Indriya : An intuitive interface for designing social behaviors of humanoid robots for human-in-the-loop scenarios

Praveenkumar Vasudevan[1] and Gentiane Venture[2]

Abstract— **Humans interacting with intelligent robots has been seen as a potential game changer of the future. In scenarios where robots coexist with humans in a social environment, understanding not only verbal communication, but also non-verbal communication is extremely inevitable. The non-verbal communication carries information such as intention, emotion and health of a human, that adds value to the way robots participate in an interaction. Additionally, the people who design interaction scenarios are from diverse fields who do not essentially have the required robot programming skills. In this paper we propose an easy to use and intuitive programming interface which gives the power to design robot behaviors taking into account human motions. We propose a distributed system namely Indriya which gives the capability to plug and play multi-modal motion recognition systems and diverse class of robots. We present results of NAO humanoid robot performing actions understanding human motions using Kinect motion recognition system.**

I. INTRODUCTION

The richness and diversity of Human Robot Interaction (HRI) has been described in [1] as "HRI is a challenging research field at the intersection of psychology, cognitive science, social sciences, artificial intelligence, computer science, robotics, engineering and human-computer interaction". Goodrich in his extensive survey [2] proposed two main types of HRI namely remote interaction and proximate interaction. The latter is particularly important where the humans and the robots are co-located. Proximate interaction has gained importance due to the successful encounters of putting robots to work with human beings. It has led to the development of a new class of robots called social robots. Yan et al. [3] define "A social robot is a robot which can execute designated tasks and the necessary condition turning a robot into a social robot is the ability to interact with humans by adhering to certain social cues and rules."

Social robots already entered the human spaces as entertainers, educators, caring agents and personal assistants [4]. Hence the design and the development of interaction systems need to be approached in a systematic manner wherein the robots should be able to understand the human motions and intentions in order to interact in a better way. To make it possible it is necessary to develop robotic systems with essential perceptual ability for efficient and natural interaction. Most often the on-board sensors on the robots fail to satisfy this demanding requirement due to various constraints like space, power and computational needs. Therefore consideration of augmenting exteroceptive sensors that are commonly available in the smart home/public environments to this purpose is essential.

Another important aspect in HRI is the fact that the users of such systems are from diverse backgrounds. People study various aspects such as robot ethics, social acceptance, liveliness, cultural influence etc., from various perspectives like sociology, psychology, humanities and so on. So the tools needed to design behaviors of a social robot should be intuitive and user friendly. With increased availability of social robots and cost effective motion recognition sensors, we could still observe a huge void which inhibits the exploitation of available technology for designing human motion driven robot behaviors.

The main contribution of this work is an application independent experimental platform: Indriya, wherein a social robot is augmented with essential perceptual ability to understand human motions. The behavior design of such a social robot is made possible by an easy to use behavior design interface. The resulting experimental platform could be used by people from interdisciplinary fields to design the interaction between social robots and humans. Although our behavior design framework could accommodate variety of robots and sensors, we present the scenarios targeted for the NAO [5] humanoid robot and commercially available Kinect [6] RGB-D camera for localization and gesture recognition.

II. RELATED WORK

Vision based motion capture and analysis systems have been one of the first class citizens in the human motion capture and analysis. It has been studied widely and a summary of all the approaches developed during the past decade has been presented in the surveys [7][8]. Vision based human pose estimation has traditionally suffered from the requirement to adopt an initialization pose and losing track after a few frames. These problems have been addressed by the approaches proposed by Xbox team [6] which are capable of accurately tracking human skeletons using single depth images [9]. Understanding of human motion is not complete if the action of the human could not be inferred. In the survey by Microsoft research team [10], a study on various algorithms used for human activity analysis is presented. Recently data-driven machine learning approaches have proven to be successful with recognition accuracy as high as 94.9% [6].

*This work was supported by JASSO

[1]Praveenkumar Vasudevan is a graduate student in robotics at Ecole centrale de Nantes, 1 rue de la noe, 44000 Nantes, France praveenv4k@gmail.com

[2]Gentiane Venture is an Associate Professor at the Department of Mechanical Systems Engineering, Tokyo University of Agriculture and Technology, 2-24-16 Koganei, Tokyo - 1848588, Japan venture@cc.tuat.ac.jp

The localization of humanoid robots is a challenging issue, due to rough odometry estimation, noisy onboard sensing, and the swaying motion caused by walking [11]. The Point cloud library [12], one of the most widely used 3d perception library, implements ready to use probabilistic tracking algorithms. Studies on robot localization, obstacle mapping, and path planning by equipping NAO with a consumer-level depth camera have been reported in [13]. Localization and motion planning in smart home environment using an external kinect sensor have been proposed in [11]. These methods are computationally demanding and it could cause overall performance degradation particularly when one wants to share the same sensor for both human motion recognition and localization of the robot. Tracking rectangular fiducial markers using augmented reality tool-kits like ALVAR [14] can be interesting if one could embed those markers on the humanoid robot. This is one of the simplest and cheapest solutions in terms of computational power as it can provide position and orientation of the physical markers in real time.

