A Model for Time-Synchronized Sensing and Motion to Support Human-Robot Fluency

Tariq Iqbal, Michael J. Gonzales, and Laurel D. Riek
University of Notre Dame
Notre Dame, IN, USA
{tiqbal,mgonza14,lriek}@nd.edu

ABSTRACT
The timing of actions in human-robot interaction (HRI) is important for achieving fluency. In order to accurately sense what people are doing and respond appropriately, it is important that the data captured by a robot and any external sensors are time-synchronized. In this paper, we propose a timing model to handle real-time data capture and processing to ensure all actions made by our robot occur in a time-synchronous manner. We implemented our model in ROS, and ran an HRI study to validate it. In pairs of two, participants [n = 70] interacted with each other and a robot during a dance-off, and we measured their expressiveness from multiple Kinect sensors. Our results suggest that our model is successful and effective in time-synchronizing all external sensor data and robot actions for our system. In addition, our model is not limited to specific types of cameras or sensors, and grants flexibility in capturing various types of data from different sources to paint a more accurate picture of the environment.

Keywords
Human-robot interaction, social robotics, timing models

1. INTRODUCTION
For humans and robots to work together effectively, it is important that both robots understand people and people understand robots. To address this, the concepts of fluency and legibility have become important topics in robotics research. Fluency means that robots convey their motion in a way that is legible (understandable), enabling humans to predict a robot's actions, intentions, and goals, and interact fluently with it [20, 13, 24, 8, 11].

One of the most critical elements for achieving fluency is timing. Timing is used as a measure of efficiency for communicating intent and instructions to a robot [10]. Applications such as situated dialogue, music and entertainment, performance characteristics, and expressive gestures, all of which use timing as a measure for efficiency of interaction, influence how users perceive and respond to robots. Similar to the concept of feedback in human computer interaction (HCI) [27, 34], if robots do not act, react, or provide feedback in accordance with human expectations, the interaction has failed.

One crucial problem related to timing of robotic systems focuses on temporally synchronizing actions of the system so that appropriate actions and responses occur at the right time. If a human makes a cue that will alter the actions of a robot, all parts of that robot, including its sensors, movement, and activities, should appropriately follow the same timing scheme so that it may perform the next set of actions in a synchronized manner. If this is not the case, a robot's actions may occur at the wrong time, data captured using various sensors may represent mismatched timing patterns, and the robot's interaction with a human might be affected.

Synchronizing the actions of a robotic system is particularly important considering a robot's actions may act as cues for humans. For example, a sociable robotic aide for medical adherence may make some cue to a human so that they take their medicine at a specific time [16], or an entertainment robot may play music and move so that a participant will dance. Similarly, the actions of people may affect when a robot stops performing one action, and begins performing another.

Timing of actions play a significant role in various social activities performed in human environments, including turn taking, engagement in group tasks, among others [36, 28, 9, 31, 19, 32, 21, 30, 29, 33, 18]. Timing of actions may influence social and task level strategies [9]. Thomaz et al. [36] proposed a turn taking model and developed an autonomous floor relinquishing model for a robot during a human-robot interaction scenario based on their model. To predict the engagement of a human-robot pair during an interaction, Rich et al. [28] developed a computational model to measure engagement. Timing is also important in other fields of HRI, such as assembly manufacturing [37].

To act accordingly and in a timely fashion with humans, it is important for robotic systems to be able to sense human activities on time through the data received from its attached sensors. Generally a single sensor can only provide limited information about the environment to a robot, whereas multiple sensors can provide information from dif-

Figure 1: DJ Rob

Figure 2: Two participants performing in a dance-off

ferent viewpoints [38]. As a result, some research has focused on incorporating data from multiple sensors together in order to improve the perception of the environment of a robot [22, 17].

One challenge, however, is that incorporating this data in a time-synchronized manner may not always be easy. In the case that these sensors are attached directly to the robot, it is likely that such data will be time-synchronous. However, in the case of a decentralized system of sensors, sensor data is fused locally within a set of local systems, rather than by a central unit, i.e, in a robot [38]. The coordination of sensor data is achieved by communication between the local systems.

Due to the fact that integration of time-synchronous data may directly affect robotic actions, the issue of synchronous data plays a crucial role in the context of HRI, among other fields. The problem of time synchronized data is approached in a number of fields, including: distributed systems [6], wireless sensor networks [14], robotics systems [12], and swarm robotics systems [23]. Moreover, in HRI, socially-interactive robots need to operate at a rate at which humans interact, in real-time, in addition to managing synchrony between sensors and data [15]. This makes the issue of time-synchronized data capture and processing much more difficult to manage in HRI.

