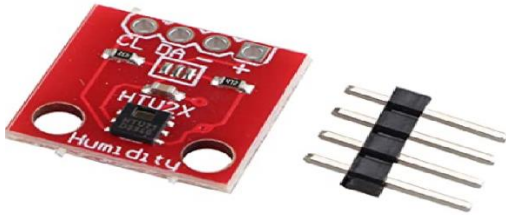


TIME COSTANT EVALUATION OF A THERMAL SENSOR AND DISTANCE STATIC CALIBRATION OF A TIME OF FLIGHT SENSOR USING THE THERMAL SENSOR FOR A MORE PRECISE EVALUATION

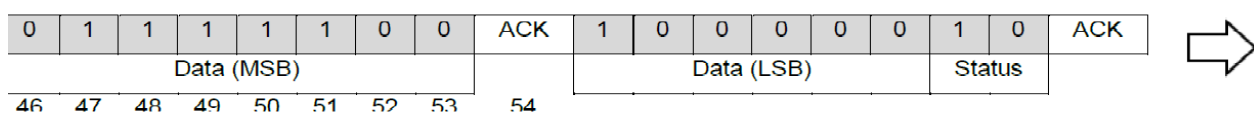
1. TEMPERATURE SENSOR

The sensor used for the thermal acquisition is called HTU21D. The data acquired by the sensor are internally digitalized and then transmitted to the Arduino board through the I2C protocol. Information about its specs are reported below:



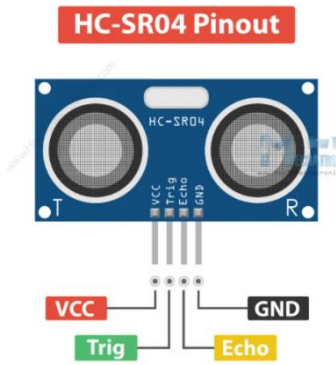
Characteristics		Symbol	Min	Typ	Max	Unit
Resolution	14 bit			0.01		°C
	12 bit			0.04		°C
Temperature Operating Range		T	-40		+125	°C
Temperature Accuracy @25°C	typ			±0.3		°C
	max		See graph 2			°C
Replacement			fully interchangeable			
Measuring time ⁽¹⁾	14 bit			44	50	ms
	13 bit			22	25	ms
	12 bit			11	13	ms
	11 bit			6	7	ms

The measurement information is saved in 14 bits:



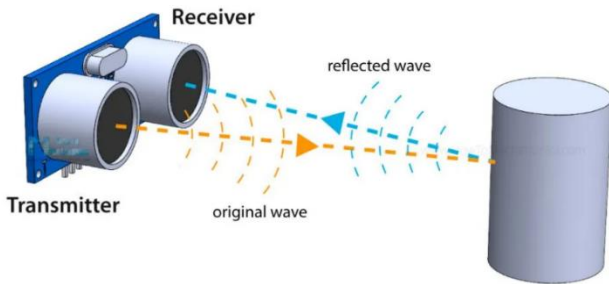
So in order to read correctly the digital value it's required to link 2 byte and then cancel the last 2 digits

2. TIME OF FLIGHT SENSOR



The distance acquisition has been made through the HC-SR04 sensor, made of 2 ultrasonic transducers. Once triggered the start condition with an impulse of $10\mu\text{s}$ in the trigger pin, the transmitter will send out an 8-cycle ultrasonic burst at 40kHz, which will travel at the speed of sound. Right after sending the 8th sonic burst, the Echo pin goes high, going down again once the reflection of the sound wave made by an obstacle has been observed. Once captured, the Echo pin goes down, and through the time passed from these 2 events, it's possible to evaluate the distance according to the following formula:

$$\text{distance} = \text{speed} * \text{time} / 2$$



However, since the speed of sound is not constant but it's (also) a function of the temperature, it's possible to obtain better measurement results with the following formula:

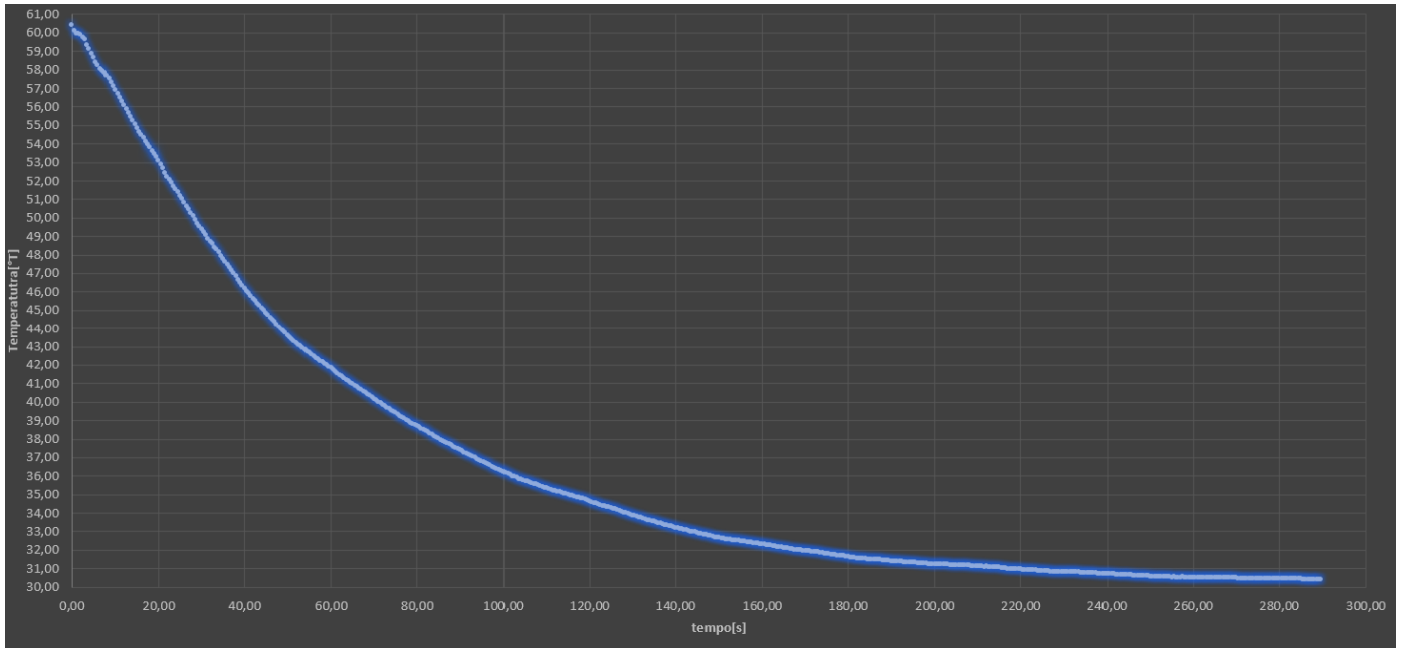
$$d = \frac{t}{2} * \sqrt{401.8 * T} \quad \text{which is a linear approximation of the overall speed of sound function.}$$

