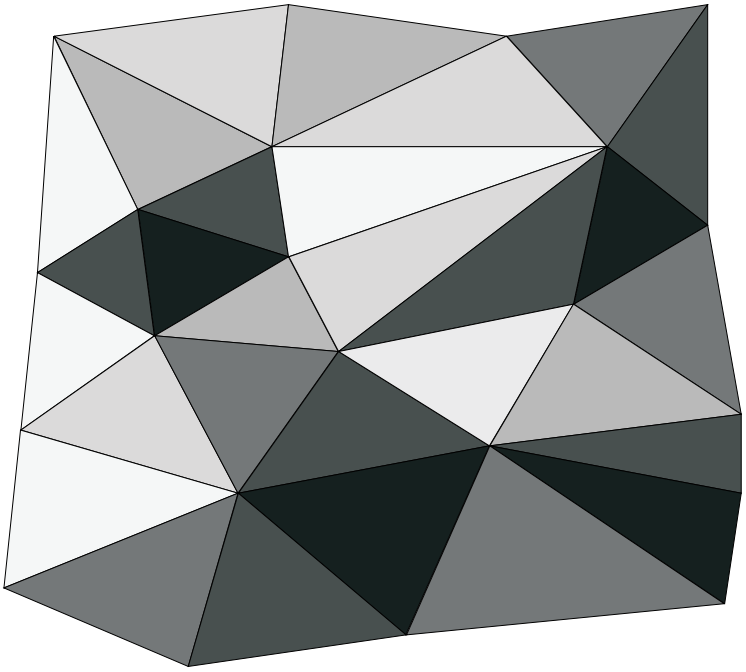
DIGITALE TERRENGMODELLER



JAN TERJE BJØRKE

2005

Digitale terrengmodeller

Jan Terje Bjørke

3. november 2005

Forord

Denne læreboken beskriver hvordan topografien til terrengoverflaten kan modelleres (representeres i en datamaskin) og hvordan slike modeller kan visualiseres. Boken er en revisjon av den første utgaven av 1989. Ved revisjonen i 1998 ble kapitlene om triangulering og geostatistikk betydelig utvidet.

Læreboken er i første rekke skrevet med tanke på undervisningen i geografiske informasjonssystemer for geomatikkstudenter ved Universitet for miljø- og biovitenskap (UMB) og Norges Teknisk-naturvitenskeplige universitet (NTNU).

UMB 3. november 2005

Jan Terje Bjørke[1]

[1]Jan Terje Bjørke er professor i geografisk informasjonsvitenskap ved UMB, 1432 Ås og forsker ved Forsvarets forskningsinstitutt.
epost: jtb@ffi.no

Innhold

0

Del I

Matematisk grunnlag

Kapittel 1

Elementer fra analytisk geometri

1.1 Noen regler for regning med matriser

Vi skal kort angi noen regler for regning med matriser. La oss anta to matriser $\mathbf{A}$ og $\mathbf{B}$. Da gjelder:

$$\mathbf{AB} \neq \mathbf{BA},$$
$$(\mathbf{AB})^t = \mathbf{B}^t\mathbf{A}^t,$$

og

$$(\mathbf{A}^t)^{-1} = (\mathbf{A}^{-1})^t.$$

Dersom $\mathbf{P}$ er en symmetrisk matrise, er $\mathbf{P}^t = \mathbf{P}$. Dersom $\mathbf{a}$ og $\mathbf{b}$ er $1 \times n$ matriser, er $\mathbf{ab}^t = \mathbf{b}^t\mathbf{a}$.

Gitt $\mathbf{Z} = \mathbf{y}^t\mathbf{Ax}$, der $\mathbf{y}$ og $\mathbf{x}$ er søylevektorer. Det totale differensial finnes ved

$$d\mathbf{Z} = d\mathbf{y}^t \cdot \mathbf{Ax} + \mathbf{y}^t\mathbf{A} \cdot d\mathbf{x}.$$

1.2 Arealberegning

1.2.1 Herons formel

Arealet av et triangel kan beregnes på grunnlag av dets tre sidelengder ved

$$A = \frac{1}{4}\sqrt{S(S-2S_1)(S-2S_2)(S-2S_3)}, \tag{1.1}$$

hvor $S = S_1 + S_2 + S_3$.

1.2.2 Arealet av en polygon

Gitt en lukket polygon med i alt n hjørner. Dersom koordinatene til hjørnene er kjent, kan polygonens areal finnes ved å summere over n antall trekanter:

$$A = \frac{1}{2} \cdot [y_1(x_2 - x_n) + y_2(x_3 - x_1) + y_3(x_4 - x_2) + \cdots + y_n(x_1 - x_{n-1})]\,. \tag{1.2}$$

Formelen kan uttrykkes mere kompakt ved

$$A = \frac{1}{2}\sum_{i=1}^{n} y_i(x_{i+1} - x_{i-1}), \text{ hvor } (x_0, y_0) = (x_n, y_n) \text{ og } (x_{n+1}, y_{n+1}) = (x_1, y_1).$$

Arealet av et triangel blir ifølge disse formlene

$$A = \frac{1}{2} \cdot [y_1(x_2 - x_3) + y_2(x_3 - x_1) + y_3(x_1 - x_2)] .$$

Fortegnet til A er avhengig av den retning polygonens hjørner er definert i. Dette kan derfor benyttes til finne polygonens definisjonsretning.

Siden arealet regnes som en sum av produkter, kan vi lett miste signifikant presisjon om koordinatene får store nok verdier. Det vil derfor lønne seg å foreta en translasjon til et lokalt origo, som plasseres innenfor polygonen, og basere arealberegningen på de reduserte koordinater.

1.3 Transformasjoner

Generelt kan transformasjoner mellom koordinatsystemer enten betraktes som om det er datapunktene som flyttes eller som om det er koordinatsystemet som flyttes. Overgang fra den ene til den andre betraktningsmåten fås ved å skifte fortegn til transformasjonsparametrene.

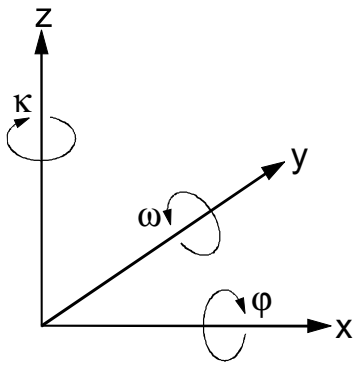
1.3.1 3D-transformasjoner

Vi skal basere formelutviklingen på et rettvinklet koordinatsystem der aksenes relative beliggenhet velges i samsvar med høyrehåndssystemet som vist i figur 1.1. Ved å la høyre hånds tommelfinger peke langs X-aksen og pekefingeren langs Y-aksen, vil langfingeren peke i Z-aksen's retning. Rotasjoner om de tre aksene defineres slik at vi har positiv rotasjon i urviserens bevegelsesretning når vi ser mot origo fra den positive aksehalvdel. De tre rotasjonsvinklene betegnes med henholdsvis φ, ω og κ, hvor φ er rotasjon om X-aksen, ω er rotasjon om Y-aksen og κ er rotasjon om Z-aksen. Dette er for øvrig samme konvensjon som benyttes innen fotogrammetrien.

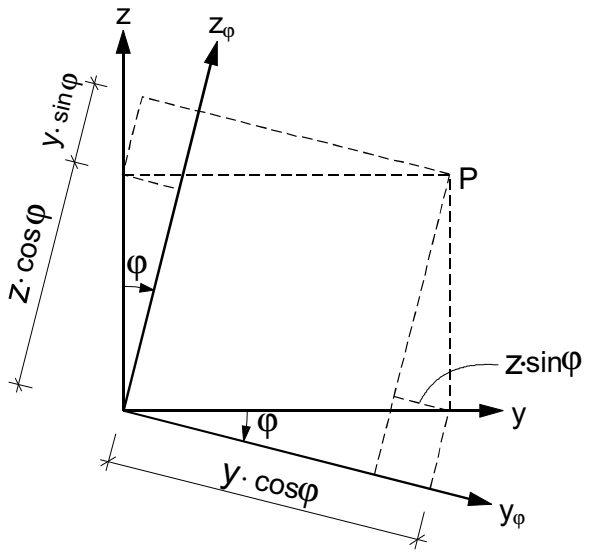
Rotasjon om X-aksen

Ifølge figur 1.2 finner vi koordinatene til punkt P etter at aksesystemet er rotert en positiv vinkel φ om X-aksen:

$$\begin{aligned} x_\varphi &= x, \\ y_\varphi &= y\cos\varphi - z\sin\varphi, \\ z_\varphi &= y\sin\varphi + z\cos\varphi. \end{aligned}$$



Figur 1.1: Definisjon av det tredimensjonale koordinatsystem



Figur 1.2: Rotasjon om X-aksen

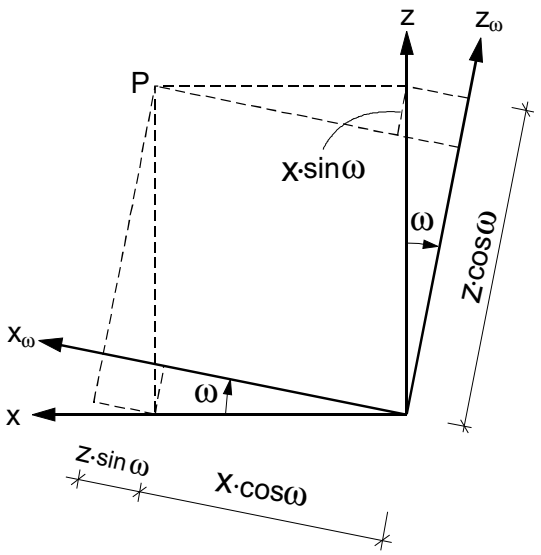
Rotasjonen kan skrives på matriseform som

$$\begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.3}$$

Rotasjon om Y-aksen

Ifølge figur 1.3 finner vi koordinatene til punkt P etter at aksesystemet er rotert en positiv vinkel ω om Y-aksen:

$$\begin{aligned} x_\omega &= x\cos\omega + z\sin\omega, \\ y_\omega &= y, \\ z_\omega &= -x\sin\omega + z\cos\omega. \end{aligned}$$



Figur 1.3: Rotasjon om Y-aksen

Rotasjonen kan skrives på matriseform som

$$\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix} = \begin{bmatrix} \cos\omega & 0 & \sin\omega \\ 0 & 1 & 0 \\ -\sin\omega & 0 & \cos\omega \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.4}$$

Rotasjon om Z-aksen

Ifølge figur 1.4 finner vi koordinatene til punkt P etter at aksesystemet er rotert en positiv vinkel κ om Z-aksen:

$$\begin{aligned} x_\kappa &= x\cos\kappa - y\sin\kappa, \\ y_\kappa &= x\sin\kappa + y\cos\kappa, \\ z_\kappa &= z. \end{aligned}$$

