Adding Speech to a Robotics Simulator

T.Shanmugapriya[1], S.Senthilkumar[2]

Assistant Professor, Department of Information Technology, SSN Engineering College, Chennai, Tamil Nadu, India[1]

Assistant Professor, Department of Electronics and Instrumentation, Bharath University, Chennai, Tamil Nadu, India[2]

Abstract We present a demo showing different levels of emergent verbal behaviour that arise when speech is added to a robotics simulator. After showing examples of (silent) robot activities in the simulator, adding speech output enables the robot to give spoken explanations of its behaviour. Adding speech input allows the robot's movements to be guided by voice commands. In addition, the robot can modify its own verbal behaviour when asked to talk less or more. The robotics toolkit supports different behavioural paradigms, including finite state machines. The demo shows an example state transition based spoken dialogue system implemented within the robotics framework. Other more experimental combinations of speech and robot behaviours will also be shown.

I. INTRODUCTION

Human-robot interaction is an area where recently much work has been focussed. There are possibilties not only to demonstrate integrated technological platforms for various input and output modalities, but also the rich interaction capabilities that spoken dialogues offer as a means of interfacing between humans and computers.

In this paper we focus on human-robot interaction related to communication on the level of providing feedback on one's actions. In particular the robot needs to give explanations about where it is going and what it is doing. This kind of interaction is important in the context of "socially interactive robots" [2]. Robots of this type need to provide a natural interface for interacting with users. They need to adopt
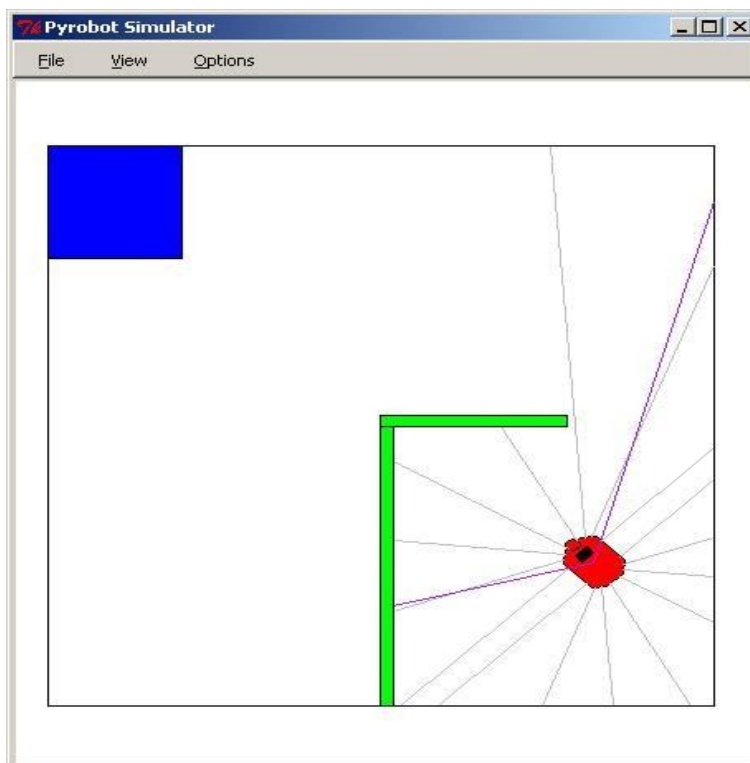

Fig. 1 Pyro Robotics Simulator: A Red Pioneer robot in Tutorial World

many of the human communication modalities and social cues that enable smooth communication among humans.

Several different levels of verbal behaviour and spoken interaction can arise when speech is added to a robotics system. These different behavioural and interactional levels are described as emergent verbal behaviour in [5], where the underlying ideas are discussed further. The current paper describes a demo that shows examples of the various behaviours in practice using a robotics simulator.

The demo is structured as follows. Section 2 introduces the robotics toolkit and shows some silent autonomous behaviours by the robot. Section 3 describes the speech interface, and gives examples of simple verbal behaviour by the robot and of human-robot interaction by voice commands from the human. The robot begins to adopt human-like behaviour and adapt to social cues for smooth communication occurs when it modifies its own verbal behaviour in response to being asked to talk less or more. Section 4 describes a state transition based spoken dialogue system demo that uses the robotics toolkit's finite state machine paradigm. Section 5 briefly introduces some areas of ongoing experimentation.