Information about the sensor specs are reported below:

Electrical Parameters	Value
Operating Voltage	3.3Vdc ~ 5Vdc
Quiescent Current	<2mA
Operating Current	15mA
Operating Frequency	40KHz
Operating Range & Accuracy	2cm ~ 400cm (1in ~ 13ft) \pm 3mm
Sensitivity	-65dB min
Sound Pressure	112dB
Effective Angle	15°
Connector	4-pins header with 2.54mm pitch
Dimension	45mm x 20mm x 15mm
Weight	9g

3. TIME CONSTANT EVALUATION

In order to evaluate the time constant I used a sort-of step response. I pre-heated the sensor up to a temperature of 60 degree, and then I let it cool down until it reached the room temperature of almost 31 degree. The sensor acquired data every 500ms. Plotting the acquired data the result is the following:



As expected it's the time response of a first order system.

According to the solution of the differential equation for first order systems: $T(t) - T(0) = (T_{\infty} - T(0)) * (1 - e^{-\frac{t}{\tau}})$

It's possible to evaluate its time constant through the following procedure:

$$T(t = 0) = 60.44(15)^{\circ}C \quad T_{\infty} = 30.44(15)^{\circ}C \quad \leftarrow \text{data acquired from the sensor}$$

- 1) Evaluating $T(t)$ for $t = \tau$

$$T(t = \tau) = T(0) - 0.632 * (T(0) - T_{\infty}) = 41.45(20)^{\circ}C$$

Where the uncertainty of $T(t = \tau)$ has been evaluated as follows:

$$u(T = \tau) = \sqrt{u(T_0)^2 + 0.632^2 u(T_0)^2 + 0.632^2 u(T_{\infty})^2} = 0.20^{\circ}C$$

(below it's reported how $u(T_0)$ has been evaluated)

- 2) Finding the corresponding t :

$$t \text{ such that } T(t = \tau) = 41.45^{\circ}C \text{ is a } t \cong 62s$$

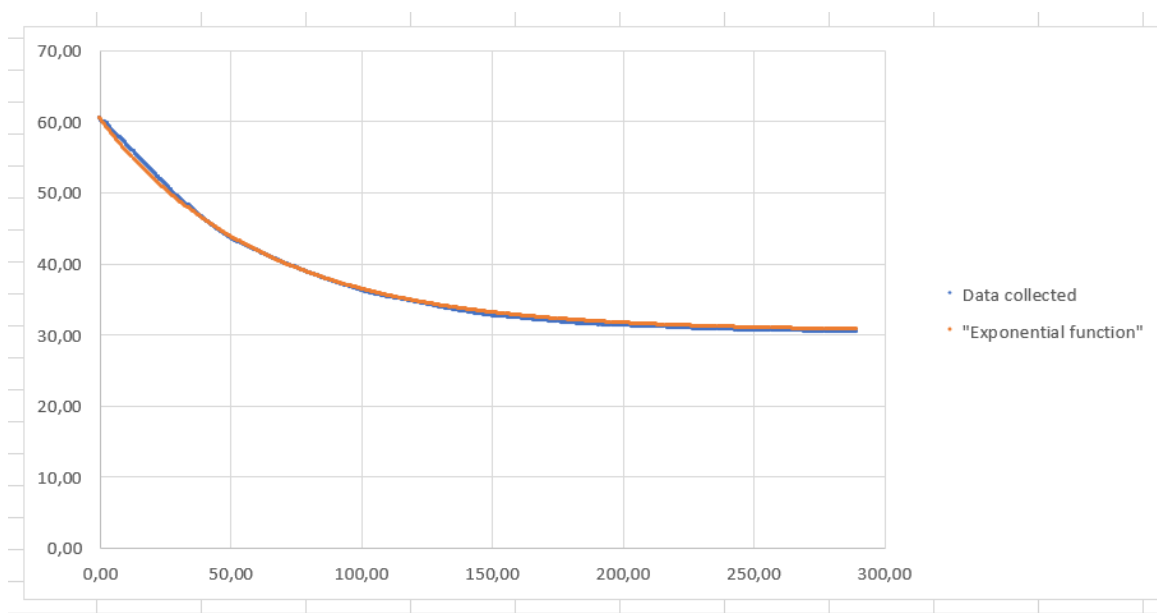
Note: for $t = 3 \tau = 186s$ the temperature reached is $T = 31,50(15)^{\circ}C$, which is the 95% of the final temperature T_{∞}

Considerations about the evaluation of $u(T)$:

In the datasheet is given the resolution of the instrument: 0.03°C , so would be possible to evaluate it's uncertainty considering the standard deviation of a uniform PDF: $u(T) = \frac{res}{\sqrt{12}} = \frac{0.01}{\sqrt{12}} = 0.0029^{\circ}\text{C}$ but it's really low, so I considered other bigger sources of uncertainty. In particular, I hypothesized that the accuracy reported in the datasheet is an extended uncertainty with factor $k=2$, so that the uncertainty is:

$$u(T) = \frac{U}{2} = 0.15^{\circ}\text{C} \text{ which I think is more reliable.}$$

Plot of both the collected data and the ideal temperature exponential decay with $\tau = 62\text{s}$



Note: the Arduino code is not here reported since it's the same of the calibration procedure (that will be later explained) with small differences.

4. STATIC CALIBRATION PROCEDURE

4.1 Description and Arduino Code

The ultrasonic sensor has been placed over the floor with a tape meter (millimeter accuracy) next to it. A box has been placed over the meter at 5cm, 10cm and so on up to 2m.

