

Nils Kr. Rossing

Måling av lufttrykk, høyde og posisjon med Arduino LÆRERVEILEDNING



NTNU



Trondheim

Institutt for
fysikk

Skolelaboratoriet
for matematikk, naturfag
og teknologi

Februar 2020

Måling av lufttrykk, høyde og posisjon med Arduino LÆRERVEILEDNING

Nils Kr. Rossing

Måling av lufttrykk, høyde og posisjon med Arduino

Trondheim 2020

Layout og redigering: Nils Kr. Rossing, Skolelaboratoriet ved NTNU

Trykk: NTNU Grafisk senter

Tekst og bilder: Nils Kr. Rossing, Skolelaboratoriet ved NTNU

Faglige spørsmål rettes til:

Skolelaboratoriet for matematikk, naturfag og teknologi

Institutt for fysikk

v/ Nils Kr. Rossing nils.rossing@ntnu.no

Skolelaboratoriet ved NTNU

Realfagbygget,

Høgskoleringen 5,

7491 Trondheim

Telefon: 73 55 11 43

Telefaks: 73 55 11 40

<http://www.ntnu.no/skolelab/>

Rev 2.2 – 29.02.20



Forord

Hftet ble laget i forbindelse med en nettverkssamling for ToF-lærere i Østlandsregionen mars 2018 og er i første rekke beregnet som et hjelpemiddel under samling og som en ressurs som ev. kan brukes av lærerne når de kommer tilbake til egen skole. Senere er det også utviklet et tilhørende elevhefte for bruk ved elevverksted for elever i videregående skole. Kurset er holdt ved Gløshaugen NTNU og på fagdag ved Bårdshaug Herregård i mars 2020.

Hftet beskriver hvordan man kan koble opp og måle lufttrykk ved hjelp av sensoren BMP180 og en Arduino UNO. Videre hvordan man kalibrerer og gjør høydeberegninger på bakgrunn av lufttrykk. Det legges vekt på å karakterisere egenskapene og kvaliteten til trykkmåleren BMP180.

For den som ønsker å gå litt videre beskrives også bruk av en billig GPS-mottaker og hvordan dataene fra denne kan visualiseres ved hjelp av Google Earth.

Hftet er hverken utfyllende eller ferdig, men må først og fremst betraktes som idehefte som grunnlag for videre utvikling.

Skolelaboratoriet ved NTNU
Februar 2020
Nils Kr. Rossing



Innhold

1	Innledning	9
2	Montering og oppkobling for trykkmåling	10
2.1	Oppkobling	10
2.2	Kort innføring i bruk av Arduino editoren (IDE)	11
2.3	Programmering	14
2.4	Beregn høyden – legg inn formelen i programmet	19
2.5	Kalibrering	21
3	Oppdragene	21
3.1	Om du befinner det på Bårdshaug i Orkanger:	21
3.1.1	Oppdrag 1 – Mål relative høyder inne på Bårdshaug konferansesenter	21
3.1.2	Oppdrag 2 – Kalibrering og finn absolutt høyde over havet	22
3.1.3	Oppdrag 4 – Mål lufttrykket i klasserommet	22
3.2	Om du befinner deg på Realfagbygget på Gløshaugen	23
3.2.1	Oppdrag 1 – Mål relative høyder i Realfagbygget	23
3.2.2	Oppdrag 2 – Kalibrering og absolutt høyde over havet	23
3.2.3	Oppdrag 3 – Mål høyden på et av høyhusene på Gløshaugen	24
3.2.4	Oppdrag 4 – Mål lufttrykket i klasserommet	24
4	Måling av lufttrykk og beregning av høyden over havet	25
4.1	Måling av lufttrykk	25
4.1.1	Måling av lufttrykk ved endring i resistans (piezo-resistivitet)	25
4.1.2	Barometeret BMP180 (Bosch)	25
4.2	Måling av høyde basert på trykkmålinger	26
4.2.1	Kalibrering av høydemåleren	30
4.3	Målinger utført med BMP180	33
4.3.1	Registrering av små endringer i lufttrykk med BMP180	33
4.3.2	Innendørs målinger av trykk i ulike etasjer	35
5	Arduino UNO og innhenting av GPS-data	37
5.1	GPS-modulen	37
5.1.1	GY-NEO6MV2 Flight (GPS-modul)	37
5.1.2	ASX00017 - Arduino GPS Shield	38
5.2	Oppkobling av GY-NEO6MV2	39
5.3	Programmering av GPS for Arduino	40

5.4	Beregning av akkumulert distanse	42
5.5	Visualisering av GPS-data i Google Earth	43
5.5.1	Installasjon av Google Earth	44
5.5.2	Plotting av en trase i Google Earth	45
5.5.3	Editering av kml-fila	48
6	Referanser	51



1 Innledning

Mikrokontrollerkortet Arduino, med alle sine varianter, gir rike muligheter til å koble til sensorer og lagringsenheter av mange slag. Arduino UNO er kanskje den mest brukte i familien av kontrollerkort, men det finnes også mange andre, f.eks. Arduino Nano som er en god kandidat dersom det er viktig med små dimensjoner. En av fordelene med å bruke Arduino UNO er at det finnes et rikt utvalg av “shield”-kort, og som lett kan plugges sammen med UNO’en og gi utvidet funksjonalitet.

Ved bruk av mikrokontrollere og sensorer kan mange fagområder knyttes sammen. Det er f.eks. lett å knytte sensorteknologi til fysikk og kjemi (ev. materialteknologi), programmering og matematikk for beregning og kalibrering av målte størrelser, og i dette tilfellet, datalogging når man beveger seg i skog og mark.

I forbindelse med Fagfornyelsen 2020 er det bruk av digitalt verktøy og programmering for modellering av naturvitenskapelige fenomener. I dette tilfellet dreier det seg om modellering av sammenhengen mellom lufttrykk, høyde over havet og temperatur.

En kan se for seg følgende organisering av undervisningsopplegget i forbindelse med et elevverksted:

- Kort introduksjon til Arduino og trykksensorer (presentasjon)
- Sammenhengen mellom lufttrykk og høyde over havet (presentasjon)
- Oppbygging av måleutrustning med trykksensor, Arduino og display (presentasjon med lab.)
- Måle lufttrykket på stedet og skrive ut på display (ferdig programvare) (lab.)
- Legg inn en formel som beregner høyden, legg inn referansehøyden for lokaliteten
- Kalibrering og måling av høyden over havet og andre lokale høydeforskjeller, relative målinger
- Diskusjon og anvendelighet
- Kort introduksjon til måling med GPS og visualisering av GPS-data

Opplegget med trykk- og høydemålinger vil kunne gjennomføres i løpet av en 2 – 3 timers økt for elever som har litt kjennskap til programmering av Arduino.

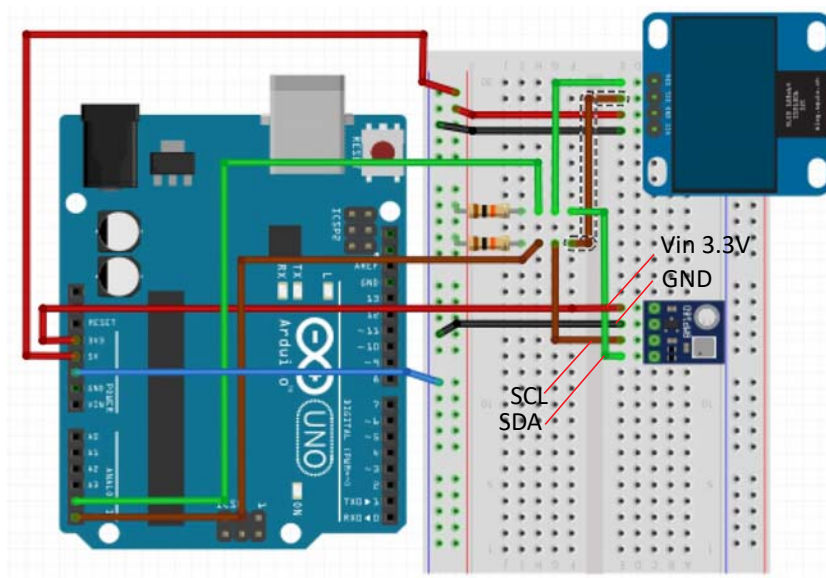
2 Montering og oppkobling for trykkmåling

I vårt måleoppsett har vi valgt å benytte en Arduino UNO og trykkmåleren/barometeret BMP180 som gir stor relativ nøyaktighet ved måling av lufttrykk. Vi bruker dessuten et SSD 1306 grafisk display og har valgt å benytte en batteriklemme for 9V batteri slik at vi kan ta med oss måleren uavhengig av PC'en.

2.1 Oppkobling

Figuren under viser oppkoblingen. Både BMP180 og SSD 1306 kommuniserer med Arduino'en via en I²C buss som er tilsluttet Arduino'en via de to analoge inngangene A4 (SDA - Data) og A5 (SCL - klokke). Kommunikasjonen skjer ved hjelp av to ledninger, en som overfører en klokkefrekvens og en som overfører data. For at denne skal fungere tilfredsstillende benyttes to motstander hver på 10 kΩ som er koblet til +5V.

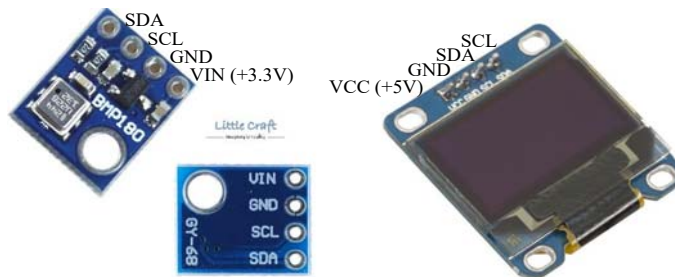
OBS! Merk at displayet er koblet til +5V og GND, mens trykkmåleren **BMP180 er koblet til +3,3 V og GND**. Dette er særdeles viktig for ikke å ødelegge trykkmåleren.



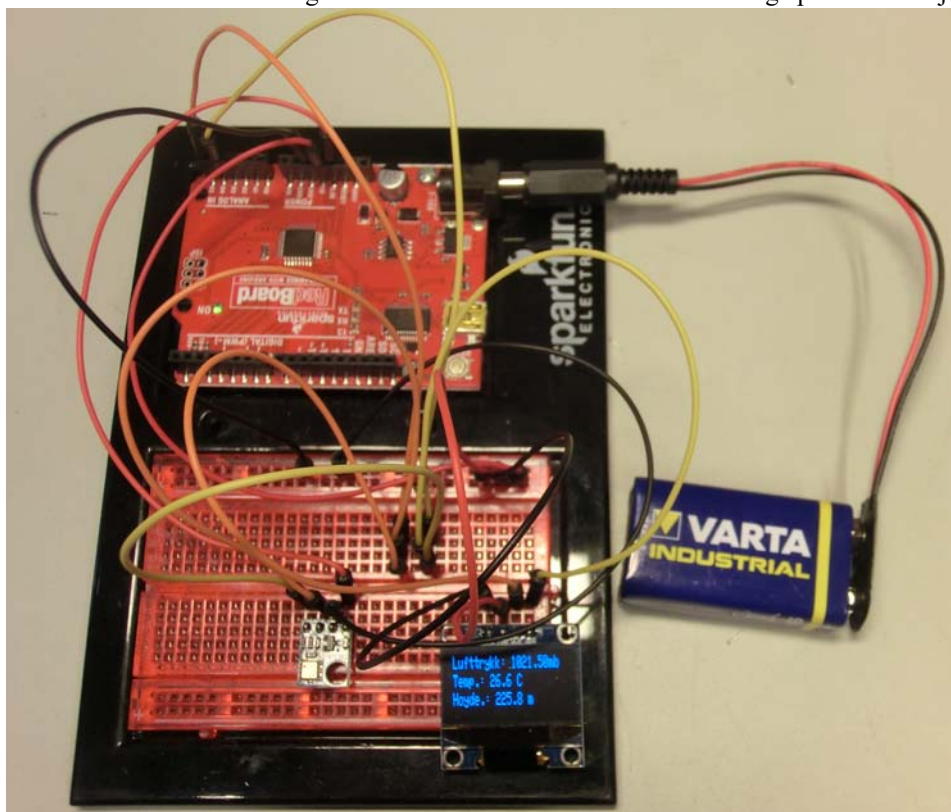
Merk at navnene på beina til BMP180 står på undersiden av kretsen.