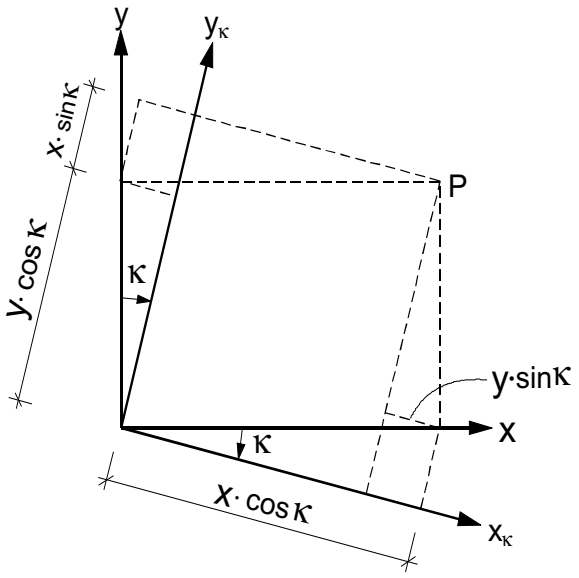
Rotasjonen kan skrives på matriseform som

$$\begin{bmatrix} x_\kappa \\ y_\kappa \\ z_\kappa \end{bmatrix} = \begin{bmatrix} \cos\kappa & -\sin\kappa & 0 \\ \sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.5}$$

Translasjon

Translasjon består i å parallellforskyve koordinataksene til et nytt origo. Ifølge figur 1.5 finner vi koordinatene til punkt P etter at aksesystemet er parallellforskjøvet til punkt (x_Q, y_Q, z_Q):

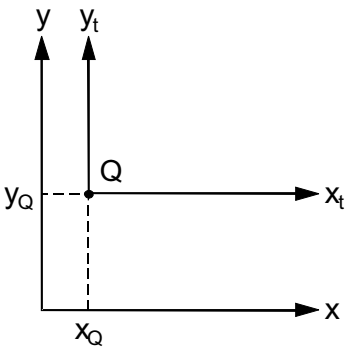
$$x_t = x - x_Q = x + T_x,$$



Figur 1.4: Rotasjon om Z-aksen

$$\begin{aligned} y_t &= y - y_Q = y + T_y, \\ z_t &= z - z_Q = z + T_z. \end{aligned}$$

Ved å benytte homogene koordinater kan translasjonen skrives som

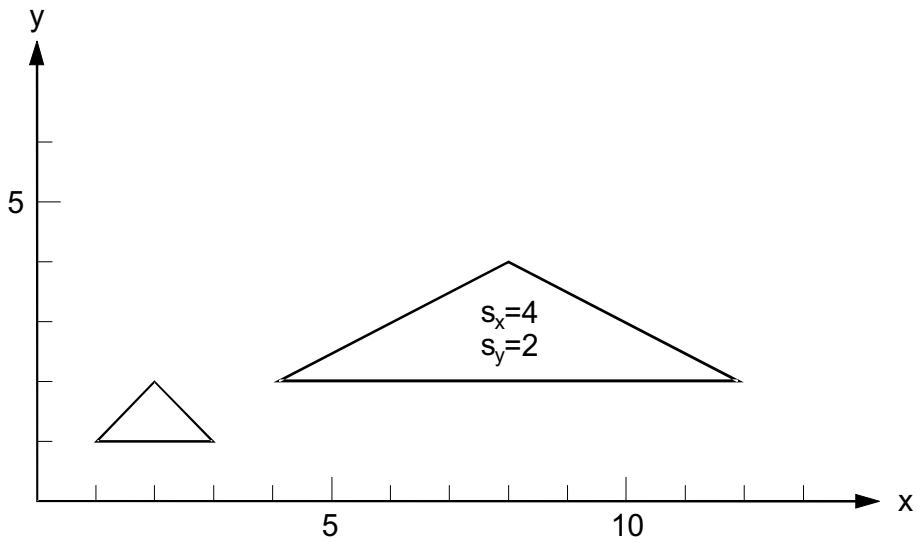


Figur 1.5: Translasjon

$$\begin{bmatrix} x_t \\ y_t \\ z_t \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.6}$$

Skalering

Skalering benyttes for å endre størrelsen på bildet av et objekt. I figur 1.6 utføres



Figur 1.6: Skalering i forhold til origo

skaleringen relativt til origo i aksesystemet. Ved å velge ulik skaleringsfaktor i de tre akseretningene, oppnås en endring av objektets form. Skalerings-transformasjonen er gitt ved

$$\begin{aligned} x_s &= xS_x, \\ y_s &= yS_y, \\ z_s &= zS_z, \end{aligned}$$

eller skrevet på matriseform

$$\begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.7}$$

1.3.2 Transformasjon til et lokalt koordinatsystem i xy-planet

Gitt en punktsverm i xy-planet og det tilfellet at vi skal transformere punktene til et lokalt system med origo i punktet p_1 og Y-akse gjennom punktene p_1 og p_2. Vi benevner koordinater i fra-systemet med (x, y) og koordinater i til-systemet med (x', y'). Transformasjonen skal utføres slik at de to basispunktenes koordinater i til-systemet blir

$$x'_1 = y'_1 = x'_2 = 0 \quad \text{og} \quad y'_2 = s'_{12}.$$

Ved å velge $s'_{12} = s_{12}$ får vi en transformasjon uten målestokksendring. I den etterfølgende utledningen vil vi ta omsyn til en mulig skalering, som imidlertid forutsettes å være lik for begge akseretninger. En slik transformasjon kalles gjerne for Helmert-transformasjon.

Vi formulerer transformasjonen slik at det først utføres en translasjon til punktet p_1, gitt ved $\mathbf{t} = \mathbf{p} - \mathbf{c}$, for deretter å skalere og rotere om p_1; gitt ved $\mathbf{r} = \mathbf{At} = \mathbf{A}(\mathbf{p} - \mathbf{c})$. Den inverse transformasjon er da $\mathbf{p} = \mathbf{A^{-1}r} + \mathbf{c}$. Siden $\mathbf{A}$ er en ortogonal matrise, er $\mathbf{A}^{-1} = \mathbf{A}^t$. Dette gir oss følgende sett av likninger for bestemmelse av matrisen $\mathbf{A}$:

$$\begin{bmatrix} x_2' \\ y_2' \end{bmatrix} = \mathbf{A} \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix}$$

som etter innsetting og litt omforming går over til

$$\begin{bmatrix} 0 \\ s_{12}' \end{bmatrix} = \begin{bmatrix} (x_2 - x_1) & -(y_2 - y_1) \\ (y_2 - y_1) & (x_2 - x_1) \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

Ved å benytte Cramer's regel fås

$$a = \frac{\begin{vmatrix} 0 & -(y_2 - y_1) \\ s_{12}' & (x_2 - x_1) \end{vmatrix}}{\begin{vmatrix} (x_2 - x_1) & -(y_2 - y_1) \\ (y_2 - y_1) & (x_2 - x_1) \end{vmatrix}} = \frac{0 \cdot (x_2 - x_1) + s_{12}'(y_2 - y_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Likningsystemets determinant må forutsettes å være $\neq 0$. Uttrykket kan forenkles til

$$a = \frac{s_{12}'}{s_{12}} \cdot \frac{y_2 - y_1}{s_{12}} = m \cdot \cos\alpha,$$

hvor m er en skaleringsfaktor. Tilsvarende får vi at

$$b = \frac{s_{12}'}{s_{12}} \cdot \frac{x_2 - x_1}{s_{12}} = m \cdot \sin\alpha.$$

Følgelig har vi at

$$m = \sqrt{a^2 + b^2}.$$

$$\begin{array}{l} \mathbf{r} = \mathbf{A}(\mathbf{p} - \mathbf{c}) \\ \mathbf{p} = \mathbf{A}^{-1}\mathbf{r} + \mathbf{c} \\ \textit{hvor :} \\ \mathbf{r} = \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} \quad \mathbf{p} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \\ \\ \mathbf{A}^{-1} = \mathbf{A}^t \qquad \left.\begin{array}{l} a = m \cdot \frac{y_2 - y_1}{s_{12}} \\ \\ b = m \cdot \frac{x_2 - x_1}{s_{12}} \end{array}\right\} \textit{forutsatt} \quad s_{12} \neq 0 \\ \\ m = \frac{s_{12}'}{s_{12}} \\ \\ s_{12} = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \end{array} \tag{1.8}$$

I noen tilfeller ønsker vi å utlede transformasjonsparametrene i en Helmerttransformasjon eller en affin transformasjon, på grunnlag av flere punkter enn det som strengt tatt behøves for å bestemme likningssystemet. Minste kvadraters metode kommer her til anvendelse. Ved å gå veien om tyngdepunktskoodinater til datapunktene, lar sluttede formler for utjevningen seg enkelt utlede.

1.4 Finne skjæringspunktet mellom to linjer i xy-planet

Det å finne skjæringspunktet mellom to rette linjer, kan synes som en triviell oppgave. Problemet er å finne en representasjon av linjene som gir effektive datamaskinprogrammer. Vi skal angi tre metoder: (1) bruk av transformasjoner, (2) punktvektor metode og i et senere kapittel (3) bruk av homogene koordinater.

1.4.1 Bruk av lokal transformasjon

I likningen for en rett linje $y = ax + b$, er a ikke er definert dersom linjen er parallell med Y-aksen. I dette tilfellet har vi ikke lenger en en-til-en forbindelse (en funksjon) mellom elementene $x \in X$ og $y \in Y$, men vi har en en-til-mange forbindelse (en relasjon) fra X til Y. Problemet lar seg omgå ved å gå veien om en lokal transformasjon.

Vi benevner parametrene for de to linjene med a,b og c,d og vi angir de fire endepunktene for linjene med indekser 1 til 4. Vi har følgende likningssystem til bestemmelse av de to linjenes parametre i det lokale koordinatsystemet:

$$\begin{aligned} y_1 &= ax_1 + b, \\ y_2 &= ax_2 + b, \\ y_3 &= cx_3 + d, \\ y_4 &= cx_4 + d. \end{aligned}$$

Det lokale koordinatsystemet velges slik at

$$x_1 = y_1 = x_2 = 0.$$

Skjæringspunktet Q er da gitt ved:

$$y_Q = d \qquad og \qquad x_Q = 0$$

som gir

$$y_Q = \frac{y_3 x_4 - y_4 x_3}{x_4 - x_3} \quad forutsatt \ \ (x_4 - x_3) \neq 0. \tag{1.9}$$

Det gjenstår nå å transformere skjæringspunktets koordinater til det globale koordinatsystemet.

Likning 1.9 byr på et numerisk problem når

$$\mid x_4 - x_3 \mid < \delta,$$

hvor δ er en liten positiv størrelse som velges ut fra datamaskinens regnenøyaktighet. Spesialtilfellet inntrer dersom punkt 3 og 4 er nær sammenfallende, eller når de to linjene er nær parallelle, altså når skjæringspunktet ikke er definert. Dersom punktene 1 og 2 er sammenfallende, vil beregningen terminere på grunn av at den lokale transformasjonen ikke lar seg utføre.

Den angitte metoden gir i alt 47 aritmetiske operasjoner, tilordninger og tester. Metoden er ikke beregningsmessig den mest effektive og anbefales ikke implementert. Den neste metoden som skal presenteres, er langt mere attraktiv.

