Ola Kristoffer Hoff & Oline Amundsen

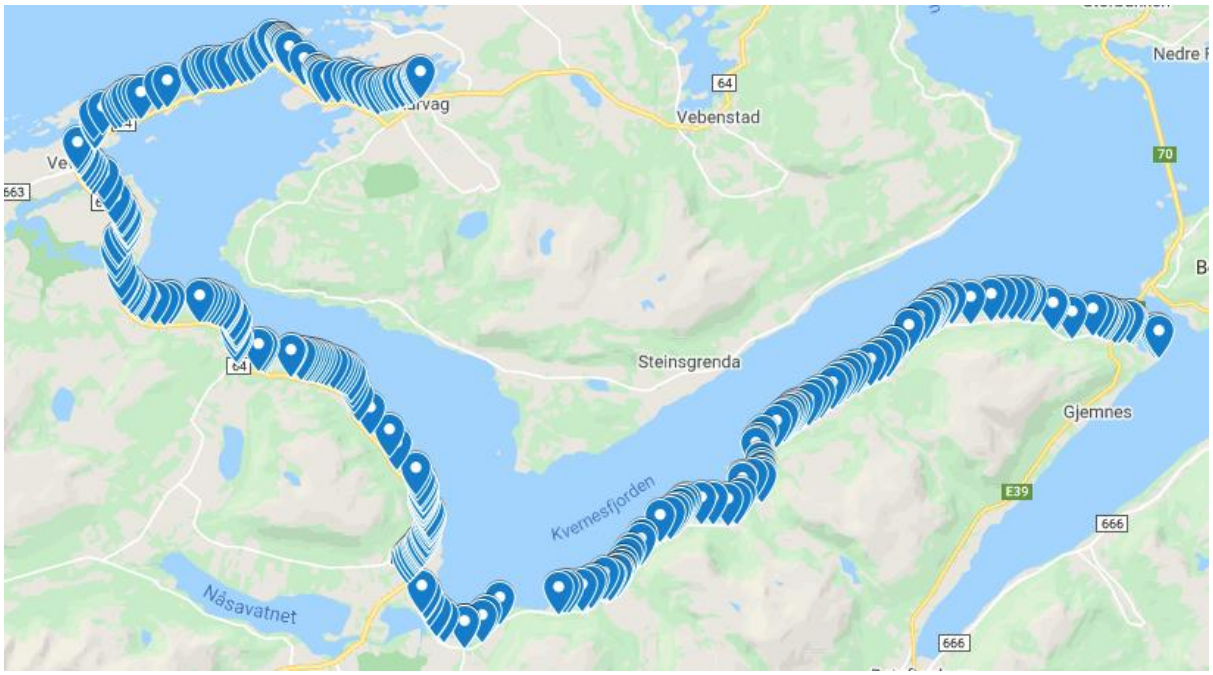
Øving 8 – Kartnavigasjon
Algoritmer og datastrukturer

Jeg har samarbeidet med Oline Amundsen. Vi har gjort alt sammen, samarbeidet har vært faglig diskusjon, ikke fordeling av arbeidsmengde.

