

PSCR 2022 The Digital Experience

A Platform for Experimental Research in Critical Voice Communications

Jan Janak, Artiom Baloian, Kahlil Dozier,
Dan Rubenstein, Henning Schulzrinne - Columbia University
Lauren Berny, Charles Jennings - John Jay College



COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

NIST

**National Institute of
Standards and Technology**
U.S. Department of Commerce

Disclaimer

This presentation was produced by guest speaker(s) and presented for publication in the National Institute of Standards and Technology's PSCR 2022 The Digital Experience. The contents of this presentation do not necessarily reflect the views or policies of the National Institute of Standards and Technology or the U.S. Government.

This work was performed under the following financial assistance award 70NANB19H003 from U.S. Department of Commerce, National Institute of Standards and Technology.

Posted with Permission.

Outline

1. Project Objectives
2. Testbed Design
3. Remote Listening Experiments
4. Interactive Game Experiments
5. Summary & Next Steps

Project Objectives

How does the quality of the communication channel affect first responder communications?

- Four phase approach:
 - a. **Build a communication testbed**
 - i. Emulate real mission critical voice (MCV) systems
 - ii. Configurable audio & network impairments
 - b. **Experiment with trained first responders**
 - i. Communicate using the testbed in a controlled environment
 - c. **Measure communication performance**
 - i. Analyze data collected during experiments
 - d. **Build mathematical models**
 - i. Channel conditions -> performance measures

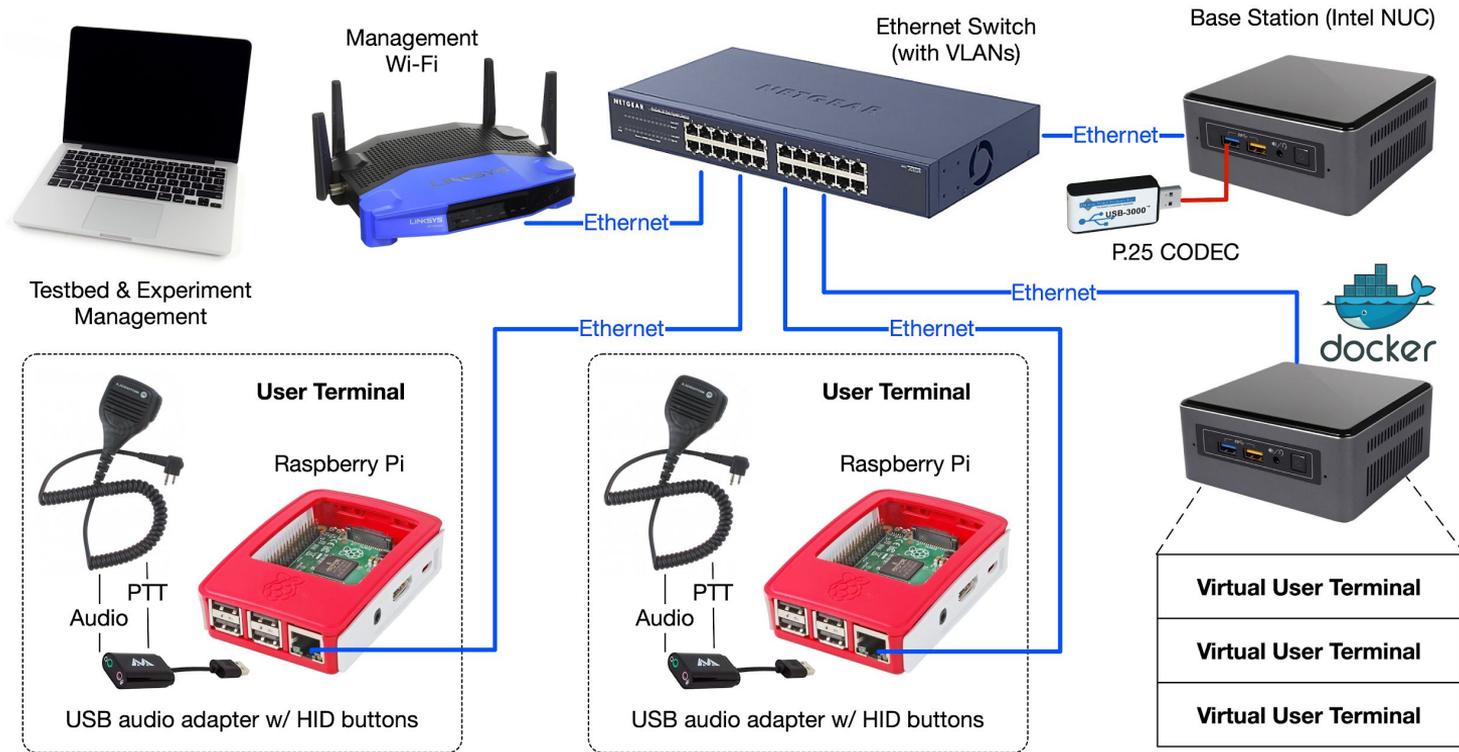
Quality of Experience (QoE) Measures

- Comprehension errors
 - repeat transmitted messages
- Task errors
 - wrong information recorded
- Usage errors
 - pressing push-to-talk (PTT) button too early or too late
- Length and latency of responses
 - pauses between requests and start of transmission
- Subjective ratings of user experience
 - rated frustration with radios

