DEVELOPMENT OF A LOW-COST MOBILE EMBEDDED HANDHELD AIRCASTING DEVICE

Iem Heng, NYC College of Technology; Andy S. Zhang, NYC College of Technology; Michael Heimbinder, HabitatMap, Inc.; Pawel Obrok, Lunar Logic Polska LLP

Abstract

Handheld AirCasting devices represent an environmental sensing platform that allows participants, known as AirCasters, to record, broadcast and map air status updates in real-time using their Android smartphones and/or tablets. Initial development efforts in this study focused on recording variability in sound levels, temperature, humidity and carbon monoxide. The data collected and annotated by the AirCaster was uploaded to an interactive web-based map that displays both individual and aggregated routing information. Each AirCasting session allowed the user to capture real-world measurements, annotate the data to tell their story, and share the data with local and world-wide communities via the CrowdMap. The AirCasting platform also employs color-changing clothing accessories to communicate the status of the air in a particular place and time to both the AirCaster and people within the immediate vicinity.

Introduction

Air pollution in the U.S. is estimated to cause in excess of $78 billion in damages annually [1]. Of primary concerns are the human health effects of air pollution, including premature mortality and chronic illnesses such as bronchitis and asthma. Despite the tremendous economic costs and pervasive negative health impacts of bad air, air pollution often goes unnoticed because it is largely invisible. Much of what happens in our immediate environment passes without note despite the contribution that events such as recording and crowdsourcing might have on our understanding of us and our communities. AirCasting captures a spectrum of that lost reality and returns it to us as data that is consistent in its units of measurement and, therefore, easily meshed with data from other AirCasters. By making it possible for AirCasters to annotate specific environmental events in time and space, the authors supplemented the qualitative information reported through conscious human experience with the quantitative information from sensing handheld devices that observe and record aspects of our environment that are either impossible to perceive directly (e.g., pollutant gas concentrations) or difficult to quantify and communicate in a consistent manner (e.g., sound levels). AirCasting allows individuals to broadcast what is happening with their environment, crowdsource their own information with that from other AirCasters, and identify patterns and commonalities. Thus, this AirCasting device makes air pollution visible, thereby empowering communities advocating for healthy environments.

Unlike commercially available off-the-shelf chemical detection devices, the AirCasting device, with its novel design and small size, could potentially provide an advanced warning of impending danger. Comparing the AirCasting device to two commercially available off-the-shelf hazardous-vapor warning devices—the LCD 3.3 [2] and the Nose Gas Sensor [3]—the AirCasting's Bluetooth modem allows it to communicate with the world-wide web by connecting to devices such as smartphones or tablets; whereas, the LCD 3.3 and Nose Gas Sensor have no way of connecting to the web. Additionally, neither the LCD 3.3 nor the Nose Gas Sensor is small or within the price-range of ordinary consumers. This makes the AirCasting device an improvement over currently available devices for use in a chemical detection or early warning system. Also, the AirCasting device is a low-cost environmental monitoring system that ordinary consumers can afford. The current custom-made AirCasting prototype costs about $130-$150 per unit to produce; whereas, current off-the-shelf monitoring devices start around $2,000 per unit.

Microcontroller

Handheld AirCasting devices include a microcontroller. In general, a microcontroller is a small computer on a single integrated circuit containing a processor core, memory and programmable input/output (I/O) peripherals. There are many types of microcontrollers that are available on the market—PIC 18, ATMega 32, Arduino, Netduino, 68HC12, BASIC Stamp, VEX Cortex Microcontroller, NXT Microcontroller and the BasicATOM 28X, among others. In this study, an Arduino UNO microcontroller was used for this custom-designed handheld device due to its low-cost. Figure 1 illustrates the physical Arduino UNO microcontroller and 3D Arduino hardware schematic layout. The 3D Arduino hardware schematic was created using the Fritzing software program [4]. The Fritzing software program is a freeware program that is capable of making 2D and 3D schematic diagrams.