Each measurement consists in the evaluation of the temperature and 10 times of flight measurements. With the combination of these information is possible to compute the distance, according to the formula:

$$d = \frac{t}{2} * v_{sound} \cong \frac{t}{2} * \sqrt{401.8 * T}$$

Where t is the avg value of the 10 measurements.

The Arduino code used for the measurements is here reported:

```
#include <Wire.h>
// distance sensor
const int trigPin = 9;
const int echoPin = 10;

//temperature sensor
const byte Temp_meas_trigger = 0xF3;
const byte Write = 0xE6; //command
const byte Read = 0xE7; // command
const byte Soft_Reset = 0xFE; //command
const byte Taddress = 0x40; //indirizzo fisico x lettura
const byte Taddress_write = 0x80; //x scrittura

void setup() {
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    Wire.begin(); //initialize the library wire
    reset(Taddress_write, Soft_Reset); //temperature sensor reset
}

void loop() {
    float dist = distanza();
    Serial.print("distanza[mm] =");
    Serial.println(dist);
    delay(100);
    exit(0);
}
```

```

void reset(int Taddress, int Soft_reset){
    Wire.beginTransaction(Taddress_write);
    Wire.write(Soft_Reset);
    Wire.endTransmission(true);
    delay(500);
}

float read_temp(int Taddress, int Temp_meas_trigger){ //temperature reading
    Wire.beginTransaction(Taddress);
    Wire.write(Temp_meas_trigger);
    Wire.endTransmission(false);
    delay(100);
    if (Wire.requestFrom(Taddress, 2) != 2){
        Serial.print("byte non disponibili");
        return 0;
    }
    long V = Wire.read();
    long V2 = Wire.read();
    Wire.endTransmission(true);
    long V3 = estrai_ultimi_2bit(V2);
    V = (V<<8) | V3;
    float Temp = -46.85+175.72*V/pow(2,16);
    //Serial.print();
    return Temp;
}

long estrai_ultimi_2bit(long a){ //pone a 0 gli ultimi due bit
    long x = 0;
    int k=7;
    int vett[8];
    int i = 0;

    for(i=0; i<8; i++) //inizializzo vettore a 0
        vett[i] = 0;

    for(int i=0; i<8; i++){ //vettore in binario dell'intero a, ma al contrario
        vett[i] = a%2;
        a = a/2;
    }

    for(i=0, k=7; i<4; i++,k--){ //cambio ordine vettore
        int aux = vett[i];
        vett[i] = vett[k];
        vett[k] = aux;
    }
    vett[6] = 0;
    vett[7] = 0; //ultime 2 cifre a 0
    for(i=0; i<8; i++){
        x = x + vett[i] * potenza(2,7-i); //valore intero con zero per le ultime 2
        cifre binarie
    }

    return x;
}

int potenza(int base, int esponente){
    int i, result = 1;
    for (i = 0; i < esponente; i++){
        result *= base;
    }
    return result;
}

```

```

float distanza(void){
    float vel = vsuono();
    Serial.print("vsuono[m/s] = ");
    Serial.println(vel);
    delay(100);
    float duration = calcolo_tempo();
    float distance = float_duration/2*vel* pow(10,-3);
    return distance;
}

float calcolo_tempo(){ //returns avg time after 10 meas
    long duration[10];
    long somma = 0;
    Serial.print("misure tempo:\n");
    for(int i=0; i<10; i++){
        digitalWrite(trigPin, LOW);
        delayMicroseconds(2);
        digitalWrite(trigPin, HIGH);
        delayMicroseconds(10);
        digitalWrite(trigPin, LOW);
        duration[i] = pulseIn(echoPin, HIGH); //return time in microseconds
        Serial.println(duration[i]);
        delay(100);
    }
    for(int i=0; i<10; i++){
        somma = somma + duration[i];
    }
    float durata_media = (float)somma/10;
    Serial.print("durata media:\n");
    Serial.println(durata_media);
    delay(100);
    return durata_media;
}

float vsuono(){
    float temp = read_temp(Taddress, Temp_meas_trigger) + 273.15;
    float vel = sqrt(401.8*temp); //speed_of_sound function of T formula
    Serial.print("temperatura [K] = ");
    Serial.println(temp);
    delay(100);
    return vel;
}

```

Some comments:

- The reset command is used for rebooting the sensor switching the power on and off again. In the datasheet its usage is suggested before acquiring data.
- The HTU21D sensor uses the I2c protocol and returns the temperature measurement in 14 bits, so it's required to link 2 bytes and then cancel the last 2 bits of the LSB. These operations are made by the "estrai_ultimi_2bit" function.
- According to the datasheet of the sensor, given the acquired data, in order to read a temperature the following formula have to be applied:

```
float Temp = -46.85+175.72*v/pow(2,16);
```

which returns a temperature in Celsius degrees.

