An Intuitive Interface for designing Social behaviors based on Human motions

Praveenkumar Vasudevan[1] and Gentiane Venture[2]

[1] Graduate Student, École Centrale de Nantes, Nantes, France,
praveenv4k@gmail.com
[2] Associate Professor, Tokyo University of Agriculture and Technology, Japan
venture@cc.tuat.ac.jp

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 architecture 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.

Keywords: human robot interaction, motion recognition, robot behaviors, kinect, nao

1   Introduction

Human Robot Interaction (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[11]. Goodrich[12] in his extensive survey proposed two main types of HRI namely Remote interaction or Teleoperation and Proximate interaction. The latter is particularly important where the humans and the robots are co-located (for example, service robots may be in the same room as humans). 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. [30] defines *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.*

The Social robots already entered the human spaces as entertainers[3], educators[3], caring agents[1] and personal assistants[4]. The design and 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 cognitive skills 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 requirements. Hence consideration of augmenting exteroceptive sensors that are commonly available in the smart home/public environments is important.

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 will be to develop an application independent experimental platform wherein a social robot will be augmented with essential perceptual ability to understand human motions. The behavior design of such a social robot will be made possible by an easy to use behavior design interface. The resulting experimental platform could be used by people from interdisciplinary fields to design, evaluate and study specific aspects of the interaction between social robots and the human. We propose to use commercially available Kinect RGB-D cameras for the relative localization of human and robot and also as a gesture recognition system.

2 Related work

Vision based motion capture and analysis has been studied widely and a summary of all the approaches developed during the past three decades have been presented in the surveys [20][21][22]. 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[15] team which are capable of accurately predicting the 3D positions of body joints using single depth images without using any temporal information[27]. Unlike the approaches used in the Kinect SDK, the approach presented in [8] uses both the depth and color(RGB-D) data for human body detection and pose estimation using a customizable human kinematic model. Understanding of human motion is not complete if the action of the human could not be understood. In the survey by Microsoft research team[14], a background study on various algorithms used for human activity analysis is presented. Recently data-driven machine learning approaches like neural networks, support vector machines, clustering, decision trees and bayesian networks have proven to be successful with recognition accuracy as high as 94%[16].
The localization of humanoid robots is a challenging issue, due to rough odome-

try estimation, noisy onboard sensing, and the swaying motion caused by walking[9]. The Point cloud library[25] which is one of the most widely used 3D perception software library, implements ready to use probabilistic tracking algorithms[29]. Studies on robot localization, obstacle mapping, and path planning in multilevel 3D environments by equipping Nao with a consumer-level depth camera has been reported in [19]. Localization and motion planning in smart home environment using an external Kinect sensor have been proposed in [9]. In [10] a robust particle filter parallelized on a GPU that can track a known 3D object model over a sequence of RGB-D images is proposed. However all these methods are computationally demanding and it could cause overall performance degradation particularly when we want to share the same sensor for both human motion recognition and localization of the robot. Tracking rectangular fiducial markers using augmented reality toolkits like ARToolKit[18] can be interesting if we could embed those markers on the humanoid robot. This is one of the simplest and cheapest solution in terms of the computational power as it can provide position and orientation of the physical markers in real time.

The users of social robots do not have necessary 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 several flow-chart based visual programming languages[17],[2] which allow non-programmers to create robot applications using a set of pre-built behavioral blocks. These programs are very intuitive and allow the users to realize complex sequence of movements and sequential behaviors. However 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[28], scripting techniques like Universal Robotic Body Interface(URBI)[6] and ROS[23] have been proposed in the literature. Though these programs provide modular and distributed architecture, support multiple sensors and robots etc., all these tools require high level of skill in robotics and programming to use them. Recently non-domain-specific solution like *Target Drives Means* is proposed in [7], however it lacks an intuitive interface.

Recently there has been lot of efforts to teach programming to children and people without computer science background[5][13]. 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[13] 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.

3   Motion Driven Behavior Interface

We propose a light weight interface for designing human motion driven behaviors taking inspiration from distributed architecture like ROS[23] and intuitive visual programming techniques[13].