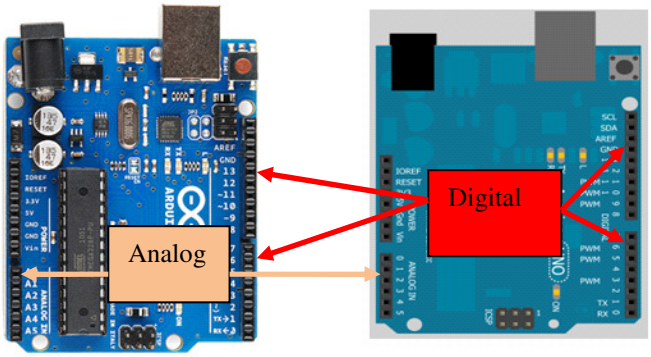

Figure 1. Arduino UNO Microcontroller and 3D Arduino Schematic Layout

The Arduino has two power pins: one for 5V and the other for 3.3V. In addition, it has a Vin pin which is mostly from the input of the power supply, and it is around 9V. Further, the Arduino has two ground pins along with a reset pin.

The Arduino UNO has I/O pins labeled on the right side of the microcontroller (see Figure 1). There are 14 digital pins, and of the 14 pins of which 6 provide the pulse width modulation (PWM) output. PWM can be used as digital or as PWM itself. PWM pins act like a variable analog output that can be outputted. Standard digital inputs and outputs, on the other hand, mean simply turning on/off an LED. In other words, digital means on or off, yes or no. These pins not only serve as an outputs but inputs as well.

The Arduino UNO also has 6 analog input pins labeled on the left side of the microcontroller (see Figure 1). The analog inputs are generally used for sensors with analog outputs such as temperature, humidity or gas (CO, CO_2, NO, NO_2, etc.). The Arduino UNO has one set of communication pins (TX/RX). Communication pins are used for serial inputs and outputs for wireless communication between devices. Some examples of devices that use these communication pins are Bluetooth transceivers or ZigBee transceivers, which are useful and practical in educational capstone projects. Heng et al. provide some examples of capstone projects [5].

Sensor Calibrations

The low-cost mobile embedded handheld AirCasting device has three different types of sensors: A temperature sensor (LM335A), a humidity sensor (HIH-4030) and a carbon monoxide (CO) gas sensor. The signal data acquisition from the three sensors is raw analog data. The raw data must be calibrated with respect to that sensor. For instance, the calibration for the raw analog value of the LM335A temperature sensor would be as follows:

```
float Kelv = (((analogRead(A2) / 1023.00) *5) *100;
// Kelv = Kelvin
float Cel = Kelv – 273.15; // Cel = Celsius
float Fah = (9.00/5)*Cel + 32; // Fah = Fahrenheit
```

Note that the analogRead( ) function reads the voltage applied to one of the analog pins from the Arduino UNO in Figure 1. In this case, the analog pin is A2. This analogRead ( ) function returns a number between 0 and 1023 [6], which represents voltages between 0 and 5 volts (V). Also, the factor of 100 in the Kelvin equation comes from the fact that the LM335A works like a Zener diode with a breakdown voltage proportional to the absolute temperature at 10mV/°K [7].

Unlike the calibration of the LM335A, the Honeywell corporation provides a chart [8] on output voltage versus relative humidity, as shown in Figure 2.