Testbed Design

- Open hardware & software
- Affordable off-the-shelf components (Raspberry Pi for user terminals)
- Emulates analog & digital MCV systems (including P.25 Phase 1 & 2)
- Programmable audio & network impairments
- Support for interactive and listening (at-home) experiments
- Testbed and experiment management from the browser
- Python environment for experiment evaluation and analysis

Testbed Hardware Architecture



Testbed Prototype at Columbia University



Testbed User Interface

The screenshot displays two side-by-side control panels for user terminals. The left panel is for 'User-Terminal-[ut1-f36342]' (Raspberry Pi 3 Model B Rev 1.2) and the right panel is for 'User-Terminal-[ut3-d13240]' (Raspberry Pi 3 Model B Rev 1.3). Each panel contains a grid of controls for various parameters:

- Peer SIP URI
- Audio CODEC (set to P.25 Phase 2)
- Microphone background (city_ambience.wav / street.wav)
- Link bit rate [bit/s]
- Audio bandwidth [Hz] (set to 4000)
- Mouth to ear delay [ms] (set to 0)
- PTT button delay distribution (Unif., Exp., Norm. radio buttons)
- Packet time [ms] (set to 20)
- PTT button delay [ms]
- Microphone volume [%]
- Speaker volume [%]
- Mic. background volume [%]
- Link error rate [%]
- Link packet loss rate [%]
- In-room audio volume [%]
- Microphone input (set to None)
- In-room audio (set to None)

Each panel includes an 'APPLY SETTINGS' button and a row of status icons at the bottom.

User Terminal Control

The screenshot shows the 'Media Files' section of the Testbed UI. It features a table with columns for Actions, Filename, Description, Date Modified, and Size [B]. A search bar is located at the top right. A large 'Audio Recordings' text overlay is present in the center of the table.

Actions	Filename	Description	Date Modified ↓	Size [B]
> [download] [edit] [delete]	harvard.wav	Harvard Sentence	3/22/2020, 5:36:33 PM	750886
> [download] [edit] [delete]	beep.wav	Motorola PTT Beep	3/22/2020, 4:50:59 PM	23064
> [download] [edit] [delete]	traffic-jam.wav	Traffic Jam	3/22/2020, 4:50:59 PM	2782172
> [download] [edit] [delete]	street.wav	Street Noise	3/22/2020, 4:50:59 PM	8769932

At the bottom right, there is an 'UPLOAD' button.

Audio Recordings

The screenshot shows the 'Experiment Management' section of the Testbed UI. It features a form for creating an experiment with fields for 'Experiment Name' (Intelligibility Evaluation) and 'Configuration'. A 'Description' field contains text about the experiment's purpose. Below the form is a 'Previous Runs' table with columns for Status, Experimenter, Subjects, Started, Ended, Notes, and Actions.

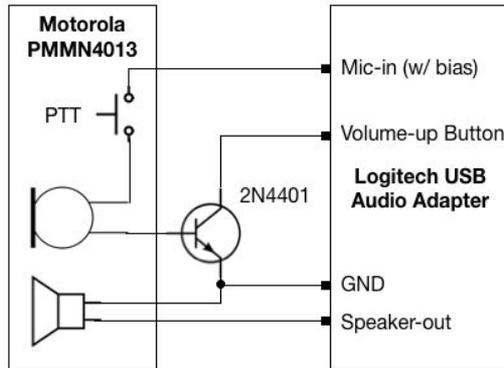
Status	Experimenter	Subjects	Started ↓	Ended	Notes	Actions
> [checkmark]	Jan Janak		5/27/2020, 1:45:11 PM	5/27/2020, 1:46:02 PM		[trash]

At the bottom right, there is a 'NEW' button.