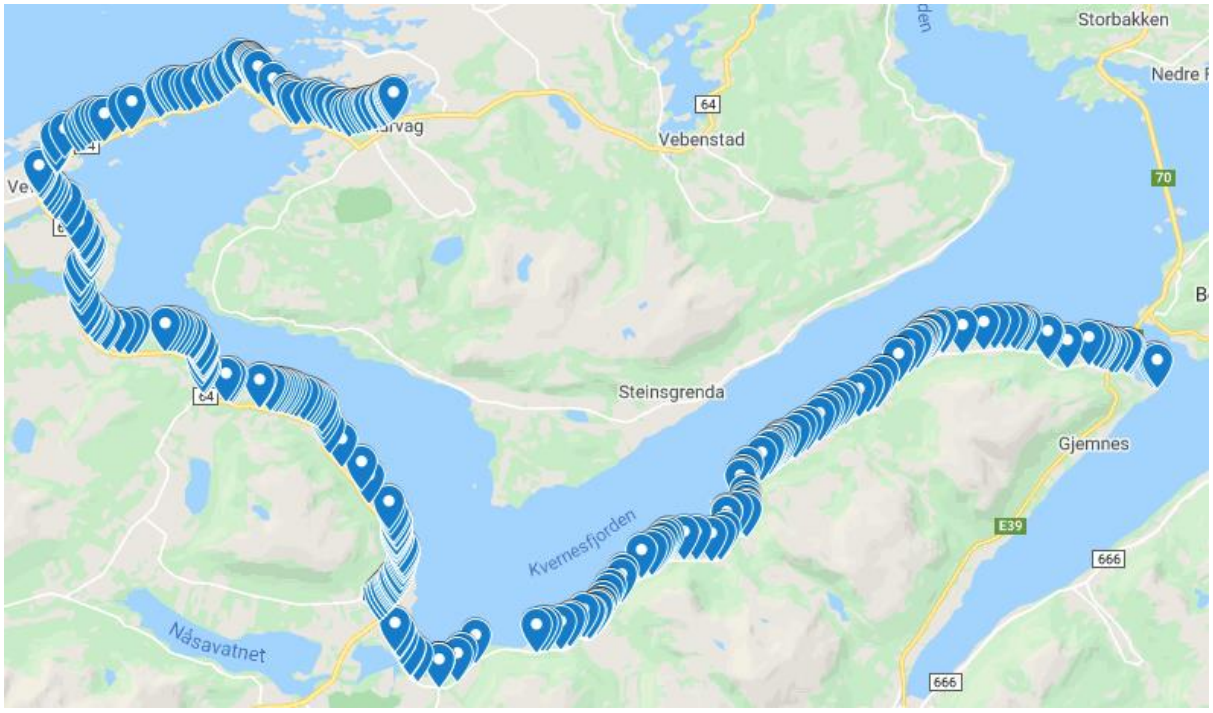
Korteste vei A* og Dijkstra

Vi endte opp med å bruke en GoogleMaps funksjonalitet, hvor vi la inn alle bredde- og lengdegradene til nodene i et Excel-dokument (legges ved som vedlegg) som GoogleMaps leste og lagde det vi ser under. PS: måte dele innlesingen i to, da GoogleMaps hadde en maks kapasitet på 2000 noder om gangen. Alle filer som ble lastet ned ble lagt i «src»-mappen. Tidtakingen på A* og Dijkstra er gjort med a ta System.nanoTime() før og etter metoden kalles, og deretter regne ut tiden som har gått, og gjøre om fra nanosekunder til sekunder.