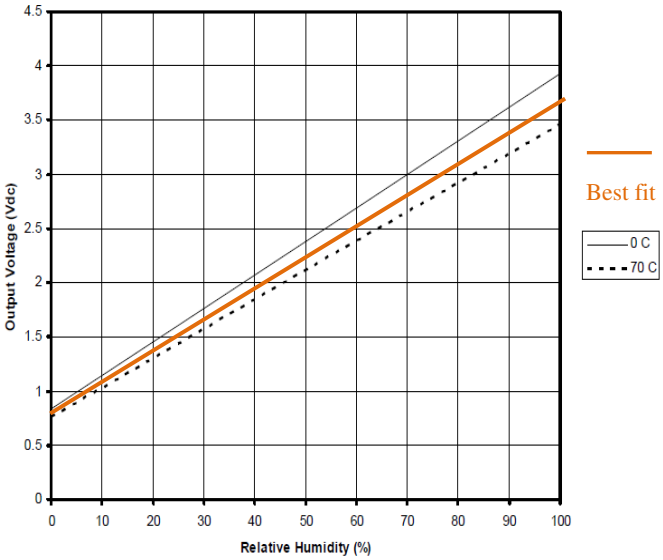

Figure 2. Output Voltage versus Relative Humidity

Based on Figure 2, the best linear fit and linear equation can be used to calibrate the relative humidity value from the raw analog value.

```
float MaxV = (3.27-(0.006706*Cel));
float RelativeHumi = ((((analogRead(A0)/1023.00)*5)-
0.8)/MaxV)*100;
```

Note that the maximum voltage (MaxV) value drops to 0.006706 for each degree Celsius over 0°C. The voltage at

0°C is 3.27. This is corrected for zero percent voltage offset, which is approximately 0.8.

Similar to the calibration of relative humidity, the calibration for carbon monoxide (CO) is based on the sensitivity characteristics of the gas concentration chart. For instance, the gas concentration chart [9] in Figure 3 is used as part of the calibration for the CO concentration in PPM (parts per million).

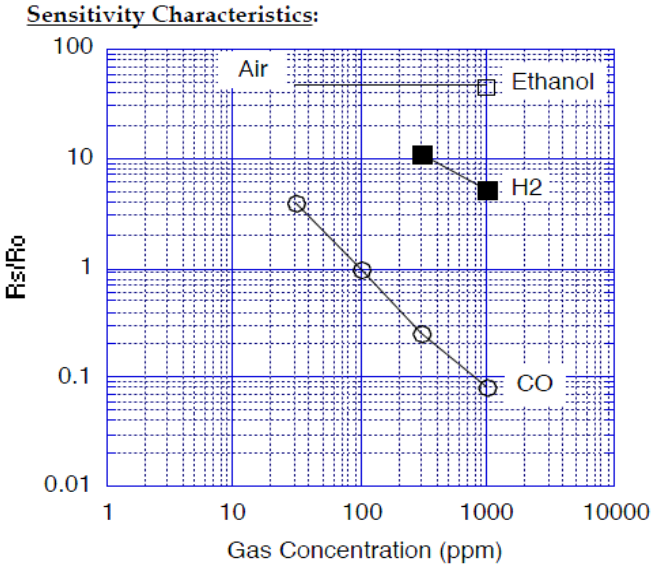

Figure 3. CO Concentration in PPM

The Y-axis in Figure 3 indicates the sensor resistance ratio, Rs/Ro, which is defined as follows:

Rs = Sensor resistance of displayed gases at various concentrations

Ro = Sensor resistance in 100 PPM CO

Figure 3 represents the typical sensitivity characteristics of the CO concentration level which increases as the sensor resistance decreases. As the value of the sensor resistor, Rs, decreases, the voltage across Rs decreases because the fixed resistor and the sensor resistor are connected in series. This is illustrated in Figure 4. Another way of looking at Ro in Figure 3 is the level of exposed gas to the sensor in a confined space. For instance, if 100 PPM were poured into the container in a confined space, what would the Rs sensor read? It may read 98 PPM or 102 PPM. Using the voltage-divider concept, the formula for defining the sensor is as follows:

$$Rs = \frac{Vc \times R_L}{Vout} - R_L \qquad (1)$$

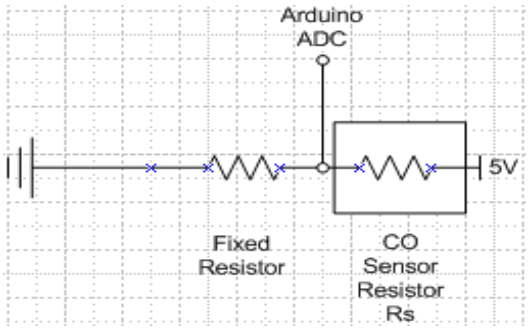

Figure 4. CO Sensor Resistor

From Equation (1), Vc is the voltage input, 5 Volts, from the Arduino UNO microcontroller. R_L is the load resistance (in this case, 39kΩ was used) that was connected to the CO gas sensor. Vout is a voltage signal from the CO gas sensor, which varied depending on the PPM CO concentration. The value of Rs in Equation (1) changes according to the amount of CO gas present—as seen in Figure 3; the typical detection range for the CO gas sensor is 30 to 1000 PPM. If the resistance value of Rs is the same as Ro, then 100/100 = 1, which correlates to 100 PPM in Figure 3. In theory, Ro represents the X axis in Figure 3, if conditions are perfect.

Hardware Schematic of the Handheld AirCasting Device

Using the Fritzing software program, the overall schematic layout for the handheld AirCasting device was designed, as illustrated in Figure 5.

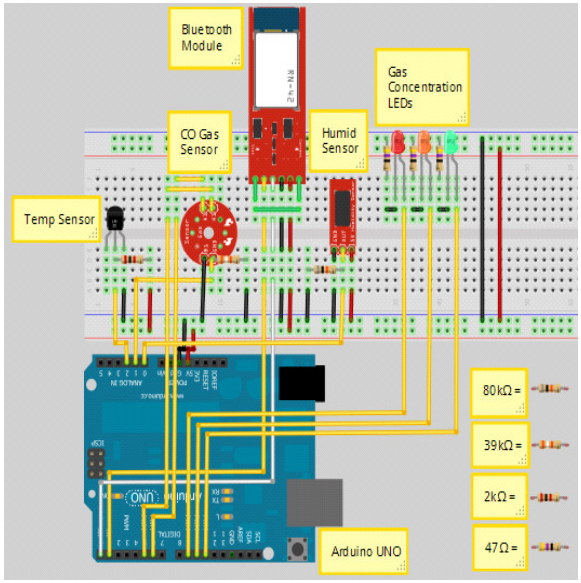

Figure 5. Schematic Layout of the Handheld AirCasting Device

From Figure 5, the yellow, red, and black wires are used as data signal communication, voltage and ground, respectively. The signal lines for the three sensors (temperature, CO gas and humidity) connect to the Arduino analog pins A2, A1 and A0. The red and black lines are connected to the Arduino 5V and ground pins, respectively. The yellow and white wires from the Bluetooth module are used as the transceivers and are connected to Arduino pins TX (transmitter – yellow wire) and RX (receiver – white wire). The use of a Bluetooth module provides the wireless communication lines between the AirCasting device and a smartphone or tablet. Also, the three color LEDs indicate the level of CO gas concentration. The green, orange and red LEDs indicate the least, medium and highest CO concentrations, respectively, making the AirCasting a unique mobile handheld device.

Prototyping of the Handheld AirCasting Device

Based on the hardware schematic layout of the handheld AirCasting device in Figure 5, the physical prototytpe for this device was made. Figure 6 shows the progress stages in designing the AirCasting device.