3.1 System Architecture

The important components of the system are

- **User Interface**: The user interface is a web application that runs on any latest web-kit browsers supporting WebGL technology.
  - *Behavior Designer*: The Behavior designer surface could be used by the user to drag and drop the behavior blocks and construct the program using the set of motion capabilities registered by the active motion recognition nodes and the set of robot action capabilities registered by the active robot interface nodes. The behavior described using the designer 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.
- **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 and status. 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 distributed components.
  - Embedded Web Server: The web server embedded in the application serves the file and data requests of the web client.
  - *Context Orchestrator*: The orchestrator collect uptodate information about the robot and human in the environment from the perception system and updates the Context.
  - *Bootstrapper* : The bootstrapper takes care of initializing the system and starting up all the pre-configured nodes. It also takes of starting and stopping the behavior programs when requested by the user.
  - 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 is parsed in order to create a memory model. The Behavior program node monitors the action 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 gestures and motions to the application. Additionally each motion recognition module registers a set of actions/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 actions that could be invoked on the robot associated with it.
- *Localization Node* : A dedicated node which uses the perception system to resolve and publish the current position of the robot and the human which is very important for interaction.
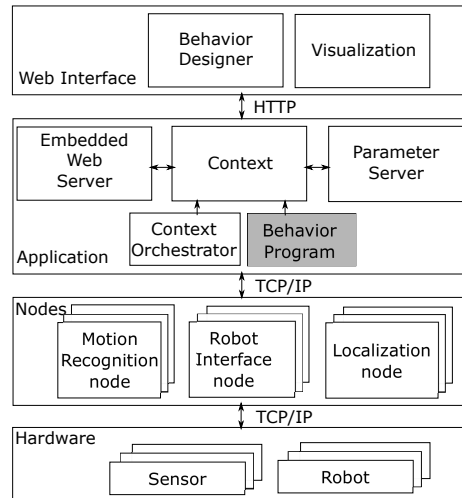
Fig. 1. System Architecture

3.2 Workflow

Describe the workflow of the operations

3.3 Implementation

For the implementation

- Zero mq for communication
- Protobuf for messaging description
- Google blockly

4 Experimentatal Setup

The software is tested on a 64-bit Intel Core i7-4790 CPU 3.60 Ghz with 8 GB of RAM on a Microsoft Windows 8.1 operating system. The Kinect for Windows

V2 is used as the motion capture system and robot is used is Nao Humanoid V5 powered by latest NaoQi OS V2.3. A set of gestures are created using the Visual Gesture Builder that comes with the Kinect Sdk[16]. The process involves capturing the clips of the motions, tagging the clips with the appropriate gestures and training the gesture recognizer. The trained gestures are then exported as a Visual gesture database and integrated with the motion recognition node. Similarly a set of actions of the Nao humanoid robot are developed as python scripts that are being named corresponding to the actions. The Robot interface node is responsible for invoking these actions whenever requested. The python scripts uses Naoqi python sdk which exposes a huge set of API for motion control, sensor data access etc.,

5   Conclusions

For the moment we have evaluated our system only for the Kinect motion capture system working seamlessly with the Nao humanoid robot for a set of predefined gestures and robot actions. We are planning to develop an extensive database containing commonly encountered gestures and also an extensive set of primitive robot actions. These will be made available to the end user through the intuitive programming interface which could be then used for defining complex motion driven behaviors. Additionally we are also planning to integrate our system to work with other modes of motion recognition like IMU, Accelerometers and Gyroscopes that are available in smart-phones and wearable devices. Similarly we would like to integrate other system with other robots like Pepper which we expect to receive soon.

6   Acknowlegdements

References

1. Aldebaran. Ask nao: Autism solution for kids. https://asknao.aldebaran.com. Accessed: 2014-12-12.
2. Aldebaran. Choregraphe software. http://www.aldebaranrobotics.com/en/Discover-NAO/Software/choregraphe.html. Accessed: 2014-11-20.
3. Aldebaran. Nao humanoid robot. https://www.aldebaran.com/en/humanoid-robot/nao-robot. Accessed: 2014-11-15.
4. Aldebaran. Project romeo. http://projetromeo.com/en/scenario. Accessed: 2015-04-20.
5. Lifelong Kindergarten Group at the MIT Media Lab. Scratch. https://scratch.mit.edu/. Accessed: 2015-05-15.