Experiment Management

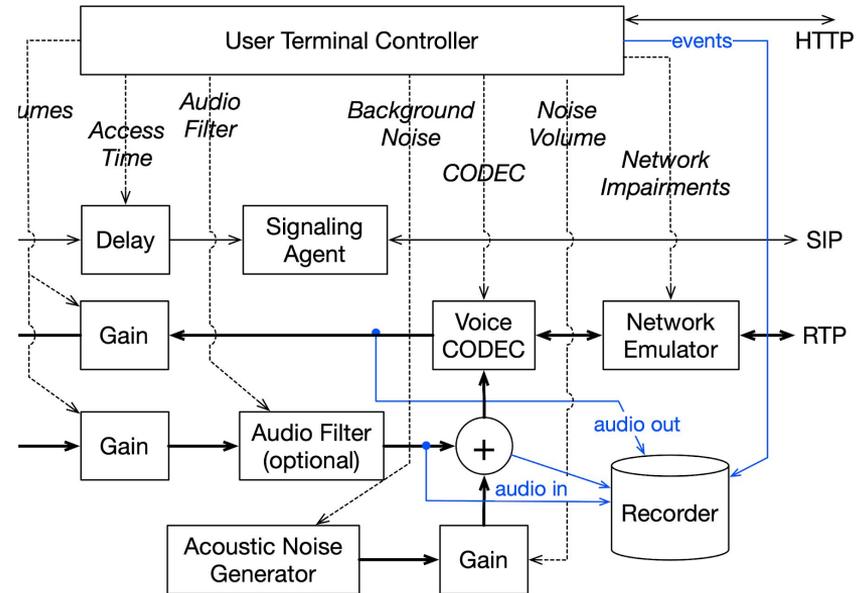
User Terminal Hardware Architecture

- Connect Motorola speaker to Raspberry Pi
- Emulates PTT communication device
- The device applies audio impairments



User Terminal Software Architecture

- Custom controller application
- PulseAudio for audio processing
- Baresip SIP user agent (P.25 codec)
- Linux kernel network emulator
- Custom event & audio recorder



Supported Impairments

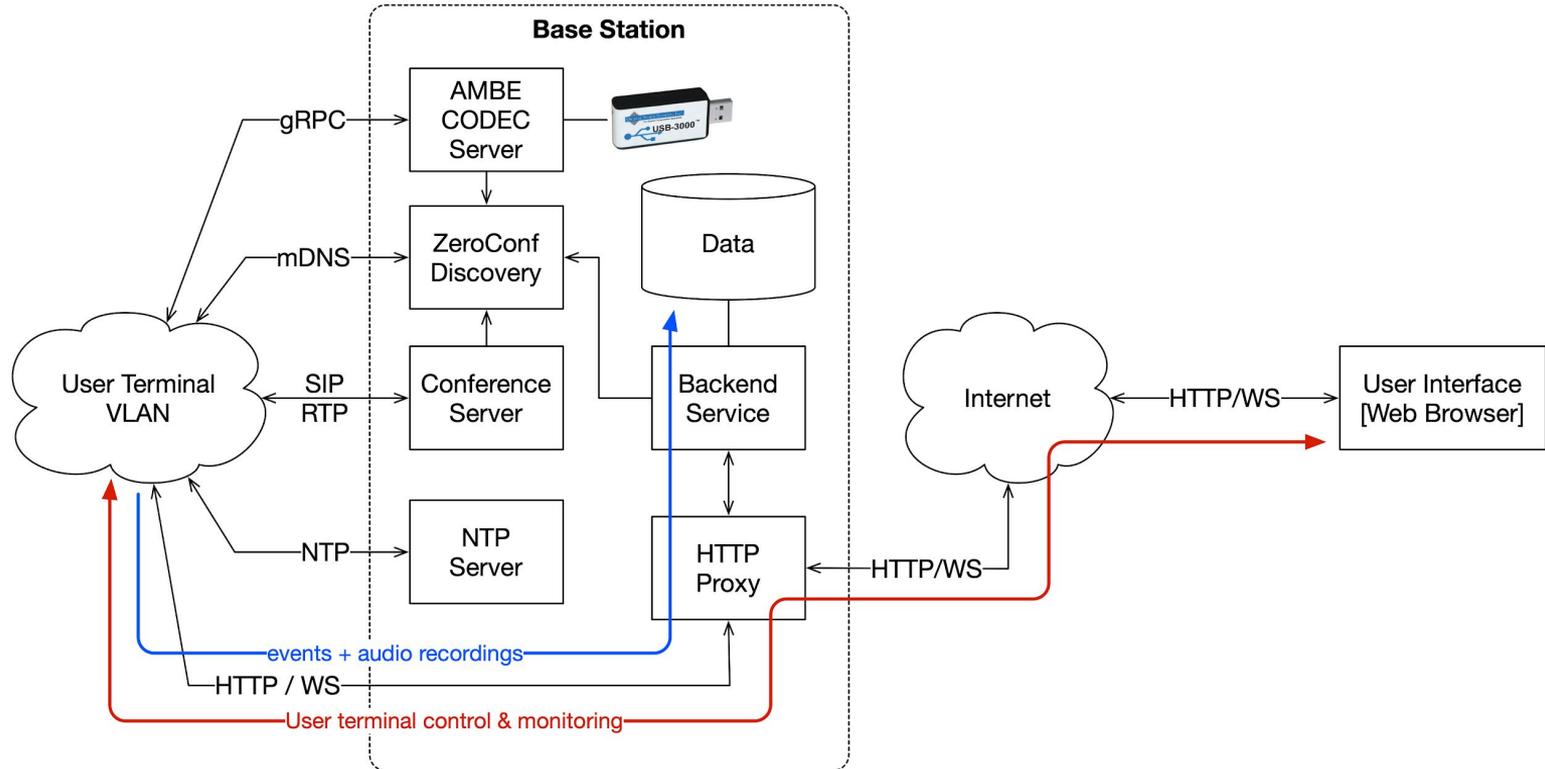
Audio

- Background noise injection
- Fine-grained volume control
- Audio filtering (bandwidth, low-pass)
- Variable PTT delay (mic audio cut-off)
- Voice compression (codec selection)