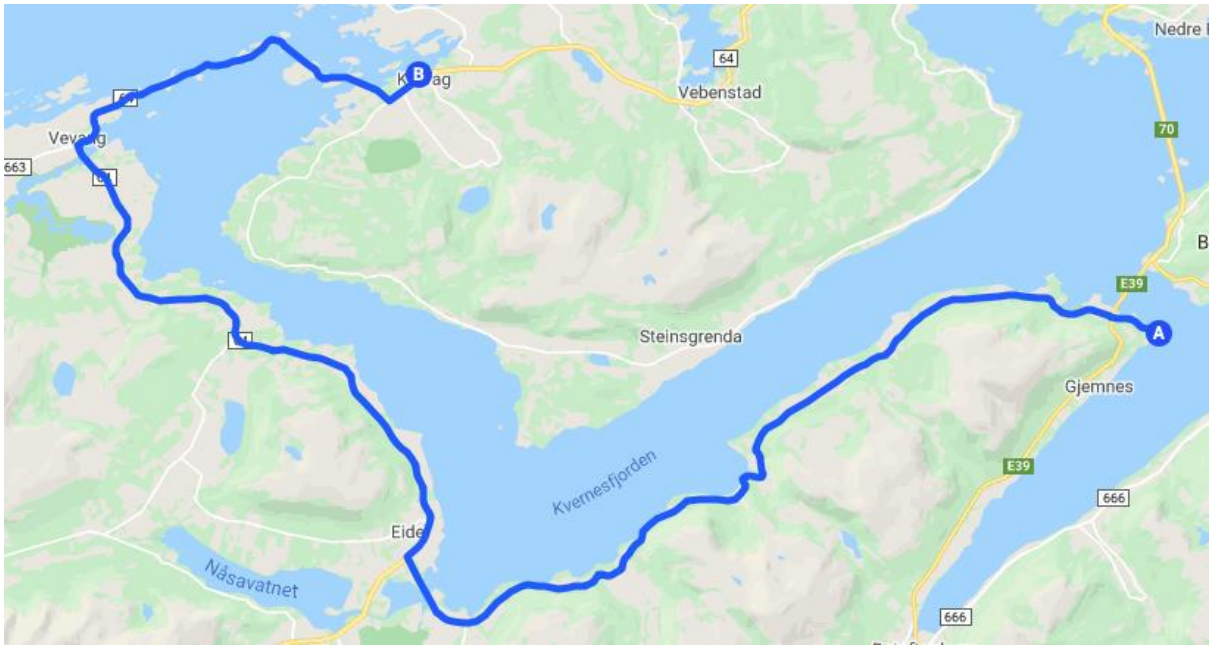
```
long startTime = System.nanoTime();
graf.dijkstra(startNode, sluttNode);
long endTime = System.nanoTime();
long total = endTime - startTime;
System.out.println((double)total / 1000000000 + " sekunder");
```



Dijkstras vei forslag for Kårvåg – Gjemnes.



A* sitt vei forslag for Kårvåg – Gjemnes.