1.4.2 Punkt-vektor representasjon

Den foregående metoden med transformasjon innebærer at først må vi beregne skjæringspunktet mellom de forlengede linjesegmenter for deretter å undersøke om skjæringspunktet befinner seg innenfor begge linjesegmentene. Det blir derfor helt i slutten av beregningsfasen at vi får svar på om linjene krysser hverandre innenfor linjesegmentene. Ved å anvende den såkalte *punkt-vektor metoden* kan vi tidlig i beregningsfasen avgjøre om linjesegmentene skjærer hverandre eller ikke. Deretter kan et eventuelt skjæringspunkt beregnes. Dette gjør at spørsmål om to linjesegmenter skjære hverandre, kan besvares i løpet av et fåtall aritmetiske operasjoner.

Anta et linjesegment $\mathbf{L}$ mellom punktene P_0 og P_1. Punkt-vektor formen for linjesegmentet er gitt ved en parameterframstilling av linja:

$$\mathbf{L} = \{(x_0, y_0) + (t[x_1 - x_0]\ ,\ t[y_1 - y_0]) \quad \mid \ 0 \leq t \leq 1\} \tag{1.10}$$

som er mengden av alle punkter som ligger på den rette forbindelseslinjen mellom P_0 og P_1. Benevningen punkt-vektor representasjon kommer av at det første leddet i likning 1.10 er representasjonen for et punkt og at det andre leddet kan oppfattes som en vektor. Lengden av vektoren bestemmes av den skalare størrelsen t. Størrelsen t ligger i intervallet $[0, 1]$ for alle punkter som tilhører linjesegmentet. Dersom $t < 0$ eller $t > 1$, ligger punktet henholdsvis til venstre for P_0 eller til høyre for P_1.

Vi kan nå stille opp betingelseslikningen for at to linjesegmenter $\mathbf{L}$ og $\mathbf{L}'$ skjærer hverandre i punktet P_Q, gitt ved

$$P_Q = \mathbf{L} \cap \mathbf{L}'$$

som er det punktet som passer i likningen

$$P_Q = P_0 + t_Q(P_1 - P_0) = P'_0 + t'_Q(P'_1 - P'_0).$$

Ved å sette inn koordinatene til punktene fås

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + t_Q \begin{bmatrix} x_1 - x_0 \\ y_1 - y_0 \end{bmatrix} = \begin{bmatrix} x'_0 \\ y'_0 \end{bmatrix} + t'_Q \begin{bmatrix} x'_1 - x'_0 \\ y'_1 - y'_0 \end{bmatrix} \tag{1.11}$$

som etter litt omforming går over til

$$\begin{bmatrix} (x_1 - x_0) & -(x_1' - x_0') \\ (y_1 - y_0) & -(y_1' - y_0') \end{bmatrix} \begin{bmatrix} t_Q \\ t_Q' \end{bmatrix} = \begin{bmatrix} x_0' - x_0 \\ y_0' - y_0 \end{bmatrix} \tag{1.12}$$

Her er t_Q og t_Q' ukjente størrelser som likningene skal bestemme. Ved å benytte Cramer's regel fås løsningen for de ukjente

$$t_Q = \frac{D_2^{(1)}}{D_2} \tag{1.13}$$

$$t_Q' = \frac{D_2^{(2)}}{D_2} \tag{1.14}$$

hvor de tre determinantene er gitt ved

$$D_2 = \begin{vmatrix} (x_1 - x_0) & -(x_1' - x_0') \\ (y_1 - y_0) & -(y_1' - y_0') \end{vmatrix} \tag{1.15}$$

$$D_2^{(1)} = \begin{vmatrix} (x_0' - x_0) & -(x_1' - x_0') \\ (y_0' - y_0) & -(y_1' - y_0') \end{vmatrix} \tag{1.16}$$

$$D_2^{(2)} = \begin{vmatrix} (x_1 - x_0) & (x_0' - x_0) \\ (y_1 - y_0) & (y_0' - y_0) \end{vmatrix} \tag{1.17}$$

Determinanten D_2 er null dersom de to linjene er parallelle. Dersom $D_2 \neq 0$ og både t_Q og t_Q' ligger i intervallet $[0-\varepsilon, 1+\varepsilon]$, har vi skjæring eller nær skjæring (innenfor ε) mellom linjene.

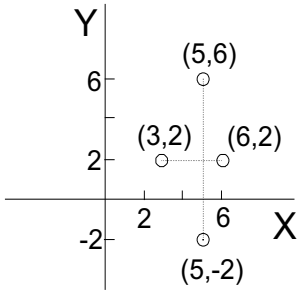
Punkt-vektor metoden tillater at spørsmålet om to linjesegmenter skjærer hverandre, kan besvares ved 24 aritmetiske operasjoner. I tillegg kommer 6 aritmetiske operasjoner for å beregne et eventuelt skjæringspunkt mellom linjene. Dette gir færre aritmetiske operasjoner enn den tidligere angitte transformasjonsmetoden. En annen attraktiv side ved metoden er at skjæringspunktet framkommer ved operasjoner på koordinatdifferanser. Dette vil hindre "overflowdersom koordinatene har store tallverdier.

Vi vil illustrere formlene ved et regneeksempel. La oss anta situasjonen i figur 1.7 der de to linjene er parallelle med henholdsvis X-aksen og Y-aksen; gitt ved punktene $(x_0, y_0) = (3, 2)$, $(x_1, y_1) = (6, 2)$ og $(x_0', y_0') = (5, -2)$, $(x_1', y_1') = (5, 6)$. Dette gir etter likning 1.12

$$\begin{bmatrix} (6-3) & -(5-5) \\ (2-2) & -(6+2) \end{bmatrix} \begin{bmatrix} t_Q \\ t_Q' \end{bmatrix} = \begin{bmatrix} 5-3 \\ -2-2 \end{bmatrix}$$

som går over til

$$\begin{bmatrix} 3 & 0 \\ 0 & -8 \end{bmatrix} \begin{bmatrix} t_Q \\ t_Q' \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \end{bmatrix}$$



Figur 1.7: Eksempel

Dette gir oss determinantene

$$D_2 = \begin{vmatrix} 3 & 0 \\ 0 & -8 \end{vmatrix} = -24$$

$$D_2^{(1)} = \begin{vmatrix} 2 & 0 \\ -4 & -8 \end{vmatrix} = -16$$

$$D_2^{(2)} = \begin{vmatrix} 3 & 2 \\ 0 & -4 \end{vmatrix} = -12$$

Herav får vi

$$t_Q = \frac{-16}{-24} = \frac{2}{3} \quad \text{og} \quad t'_Q = \frac{-12}{-24} = \frac{1}{2}$$

Siden begge t-ene ligger intervallet $[0, 1]$, vil linjesegmentene krysse hverandre mellom de angitte endepunkter. Koordinatene til skjæringspunktet blir i henhold til likning 1.11:

$$P_Q = \begin{bmatrix} 3 \\ 2 \end{bmatrix} + \frac{2}{3} \begin{bmatrix} 6-3 \\ 2-2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

og

$$P_Q = \begin{bmatrix} 5 \\ -2 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 5-5 \\ 6+2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

som viser at vi har funnet et punkt som tilfredsstiller snittbetingelsen for de to linjene.

1.5 Teknikker med homogene koordinater

1.5.1 Grunnleggende betraktninger

Den homogene representasjonen av et objekt i n-dimensjoner er et objekt i $(n + 1)$ dimensjoner. Koordinatene i det n-dimensjonale rommet kalles *ordinære koordinater*

mens de i det $(n+1)$-dimensjonale rom kalles *homogene koordinater*. Et punkt $(x, y) \in \mathbb{R}^2$ er representert ved de homogene koordinater

$$(wx, wy, w).$$

Størrelsen w er en skaleringsfaktor. Den homogene representasjonen framkommer altså ved en transformasjon fra et n-dimensjonalt rom til et $(n+1)$-dimensjonalt rom. De homogene koordinatene transformeres tilbake til ordinære koordinater ved å dividere med w, forutsatt at w er forskjellig fra null. Anta for eksempel et punkt $(x, y, z) \in \mathbb{R}^3$. De ordinære koordinater i det tredimensjonale rom representeres ved de homogene koordinater (wx, wy, wz, w). Siden avbildningen er avhengig av skaleringsfaktoren w, har de ordinære koordinater til et punkt uendelig mange avbildninger i $(n+1)$ rommet. Dette har den praktiske fordel at løsningen av en beregningsoppgave lettere kan tilpasses tallområdet til den aktuelle datamaskin.

Homogene koordinater ble opprinnelig utviklet for å bevise teoremer i den projektive geometri. Det viser seg at metodene kan ha fordeler ved beregninger på geometrier, fordi transformasjoner, skjæringsberegninger o.l. kan uttrykkes på en rasjonell måte i forhold til beregningstid og tester mot numeriske problemer (divisjon med null). Om en aktuell geometrioppgave skal løses med homogene koordinater eller ikke, må vurderes i det enkelte tilfellet. Ulempen med homogene koordinater er at vi utvider matrisene med en ny dimensjon. Dette kan i noen tilfeller føre til økt antall aritmetiske operasjoner i forhold til om vi hadde benyttet ordinære koordinater. Den videre framstilling følger delvis Newman et al. [NS84]. En utdypende behandling av homogene koordinater kan finnes i Maxwell [Max46] og Maxwell [Max51].

1.5.2 Punkter og linjer i et 2-dimensjonalt vektorrom

Representasjon av et punkt i et 2D rom

Et punkt i et todimensjonalt vektorrom kan representeres ved en vektor i et tredimensjonalt vektorrom gitt ved transformasjonen:

$$f : (x, y) \rightarrow \mathbf{v}^t = (v_1, v_2, v_3) = (wx, wy, w). \tag{1.18}$$

Skaleringsfaktoren w kan velges fritt, med unntak av verdien null. Dersom w er null, lar transformasjonen tilbake til det todimensjonale rom seg ikke definere. Forutsatt at $w \neq 0$, fås transformasjon tilbake til ordinære koordinater ved å dividere med w. Den inverse transformasjon er følgelig:

$$f^{-1} : (v_1, v_2, v_3) \rightarrow (x, y) = \left(\frac{v_1}{v_3}, \frac{v_2}{v_3}\right) \qquad forutsatt \;\; v_3 \neq 0. \tag{1.19}$$

Representasjon av en linje i et 2D rom

Betingelsen for at et punkt V befinner seg på linjen L, er at punktets koordinater passer i den skalare likningen:

$$av_1 + bv_2 + cv_3 = a(wx) + b(wy) + c(w) = 0.$$

Dette er ekvivalent med

$$\mathbf{v} \cdot \mathbf{l} = \mathbf{0}$$

som er prikkproduktet mellom punktets vektor $\mathbf{v}$ og linjens vektor $\mathbf{l}$, hvor $\mathbf{l}$ er gitt ved

$$\mathbf{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Dersom innerproduktet ikke er lik null, ligger ikke punktet V på linjen L. Fortegnet til innerproduktet kan benyttes til å avgjøre om punktet ligger på den ene eller den andre siden av L. Følgelig er $\mathbf{v} \cdot \mathbf{l}$ et uttrykk for V's avstand fra L, men for å finne den aktuelle perpendikulære avstanden må innerpoduktet normaliseres. Vi har at

$$\text{avstand} = (\mathbf{v} \cdot \mathbf{l}) \frac{1}{w\sqrt{a^2 + b^2}} \qquad \text{hvor } \; w = \mathbf{v} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{1.20}$$

