Attāluma mērīšana, arduino bibliotēkas

Ivars Driķis

2019. gada 4. septembrī

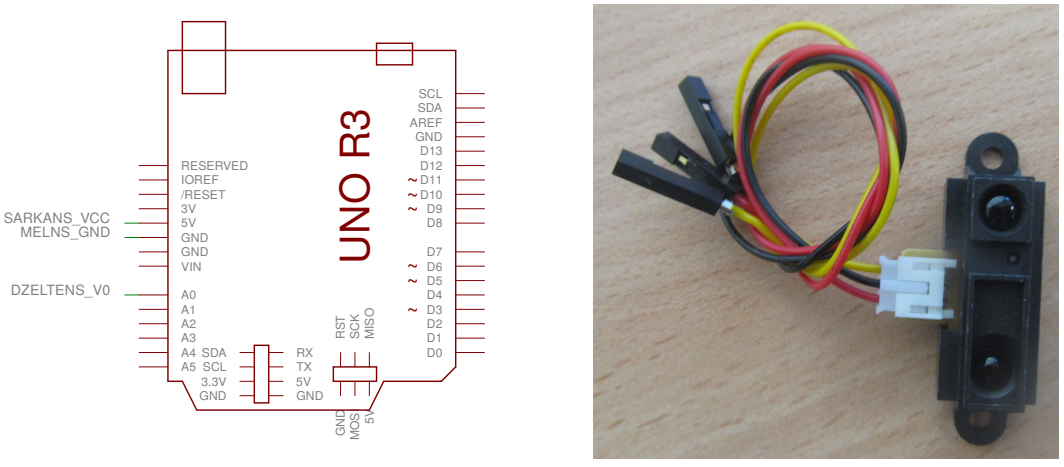
Šajā dokumentā vēlos paskaidrot kas ir Arduino bibliotēkas un ar divu piemēru palīdzību parādīt kāpēc tās ir lietderīgi lietot.

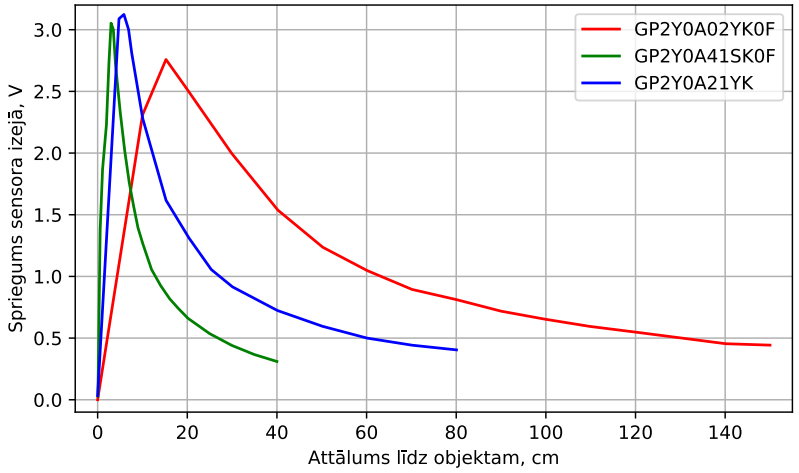
1 Attāluma mērīšana izmantojot infrasarkano staru attāluma sensorus

Vislētākais in visplašāk izmantotais attāluma mērīšanas vieds ir izmantojot ultraskaņas attāluma sensoru. Par to nedaudz vēlā, 2 sadaļā. Precīzāku attālumu mērīšanu parasti veic izmantojot nedaudz dārgākos Sharp infrasarkano staru attāluma sensorus no GP2Y0A sērijas [1, 2, 3], un tie izskatās kā 2 zīmējumā. Lietošanā šie sensori ir maksimāli vienkārši: padodam barošanu, šajā attēlā tas ir melnais vads GND un barošanu +5V kas ir sarkanais vads. Dzeltenajā vadā sensors uzstāda spriegumu, kura vērtība ir atkarīga no attāluma līdz tuvākajam pietiekoši lielam objektam. Iegādājoties šo sensoru pārbaudiet, vai šis kabelis ietilpst komplektā, jo lai salodētu balto spraudni vajadzīgas ļoti labas lodēšanas iemaņas. Var lodēt tieši pie sensora, bet tas izskatīsies neglīti. Kabeļa vadu krāsas var atšķirties, bet parasti elektronikā ievēro šādu krāsu izvēli: viskarstākā ir barošana, visaukstākā ir GND. Jebkurā gadījumā, varam novietot sensoru kā redzams attēlā un noskaidrot kas atbilst kurai krāsai.