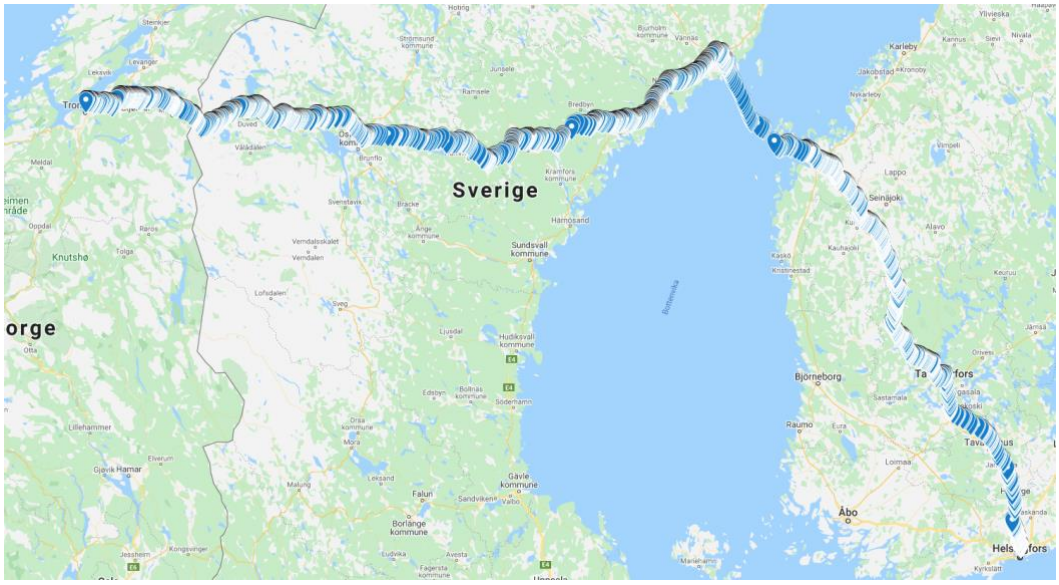


Google sitt vei forslag for Kårvåg – Gjemnes

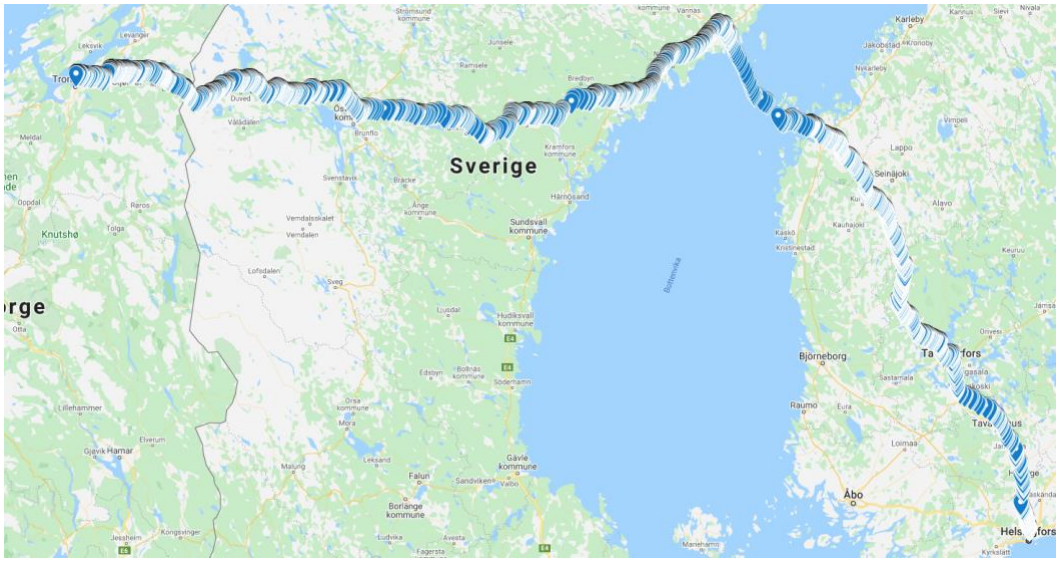
```
0.014266476 sekunder
Ferdig
Antall noder (Path): 315
Antall noder (totalt): 11415
Tid: Timer: 0, Minutter: 40, Sekunder: 10
```

```
0.008253238 sekunder
Ferdig
Antall noder (Path): 315
Antall noder (totalt): 7990
Tid: Timer: 0, Minutter: 40, Sekunder: 10
```

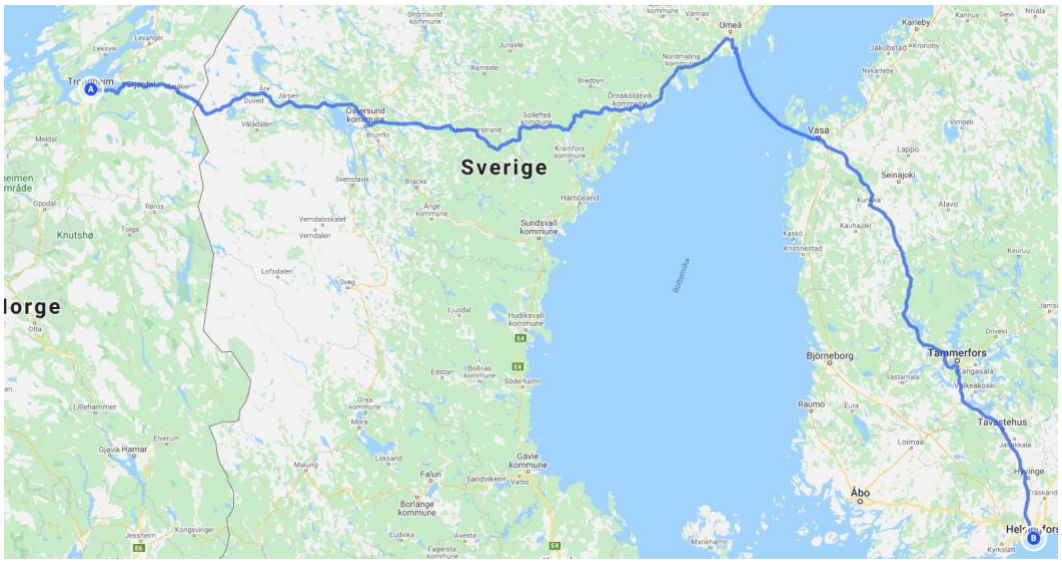
Dijkstras info (øverst) (Kårvåg - Gjemnes)
A* sin info, her ser vi at A* sjekket nesten 3500 færre noder (~30% mindre). Dette er mye sannsynlig da avstanden mellom Kårvåg og Gjemnes ikke er så stor og Dijkstra vil «snuble» borti den ganske tidlig. A* var til tross for den korte avstanden fortsatt være raskere enn Dijkstra.



