote controller is used for providing basic instructions (record, play, delete, etc) during demonstration of new example gesture motions. As illustrated in Figure 4, the sensors are attached on a detachable sleeve, in the form of an easy-to-wear gesture vest. The vest is connected to a computer via wired USB-RS 485 converters, optionally connected to a wireless USB hub.

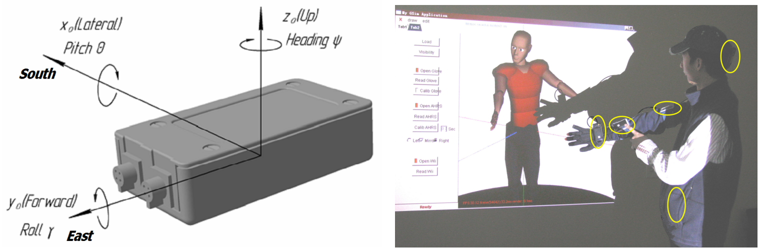

Fig. 4. Left: miniAHRS m2 sensor used in our system. Right: Our mocap vest, with a data glove and 5 miniAHRS sensors placed on spine, head, right upper-arm, right forearm and right hand. The locations are highlighted with yellow circles.

Each sensor measures its orientation in global coordinates based on triaxial gyro, accelerometer and magnetometer. Each measured rotation $q_{reading}$ is represented in quaternion form in respect to a reference frame with X axis pointing South, Y axis pointing East, and Z axis pointing upwards, as shown in Figure 4-left. The maximum update rate is 120 Hz in quaternion mode. These commercially-available sensors provide good results but different sensors and technologies can also be used [17] [18].

Calibration: The sensors use magnetometers to acquire absolute orientation based on the earth's magnetic field. We perform an initial orientation calibration by standing in a T-pose facing North to match the zero-rotation of the sensor with the negative X pointing North, as shown in Figure 5(2). Since the sensors can be in slightly different positions every time the performer wears the vest, a calibration is done before each new capture session. The calibration will record a reference rotation q_{calib} for each sensor.

Skeleton Mapping: In order to map the sensed rotations to our skeleton representing the character to be animated we have to transform the rotations to suitable frames in local coordinates. First we transform $q_{reading}$ to a Z-up coordinate system, producing a rotation compatible with the sensor's native coordinate system (Z-up, see

Figure 4-left). In this way, for example, a rotation $q_{reading}$ along South axis will produce the corresponding quaternion rotation $q_{Z\text{-up}}$ along X axis after applying the calibration quaternion. Rotation $q_{Z\text{-up}}$ is obtained with:

$$q_{Z-up} = q_{reading} \cdot q_{calib}^{-1} \cdot$$

The virtual character we're using follows a Y-up coordinate system as shown in Figure 5(1). In order to drive our character, we use pre-rotations and post-rotations that transform the rotation from miniAHRS Z-up coordinate to our Y-up coordinate:

$$q_{Y-up} = q_{preRot} \cdot q_{Z-up} \cdot q_{postRot},$$

where q_{preRot} is 120º rotation along axis (-1, 1, 1) in quaternion form that rotates the character from the Y-up frame to the Z-up frame (see Figure 5). Rotation $q_{postRot}$ produces the opposite rotation which is -120º rotation along same axis (-1, 1, 1).

Before applying $q_{Y\text{-up}}$ to character's joints, it has to be transformed to local coordinates. The joint structure of our character is shown on the right side of Figure 5. Every time the character's skeleton is updated with the sensed values, the rotation of each joint in global coordinates is computed by Forward Kinematics, i. e., by recursively multiplying the local rotations of all parent joints, starting at the root joint. Global rotations can then be transformed to local coordinates with:

$$q_{local} = q_{parent}^{-1} \cdot q_{Y-up},$$

where q_{local} is the final rotation to be sent to the joint being mapped and q_{parent} is it's parent rotation in global coordinates.

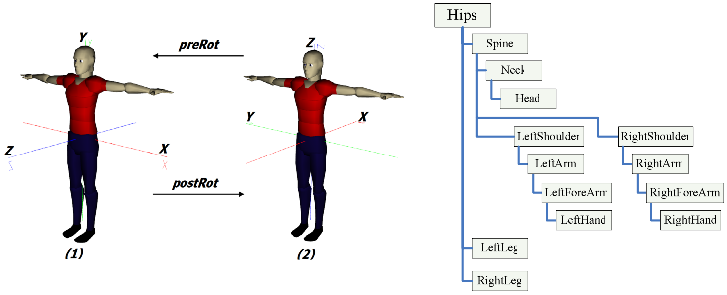

Fig. 5. Left: the different coordinate systems. Right: character structure.

