Visualisering og Simulering Mappeoppgave Rapport

Kandidatnummer: 869

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Innhold

[Git 3](#_Toc115919421)

[Introduksjon 3](#_Toc115919422)

[Metode 3](#_Toc115919423)

[Geodata til SurfaceMesh 3](#_Toc115919424)

[Barysentriske koordinater 4](#_Toc115919425)

[Regn 5](#_Toc115919426)

[Kvadratisk BSpline 5](#_Toc115919427)

[Resultater 6](#_Toc115919428)

[Meny/Buttons 6](#_Toc115919429)

[SurfaceMesh 6](#_Toc115919430)

[BSplines 6](#_Toc115919431)

[RollingBall 7](#_Toc115919432)

[Diskusjon 7](#_Toc115919433)

# Git

https://github.com/adamaske/VSIMFolder

# Introduksjon

For å gjøre denne oppgaven har jeg brukt Oblig 3 som startpunkt. Ettersom oppgaveteksten sier det er lov å samarbeide, har jeg og Sivert K samarbeidet, slik som vi også har gjort på andre obliger/innleveringer. Målet for denne oppgaven er å lage et mesh utifra en punktsky som er hentet fra hoydedata.no, programmere regn til og falle på meshet og vise kvadratiske B-splines etter stien regnet har gått.

Jeg bruker QT for å visualisere dette og koder i C++.

# Metode

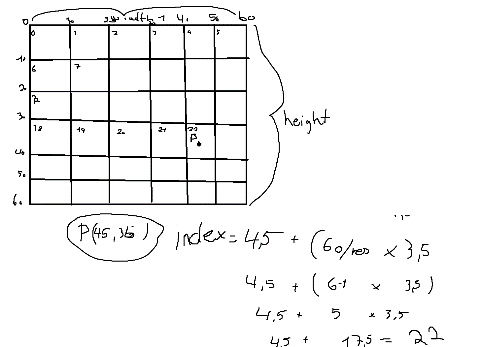
## Geodata til SurfaceMesh

Tekstfilen som jeg har brukt finnes på «../VSIM\_Folder/HeightData/fullData.txt». Filen inneholder 3.1 millioner punkter, x, y og z koordinater. SurfaceMesh::SurfaceMesh leser tekstfilen og lager et mesh fra dataen.

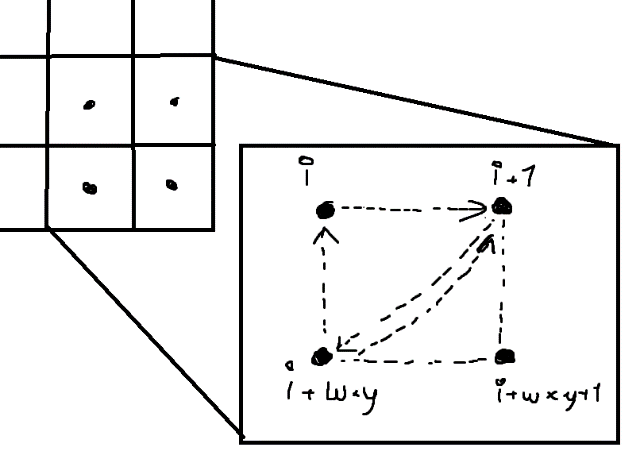
Når filen er lest har jeg en std::vector<float> kalt points, denne inneholer alle koordinatene til punktskyen på rekke. For å lage

For å lage et convex hull har jeg en struct, Quad, i surfacemesh.h. Quad består av fire koordinater. For hver Quad legger jeg til en Vertex i mVertices og blir initialisert med sentrum av Quad’en som posisjon.

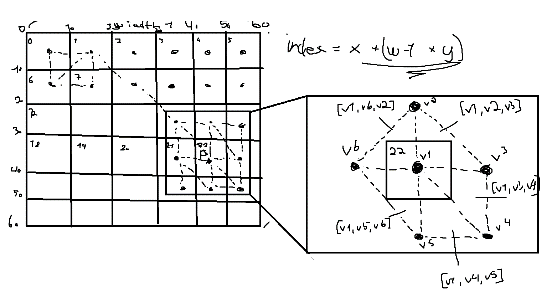
For å finne hvilket punkt som tilhører hvilken Quad bruker jeg rekkefølgen i mQuads og koordinatene til punktet. X koordinaten finner bredde-indeksen til Quaden. For å finne hvem høyde-indeks så ganger jeg bredden av convex hullet med Y koordinaten til punktet, så adderer med bredde-indeksen. Vist i tegningen under.



For å bruke glDrawElements indekserer jeg vertexene. Tegnignen under viser hvordan den lager trianglene. To triangler blir laget per Quad. I en dobbel løkke som går fra 0 til høyden og bredden definert for convex-hullet. I den siste løkken er det en løkke til, som går 2 ganger. Første gang indekserer den første trianglet, ander gang triangel nummer 2.



## Barysentriske koordinater



Denne tegningen viser hvem triangler jeg sjekker i SurfaceMesh::GetHeight funksjonen. Jeg må sjekke disse 5 trianglene fordi punktet kan være på hvem som helst av de. Dette er på grunn av jeg finner hvem vertex som er nærmest punktet på samme måte som jeg finner hvilken quad hvert punkt tilhører. Trianglet som returnerer u, v og w som positive er trianglet som blir sendt tilbake til RollingBall::DoPhysics.

## Regn

Jeg bruker klassen RollingBall som regndråper. RenderWindow har en std::vector mRain som lagrer pekere til ballene.