Dijkstras vei forslag for Trondheim – Helsinki.



A* sitt vei forslag for Trondheim – Helsinki.



Dette er Google sitt forslag til Trondheim - Helsinki.

```
3.552625682 sekunder
Ferdig
Antall noder (Path): 3192
Antall noder (totalt): 5608181
Tid: Timer: 15, Minutter: 28, Sekunder: 8
1.982744462 sekunder
Ferdig
Antall noder (Path): 3192
Antall noder (totalt): 2796247
Tid: Timer: 15, Minutter: 28, Sekunder: 8
```

Dijkstras info (øverst) (Trondheim - Helsinki)

A* sin info, har ser vi at A* bare trengte å se på rundt halvparten av nodene Dijkstra gjorde. Og dermed også fant samme vei på litt over halvannet sekund kortere tid.

Sjekker at alle nodene i svaret til A* og Dijkstra er de samme.

```
for (int i = 0, count = 0; i < dik.length && i < ast.length; i++)
{

    if (dik[i] != ast[i] && count < 100)
    {
        System.out.println("Dik: " + dik[i] + " Ast: " + ast[i] + " index: " + i);
        count++;
    }
}
```

Denne vil print'e ut alle forskjellene mellom ruten A* og Dijkstra fant. Vår printet ikke noe, altså de er identiske.

Vi kjørte 100 tester, hvor vi valgte ut to tilfeldige steder og fant rute, avstand og tok tiden på dette, alle var under 10 sekunder, med et gjennomsnitt på 1.41s for A* og 1.99s for Dijkstra. Her er koden for sjekken:

```
AVG Dijkstra: 1.9905208754499997
AVG AStar: 1.41072375363
```

```
double avgTimeDijkstra = 0, avgTimeAStar = 0;
Random random = new Random();
int runder = 100;
for (int i = 0; i < runder; i++)
{
    startNode = random.nextInt(graf.get_noder().length);
    sluttNode = random.nextInt(graf.get_noder().length);

    graf.prepGrafIgjen();
    long startTime = System.nanoTime();
    graf.dijkstra(startNode, sluttNode);
    long endTime = System.nanoTime();
    long total = endTime - startTime;
    if ((double)total / 1000000000 > 10.0)
    {
        System.out.println("Tok mer enn 10 sekunder,,,");
    }
    avgTimeDijkstra += ((double)total / 1000000000) / runder;

    graf.prepGrafIgjen();
    startTime = System.nanoTime();
    graf.aStar(startNode, sluttNode);
    endTime = System.nanoTime();
    total = endTime - startTime;
    if ((double)total / 1000000000 > 10.0)
    {
        System.out.println("Tok mer enn 10 sekunder,,,");
    }
    avgTimeAStar += ((double)total / 1000000000) / runder;
    System.out.println(i);
}
System.out.println("AVG Dijkstra: " + avgTimeDijkstra + "\nAVG AStar: " + avgTimeAStar);
```