In order to address these issues, we designed a timing model for a robotic system which can handle time-synchronous data in real-time. Our timing model is designed to handle real-time data capture and processing, and to ensure the occurrence of all the actions across our system in a time-synchronous fashion. In addition, our model is not limited to any specific type of sensors, but rather a wide variety of sensors, including video sensors, audio sensors, accelerometers, among others.

We evaluated our model using an entertainment robot, called a DJ Rob, which interacted with dancing participants and measured their expressiveness in real time (See Figures 1 and 2). This measure of expressiveness, along with timing, was used to direct our robot to perform specific actions and to provide user feedback. Our results suggest that our model was successful and effective in synchronizing all sensor data, actions, and activities in our system.

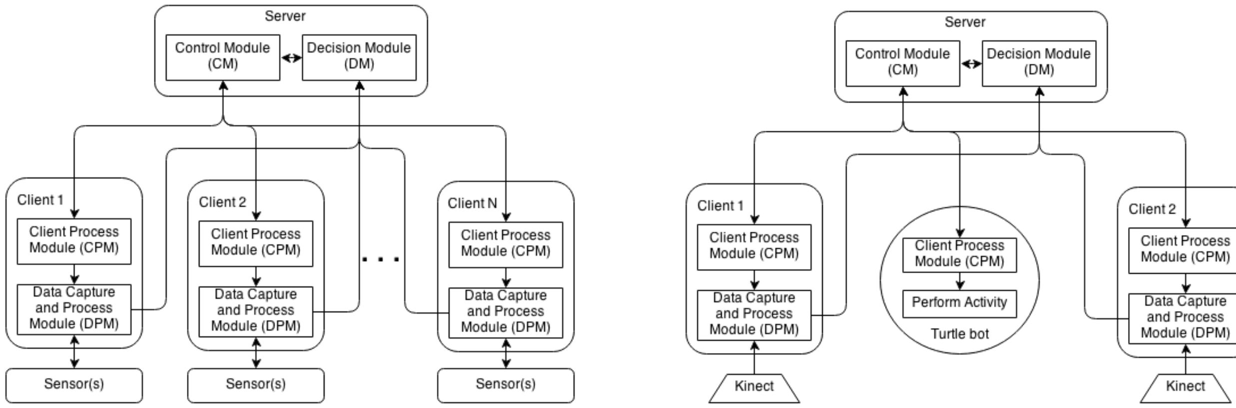
2. SYSTEM MODEL

Our timing model is designed using a client-server architecture (See Figure 3a). The server node was responsible for communicating with all client nodes. Each client node was an independent system, and may or may not be connected with its own sensors. A client node may respond to server messages, capture data using its own sensors, process the captured data, and return the processed result to the server. The server node controlled the duration of the data capture and processing of each client system.

We implemented the model using the Robot Operating System (ROS) publisher-subscriber architecture [3]. ROS is an open source platform which provides libraries and tools to develop robotics applications. All nodes in our system run the Electric version of ROS.

In our model, different parts of the model communicate with each other via ROS topics [4]. An ROS topic is a standard unidirectional message passing protocol and uses a publisher/subscriber based model. A publisher can publish messages via an ROS topic, and any number of subscribers can subscribe to that topic. Different parts of the model can communicate with each other by publishing or subscribing to a specific topic.

The server node of our system consists of two modules: the control module (CM), and the decision module (DM). The main task of the CM is to communicate and control the client nodes. This module also communicates with other modules of the server, and is responsible for synchronous data capture and processing in all client nodes. After capturing and processing the sensor data, all client nodes communicate with the DM of the server, and sends the processed result to the DM. The decision module can make decisions based on the processed data received from the clients. The decision module may also communicate with robots, controlling their movement.



(a) Our proposed system model

(b) Implementation of our proposed model

Figure 3: The architecture of our proposed timing model, shown in (a), and our implemented system, shown in (b). Using our model, each client may either connect to various sensors or robotic agents to perform specific actions.

Each client node consists of a client process module (CPM) and a data capturing and process module (DPM). The CM of the server node communicates with the CPM module of the client nodes. Based on the message received from the CM, the client process module is responsible for synchronizing the local machine time to a global time server. After synchronizing the time, the CPM is responsible for sending an acknowledgement (ACK) message to let the CM know that the time-synchronization is complete.