Lai ilustrētu atšķirību starp šiem sensoriem, zīmējumā 2 apvienoju to izejas spriegumus atkarību no attāluma līdz objektam. Ar šiem sensoriem ir jābūt uzmanīgam!

Att. 1: Infrasarkano staru attāluma sensors un tā pieslēgšana Arduino savietojamam mikrodatoram.





Att. 2: Spriegums infrasarkano staru attāluma sensora izejā atkarībā no attāluma līdz objektam. Dati no dokumentācijas [1, 2, 3].

- Lai arī sensori GP2Y0A21YK, GP2Y0A41SK0F un GP2Y0A02YK0F izskatās praktiski identiski, programma kas ir izveidota vienam sensoram, ar citu sensoru strādās nepareizi!
- Sakarība starp spriegumu sensora izejā un attālumu diemžēl nav viennozīmīga. Tas nozīmē, ka piemēram 2V spriegums sensora GP2Y0A02YK0F izejā atbilst gadījumam, kad objekts atrodas vai nu apmēram 9 cm vai arī 16 cm attālumā un nav iespējams noteikt kurš gadījums īsti ir. Šo dilemmu risina vienkārši, objektus neliek pārāk tuvu sensoram. Sensoru dokumentācijā ir norādīts šis tuvākais izmērāmais attālums. Tālākais izmērāmais attālums atkarīgs gan no sensora tipa gan arī no objekta izmēriem.

Sensors	Izmērāmais attālums
GP2Y0A21YK [1]	no 10 līdz 80 cm
GP2Y0A41SK0F [2]	no 4 līdz 30 cm
GP2Y0A02YK0F [3]	no 15 līdz 150 cm

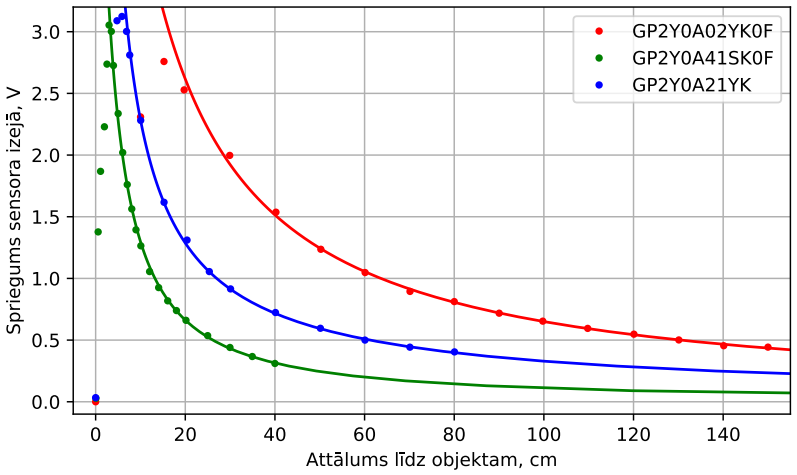
Attāluma aprēķināšanai visvienkāršāk ir izmantot matemātiskas formulas kuras ir piemeklētas tā, lai aprēķinātais attālums labi sakrīt ar dokumentācijā dotajiem un zīmējumā 2 redzamajiem lielumiem. Lai U ir nomērītais spriegums voltos un L ir attālums līdz objektam centimetros

$$L_{\mathrm{GP2Y0A21YK}} = \frac{27.45}{U^{1.167}} - 0.455 \tag{1}$$

$$L_{\mathrm{GP2Y0A41SK0F}} = \frac{16.55}{U^{0.832}} - 3.309 \tag{2}$$

$$L_{\mathrm{GP2Y0A02YK0F}} = \frac{73.82}{U^{0.930}} - 10.18 \tag{3}$$

To, kas sanāk, redzams zīmējumā 3. Parasti grafikus zīmē tā, lai uz x ass (horizontāli), būtu atlikti cēloņi un uz y ass (vertikāli) sekas. Piemēram, zīmējumu 2 iegūst šādi: novieto objektu 20 cm attālumā no sensora un izmēra spriegumu, pēc tam objektu pārvieto 25 cm attāluma un atkal izmēra spriegumu. Tātad cēlonis ir attālums līdz objektam bet sekas ir nomērītais spriegums. Zīmējumā 3 viss ir savādāk, nomērītais spriegums kā cēlonis ir atlikts