Network

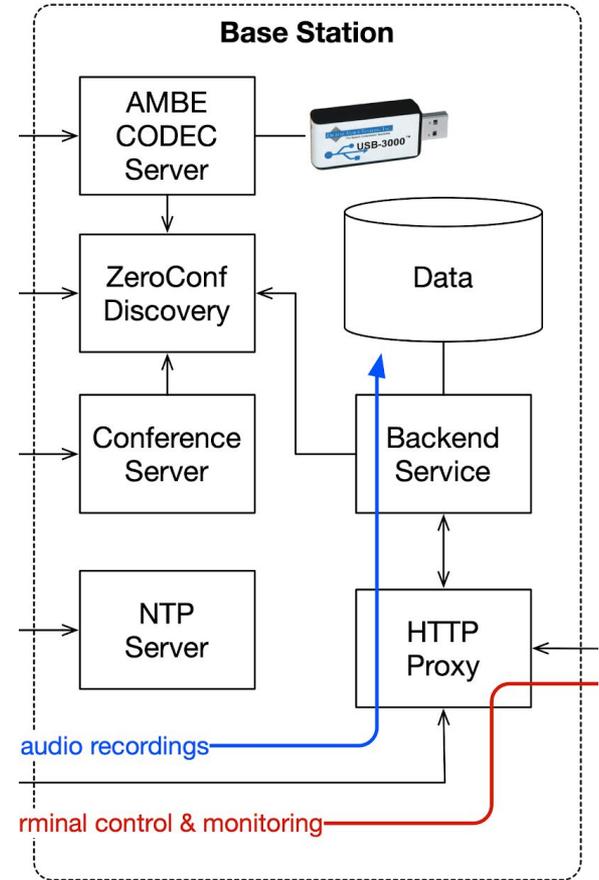
- Configurable mouth-to-ear delay
- Maximum bandwidth limitation
- Modulation rate (baud rate)
- Packet loss
- Bit error injection

Base Station Software Architecture



Base Station Services

- Custom Advanced Multi-Band Excitation (AMBE) codec server
- ZeroConf for service discovery
- SIP conference server emulates RF channels
- NTP for user terminal time synchronization
- MongoDB for data storage



Remote Listening Experiments

- A series of impaired audio recordings
- Browser UI for playback & data collection
- Anti-cheat design (play once only, no pause)
- Accessible from first responders' homes

Start Experiment "Intelligibility Evaluation" [X]

Experimenter *
Jan Janak

Test Subjects
John Doe

Please provide subjects separated by commas (optional)

Notes
Test experiment run

(optional)

Gender * Age *
Male 25

Experiment "Intelligibility Evaluation" is running [X]

Elapsed time: 00h 01m 04s
Step 2 of 3

* Please, listen carefully, identify the sound you hear, and provide your answer by selecting one of the options.

PLAY

Your answer

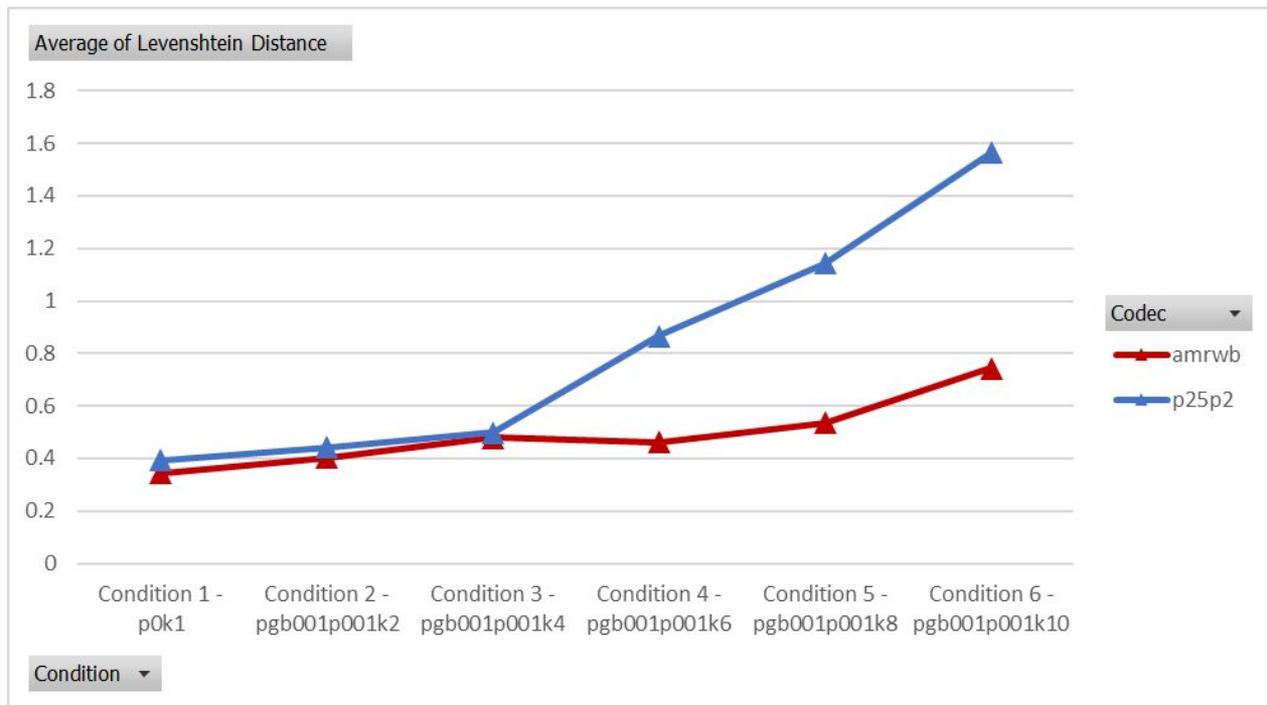
- Train station
- Train is passing by
- Subway station
- Airport
- City street