Quality Evaluation: We have compared the results achieved by our mocap vest against results recorded with the Vicon optical-based motion capture system, which captures positions in global coordinates with precision generally better than 4 millimeters. We have captured motions simultaneously with the two systems, wearing our mocap vest underneath the Vicon suit with the reflective markers placed following the suggestions in Vicon's manual. The sensor readings were acquired at a rate of 30 frames per second and were directly mapped to our skeleton. We used Vicon Blade software to fit the same skeleton used by our system to the optical markers, and we initialized the reconstruction with identical initial poses. The global reference frame of Vicon's reconstruction was also transformed to be the same as in our system. Since our mocap vest does not capture lower body motions, the performer maintained the waist at the initial position. Although neither reconstruction is perfect, Vicon matches the original motions visibly better, thus we treat it as ground truth in our comparison analysis. Evaluation is done by comparing the rotations for a set of corresponding joints on both skeletons.

The left side of Figure 6 shows one frame of a 85-seconds range-of-motion capture with the right arm covering most of its reachable space. Our reconstruction is shown on the left and Vicon's on the right. The left side of Figure 7 shows the trajectories of the right hand joint along a portion of the captured motion, with our reconstruction marked with red, and Vicon's marked with blue. The plots are different views from the same motion frame. Vicon's reconstruction matches the original motion slightly better as it captures more joints than our vest, though our vest is able to better reconstruct subtle motions such as slight wrist flicks and more precise head orientation as shown in Figure 6-left (note that our current vest does not capture the left arm, which is kept in its initial configuration).

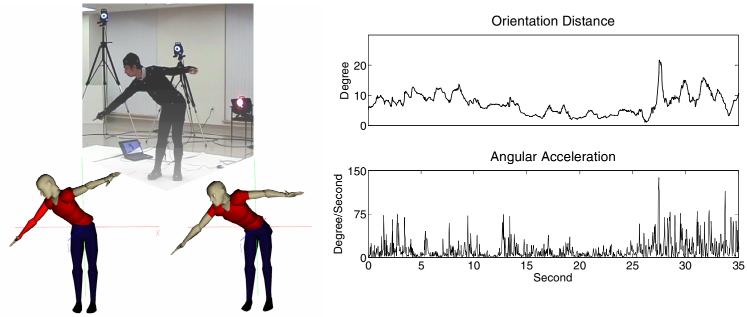

Fig. 6. Comparison to the Vicon system. Left: one captured posture, our reconstruction (left) and Vicon's reconstruction (right). Right: right shoulder joint values over 35 seconds of motion. The top plot shows the distance between the joint values of the two reconstructions. The bottom plot shows that the error increases as the changes in angular acceleration increases.

Figure 8 shows a graph of a set 3-DOF joints (neck, right shoulder, spine and right hand) over 50 seconds of motion. The top three plots visualize the joint Euler angles (along X, Y and Z axis respectively) as reconstructed by us (dash red) and Vicon (solid black), while the bottom plot shows the orientation difference between our reconstruction and that of the Vicon system. The orientation difference is introduced by computing the quaternion distance [19] of a pair of corresponding joints (between us and Vicon) defined as:

$$\theta = \arccos(q_1 \cdot q_2)\,.$$

As can be seen from the plots, the orientation differences are generally below 10º~20º. Joint rotations from our mocap system could mostly follow that of Vicon, except for some motion periods where the difference becomes slightly larger, in particular when angular acceleration of the motion greatly increases or decreases (see Figure 6-right). This is due to the characteristics of the miniAHRS sensor as the orientation sensing relies more on gyroscope and accelerometer to achieve fast responses when angular acceleration increases, and depends more on magnetometer when motion becomes slow to then give fairly accurate readings.