These processes also subscribe to the ROS topic broadcasted by the CM. If the CM sends a message to start capturing and processing sensor data, then the CPM communicates with the DPM to make sure that the data capture and processing starts and ends on time. After getting specific message from its own CPM, the DPM starts capturing and processing data for a specific period of time. Following data capture, the DPM publishes the processed data. The DM of the server node and robot nodes subscribe to this broadcasted topic from the DPM. The overall system model is presented in Figure 3a. The arrow represents the direction of communication between modules.

One of the main challenges in synchronized data acquisition is capturing and processing data in a time-synchronized manner. All events that occur at the same time should have the same time stamp in the captured data among all clients. If the events do not represent captured data from each sensor for the same period of time, then it is difficult to make a proper global decision based on the processed data. For example, if event data captured from different sensors are reported from different periods of time, then the decision taken based on the time stamps represents an erroneous result.

In addition, in the case of sensors connected via wired/wireless connections, we encounter the issue of network latency. Depending on the proximity and network congestion, different nodes will receive the broadcasted message at different times.

To overcome these difficulties, we implemented a simple message passing protocol using ROS topics, presented in Figure 4. In our protocol, the CM broadcasts a 'sync_time' message. All CPMs of the client nodes and robot nodes subscribe to this topic. Depending on network traffic, different client nodes receive this message at different times.

When a CPM of a client node receives this message, it synchronizes its local time to a global time server using the Network Time Protocol (NTP) [26]. At the same time, the server also synchronizes its time to the global server. Following time synchronization, all CPMs broadcasted an acknowledgement message via an ROS topic. The CM in the server node subscribes to this message and listens for an ACK message from all client nodes to indicate synchronized time in all nodes.

When all client nodes are time-synchronized, the control module of the server broadcasts a 'capture_data' message. This topic consists of three fields. The first field is the 'CR_T' field, which represents the current time of the server system. The second field is the 'STR_AFT' field, which represents a time value in seconds. This field tells the CPM of the client systems to start data capturing and processing after STR_AFT seconds from the CR_T time. Finally, the last field 'DUR' represents the duration for data processing in seconds.

Depending on network traffic, different nodes may receive the 'capture_data' message in different times. The client process module of a client node tells its DPM to start capturing and processing data. All DPMs will wait for a global time to occur, calculated by adding STR_AFT with the CR_T value. As all client nodes are synced with the same global time, all clients begin capturing sensor data at the same time. STR_AFT value should be chosen carefully based on the network congestion, so that all the clients receive the 'capture_data' message within STR_AFT seconds after broadcasted from the server. After the 'DUR' period is over, the DCP module broadcasts the processed result via an ROS topic.

After receiving the processed results from the clients, the

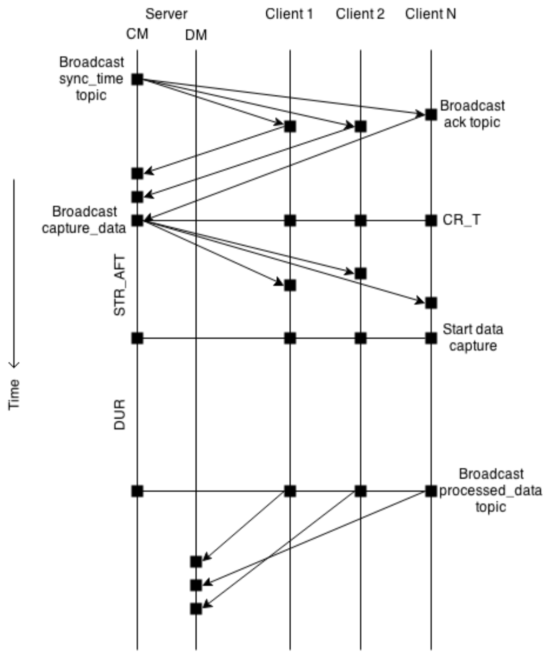


Figure 4: Message passing protocol of our system

DM can make a global decision. Depending on the decision, the DM may instruct the robot connected to the server to act accordingly based on its surroundings, or perform an action.

3. MODEL VALIDATION

To validate our model, we implemented a system to capture and process RGB-D data from sensors, as well as control robot movements and music in a time-synchronous fashion. This system is composed of three clients and a server: two Microsoft Kinect sensors each connected to a seperate computer, a mobile Turtlebot robot (called DJ Rob) [5] who performed specific actions based on pre-defined events, and a stationary system which handled music playback and system actions.