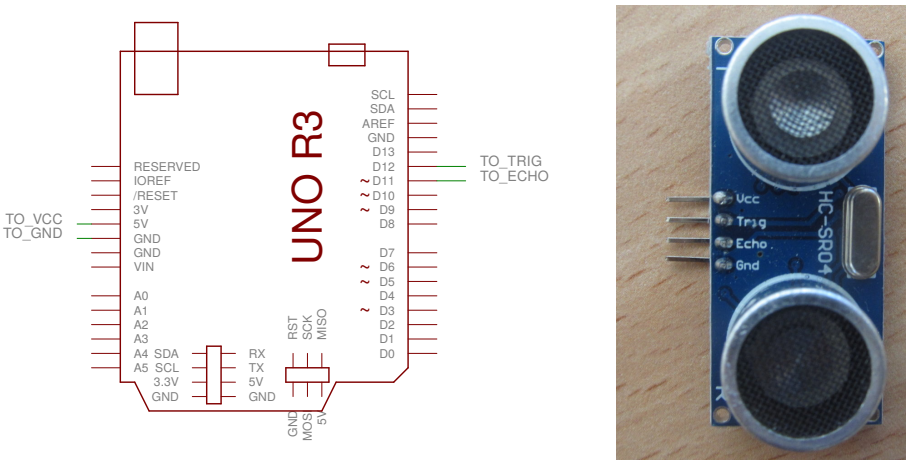
Att. 3: Formulas (1), (2) un (3) darbībā.

vertikāli, bet aprēķinātais attālums kā sekas ir atlikts horizontāli. Šādi zīmēju tāpēc, lai šie abi attēli būtu maksimāli līdzīgi.

Atliek to visu sarakstīt programmā

Listing 1: InfraRed.ino

```
1  void setup()
2  {
3    Serial.begin( 9600 );
4  }
5
6  void loop()
7  {
8    float volts = analogRead(A0) * 5./1024.;
9    Serial.print("Distance ");
10   Serial.print( getDistGP2Y0A41SK0F(volts) );
11   Serial.print(" cm");
12   delay(1000);
13 }
14
15 float getDistGP2Y0A21YK( float volts )
16 {
17   return 27.45*pow(volts,-1.167)-0.455;
18 }
19
20 float getDistGP2Y0A41SK0F( float volts )
21 {
22   return 16.55*pow(volts,-0.832)-3.309;
23 }
24
25 float getDistGP2Y0A02YK0F( float volts )
26 {
27   return 73.82*pow(volts,-0.930)-10.18;
28 }
```



Att. 4: Ultraskaņas attāluma sensora HC-SR04 pieslēgšana Arduino savietojamam mikrodatoram.

2 Attāluma mērīšana izmantojot ultraskaņas attāluma sensoru

Viens no izplatītākajiem un lētākajiem veidiem kā mērīt attālumu ir izmantot ultraskaņas attāluma sensoru HC-SR04 [4]. Ar tā palīdzību mēra laiku, kas nepieciešams lai ultraskaņas impulss veiktu attālumu no sensora līdz objektam un pēc tam atpakaļ līdz sensoram. Pēc tam, izmantojot skaņas ātrumu gaisā, izrēķina attālumu līdz objektam. Visas nepieciešamās lietas veic bibliotēka NewPing [5]. Tālāk apskatīsim šīs bibliotēkas pielietojuma standarta piemēru.