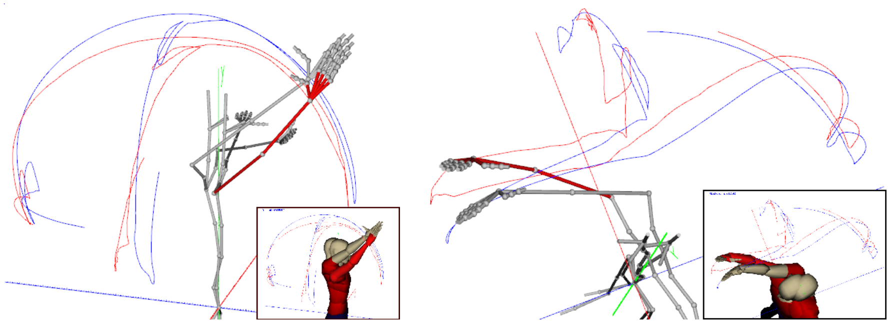

Fig. 7. Trajectories of the right hand joint with our reconstruction (marked in red) and Vicon's (marked in blue). The two plots are different views of same motion frame.

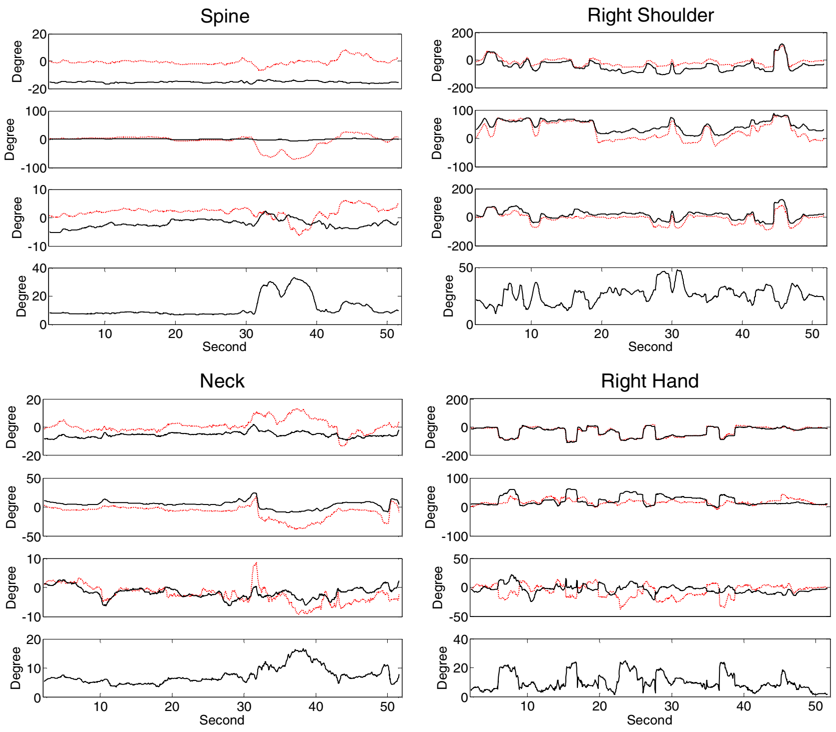

Fig. 8. Euler angle plots for a set of 3-DOF joints (neck, right shoulder, spine and right hand) over 50 seconds of motion. The top three plots compare joint Euler angles as reconstructed by us (dash red) and Vicon (solid black). The bottom-most plot shows the orientation difference between the two reconstructions.

5 Discussion

Our mocap gesture vest system is specifically built for on-line motion acquisition with fairly good results. Our single-arm configuration for capturing single arm gestures with 5 sensors provides enough flexibility for on-line motion demonstration of demonstrative gestures, where example motions will be demonstrated interactively and converted to new gesture examples to populate clusters in the database. After a few demonstrations are performed, object targets to point to can be given in any scenario. Whenever a new target in the scene is selected, the appropriate cluster is identified and new motions are synthesized by the inverse blending algorithm. The mocap system is robust, easy to wear, intuitive to use, and also relatively cheap in cost.

The motion stream produced by the sensors is continuously monitored during performances and annotated by simple commands given by the user through the Wii controller such as: create new example motion, refine existing motion, replace current candidate motion and delete from database. These simple commands achieve a seamless and intuitive human-computer interface for informing how to interactively segment, classify and cluster the input motion into meaningful gesture segments.

The system automatically segments and organizes the collected gestures into gesture database to be used by the Inverse Blending. For our current experiments with pointing gestures, segmentation is implemented in a straightforward way simply by observing the zero crossing of the velocity vector of the hand. This works well as pointings often have a unique maximal stroke point before returning to a rest posture. Sliding-window filters are applied to eliminate false detections so that the segmentation is immune to jitters and noises in the captured motions. Parameters of the filters are fine tuned by hand. Auto time-alignment of the segments is applied before storing the new segments in the database.