We developed a game in which two participants performed in a dance-off. This dance-off consisted of participants dancing to music for a 60 second duration. During this time, DJ Rob danced along with participants by moving forward and back, allowing his head to bobble, and turning between each participant to simulate judging their dance moves. Figure 2 shows two participants performing in a dance-off.

During the dance-off, we measured expressiveness using the Kinect sensors, and processed that data in real-time using our clients. Based on the processed data, the server makes a decision and causes a DJ Rob (who is decorated to look like a dance jockey) to perform a specific set of actions based on timing information and RGB-D data (see Figure 1). Figure 3b shows different parts of our implemented system.

We defined expressiveness as the enthusiasm in the body movements during our study. According to Barakova et al. [7], and Lourens et al. [25], amplitude and acceleration of the movements of the body joints are sufficient to estimate expressiveness in game settings. Based on these findings, Tetteroo et al. [35] used an area of movement per time unit as a measure of expressiveness. To assess quantitatively, we used a similar measure of expressiveness used by Tetteroo et al. [35] in our study. We calculated expressiveness by taking the average of the product of speed and distance of body joint movements over time.

Expressiveness is thus defined as:

$$expressiveness = \sum_{time=1}^{T} \sum_{joint=1}^{Joints} (speed \times distance)_{joint}/T;$$

Based on the skeletal information, the DPM module of our system calculates the measure of expressivity. For body joint tracking, we used the OpenNI-based PI_tracker package [2] to perform skeletal tracking of participants using the Kinect sensors. This tracker gives us coordinates and orientations of 15 body joints. Each client node tracked the skeletal joint position of a participant standing in front of its sensor.

Following each session, each client published their respective expressiveness measure to the decision module of the server node through an ROS topic. Based on the value of expressiveness, the DM determined the more expressive person based on the largest calculated value of expressiveness. The DM then broadcasts the more expressive participant's number, which is processed by DJ Rob, causing it to perform a gesture toward the more expressive participant.

To assess our system model, we conducted a controlled pilot, followed by a real-world experiment with 70 participants (in pairs of two) at a National Robotics Week event held on campus at the University of Notre Dame [1]. Our experiment allowed us to assess the effectiveness of our system model in a real-world environment. For these sessions, all actions made by our robot, as well as music, data capture, and processing are all controlled using our time-synchronous model. These actions are designed to begin at the same time, such that any actions made by dancers is not missed while measuring expressiveness. We captured all data in real-time and processed the result at the end of a one-minute duration to determine the winning action made by DJ Rob.

4. RESULTS

In general, our overall system model correctly handled all data capture, processing, and actions for our system in a time synchronous manner. In our pilot experiments, we first validated that the robot determined 'no winner' in the case no expressiveness was measured from either client while the music was playing. As the robot determined 'no winner', it did not perform any predefined actions with the participants. We then validated that our robot could determine a winner in the case that a participant only made a small number of movements, while the other did not move. We alternated this level of expressiveness between the first and second user to ensure that our system successfully captured, processed, and synchronized data with the server, so that the robot could successfully determine the more expressive person. In each of these cases, our robot was successful in performing the correct action.

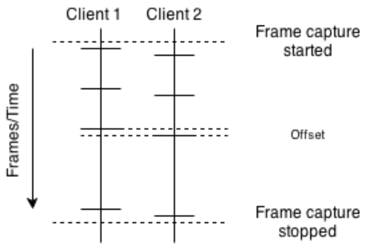


Figure 5: Capturing data in two clients

In addition, our system performed as expected in our real-world experiment. We performed 35 tests, where a pair of participants performed in a dance scenario. During all the tests with our participants, our robot was successful in determining a winner based on the expressiveness measure. To verify that the robot performed the correct action, we manually reviewed each log file following each session with participants to verify that the robot's decision matched that of the participant who had the higher calculated value of expressiveness. In all cases, our robot correctly moved toward the more expressive person with the higher value of expressiveness reported to the server.

While both clients began capturing data at the same time, we found a small difference in timing between the corresponding frames captured from each device. This is due to the fact that the frames captured from each device may have some small offset from each other due to the device's sampling rate and when it is first powered on. For example, while both clients may start capturing data at time zero and each Kinect has a 20 frames per second (FPS) frame rate, the first Kinect might record its first frame at a slight offset from the second. See Figure 5 for an example of this issue.

Considering the frame rate for both Kinects is 20 FPS, each frame is captured on average every 0.05s. Based on our data, the computed average difference between the captured time of the first frames for our two clients across all of our experimental sessions was 0.017 seconds, which is significantly smaller than the time between two consecutive frames.