An example of what is acquired in the serial monitor is here reported:

```
1);  
2);  
3);  
4);  
5);  
6);  
7);  
8);  
9);  
10);  
11);  
12);  
13);  
14);  
15);  
16);  
17);  
18);  
19);  
20);  
21);  
22);  
23);  
24);  
25);  
26);  
27);  
28);  
29);  
30);  
31);  
32);  
33);  
34);  
35);  
36);  
37);  
38);  
39);  
40);  
41);  
42);  
43);  
44);  
45);  
46);  
47);  
48);  
49);  
50);  
51);  
52);  
53);  
54);  
55);  
56);  
57);  
58);  
59);  
60);  
61);  
62);  
63);  
64);  
65);  
66);  
67);  
68);  
69);  
70);  
71);  
72);  
73);  
74);  
75);  
76);  
77);  
78);  
79);  
80);  
81);  
82);  
83);  
84);  
85);  
86);  
87);  
88);  
89);  
90);  
91);  
92);  
93);  
94);  
95);  
96);  
97);  
98);  
99);  
100);  
101);  
102);  
103);  
104);  
105);  
106);  
107);  
108);  
109);  
110);  
111);  
112);  
113);  
114);  
115);  
116);  
117);  
118);  
119);  
120);  
121);  
122);  
123);  
124);  
125);  
126);  
127);  
128);  
129);  
130);  
131);  
132);  
133);  
134);  
135);  
136);  
137);  
138);  
139);  
140);  
141);  
142);  
143);  
144);  
145);  
146);  
147);  
148);  
149);  
150);  
151);  
152);  
153);  
154);  
155);  
156);  
157);  
158);  
159);  
160);  
161);  
162);  
163);  
164);  
165);  
166);  
167);  
168);  
169);  
170);  
171);  
172);  
173);  
174);  
175);  
176);  
177);  
178);  
179);  
180);  
181);  
182);  
183);  
184);  
185);  
186);  
187);  
188);  
189);  
190);  
191);  
192);  
193);  
194);  
195);  
196);  
197);  
198);  
199);  
200);  
201);  
202);  
203);  
204);  
205);  
206);  
207);  
208);  
209);  
210);  
211);  
212);  
213);  
214);  
215);  
216);  
217);  
218);  
219);  
220);  
221);  
222);  
223);  
224);  
225);  
226);  
227);  
228);  
229);  
230);  
231);  
232);  
233);  
234);  
235);  
236);  
237);  
238);  
239);  
240);  
241);  
242);  
243);  
244);  
245);  
246);  
247);  
248);  
249);  
250);  
251);  
252);  
253);  
254);  
255);  
256);  
257);  
258);  
259);  
260);  
261);  
262);  
263);  
264);  
265);  
266);  
267);  
268);  
269);  
270);  
271);  
272);  
273);  
274);  
275);  
276);  
277);  
278);  
279);  
280);  
281);  
282);  
283);  
284);  
285);  
286);  
287);  
288);  
289);  
290);  
291);  
292);  
293);  
294);  
295);  
296);  
297);  
298);  
299);  
300);  
301);  
302);  
303);  
304);  
305);  
306);  
307);  
308);  
309);  
310);  
311);  
312);  
313);  
314);  
315);  
316);  
317);  
318);  
319);  
320);  
321);  
322);  
323);  
324);  
325);  
326);  
327);  
328);  
329);  
330);  
331);  
332);  
333);  
334);  
335);  
336);  
337);  
338);  
339);  
340);  
341);  
342);  
343);  
344);  
345);  
346);  
347);  
348);  
349);  
350);  
351);  
352);  
353);  
354);  
355);  
356);  
357);  
358);  
359);  
360);  
361);  
362);  
363);  
364);  
365);  
366);  
367);  
368);  
369);  
370);  
371);  
372);  
373);  
374);  
375);  
376);  
377);  
378);  
379);  
380);  
381);  
382);  
383);  
384);  
385);  
386);  
387);  
388);  
389);  
390);  
391);  
392);  
393);  
394);  
395);  
396);  
397);  
398);  
399);  
400);  
401);  
402);  
403);  
404);  
405);  
406);  
407);  
408);  
409);  
410);  
411);  
412);  
413);  
414);  
415);  
416);  
417);  
418);  
419);  
420);  
421);  
422);  
423);  
424);  
425);  
426);  
427);  
428);  
429);  
430);  
431);  
432);  
433);  
434);  
435);  
436);  
437);  
438);  
439);  
440);  
441);  
442);  
443);  
444);  
445);  
446);  
447);  
448);  
449);  
450);  
451);  
452);  
453);  
454);  
455);  
456);  
457);  
458);  
459);  
460);  
461);  
462);  
463);  
464);  
465);  
466);  
467);  
468);  
469);  
470);  
471);  
472);  
473);  
474);  
475);  
476);  
477);  
478);  
479);  
480);  
481);  
482);  
483);  
484);  
485);  
486);  
487);  
488);  
489);  
490);  
491);  
492);  
493);  
494);  
495);  
496);  
497);  
498);  
499);  
500);  
501);  
502);  
503);  
504);  
505);  
506);  
507);  
508);  
509);  
510);  
511);  
512);  
513);  
514);  
515);  
516);  
517);  
518);  
519);  
520);  
521);  
522);  
523);  
524);  
525);  
526);  
527);  
528);  
529);  
530);  
531);  
532);  
533);  
534);  
535);  
536);  
537);  
538);  
539);  
540);  
541);  
542);  
543);  
544);  
545);  
546);  
547);  
548);  
549);  
550);  
551);  
552);  
553);  
554);  
555);  
556);  
557);  
558);  
559);  
560);  
561);  
562);  
563);  
564);  
565);  
566);  
567);  
568);  
569);  
570);  
571);  
572);  
573);  
574);  
575);  
576);  
577);  
578);  
579);  
580);  
581);  
582);  
583);  
584);  
585);  
586);  
587);  
588);  
589);  
590);  
591);  
592);  
593);  
594);  
595);  
596);  
597);  
598);  
599);  
600);  
601);  
602);  
603);  
604);  
605);  
606);  
607);  
608);  
609);  
610);  
611);  
612);  
613);  
614);  
615);  
616);  
617);  
618);  
619);  
620);  
621);  
622);  
623);  
624);  
625);  
626);  
627);  
628);  
629);  
630);  
631);  
632);  
633);  
634);  
635);  
636);  
637);  
638);  
639);  
640);  
641);  
642);  
643);  
644);  
645);  
646);  
647);  
648);  
649);  
650);  
651);  
652);  
653);  
654);  
655);  
656);  
657);  
658);  
659);  
660);  
661);  
662);  
663);  
664);  
665);  
666);  
667);  
668);  
669);  
670);  
671);  
672);  
673);  
674);  
675);  
676);  
677);  
678);  
679);  
680);  
681);  
682);  
683);  
684);  
685);  
686);  
687);  
688);  
689);  
690);  
691);  
692);  
693);  
694);  
695);  
696);  
697);  
698);  
699);  
700);  
701);  
702);  
703);  
704);  
705);  
706);  
707);  
708);  
709);  
710);  
711);  
712);  
713);  
714);  
715);  
716);  
717);  
718);  
719);  
720);  
721);  
722);  
723);  
724);  
725);  
726);  
727);  
728);  
729);  
730);  
731);  
732);  
733);  
734);  
735);  
736);  
737);  
738);  
739);  
740);  
741);  
742);  
743);  
744);  
745);  
746);  
747);  
748);  
749);  
750);  
751);  
752);  
753);  
754);  
755);  
756);  
757);  
758);  
759);  
760);  
761);  
762);  
763);  
764);  
765);  
766);  
767);  
768);  
769);  
770);  
771);  
772);  
773);  
774);  
775);  
776);  
777);  
778);  
779);  
780);  
781);  
782);  
783);  
784);  
785);  
786);  
787);  
788);  
789);  
790);  
791);  
792);  
793);  
794);  
795);  
796);  
797);  
798);  
799);  
800);  
801);  
802);  
803);  
804);  
805);  
806);  
807);  
808);  
809);  
810);  
811);  
812);  
813);  
814);  
815);  
816);  
817);  
818);  
819);  
820);  
821);  
822);  
823);  
824);  
825);  
826);  
827);  
828);  
829);  
830);  
831);  
832);  
833);  
834);  
835);  
836);  
837);  
838);  
839);  
840);  
841);  
842);  
843);  
844);  
845);  
846);  
847);  
848);  
849);  
850);  
851);  
852);  
853);  
854);  
855);  
856);  
857);  
858);  
859);  
860);  
861);  
862);  