Figuren under viser nærbilder av de to kretsene.



Oppkoblingen kan bli omtrent som på bildet under. Merk at man bruker kabler med de fargene man har. Det kan være vanskelig å være helt konsekvent mht til samme farge på samme linje.



Bruk overgangen fra batteriplugg til batterikontakt og et 9V batteri til å gjøre måleinstrumentet uavhengig av PC'en.

2.2 Kort innføring i bruk av Arduino editoren (IDE)

Dette er en meget kortfattet innføring i bruk av editoren til Arduino. Hopp over om dette er kjent.

1. Installasjon av programvaren

Det er normalt ikke nødvendig å installere en ny utgave dersom en gammel er installert i maskinen fra før. Eldre versjoner vil sannsynligvis fungere godt.

Programvaren hentes fra:

<https://www.arduino.cc/en/Main/Software>

Download the Arduino IDE



Velg: *Windows installer* (For Mac – *Mac OS X*)

Velg: *Just Download*

Velg: *Kjør*



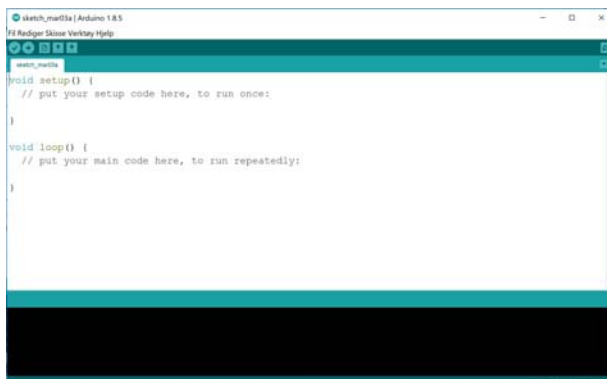
Velg: *Følg prosedyren i installasjonen*

Velg: *Å installere drivere*

2. Start programmet og installer tre biblioteker

Klikk på ikonet til Arduino og start programmet.

Du vil forhåpentlig se følgende programvindu åpne seg etter en liten stund.





For å kunne kommunisere med displayet (SSD1306) og trykkmåleren (BMP180), må vi installere tre biblioteker. Disse bibliotekene er komprimert i zip-filer og kan lastes ned fra:

<https://www.ntnu.no/skolelab/bla-hefteserie>

under fanen: *Måling av lufttrykk, høyde og posisjon*

Adafruit_SSD1306-master.zip

Adafruit-GFX-Library-master.zip

BMP180_Breakout_Arduino_Library-master.zip

Lagre bibliotekene i nedlastingskatalogen eller et annet sted der du finner dem igjen.

Bibliotekene installeres på følgende måte:

Åpen Arduino-editoren



Velg: *Skisse*

Velg: *Inkluder bibliotek*

Velg: *Legg til .zip bibliotek* (øverst i nedtrekksmenyen)

Finn biblioteket i katalogen *Nedlasting*

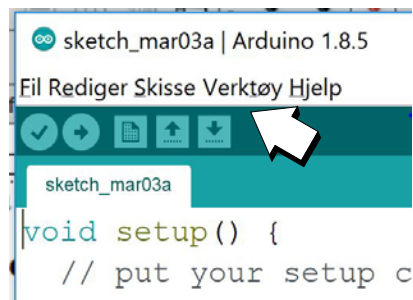
Velg: *Open*

... og biblioteket installeres

Gjenta det samme for alle tre bibliotekene.

3. Start programmet og koble Arduino'en til en USB-kontakt

Se på menylinjen øverst til venstre.



Velg: *Verktøy*

Velg: *Kort*

Velg: *Arduino/Genuino UNO*

Velg: *Verktøy*

Velg: *Port*


Velg: Det høyeste COM nummeret, eller velg porten merket med Arduino/Genuino UNO


Det skal nå være opprettet kontakt mellom programeditoren (IDE) og Arduinokortet


4. Kort brukerveiledning for program-editoren

De viktigste kommandoene for å betjene bruken av editoren er oppsummert under:




 Sjekk at koden er fri for skrivefeil (syntaksfeil)

 Kompiler og send koden til kortet

 Åpne en ny skisse og start med blanke ark

 Last opp en programskisse fra disken

 Lagre programskisse på disken

2.3 Programmering

Vi har laget et forenklet program for måling av lufttrykk og temperatur og skriving til displayet. Dersom du legger inn programmet slik det er, vil du få opp lufttrykk og temperatur på displayet. Dette er det lufttrykket som sensoren leverer her og nå.

Programkoden kan lastes ned fra siden:

<https://www.ntnu.no/skolelab/bla-hefteserie>

Under fanen “*Måling av lufttrykk, høyde og posisjon med Arduino*”.

```
// Programmet er beregnet for å måle barometertrykk og temperatur ved hjelp av  
// BMP180 og vise resultatet på et display av typen SSD1306.
```



```
// Inkludering av biblioteker
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h> // Display
#include <SFE_BMP180.h>      // Trykkmåling

// Deklarasjon av klasser
Adafruit_SSD1306 display(4); // Deklarasjon av enheten "display" av klassen
                             Adafruit_SSD1306
SFE_BMP180 pressure;        // Deklarasjon av enheten "pressure" av klassen
                             SFE_BMP180

void setup()
{
  Serial.begin(9600);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Initialiser display med adr.0x3C
  pressure.begin();
}

void loop()
{
  // Hent trykk- og høydeinformasjon

  double P,T;                // Deklarer variabler

  P = getPressure();         // Hent måleverdi av trykk
  T = getTemperature();     // Hent måleverdi av temperatur

  // Deklarerer variable for beregning av høyden
  float Hl=0;               // Referansehøyde i meter
  float Pl=1000.0; // Referansetrykk i mbar ved referansehøyde
  float Tl=0.0;             // Temperatur i grader C, husk at formelen krever grader Kelvin
  float a=0.0065; // TMemperaturgradient i K/m
  float R=287.06; // Den spesifikke gasskonstanten i J/kg K
  float g0=9.81; // Tyngdeakselerasjonen i m/s2

  // Legg inn referansetrykk, høyde og temperatur i formelen og beregn høyden

  //   Beregning av høyden skrives inn her,
  //   bruk parametrene som er deklart over
```

```

// Klargjør display for utskrift
display.clearDisplay();      // Slett informasjon på display
display.setTextSize(1);     // Sett størrelse på tekst
display.setTextColor(WHITE); // Hvit tekst på sort bakgrunn

// Skriver ut lufttrykk med to desimaler
display.setCursor(0,0);     // Plasser markør øverst til venstre
display.print("Lufttrykk: "); // Skriv "Lufttrykk
display.print(P,2);        // Skriv verdien til lufttrykk med to desimaler
display.println("mb");     // Skriv benevnning mbar (mb)

// Skriver ut temperatur med en desimal
display.setCursor(0,10);   // Flytt markør til neste linje
display.print("Temp: ");   // Skriv "Temp:", Temperatur
display.print(T,1);        // Skriv den målte temperaturen i ...
display.println(" C");     // ... grader C

// Skriv inn kode for å skrive ut høyden med en desimal på displayet
display.setCursor(0,20);   // Flytt markør til neste linje

// Utskrift av høyde til display skrives inn her
// Bruk kodelinjene over som eksempler

display.display();        // Overfør informasjonen til displayet og vis
delay(500);              // Vent i 500 msek.
}

double getPressure()
{
  char status;
  double T,P,p0,a;        // Deklarer lokale variable

  // For å foreta en trykkmåling så trengs temperaturen
  status = pressure.startTemperature(); // Forespørsel om temperaturmåling
  if (status != 0)        // Om forespørselen er vellykket, gå videre
  {
    delay(status);        // Vent til målingen er klar
    status = pressure.getTemperature(T); // Mål temperaturen
    if (status != 0)      // Om målingen er vellykket gå videre
    {

```




```
        status = pressure.startPressure(3); // Forespørsel om trykkmå-
ling //
Forespørsel om trykkmåling, (3) høyest presisjon
    if (status != 0) // Om forespørselen er vellykket, gå videre
    {
        delay(status); // Vent til målingen er klar
        status = pressure.getPressure(P,T);// Foreta trykk og temperaturmåling
        if (status != 0)
        {
            return(P);
        }
        else Serial.println("Feil ved forespørsel av trykkmåling\n");
    }
    else Serial.println("Feil ved måling av trykk\n");
}
else Serial.println("Feil ved forespørsel av temperaturmåling\n");
}
else Serial.println("Feil ved måling av temperatur\n");
}

double getTemperature()
{
    char status;
    double T;
    status = pressure.startTemperature(); // Forespørsel om temperaturmåling
    if (status != 0) // Om forespørselen er vellykket, gå videre
    {
        delay(status); // Vent til målingen er klar
        status = pressure.getTemperature(T);// Mål temperaturen
        if (status != 0) // Om målingen er vellykket gå videre
        {
            return(T);
        }
        else Serial.println("Feil ved forespørsel av temperaturmåling\n");
    }
    else Serial.println("Feil ved måling av temperatur\n");
}
```

Blokkdiagram for koden

Figuren til høyre gjengir et blokkdiagram av programmet slik at det skal være lettere å forstå koden.

Start

Programmet starter på nytt hver gang strømmen slås på eller man trykker på RESET-knappen på kortet. Den starter også på nytt når man slår på monitoren i programeditoren.

Biblioteker og deklarasjoner

De fleste av de kommandoene vi bruker når vi koder er funksjoner som kalles opp fra biblioteker. For å bruke display, trykksensor og kommunikasjon på linjene mellom kretsene (I²C), så kreves spesiellagede funksjoner, disse er samlet i biblioteker som må inkluderes i starten av programmet (`#include ...`).

Variabler som skal kunne brukes i alle funksjoner (globale variable) må deklarerer utenfor funksjonene. Disse deklarerer ofte her i starten.

Setup () – funksjonen

Setup() – funksjonen kjøres bare en gang, hver gang programmet starter. Her ligger initialisering som ikke trengs og gjentas mens programmet kjører.

Dette gjelder initialisering av kommunikasjon mellom kortet og monitoren i PC'en (`Serial.begin(9600);`).

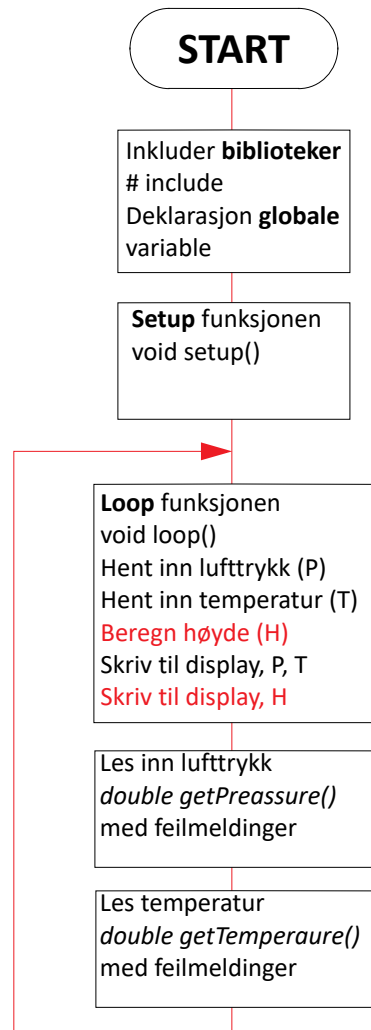
Videre gjelder det initialisering av displayet (`display.begin()`), og av trykkmåleren (`pressure.begin()`).