While we did see a small difference on the millisecond level in our data capture, such a difference will always occur in devices depending on when the first frame occurs, and is negligible. This difference is a result of each sensor's sampling rate, without respect to the time of the client itself. In audio recording devices, for example, such a difference will be much smaller considering a higher sampling rate, and should not affect synchronization when capturing data. Thus, our results suggest that all data was captured and processed in a time synchronous way across all parts of our system.

5. DISCUSSION

Timing of actions by both the human and robot is an important factor in HRI. It is important in HRI for robots to perceive human actions and act accordingly in a timely manner. To achieve appropriate action synchronization, it is essential that sensor data and the entire robotic system is time-synchronized. Our proposed timing model addresses these issues of real-time data capture and processing from multiple sensors. Our model also ensures the occurrence of all the actions across the system in a time-synchronous fashion.

We anticipate that our model will be useful and beneficial to a number of domains. For example, capturing data from multiple sensors around a room will help a robot have an accurate and real-time picture of the environment around it. In addition, multiple robots could be synchronized in this way to send data from their own respective sensors to a main server to process for a specific time-sensitive task.

Such a model may also be beneficial for researchers in the fields of surveillance or computer vision. Our model may provide an efficient way to keep a global synchronous time space across different parts of a surveillance network. In addition, many computer vision algorithms may take advantage of incorporating data from multiple sensors with camera data for better results using our model.

Furthermore, our model is not limited to specific cameras or sensors. A wide variety of sensors, including audio sensors, accelerometers, among others can be used. Our work gives researchers flexibility in capturing various types of data from a number of sources and fusing them together to obtain a more accurate picture of their surroundings in real-time. Based on this work, we are currently in the process of developing a portable and reusable Robot Operating System (ROS) stack to handle time synchronous actions across all parts of our system interacting with humans.

6. REFERENCES

[1] National Robotics Week 2013. `http://engineering.nd.edu/NDNRW`, Jan. 2014.

[2] Pi Tracker Package. `http://wiki.ros.org/pi_tracker`, Jan. 2014.

[3] Robot Operating System (ROS). `http://www.ros.org/`, Jan. 2014.

[4] Ros Topic. `http://wiki.ros.org/rostopic`, Jan. 2014.

[5] Turtlebot Robot. `http://www.willowgarage.com/turtlebot`, Jan. 2014.

[6] K. Arvind. Probabilistic clock synchronization in distributed systems. *IEEE Trans. Parallel Distrib. Syst.*, 5(5):474–487, May 1994.

[7] E. I. Barakova and T. Lourens. Expressing and interpreting emotional movements in social games with robots. *Pers. Ubiquitous Comput.*, 14(5):457–467, Jan. 2010.

[8] M. Cakmak, S. S. Srinivasa, M. K. Lee, S. Kiesler, and J. Forlizzi. Using spatial and temporal contrast for fluent robot-human hand-overs. In *Proceedings of the 6th international conference on Human-robot interaction*, pages 489–496. ACM, 2011.

[9] C. Chao and A. Thomaz. Timing in Multimodal Turn-Taking Interactions: Control and Analysis Using Timed Petri Nets. *J. Human-Robot Interact.*, 1(1):4–25, Aug. 2012.

[10] J. W. Crandall, M. A. Goodrich, D. R. Olsen Jr, and C. W. Nielsen. Validating human-robot interaction schemes in multitasking environments. *IEEE*

Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 35(4):438–449, 2005.

[11] K. Dautenhahn, S. Woods, C. Kaouri, M. Walters, K. L. Koay, and I. Werry. What is a robot companion - friend, assistant or butler? In *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2005).*, pages 1192–1197, 2005.

[12] E. Dias, André and Almeida, Jose and Martins, Alfredo and Silva. Real-Time Visual Ground-Truth System for Indoor Robotic Applications. *Pattern Recognit. Image Anal.*, pages 304–313, 2013.

[13] A. D. Dragan, K. C. Lee, and S. S. Srinivasa. Legibility and predictability of robot motion. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 301–308. IEEE, 2013.

[14] J. Elson and K. Römer. Wireless Sensor Networks : A New Regime for Time Synchronization. *ACM SIGCOMM Comput. Commun. Rev.*, 33(1):149–154, 2003.

[15] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Rob. Auton. Syst.*, 42(3-4):143–166, Mar. 2003.