The users of social robots do not have necessarily backgrounds in programming and design of robot behaviors. The main challenge in the behavior design is the ability to define the behavior which can abstract complex data flows from the end user. There exists flow-chart based visual programming languages [5] which allow non-programmers to create robot applications using a set of pre-built behavioral blocks. These programs are very intuitive but when it comes to designing reactive behaviors for human-in-the-loop scenarios, the existing visual programming methods increase the cognitive load on the end users. Specialized robot programming techniques like Task description language [15] and middlewares like ROS [16] have been proposed in the literature. Though these systems provide modular and distributed architecture, support multiple sensors and robots etc., these require high level of skill in robotics and programming. Recently non-domain-specific solution like Targets-Drives-Means is proposed in [17], however it lacks an intuitive interface.

Nowadays there has been a lot of efforts to teach programming to children and people without computer science background [18][19]. These tools are very intuitive and have already been proven to be used by novice programmers to build games and educational applications. The Blockly library [19] from Google offers a complete client side JavaScript library which could be used for developing custom blocks and code generators as per the application requirements.

III. INTUITIVE BEHAVIOR INTERFACE

We propose a light weight interface for designing human motion driven behaviors taking inspiration from distributed architecture [16] and intuitive visual programming techniques [19].

A. System Architecture

The system setup and architecture are shown in Fig **??**. The principal components of the architecture are

- **Application Components**
  - *Context*: The application context contains the complete description of the world. It contains latest information about all the robots including their location, sensor data, status etc., It also contains information about all the humans in the environment along with their active motions/gestures as supplied by the motion recognition modules.
  - *Parameter Server*: The parameter server acts as a central repository for managing the parameters of the system and of the distributed components.
  - *Embedded Web Server*: The web server embedded in the application serves the file and data requests from the web client.
  - *Context Orchestrator*: The orchestrator keeps the Context uptodate by synchronizing with information of the robots and humans published by the distributed components.
  - *Behavior Program* : A dynamic component that will be created when the user starts the program he/she designed using the user interface. The declarative description of the behavior described in Section III-B is parsed in order to create a memory model. The behavior program node monitors the application context for the motion triggers and invokes the corresponding robot actions according to the way it is being described in the program.
- **Distributed Components** : These are nodes in the system each with a specific goal that can be started/stopped at any time during the entire application life-cycle without affecting the other nodes or the system. All the nodes will communicate with the application using message passing techniques. They can run in any machine inside the network.
  - *Motion Recognition Node* : A dedicated node that interacts with a motion recognition sensor and sends the detected motions and gestures to the application. Additionally each motion recognition module registers a set of motions/gestures that could be detected with the sensor associated with it.
  - *Robot Interface Node* : A dedicated node that interacts with a specific robot and can invoke a set of actions on it. It also sends periodic update about the robot status to the application. Moreover it registers a set of parameterizable actions that could be invoked on the robot associated with it.
  - *Localization Node* : A dedicated node which uses the perception system to compute the position and orientation of the robot and humans in the environment.
- **User Interface**: The user interface is a web application that runs on any latest web-kit browsers supporting WebGL technology. The prime goal of this UI is to make it suitable for environments adopting bring your own device (BYOD) policy.
  - *Behavior Designer*: The Behavior designer surface

could be used by the user to drag and drop the behavior blocks and construct the program by putting together motion recognition blocks and robot action blocks. The designed behavior will be encoded into a declarative XML format and sent to the server when the user request to start the program. The designer offers a full range of capabilities like Create/Edit/Delete/Save behavior programs.
  - *Visualization*: The visualization could be used to see the interaction of the human and robot inside a virtual 3D environment.

B. Behavior Program

The behavior program is structured in a simple way so that it could be easily understood by the end user. The conceptual model of behavior program is shown in Fig. **??** and the block level implementation is shown in Fig. **??**. The behavior program is composed of:

- *Start-up and Exit Blocks*: The start-up block will be executed once when the user starts the program. The user can add a set of actions to be performed when the program starts. Similarly the exit block will be executed once when the lifetime of all the configured behavior blocks expire. Both these blocks are optional and there cannot be more than one start-up and exit blocks in a program.
- *Behavior Block*: The behavior block is composed of
  - A **trigger** that activates this block. The trigger source could be either of human presence/absence, gesture, vicinity of human, verbal command etc.,
  - The **lifetime** of each behavior block could be configured to run only once, forever or until a condition is met.
  - The **priority** of the block could be set to low, normal or high and the execution is done based on fixed-priority pre-emptive scheduling.
  - Similar to the behavior program level, at each behavior block level a set of **startup** and **exit** actions could be set which would be executed only once during the creation and termination respectively. The **cyclic** actions will be performed each time the trigger condition is met.

