0.0.1 Komponent 5: Anvendt matematik

Denne komponent beskriver ikke opbygningen af systemet, men derimod de matematiske formler, der er blevet anvendt for at nå frem til resultaterne. Hver formel vil beskrives og der vil forklares, hvorefter det forklares i hvilken sammenhæng de bruges.

Haversine

Denne formel bruges som en hjælpefunktion i forbindelse med udregninger på ruter. Formlen bruger to punkters geografiske position, og udregner fugleflugts afstanden mellem dem i meter, selv hvis punkterne er langt nok fra hinanden til, at jordens krumning spiller en rolle. Jordens krumning spiller ikke en rolle i dette system, da punkterne vil være relativt tæt på hinanden. Følgende variabler og konstanter tages i brug i denne formel:

d:	Fugleflugts afstand mellem to punkter i meter.
R:	Jordens gennemsnits radius. Konstant sat til 6371 kilometer.
θ_1, θ_2:	Længdegrad for punkt 1 og punkt 2
λ_1, λ_2:	Breddegrad for punkt 1 og punkt 2
a, c	Subresultater

Igennem formel (1) til (3) kan det ses, hvordan Haversine udregningerne foretages.

$$a = sin^2(\frac{\Delta\theta}{2}) + cos(\theta_1) * cos(\theta_2) * sin^2(\frac{\Delta\lambda}{2}) \tag{1}$$

$$c = 2 * atan2(\sqrt{a}, \sqrt{1-a}) \tag{2}$$

$$d = R * c * 1000 \tag{3}$$

atan2 returner et grad værdi mellem -180°til 180°. Det er dog ikke nødvendigt at tage højde for dette i denne sammenhæng, da det ikke er nødvendigt at se på graderne som en helcirkel fra 0°til 360°.
Hjemmesiden, simulatoren og den distribuerede database tager alle brug af Haversine formlen.

- I databasen bruges den i udregningen for en bus tid til ankomst ved et stoppested. Et eksempl på dette kunne være, når afstanden fra bussen til det valgte stoppested skal

udregnes. Her vil afstanden mellem hvert rutepunkt udregnes ved hjælp af Haversine formlen, og ligges sammen. Resultatet vil være den søgte afstand.

- I simulatoren bruges den når bussens næste position skal udregnes. Der er fundet, hvor langt bussens skal køre, og der undersøges så mellem hvilke to rutepunkter, bussens skal være. Dette gøres ved at udregne afstanden mellem rutepunkter ved hjælp af Haversine formlen, finde ud af ved hvilket punkt afstanden er større end den bussens skal køre, hvorefter bussens nye position før dette punkt findes.
- På hjemmesiden bruges den når det skal udregnes, hvilke to rutepunkter et stoppested skal ligge mellem. I denne situation skal det undersøges om stoppestedet ligger en hvis afstand væk fra et punkt. Her til bruges Haversine til at udregne distancen.

Formlen og implementeringen i koden er ikke lavet selv, men derimod hentet fra http://www.movable-type.co.uk/scripts/latlong.html. Udregninger og implementeringer fra denne side er Open-Source og lavet af Chris Veness. Den eneste modifikation der er blevet implementeret er, at det endelige resultat er konverteret fra kilometer til meter

Kurs

Hvis et objekt bevæger sig mod et punkt, bruges denne formel til at udregne, hvilken retning objektet bevæger sig, på en 360°-skala. I denne udregning er nord sat til 0°/360°. Der blevet gjort brug af følgende variable:

b:	Kursen objektet følger på en 360°-skala.
θ_o, θ_p:	Længdegrad for objektet og punktet
λ_o, λ_p:	Breddegrad for objekter og punktet
x, y	Subresultater

Igennem formel (4) til (6) kan det ses, hvordan kursen udregnes.

$$x = cos(\theta_o) * sin(\theta_p) - sin(\theta_o) * cos(\theta_p) * cos(\Delta\lambda) \tag{4}$$

$$y = sin(\Delta\lambda) * cos(\theta_p) \tag{5}$$

$$b = atan2(y, x) \tag{6}$$

Da atan2 returner en værdi mellem -180°til 180°, er det nødvendigt at konvertere denne

værdi til en 360°. Dette gøres ved hjælp af formel 7, hvor "%"er modulo.

$$b_{360} = ((b + 360) \tag{7}$$

Kun simulatoren gør brug af kursen. Den tages i brug, når der skal udregnes, hvor bussens nye position skal være. Det er tidligere udregnet, mellem hvilke to punkter bussen skal ligge, og der skal derfor udregnes et ny position mellem disse to punkter. Til denne udregning bruges kursen.

Formlen og implementeringen i koden er ikke lavet selv, men derimod hentet fra http://www.movable-type.co.uk/scripts/latlong.html. Udregninger og implementeringer fra denne side er Open-Source og lavet af Chris Veness.