Sensors redzams 4 zīmējumā, skatāmies uz tā ieeju/izeju marķējumiem. Vispirms ieejas Gnd un Vcc pieslēdzam atbilstošajām mikrodatora izejām. Savukārt ieeju Trig un Echo izeju varam pieslēgt pie jebkurām no mikrodatora ciparu ieejām/izejām (D0 un D1 bez īpašas vajadzības nevajadzētu izmantot, jo tad lai aukšuplādētu programmu, sensors uz laiku būs jāatslēdz). Tikai sava izvēle ir jādara zināma programmai. Parasti gan rīkojas otrādi, apskatās programmā kur kas jāpieslēdz un tā arī dara. Tāpēc skatāmies 2 programmā 3 un 4 rindiņu. Tur teksts, ja tulkojam mainīgo nosaukumus, ka Trig pieslēdzam pie 12 ciparu ieejas/izejas, bet Echo izeju pie 11 ciparu ieejas/izejas. Pie viena 5 rindiņā uzstādām maksimālo attālumu centimetros, kuru vēlamies mērīt. Mēs protams varējām 3-7 rindiņu vietā rakstīt uzreiz NewPing sonar(12,11,200);, tikai tad būtu jāmeklē dokumentācijā kur īsti kas jāslēdz un ko nozīmē mistiskais skaitlis 200. Programmētāji parasti ļoti nevēlas rakstīt komentārus savām programmām, tāpēc piemērā lietotais stils ļauj nedaudz paslinkot un tajā paša laikā ietaupīt milzīgi daudz laika, ja vajadzēs programmu lietot pēc mēneša vai pat vairākiem gadiem. Interesanti, ka šādu mainīgo izveide programmas darbības efektivitāti nesamazina. Mulsinošais const pasaka kompilatoram, ka mēs šo mainīgo vērtības nemainīsim. Tāpēc šādi mainīgie vietu datora atmiņā neaizņem, jo kompilators uzreiz ievieto to vērtības vajadzīgajās vietās.

Listing 2: UltraSonic.ino

```
1 #include <NewPing.h>
2
3 const int pinTrig = 12;
4 const int pinEcho = 11;
5 const int maxDist = 200;
```

```
6
7 NewPing sonar(pinTrig, pinEcho, maxDist);
8
9 void setup()
10 {
11   Serial.begin( 9600 );
12 }
13
14 void loop()
15 {
16   delay(50);
17   Serial.print("Ping: ");
18   Serial.print(sonar.ping_cm());
19   Serial.println("cm");
20 }
```

Salīdzināsim programmas un 1 un 2. Otrā veic daudz komplicētākas darbības: ierosina ultraskaņas signāla noraidīšanu, gaida tā pienākšanu atpakaļ un izmantojot nomērīto laiku aprēķina attālumu līdz objektam, taču programma 2 ir pat vienkāršāka par programmu 1! Un to visu panāk izmantojot *bibliotēkas*.

Kad mēģinām šo projektu kopmilēt, tad saņem kļūdas paziņojumu: *fatal error NewPing.h: No such file or directory*. Tas nozīmē, ka jūsu sistēmā pagaidām vēl nav instalēta NewPing bibilotēka. Kā to izdarīt, parādīts attēlā 5.

