Voice Localization Development Log: the 2nd

Testing sensor quality and viability of planned method

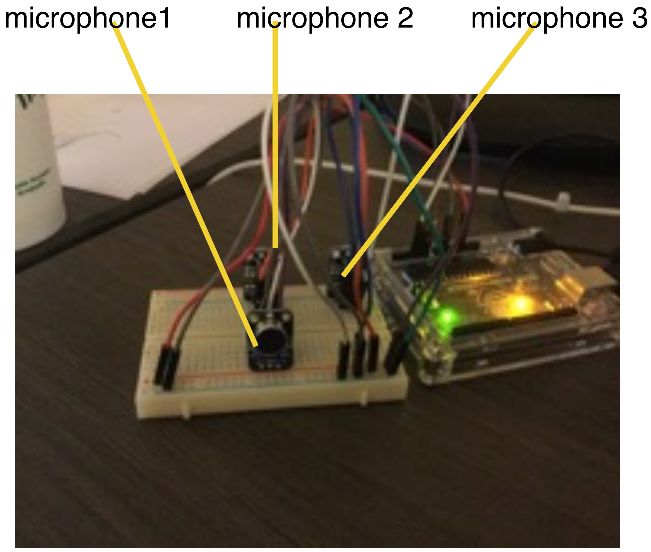
Wenxuan Tang
12/6/2015

In the log, we tested sound strength value with complete 3 microphones.

Facing microphone 1 in a direction as shown on the right.
Distance: 0.5m.
Source: Human continuous voice with different frequency (Yeah it's me! :D)
Expect: Mic1 biggest

Result: (sum of strength)

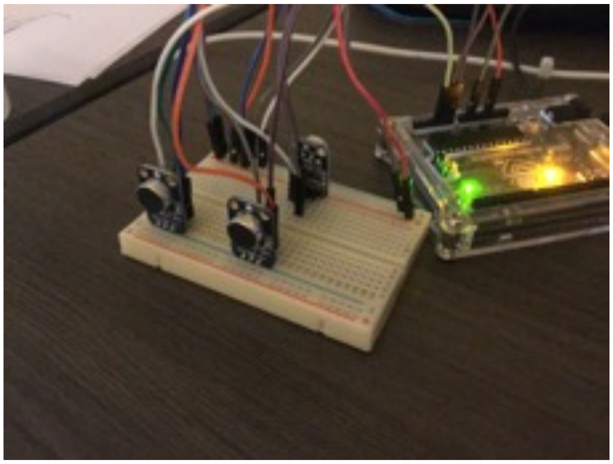
mic1	127370
mic2	116070
mic3	102360



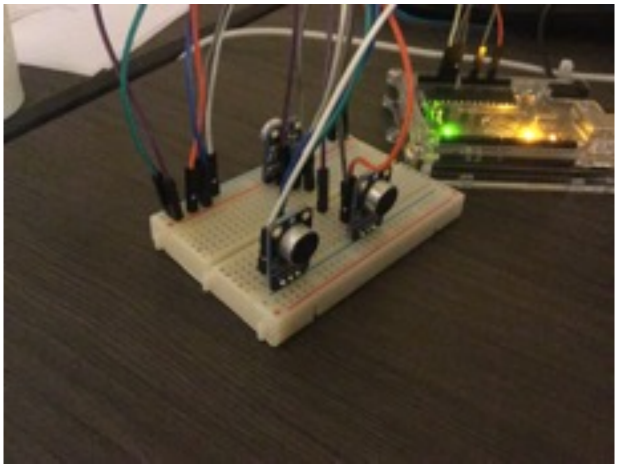
Facing microphone 2 in a direction as shown on the right.
Distance: 0.5m.
Source: Human continuous voice with different frequency (Yeah me again! :D)
Expect: Mic2 biggest

Result: (sum of strength)