Loop () – funksjonen

Denne gjentar seg så lenge kortet har spenning. Når en kommer til slutten, så begynner programmet fra toppen av loop()-funksjonen igjen. For hver runde leses temperatur og trykk. Hvor ofte det gjøres målinger er hovedsakelig bestemt av hvor lang tid det tar å gå gjennom loopen. Kommandoen `delay(500);` på slutten av loopen legger inn en pause på 500 millisekunder. Det er hovedsakelig denne som bestemmer hvor ofte det tas målinger.

Avlesing av trykk og temperatur

Først leses trykk (`P = getPressure();`) og temperatur (`T = getTemperature();`). De avleste verdiene legges i variablene `P` og `T`.





Deklarasjon av variabler og konstanter

Dernest deklarerer en del lokale variabler (P_1 , H_1 , T_1)¹ og konstanter (a , R , g) som skal brukes når man legger inn formelen for beregning av høyden, H .

Beregning av høyden

I programskissen er det vist med en tekst hvor formelen for å beregne høyden kan legges inn.

Display resultater

De neste avsnittene inneholder kommandoer som skriver data til displayet. Dette er ulike kommandoer av typen, `display.print` og `display.println` o.l. Se forøvrig avsnittet Tips til programmering på side 20. Tanken er at når formelen for beregning av høyde legges inn så kan også den skrives ut på displayet.

double getPressure() – funksjon

Denne funksjonen leser av trykksensoren og sjekker om det oppnås kontakt med sensoren. Der-som det ikke oppnås kontakt, så skrives det ut en feilmelding på IDE-monitoren. For at funksjonen skal kunne beregne trykket må den også lese av temperaturen. Temperaturavlesningen er inkludert i funksjonen. Funksjonen returnerer verdien av lufttrykket som legges inn i variabelen P .

double getTemperature() – funksjon

Denne funksjonen leser av temperatursensoren som er inkludert i trykksensoren. Funksjonen returnerer verdien av temperaturen som legges inn i variabelen T .

2.4 Beregn høyden – legg inn formelen i programmet

Når vi har fått programmet til å oppføre seg som det skal, vil vi beregne høyden som funksjon av lufttrykket og temperaturen. Vi kan da bruke formelen²:

$$h = \frac{T_1}{a} \left(\left(\frac{p}{p_1} \right)^{\frac{aR}{g_0}} - 1 \right) + h_1 \quad (2.1)$$

Hvor:

h	Beregnet høyde i meter
p_1	Trykk i Pa ved referansehøyden h_1
h_1	Referansehøyden i meter
T_1	Temperatur ved referansehøyden h_1 i K(elvin)
p	Målt trykk i Pa

1. Ser at jeg har blandet store og små bokstaver når det gjelder programmet og formelen, men det er altså de samme parametrene.
2. Se lign. (4.2), side 28 for nærmere omtale.

a	Temperaturgradient, foreslått verdi -0,0065 K/m
g_0	Tyngdeakselerasjonen 9,81 m/s ²
R	Den spesifikke gasskonstant 287,06 J/kg K

Legg formelen inn i programmet på det markerte stedet. Bruk variablene som er definert foran i programmet. Legg også inn noen kodelinjer som skriver høyden til displayet med riktig tekst og benevning.

Tips til programmeringen for å beregne høyden

Aritmetiske funksjoner:

+	Addisjon
-	Subtraksjon
*	Multiplikasjon
/	Divisjon
<code>pow (grunntall, eksponent);</code>	Eksponent
<code>double log(double x);</code>	Naturlig logaritme, definer argument og resultat som <i>double</i>

for-loop(){ }

Dersom man ønsker å midle verdier over flere målinger kan man bruke en for-loop:

```
for (int i=0; i <= 255; i++){ }           // Denne går 256 ganger i for-loopen.
                                           // Det er det som står mellom klanne-
                                           // parentesene som gjentas 255 ganger
```

Funksjoner for skriving til lokalt display

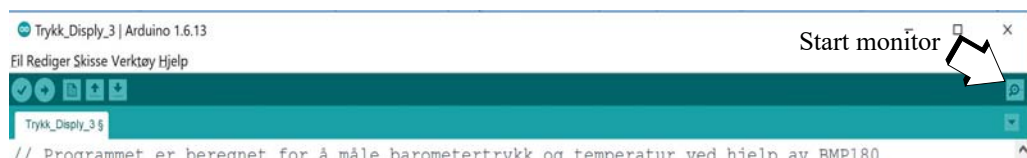
```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Initialiser display med adr.0x3C
display.clearDisplay();                     // Slett informasjon på display
display.setTextSize(1);                    // Sett størrelse på tekst
display.setTextColor(WHITE);               // Hvit tekst på sort bakgrunn
display.setCursor(0,10);                    // Flytt markør til neste linje
display.print("<tekst>");                   // Skriv ut tekst uten linjeskift
display.print(T,1);                         // Skriv den målte temperaturen m/en desimal
display.println("<tekst>");                 // Skriver ut tekst med linjeskift
display.display();                          // Overfør informasjonen til displayet og vis
```

Funksjoner for skriving til IDEs monitor

```
Serial.begin(9600) // Initialiser skriving til monitor m/9600baud
Serial.print("<tekst>");                     // Skriv ut tekst uten linjeskift
Serial.print(T,1);                          // Skriv den målte temperaturen m/en desimal
Serial.println("<tekst>");                   // Skriver ut tekst med linjeskift
```



Monitoren åpnes ved å trykke på:



2.5 Kalibrering

Dersom vi ønsker å måle absolutt høyde må vi kalibrere målingene. Dette kan vi gjøre ved å finne trykket hos en nærliggende målestasjon (p_1) ved en kjent høyde (h_1). Her kan vi anta at målingene er lufttrykket på stedet ved den angitte høyden og ikke er omregnet til havnivå.

Undersøk målestasjoner i nærområdet. Et godt sted å begynne er:

<http://www.xgeo.no/index.html?p=klima>

Offisiell portal for målestn. i Norge

Velg lufttrykk i tabellen under stasjoner og velg en nærliggende stasjon fra kartet. En vil da finne siste registrerte lufttrykk og stasjonens høyde over havet. Disse verdiene settes så inn i programmet som $P1$ og $H1$.

Forøvrig les mer om kalibrering i avsnitt 4.2.1, side 30.

3 Oppdragene

3.1 Om du befinner det på Bårdshaug i Orkanger:

3.1.1 Oppdrag 1 – Mål relative høyder inne på Bårdshaug konferansesenter

Ta utgangspunkt i den bygningen dere befinner dere i og mål relativ høyde for hver etasje, fra kjeller til toppetasje så lavt og høyt dere kommer. Sammenlign målingene med hva andre har fått.



3.1.2 Oppdrag 2 – Kalibrering og finn absolutt høyde over havet

Forsøk å kalibrer instrumentet etter beste evne ut fra data hentet inn fra ulike vær-stasjoner funnet på nettet (se side 30). Sammenlign målinger utført av andre som befinner seg på samme sted og se på avvik. Hvordan kan vi ev. forklare avvik i målinger gjort av absolutt høyde?

Kartet under er et utsnitt fra området rundt Bårdshaug Herregård.



Kartet er hentet fra:

<https://norgeskart.no/>

`#!?project=seeiendom&layers=1002,1013,1014,1015&zoom=16&lat=7029266.67&lon=241906.09&panel=searchOptionsPanel&sok=B%C3%A5rdshaug&markerLat=7029421.748763504&markerLon=241850.8050601559`

Gå inn i kartdatabasen og finn høyden over havet utenfor bygningen.

Hvilke avvik registrerer dere mellom høyden hentet fra kartdata og den dere måler etter kalibrering?

3.1.3 Oppdrag 4 – Mål lufttrykket i klasserommet

I dette oppdraget er oppgaven å registrere lufttrykket i klasserommet når man åpner og lukker døra. For at dette skal være mulig må man gjøre følgende:

1. Sample trykket så raskt som mulig (responstiden er maks. 7,5 msek)
2. Legge inn kommandoer som skriver data til monitoren
3. Sørg for å legge ut data slik at det lett lar seg kopiere inn i f.eks. Excel eller et annet analyseprogram. Skill gjerne verdiene med komma eller semikolon. Lag ny linje for hver måling.



3.2 Om du befinner deg på Realfagbygget på Gløshaugen

3.2.1 Oppdrag 1 – Mål relative høyder i Realfagbygget

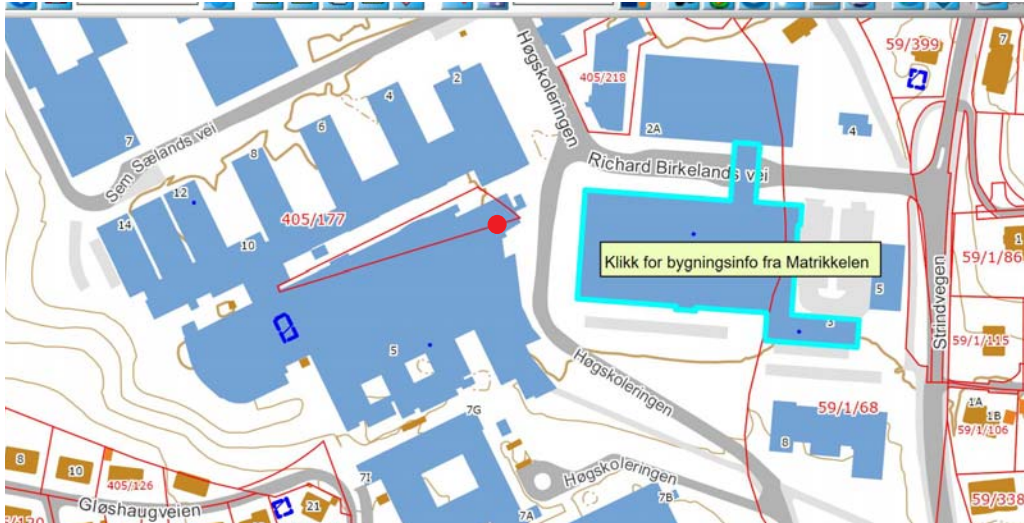
Ta utgangspunkt i den bygningen dere befinner dere i og mål relativ høyde for hver etasje, fra kjeller til toppetasje så lavt og høyt dere kommer. Sammenlign målingene med hva andre har fått. Bygget har 9 etasjer.



3.2.2 Oppdrag 2 – Kalibrering og absolutt høyde over havet

Forsøk å kalibrer instrumentet etter beste evne ut fra data hentet inn fra ulike vær-stasjoner funnet på nettet (se side 30). Sammenlign målinger utført av andre som befinner seg på samme sted og se på avvik. Hvordan kan vi ev. forklare avvik i absolutte målinger?

Kartet under er et utsnitt fra Gløshaugen. Skolelaboratoriet befinner seg nær den røde prikken.



Kartet er hentet fra:

<https://kart5.nois.no/trondheim/Content/Main.asp?layout=trondheim&time=1550437879&vw=asv>

Gå inn i kartdatabasen og finn høyden over havet utenfor bygningen. Ev. på laboratoriet dere befinner dere. Hvilke avvik registrerer dere i forhold til høyden fra kartdata og den dere måler etter kalibrering?

3.2.3 Oppdrag 3 – Mål høyden på et av høyhusene på Gløshaugen

Det høyeste punktet her på Gløshaugen er sannsynligvis på toppen av en av høyblokkene. Ta med dere måleinstrumentet å finn ut følgende:

- Mål absolutt høyde ved foten av høyblokka.
- Ta heisen opp i 13. etasje og finn den absolutte høyden der. Estimer høyden fra der dere klarer å komme til å måle og til toppen.
- Finn høyden av blokka fra foten til toppen.
- Sammenlign med andre som har gjort det samme.



3.2.4 Oppdrag 4 – Mål lufttrykket i klasserommet

I dette oppdraget er oppgaven å registrere lufttrykket i klasserommet når man åpner og lukker døra. For at dette skal være mulig må man gjøre følgende:

1. Sample trykket så raskt som mulig (responstiden er maks. 7,5 msek)
2. Legge inn kommandoer som skriver data til monitoren
3. Sørg for å legge ut data slik at det lett lar seg kopiere inn i f.eks. Excel eller at annet analyseprogram. Skill gjerne verdiene med komma eller semikolon. Lag ny linje for hver måling.