En rett linje mellom to punkt

Gitt to punkter P og V og deres homogene vektorer $\mathbf{p}$ og $\mathbf{v}$. Betingelsen for at disse to punktene skal ligge på linjen L er gitt ved de prikkproduktene

$$\begin{aligned} \mathbf{p} \cdot \mathbf{l} &= 0, \\ \mathbf{v} \cdot \mathbf{l} &= 0, \end{aligned}$$

hvor vektoren $\mathbf{l}$ representerer den rette linjen L. De to prikkproduktene er begge lik null når vektoren $\mathbf{l}$ er kryssproduktet mellom de to vektorene $\mathbf{p}$ og $\mathbf{v}$. Løsningen for vektoren $\mathbf{l}$ er følgelig gitt ved

$$\mathbf{l} = \mathbf{v} \times \mathbf{p}. \tag{1.21}$$

Kryssproduktet mellom vektorene utledes med utgangspunkt i følgende to likninger:

$$\begin{aligned} ap_1 + bp_2 + cp_3 &= 0, \\ av_1 + bv_2 + cv_3 &= 0. \end{aligned}$$

Ved å multiplisere med henholdsvis v_1 og $-p_1$ og summere fås

$$b(p_2v_1 - v_2p_1) + c(p_3v_1 - v_3p_1) = 0.$$

På grunn av at vi har frihet til å velge verdi for skaleringsfaktoren c, kan vi sette $c = -(p_2v_1 - v_2p_1)$. Ved å innføre denne verdien for c, ser vi at likningen ovenfor er tilfredsstilt når $b = (p_3v_1 - v_3p_1)$. Vi får følgelig denne løsningen for linjens parametre:

$$\mathbf{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} p_2v_3 - p_3v_2 \\ p_3v_1 - p_1v_3 \\ p_1v_2 - p_2v_1 \end{bmatrix} \tag{1.22}$$

Størrelsen $p_1v_2 - p_2v_1$ blir null dersom punktene P og V er sammenfallende. Som vi ser vil denne situasjonen ikke skape numeriske problemer for den homogene representasjonen. Problemet vil derimot oppstå ved en projeksjon tilbake til det todimesjonale rom.

Skjæringspunktet mellom to linjer

Gitt to linjer L og K og deres homogene vektorer

$$\mathbf{l}^t = (a, b, c) \quad \text{og} \quad \mathbf{k}^t = (d, e, f).$$

Det punktet R som ligger i skjæringspunktet mellom L og K, må ha koordinater som passer i de to likningene:

$$\begin{aligned} \mathbf{r} \cdot \mathbf{l} &= 0, \\ \mathbf{r} \cdot \mathbf{k} &= 0. \end{aligned}$$

Vi benytter en tilsvarende teknikk som da likning 1.22 ble utledet og finner at $\mathbf{r}$ er gitt ved kryssproduktet $\mathbf{l} \times \mathbf{k}$. Løsningsvektoren for punktet R blir:

$$\mathbf{r} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} bf - ce \\ cd - af \\ ae - bd \end{bmatrix} \tag{1.23}$$

Ifølge likning 1.19 er transformasjonen tilbake til ordinære koordinater gitt ved

$$(x, y) = \left(\frac{r_1}{r_3}, \frac{r_2}{r_3}\right) \qquad forutsatt \quad r_3 = (ae - bd) \neq 0. \tag{1.24}$$

Størrelsen $r_3 = 0$ i to tilfeller:

1. Dersom de to linjene er parallelle.

2. Dersom de punktene som ligger til grunn for definisjon av en linje, er sammenfallende. Altså i det tilfellet at den homogene koordinaten c i likning 1.22 er lik null.

Ved å benytte homogene koordinater klarer vi å beregne skjæringspunktet mellom to linjer uten å komme opp i numeriske problem av den typen en rett fram løsning med utgangspunkt $y = ax + b$ vil gi. Metoden med homogene koordinater krever 45 aritmetiske operasjoner, tester og tilordninger. Dette er ikke så gunstig som punkt-vektormetoden som baserer seg på likning 1.10.

Talleksempel

Gitt to linjer L og K i det todimensjonale rom og to punkter på hver av linjene. La L være gitt ved punktene (1,1) og (1,4), og la K være gitt ved punktene (0,2) og 2,2). Som vi ser er linjene parallelle med koordinataksene og har skjæringspunktet (1,2). Siden transformasjonen fra ordinære koordinater til homogene koordinater er en *en-til-mange* transformasjon, har vi frihet til å velge skaleringsfaktor. For å illustrere dette velger vi ulike skaleringsfaktorer (henholdsvis 1,2,3 og 4) for de fire transformasjonene og har ifølge likning 1.18:

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 4 \end{bmatrix} \rightarrow \begin{bmatrix} 2 \\ 8 \\ 2 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 2 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 6 \\ 3 \end{bmatrix} \qquad \begin{bmatrix} 2 \\ 2 \end{bmatrix} \rightarrow \begin{bmatrix} 8 \\ 8 \\ 4 \end{bmatrix}$$

Vektorrepresentasjonen (parametrene) for de to linjene er ifølge likning 1.22:

$$\mathbf{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 1 \cdot 2 - 1 \cdot 8 \\ 1 \cdot 2 - 1 \cdot 2 \\ 1 \cdot 8 - 1 \cdot 2 \end{bmatrix} = \begin{bmatrix} -6 \\ 0 \\ 6 \end{bmatrix}$$

$$\mathbf{k} = \begin{bmatrix} c \\ d \\ e \end{bmatrix} = \begin{bmatrix} 6 \cdot 4 - 3 \cdot 8 \\ 3 \cdot 8 - 0 \cdot 4 \\ 0 \cdot 8 - 6 \cdot 8 \end{bmatrix} = \begin{bmatrix} 0 \\ 24 \\ -48 \end{bmatrix}$$

Ved å transformere tilbake til 2-dimensjoner får vi følgende likninger for de to linjene:

$$-1 \cdot x + 0 \cdot y + 1 = 0 \qquad\qquad 0 \cdot x + 1 \cdot y - 2 = 0$$

som gir $x = 1$ og $y = 2$.

Skjæringspunktets koordinater kan finnes med utgangspunkt i likning 1.23:

$$\mathbf{r} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} 0 \cdot (-48) & - & 6 \cdot 24 \\ 6 \cdot 0 & - & (-6) \cdot (-48) \\ -6 \cdot 24 & - & 0 \cdot 0 \end{bmatrix} = \begin{bmatrix} -144 \\ -288 \\ -144 \end{bmatrix}$$

Vi projiserer så tilbake til det todimesjonale rom ved likning 1.19:

$$(-144, -288, -144) \rightarrow (1, 2)$$

som er det resultatet vi ventet å komme fram til.

1.5.3 Punkter, linjer og plan i et 3-dimensjonalt vektorrom

Representasjon av et punkt i et 3D-rom

Et punkt i det tredimensjonale rom $\mathbb{R}^3$ kan representeres ved en vektor i det firedimensjonale rom $\mathbb{R}^4$:

$$\mathbf{v}^t = (v_1, v_2, v_3, v_4) = (wx, wy, wz, w), \tag{1.25}$$

som er en direkte utvidelse av 2D-konseptet.

Representasjon av en linje i et 3D-rom

La endepunktene for linjesegmentet være representert ved vektorene $\mathbf{p}$ og $\mathbf{r}$. Linjen mellom punktene kan defineres parametrisk ved

$$\mathbf{v} = \mathbf{p} + t(\mathbf{r} - \mathbf{p}) = \begin{bmatrix} \mathbf{p} & (\mathbf{r} - \mathbf{p}) \end{bmatrix} \begin{bmatrix} 1 \\ t \end{bmatrix} = \mathbf{L} \begin{bmatrix} 1 \\ t \end{bmatrix} \tag{1.26}$$

hvor $0 \leq t \leq 1$ og hvor $\mathbf{L}$ er en 4×2 matrise. $\mathbf{L}$ formuleres slik at t går fra null i det ene endepunktet av linjesegmentet til 1 i det andre endepunktet. Dersom t ligger utenfor det angitte intervallet, definerer t et punkt som ligger på forlengelsen til linjesegmentet. Som vi ser, er dette en formulering som tilsvarer punkt-vektorformuleringen gitt i likning 1.10.

Eksempel: finn den parameteriserte representasjonen av linjen gjennom 3D-punktene (1,0,1) og (0,1,1). Vi transformerer til homogene koordinater og velger $w = 1$. De homogene koordinater for punktene blir da (1,0,1,1) og (0,1,1,1). Linjen gjennom de to gitte punkter er da definert ved

$$\mathbf{v} = \begin{bmatrix} 1 & (0-1) \\ 0 & (1-0) \\ 1 & (1-1) \\ 1 & (1-1) \end{bmatrix} \begin{bmatrix} 1 \\ t \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ t \end{bmatrix}$$

Dersom vi for eksempel velger t=1/2, får vi punktet

$$\mathbf{v} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1 \\ 1 \end{bmatrix}$$

som er det punktet som ligger midtveis mellom endepunktene til linjesegmentet.

Representasjon av et plan i et 3D-rom

Betingelsen for at et punkt V befinner seg i planet Γ, er at punktets koordinater passer i den skalare likningen

$$av_1 + bv_2 + cv_3 + dv_4 = a(wx) + b(wy) + c(wz) + d(w) = 0.$$

Dette kan uttrykkes ved prikkproduktet mellom punktets vektor og planets vektor:

$$\mathbf{v} \cdot \gamma = \mathbf{0}, \tag{1.27}$$

hvor planets vektor er gitt ved

$$\gamma = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \tag{1.28}$$

Avstanden fra et punkt til et plan

Fortegnet til innerproduktet kan benyttes til å finne ut på hvilken side av planet Γ punktet V ligger. Følgelig er $\mathbf{v} \cdot \gamma$ et uttrykk for V's avstand fra Γ, men for å finne den aktuelle perpendikulære avstanden må innerpoduktet normaliseres. Vi har at

$$\text{avstand} = (\mathbf{v} \cdot \gamma)\frac{1}{w\sqrt{a^2+b^2+c^2}} \qquad \text{hvor } \; w = \mathbf{v} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{1.29}$$

Skjæringspunktet mellom en linje og et plan

Skjæringspunktet mellom en linje og et plan finnes enkelt ved å representere linjen på parameterform. Problemet reduseres derved til å bestemme parameteren t til skjæringspunktet V. Skjæringspunktet finnes følgelig ved å løse likningen

$$\mathbf{v} \cdot \gamma = \mathbf{0}. \tag{1.30}$$

Ved å innføre uttrykkene for vektor $\mathbf{v}$ og γ fra likningene 1.26 og 1.28, fås følgende løsning for t:

$$t = \frac{-[p_1a + p_2b + p_3c + p_4d]}{(r_1-p_1)a + (r_2-p_2)b + (r_3-p_3)c + (r_4-p_4)d} = \frac{D_1}{D} \tag{1.31}$$

hvor D forutsettes $\neq 0$.