mic1	100680
mic2	101010
mic3	86338



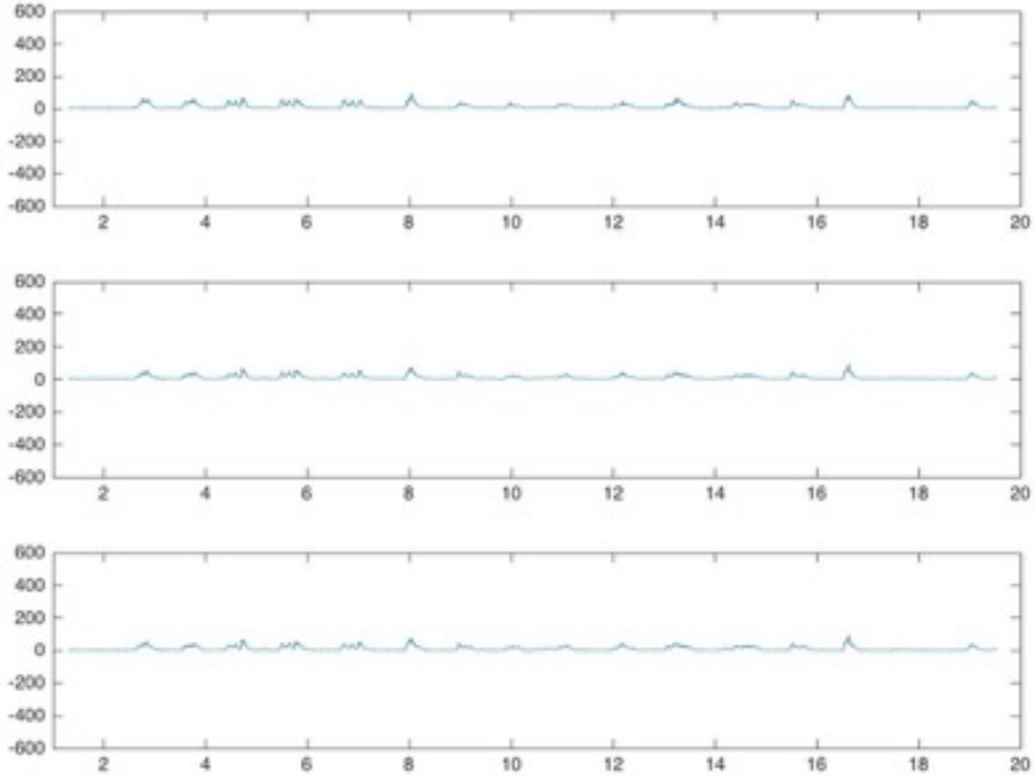
Facing microphone 3 in a direction as shown on the right.
Distance: 0.5m.
Source: Human continuous voice with different frequency (Yeah! :D)
Expect: Mic3 biggest



Result: (sum of strength)

mic1	112360
mic2	111820
mic3	99839

Typical plot like this:

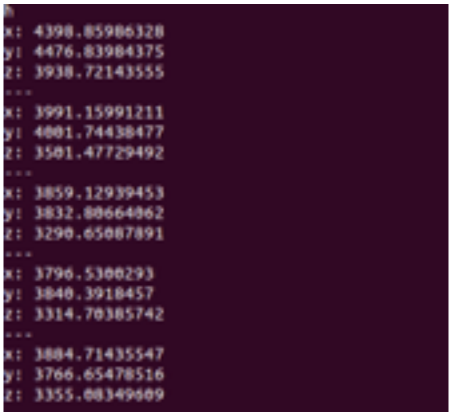


Analysis:

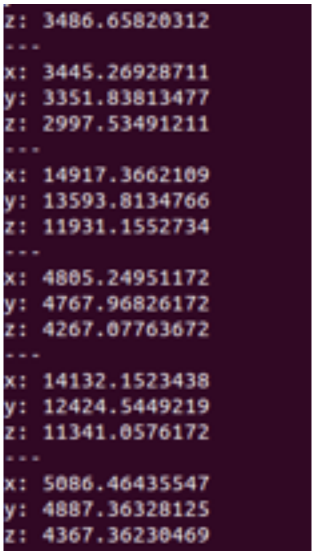
we can see that when facing mic1, mic1 is the biggest.
when facing mic2, mic2 is the biggest, though it does not bigger than mic1 too much(which we consider mic1 has higher gain)
When facing mic3, though mic3 is still the smallest, but no worries, by observation we can deduct that mic3 has the lowest gain of these three. By percentage-of-the-highest, mic3 has 80%, 85% and 89% in three tests above, it’s still consistent.

Supplement：

We just finished developing Arduino ROS publisher, it publishes a sum of strength in an interval of time. the Data is like below, left is quiet(not absolute quiet) ,right is when I was talking(0.5m, median volume)



```
x: 4398.85986328
y: 4476.83984375
z: 3938.72143555
---
x: 3991.15991211
y: 4001.74438477
z: 3501.47729492
---
x: 3859.12939453
y: 3832.80664062
z: 3290.65087891
---
x: 3796.5300293
y: 3840.3918457
z: 3314.70385742
---
x: 3884.71435547
y: 3766.65478516
z: 3355.08349609
```



```
z: 3486.65820312
---
x: 3445.26928711
y: 3351.83813477
z: 2997.53491211
---
x: 14917.3662109
y: 13593.8134766
z: 11931.1552734
---
x: 4805.24951172
y: 4767.96826172
z: 4267.07763672
---
x: 14132.1523438
y: 12424.5449219
z: 11341.0576172
---
x: 5086.46435547
y: 4887.36328125
z: 4367.36230469
```

Conclusion:

That this log is consistent with the 1st log, tests are almost done and it’s ready for development.