II. PYRO ROBOTICS

The robotics framework used in the demo is Pyro [1], an open source Python toolkit for exploring topics in artificial intelligence and robotics. Pyro can be downloaded from **http://pyrobotics.org**. Excellent tutorial materials are available at **http://pyrobotics.org/?page=PyroModulesContents** and a video at **http://pyrobotics.org/video/** shows how to get started.

Although Pyro can be used to control a range of real robots as well as simulations, the demo uses only simulated robots running in simulated worlds. In Figure 1 the robot is a "Red Pioneer" and the world is Tutorial World. The robot has sensors that can detect obstacles and other objects in front, behind, left and right.

The robot's "brain" is a Python program that controls the robot's behaviour. At the most basic level, the robot can move forward or back and can turn itself to left or right. Appropriate reactions to signals from the sensors can be programmed, so that the robot begins to behave autonomously. Simple autonomous behaviours include Avoid (turning away from detected objects) and Wander (making random decisions to turn slightly to left or right).

```
def determineMove(self, front, left, right):
if front < 0.5:
#print "obstacle ahead, hard turn" return(0, .3)
elif left < 0.8:
#print "object detected on left, slow turn" return(0.1, -.3)
elif right < 0.8:
#print "object detected on right, slow turn" return(0.1, .3)
else:
#print "clear" return(0.5, 0.0)
```

Fig. 2 Diagnostic print statements in the Avoid brain program

A fragment of the Python program for the Avoid brain is shown in Figure 2. This function determines the robot's next move by checking signals from its sensors. Note that the sensor checks such as **if left < 0.8** and the next moves such as **return(0.1, -.3)** are specified in low-level numerical formats. However, as Pyro is intended for teaching robotics, helpful diagnostic print statements such as **#print "object detected on left, slow turn"** have been left
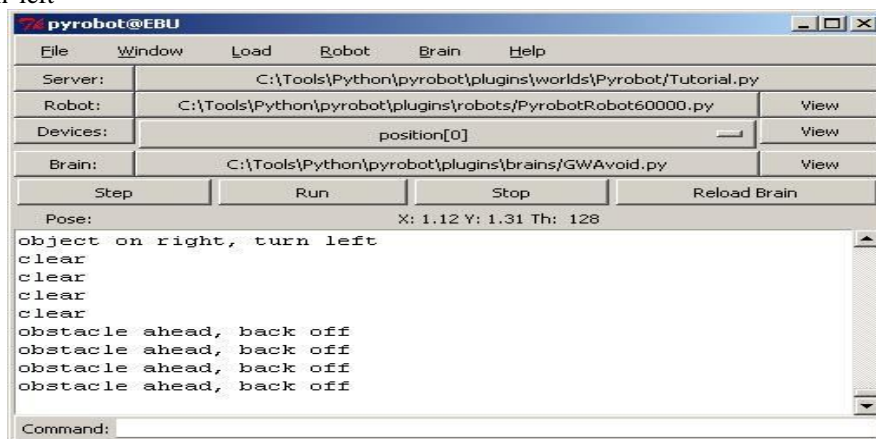

Fig. 3 Pyro Robotics Control Panel

in the code by the programmer as explanatory comments. These print statements provide a basis for the robot's spoken behaviour in Section 3.

The choice of world, robot, and brain is made with the Pyrobot graphical user interface shown in Figure 3. Different brain programs can be loaded and changed very quickly using this control panel. Here a modified version of the Avoid brain has been loaded. This version avoids detected objects and prints brief messages to the console explaining its movements. "Clear" means that the way ahead is clear and the robot simply moves forward.

Commands can also be typed into the command line at the bottom of the control panel. These commands use the same numerical format as the brain programs, for example `robot.rotate(-0.3)` to make a small turn to the right. More human- oriented voice commands are described in Section 3.

Adding Speech

The demo uses a Windows laptop. Speech recognition and synthesis are performed by the MicroSoft speech engine (Speech SDK 5.1). The interface between the Pyro robotics toolkit and the speech engine is pyspeech [3], which provides convenient functions for text-to-speech and for recognizing words from a given list.