RollingBall::DoPhysics henter et Result fra SurfaceMesh::GetHeight. Result ineholder vertexene til triangelet som ballen er på og høyden som ballen skal være på for å være inntil flaten. Hovedfunksjonen i DoPhysics er nyPos = oldPos + oldVel/60 + 1/2acc/60. Denne funksjonen avgjør hvor ballen flytter seg. Hvis Result ikke returner med vertexer så blir bare gravity variablet akselerasjonen.

## Kvadratisk BSpline

Når regnet er aktivt blir RollingBall::AddLife kalt på hver RollingBall i mRain hver frame. Dette er for å holde telling på hvor lenge regndråpen har levd slik at den legger til en kontroll punkt hver RollingBall::mSplinePointCounter. I RenderWindow::DoRain så sjekker jeg om regnet har levd 150 frames, hvis de har det så aktiverer jeg B-splinen.

RollingBall::EvaluateBezier og FindKnotInterval er hentet fra kompendiumet. I RollingBall::CreateSpline setter jeg opp variablene som trengs for å gjøre evalueringen.

Skjøtevektoren blir lagd dynamisk og ender seg basert på hvor mange kontrollpunkter. Variablet s = d+1 er antall like skjøter som skal i starten og slutten av skjøtevektoren. Algoritmen går gjennom n+d+1-s=n, på første indeks i løkken legger den til en s-mengde av i, som blir 0,0 og 0. Når den finner at den er s nærme slutten så fyller den resten med (i, i, i).

EvaluateBezier blir kalt fra x = 0 til x = 1, med steg på 0.01. Vectorene den returnere blir lagret i koordinatene til vertexer i en <Vertex>-vector. Disse vertexene danner da mSpline som er en VisualPoint, initialisert med vertex-vektoren. Når splinen er lagd så kaller ballen VisualPoint::Draw fra sin egen draw() funksjon.

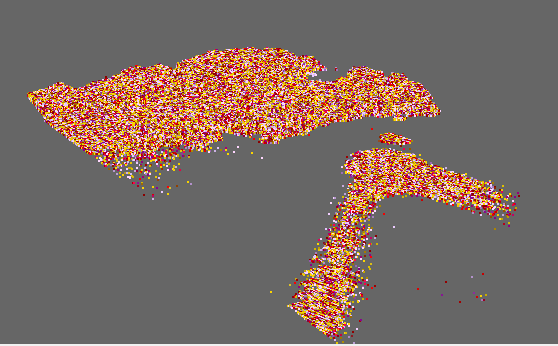
# Resultater

## Meny/Buttons

Jeg har 2 knapper. De er definert i ui\_mainwindow.h, rainButton og drawModeButton. De bruker begge on\_toggled funksjon i mainwindow.cpp. Regn knappen starter regnet og draw-knappen endrer hvordan SurfaceMesh er tegnet på skjermen.

## SurfaceMesh

Ved å gjøre hver linje fra tekstfilen om til en vertex og bruke GL\_POINTS til å tegne de blir dette resultat. Dette resultatet stemmer med geodataen og stedet jeg valgte, Ono i Larvik.



Dette bildet er et steg videre, da vertexer er laget i et kvadratisk grid og høyden til punktene er lagt til. Det er noe vanskelig å se, men dette stemmer dårlig overrens med punktene. Dette kan skyldes at perspektivet er galt så det ser uriktig ut. Funksjonen min for å finne hvem punkt som skal til hver quad har jeg sjekket flere ganger gir riktig Quad indeks. Det kan også skyldes hvordan høyden beregnes, det kan være at det er så mange punkter at SurfaceMesh::Quad sin «height / amount» funksjon går mot 1 jo større «amount» blir.

Et bilde som inneholder fargerik, flagg

Automatisk generert beskrivelse

## BSplines

De røde, blå og gule strekene/pixlene er punktene som RollingBall::CreateSpline fra RollingBall::EvaluateBezier. På det øverste bildet har regn dråpene falt nedover før splinen blir lagd. På det nederste har de gått mot kamera langs meshet i en stund, det er også kurve på noen av de blå strekene hvor regnet har fulgt meshet. Så splinene korresponderer med hvor og hvordan ballen har beveget seg, men noe er galt med funksjonen.

Jeg trur feilen ligger i enten hvordan jeg lager skjøtevektoren eller C vektoren. Jeg har ikke helt skjønt hva som skal i C vektoren, så jeg puttet kontrollpunktene inn i den.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Et bilde som inneholder fargerik

Automatisk generert beskrivelse

## RollingBall

Regnet ruller ikke over flaten som det er meninga de skal. Enten så bare står de stille da de treffer meshet eller så beveger alle seg i negativ z retning, selv mens de beveger seg langs flaten. Jeg bruker akkurat samme funksjon som i Oblig3. Så jeg er svært sikker det skyldes SurfaceMesh::GetHeight funksjonen, den finner feil vertexer

# Diskusjon

Jeg syntes oppgaven er grei, selv om jeg er noe misfornøyd med resultatet mitt. Jeg føler jeg har god forståelse for alle temaene utenom B-Splines. Jeg har god forståelse for både meshet, indekseringen og fysikken som blir utregnet. Men jeg klarer ikke å diagnostisere hva som er galt med meshet, hvorfor høydedataen blir feil.

Jeg prioriterte høydekruvene og parametrisk flate sist, så de har jeg ikke fått gjort. Om jeg skulle gjort oppgaven på nytt igjen ville jeg droppet å stirre så lenge på SurfaceMesh::SurfaceMesh for å få punkter og indeksering til å fungere. Men på grunn av at de barysentriske koordinatene er også avhengig av riktig indeksering og vertexer så så jeg meg blind på problemet.