Appendix A: arduino sender code(3 mics)

```
double offset_mic_x;
double offset_mic_y;
double offset_mic_z;


////////////PORT Specify////////////////////////
#define PORT_MIC_X A0   // indicate which AD port you connect
#define PORT_MIC_Y A1
#define PORT_MIC_Z A2



///////////Parameter Setup/////////////////////
long const BAUD_RATE = 230400;  //Serial Baud Rate
float const FILTER_THIS = 0.1;  //Low-pass Filter Param
float const FILTER_LAST = 0.9;  //Low-pass Filter Param


//////////////////////////////////////////////

void setup() {

  Serial.begin(BAUD_RATE);
  offset_mic_x = auto_offset_compute(PORT_MIC_X);
  offset_mic_y = auto_offset_compute(PORT_MIC_Y);
  offset_mic_z = auto_offset_compute(PORT_MIC_Z);

  led_blink();

}

void loop() {

  get_mic_strength("mic_x");
  get_mic_strength("mic_y");
  get_mic_strength("mic_z");

}

double get_mic_strength(String mic){
// this function is for getting microphone strength value
// which is basically a filtered absolute microphone altitude
// argument can be "mic_x" "mic_y" or "mic_z"
  int raw_input = 0;

  if (mic.equals("mic_x")){

      raw_input = analogRead(PORT_MIC_X)-offset_mic_x;
      if(raw_input < 0) raw_input = -raw_input;

      double static this_output_x;
      double static last_output_x;
      this_output_x = FILTER_THIS * raw_input + FILTER_LAST * last_output_x;
      Serial.println(this_output_x);
      last_output_x = this_output_x;
      return this_output_x;

  }else if (mic.equals("mic_y")){

      raw_input = analogRead(PORT_MIC_Y)-offset_mic_y;
      if(raw_input < 0) raw_input = -raw_input;

      double static this_output_y;
      double static last_output_y;
      this_output_y = FILTER_THIS * raw_input + FILTER_LAST * last_output_y;
      Serial.println(this_output_y);
      last_output_y = this_output_y;
      return this_output_y;

  }else if (mic.equals("mic_z")){
      raw_input = analogRead(PORT_MIC_Z)-offset_mic_z;
      if(raw_input < 0) raw_input = -raw_input;

      double static this_output_z;
      double static last_output_z;
      this_output_z = FILTER_THIS * raw_input + FILTER_LAST * last_output_z;
      Serial.println(this_output_z);
      last_output_z = this_output_z;
      return this_output_z;
  }

  return 0.0; //indicating failure

}

double auto_offset_compute(int port){
  // this is for computing zero offset of a particular AD port
  // takes argument A0~A5, returns offset
  double sum = 0;
  for (int i=0; i<10000; i++ ){
    sum += analogRead(port);
    delay(0.1);
  }
  return sum/10000.0;
}

void led_blink(){
  pinMode(13,OUTPUT);
  digitalWrite(13,HIGH);delay(500);
  digitalWrite(13,LOW);delay(500);
  digitalWrite(13,HIGH);delay(500);
  digitalWrite(13,LOW);delay(500);
}
```

Appendix B: matlab receiver code(3 mics)

```
%% clean up
clear all;
newobjs = instrfind;
if ~isempty(newobjs)
   fclose(newobjs);
   delete(newobjs);
end
%% start up
s = serial('/dev/tty.usbmodem1451');
set(s,'BaudRate',230400);
fopen(s);
pause(5);

%% read data
out_datas_1 = zeros(1,10000);
out_times_1 = zeros(1,10000);
out_datas_2 = zeros(1,10000);
out_times_2 = zeros(1,10000);
out_datas_3 = zeros(1,10000);
out_times_3 = zeros(1,10000);


tic;
try
   for i = 1:10000
      out_line = fscanf(s,'%f');
      out_datas_1(i) = out_line;
      out_times_1(i) = toc;
      out_line = fscanf(s,'%f');
      out_datas_2(i) = out_line;
      out_times_2(i) = toc;
      out_line = fscanf(s,'%f');
      out_datas_3(i) = out_line;
      out_times_3(i) = toc;
   end
catch exception
   warning('read failed');
   msgString = getReport(exception);
   warning(msgString);
end
%% cleanup
fclose(s);
delete(s);

%% plotting
figure(1);
subplot(3,1,1);
plot(out_times_1,out_datas_1);
axis([1,20,-600,600]);
```

```
subplot(3,1,2);
plot(out_times_2,out_datas_2);
axis([1,20,-600,600]);

subplot(3,1,3);
plot(out_times_2,out_datas_2);
axis([1,20,-600,600]);

sum(out_datas_1)
sum(out_datas_2)
sum(out_datas_3)
```