4 Måling av lufttrykk og beregning av høyden over havet

4.1 Måling av lufttrykk

4.1.1 Måling av lufttrykk ved endring i resistans (piezo-resistivitet)

Den piezo-resistive effekten er forskjellig fra den piezo-elektriske effekten. Den piezo-resistive effekten ble oppdaget av *Lord Kelvin* i 1856. Først i 1954 oppdaget C.G. Smith at germanium- og silisiumkrystaller hadde spesielt store variasjoner i ledningsevnen når de ble utsatt for mekanisk stress. Ledningsevnen til materialer er avhengig av mengden ladningsbærere i ledningsbåndet og hvor lett elektroner kan frigjøres fra valensbåndet. Dette er igjen avhengig av størrelsen på *båndgapet* mellom lednings- og valensbåndet i materialet. Når de nevnte materialene utsettes for stress, vil båndgapet endre seg og dermed også ledningsevnen.

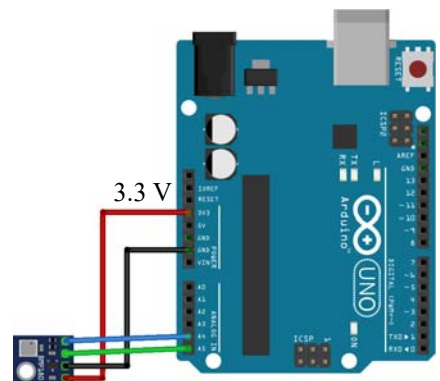
4.1.2 Barometeret BMP180 (Bosch)

BMP180 er en trykksensor som går under betegnelsen *barometer*. Kretsen erstatter BMP085. Senere er også BMT280 kommet på markedet. Selve sensoren er basert på piezo-resistivitet teknologi og levert av firmaet Robert Bosch. Kretsen leverer trykkdata via en digital I²C buss. I tillegg til trykksensoren inneholder kretsen en temperatursensor. Her er noen nøkkeldata:

- Måleområde: 300 – 1100 hPa (millibar) – som typisk tilsvarer 9 000 til –500 m
- Oppløsning, trykk: 0,01 hPa (0,01 mbar)
- Oppløsning, høyde: 0,25 meter
- Temp. nøyaktighet (abs.) $\pm 1\text{ }^{\circ}\text{C}$ (0 – 65 $^{\circ}\text{C}$), $\pm 0,5\text{ }^{\circ}\text{C}$ @ 25 $^{\circ}\text{C}$
- Oppløsning i temp.: 0,1 $^{\circ}\text{C}$
- Konverteringstid trykk: 7,5 ms (standard)
- Supplyspenning: 1,8 – 3,6 V
- Lavt effektforbruk: 5 μA ved 1 måling pr. sek.
- Responstid: 7,5 ms (maks)
- Standby strøm: 5 μA
- Langtidsstabilitet: $\pm 1\text{ hPa}$ pr. 12 måneder.
- Absolutt nøyaktighet – 4,0 – + 2,0 hPa (mbar)

Kretsen leveres fra bl.a. Sparkfun og er montert på et kretskort for lettere å kunne kobles til f.eks. en Arduino (“*breakout board*”).

For å lese dataene via I²C bussen brukes et spesielt bibliotek: `#include <wire.h>`



For mer informasjon se: <https://www.sparkfun.com/tutorials/253>, her finner du også databladet og programvare for Arduino.

Biblioteket som følger med BMP180 beregner også høyden på bakgrunn av trykkmålinger:

$$h = 44330 \cdot \left(1 - \left(\frac{p}{p_0} \right)^{\frac{1}{5,255}} \right) \quad (4.1)$$

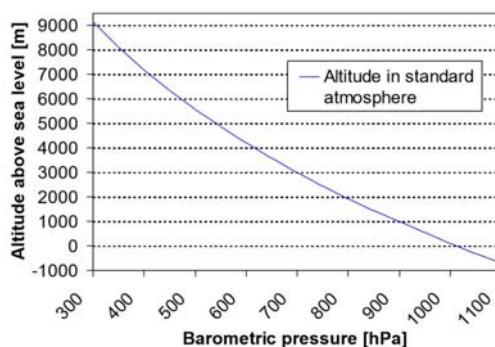
Hvor:

h Beregnet relativ høyde i forhold til referansehøyde ved p_0 (gjærne ved havnivå)

p Målt trykk

p_0 Målt referansetrykk

Figuren under viser en typisk sammenheng mellom trykk og høyde.



For en mer generell behandling, se avsnitt 4.2, side 26.

4.2 Måling av høyde basert på trykkmålinger

Trykk måles normalt i Pascal hvor $1 \text{ Pa} = 1 \text{ N/m}^2$.

Tidligere ble trykk målt i atmosfærer (atm), mmHg eller Bar.

En normalverdi for lufttrykket er:

$$1 \text{ atm} = 760 \text{ mmHg} = 1.01325 \text{ Bar} = 1013.25 \text{ mBar} = 101325 \text{ Pa} = 1013.25 \text{ hPa}$$

Vi legger merke til at h(ekto)Pa er det samme som m(illi)bar.

Lufttrykket er bestemt av tyngden til det "havet" av luft som vi befinner oss på bunnen av. Lufttrykket er derfor avhengig av mengden luft som til en hver tid befinner seg over hodet på oss. Vekta av luftmengden er avhengig av tyngdekraften, tykkelsen og tettheten til luftlaget, som igjen er avhengig av hvordan lufta forflytter seg og av temperaturen, dvs. værforholdene. Som vi ser er det mange faktorer å ta hensyn til. Likevel finnes det matematiske modeller som gjør at en kan



gjøre rimelig nøyaktige høydemålinger på bakgrunn av trykkmålinger. Imidlertid er kalibrering særdeles viktig i denne sammenhengen

En regner normalt at trykket faller med 1 millibar pr. 8 meter, eller ca 12.5 millibar pr. 100 meter. Dette stemmer ikke så verst for de første 2000 meter, deretter minker trykket mindre for hver 1000 meter.

Normalt refereres alle trykkmålinger til havnivået. En meteorologisk stasjon som oppgir barometerstand ved stasjonen, har ofte regnet om verdiene til havnivået, dette er ikke tilfellet hos det anbefalte nettstedet: <http://www.xgeo.no/index.html?p=klima>

Tabellen under viser typiske verdier for sammenhengen mellom trykk, lufttetthet, temperatur og høyde over havet.

HoH	Temperatur	Lufttrykk	Tetthet	
(m)	(C)	(hPa)	(kg/m ³)	
0000	15.0	1013	1.2	
1000	8.5	900	1.1	
2000	2.0	800	1.0	(Galdhøpiggen)
3000	-4.5	700	0.91	
4000	-11.0	620	0.82	
5000	-17.5	540	0.74	
6000	-24.0	470	0.66	
7000	-30.5	410	0.59	
8000	-37.0	360	0.53	
9000	-43.5	310	0.47	(Mount Everest)
10000	-50.0	260	0.41	(Marsjhøyde rutefly)
11000	-56.5	230	0.36	
12000	-56.5	190	0.31	
13000	-56.5	170	0.27	
14000	-56.5	140	0.23	
15000	-56.5	120	0.19	
16000	-56.5	100	0.17	
17000	-56.5	90	0.14	
18000	-56.5	75	0.12	
19000	-56.5	65	0.10	
20000	-56.5	55	0.088	
21000	-55.5	47	0.075	
22000	-54.5	40	0.064	
23000	-53.5	34	0.054	
24000	-52.5	29	0.046	
25000	-51.5	25	0.039	
26000	-50.5	22	0.034	
27000	-49.5	18	0.029	
28000	-48.5	16	0.025	
29000	-47.5	14	0.021	
30000	-46.5	12	0.018	
31000	-45.5	10	0.015	
32000	-44.5	8.7	0.013	
33000	-41.7	7.5	0.011	
34000	-38.9	6.5	0.0096	
35000	-36.1	5.6	0.0082	

Omregningen fra trykk til høyde må også ta hensyn til temperaturen. Temperaturen vil dessuten forandre seg med høyden.

Det er normalt lettere å forholde seg til en omregningsformel enn en tabell. Ulempen med en formel er at de mange parametrene kan gi stor usikkerhet i beregningen. Følgende formel er ikke uvanlig å bruke:

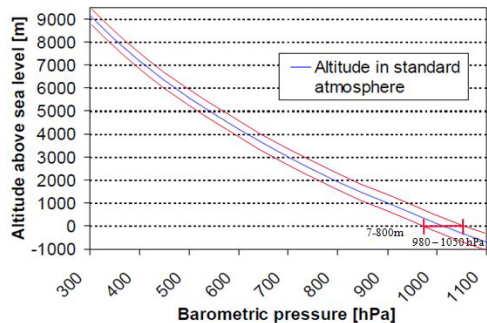
$$h = \frac{T_1}{a} \left(\left(\frac{p}{p_1} \right)^{\frac{aR}{g_0}} - 1 \right) + h_1 \quad (4.2)$$

Hvor:

- h Beregnet høyde i meter
- h₁ Starthøyde i meter
- T Temperatur i Kelvin
- T₁ Starttemperatur i høyden h₁
- a Temperaturgradient, foreslått verdi -0,0065 K/m
- p Målt trykk i Pa
- p₁ Trykk i Pa ved starthøyden
- g₀ Tyngdeakselerasjonen 9,81 m/s²
- R Den spesifikke gasskonstant 287,06 J/kg K

Denne formelen kan enten legges inn i datainnsamlingsenheten vår, men det er bedre at den legges inn i programvaren som behandler dataene. Har man rådataene fra måleenheten, er muligheten for etterbehandling enn om man bare har de omregnede dataene.

Diagrammet til høyre viser sammenhengen mellom trykk og høyde med økende høyde over havnivået. Normale variasjoner i lufttrykket ved bakken kan være fra 980 hPa til 1050 hPa (millibar). Denne naturlige variasjonen kan derfor gi en absolutt endring i høydeberegningen på 7 – 800 meter dersom man ikke kalibrerer på bakkenivå.



Vi skjønner derfor at det er svært viktig med hyppig kalibrering dersom man skal kunne stole på høydemåleren.

Det finnes imidlertid flere ulike modeller for sammenhengen mellom trykk og høyde. Her er en alternativ³ modell hentet fra en teknisk note publisert av firmaet Vaisala som utvikler instrumenter for værobservasjoner og værstasjoner.

3. http://www.vaisala.fi/Vaisala%20Documents/Measurement%20Theory/height_calculation.pdf



$$h = \left(\frac{R}{g}\right) T_m \ln\left(\frac{p_0}{p}\right) \quad (4.3)$$

Hvor:

- h Beregnet høyde over havet
- R Den spesifikke gasskonstant 287,06 J/kg K
- g Tyngdeakselerasjonen 9,81 m/s²
- T_m Gjennomsnittlig lufttemperatur i det aktuelle måleområde
- p_0 Lufttrykk ved havnivået
- p Målt lufttrykk på det aktuelle stedet

Det finnes også omregningskalkulatorer på nett. Firmaet MIDE Engineering Solutions har en på nettsiden: <https://www.mide.com/pages/air-pressure-at-altitude-calculator>.

Her skriver man inn trykk og temperatur ved havnivå (fersk-vare) i tillegg til målt trykk på den lokasjonen man befinner seg på og man får et estimat av høyden på stedet. Det er viktig at man bruker en referanseshøyde ved havnivå som er i nærheten av stedet der man befinner seg og det er ikke alltid så lett.

Calculate Altitude from Air Pressure

Pressure at Sea Level Pa

Temperature °C

Air Pressure at Altitude Pa

CALCULATE

Altitude = m

Tilsvarende kan man finne et estimat av trykket der man befinner seg dersom man kjenner trykket og temperaturen ved havnivået og i hvilken høyde man befinner seg.