NEXT STEP

License Plate Listening Experiment

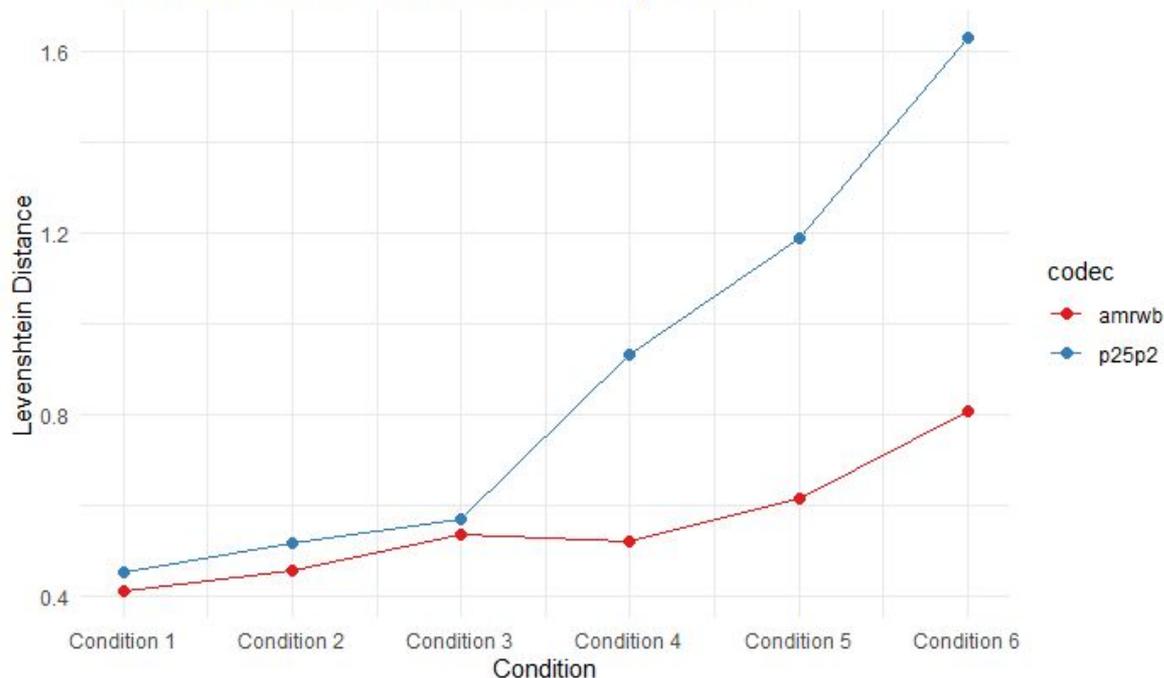
- New Jersey license plates using NATO alphabet
- 48 participants x 72 trials = 3,456 total trials
- Correlated (Gilbert-Elliot model) frame loss and bit errors
- Compare codec performance: P.25 Phase 2 versus AMR

Listening Experiment Results



Listening Experiment Results (continued)

Levenshtein Distance Across Condition By Codec



Differences Between Codecs:

Condition 1, p0k1 (reference group)

Condition 2, pgb001p001k2 ($p = .847$)

Condition 3, pgb001p001k4 ($p = .933$)

Condition 4, pgb001p001k6 ($p < .001$)

Condition 5, pgb001p001k8 ($p < .001$)

Condition 6, pgb001p001k10 ($p < .001$)