Ny position mellem to punkter

Hvis et objekt skal placeres mellem to punkter, en hvis distance udfra det første punkt, med kurs mod det andet punkt, tages denne formel i brug. Kursen mellem de to punkter er givet på en 360°skala, hvor nord er sat til 0°/360°. Igennem udregnningen er det blevet gjort brug af følgende variabler og konstanter:

R:	Jordens gennemsnits radius. Konstant sat til 6371 kilometer.
b:	Kursen objektet følger fra initial punktet på en 360°-skala.
d:	Distancen objektet skal bevæge sig ud fra initial punktet.
θ_o, θ_p:	Længdegrad for objektet og punktet
λ_o, λ_p:	Breddegrad for objektet og punktet
x, y	Subresultater

Igennem formel (8) til (11) kan det ses, hvordan punktet udregnes.

$$\theta_o = asin(sin(\theta_p) * cos(\frac{d}{R * 1000}) + cos(\theta_p) * sin(\frac{d}{R * 1000}) * cos(b) \quad (8)$$

$$x = sin(b) * sin(\frac{d}{R * 1000}) * cos(\theta_p) \quad (9)$$

$$y = cos(\frac{d}{R * 1000}) - sin(\theta_p) * sin(\theta_o) \quad (10)$$

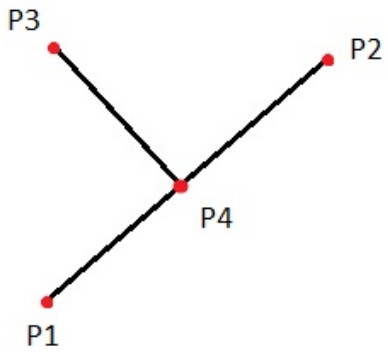
$$\lambda_o = \lambda_p + atan2(x, y) \quad (11)$$

Kun simulatoren tager brug af denne funktion. Den skal bruges i sammenhæng med at udregne, hvor en bus skal placeres på ruten ved en ny opdatering. Distancen bussen skal bevæge sig er tidligere blevet udregnet, samt det punkt på ruten, bussen skal være før. Kursen findes ved hjælp af bussens nuværende position samt det udregnede rutepunkt. Ved hjælp af startpunktet, distancen og kursen, udregnes bussens nye position. Ved udregninger af breddegraden af objektet bruges atan2, og da der skal returneres et antalgrader, skal denne værdi konvertes til en 360°version. Dette gøres ved hjælp af formel 7

Formlen og implementeringen i koden er ikke lavet selv, men derimod hentet fra http://www.movable-type.co.uk/scripts/latlong.html. Udregninger og implementeringer fra denne side er Open-Source og lavet af Chris Veness.

Tætteste punkt på en linje

Et linjestykke er spændt op mellem to punkter. Et tredje punkt kan ligge et vilkårligt stykke ud fra denne linje. Det tredje punkts tætteste punkt på linjen, vil være det punkt, hvis linjestykke skabt med det tredje punkt, er ortogonal med linjestykket mellem det første og andet punkt. Situationen kan ses på figur 1, hvor P1 og P2 er de punkter der spænder det originale linjestykket, P3 er det vilkårlige punkt, og P4 er det vilkårlige punkts tætteste punkt på linjen mellem P1 og P2. Udregningen er kun relevant, hvis linjestykket mellem P1 og P2 ikke er horizontal eller vertikal. Disse situationer forklares senere.
Igennem udregningerne gøres der brug af disse variabler:



Figur 1: Situationen 1: Tætteste punkt på en linje

A:	Linjestykket mellem P1 og P2 hældningskoefficient.
B:	Linjestykket mellem P1 og P2 skæring med y-aksen.
$\theta_1, \theta_2, \theta_4, \theta_4$:	Længdegrad for de fire punkter.
$\lambda_1, \lambda_2, \lambda_3, \lambda_4$:	Breddegrad for de fire punkter.

Denne formel virker kun for relative tætte punkter, hvor der skal tages hensyn til jordens hældning, og jorden derfor kan ses som et plan. Igennem formel (12) til (15), kan det ses, hvordan punktet findes.