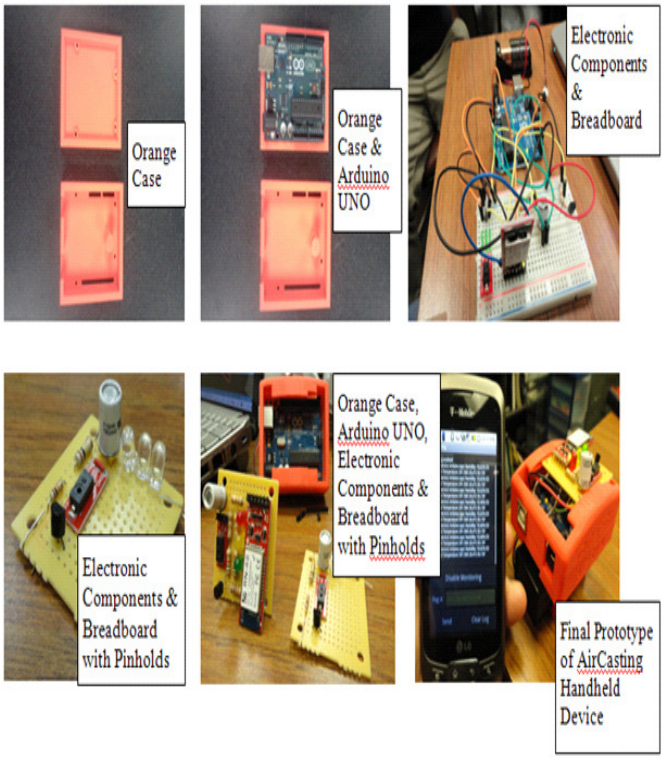

Figure 6. Design Stages of the Handheld AirCasting Device

The custom-made AirCasting device cover (orange case) is used for holding the Arduino UNO microcontroller and electronic components (sensors, resistors, LEDs and Bluetooth) along with the breadboard. First, a computer model of the AirCasting device was created using Autodesk Inventor software. Figure 7 is a computer rendering of the device. Then, a phyical prototype was made using a 3D rapid prototyping machine, as shown in Figure 7. The electronic components were soldered on the back side of the breadboard. After all of the electronic components were soldered on the breadboard, the completed breadboard was mounted in the orange case. This completes the design stages of the AirCasting prototype.

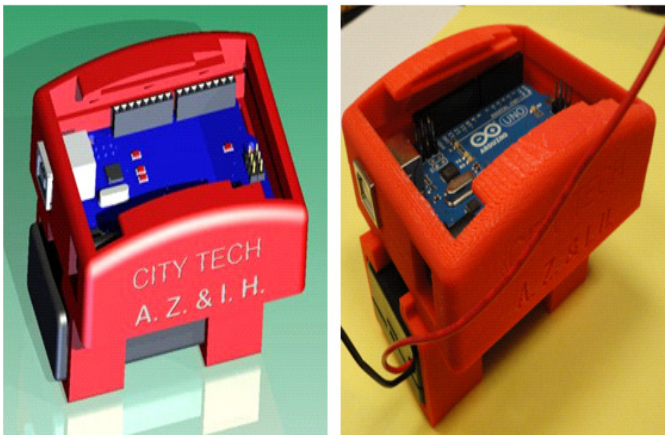

Figure 7. Computer Rendering and Physical Prototype of the Handheld AirCasting Device

Software Design of the Handheld AirCasting Device

Following is a discussion of how the software program communicates and interfaces between the AirCasting device and mobile smartphones or tablets. The source code was written in Arduino Sketch to communicate and interface with the electronic components, such as the temperature sensor, humidity sensor, CO gas sensor and LEDs, as shown in Figure 6. Upon the success of interfacing with the sensors and LEDs in Arduino Sketch, the Android library MeetAndroid was imported into the Arduino library folder so that the data acquisition could be sent from the Arduino Serial Monitor to the Amarino Application (App) progam. The Android library MeetAndroid is part of the Amarino driver device that must be imported into the library folder of Arduino Sketch. The Amarino progam [10] is a freeware program that incorporates a plug-in mechanism which allows programmers and developers to integrate their events into Amarino. This, then, provides a gateway to

communicate with smartphones or tablets based on the Android open-source operating system. Figure 8 illustrates the details of communication between Arduino Sketch, the Amarino App, AirCasting App and Android operating system.