Note: Graph is predicted values

Interactive Experiment in EDGE* Environment

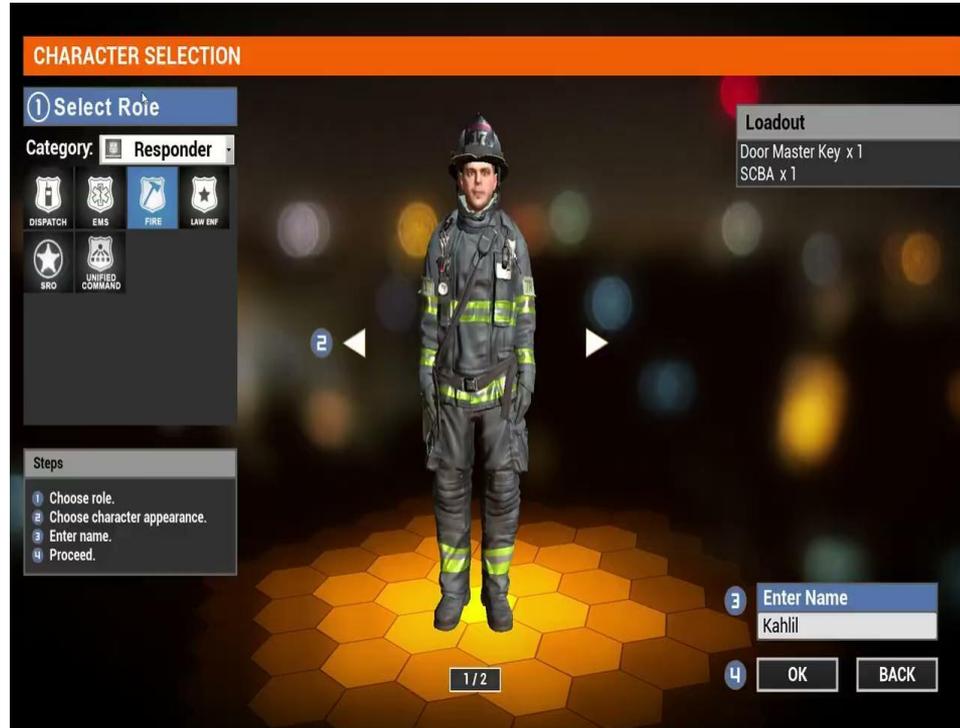
- Virtual training platform for first responders
- Coordinated response to critical incidents
- Developed by the U.S. Department of Homeland Security (DHS)
- Built on the Unreal game engine
- Two training environments: hotel and school

* Enhanced Dynamic Geo-Social Environment (EDGE)

Source: <https://www.dhs.gov/science-and-technology/EDGE>

EDGE Gameplay

<https://www.youtube.com/watch?v=nFoLQ4M2CRc>



EDGE+MCV Testbed @ Columbia

Dispatcher



User Terminal

Raspberry Pi



USB audio adapter w/ HID buttons

voice

Testbed
base station



Impairment
control

First responder (test subject)



User Terminal

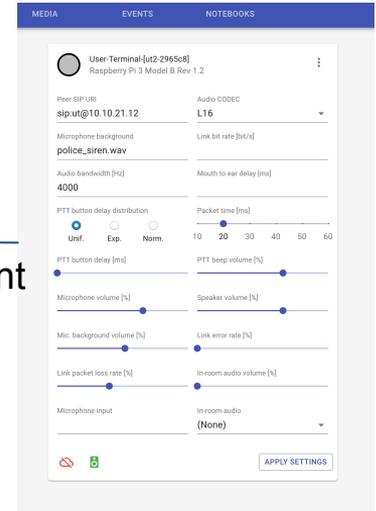
Raspberry Pi



USB audio adapter w/ HID buttons

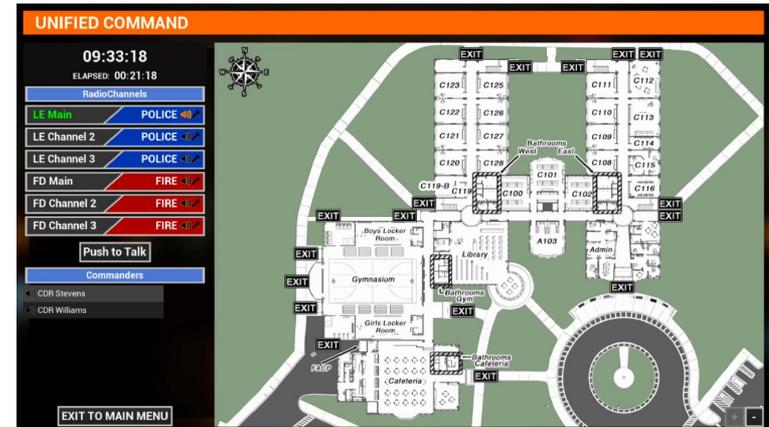
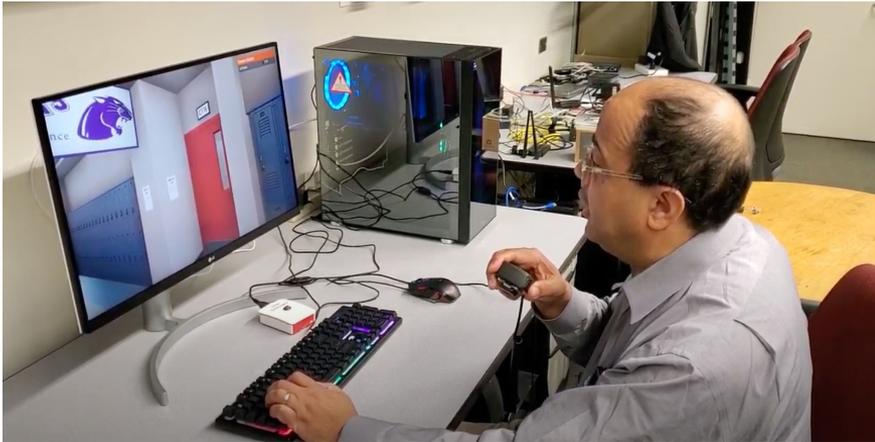
voice