Skjæring mellom tre plan

Skjæringspunktet mellom tre plan finnes ved å løse de tre likningene

$$\begin{aligned} \mathbf{v} \cdot \gamma_1 &= 0, \\ \mathbf{v} \cdot \gamma_2 &= 0, \\ \mathbf{v} \cdot \gamma_3 &= 0. \end{aligned}$$

Vi har her tre likninger til bestemmelse av skjæringspunktets fire koordinater. Følgelig må vi skaffe oss en likning til. Siden vi har frihet til å velge skaleringsfaktoren w, kan vi etablere den fjerde likningen ved å sette $v_4 = w$. Likningssystemet kan da skrives som

$$\begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ w \end{bmatrix} \tag{1.32}$$

Tre punkter definerer et plan

Tre punkter P, R og V definerer et plan dersom de ikke er kollineære. Vi krever at $\mathbf{p} \cdot \gamma = \mathbf{0}$, $\mathbf{r} \cdot \gamma = \mathbf{0}$ og $\mathbf{v} \cdot \gamma = \mathbf{0}$. Siden vi har frihet til å velge skaleringsfaktoren, skaffer vi oss på dette grunnlaget den fjerde likningen ved å sette $d = k$. For k kan vi velge alle tall som er forskjellige null.

$$\begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ r_1 & r_2 & r_3 & r_4 \\ v_1 & v_2 & v_3 & v_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ k \end{bmatrix} \tag{1.33}$$

Dersom koeffisientmatrisen Q lar seg invertere, har vi at

$$\gamma = \mathbf{Q}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ k \end{bmatrix}$$

1.5.4 Transformasjoner ved bruk av homogene koordinater

Digitale terrengmodeller kan benyttes til å lage perspektivbilder av terrenget. Dette krever at punkter transformeres mellom ulike koordinatsystem. Transformasjonene vil innbefatte skalering, rotasjon, translasjon og overgang til sentralprojeksjon. Ved å anvende homogene koordinater lar disse transformasjonene seg kjede sammen ved matrisemultiplikasjoner. I de tilfeller at vi ønsker å beregne en serie bilder (animasjoner), kan det av hensyn til beregningstiden lønne seg å benytte differensielle teknikker. Dette vil ikke bli behandlet her.

Translasjon

Anta et 3D-punkt med de homogene koordinater $(x, y, z, 1)$, hvor vi har satt skaleringsfaktoren w lik 1. En 3D-translasjon av dette punktet gir

$$\begin{aligned} x' &= x + T_x, \\ y' &= y + T_y, \\ z' &= z + T_z. \end{aligned}$$

Disse likningene kan skrives på matriseform som

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.34}$$

Skalering

Vi benytter en tilsvarende framgangsmåte som under translasjon og får følgende uttrykk for skaleringen:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.35}$$

Rotasjon

En 3D-rotasjon kan dekomponeres i tre rotasjoner; en rotasjon om hver av aksene. Hvilken akse vi velger å rotere om først, er likegyldig bare man er konsekvent.

Vi betegner de tre rotasjonsmatrisene $\mathbf{R}_1$, $\mathbf{R}_2$ og $\mathbf{R}_3$. Den tredimensjonale rotasjonen er da gitt ved

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \mathbf{R_3 R_2 R_1} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.36}$$

Sammenkjeding av transformasjoner

Siden vi har uttrykt de tre basisoperasjonene translasjon, skalering og rotasjon ved hjelp av matrisemultiplikasjoner, lar disse seg kjede sammen til en enkelt transformasjonsmatrise. Anta for eksempel at vi skal utføre en translasjon $\mathbf{T}_1$, deretter en skalering $\mathbf{S}$, så en rotasjon $\mathbf{R}$ og til slutt en translasjon $\mathbf{T}_2$. Det som nå er viktig, er at en er påpasselig med den rekkefølgen matrisemultiplikasjonene utføres i. Som kjent har en jo i alminnelighet at $\mathbf{PQ} \neq \mathbf{QP}$. Vi får i vårt eksempel følgende uttrykk:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \mathbf{T}_2\mathbf{R}\mathbf{S}\mathbf{T}_1 \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.37}$$

Ved at vi slår sammen de fire matrisene til en matrise, vil vi for samtlige punkter som skal transformeres, ha redusert antall matrisemultiplikasjoner fra fire til en. Dette er en meget attraktiv side ved den måten vi har anvendt homogene koordinater på.

Det kan se ut som om vi må utføre en matrisemultiplikasjon med en 4×4 matrise. Dette er imidlertid ikke tilfellet, siden den fjerde linjen til $\mathbf{M}$ alltid vil være (0, 0, 0, 1). En forkortet beregning kan derfor basere seg på

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.38}$$

Del II

Modellering av terrengoverflater

Kapittel 2

Digitale terrengmodeller

2.1 Innledning

En digital terrengmodell, som også kalles digital høydemodell, beskriver formen til en terrengoverflate. I sin tid ble begrepet digitale terrengmodeller brukt i en noe videre betydning enn begrepet digitale høydemodeller, men i dag virker det som om begrepene brukes med identisk meningsinnhold. Begrepet digital henspeiler på at modellen inneholder informasjon om terrengets høydeforhold (topografi) i et utvalg av datapunkter. Terrengoverflaten er en kontinuerlig flate som ved digitalisering blir representert ved diskrete datapunkter. Begrepet modell henspeiler på at for å kunne beregne høyden til et vilkårlig valgt punkt, må det foreligge en beregningsprosedyre; en interpolasjonsmetode. En digital terrengmodell består altså av en punktmengde der vi kjenner (x, y, h) og en interpolasjonsmetode som gjør det mulig å beregne h når (x, y) er gitt.

Digitale terrengmodeller ble tatt i bruk i slutten av 1960 årene. I dag spiller digitale terrengmodeller en viktig rolle innenfor en rekke anvendelser. Bruken av nøyaktige GPS-mottakere har blant annet bidratt til å aktualisere digitale terrengmodeller innenfor navigasjon og styring av bevegelige objekter. Noen tradisjonelle anvendelser av digitale terrengmodeller er:

- beregne volum ved terrenginngrep;
- beregne volum av grusforekomster, vannmagasin o.l.;
- beregne høyder langs profillinjer, benyttes gjerne i forbindelse med veiplanlegging;
- beregne terrenghelning og gradient;
- studere siktforhold, for eks. finne ut hvilke deler av terrengoverflaten en ikke kan se fra en gitt standplass;

- kartografisk framstilling av terrengets topografi, som for eksempel perspektivtegninger, analytisk skyggelegging, helningskart, isolinjekart etc.

- animasjon, for eks. simulere at en vandrer rundt i et terreng.

- styring av bevegelige objekter som fly, båter, biler, missiler etc.

Tradisjonelt forutsetter digitale terrengmodeller at det til en gitt (xy)-koordinat tilordnes kun en h-verdi. Altså funksjonen $f : xy \rightarrow h$ forutsettes å være en-til-en. Den inverse funksjon $f^{-1} : h \rightarrow xy$ må selvsagt være en-til-mange. Kravet om at f må være en-til-en gjør det umulig å modellere terreng med overheng. I praksis representerer dette ikke noe stort problem, fordi overheng på terrengoverflaten forekommer sjelden. Terrengmodeller av den nevnte typen kalles ofte for 2.5D modeller. Dette for å signalisere at det ikke dreier seg om ekte 3D modeller. Vi vil i de etterfølgende presentasjoner legge hovedvekten på 2.5D modeller, men vi vil også presentere en trianguleringsalgoritme som bygger opp en 3D modell.

2.2 Klassifikasjon av digitale terrengmodeller

Digitale terrengmodeller kan klassifiseres på grunnlag av de krav modellene stiller til diskretiseringsmønster og de interpolasjonsmetoder som benyttes.

- Diskretiseringsmønstre:

    1. Terrenglinjer
    2. Parallelle profiler
    3. Isolinjer
    4. Rutenett
    5. Vilkårlig fordelte punkt
    6. Kombinerte metoder

- Interpolasjonsmetoder:

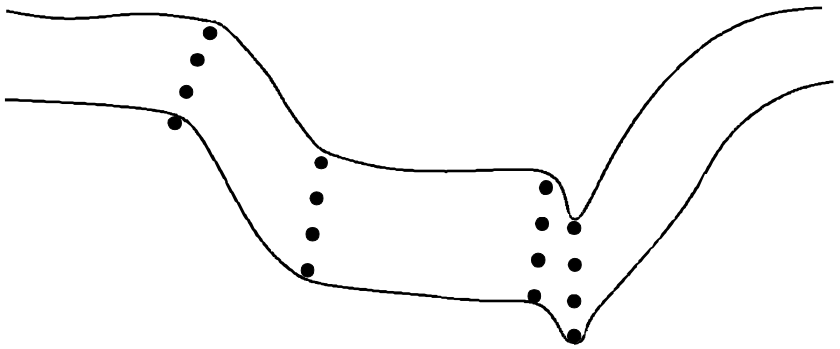
    1. Interpolasjon ved bruk av globale funksjoner
    2. Fasettmetoder (interpolasjon ved bruk av lokale flater) : interpolasjon i et rutenett, interpolasjon i et trekantnett, spline-funksjoner
    3. Punktvis interpolasjon: flytende flate (eng. moving surface), kriging
    4. Kombinerte metoder

2.3 Diskretiseringsmønstre

De krav terrengmodellen stiller til diskretiseringsmønstret (samplingsmønster), bør innordne seg etter de metoder og det utstyr som benyttes til datafangsten. Det er derfor ideelt om en terrengmodell ikke legger begrensninger på diskretiseringsmønstret, men aksepterer det valg av diskretiseringsmønster som er funnet hensiktsmessig under digitaliseringen. Valg av diskretiseringsmønster og antall datapunkter er avgjørende for hvor nøyaktig terrengmodellen beskriver topografien til den virkelige terrengoverflaten.

2.3.1 Knekklinjer

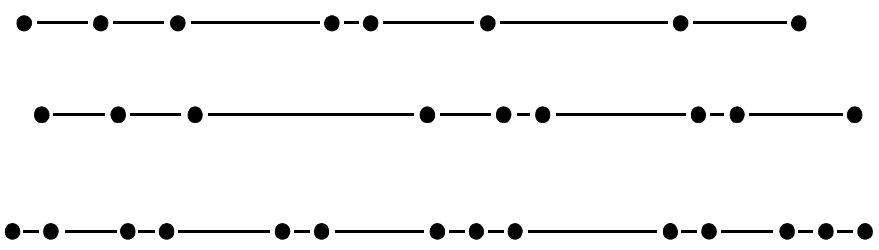
Med *knekklinjer* eller terrenglinjer mener vi linjer som definerer hvor vi har markante endringer i terrengoverflatens helning; se figur 2.1. Eksempler på terrenglinjer er: vegkant, topp og fot av skrent. Det faller umiddelbart logisk og naturlig å sørge for at diskretiseringen gir en god beskrivelse av knekklinjene.