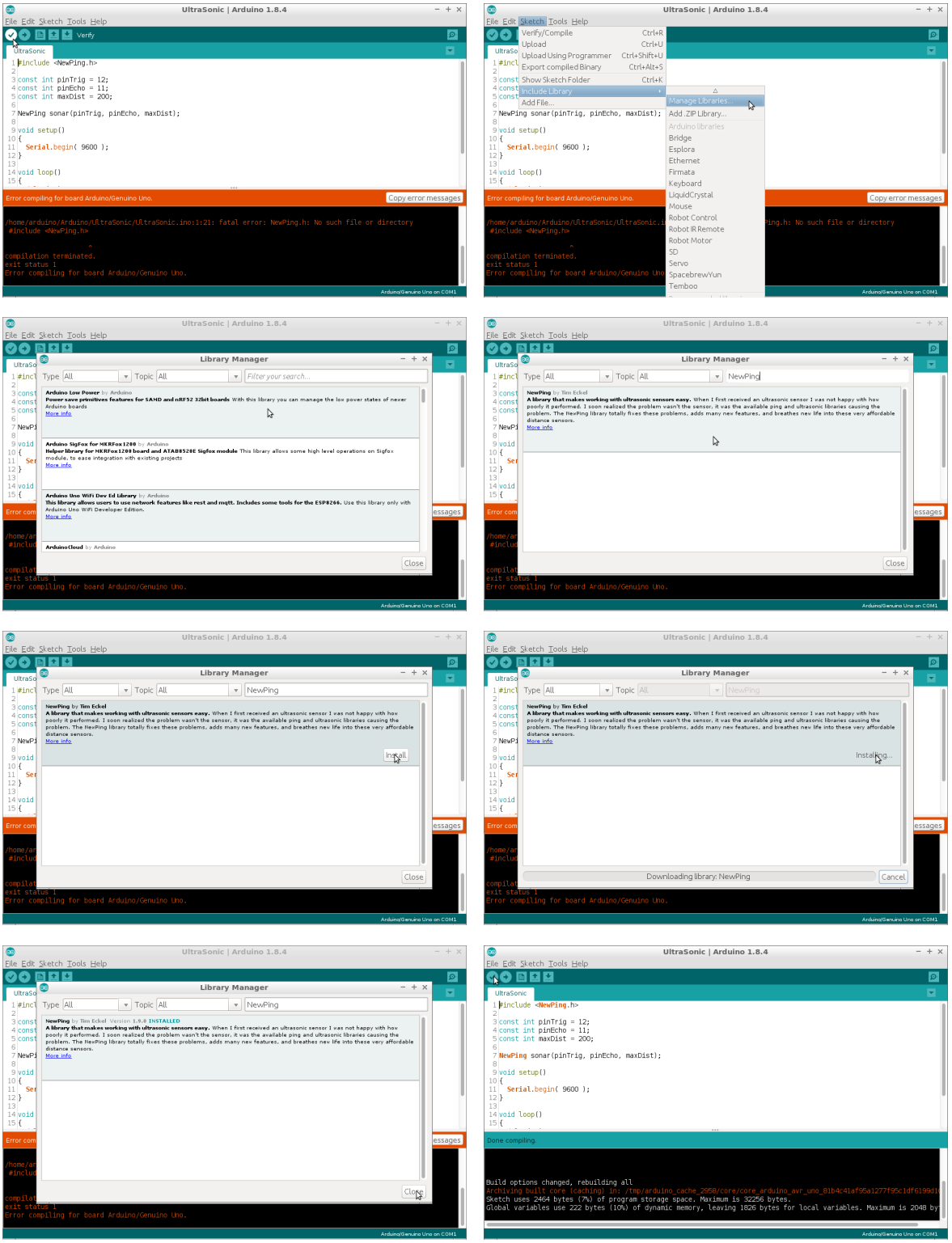
1. Izvēlamies Sketch→Include Library→ Manage Libraries
2. Include Library meklēšanas logā rakstām NewPing
3. Ar peli uzklikšķinām uz pelēkā fona
4. Klikšķinām uz Install. Sagaidām kamēr gatavs, veram Manage Libraries logu ciet.
5. Kompilējam programmu vēlreiz, un viss kārtībā.

Kas tad notiek bibliotēkas instalēšanas procesa laikā? Izrādās, ka katalogā Arduino/libraries ir parādījies katalogs NewPing, kā redzam 6 zīmējumā. Tas savukārt satur divus katalogus src un examples. Pirmajā ir divi faili NewPing.h un NewPing.cpp kas arī ir NewPing bibliotēka. Savukārt katalogā src ir šīs bibliotēkas lietošanas piemēri.

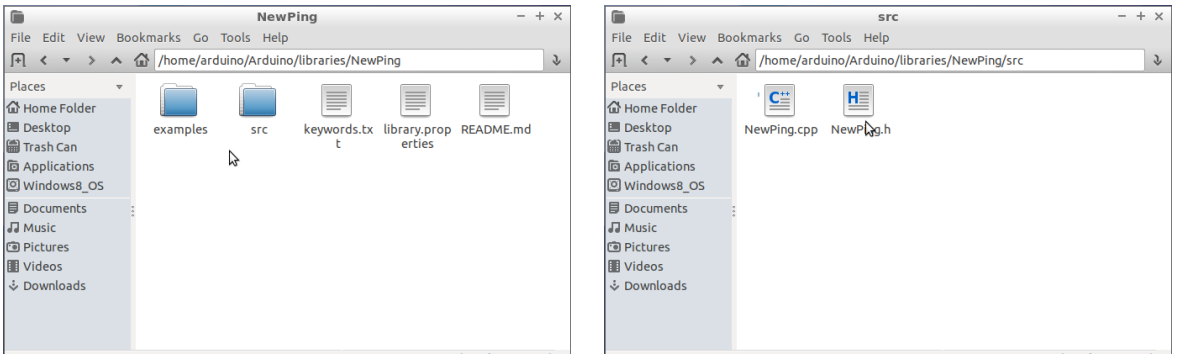
Failus NewPing.h un NewPing.cpp varam paskatīties, bet ja nekas nav skaidrs, nevajag uztraukties. Pilnīgi pietiek zināt tikai trīs lietas:

- Programmas sākumā jāraksta #include <NewPing.h>. Šajā failā ir aprakstīti objekti un funkcijas, kurai mums būs nepieciešami šo bibliotēku lietojot.
- Jādefinē objekts, kas strādās ar sensoru NewPing sonar(pinTrig , pinEcho , maxDist). Argumentos ir to divu ciparu ieejas/izejas numuri, pie kurām pieslēgts sensors ka arī sensora maksimālais darbības attālums.
- Kad nepieciešams attālums līdz objektam, tad vienkārši izsaucam funkciju sonar.ping_cm().