[16] M. J. Gonzales and L. D. Riek. A sociable robotic aide for medication adherence. In *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '12, pages 38:1–38:4, New York, NY, USA, 2012. ACM.

[17] G. Hager and H. Durrant-Whyte. Information and multi–sensor coordination. *Uncertainty in Artificial Intelligence*, 2:381–394, 1988.

[18] C. Hayes, M. O'Connor, and L. Riek. Avoiding robot faux pas: Using social context to teach robots behavioral propriety. In *2014 9th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2014.

[19] C. J. Hayes, C. R. Crowell, and L. D. Riek. Automatic processing of irrelevant co-speech gestures with human but not robot actors. In *Proceedings of the 8th ACM/IEEE International Conference on Human-robot Interaction*, HRI '13, pages 333–340, Piscataway, NJ, USA, 2013. IEEE Press.

[20] G. Hoffman and C. Breazeal. Cost-based anticipatory action selection for human–robot fluency. *IEEE Transactions on Robotics*, 23(5):952–961, 2007.

[21] T. Iqbal and L. Riek. Assessing group synchrony during a rhythmic social activity: A systemic approach. In *6th Conference of the International Society for Gesture Studies (ISGS)*, 2014.

[22] O. Kermorgant and F. Chaumette. Multi-sensor data fusion in sensor-based control: Application to multi-camera visual servoing. *2011 IEEE Int. Conf. Robot. Autom.*, (3):4518–4523, May 2011.

[23] Y. Khaluf, S. Micus, and F. Weiss. Master Election for Time Synchronization in Swarm Robotic Systems. *2012 IEEE 10th Int. Symp. Parallel Distrib. Process. with Appl.*, pages 285–292, July 2012.

[24] G. Klien, D. Woods, J. Bradshaw, R. Hoffman, and P. Feltovich. Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*, 19(6):91–95, 2004.

[25] T. Lourens, R. van Berkel, and E. Barakova. Communicating emotions and mental states to robots in a real time parallel framework using Laban movement analysis. *Rob. Auton. Syst.*, 58(12):1256–1265, Dec. 2010.

[26] D. Mills. Internet time synchronization: the network time protocol. *IEEE Trans. Commun.*, 39(10):1482–1493, 1991.

[27] M. A. Pérez-Quiñones and J. L. Sibert. A collaborative model of feedback in human-computer interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 316–323. ACM, 1996.

[28] C. Rich, B. Ponsler, A. Holroyd, and C. L. Sidner. Recognizing engagement in human-robot interaction. *2010 5th ACM/IEEE Int. Conf. Human-Robot Interact.*, pages 375–382, Mar. 2010.

[29] L. D. Riek. Wizard of oz studies in hri: A systematic review and new reporting guidelines. *Journal of Human-Robot Interaction*, 1(1), 2012.

[30] L. D. Riek. Embodied computation: An active-learning approach to mobile robotics education. *IEEE Transactions on Education*, 56(1):67, 2013.

[31] L. D. Riek, P. C. Paul, and P. Robinson. When my robot smiles at me: Enabling human-robot rapport via real-time head gesture mimicry. *Journal on Multimodal User Interfaces*, 3(1-2):99–108, 2010.

[32] L. D. Riek, T.-C. Rabinowitch, P. Bremner, A. G. Pipe, M. Fraser, and P. Robinson. Cooperative gestures: effective signaling for humanoid robots. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, pages 61–68. IEEE Press, 2010.

[33] L. D. Riek and P. Robinson. Challenges and opportunities in building socially intelligent machines [social sciences]. *Signal Processing Magazine, IEEE*, 28(3):146–149, 2011.

[34] V. Roto and A. Oulasvirta. Need for non-visual feedback with long response times in mobile hci. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 775–781. ACM, 2005.

[35] D. Tetteroo, A. Shirzad, M. S. Pereira, M. Zwinderman, D. Le, and E. Barakova. Mimicking expressiveness of movements by autistic children in game play. In *2012 International Confernece on Social Computing (SocialCom)*, pages 944–949. IEEE, 2012.

[36] A. L. Thomaz and C. Chao. Turn-taking based on information flow for fluent human-robot interaction. *AI Magazine*, 32(4):53–63, 2011.

[37] R. Wilcox, S. Nikolaidis, and J. A. Shah. Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing. In *Robotics: Science and Systems*, 2012.

[38] N. Xiong and P. Svensson. Multi-sensor management for information fusion: issues and approaches. *Inf. Fusion*, 3(2):163–186, June 2002.