Ola Kristoffer Hoff & Oline Amundsen

Finne bensin- og ladestasjoner

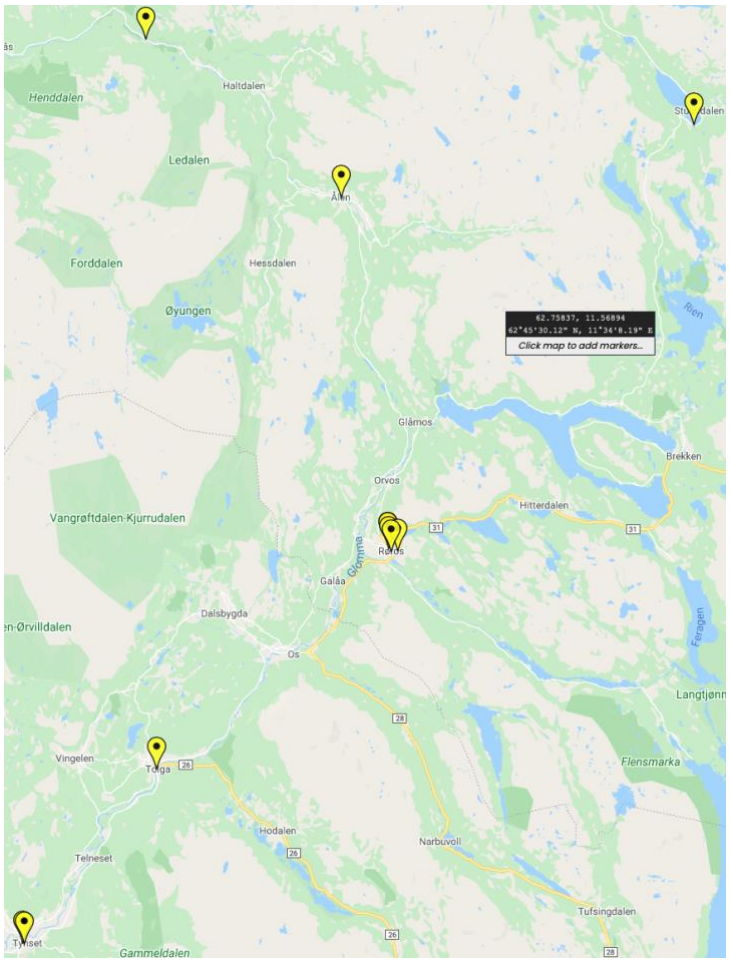
Vi endte opp med å bruke nettsiden maps.co for disse punktene.

Her har vi funnet de 10 nærmeste bensinstasjonene til Trondheim lufthavn, Værnes.

Rådata fra innlegging: (breddegrader, lengdegrader, node nummer, og bare litt farge. Første er startnoden.)

```
63.4531052,10.9183067,6198111,#F00000
63.4522069,10.9159698,Node: 1036307,#FFFF00
63.4475976,10.912719,Node: 4860352,#FFFF00
63.4449821,10.9110906,Node: 5918490,#FFFF00
63.4545841,10.9376432,Node: 6196055,#FFFF00
63.4117161,10.7992113,Node: 851308,#FFFF00
63.4681201,10.9548384,Node: 5013780,#FFFF00
63.4672693,10.9211686,Node: 2650394,#FFFF00
63.4201165,10.7557221,Node: 6318399,#FFFF00
63.4669757,10.9179984,Node: 1149815,#FFFF00
63.4686157,10.9144243,Node: 1330720,#FFFF00
```

Her har vi funnet de 10 ladestasjonene som er nærmest Røros hotell.



Nærmere bilde av Røros.

Nærmere bilde av Tynset.

Rådata fra innlegging: (breddegrader, lengdegrader, node nummer, og bare litt farge. Første er startnoden)
PS: Det er en ladestasjon på Røros hotell, som da gir oss den mer oransje markeringen, en overlapping av gul og rød.

```
62.5795954,11.3787878,1117256,#F00000
62.5795954,11.3787878,Node: 1117256,#FFFF00
62.5740357,11.3850772,Node: 6309479,#FFFF00
62.5760837,11.3802187,Node: 5555950,#FFFF00
62.5742267,11.3957116,Node: 6300826,#FFFF00
62.8417913,11.301628,Node: 2256064,#FFFF00
62.4084784,10.9979451,Node: 6321178,#FFFF00
62.275225,10.7760685,Node: 1588203,#FFFF00
62.2746228,10.7792255,Node: 1588182,#FFFF00
62.9606727,10.9792146,Node: 5256418,#FFFF00
62.8966394,11.8850357,Node: 4717026,#FFFF00
```