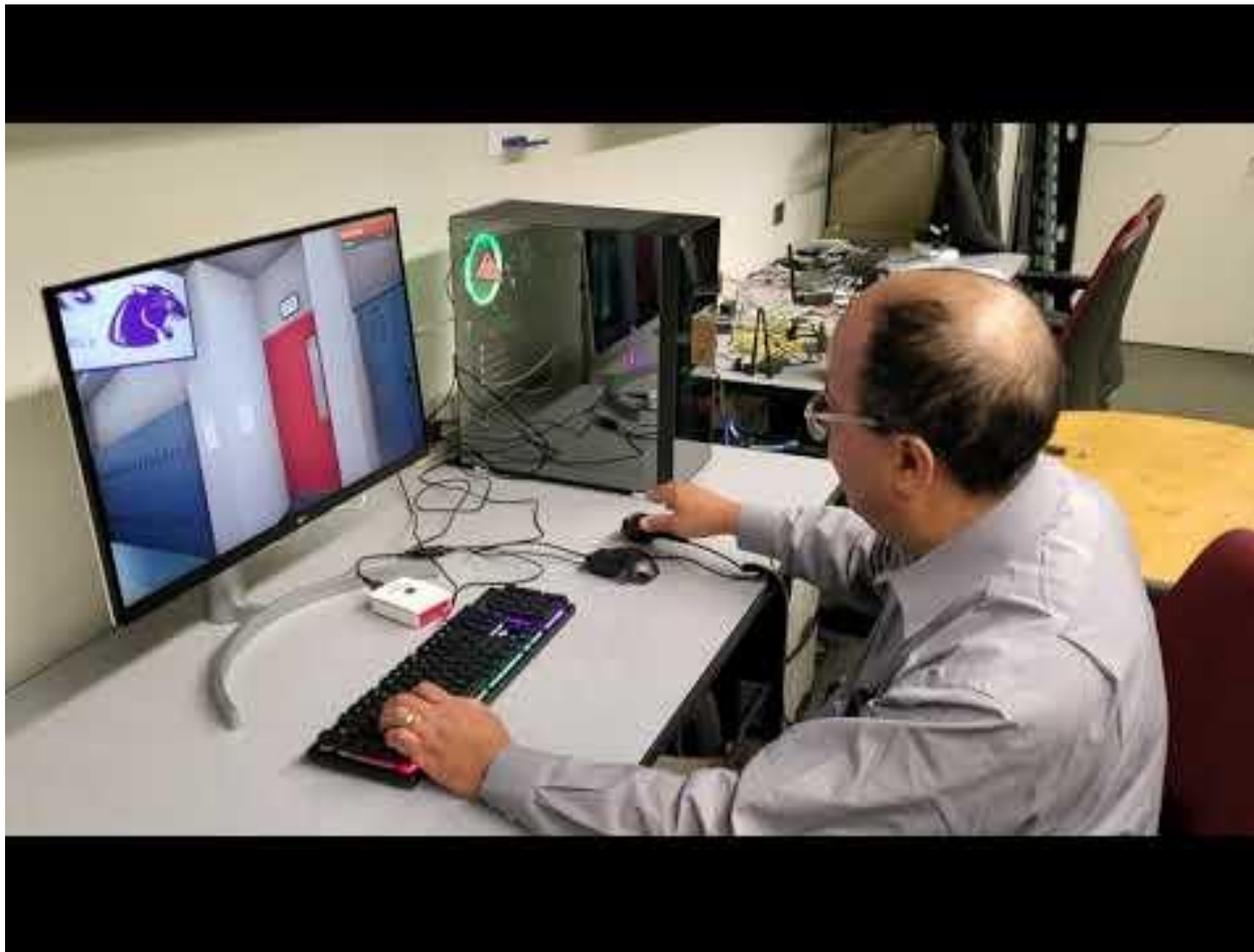
Experimenter



EDGE First Responder Experiment

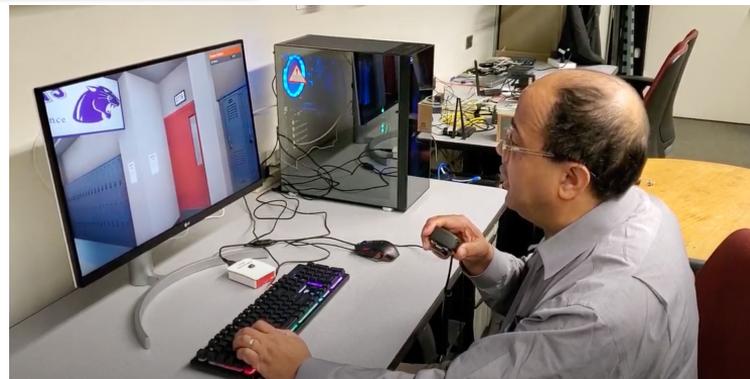
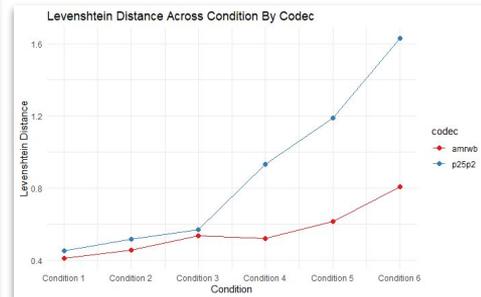
- Human test subject in EDGE
- Two-way communication with dispatcher via MCV testbed
 - Variables: MtE delay, PTT delay, access time





Summary

- Open testbed for experiments with MCV
 - Remote listening experiments
 - Interactive delay-based experiments
- Ongoing human subject experiments
 - Listening experiment performed & evaluated
 - Interactive (EDGE) experiment underway



<https://irtlab.gitlab.io/mcv-testbed>

Next Steps

1. Run experiments with first responders on Columbia University campus
2. Measure true Mouth-to-Ear (MtE) delay of the testbed
3. Develop mathematical models to estimate performance measures from channel condition (impairment levels)

Additional Resources

- Project website is online
- Testbed source code available under the MIT license
- Free software developed for the project:
 - PulseAudio client for NodeJS: <https://github.com/janakj/pulseaudio.js>
 - Client library for DVSI AMBE vocoder chips: <https://github.com/janakj/ambe>

Project Website

<https://irtlab.gitlab.io/mcv-testbed>

A Platform for Experimental Research in Critical Voice Communications

The Land Mobile Radio (LMR) emulator is an open platform for conducting experiments with Mission Critical Voice (MCV) communication systems. MCV systems are employed by most public safety organizations such as law enforcement, emergency medical services, fire, search and rescue, military, and by selected private organization (taxi, logistics, and traffic control). A variety of environmental factors and communication system properties can impair intelligibility, i.e., the comprehensibility of transmitted speech in given conditions. The LMR emulator project provides components that can be used to build a model of a MCV communication system in a controlled environment (lab), plan and execute experiments involving human subjects, and systematically evaluate the influence of various environmental and system parameters on intelligibility.