Figur 2.1: Eksempel på diskretisering langs terrenglinjer

2.3.2 Parallelle profiler

Diskretiseringsmønstret defineres i dette tilfellet ved en mengde parallelle linjer. Langs linjene kan datapunktene velges med vilkårlig avstand. Metoden er populær i

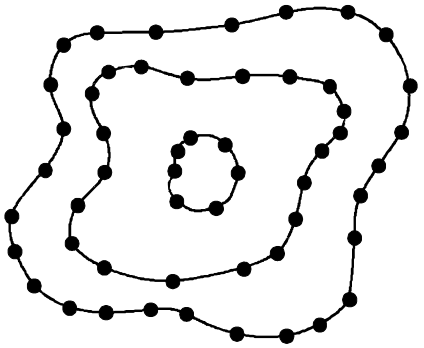


Figur 2.2: Eksempel på diskretisering langs parallelle linjer

forbindelse med vegplanlegging. Ved å bestemme terrenghøyder langs snitt vinkelrett på vegens lengdeakse, blir det matematisk sett en enkel oppgave å beregne masser i fyllinger og skjæringer. Ulempen ved å diskretisere langs parallelle linjer, er at markante former mellom profillinjene ikke lar seg fange opp. Dette kan imidlertid kompenseres ved å gjøre avstanden mellom linjene tilstrekkelig liten.

2.3.3 Isolinjer

Når fotogrammetriske instrumenter benyttes til datafangsten, er det ofte hensiktsmessig å digitalisere skjæringslinjene mellom parallelle plan og terrengoverflaten. Det er vanlig å legge planene parallelt med referanseflaten for høydeangivelsen. Ulempen



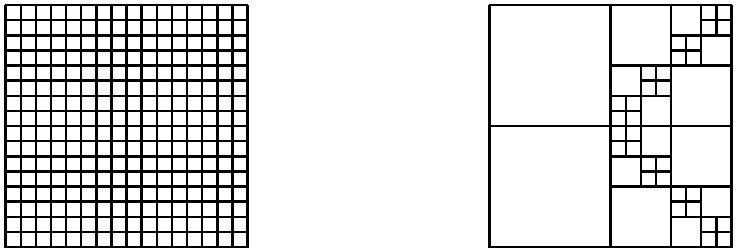
Figur 2.3: Eksempel på diskretisering langs isolinjer

ved metoden er av samme karakter som for parallelle profiler. En forskjell er at parallelle profiler gjerne legges med fast horisontal avstand, mens isolinjene ligger med fast vertikal avstand. Dette gjør at jo brattere terrenget er, jo mindre blir den horisontale avstanden mellom isolinjene.

2.3.4 Rutenett

Rutenettsmodeller er svært utbredt, noe som i hovedsak skyldes modellenes enkle struktur; de lar seg lett representere ved 2D-tabeller eller array. Diskretiseringsmønstret framkommer ved at vi legger et horisontalt rutenett over terrengoverflaten og velger datapunkter i de enkelte rutenettskryss. Rutenettet kan utformes på flere måter. Den enkleste utforming fås ved å benytte *fast rutestørrelse* over hele modellens definisjonsområde. Ulempen ved dette er at datatettheten ikke lar seg dynamisk tilpasse terrengets ruhet.

En større grad av fleksibilitet oppnås ved å benytte *variabel rutestørrelse*. Uten å miste rutemodellens enkle implementasjon, er en måte å realisere variabel rutestørrelse å dele området i større blokker med en regulær ruteoppdeling innenfor hver av blokkene. Den totale modellen kan derved oppfattes som unionen av en



fast rutestørrelse variabel rutestørrelse

Figur 2.4: Eksempel på diskretisering i et rutenett. Rutenettet kan utformes med fast rutestørrelse eller variabel rutestørrelse. I det siste tilfellet er kvadtreprinsippet lagt til grunn for variasjon av rutestørrelsen.

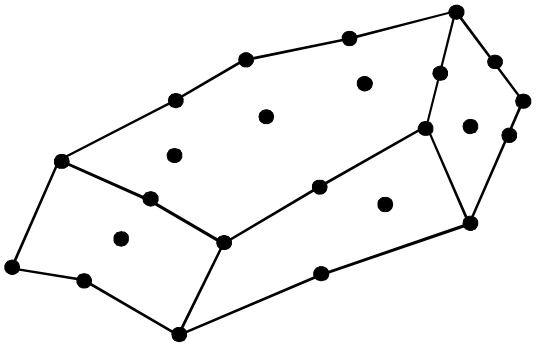
rekke delmodeller med individuell oppløsning. Rutestørrelsen innenfor en blokk bestemmes ut fra behovet for modellens oppløsning. Et annet alternativ er å foreta opprutingen i tråd med en *kvadtretankegang* som vist i figur 2.4.

2.3.5 Vilkårlig fordelte datapunkt

Med vilkårlig fordelte datapunkt mener vi ikke at fordelingen kan gjøres planløst, men at terrengmodellen ikke stiller bestemte krav til utformingen av diskretiseringsmønstret. Uansett hvilken interpolasjonsmetode som benyttes, er det om å gjøre å sample lurt, d.v.s. velge karakteristiske terrengpunkter. Fordelen med at modellen aksepterer vilkårlig fordelte datapunkt, ligger i at datapunktene kan plasseres på strategiske steder i terrenget. På denne måten kan vi la datapunktenes tetthet og plassering rette seg etter terrengets ruhet som vist i figur 2.5. Datapunktenes plassering er helt avgjørende for modellens nøyaktighet. For å kunne ta stilling til hvor tett datapunktene må ligge og hvor de bør plasseres, må vi vite noe om de prinsipper modellens interpolasjon bygger på.

2.3.6 Kombinerte metoder

Ulempen ved å diskretisere etter repeterende mønstre som rutenett, parallelle linjer etc., kan oppveies ved å supplere med knekklinjer. På dette viset kan rutenettsmodeller bli i stand til å takle markante terrengkanter uten at rutestørrelsen må bli svært liten.



Figur 2.5: Eksempel på diskretisering ved strategisk plassering av datapunktene.

2.4 Samplingsstrategi

Samplingen kan baseres på visuell bedømmelse av hvor datapunktene helst bør plasseres eller digitalisering med visse mellomrom basert på differanser i tid eller avstand i (X, Y)-planet. Ved bruk av multistråleekkolodd, for eksempel, samples automatisk datapunkter med visse tidsintervaller. Ved bruk av fotogrammetriske arbeidsstasjoner er det mulig å benytte progressiv sampling. Prinsippet her er at maskinen på grunnlag av de målte punkter evaluerer hvor neste målepunkt bør ligge. En fordel med en slik metode er at tettheten på datapunktene blir tilpasset terrengets formvariasjon. Progressiv sampling er en iterativ prosedyre. Først gjøres en glissen sampling, deretter fortettes punktmengden gradvis inntil modellen holder det fastsatte nøyaktighetskriteriet.

2.5 Resampling

Dersom terrengmodellen forlanger et bestemt diskretiseringsmønster, for eks. et rutenett, kan vi fra en opprinnelig punktmengde avlede en ny punktmengde som tilfredsstiller det ønskede samplingsmønstret. Dette kalles for *resampling*. Digitaliseringen kan i første omgang utføres uten at vi binder oss til å velge datapunktene i det fastsatte mønsteret. I neste omgang interpoleres h-verdien til punkter i det fastsatte diskretiseringsmønstret. De resamplede punktene benyttes så for å definere terrengmodellen. Metoden benyttes ofte i forbindelse med rutenettsmodeller. Ulempen ved metoden er tapet av nøyaktighet som følge av at originale observasjoner erstattes med avledede observasjoner. Resamplingen utføres gjerne på grunnlag av flytende flate eller kriging.

2.6 Interpolasjonsmetoder

Oppgaven til en interpolator i en digital terrengmodell er å definere overflaten til det virkelige terrenget på grunnlag av en mengde diskrete datapunkt. Vi benevner

interpolatoren I, mengden av datapunkter $\mathbf{K}$, mengden av alle punkter på den virkelige terrengoverflaten $\overline{\mathbf{R}}$ og den interpolerte terrengoverflaten $\mathbf{R}$. Anta at vi har en mengde diskrete datapunkter gitt ved

$$\mathbf{K} = \{(x_i, y_i, h_i) \mid i = 1, n\}.$$

Interpolatorens oppgave er definert ved

$$\begin{cases} \mathbf{R} = \{(x, y, h) \mid \ (x, y, h) \in \ I(\mathbf{K})\} \\ \mathbf{K} \subset \mathbf{R} \end{cases}$$

Dersom vi ikke krever at $\mathbf{K}$ skal være en delmengde av $\mathbf{R}$, åpnes for den mulighet at I kan definere en generalisert modell av terrenget. På grunn av at I ikke er i stand til å rekonstruere det virkelige terrenget eksakt, vil $\mathbf{R} \neq \overline{\mathbf{R}}$. Vi må også regne med en viss usikkerhet i bestemmelsen av $\mathbf{K}$ slik at vi må addere visse beløp for at datapunktene skal bli en delmengde av $\overline{\mathbf{R}}$; gitt ved

$$\{(x_i, y_i, h_i + \varepsilon_i) \mid i = 1, n\} \subset \overline{\mathbf{R}}.$$

Det benyttes en rekke interpolatorer innen digitale terrengmodeller. Valg av interpolator vil bero på:

Datapunktenes tetthet Ved stor punkttetthet kan vi benytte mindre nøyaktige interpolatorer enn ved glissent datagrunnlag.

Prosesseringstida For sanntidsapplikasjoner spiller prosesseringstida en avgjørende rolle. I andre tilfeller kan prosesseringstida være mindre avgjørende, mens det er nøyaktigheten til interpolasjonen som er den kritiske faktoren.

Interpolatorens nøyaktighet Enkelte interpolatorer har en tendens til å runde av terrengformer; noe som for visse anvendelser kan være uakseptabelt. Generelt kan sies at ulike interpolatorer kan gi ganske forskjellige resultat. Det er derfor viktig å kjenne interpolatorens matematiske grunnlag og ha en formening om i hvilke situasjoner den kan produsere uakseptable resultater. Kriging framstår som en nøyaktig interpolator.

Vi har tre mulige strategier for valg av interpolator:

Global funksjon Dette vil vanligvis være en ikke-farbar vei om metoden forsøkes anvendt over store terrengområder. I forbindelse med trendanalyser er metoden anvendbar.

Fasettmetoder Terrengoverflaten deles inn i segmenter og det tilordnes en lokal funksjon til hvert segment. Til sammen vil de lokale funksjonene interpolere terrengoverflaten i hele modellens definisjonsområde. Denne metoden benyttes av rutenettsmodeller og trekantmodeller. En annen betegnelse på metoden er *spline* eller stykkvise polynomer.

Punktvis interpolasjon Metoden krever at for hvert eneste punkt som skal interpoleres, må det beregnes et nytt sett av parametre i interpolasjonsfunksjonen. Dette gjør at metoden blir beregningsmessig ressurskrevende. Flytende flate og kriging hører til denne gruppen interpolatorer.

Kombinerte metoder I noen tilfeller kan det være aktuelt å kombinere for eksempel punktvis interpolasjon og lokale flater. Begrunnelsen for dette kan være at vi ønsker å fortette datamengden før de lokale flatene dannes. Derved kan det oppnås en bedre nøyaktighet enn om de lokale flatene ble dannet på grunnlag av den opprinnelige datamengden.

2.6.1 Interpolasjon i et triangelnett

Interpolasjon i et triangelnett er en fasettmetode. Det innebærer at trianglene får tilordnet hver sin interpolator, og at en interpolator ikke benyttes utenfor det triangel den er tilordnet.