Calculate Air Pressure at Altitude

Pressure at Sea Level Pa

Temperature °C

Altitude m

Air Pressure at Altitude = Pa

4.2.1 Kalibrering av høydemåleren

En høydemåler basert på måling av lufttrykk kan kalibreres på flere ulike måter:

- Ved at man kjenner lufttrykket og høyden over havet til stedet der man starter målingen. I så fall kan man benytte lign. (4.2). Denne metoden er aktuell når man skal sende opp en sonde eller gå en tur i terrenget, og og ønsker absolutt høyde over havet.
- Ved at man nullstiller høydemåleren på bakkenivå, eller fra det stedet der man ønsker å måle høyden i forhold til. Denne måten vil gi en relativ høydemåling i forhold til starthøyden for måleserien.
- Ved at man finner en offisiell målestasjon i nærheten, for eksempel en flyplass eller en havn, som viser en oppdatert verdi av lufttrykket. Slike målestasjoner oppgir også deres høyde over havet, ev. at de referer målingene sine til havnivået. Det er imidlertid viktig å finne ut hvilken praksis de følger. Ulempen med flere slike målestasjoner er at de ikke viser kontinuerlige målinger, men oppdaterer målingene for eksempel hver time eller hvert døgn. Dessuten kan de være et stykke unna stedet der man befinner seg. En slik kalibrering burde imidlertid gi en rimelig verdi for absolutt høyde over havet.

Uansett hvilken metode som benyttes så man foreta relativt hyppige kalibreringer siden lufttrykket kan endre seg svært raskt.



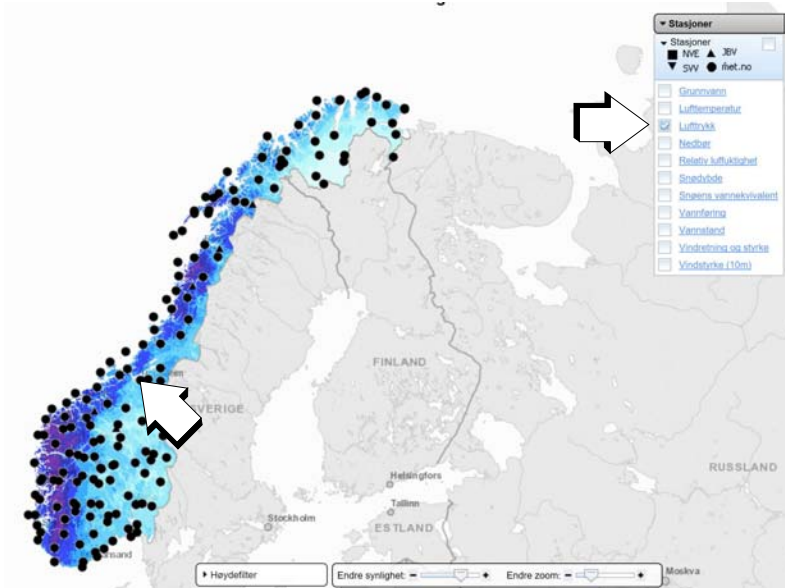
Her er noen eksempler på slike målestasjoner:

[se.norge.no](http://www.xgeo.no/index.html?p=klima) (<http://www.xgeo.no/index.html?p=klima>)

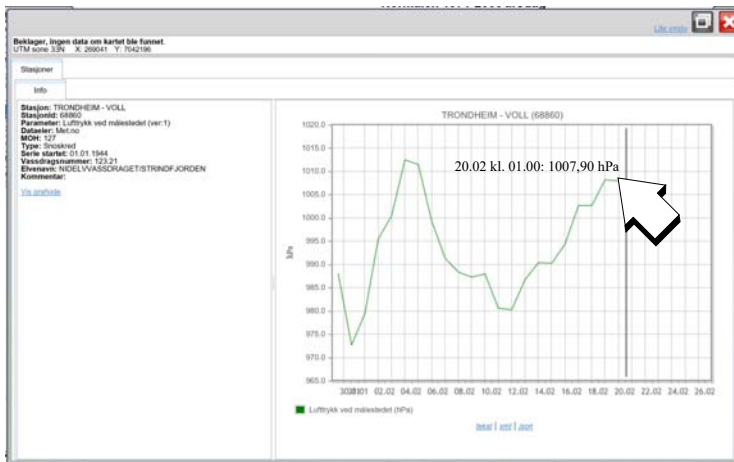
Dette er en åpen portal på Internett, som viser daglig oppdaterte kart over snø-, vær- og vannforhold og klima for Norge – og mye mer.



Velg så stasjoner og kryss av for “Lufttrykk” og velg en målestasjon nær der du befinner deg.



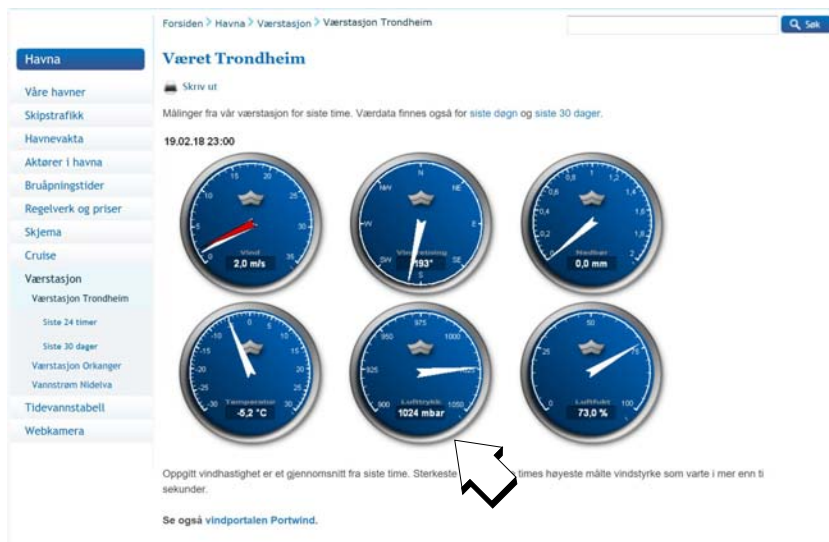
Etter å ha valgt målestasjon får man opp en graf som viser lufttrykket siste døgn. Ulempen er imidlertid at målingene kun oppdateres en gang i døgnet. Det kan derfor være store avvik mellom avlest verdi og virkelig verdi for lufttrykket på det ønsket tidspunkt.



Værstasjoner

En kan også gå inn på oversikten over norske værstasjoner. Det finnes en uoffisiell oversikt på <http://www.bjonnes.net/weatherstations/>. Det er imidlertid noe varierende om disse stasjonene leverer informasjon om lufttrykk. Følgende <https://portwind.no/> gir oversikt over vindretning og vindstyrke i et utvalg havner i Norge. Imidlertid har de fleste utelukket informasjon om lufttrykk.

Under er vist et eksempel fra Trondheim havn som også måler lufttrykket. Denne oppdateres hver time, men oppgir målingen på nærmeste hele mbar.





4.3 Målinger utført med BMP180

Her skal vi vise to enkle forsøk utført med Arduino og BMP180.

4.3.1 Registrering av små endringer i lufttrykk med BMP180

For å illustrere hvor følsom BMP180 er så har vi gjort følgende enkle forsøk:

Vi har latt Arduino med BMP180 måle trykket i rommet og latt programmet skrive trykket til monitoren med følgende enkle programkode:

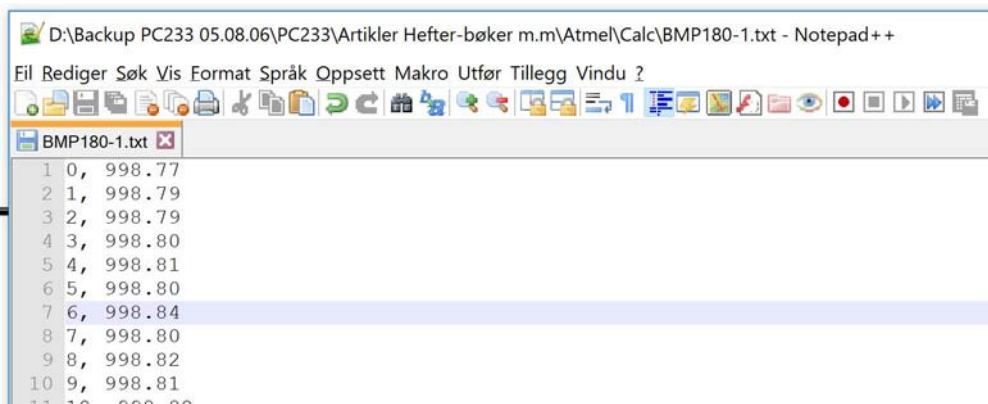
```
Serial.print(i);  
  Serial.print(", ");  
  Serial.println(P, 2);  
  i = i+1;  
  delay(10); // Måler med 10 msek. mellomrom
```

På denne måten vil vi få en lang rekke trykkmålinger skrevet ut i monitoren omtrent som utsnittet under:

```
0, 998.77  
1, 998.79  
2, 998.79  
3, 998.80  
4, 998.81  
5, 998.80  
6, 998.84  
...
```

I alt tok vi nær 500 målinger av trykket i løpet av ca. 30 sek. Til venstre for trykket er en tellevariabel som forteller oss nummeret til målingen. Vi må også passe på å legge inn et skilletegn mellom de to tallene på samme linje. Her har vi valgt komma, men et semikolon kan være bedre.

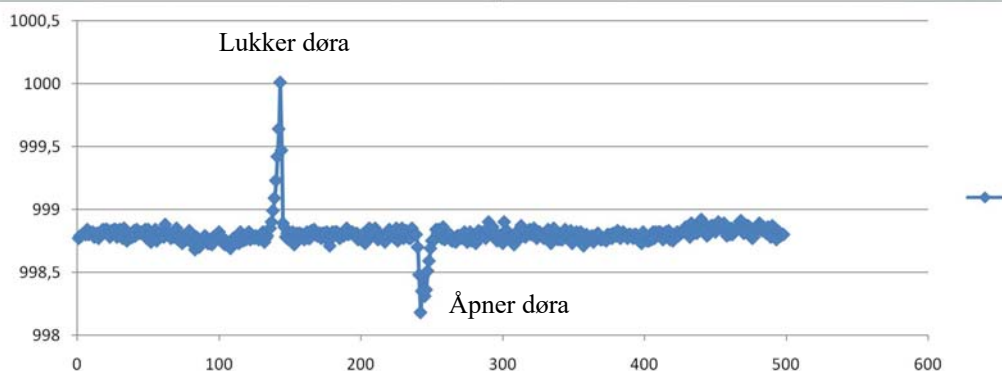
Vi kopierte tallrekken med programmet NotePad som vist på figuren under. Det fine med NotePad er at den kopierer data uten å trekke noe fra eller legge noe til.



Derne st lagrer vi dataene i en tekstfil som vi f.eks. kan importere til Excel. Dataene blir liggende som to kolonner som vist under. Her har vi sørget for at desimaltegnet er et komma, som er viktig for at Excel skal oppfatte verdiene som tall og ikke som tekst.

Type	Data	Diagram	
Diagram 3			
A	B	C	D
1	0	998,77	
2	1	998,79	
3	2	998,79	
4	3	998,8	
5	4	998,81	
6	5	998,8	
7	6	998,84	
8	7	998,8	

Dermed kan vi plote dataene som funksjon av tiden. På figuren under ser vi hvordan trykket endrer seg med tiden. Vi legger også merke til to topper, en som rager opp over gjennomsnittet og en som stikker ned under gjennomsnittet. Disse to avvikene skyldes at vi lukket døra til rommet for så å åpne den igjen. Ser vi nøye på grafen vil vi oppdage at utsvingene er relativt beskjedene, men rager godt opp over støyen.



Det må innrømmes at forsøket ble gjort i et relativt lite rom og med kraftige bevegelser av døra.

