Motivation

As part of the “Survivor Buddy” project, we have created an open source speech translation toolkit which allows written or spoken word from multiple independent controllers to be translated into either a single synthetic voice, synthetic voices for each controller, or unchanged natural voice of each controller. The human controllers can work via the internet or be physically co-located with the Survivor Buddy robot. The toolkit is expected to be of use for exploring voice in general human-robot interaction.

The Survivor Buddy project is motivated by our prior work which suggests that a trapped victim of a disaster, or other human who is dependent, will treat a rescue robot as a social medium and that the choice of robotic voice will be important. The robot will be both a medium to the “outside” world and a local, independent entity devoted to the victim (e.g., a buddy). One function of the medium is to provide two-way audio communication between the survivor and the emergency response personnel, but more interesting capabilities emerge by fully exploiting web applications. A web-enabled robot with a LCD screen could permit the survivor to videoconference with responders (or family), watch live TV, movies, or listen to music. Because the robot would represent multiple people and perform multiple roles, the choice of voice might influence trust, likability, and enhance cognitive functions.

The potential significance of choice of robotic voice in a social medium was highlighted by our recent experiences with “SciGirls.” In an episode of “SciGirls,” a Public Broadcast System science reality show, four middle-school girls hypothesized that an extroverted Survivor Buddy would be better liked than an introverted Survivor Buddy, used a text-to-speech interface to generate two synthetic voices for the robot with the pitches and frequencies associated with introversion and extroversion, then tested their hypothesis by having 12 of their friends play tic-tac-toe with the robot.

Voice Toolkit

In order to explore questions as to the relative contribution of voice versus nonverbal cues and about the type of voice for a social medium, we have built a speech translation toolkit. The toolkit allows natural voice communication as well as extends existing voice recognition and text-to-speech systems into a framework of network services. By wrapping recognition and generation abilities as network services, users can choose the best tool available, regardless of their platform or installed libraries. Wrapping a service is a simple process, which involves fashioning a program with
a network layer around the desired framework. For example, the base implementation provides recognition services based on Microsoft’s Speech APIs and CMU’s Sphinx 4 with generation services built on FreeTTS or Apple’s built in operating system voices. The network wrapper allows connected clients full functionality of the existing voice framework.

The toolkit is easy to deploy, using peer-to-peer communication between the involved parties and access to a central server for a distribution list of connected clients and services. The client software, written in Java, is divided into three distinct clients distinguished by their roles, is platform independent and easily distributable. It has the following functions:

- **Natural Voice Pass through between all parties connected.** The number of connected parties is limited by bandwidth, usually up to three communicate easily. Stream rates are able to be adjusted when bandwidth is limited. Currently, the Speex codec is used to allow single streams to reach usable rates as low as 20 kilobits per second. The toolkit specifications also allow for implementation of alternate codecs.

- **Voice alteration via real time effects on voice stream.** Because the toolkit provides byte level access to the audio streams, we are able to employ digital signal processing techniques to control attributes like reverb or echo.

- **Voice standardization via voice recognition then synthesis.** A controller can choose a synthetic voice to use and speak in that voice via a speech recognition process followed by a speech generation process.

- **Text To Speech with many choices for voices and options.** for example the FreeTTS package provides the voices kevin16, mbrola1, mbrola2, and mbrola3, as well the option to create voices with FestVox. Other packages, as simple as built in operating system voices, provide varying voices as well. Packages like the CSLU Toolkit, which allows tweaking of voice parameters to create new and unique voices, may also be supported through the tuning parameters feature.

- **Prerecorded audio files can be played back quickly.** this is useful for controllers who may need to following a coaching guide in order to handle the situation in the best way possible. Prerecorded files with the most frequently needed phrases, in the best voices, can aide the controller in conversing with a victim.

As shown in Fig. 2, the toolkit is used by three software clients and a central server which form the social medium testbed. Following (Riddle, Murphy, and Burke 2005), testbed assumes that there is a victim, a robot (Survivor Buddy), remote controllers (such as structural or medical specialists or family members), and a facilitator (responder) coordinating the interactions. The robot client is managed remotely, and supports simple listening and responding. The guest client provides an interface designed to allow a remote user to communicate with facilitators and the victim, at controlled intervals. The facilitator client acts as a gatekeeper between the other clients, as well as participates in the communications. The facilitator client allows management of which clients can speak with the victim and who can login to the system in general. Additionally, the facilitator client is able to decide who may access specific data on the central server. For example, the central server may collect and distribute information about the robot’s proximity to a victim. This data may be relevant and useful to a facilitator, but not to a guest. The central server software serves as a data communication mechanism between the software clients, manages services on the network, such as generation and synthesis, and maintains a directory of users and permissions for logging into the system. The server is also used to coordinate communication through firewalls, via UDP hole punching techniques. Once a client connects to the central server, it is distributed a list of other connected clients, meaning each client must only know the central server address ahead of time.

**Current Work**

The toolkit will be used in experiments starting in Fall 2010 which will explore the appropriate identity (loyalty, expression, locus of control) and type of voice (controller, synthetic for each controller, single synthetic) for the robot serving as a social medium. The expanded toolkit will also permit the association of scripts, policies, priorities, and decision-making aids with different controller roles. Code is available upon request.

**Acknowledgments**

This work was supported in part by NSF Grant IIS-0905485 “The Social Medium is the Message” and by a HRI grant from Microsoft External Research.

**References**