Lineær interpolasjon

Det mest nærliggende er å velge en lineær interpolator gitt ved

$$h = a_0 x + a_1 y + a_2.$$

Dette er likningen for et plan hvis parametre bestemmes på grunnlag av vedkommende triangels tre hjørner. Ved å transformere til et lokalt koordinatsystem slik at $y_1 = x_1 = x_2 = 0$, blir løsningen for parametrene meget enkel. Vi får

$$\begin{aligned} a_2 &= h_1, \\ a_1 &= (h_2 - h_1)/y_2 \text{ hvor } y_2 \neq 0, \\ a_0 &= (h_3 - a_1 y_3 - h_1)/x_3 \text{ hvor } x_3 \neq 0. \end{aligned}$$

Som vi ser vil likningssystemet ikke gi en entydig løsning dersom de tre punktene er kollineære (punktene ligger på den samme rette linjen) eller om punktene ligger i et vertikalt plan. Selv om det er en ulempe at vertikale plan ikke lar seg definere i en 2.5D terrengmodell, vil dette normalt ikke være noe stort praktisk problem.

6-parameters polynom

Enkelte har forsøkt å benytte andre interpolatorer enn den lineære ved interpolasjon i triangelnett. Motivet for dette har vært å gi isolinjer en avrundet form i motsetning til det noe kantete utseendet den lineære interpolatoren gir. En slik interpolator er beskrevet av McLain [McL76]. En flate bestemmes for hvert av trianglets hjørner ved en flytende flate"metode (denne beskrives senere). McLain [McL76] benytter til dette de 6 første parametre i det bikubiske polynomet. I hvert triangel finnes så den

endelige interpolasjon av isolinjene ved et vektet middeltall av de tre flatene som er tilordnet triangelhjørnene; gitt ved

$$F = \frac{w_1 f_1 + w_2 f_2 + w_3 f_3}{w_1 + w_2 + w_3} \tag{2.1}$$

For å sikre en glatt overgang fra det ene triangel til det andre, bestemmes vektene slik at w_i er null langs den siden som er motstående til det i-te hjørnet. Dette oppnås ved å gjøre w_i proporsjonal med n-te potens d_i^n til avstanden fra den aktuelle trekantsiden. McLain [McL76] anbefaler å velge $n = 3$. Utrekning av vektene krever at det for hvert interpolert punkt beregnes tre avstander; en avstand til hver av sidene. Dette kan imidlertid gjøres rasjonelt ved å benytte homogene koordinater og beregne parametrene for de tre linjene en gang for alle. Linjenes parametre skaleres slik at $d_i = 1$ for hjørne i.

2.6.2 Interpolasjon i et rutenett

Interpolasjon i et rutenett er også en fasettmetode. Tidligere ble denne typen interpolasjon mye benyttet.

Bilineær interpolasjon

Den bilineære interpolasjon har fire parametre, som bestemmes på grunnlag av hjørnene til vedkommende rute; gitt ved

$$h = a_0 x + a_1 y + a_2 xy + a_3.$$

12-parameters polynom

Enkelte benytter 12 av de 16 leddene til det bikubiske polynomet. Ved å sette betingelser for helningen; de partiellderiverte med omsyn til x og y i de to akseretninger, kan vi skaffe oss to ekstra likninger for hvert rutehjørne. Sammen med den likningen som kan stilles opp med utgangspunkt rutehjørnets h-verdi, får vi i alt tre likninger. Dette gir oss til sammen 12 likninger for hver rute. Spørsmålet er hvordan vi kan skaffe oss informasjon om helningen i rutehjørnene. Måten dette kan gjøres på, er å anvende en flytende flate"metode. Vi bestemmer altså en flate for hvert rutehjørne og benytter denne flaten til å beregne helningen i de to akseretningene. Den totale flaten kan selvsagt ikke bli glatt over alt, så lenge det bare er i rutehjørnene vi har innført betingelsen om glatthet. Langs sidekantene kan metoden gi diskontinuitet i høydeverdiene, siden vi ikke har innført en fullstendig betingelse om kontinuitet. For en videre utdyping av dette emneområdet vises til den generelle litteraturen om spline.

2.6.3 Flytende flate

Alle interpolasjonsmetoder som for hvert interpolert punkt krever at det beregnes en flate, hører til gruppen *flytende flate* (eng. moving surface) interpolasjon. Som interpolasjonsflate benyttes et polynom med inntil 16 parametre. Polynom av høyere grad forsøker en å unngå på grunn av den *oscillerende* effekt. Interpolasjonspolynomet defineres algebraisk ved ett eller flere ledd av det bikubiske polynom

$$\begin{aligned} z &= a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 + a_{30}x^3 + a_{21}x^2y + \\ &\quad + a_{12}xy^2 + a_{03}y^3 + a_{31}x^3y + a_{22}x^2y^2 + a_{13}xy^3 + a_{32}x^3y^2 + \\ &\quad + a_{23}x^2y^3 + a_{33}x^3y^3. \end{aligned} \tag{2.2}$$

Det som kjennetegner flytende flate metoder, er at flatens parametre blir bestemt ved en minste kvadraters tilpassing og at referansepunktene inngår med vekter som avtar monotont med avstanden til det interpolerte punktet. Dette gjør at for hvert interpolert punkt må det beregnes nye vekter, som i sin tur krever at et nytt sett parametre må beregnes. Metoden blir derfor beregningsmessig ressurskrevende.

Minste kvadraters tilpassing

Avviket mellom interpolert høyde og gitt høyde i datapunktene defineres ved

$$\mathbf{v} = \mathbf{f} - \mathbf{B}\boldsymbol{\Delta}, \tag{2.3}$$

hvor

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \quad \mathbf{f} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1u} \\ b_{21} & b_{22} & \cdots & b_{2u} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nu} \end{bmatrix} \quad \boldsymbol{\Delta} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_u \end{bmatrix}$$

Her er

n antall datapunkter;

u antall ukjente parametre;

$\mathbf{f}$ en vektor over de gitte høydeverdier til datapunktene;

$\boldsymbol{\Delta}$ en vektor over de ukjente parametre til det valgte interpolasjonspolynomet;

og $\mathbf{B}$ en koeffisientmatrise der b_{ij} er koeffisienten til den ukjente parameter j for datapunktet i.

Vi bestemmer $\mathbf{\Delta}$ slik at

$$\sum_{i=1}^{n} p_i v_i v_i = \min, \tag{2.4}$$

hvor p-ene er vekter. Fastsettelse av vektene representerer et springende punkt. Siden flytende flate metoder benytter heuristiske metoder for fastsettelse av vektene, gis ikke noe entydig svar på hvordan disse skal fastsettes. En vanlig framgangsmåte er å sette

$$p_i = \frac{1}{d_i^2}$$

hvor d er avstanden mellom det i-te datapunktet og det punktet høyden skal interpoleres i. Likning 2.4 kan skrives på matriseform som

$$\mathbf{v}^t\mathbf{P}\mathbf{v} = \min, \tag{2.5}$$

der $\mathbf{P}$ er en diagonalmatrise over vektene til datapunktene; gitt ved

$$\mathbf{P} = \begin{bmatrix} p_{11} & 0 & 0 & \cdots & 0 \\ 0 & p_{22} & 0 & \cdots & 0 \\ 0 & 0 & p_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & p_{nn} \end{bmatrix}$$

Vi kombinerer de to likningene 2.3 og 2.5 og får

$$\begin{aligned} \mathbf{v}^t\mathbf{P}\mathbf{v} &= (\mathbf{f} - \mathbf{B}\mathbf{\Delta})^t\,\mathbf{P}(\mathbf{f} - \mathbf{B}\mathbf{\Delta}) \\ &= (\mathbf{f}^t - \mathbf{\Delta}^t\mathbf{B}^t)\mathbf{P}(\mathbf{f} - \mathbf{B}\mathbf{\Delta}) \\ &= \mathbf{f}^t\mathbf{P}\mathbf{f} - \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{f} - \mathbf{f}^t\mathbf{P}\mathbf{B}\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B}\mathbf{\Delta}. \end{aligned} \tag{2.6}$$

Minimumsvilkåret i likning 2.5 finnes ved

$$\frac{\partial(\mathbf{v}^t\mathbf{P}\mathbf{v})}{\partial\mathbf{\Delta}} = 0.$$

Det totale differensial er gitt ved

$$\begin{aligned} d(\mathbf{v}^t\mathbf{P}\mathbf{v}) &= -d\mathbf{\Delta}^t \cdot \mathbf{B}^t\mathbf{P}\mathbf{f} - \mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + d\mathbf{\Delta}^t \cdot \mathbf{B}^t\mathbf{P}\mathbf{B}\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} \\ &= -(\mathbf{B}^t\mathbf{P}\mathbf{f})^t \cdot d\mathbf{\Delta} - \mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + (\mathbf{B}^t\mathbf{P}\mathbf{B}\mathbf{\Delta})^t \cdot d\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} \\ &= -\mathbf{f}^t\mathbf{P}^t\mathbf{B} \cdot d\mathbf{\Delta} - \mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}^t\mathbf{B} \cdot d\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} \\ &= -\mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} - \mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + \mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} \\ &= -2\mathbf{f}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta} + 2\mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} \cdot d\mathbf{\Delta}. \end{aligned}$$

Dette gir oss følgende likning til bestemmelse av $\mathbf{\Delta}$:

$$\frac{\partial(\mathbf{v}^t\mathbf{P}\mathbf{v})}{\partial\Delta} = -2\mathbf{f}^t\mathbf{P}\mathbf{B} + 2\mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} = 0.$$

Herav får vi

$$\begin{aligned}
\mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B} &= \mathbf{f}^t\mathbf{P}\mathbf{B} \\
(\mathbf{\Delta}^t\mathbf{B}^t\mathbf{P}\mathbf{B})^t &= (\mathbf{f}^t\mathbf{P}\mathbf{B})^t \\
\mathbf{B}^t\mathbf{P}^t\mathbf{B}\mathbf{\Delta} &= \mathbf{B}^t\mathbf{P}^t\mathbf{f} \\
\mathbf{B}^t\mathbf{P}\mathbf{B}\mathbf{\Delta} &= \mathbf{B}^t\mathbf{P}\mathbf{f}.
\end{aligned}$$

Løsningen for $\mathbf{\Delta}$ blir

$$\boxed{\begin{array}{ll} & \mathbf{\Delta} = \mathbf{N}^{-1}\mathbf{t} \\ hvor: & \\ & \mathbf{N} = \mathbf{B}^t\mathbf{P}\mathbf{B} \\ & \mathbf{t} = \mathbf{B}^t\mathbf{P}\mathbf{f} \end{array}} \tag{2.7}$$

I forbindelse med implementasjon av denne algoritmen kan vi dra nytte av at $\mathbf{P}$ er en tynn matrise.