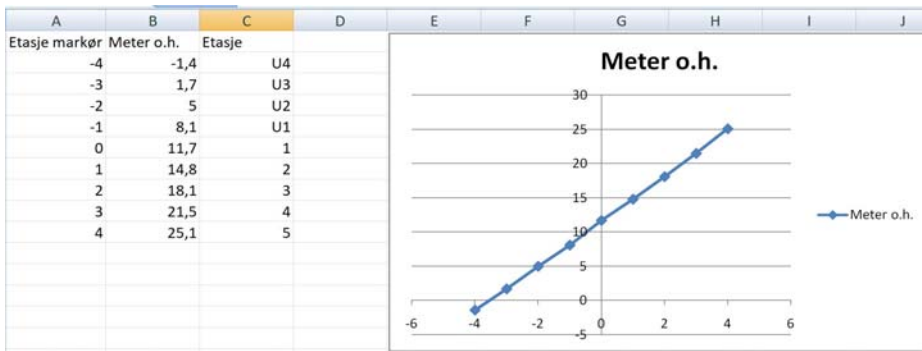


4.3.2 Innendørs målinger av trykk i ulike etasjer

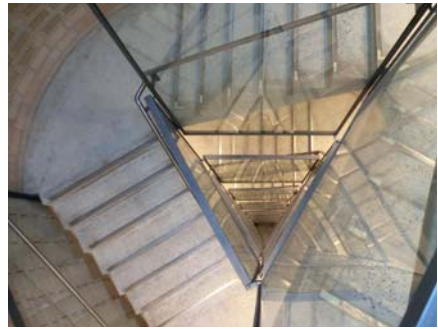
Disse målingene er gjort innendørs i Realfagbygget ved NTNU. Dette bygget har 9 etasjer hvorav fire er under bakken. Det er foretatt ni målinger ved samme høyde over gulvet i hver av de ni etasjene. Disse er lagt inn i et regneark.



Figuren under viser resultatene av måleserien, dvs. sammenhengen mellom etasje og relativ høyde. Vi ser at den relative nøyaktigheten er rimelig god, men den absolutte er feil, da det kan se ut til at underetasje U4 ligger under havets overflate. Dette skyldes hovedsakelig at utrustningen ikke er godt nok kalibrert. For å få en mest mulig stabil måling har vi midlet lufttrykket over 100 enkeltmålinger. Det vil si at hver måling tar ca. 1 – 2 sekunder. Likevel ser vi at målingene varierer med 50 – 60 cm fra måling til måling ved samme høyde. Legg også merke til at vi har latt første etasje være etasje 0 slik at “x-aksen” blir riktig, dette problemet slipper man f.eks. i England.



Bildet til høyre viser trappesjakta der målingene ble gjort. Det ble også foretatt målinger med laser. Denne målingen gikk fra målehøyden i øverste etasje og ned til gulvet i U4. Denne viste en total høyde på 31.2 meter. Dersom vi legger til høyden fra målepunktet og ned til gulvet i U4 så får ved en total høyde på 26,3 m som er et betydelig avvik fra lasermålingen. Her er det rom for diskusjon og videre utforskning for å finne avviket. Skyldes det lufttrykkmålingen eller lasermålingen. Kanskje kontrollmålingen burde ha vært gjort med målebånd.



5 Arduino UNO og innhenting av GPS-data

5.1 GPS-modulen

GPS eller *Global Positioning System* består av 24 satellitter som kretser omkring jorda med en omløpstid på 11 t 58 min. i en høyde av ca. 20200 km over jordoverflata. Normalt vil dette antallet være tilstrekkelig for, til enhver tid, å ha fri sikt til 8–10 satellitter i åpent terreng. Hver satellitt sender ut et kodet tidssignal som mottas av mottakerne. I tillegg til å inneholde informasjon om nøyaktig tid, inneholder signalet tidspunkt for utsendelse og en lang kode som mottakerne bruker for nøyaktig å bestemme tidspunktet for mottatt signal. På denne måten kan mottakersystemet bestemme tiden hvert av signalene bruker fra hver av satellittene til mottakeren. De målte tidsforsinkelsene brukes så til å beregne posisjonen til mottakeren [1].

En **GPS-modul** er et sett av GPS mottakere som gjør det mulig å følge flere GPS-satellitter samtidig. Fire er et minimum for å kunne beregne koordinater pluss høyde, og seks er ikke uvanlig.

5.1.1 GY-NEO6MV2⁴ Flight (GPS-modul)

Teknisk spesifikasjon

Dette er en vanlig brukt GPS-modul. Den kan bl.a. kjøpes fra www.kultogbillig.no for kr. 119,- inkl. MVA. Enkelte e-bay-forhandlere⁵ selger dem helt ned til kr. 35,-. Den består av to separate enheter forbundet med en koaksialkabel. Det ene er antennemodulen, den andre analyse- og datamodulen. Enkelte GPS-moduler har også antenner som er langt mindre enn den på bildet til høyre. Disse fungerer også men har større problemer med å låse til satellittene når antennen er i bevegelse.

- Enheten er bygget opp rundt en ARM-prosessor
- Mottakerfrekvens: 1575,42 MHz
- Antall kanaler: 50 kanaler
- Følsomhet: -161dBm ("Tracking & Navigation")



4. Datablad: [https://www.openimpulse.com/blog/wp-content/uploads/wpsc/downloadables/GY-NEO6MV2-GPS-Module-Datasheet.pdf](https://www.openimpulse.com/blog/wp-content/uploads/wp-content/uploads/wpsc/downloadables/GY-NEO6MV2-GPS-Module-Datasheet.pdf)

5. <https://www.ebay.com/itm/GYNEO6MV2-GPS-Module-NEO-6M-GY-NEO6MV2-Board-with-Antenna-for-Arduino-New-/311659967066>

- Nøyaktighet: - Horisontal: 2,5 m
- Hastighet: 0,1 m/s
- Kurs retning: 0,5°
- Tid: 30 ns (tidspulssignalets nøyaktighet)
- Dato: Leverer data og tid ned til sekunder
- Låsetid: Varm (Hot) start: 1 sek. i gjennomsnitt
Kald start: 27 sek. i gjennomsnitt
- Maks verdier: Maks. målehøyde: 50 000 meter
Maks. hastighet 500 m/s
Maks. akselerasjon 4 g
Maks. oppdatering 5 Hz
- Forsyning: Spenning: 2,7 – 3,6 V DC
Strømforbruk: Max. 47 mA
- Dig. utgang: Spenningsnivå: TTL nivå (0 – 2,85 V)
UART, USB (12 Mbit/sek), SPI (100 kbit/sek)
9 600 baud (symboler/sek.)
Format: NMEA GSV, RMC, GSA, GGA, GLL, VTG, TXT
- Størrelse: 25 x 25 x 8 mm (antennemodul) +
25 x 35 x 5 mm (kontrollmodul)
- Temperaturområde: –40°C til 85°C
- Backup batteri: For å holde data ved “Power down”

5.1.2 ASX00017 - Arduino GPS Shield⁶

Dette er en GPS-mottaker som bl.a. selges hos ELFA DISTRELEC til en stykkpris á kr. 242,- Denne er montert på et lite kretskort (25 x 45 mm) og inkluderer elektronikk, antenne og et knappebatteri. Kortet er forberedt for å motta signaler fra alle tre satellittsystemene: GLONASS, Galileo og GPS. Kortet arbeider på 3,3V og vil ikke tåle å bli koblet til 5V.

Kretsen kommuniserer enten direkte ned på et ASX00029 Arduino MKR WAN 1310 eller via I2C-buss. Biblioteket kan lastes ned fra Arduinos egne sider: <https://www.arduino.cc/en/Reference/ArduinoMKRGPS>

ASX00017



ASX00029



6. <https://www.arduino.cc/en/Guide/MKRGPSShield>

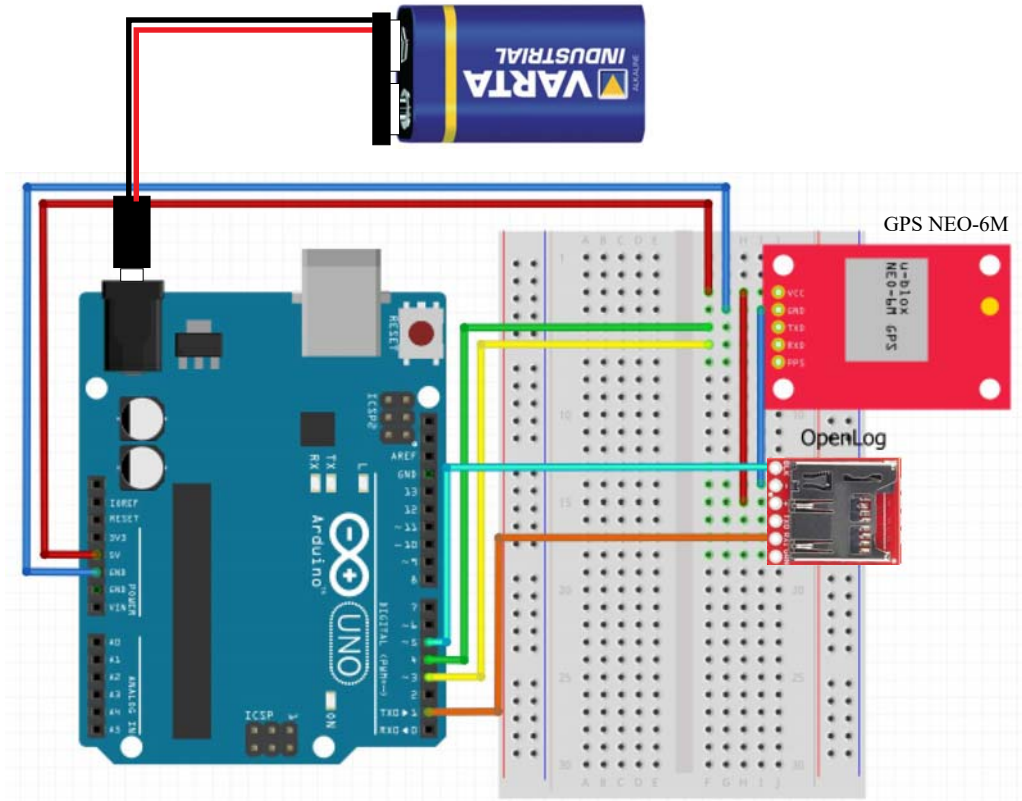


5.2 Oppkobling av GY-NEO6MV2

Kontroll-modulen kan kobles til Arduino UNO via fire terminaler:

V _{CC}	3,3 V
Rx (GPS)	Pin 3 (fungerer som Tx – UNO via spenningsdeler)
Tx (GPS)	Pin 4 (fungerer som Rx – UNO)
GND	GND

Ved å lodde en stiftlist til terminalene kan en lett koble opp modulen på et koblingsbrett for testing som vist i figuren under⁷.



Denne varianten av GPS-modulen benytter 5 V spenningsforsyning slik det er lett å koble GPS-modulen opp mot UNO'en.

7. <http://arduinstuff.blogspot.no/2014/05/neo6mv2-gps-module-with-arduino-uno-how.html>

5.3 Programmering av GPS for Arduino

Lagring av data fra GPS-modul GY-NEO6MV2

For omtale av GPS-kretsen og oppkobling se avsnitt 5.1.1, side 37.

I dette avsnittet skal vi hente inn data fra GPS-enheten GY-NEO6MV2 og lagre disse på SD-kortet. Alternativt kan en skrive ut GPS dataene i monitoren til Arduino editoren og kopiere dem over i en fil, i så fall må man bære med seg PC'en.

Installasjon av bibliotek

Biblioteket *TinyGPS* må installeres for å kunne hente ut data fra GPS-kontrollkortet. I tillegg må vi installere *SoftwareSerial* for å kunne kommunisere på to valgte datalinjer (Tx/Rx). I vårt tilfelle velges disse som dataportene D3 (Tx) og D4 (Rx) på Arduino-kortet.