863);  
864);  
865);  
866);  
867);  
868);  
869);  
870);  
871);  
872);  
873);  
874);  
875);  
876);  
877);  
878);  
879);  
880);  
881);  
882);  
883);  
884);  
885);  
886);  
887);  
888);  
889);  
890);  
891);  
892);  
893);  
894);  
895);  
896);  
897);  
898);  
899);  
900);  
901);  
902);  
903);  
904);  
905);  
906);  
907);  
908);  
909);  
910);  
911);  
912);  
913);  
914);  
915);  
916);  
917);  
918);  
919);  
920);  
921);  
922);  
923);  
924);  
925);  
926);  
927);  
928);  
929);  
930);  
931);  
932);  
933);  
934);  
935);  
936);  
937);  
938);  
939);  
940);  
941);  
942);  
943);  
944);  
945);  
946);  
947);  
948);  
949);  
950);  
951);  
952);  
953);  
954);  
955);  
956);  
957);  
958);  
959);  
960);  
961);  
962);  
963);  
964);  
965);  
966);  
967);  
968);  
969);  
970);  
971);  
972);  
973);  
974);  
975);  
976);  
977);  
978);  
979);  
980);  
981);  
982);  
983);  
984);  
985);  
986);  
987);  
988);  
989);  
990);  
991);  
992);  
993);  
994);  
995);  
996);  
997);  
998);  
999);  
1000);  
1001);  
1002);  
1003);  
1004);  
1005);  
1006);  
1007);  
1008);  
1009);  
1010);  
1011);  
1012);  
1013);  
1014);  
1015);  
1016);  
1017);  
1018);  
1019);  
1020);  
1021);  
1022);  
1023);  
1024);  
1025);  
1026);  
1027);  
1028);  
1029);  
1030);  
1031);  
1032);  
1033);  
1034);  
1035);  
1036);  
1037);  
1038);  
1039);  
1040);  
1041);  
1042);  
1043);  
1044);  
1045);  
1046);  
1047);  
1048);  
1049);  
1050);  
1051);  
1052);  
1053);  
1054);  
1055);  
1056);  
1057);  
1058);  
1059);  
1060);  
1061);  
1062);  
1063);  
1064);  
1065);  
1066);  
1067);  
1068);  
1069);  
1070);  
1071);  
1072);  
1073);  
1074);  
1075);  
1076);  
1077);  
1078);  
1079);  
1080);  
1081);  
1082);  
1083);  
1084);  
1085);  
1086);  
1087);  
1088);  
1089);  
1090);  
1091);  
1092);  
1093);  
1094);  
1095);  
1096);  
1097);  
1098);  
1099);  
1100);  
1101);  
1102);  
1103);  
1104);  
1105);  
1106);  
1107);  
1108);  
1109);  
1110);  
1111);  
1112);  
1113);  
1114);  
1115);  
1116);  
1117);  
1118);  
1119);  
1120);  
1121);  
1122);  
1123);  
1124);  
1125);  
1126);  
1127);  
1128);  
1129);  
1130);  
1131);  
1132);  
1133);  
1134);  
1135);  
1136);  
1137);  
1138);  
1139);  
1140);  
1141);  
1142);  
1143);  
1144);  
1145);  
1146);  
1147);  
1148);  
1149);  
1150);  
1151);  
1152);  
1153);  
1154);  
1155);  
1156);  
1157);  
1158);  
1159);  
1160);  
1161);  
1162);  
1163);  
1164);  
1165);  
1166);  
1167);  
1168);  
1169);  
1170);  
1171);  
1172);  
1173);  
1174);  
1175);  
1176);  
1177);  
1178);  
1179);  
1180);  
1181);  
1182);  
1183);  
1184);  
1185);  
1186);  
1187);  
1188);  
1189);  
1190);  
1191);  
1192);  
1193);  
1194);  
1195);  
1196);  
1197);  
1198);  
1199);  
1200);  
1201);  
1202);  
1203);  
1204);  
1205);  
1206);  
1207);  
1208);  
1209);  
1210);  
1211);  
1212);  
1213);  
1214);  
1215);  
1216);  
1217);  
1218);  
1219);  
1220);  
1221);  
1222);  
1223);  
1224);  
1225);  
1226);  
1227);  
1228);  
1229);  
1230);  
1231);  
1232);  
1233);  
1234);  
1235);  
1236);  
1237);  
1238);  
1239);  
1240);  
1241);  
1242);  
1243);  
1244);  
1245);  
1246);  
1247);  
1248);  
1249);  
1250);  
1251);  
1252);  
1253);  
1254);  
1255);  
1256);  
1257);  
1258);  
1259);  
1260);  
1261);  
1262);  
1263);  
1264);  
1265);  
1266);  
1267);  
1268);  
1269);  
1270);  
1271);  
1272);  
1273);  
1274);  
1275);  
1276);  
1277);  
1278);  
1279);  
1280);  
1281);  
1282);  
1283);  
1284);  
1285);  
1286);  
1287);  
1288);  
1289);  
1290);  
1291);  
1292);  
1293);  
1294);  
1295);  
1296);  
1297);  
1298);  
1299);  
1300);  
1301);  
1302);  
1303);  
1304);  
1305);  
1306);  
1307);  
1308);  
1309);  
1310);  
1311);  
1312);  
1313);  
1314);  
1315);  
1316);  
1317);  
1318);  
1319);  
1320);  
1321);  
1322);  
1323);  
1324);  
1325);  
1326);  
1327);  
1328);  
1329);  
1330);  
1331);  
1332);  
1333);  
1334);  
1335);  
1336);  
1337);  
1338);  
1339);  
1340);  
1341);  
1342);  
1343);  
1344);  
1345);  
1346);  
1347);  
1348);  
1349);  
1350);  
1351);  
1352);  
1353);  
1354);  
1355);  
1356);  
1357);  
1358);  
1359);  
1360);  
1361);  
1362);  
1363);  
1364);  
1365);  
1366);  
1367);  
1368);  
1369);  
1370);  
1371);  
1372);  
1373);  
1374);  
1375);  
1376);  
1377);  
1378);  
1379);  
1380);  
1381);  
1382);  
1383);  
1384);  
1385);  
1386);  
1387);  
1388);  
1389);  
1390);  
1391);  
1392);  
1393);  
1394);  
1395);  
1396);  
1397);  
1398);  
1399);  
1400);  
1401);  
1402);  
1403);  
1404);  
1405);  
1406);  
1407);  
1408);  
1409);  
1410);  
1411);  
1412);  
1413);  
1414);  
1415);  
1416);  
1417);  
1418);  
1419);  
1420);  
1421);  
1422);  
1423);  
1424);  
1425);  
1426);  
1427);  
1428);  
1429);  
1430);  
1431);  
1432);  
1433);  
1434);  
1435);  
1436);  
1437);  
1438);  
1439);  
1440);  
1441);  
1442);  
1443);  
1444);  
1445);  
1446);  
1447);  
1448);  
1449);  
1450);  
1451);  
1452);  
1453);  
1454);  
1455);  
1456);  
1457);  
1458);  
1459);  
1460);  
1461);  
1462);  
1463);  
1464);  
1465);  
1466);  
1467);  
1468);  
1469);  
1470);  
1471);  
1472);  
1473);  
1474);  
1475);  
1476);  
1477);  
1478);  
1479);  
1480);  
1481);  
1482);  
1483);  
1484);  
1485);  
1486);  
1487);  
1488);  
1489);  
1490);  
1491);  
1492);  
1493);  
1494);  
1495);  
1496);  
1497);  
1498);  
1499);  
1500);  
1501);  
1502);  
1503);  
1504);  
1505);  
1506);  
1507);  
1508);  
1509);  
1510);  
1511);  
1512);  
1513);  
1514);  
1515);  
1516);  
1517);  
1518);  
1519);  
1520);  
1521);  
1522);  
1523);  
1524);  
1525);  
1526);  
1527);  
1528);  
1529);  
1530);  
1531);  
1532);  
1533);  
1534);  
1535);  
1536);  
1537);  
1538);  
1539);  
1540);  
1541);  
1542);  
1543);  
1544);  
1545);  
1546);  
1547);  
1548);  
1549);  
1550);  
1551);  
1552);  
1553);  
1554);  
1555);  
1556);  
1557);  
1558);  
1559);  
1560);  
1561);  
1562);  
1563);  
1564);  
1565);  
1566);  
1567);  
1568);  
1569);  
1570);  
1571);  
1572);  
1573);  
1574);  
1575);  
1576);  
1577);  
1578);  
1579);  
1580);  
1581);  
1582);  
1583);  
1584);  
1585);  
1586);  
1587);  
1588);  
1589);  
1590);  
1591);  
1592);  
1593);  
1594);  
1595);  
1596);  
1597);  
1598);  
1599);  
1600);  
1601);  
1602);  
1603);  
1604);  
1605);  
1606);  
1607);  
1608);  
1609);  
1610);  
1611);  
1612);  
1613);  
1614);  
1615);  
1616);  
1617);  
1618);  
1619);  
1620);  
1621);  
1622);  
1623);  
1624);  
1625);  
1626);  
1627);  
1628);  
1629);  
1630);  
1631);  
1632);  
1633);  
1634);  
1635);  
1636);  
1637);  
1638);  
1639);  
1640);<
```