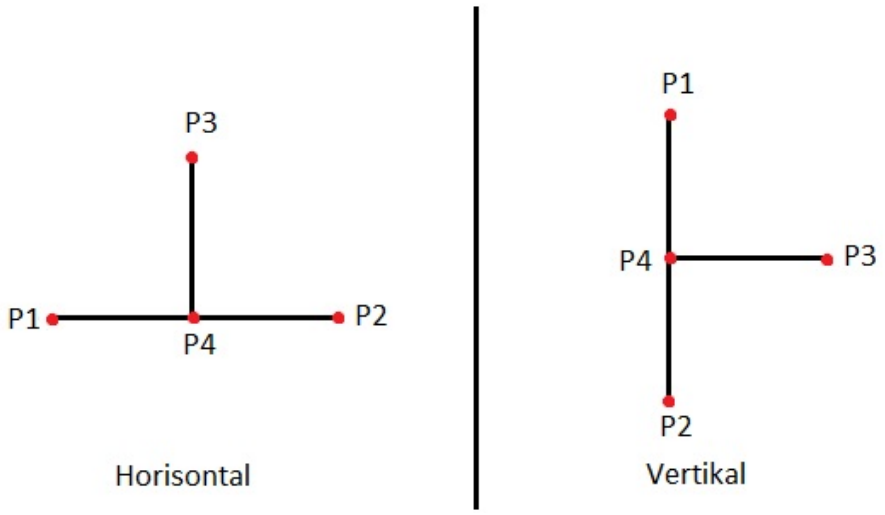
$$A = \frac{\theta_2 - \theta_1}{\lambda_2 - \lambda_1} \tag{12}$$

$$B = \theta_1 + A * (-\lambda_1) \tag{13}$$

$$\theta_4 = \frac{A * \theta_3 + \lambda_3 - A * B}{A^2 + 1} \tag{14}$$

$$\lambda_4 = \frac{A^2 * \theta_3 + A * \lambda_3 + B}{A^2 + 1} \tag{15}$$

I tilfælde af at linjen mellem punkt 1 og punkt 2 er horisontal eller vertikal, er der under implementering skabt special tilfælde. Hvis linjen er horisontal sættes $\theta_4 = \theta_1$ og $\lambda_4 = \lambda 3$. Hvis linjen er vertikal sættes $\theta_4 = \theta_3$ og $\lambda_4 = \lambda_1$. Situation ses på figur 2



Figur 2: Situationen 2 og 3: Horisontalt eller vertikalt linjestykke

Denne formel tages i brug i databasen og på hjemmesiden.

- I database tages denne udregning i brug når en bus tid til ankomst ved et stoppested skal udregnes. I denne process skal der på et tidspunkt udregnes, hvilket rutepunkt, bussen er tættest på. Dette gøres ved hjælp af en kombination af denne formel, samt Haversine formlen. Ved hvert linjestykke skabt af punkterne på ruten, udregnes det tætteste punkt for bussen på dette linje stykke. Mellem bussen og dette punkt gøres der brug af Haversine, for at udregne afstanden fra bussen til linjestykket. Endepunktet for det linjestykke, hvor bus til linje afstanden er kortest, må være det rutepunkt bussen er tættest på.

- På hjemmesiden skal der udregnes, mellem hvilke to punkter et stoppested skal ligge. Udregningen fortages på sammen måde, som i databasen, hvor bussen blot er erstattet med et stoppested. Resultatet er det rutepunkt, stoppested skal ligge før.

Den linære funktion (Ax + B) der skabes til linjestykket vil ikke kun strække sig mellem de to længde- og breddegrader der gives. Der kan derfor opstå en situation, hvor objektet egentlig ligger tættest på ét linjestykke, men den linje der bliver skabt af et andet linjestykke vil have en mindre ortogonal distance hen til objektet. Derfor er det vigtigt, at der ved implementering tages højde for, at det kun er det kun er linjestykket der undersøges og ikke hele linjen. Dette sikres ved, at en eller flere af de følgende fire regler ikke må være gældende for $\theta_4 og \lambda_4$:

- $\theta_4 > \theta_1 \quad \& \quad \theta_4 > \theta_2$
- $\theta_4 < \theta_1 \quad \& \quad \theta_4 < \theta_2$
- $\lambda_4 > \lambda_1 \quad \& \quad \lambda_4 > \lambda_2$
- $\lambda_4 < \lambda_1 \quad \& \quad \lambda_4 < \lambda_2$

Hvis blot en af disse regler passer, er punktet ugyldigt, da det ikke ligger på linjestykket.

Denne formel er lavet på baggrund af information fundet på to hjemmesider.
http://demo.activemath.org/ActiveMath2/search/show.cmd?id=mbase://AC_UK_calculus/functions/ex_linear_equation_two_points beskriver hvordan en lineær funktion findes ved to punkter.
http://math.ucsd.edu/~wgarner/math4c/derivations/distance/distptline.htm beskriver hvordan det ortogonale punkt findes ved hjælp af en lineær funktion.

Grader og radianer konvertering

Nå de forrige fire formler skal implementeres er det ofte nødvendigt at konvertere mellem radianer og grader. På formel (16) kan konverteringen fra grader til radianer ses, og på formel (17) kan konverteringen fra radianer til grader ses.

$$Radianer = \frac{Grader * \pi}{180} \tag{16}$$

$$Grader = \frac{Radianer * 180}{\pi} \tag{17}$$