Tiem, kas vēlas zināt vairāk, varam sākt lasīt ar Arduino Language Reference [**?**]. Tālāk ieteicams lasīt labu M un pēc tam M++ grāmatu. Ir kur augt.



Att. 5: NewPing bibliotēkas instalēšanas soļi.



Att. 6: NewPing bibilotēkas instalēšanas rezultāts.

3 Bibliotēka infrasarkano staru attāluma sensoriem

Lai nostiprinātu zināšanas par bibliotēkām, no programmas 1 izveidosim bibliotēku IrDist. Vispirms bibliotēkas funkcijas aprakstošais *header* fails IrDist.h, kuram ir jāatrodas katalogā Arduino/libraries/IrDist/

Listing 3: Arduino/libraries/IrDist/IrDist.h

```
1 float getDistGP2Y0A21YK( float volts );
2 float getDistGP2Y0A41SK0F( float volts );
3 float getDistGP2Y0A02YK0F( float volts );
```

Pēc tam definējam pašas funkcijas failā IrDist.cpp, kuram arī ir jāatrodas katalogā Arduino/libraries/IrDist/. Šeit pirmajā rindā iekļaujam iepriekš aprakstīto *header* failu IrDist.h. Tā vajag darīt, jo tas ir pašpārbaudes tests, kas pārbauda vai definē tieši tādas pašas funkcijas kādas aprakstītas *header* failā. Otrajā rindiņā mēs aprakstām matemātiskās funkcijas, no kuram mums ir nepieciešama pow.

Listing 4: Arduino/libraries/IrDist/IrDist.cpp

```
1 #include "IrDist.h"
2 #include <math.h>
3
4 float getDistGP2Y0A21YK( float volts )
5 {
6   return 27.45*pow(volts,-1.167)-0.455;
7 }
8
9 float getDistGP2Y0A41SK0F( float volts )
10 {
11   return 16.55*pow(volts,-0.832)-3.309;
12 }
13
14 float getDistGP2Y0A02YK0F( float volts )
15 {
16   return 73.82*pow(volts,-0.930)-10.18;
17 }
```

Beigu beigās no programmas 1 pāri paliek tikai 15 rindiņas.

Listing 5: IrDistExam.ino

```
1 #include "IrDist.h"
2
```

```
3  void setup()
4  {
5    Serial.begin( 9600 );
6  }
7
8  void loop()
9  {
10   float volts = analogRead(A0) * 5./1024.;
11   Serial.print("Distance ");
12   Serial.print( getDistGP2Y0A41SK0F(volts) );
13   Serial.print(" cm");
14   delay(1000);
15 }
```

Literatūras saraksts

[1] Infrared Proximity Sensor - Sharp GP2Y0A21YK. https://www.sparkfun.com/datasheets/Components/GP2Y0A21YK.pdf. [Online; accessed 2017.02.10].

[2] Infrared Proximity Sensor Short Range - Sharp GP2Y0A41SK0F. http://www.sharp-world.com/products/device/lineup/data/pdf/datasheet/gp2y0a41sk_e.pdf. [Online; accessed 2017.02.10].

[3] Infrared Proximity Sensor Long Range - Sharp GP2Y0A02YK0F. https://www.sparkfun.com/datasheets/Sensors/Infrared/gp2y0a02yk_e.pdf. [Online; accessed 2017.02.10].

[4] Ultrasonic Ranging Module HC-SR04. http://www.micropik.com/PDF/HCSR04.pdf. [Online; accessed 2017.02.10].

[5] NewPing Library for Arduino. http://playground.arduino.cc/Code/NewPing. [Online; accessed 2017.02.10].

[6] Arduino Language Reference. https://www.arduino.cc/reference/en/. [Online; accessed 2018.04.24].