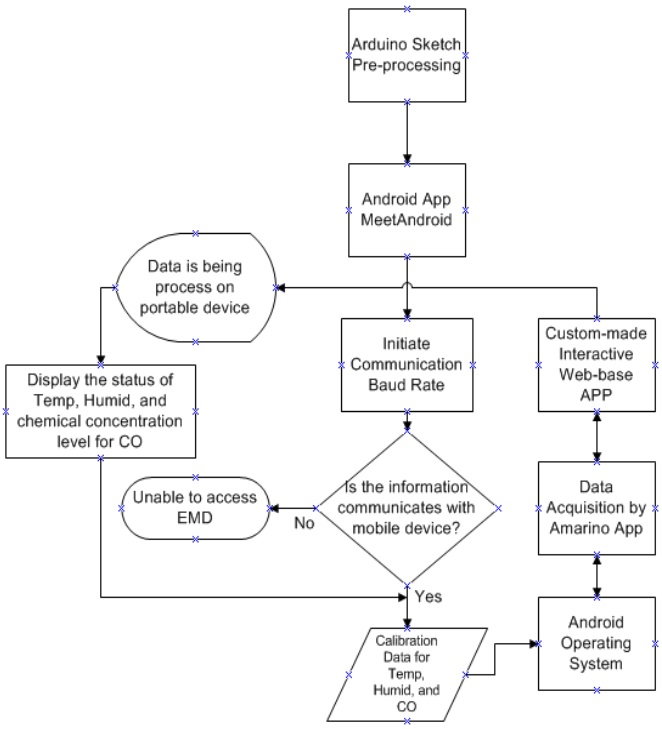

Figure 8. Flowchart of the Handheld AirCasting Device

Since the Amarino program is capable of acquiring the data acquisition from the AirCasting device through Arduino, the AirCasting App was developed to retrieve the data from Amarino App. This AirCasting App is an interactive web-based map that displays both individual and aggregated routing information on sound levels, temperature, humidity and carbon monoxide. It is written in the high-level Java and XML programming languages. The source codes of the AirCasting App are too long to be included here. Instead, the block diagram of the AirCasting App is introduced in Figure 9.

The system in Figure 9 is composed of two main parts: an Android App and a Ruby-on-Rails [11] backend. The main purpose of the Android App is to gather environmental data which is subsequently stored in the backend. It also allows the user to view data gathered and stored in the backend. The web application's purpose is twofold: it exposes an API which the Android App can use to submit data, and has a web user interface for a more comfortable way to browse the data.

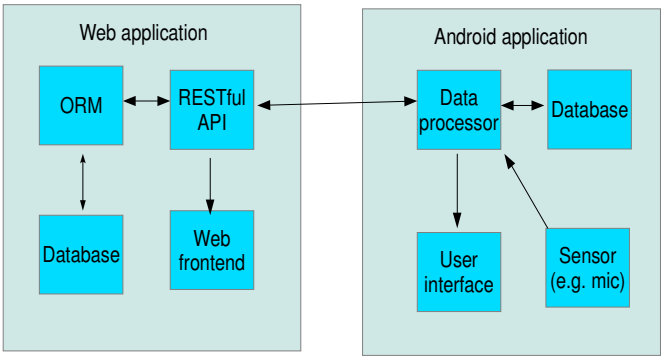

Figure 9. Block Diagram of the AirCasting Application

Android Application

The Android App (see Figure 9) accesses an environmental sensor connected to the phone (thus far, only the built-in microphone) and stores measurements obtained from that sensor (in the case of a microphone, sound volume levels). The measurements are annotated with the time and location where they are taken and form a sequence called a session. The users can add their own notes (to sessions) to indicate that some special events have occurred while measuring. Sessions are stored in a database on the phone so that the user is quickly able to access their data. Using the API, the application will also upload sessions to the backend when network connectivity is available. The application's user interface allows the user to view both their local data and aggregated data obtained from the web application's API.

Web Application

The most important part of the web application is the API it exposes. A RESTful HTTP approach [12] was used in this study and all data were transmitted as JSON [13] for maximum simplicity. The most important functionalities of the API are:

- Uploading sessions
- Accessing raw session data - this is used, for example, when the user is viewing a single session
- Accessing aggregated data - this is used to generate a "crowd map" represented with color-coded averaged data overlaid on top of a map

The API uses an Object Relational Mapping (ORM) [14] layer to write and read data from a database; thanks to this, the sessions from various users are kept in one place and aggregate data may be calculated. The web interface is functionally very similar to the user interface available in the Android App; it uses the data exposed by the API and renders it for the user. The complete source codes for the AirCasting App are available on the following AirCasting

website: http://aircasting.org. The sample of the AirCasting App for sound levels can be seen in Figure 10.