Biblioteket kan hentes fra:

<https://github.com/mikalhart/TinyGPS/releases/tag/v13>

Hent ned zip-fila som heter: *Source code (zip)*. Ikke pakk ut fila, men lagre den på et sted du kan finne den igjen. Biblioteket installeres ved å velge Skisse/Include Library/Add .zip library i Arduino editoren (IDE) og hent opp fila⁸.

Enkleste kode

Foretrukket format på koordinat- og høydedataene er:

lengdegrad,breddegrad,høyde

dersom man senere ønsker å matte koordinatene inn i f.eks. Google Earth. Alle verdier er oppgitt som desimaltall og uten mellomrom etter komma.

Koden under viser den enkleste utgaven av programvaren for å hente inn koordinatdata og skrive til både monitoren i Arduino (IDE) og til SD-kortet:

```
#include <SoftwareSerial.h>
#include <TinyGPS.h>
/* This sample code demonstrates the simplified use of a TinyGPS object.
   It requires the use of SoftwareSerial, and assumes that you have a
   9600-baud serial GPS device hooked up on pins 4(Rx) and 3(Tx).
*/
TinyGPS gps;                               // Definerer en TinyGPS type kalt gps
SoftwareSerial ss(4,3);                     // Definerer en SoftwareSerial type kalt ss
static void smartdelay(unsigned long ms);

void setup()
{
  _____
```

8. Ev. se <https://github.com/mikalhart/TinyGPS/releases/tag/v13>



```
Serial.begin(115200);           // Sett datahastighet til monitor
ss.begin(9600);                // Sett datahastighet til GPS-kontrollkort
}

void loop()
{
  float flat, flon;            // Definer variable for lengde- og
  breddegrad                  // breddegrad
  int year;                    // Definer variable for år, dato og tid
  byte month, day, hour, minute, second, hundredths;
  unsigned long age;          // Definer variabel for age

  gps.f_get_position(&flat, &flon, &age); // Henter breddegrader og lengdegrader
  Serial.print(flon,6); // Skriver ut lengdegrader i grader desimalt
  Serial.print(", "); // Sett inn komma som skilletegn
  Serial.print(flat,6); // Skriver ut breddegrader i grader desimalt
  Serial.print(", "); // Sett inn komma som skilletegn
  Serial.print(gps.f_altitude(), 2); // Skriver ut høyde over havet i meter
  Serial.println();

  smartdelay(1000);
}

static void smartdelay(unsigned long ms)
{
  unsigned long start = millis(); // Hent antall millisekunder siden reset av
  Arduino-kortet
  do // Testen gjøres etter loopen er kjørt
  {
    while (ss.available())gps.encode(ss.read());
  }
  while (millis() - start < ms); // Testen gjøres før loopen er kjørt
}
```

Etter innsamling av data kan disse lastes opp i f.eks. Notepad++ eller i Excel. Dersom man ønsker å visualisere traseen i Google Earth så klippes dataene inn i kml-koden vist i avsnitt 5.5, side 43.

Kode med feilsjekk og flere data

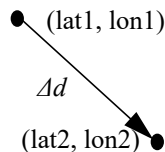
Den enkle programkoden omtalt i forrige avsnitt inneholder ingen sjekk mht. om dataoverføringen mellom GPS-enheten og Arduino er korrekt.

5.4 Beregning av akkumulert distanse

Det finnes i dag en rekke App'er for pad'er og smarttelefoner som beregner gangavstand etter som man beveger seg i gater eller i terrenget, disse er basert på smarttelefonens eller padens innebygde GPS mottaker. Et eksempel på en slik er Runkeeper⁹.

Har vi først koblet en GPS-mottaker til vår Arduino, så er det også mulig å beregne tilbakelagte distanser. Siden høydemålingene er relativt usikre så nøyer vi oss med å bruke lengde- og breddegradene.

Som vist på figuren over så har vi to sett med koordinater (lat_1, lon_1 og lat_2, lon_2) og ønsker å beregne avstanden, Δd , mellom disse to settene av koordinater. Siden jorda er krummet som en kule så er ikke dette en triviell beregning. Under er gjengitt formelapparatet for beregningen¹⁰:



$$\varphi_1 = \frac{\pi \cdot lat_1}{180} \quad (5.1)$$

$$\varphi_2 = \frac{\pi \cdot lat_2}{180} \quad (5.2)$$

$$\Delta\varphi = \varphi_2 - \varphi_1 \quad (5.3)$$

$$\Delta\lambda = \pi \cdot \frac{(lon_2 - lon_1)}{180} \quad (5.4)$$

$$a = \sin\left(\frac{\Delta\varphi}{2}\right)^2 + \cos(\varphi_1) \cdot \cos(\varphi_2) \cdot \sin\left(\frac{\Delta\lambda}{2}\right)^2 \quad (5.5)$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}) \quad (5.6)$$

$$\Delta d = R \cdot c \quad (5.7)$$

Hvor:

φ_1	= Breddegrad posisjon 1 (lat_1) målt i radianer
φ_2	= Breddegrad posisjon 2 (lat_2) målt i radianer
atan2	= Arkus tangens med to argumenter for å beholde informasjon om kvadrant
Δd	= Distansen i meter
R	= Jordradien i m ($6,371 \cdot 10^6$ m)

Dette er en relativt nøyaktig beregning, men kan være litt plundrete da standardbiblioteket til Arduino f.eks. ikke tilbyr $\text{atan2}()$. Imidlertid finnes det en forenklet versjon som kan være nyttig når man ikke trenger stor nøyaktighet eller når avstandene mellom målepunktene er liten.

9. <https://runkeeper.com/>

10. <http://www.movable-type.co.uk/scripts/latlong.html>



Ekvirektangulær tilnærming

I vårt tilfelle er avstandene noen meter slik at det skulle være helt uproblematisk å bruke den forenklete beregningsmetoden:

$$\varphi_1 = \frac{\pi \cdot lat_1}{180} \quad (5.8)$$

$$\varphi_2 = \frac{\pi \cdot lat_2}{180} \quad (5.9)$$

$$\Delta\lambda = \pi \cdot \frac{(lon_2 - lon_1)}{180} \quad (5.10)$$

$$x = \Delta\lambda \cdot \cos\left(\frac{(\varphi_1 + \varphi_2)}{2}\right) \quad (5.11)$$

$$y = \varphi_2 - \varphi_1 \quad (5.12)$$

$$\Delta d = R \cdot \sqrt{x^2 + y^2} \quad (5.13)$$

Hvor:

- φ_1 = Breddegrad posisjon 1 (lat_1) målt i radianer
- φ_2 = Breddegrad posisjon 2 (lat_2) målt i radianer
- Δd = Distansen mellom punktene i meter
- R = Jordradien i m ($6,371 \cdot 10^6$ m)

Akkumulert distanse

Den akkumulerte distansen finner vi da enkelt ved å summer opp alle de små distansene mellom hvert målepunkt. Vi må passe på at målepunktene ikke blir for sjeldne slik at vi mister de fine detaljene i registreringen.

$$d = \sum_1^n \Delta d \quad (5.14)$$

Hvor

- d = Lengden av traseen, dvs. summen av de mange små avstandene
- n = Antallet målepunkter

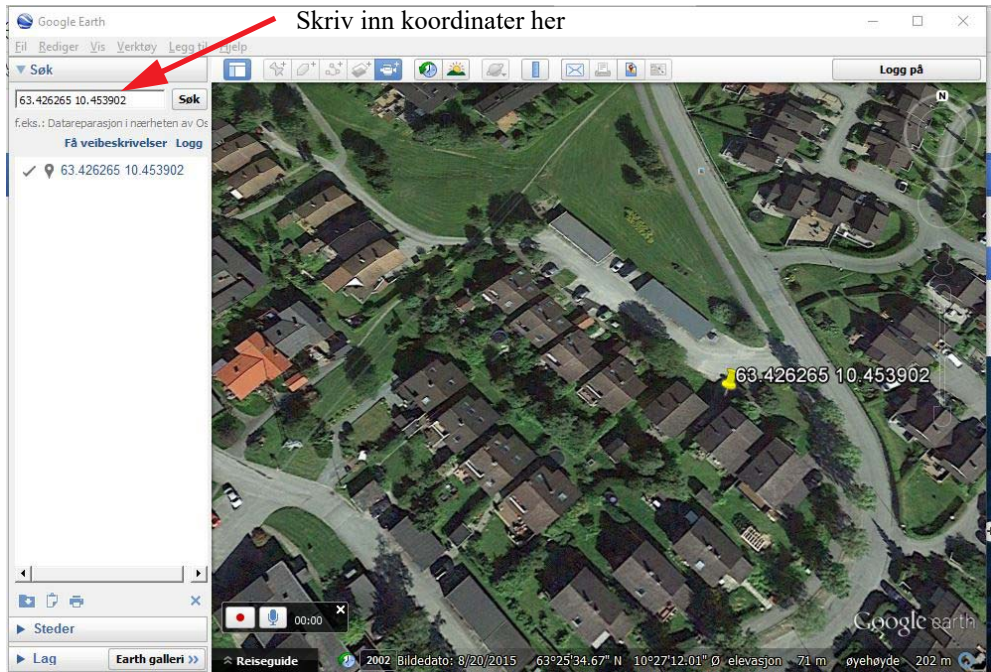
5.5 Visualisering av GPS-data i Google Earth

Data fra GPS-moduler krever gjerne en litt annen behandling enn regneark. Slike data kan f.eks. plottes i Google Earth.

I dette avsnittet skal vi se hvordan vi kan hente inn de lagrede dataene og plote dem i Google Earth.

5.5.1 Installasjon av Google Earth

Google Earth kan lastes ned og installeres fra følgende adresse: <https://www.google.com/earth/>
Skjermbildet under viser brukergrensesnittet for Google Earth.



I øverste venstre hjørne finnes et søkefelt. Her kan man enten skrive en adresse, eller man kan skrive inn koordinater (med desimaler). I dette tilfellet har jeg skrevet inn:

Breddegrad: 63.426265
Lengdegrad: 10.453902

Da vil Google Earth zoomme inn akkurat til disse koordinatene som er rett utenfor veggen der jeg sitter. Dvs. det kan se ut som den bommer med et par meter (ringen angir plasseringen av mottakeren).

Høydeangivelsen for det angitte stedet varierer mellom 60 – 80 meter over havet (Google Earth angir en høyde som varierer mellom 65 – 75 meter).





5.5.2 Plotting av en trase i Google Earth

Dersom man ønsker å plote en trase i Google Earth må man benytte et programmeringsspråk kalt “Keyhole Markup Language” (KML). Dette er et “markup language” utviklet for visualisering av to- og tredimensjonale strukturer knyttet til kartdata. Språket ble utviklet i forbindelse med etableringen av Google Earth som ble lansert i 2004. I 2008 ble KML godkjent som en internasjonal standard for denne type geografisk visualisering.

Siden språket er temmelig omfattende og vi trenger kun å bruke en beskjeden del av det. Vi benytter derfor en ferdige programkode og klipper inn våre data for lengde-, breddegrad og høyde. Disse legges inn som en liste med data i kml-koden (se under), før kml-fila lagres med et ønsket navn.

Koordinater og høyde data legges inn som vist under:

-112.2550785337791,36.07954952145647,2357

Lengdegrader [°], Breddegrader [°], Høyde [m]

Programkode skrevet i kml (legg merke til at et sett med eksempelkoordinater og høyde er klippet inn).