4.2 Evaluation of the measurement uncertainty:

The distance is given by the following formula: $d = \frac{t}{2} * v_{sound} \cong \frac{t}{2} * \sqrt{401.8 * T}$

So in order to compute the uncertainty, the Propagation law have to be applied:

$$u_y = \sqrt{\sum_{i=1}^n \left(\frac{\partial f}{\partial x_i} * u_{x_i} \right)^2} \quad \text{which in our case consist in:}$$

$$u(d)_2 = \sqrt{\left(\frac{\partial d}{\partial t} * u(t) \right)^2 + \left(\frac{\partial d}{\partial T} * u(T) \right)^2} \quad \text{where} \quad \frac{\partial d}{\partial t} = \frac{1}{2} * \sqrt{401.8 * T} \quad \text{and} \quad \frac{\partial d}{\partial T} = \frac{t}{4\sqrt{T}} * \sqrt{401.8}$$

For each distance I computed 10 time measurements, so the uncertainty (type A) is evaluated as follows:

$$\bar{t} = \frac{1}{N} * \sum_{i=1}^N (t_i)$$

$$s^2(t) = \frac{1}{N-1} \sum_{i=1}^N (t_i - \bar{t})^2 \quad u(t) = \sqrt{s^2(t)/n}$$

As for the thermal sensor, I also considered as source of uncertainty the accuracy reported in the datasheet, considered as extended uncertainty with factor k=2, so that $u_2(d) = U/2 = 1.5mm$

Finally I evaluated the combined uncertainty $u_c(d) = \sqrt{u(d)_1^2 + u(d)_2^2}$

4.3 uncertainty results with respect to measured data

Here are reported the main results of the acquisition procedure and the evaluation of the uncertainty.