IV. USING THE TEMPLATE

Use this sample document as your LaTeX source file to create your document. Save this file as **root.tex**. You have to make sure to use the cls file that came with this distribution. If you use a different style file, you cannot expect to get required margins. Note also that when you are creating your out PDF file, the source file is only part of the equation. *Your TeX $\rightarrow$ PDF filter determines the output file size. Even if you make all the specifications to output a letter file in the source - if your filter is set to produce A4, you will only get A4 output.*

It is impossible to account for all possible situation, one would encounter using TeX. If you are using multiple TeX files you must make sure that the “MAIN“ source file is called root.tex - this is particularly important if your conference is using PaperPlaza’s built in TeX to PDF conversion tool.

A. Headings, etc

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named Heading 1, Heading 2, Heading 3, and Heading 4 are prescribed.

B. Figures and Tables

Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation Fig. 1, even at the beginning of a sentence.

TABLE I
AN EXAMPLE OF A TABLE

One	Two
Three	Four

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an document, this method is somewhat more stable than directly inserting a picture.

Fig. 1. Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity Magnetization, or Magnetization, M, not just M. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write Magnetization (A/m) or Magnetization A[m(1)], not just A/m. Do not label axes with a ratio of quantities and units. For example, write Temperature (K), not Temperature/K.

V. CONCLUSIONS

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

APPENDIX

Appendixes should appear before the acknowledgment.

ACKNOWLEDGMENT

We would like to thank the members of GVLab at Tokyo University of Agriculture and Technology for helping us arranging the resources and participating in the experimentation.

References are important to the reader; therefore, each citation must be complete and correct. If at all possible, references should be commonly available publications.

REFERENCES

[1] K. Dautenhahn, "Methodology and themes of human-robot interaction: a growing research field," *Int. J. of Advanced Robotic Systems*, vol. 4, no. 1, pp. 103–108, 2007.

[2] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: a survey," *Foundations and trends in human-computer interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[3] H. Yan, M. H. Ang Jr, and A. N. Poo, "A survey on perception methods for human–robot interaction in social robots," *Int. J. of Social Robotics*, vol. 6, no. 1, pp. 85–119, 2014.

[4] Aldebaran. https://www.aldebaran.com/en. Accessed: 2014-11-15.

[5] Aldebaran, "NAO humanoid robot." https://www.aldebaran.com/en/humanoid-robot/nao-robot-working. Accessed: 2014-11-15.

[6] Microsoft, "Kinect for Windows." https://www.microsoft.com/en-us/kinectforwindows/. Accessed: 2014-11-20.

[7] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2, pp. 90–126, 2006.

[8] R. Poppe, "Vision-based human motion analysis: An overview," *Computer vision and image understanding*, vol. 108, no. 1, pp. 4–18, 2007.

[9] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake, "Efficient human pose estimation from single depth images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 12, pp. 2821–2840, 2013.

[10] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *Cybernetics, IEEE Transactions on*, vol. 43, no. 5, pp. 1318–1334, 2013.

[11] E. Cervera, A. A. Moughlbay, and P. Martinet, "Localization and navigation of an assistive humanoid robot in a smart environment," in *IEEE Int. Workshop on Assistance and Service Robotics in a Human Environment*, Oct, 2012.

[12] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Robotics and Automation (ICRA), 2011 IEEE Int. Conf. on*, pp. 1–4, IEEE, 2011.

[13] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3D environments based on depth camera data," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS Int. Conf. on*, pp. 692–697, IEEE, 2012.

[14] V. T. R. C. of Finland, "ALVAR : A library for virtual and augmented reality." www.vtt.fi/multimedia/alvar.html. Accessed: 2015-02-20.

[15] R. Simmons and D. Apfelbaum, "A task description language for robot control," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ Int. Conf. on*, vol. 3, pp. 1931–1937, IEEE, 1998.

[16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," *ICRA workshop on open source software*, vol. 3, no. 3.2, p. 5, 2009.

[17] V. Berenz and K. Suzuki, "Targets-drives-means: A declarative approach to dynamic behavior specification with higher usability," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 545–555, 2014.

[18] M. M. Lab, "Scratch." https://scratch.mit.edu/. Accessed: 2015-05-15.

[19] Google, "Blockly." https://developers.google.com/blockly/. Accessed: 2015-04-31.