The emulator source code is released under a permissive software license and is designed to work with commercial off-the-shelf hardware components. Various push-to-talk (PTT) and full-duplex communication scenarios can be implemented. Key parameters that influence intelligibility are run-time configurable including background noise, audio levels, speech codes, channel acquisition time, packet loss, and error rate. The emulator is extensible and provides straightforward HTTP APIs for experiment analysis and integration with third-party services and applications.

Key Features

- A web-based software architecture that provides:
 - Browser-based user interface for experiment design, execution (programmable & interactive), and evaluation
 - Remote user terminal control, with access to audio streams in real-time from the browser
 - Generation of experimental data on user terminals (time-synchronized events, audio recordings) and collection of the data on the base station
- HTTP APIs for experimental data access, analysis, evaluation, and integration with third-party tools
- Flexible hardware architecture:
 - User terminals running on Raspberry Pi or virtualized in Docker containers
 - Ethernet or Wi-Fi networking with link emulation independently configurable on each user terminal (bandwidth, bitrate, delay, loss, error injection)
 - Support for DVSI's hardware AMBE codecs
- Software Defined Radio (SDR) gateway for communication with real radios (experimental, analog FM only)
- Programmable audio processing architecture in user terminals:
 - Low-latency audio processing (down to 10 ms)
 - Push-to-talk (PTT) and full-duplex communication support, configurable PTT steps
 - Background noise injection and mixing, input/output redirection and recording
 - Fine-grained volume control for each audio stream with dB steps
 - Supported codecs include: IMBE & AMBE ([Project 25](#)), GSM & AMR ([Etsi/3gpp: TETRA](#)), G.711, Opus, [CODEC2](#) (experimental)

System Overview

The testbed consists of a base station (Intel NUC or similar platform) and a collection of user terminals (Raspberry Pi with speaker-microphones), all connected with a VLAN-capable Ethernet network. The testbed and experiments are managed from a laptop connected to the base station via a dedicated management Wi-Fi network. The base station manages all user terminals involved in the experiment and collects experimental data (events and audio recordings) from the terminals after the experiment has ended. The data is then accessible via a RESTful API.