The printed messages shown in Figure 3 are turned into speech by the text-to- speech component of pyspeech. This enables the robot to achieve a basic level of one-way communication by explaining its movements. However, when avoidance actions are not needed the robot just repeats "clear, clear, clear..." constantly while moving forward. This quickly becomes irritating as the robot is not saying anything new. It is easy to improve the robot's verbal behaviour by not repeating the last utterance. Then the robot only speaks when it has something interesting to say.

Voice commands such as "go forward, go back, turn left, turn right" can be rec- ognized by the speech recognition component of pyspeech. These commands can be used for direct human control of the robot at the basic level. However, it would be tedious to specify all the robot's movements at the basic level, and the program in the robot's brain enables it to perform its own autonomous behaviour. Human voice commands need to be used only when the robot becomes "stuck", which happens from time to time.

More importantly, speech input introduce the possibility of two-way communi- cation. A nice example of human robot interaction in the demo is that the human can ask the robot to "talk less" or to "talk more". These commands change the robot's verbosity level. At high verbosity the robot says "clear, clear, clear" constantly while moving forward. At medium verbosity the robot says new things but does not keep repeating the same thing. If told to "shut up" the robot switches immediately to the lowest verbosity level where it continues to think and behave autonomously but keeps its thoughts to itself and says nothing.

III.   SPOKEN DIALOGUES

The Pyro robotics framework supports several different paradigms, including finite state machines, reinforcement learning, fuzzy decision-making, neural networks and evolutionary algorithms. The toolkit aims to allow experimentation with different approaches to developing robot behaviours.

We use Pyro's finite state machines paradigm to implement state transitions for dialogue control, a well-known approach in spoken dialogue systems [4]. Other uses of finite state machines in robotics are described in [1]. When finite state machines are combined with a speech interface, state transition based spoken dialogue systems can be implemented directly within the Pyro robotics framework.

This part of the demo presents a "classical" spoken dialogue system for flight reservations in which dialogue control is implemented by state transitions. As the purpose of the demo is to show the state transition mechanism clearly, the robot's behaviour is purely verbal and it does not move. In addition to the states for getting the departure and destination places and the departure and return days, the demo includes states for summarizing the trip (shown in Figure 4) and for starting over in case of misunderstandings.

```
class summary(State):
def step(self):
speech.say("Starting  from " + self.brain.departCity) speech.say("Going to  " +
self.brain.arriveCity) speech.say("Starting  on " + self.brain.departDay) speech.say("Returning
on " + self.brain.returnDay) speech.say("Is that all  OK?")
```

```
phrase = speech.input("", confirms + rejects)
if phrase in confirms:
self.goto('goodbye')  # successful conclusion elif phrase in rejects:
self.goto('restart')  # restart entire dialogue else:
self.goto(self.name)  # retry the summary
```

Fig. 4 Extract from "summary" state in state transition dialogue program

IV.   FURTHER WORK

Other more experimental combinations of speech and robot behaviours will also be shown in the demo. Current work on developing open-domain conversational interaction using Wikipedia as a knowledge source is discussed in [5]. The main idea is that the robot should be able to go about performing its own autonomous behaviours without unnecessary human supervision, but the human should be able to intervene when desired in order to change the robot's behaviour. The robot should therefore have a range of behaviours of different types, and the human should be able to tell the robot when to switch to a more suitable behaviour. As the Pyro robotics framework supports several different behavioural paradigms, and has mechanisms for switching between them, the combination of Pyro with the speech interface offers a good basis for experimentation in this area.

REFERENCES

1. Blank, D., Kumar, D., Meeden, L., Yanco, H.: The Pyro toolkit for AI and robotics. AI Maga- zine 27(1), 39–50 (2006)
2. Fong, T., Nourbaksh, I., Dautenhahn, K.: A survey of socially interactive robots. Robotics and
Autonomous Systems 42, 143–166 (2003)
3. Gundlach, M.: pyspeech: Python speech recognition and text-to-speech module for Windows
(2011). http://code.google.com/p/pyspeech/
4. Jokinen, K., McTear, M.: Spoken Dialogue Systems. Morgan & Claypool (2009)
5. Jokinen, K., Wilcock, G.: Emergent verbal behaviour in human-robot interaction. In: Proceed- ings of 2nd International Conference on Cognitive Infocommunications (CogInfoCom 2011). Budapest (2011)