6. J. C. Baillie. Urbi 2: Introduction to concurrent real-time programming. In *Proceedings of the Third International Workshop on Software Development and Integration in Robotics*. IEEE ICRA, May 2008.

7. Vincent Berenz and Kenji Suzuki. Targets-drives-means: A declarative approach to dynamic behavior specification with higher usability. *Robotics and Autonomous Systems*, 62(4):545–555, 2014.

8. Koen Buys, Cedric Cagniart, Anatoly Baksheev, Tinne De Laet, Joris De Schutter, and Caroline Pantofaru. An adaptable system for rgb-d based human body detection and pose estimation. *Journal of Visual Communication and Image Representation*, 25(1):39 – 52, 2014. Visual Understanding and Applications with RGB-D Cameras.

9. Enric Cervera, Amine Abou Moughlbay, and Philippe Martinet. Localization and navigation of an assistive humanoid robot in a smart environment. In *IEEE International Workshop on Assistance and Service Robotics in a Human Environment*, 2012.

10. Changhyun Choi and Henrik I. Christensen. RGB-D object tracking: A particle filter approach on GPU. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, pages 1084–1091, 2013.

11. Kerstin Dautenhahn. Methodology and themes of human-robot interaction: A growing research field. *International Journal of Advanced Robotic Systems, Vedran Kordic, Aleksandar Lazinica and Munir Merdan (Ed.)*, 2007.

12. Michael A. Goodrich and Alan C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

13. Google. Blockly. https://developers.google.com/blockly/. Accessed: 2015-04-31.

14. J. Han, L. Shao, D. Xu, and J. Shotton. Enhanced computer vision with microsoft kinect sensor: A review. *IEEE Trans. Cybernetics*, 43(5), October 2013.

15. Microsoft Inc. Kinect for windows. http://www.microsoft.com/en-us/kinectforwindows/. Accessed: 2014-11-20.

16. Microsoft Inc. Kinect for windows sdk. http://www.microsoft.com/en-us/download/details.aspx?id=44561. Accessed: 2014-11-20.

17. Microsoft Inc. Microsoft robotics developer studio 4. http://msdn.microsoft.com/library/bb483024. Accessed: 2014-11-28.

18. Hirokazu Kato and Mark Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *2nd IEEE and ACM International Workshop on Augmented Reality, IWAR '99, San Francisco, CA, USA, October 20-21, 1999*, pages 85–94, 1999.

19. Daniel Maier, Armin Hornung, and Maren Bennewitz. Real-time navigation in 3d environments based on depth camera data. In *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012), Osaka, Japan, November 29 - Dec. 1, 2012*, pages 692–697, 2012.

20. Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231 – 268, 2001.

21. Thomas B. Moeslund, Adrian Hilton, and Volker Krger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(23):90 – 126, 2006. Special Issue on Modeling People: Vision-based understanding of a persons shape, appearance, movement and behaviour.

22. Ronald Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(12):4 – 18, 2007. Special Issue on Vision for Human-Computer Interaction.

23. Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. *ICRA workshop on open source software*, 3(3.2):5, 2009.

24. Microsoft Research. Touchdevelop. https://www.touchdevelop.com/. Accessed: 2015-05-15.

25. Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (PCL). In *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, 2011.

26. Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *In In CVPR, 2011. 3*, 2011.

27. Jamie Shotton, Ross Girshick, Andrew Fitzgibbon, Toby Sharp, Mat Cook, Mark Finocchio, Richard Moore, Pushmeet Kohli, Antonio Criminisi, Alex Kipman, and Andrew Blake. Efficient human pose estimation from single depth images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2821–2840, 2013.

28. Reid G. Simmons and David Apfelbaum. A task description language for robot control. In *Proceedings 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications, October 13-17, 1998, Victoria, BC, Canada*, pages 1931–1937, 1998.

29. Ryohei Ueda. Tracking 3d objects with point cloud library. http://pointclouds.org/news/2012/01/17/tracking-3d-objects-with-point-cloud-library/. Accessed: 2015-01-10.

30. Haibin Yan, Jr. Ang, Marcelo H., and Aun Neow Poo. A survey on perception methods for humanrobot interaction in social robots. *International Journal of Social Robotics*, 6(1):85–119, 2014.