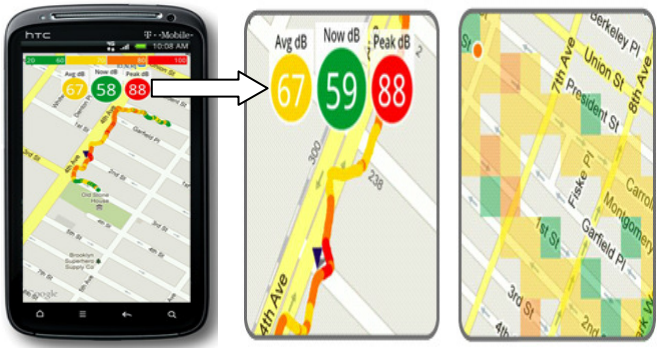

Figure 10. AirCasting App for Sound Levels

AirCasting Device: Testing and Results

Several tests of the AirCasting device were performed at different locations to record temperature, humidity and CO. The test was done by using a laptop, a Xoom tablet and a TMobile Android smartphone. The preliminary results, an average of the tests, are shown in Table 1.

Table 1. Testing and Results

Average	Temp (F)	Humid (%)	CO (PPM)
Indoor Tests	80.5	18.3	13.7
Outdoor Tests	42.3	31.5	8.1
Inside Passenger's Car	57.3	22.1	22.2
Behind Car's Exhaust Pipe	85.7	107.6	81.5

Several tests were conducted and hundreds of data points were recorded on four different scenarios by using a laptop, a tablet and a smartphone. Each scenario was then averaged, as displayed in Table 1. Of the four different scenarios, the worst case was placing the AirCasting device behind the car's exhaust pipe while the car engine was running. In this case, the average temperature, humidity and CO were approximately 85.7°F, 107.6% and 81.5 PPM, respectively. Note that the wind factor was about 7 to 10 miles per hour while these tests were being performed outside. This is why the results of the temperature, humidity and CO were a bit low due to the wind factor.

Conclusion

The handheld AirCasting device is about crowdsourcing cybernetic systems. Seemingly, every day brings to market a new portable device that is capable of augmenting human sensory experience by tapping into phenomena that are beyond the limits of human perception. On any given day, the air may be considered bad in a particular place; or our asthma may be flaring up. But sensing and communicating the quality of the air or the state of our bodies in precise and universal terms only becomes possible with the aid of scientific instruments. Previously, these instruments either did not exist or were too expensive or bulky to be purchased and carried on or in the human body. Currently, though, portable environmental sensors, tracking devices and biomonitors are rapidly becoming ubiquitous. The output from these cybernetic systems is innately social as data-basing and crowdsourcing are fundamental to the way they record information and evolve through technological iterations. By creating a platform that crowdsources data from thousands or potentially millions of cybernetic devices, a handheld AirCasting device becomes an analytics engine capable of picking out emergent patterns in human environments and biology.

Acknowledgements

This research was partially supported by a grant from the New York Hall of Science (NYSCI) and by a grant from the National Science Foundation (NSF ATE No 1003712). The authors appreciate greatly the support from the New York Hall of Science (NYSCI) and NSF. In addition, the authors would like to thank Raymond Yap (former City Tech student) who was involved in making the AirCasting device.

References

[1] U.S. Census Bureau: 2000 Census Data (SF1, SF3). (n.d.). Retrieved November 25, 2011, from http://factfinder2.census.gov/faces/nav/jsf/pages/index.xhtm

[2] Phipson, S. (n.d.). *LCD 3.3*. Retrieved May 23, 2011, from http://www.smithsdetection.com/1025_4601.ph

[3] Lin, H., & Suslick, K. (2010). A Colorimetric Sensor Array for Detection of Triacetone Triperoxide Vapor. *Journal of American Chemical Society, 44*, 15519-15521.

[4] Pardue, J. (n.d.). Fritzing with the Arduino. Retrieved December 12, 2011, from http://www.nutsvolts.com/index.php?/magazine/article/august2012_SmileysWorkshop

[5] Heng, I., Zia, F., & Zhang, A. (2011). *Wired and Wireless Port Communication. In Proceedings of The 118th Annual ASEE Conference & Exposition,* Paper #AC 2011-82, (pp. 1-17). Vancouver, British Columbia, Canada.