We have also implemented the feature of mirroring of gestures from the captured arm to the other arm. This enables symmetric full-torso gestures to be generated as in several two-handed gestures used in speech, conversations and in sign language [20]. The mirroring is done by negating both the x and w components of the original quaternion.

6 Conclusion and Future Work

We have presented our first results in building a system which allows users to intuitively model motions that virtual characters can use to demonstrate objects and procedures. Our system allows non-experts to model virtual trainers interactively for generic training and educational applications.

As future work, we are exploring several improvements and extensions, for instance: an improved inverse blending solver and the integration of locomotion in order to enable the character to also move around the objects being demonstrated.

Acknowledgements. This work was partially supported by NSF Award CNS-0723281 and a CITRIS seed project.

References

1. Gebhard, P., Michael Kipp, M.K.T.R.: What are they going to talk about? towards life-like characters that reflect on interactions with users. In: Proc. of the 1st International Conf. on Tech. for Interactive Digital Storytelling and Entertainment, TIDSE 2003 (2003)
2. Thiebaux, M., Marshall, A., Marsella, S., Kallmann, M.: Smartbody: Behavior realization for embodied conversational agents. In: Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) (2008)
3. Noma, T., Zhao, L., Badler, N.I.: Design of a virtual human presenter. IEEE Computer Graphics and Applications 20(4), 79–85 (2000)
4. Kopp, S., Wachsmuth, I.: Model-based animation of co-verbal gesture. In: Proceedings of Computer Animation 2002, pp. 252–257 (2002)
5. Rose, C., Bodenheimer, B., Cohen, M.F.: Verbs and adverbs: Multidimensional motion interpolation. IEEE Computer Graphics and Applications 18, 32–40 (1998)
6. Kita, S.: Pointing: A foundational building block of human communication. In: Kita, S. (ed.) Pointing, Where Language, Culture, and Cognition Meet. Lawrence Erlb. Ass., NJ (2003)
7. Breazeal, C., Buchsbaum, D., Gray, J., Gatenby, D., Blumberg, D.: Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots. Artificial Life 11, 1–2 (2005)
8. Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. The Neuroscience of Social Interaction 1431, 199–218 (2003)
9. Suleiman, W., Yoshida, E., Kanehiro, F., Laumond, J.P., Monin, A.: On human motion imitation by humanoid robot. In: 2008 IEEE International Conference on Robotics and Automation (ICRA), pp. 2697–2704 (2008)
10. Olenderski, A., Nicolescu, M., Louis, S.: Robot learning by demonstration using forward models of schema-based behaviors. In: Proceedings of International Conference on Informatics in Control, Automation and Robotics, Barcelona, Spain, pp. 14–17 (2005)
11. Nicolescu, M.N., Mataric, M.J.: Natural methods for robots task learning: Instructive demonstration, generalization and practice. In: Proc. of the 2nd Internat. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS), Melbourne, Australia (2003)
12. Ramesh, A., Mataric, M.J.: Learning movement sequences from demonstration. In: Proceedings of the International Conference on Development and Learning (ICDL), pp. 302–306. MIT, Cambridge (2002)
13. Billard, A., Mataric, M.J.: Learning human arm movements by imitation: Evaluation of a biologically inspired connectionist architecture. Robotics and Autonomous Systems 37(2-3), 145–160 (2001)
14. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. In: SIGGRAPH 2005: ACM SIGGRAPH 2005 Papers, pp. 1062–1070. ACM, New York (2005)
15. Kallmann, M.: Analytical Inverse Kinematics with Body Posture Control. Computer Animation and Virtual Worlds (CAVW) 19(2), 79–91 (2008)
16. Innalabs miniAHRS m2 user's manual (2008)
17. Vlasic, D., Adelsberger, R., Vannucci, G., Barnwell, J., Gross, M., Matusik, W., Popovic, J.: Practical motion capture in everyday surroundings. In: SIGGRAPH 2007: ACM SIGGRAPH 2007 papers, p. 35. ACM, New York (2007)
18. Slyper, R., Hodgins, J.: Action capture with accelerometers. In: 2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation (2008)
19. Hanson, A.J.: Visualizing Quaternions. The Morgan Kaufmann Series in Interactive 3D Technology, pp. 248–249. Morgan Kaufmann, San Francisco (2005)
20. Xiong, Y., Quek, F., McNeill, D.: Hand gesture symmetric behavior detection and analysis in natural conversation. In: IEEE Internat. Conf. on Multimodal Interfaces, p. 179 (2002)