(avg) measured times and temperatures:

dist(mm)	tmedio(μ s)	dev standard	incertezza(μ s)	meas T [K]	incertezza T[°C]
50	297	1,897366596	1	302,32	0.15
100	613	3,265986324	1	302,33	
150	894	3,341656276	1	302,34	
200	1167	3,0713732	1	302,35	
250	1465	4,423422506	1	302,36	
300	1763	6,522610248	2	302,37	
350	2014	9,818350167	3	302,38	
400	2315	11,42171421	4	302,39	
450	2576	6,583649781	2	302,40	
500	2846	5,73585216	2	302,41	
550	3161	17,00196067	5	302,42	
600	3427	18,57597014	6	302,43	
650	3701	14,41449887	5	302,44	
700	4002	36,37703793	12	302,45	
750	4272	22,44128141	7	302,46	
800	4532	37,65766854	12	302,47	
850	4832	30,75060975	10	302,48	
900	5201	46,26541785	15	302,49	
950	5323	22,80570104	7	302,50	
1000	5727	48,16649366	15	302,51	
1050	5367	64,06498437	20	302,52	
1100	6338	50,69834536	16	302,53	
1150	6606	53,73598008	17	302,54	
1200	895	51,4224335	16	302,55	
1250	7096	24,67567403	8	302,56	
1300	7391	67,7266565	21	302,57	
1350	7671	51,90118817	16	302,58	
1400	7953	56,07148216	18	302,59	
1450	8223	50,68157675	16	302,60	
1500	8636	56,53749985	18	302,61	
1550	8800	62,17904256	20	302,62	
1600	9079	25,83042994	8	302,63	
1650	9387	72,3328725	23	302,64	
1700	9654	39,96442863	13	302,65	
1750	9980	17,38581798	5	302,66	
1800	10257	51,00119824	16	302,67	
1850	10534	47,10508347	15	302,68	
1900	10786	43,48230802	14	302,69	
1950	11110	34,25460229	11	302,70	
2000	11413	48,66712329	15	302,71	

$u_1(d)$ computation:

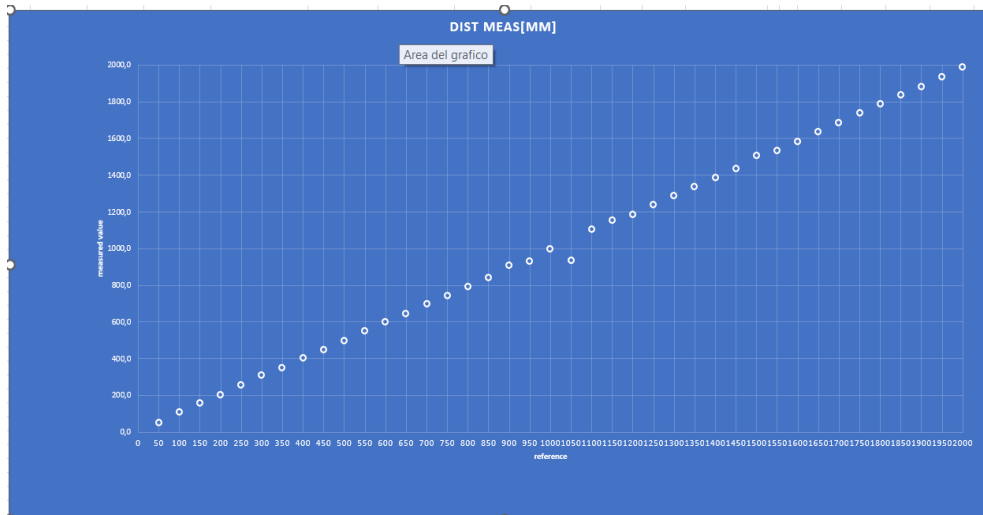
						c1(T)	c2(t,T)	u1(d)[m]
$u_{d1} = \sqrt{(c1(T) * u(t))^2 + (c2(t,T) * u(T))^2}$						174,26	8,6E-05	0,00011
						174,27	0,00018	0,00018
						174,27	0,00026	0,00019
						174,27	0,00034	0,00018
						174,28	0,00042	0,00025
						174,28	0,00051	0,00037
						174,28	0,00058	0,00055
						174,28	0,00067	0,00064
						174,29	0,00074	0,00038
						174,29	0,00082	0,00035
						174,29	0,00091	0,00095
						174,30	0,00099	0,00104
						174,30	0,00107	0,00081
						174,30	0,00115	0,00201
						174,30	0,00123	0,00125
						174,31	0,00131	0,00209
						174,31	0,00139	0,00171
						174,31	0,0015	0,00256
						174,32	0,00153	0,00128
						174,32	0,00165	0,00267
						174,32	0,00155	0,00354
						174,32	0,00183	0,00281
						174,33	0,0019	0,00298
						174,33	0,00026	0,00284
						174,33	0,00204	0,0014
						174,34	0,00213	0,00375
						174,34	0,00221	0,00289
						174,34	0,00229	0,00312
						174,34	0,00237	0,00282
						174,35	0,00249	0,00315
						174,35	0,00253	0,00346
						174,35	0,00262	0,00149
						174,36	0,0027	0,00401
						174,36	0,00278	0,00225
						174,36	0,00287	0,00108
						174,37	0,00295	0,00286
						174,37	0,00303	0,00265
						174,37	0,00311	0,00246
						174,37	0,0032	0,00197
						174,38	0,00329	0,00274

Measured distance and combined uncertainty:

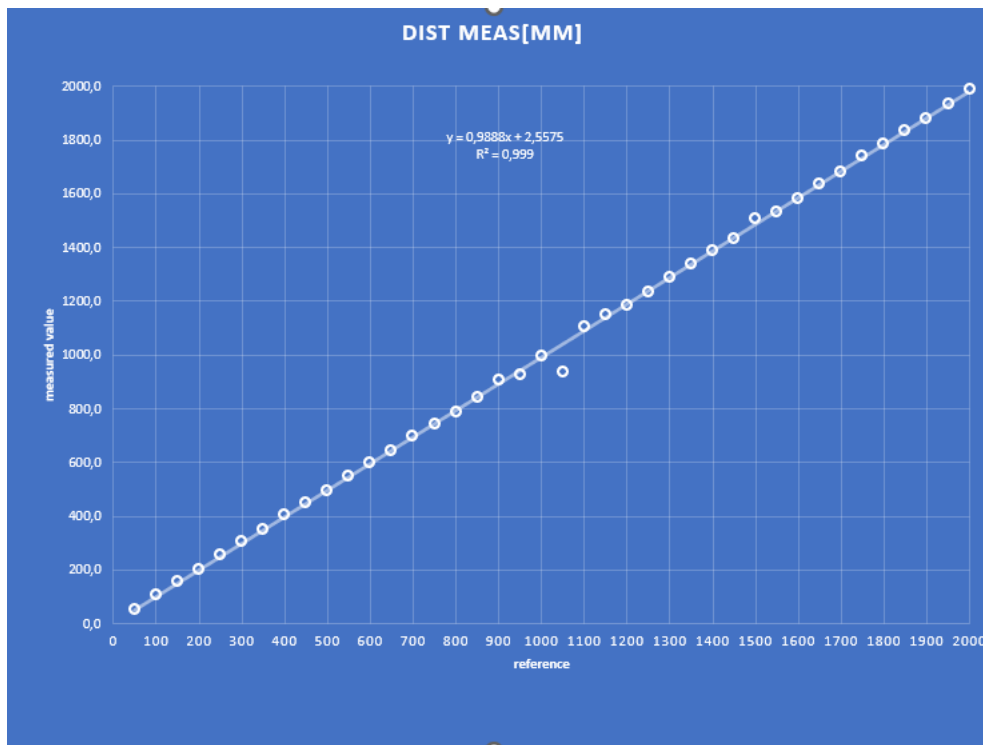
J	K	L	M	N
dist(mm)	dist meas[mm]	u1[mm]	u2[mm]	combined u [mm]
50	51,75	0,1	1,50	1,5
100	106,82	0,2		1,5
150	155,62	0,2		1,5
200	203,37	0,2		1,5
250	255,29	0,3		1,5
300	307,05	0,4		1,5
350	350,97	0,5		1,6
400	403,43	0,6		1,6
450	448,75	0,4		1,5
500	495,82	0,3		1,5
550	550,70	0,9		1,8
600	597,08	1,0		1,8
650	645,02	0,8		1,7
700	697,32	2,0		2,5
750	744,36	1,3		2,0
800	789,67	2,1		2,6
850	842,15	1,7		2,3
900	906,46	2,6		3,0
950	927,72	1,3		2,0
1000	998,15	2,7		3,1
1050	935,23	3,5		3,8
1100	1104,66	2,8		3,2
1150	1151,39	3,0		3,3
1200	1186,58	2,8		3,2
1250	1236,75	1,4		2,0
1300	1288,21	3,7		4,0
1350	1336,87	2,9		3,2
1400	1385,95	3,1		3,5
1450	1433,18	2,8		3,2
1500	1505,16	3,1		3,5
1550	1533,62	3,4		3,8
1600	1582,22	1,5		2,1
1650	1635,90	4,0		4,3
1700	1682,67	2,2		2,7
1750	1739,35	1,1		1,8
1800	1787,63	2,8		3,2
1850	1836,09	2,6		3,0
1900	1879,87	2,4		2,9
1950	1936,56	1,9		2,5
2000	1989,34	2,7		3,1

4.4 Calibration procedure:

Plotting the measured distance on y axis and the reference distance on x axis the result is the following:

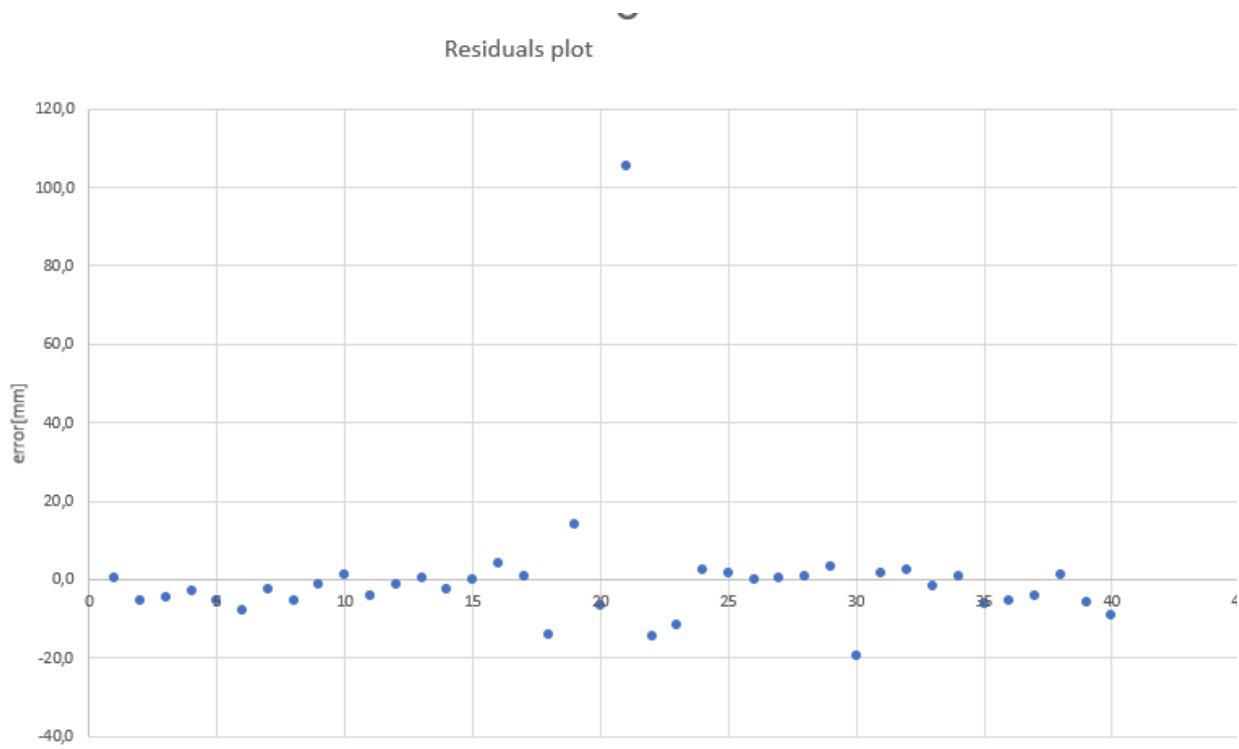


It's clearly a linear relationship so I used a linear regression model that minimizes the quadratic distance between the line and the points:



The index $R^2 = 0.999$ and the angular coefficient is 0.9888, so the line is a good approximation of the real behaviour.

Plotting the residuals:



Can be seen that they are randomly distributed, so again the regression is a “good” one.

Notice also that a point is very far from the others, probably due to an error during the data acquisition procedure. Could be possible to obtain even better results cancelling that point from the regression equation.