[6] Banzi, M. (2009). *Getting Started with Arduino*. (1st Ed.). Sebastopol, California: O'Reilly Media, Inc.

[7] McRoberts, M. (2010). *Beginning Arduino*. (1st Ed.). New York City, New York: Apress.

[8] Khardey, E. (n.d.). LM135/LM235/LM335, LM135A/LM235A/LM335A Precision Temperature Sensors. Retrieved December 18, 2011, from http://www.ti.com/product/lm335

[9] Taylor, J. (n.d.). TGS - for the detection of Carbon Monoxide. Retrieved December 18, 2011, from http://www.figarosensor.com/products/2442pdf.pd

[10] Kaufmann, B., & Buechley, L. (2010). Amarino: a Toolkit for the Rapid Prototyping of Mobile Ubiquitous Computing. *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2010).* ACM, New York, NY, USA, 291-298.

[11] Ruby, S. (2010). *Agile Web Development with Rails.* (4th Ed.). Sebastopol, California: O'Reilly Media, Inc.

[12] Richardson, L., & Ruby S. (2007). *RESTful Web Service*. (1st Ed.). Sebastopol, California: O'Reilly Media, Inc.

[13] Crockford, D. (n.d.). Introducing JSON. Retrieved January 20, 2012, from http://www.json.org

[14] King, G., Bauer, C., Andersen, M., Bernard, E., & Ebersole, S. (n.d.). HIBERNATE - Relational Persistence for Idiomatic Java. Retrieved January 20, 2012, from http://www.hibernate.org/about/org

Biographies

DR. IEM HENG earned his bachelor's degree from Providence College (Providence, RI) with double majors in Pre-Engineering Program and mathematics. In addition, he earned another bachelor's degree from Columbia University (New York, NY) in mechanical engineering and master's in applied mathematics from Western Michigan University (Kalamazoo, MI); his Ph.D. in computational and applied mathematics from Old Dominion University (Norfolk, VA). Before joining the EMT/CET department at City Tech in fall of 2007, he was a faculty member and chair of the CET department at DeVry Institute of Technology (Long Island City, NY). He worked as a researcher for NASA Langley Base in Hampton, VA, for two years. His research activities include embedded systems, robotics, mechatronics, software development for embedded systems with real time simulation, real time gaming simulation programming, and web application programming.

DR. ANDY S. ZHANG earned his Master's in mechanical engineering from the City College of New York in 1987 and his Ph.D. in mechanical engineering from the Graduate Center of the City University of New York in 1995. Prior joining the Mechanical Engineering Technology department at City Tech in 2000, he served as an engineering instructor for the JUMP, an engineering training program sponsored by the New York State Department of Transportation. Professor Zhang's research area includes materials testing, composite materials, CAD/CAE, robotics and mechatronics, and engineering technology education.

MICHAEL HEIMBINDER is a community organizer, educator, and information designer. HabitatMap's community mapping and social networking platform is a direct testament to Michael's dedication to creating online platforms that engage the public and facilitate the sharing of news and information using geographic information systems. Since launching HabitatMap in 2006 he has worked with dozens of community based organizations and schools to create planning and advocacy maps that publicize the issues they care about most. In addition to running HabitatMap, Michael is Vice-Chair and Project Manager of the Newtown Creek Alliance, where he has made community knowledge sharing the keystone of the organization's successful efforts to clean up the Creek and improve quality of life in the surrounding neighborhoods. He is also a member of the New York State Environmental Justice Interagency Task Force Mapping Work Group and an advisor to the Organization of Waterfront Neighborhoods where he consults on solid waste management issues in New York City. Michael is a graduate of Colorado College and received his M.A. in International Affairs from the New School for Social Research.

PAWEL OBROK works for Lunar Logic Polska as part of web development team. He works on the programming aspect of AirCasting App and website to receive and display air quality data.