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
  <name>Paths</name>
  <description>Examples of paths. Note that the tessellate tag is by default
    set to 0. If you want to create tessellated lines, they must be authored
    (or edited) directly in KML.</description>
  <Style id="yellowLineGreenPoly">
    <LineStyle>
      <color>7f00ffff</color>
      <width>4</width>
    </LineStyle>
    <PolyStyle>
      <color>7f00ff00</color>
    </PolyStyle>
  </Style>
  <Placemark>
    <name>Absolute Extruded</name>
    <description>Transparent green wall with yellow outlines</description>
    <styleUrl>#yellowLineGreenPoly</styleUrl>
    <LineString>
      <extrude>0</extrude>
      <tessellate>0</tessellate>
      <altitudeMode>absolute</altitudeMode>
      <coordinates>
```

-112.2550785337791,36.07954952145647,2357
-112.2549277039738,36.08117083492122,2357
-112.2552505069063,36.08260761307279,2357
-112.2564540158376,36.08395660588506,2357
-112.2580238976449,36.08511401044813,2357
-112.2595218489022,36.08584355239394,2357
-112.2608216347552,36.08612634548589,2357
-112.262073428656,36.08626019085147,2357
-112.2633204928495,36.08621519860091,2357
-112.2644963846444,36.08627897945274,2357
-112.2656969554589,36.08649599090644,2357

</coordinates>

</LineString>

</Placemark>

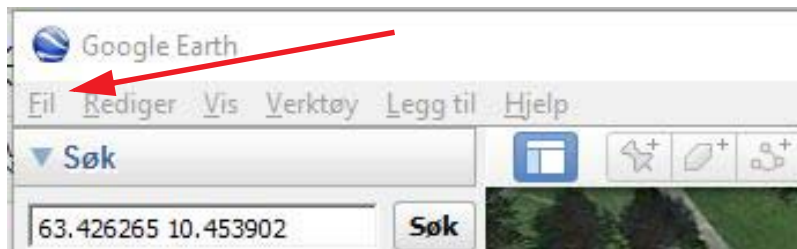
</Document>

</kml>

De koordinatene som pr. i dag ligger i eksempelet angir en helikopterflyvning

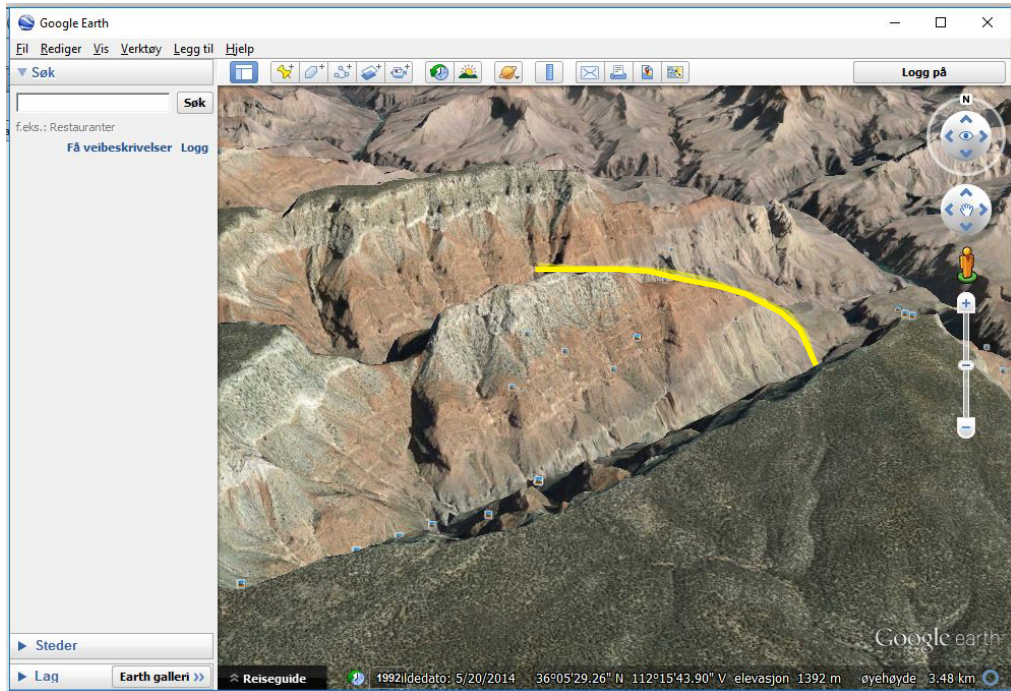
over Grand Canyon i Colorado, USA. Eksempeldataene byttes ut med de aktuelle dataene og kodefila lagres under et ønsket filnavn som ender med .kml.

Filen hentes opp i Google Earth ved å velge *Fil* og *Åpne* for å laste opp den filen hvor koden og dataene ligger. Alternativt kan en klikke på den aktuelle .kml fila.





En vil da få tegnet inn traseen som angitt av lista med koordinater og vist på riktig sted. Eksempelet under viser traseen til helikopteret som flyr over Grand Canyon (gul linje).



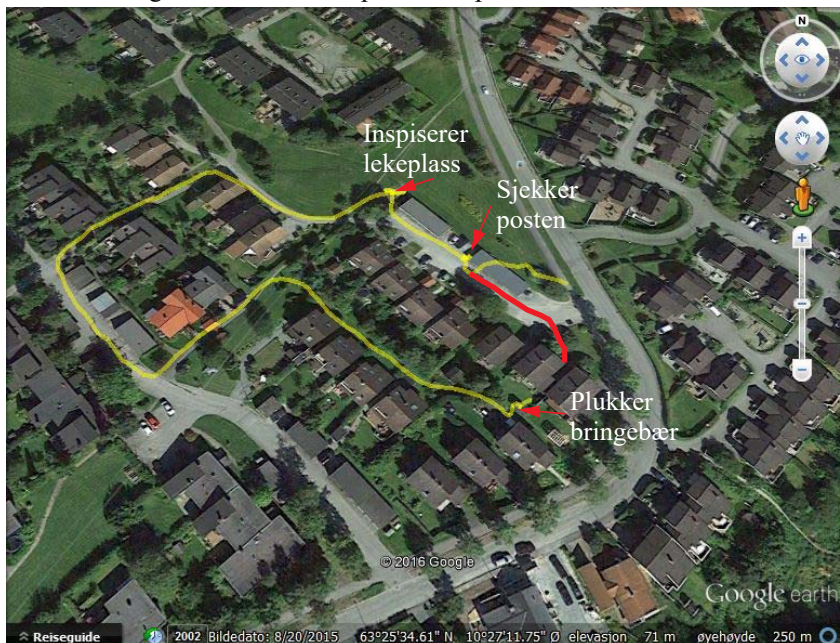
Dersom man ønsker å vise hvor man har gått en tur, så kan det være upraktisk å måtte vise absolutt høyde. Høydemålinger kan ha store avvik slik at en kan oppleve at traseen går mange meter over bakken eller forsvinner under bakken til tross for at man hadde begge beina på jorda. I slike situasjoner kan det være greit å bytte ut kommandoen:

```
<altitudeMode>absolute</altitudeMode>
```

med kommandoen:

```
<altitudeMode>clampToGround</altitudeMode>
```

Dermed vil kurven følge bakken som vist på traseen på bildet under.



Vi legger imidlertid merke til at mottakeren har problemer i starten. I dette området er avviket stort før den tar seg inn. Den røde kurven angir den riktige ruta. Deretter er den svært nøyaktig. Posisjonsdata samples hvert 3. sekund.

Ved bruk av GPS-data ved oppsending av CanSat kan man selvfølgelig ikke neglisjere høydeinformasjonen. Man bør imidlertid være klar over risikoen for relativt store avvik, og bør vurdere å legge inn høydedata fra en kalibrert barometrisk høydemåleren framfor GPS-data.

5.5.3 Editering av kml-fila

Hvilket program skal man så bruke for å legge inn koordinater og høyde. Et nyttig editeringsverktøy til dette formålet er Notepad++. Dette programmet er en litt avansert teksteditor som ikke gjør noen endringer med filformatet med mindre man ønsker det. Notepad++ kan lastes ned fra:

<https://notepad-plus-plus.org/download/>

Hvor pr. dags dato den aktuelle versjon er 7.8.4.



Figuren under viser et utsnitt av brukergrensesnittet til Notepad++.

```
C:\Arduino\CanSat\kml-files\Path.kml - Notepad++
Fil  Rediger  Søk  Vis  Format  Språk  Oppsett  Makro  Utfør  Tillegg  Vindu  ?
[Icons]
Path.kml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <kml xmlns="http://www.opengis.net/kml/2.2">
3  <Document>
4  <name>Paths</name>
5  <description>Examples of paths. Note that the tessellate tag is by default
6  set to 0. If you want to create tessellated lines, they must be authored
7  (or edited) directly in KML.</description>
8  <Style id="yellowLineGreenPoly">
9  <LineStyle>
10 <color>7f00ffff</color>
11 <width>4</width>
12 </LineStyle>
13 <PolyStyle>
14 <color>7f00ff00</color>
15 </PolyStyle>
16 </Style>
17 <Placemark>
18 <name>Absolute Extruded</name>
19 <description>Transparent green wall with yellow outlines</description>
20 <styleUrl>#yellowLineGreenPoly</styleUrl>
21 <LineString>
22 <extrude>0</extrude>
23 <tessellate>0</tessellate>
24 <altitudeMode>absolute</altitudeMode>
25 <coordinates>
26 -112.2550785337791,36.07954952145647,2357
```

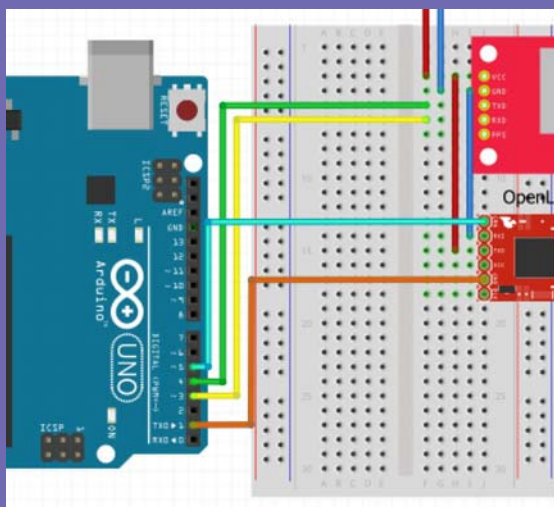
For å forenkle prosessen med å klippe inn data i kml-koden, er det praktisk å skrive koordinat- og høydedata i det formatet som programmet ønsker: Lengdegrad, breddegrad, høyde. Husk å bruke punktum som desimalpunkt og komma mellom hver verdi. **NB! Det skal ikke være mellomrom etter kommaene.** For å se hvordan dette kan gjøres se avsnitt 5.3, side 40.



6 Referanser

- [1] Gunnar Stette, *Romtekologi - Del av faget Teknologi og Forskningslære (CanSat)*, NTNU juli 2011.
- [2] Nils Kr. Rossing, *ROV - med trykk og temperaturmåling*, Rev 3.1 - 20.11.17
Heftet kan lestes ned i sin helhet fra: <https://www.ntnu.no/skolelab/bla-hefteserie>





Heftet ble laget i forbindelse med en nettverkssamling for ToF-lærere i Østlandsregionen mars 2018 og er i første rekke beregnet som et hjelpemiddel under samling og som en ressurs som ev. kan brukes av lærerne når de kommer tilbake til sin egen skole. Senere er heftet tilrettelagt som en lærerveiledning for elevverkstedaktiviteter ved Skolelaboratoriet.

Heftet beskriver hvordan man kan koble opp og måle lufttrykk ved hjelp av sensoren BMP180 og en Arduino UNO.

Videre hvordan man kalibrerer og gjør høydeberregninger på bakgrunn av lufttrykk. For den som ønsker å gå litt videre beskrives også bruk av en billig GPS-mottaker og hvordan dataene fra denne kan visualiseres ved hjelp av Google Earth.

Tilslutt beskrives hvordan en kan tenke seg å måle dybden under vann ved hjelp av trykkmåler montert i et vanntettkammer senket ned i sjøen.

Kammeret inneholder også andre sensorer for måling av lysstyrke, temperatur og saltinnhold.

Nils Kr. Rossing

Dosent ved Skolelaboratoriet

E-post: nils.rossing@ntnu.no

Prosjektleder ved Vitensenteret

E-post: nkr@vitensenteret.com



Trondheim

Institutt for
fysikk

Skolelaboriet
for matematikk, naturfag
og teknologi

Tlf. 73 55 11 43

<https://www.ntnu.no/skolelab>