I noen tilfeller velges et interpolasjonspolynom som består av bare leddet a_{00} i det bikubiske polynomet. Løsningen for $\mathbf{\Delta}$ blir i dette tilfellet meget enkel i det matrisen $\mathbf{B}$ blir en søylevektor med alle ledd lik en; gitt ved

$$\mathbf{B} = \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n1} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

Vi har da at

$$\mathbf{N}^{-1} = \frac{1}{\sum_{i=1}^{n} p_{ii}}$$

og

$$\mathbf{t} = \sum_{i=1}^{n} p_{ii} z_i.$$

Følgelig blir høyden i det interpolerte punktet

$$Z = \frac{\sum_{i=1}^{n} p_{ii} z_i}{\sum_{i=1}^{n} p_{ii}} \tag{2.8}$$

som er middelverdien etter vekt. I dette spesielle tilfellet benevnes gjerne metoden for *flytende middeltall* (eng. moving average).

Kapittel 3

Trekantmodeller

Innen GIS-feltet har trekantmodeller siden slutten av 1980 årene framstått som kanskje den viktigste metoden for å modellere geometrien til terrengoverflaten. Det finnes ulike strategier for å gjøre trianguleringen. Problemet er hvordan formulere trianguleringen slik at vi får effektive beregningsalgoritmer og hvordan tilpasse trianguleringen til det aktuelle terrenget slik at trianguleringen gir en tilfredsstillende beskrivelse av terrengets form med et minimalt antall datapunkter (generaliseringsaspektet). Trianguleringen kan gjøres i 2D, men det er også mulig å utvide problemet til 3D. Dette vil bli vist i et senere kapittel. Når et plan er delt opp i triangler, bruker vi ofte betegnelsen triangulært irregulært nettverk (TIN). Betegnelsen henspeiler på at vi oppfatter trekantsidene som kanter og trekanthjørnene som noder i et nettverk; en graf.

3.1 Datastrukturer for TIN

Et TIN kan representeres på ulike måter. Problemet er å finne en representasjon som ikke er for plasskrevende og som samtidig effektivt understøtter de aktuelle operasjoner på nettverket. En arrayrepresentasjon av grafen er ikke hensiktmessig i vårt tilfelle, på grunn av at det fra en enkelt node i et TIN bare vil være et lite antall kanter som stråler ut. Det er derfor unødvendig å ta høyde for å etablere en en-til-allekobling for nodene. En listestruktur over noder og deres utkanter er en bedre løsning. Ved operasjoner på TIN har vi ofte behov for å identifisere de enkelte trekanter. Det vil derfor være hensiktsmessig å velge en datastruktur der det er enkelt å identifisere en trekant og dens tilordnede sider og hjørner. Vi vil presentere to aktuelle datastrukturer: side-triangel struktur (eng. edge-triangle structure) og tvilling-side struktur (eng. twin-egde structure).

3.1.1 Side-triangel struktur

Vi skal beskrive en datastruktur som er benyttet i IGF-TIN (programpakke utviklet ved institutt for geodesi og fotogrammetri) og baserer oss på Bjørke [Bjø88]. Vi gjør følgende observasjoner:

- En triangelside kan inngå i en eller to triangler (i det første tilfellet ligger triangelsiden på områdets perimeter).
- Antall direkte naboer et triangel kan ha, er null, en, to eller tre. Det første vil inntre dersom det triangulerte området består av bare ett triangel.

Det opereres med to tabeller, E-tabell (edge table) og T-tabell (triangle table). E-tabellen definerer hvilke triangler de enkelte sider inngår i og T-tabellen definerer hvilke sider de enkelte triangler består av som vist i tabell 3.1. Disse relasjonene vil til sammen beskrive sammenhengen i nettverket og gjøre det mulig å navigere raskt i nettverket. Datastrukturen er meget effektiv ved generering av isolinjer. Ulempen med datastrukturen er at den inneholder redundante data.

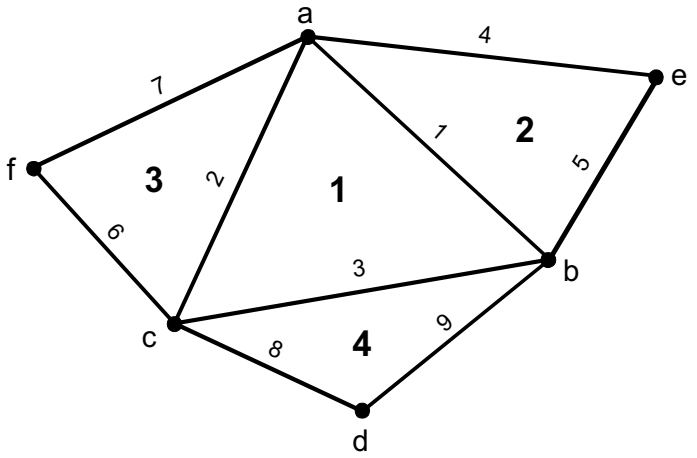
I tabell 3.1 består det triangulerte området av fire triangler. Av T-tabellen finner vi for eksempel at triangel 3 består av sidene 2, 6 og 7. Beskrivelsen av disse sidene finner vi ved å gå inn i E-tabellen under inngangene 2, 6 og 7. Under inngang 3 finner vi for eksempel at side 3 er definert ved punktene c og b og at den inngår i trianglene 1 og 4. Tallverdien -1 markerer at vedkommende nabo ikke eksisterer, d.v.s. triangelsiden ligger på randen til det triangulerte området.

Datastrukturen inneholder redundante data, fordi T-tabellen lar seg direkte utlede fra E-tabellen. E-tabellen er derfor tilstrekkelig for å beskrive nettverket. Begrunnelsen for T-tabellen er at den gjør det raskt å navigere i nettverket; for eksempel ved danning av isolinjer eller ved opptegning av terrengsnitt langs definerte profillinjer. I begge disse tilfellene trenger vi å finne hvilke sider som inngår i en gitt trekant. Dette finner vi ved direkte oppslag i T-tabellen.

3.1.2 Tvilling-side struktur

I en sidebasert datastruktur er all informasjon om trianglene implisitt i datastrukturen. Følgelig er det ikke behov for noen triangel tabell. Heller [Hel90] foreslår ut fra dette den såkalte tvilling-side strukturen. For hver triangelside lagres her en peker til ett av dens endepunkter og en peker til neste side i det aktuelle trianglet. Videre lagres en peker til sidens tvilling, d.v.s. det er den samme siden, men nå med en peker til sidens andre endepunkt og en pekerkjede som lenker sammen sidene i nabotrianglet. Et eksempel på datastrukturen er gitt i tabell 3.2 der vi har betegnet tvillingsidene med suffiksene a og b. Som det framgår av tabellen er det mulig å sløyfe kolonnen med pekere til tvillingsiden, fordi vi har gruppert tvillingsidene slik at de ligger i umiddelbar nærhet i tabellen. Ved å utnytte denne implisitte informasjonen kan vi etablere en enkel adresseberegning for en sides tvilling.

Tabell 3.1: Datastruktur for triangelsider og triangler



E-tabell (sidetabell)

side nr.	1. triangel	2. triangel	fra pkt.	til pkt.
1	1	2	a	b
2	1	3	a	c
3	1	4	c	b
4	2	-1	a	e
5	2	-1	b	e
6	3	-1	c	f
7	3	-1	a	f
8	4	-1	c	d
9	4	-1	b	d

T-tabell (triangel-tabell)

triangel nr.	1. side	2. side	3. side
1	1	2	3
2	1	4	5
3	2	6	7
4	3	8	9

3.2 Krav til trianguleringen

Det finnes flere måter å dele et plan opp i triangler på. I vår anvendelse stilles det følgende krav til denne oppdelingen:

Krav 1 *Triangler skal ikke overlappe hverandre.*

Krav 2 *Trianglene skal gjøres så små som mulig. Dette skal oppnås ved å minimere summen av deres sidelengder (tilnærmet minimum godtas).*

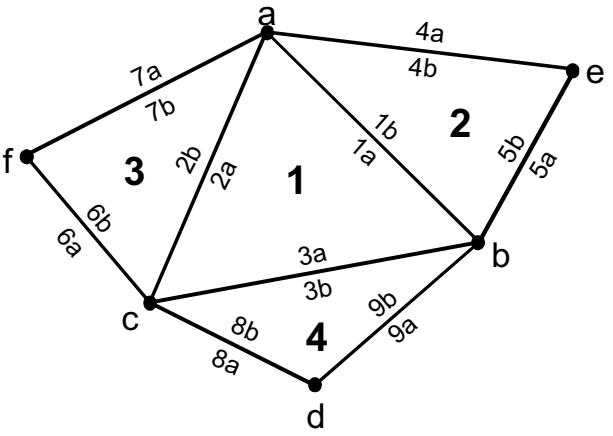
Krav 3 *Triangelsider får ikke lov til å krysse visse definerte linjer, for eksempel markante knekklinjer i terrenget.*

Det første kravet bør være absolutt for å unngå at punkter kan interpoleres fra flere enn et triangel. Det andre kravet kan vi fire noe på da nøyaktigheten til interpolasjonen neppe vil endre seg drastisk om størrelsen på trianglene avviker

Tabell 3.2: Datastruktur basert på tvillingsider

E-tabell (tabell over tvillingsider)

side nr.	neste side i trianglet	tvilling-side	punkt nr.
1a	3a	1b	a
1b	4b	1a	b
2a	1a	2b	c
2b	6b	2a	a
3a	2a	3b	b
3b	9b	3a	c
4a	-1	4b	e
4b	5b	4a	a
5a	-1	5b	b
5b	1b	5a	e
6a	-1	6b	f
6b	7b	6a	c
7a	-1	7b	a
7b	2b	7a	f
8a	-1	8b	c
8b	3b	8a	d
9a	-1	9b	d
9b	8b	9a	b

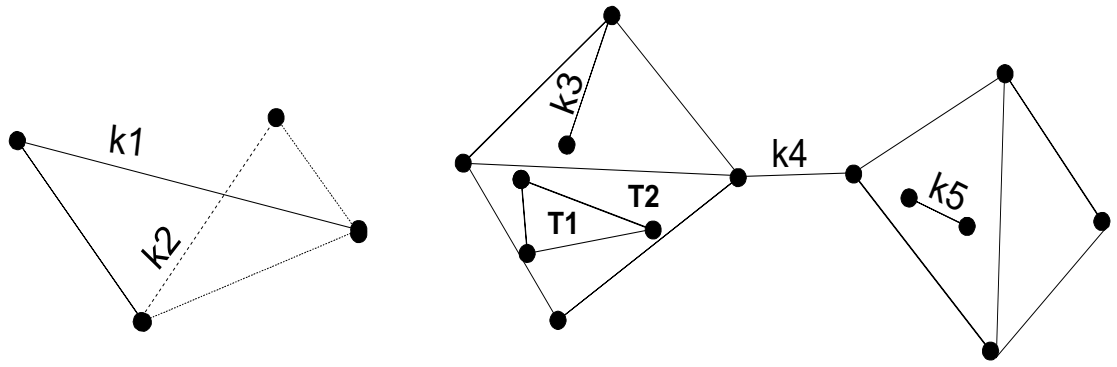


noe fra minimumskravet. Grunnen til at begrepet *trianglenes størrelse* er knyttet til lengden av sidelengdene og ikke til arealet, er at det er avstanden mellom de punkter som ligger til grunn for interpolasjonen, som er avgjørende for nøyaktigheten til de interpolerte terrenghøyder. Et tynt triangel kan ha et lite areal, men det kan likevel ha to lange sider dersom den tredje siden er tilstrekkelig kort. Det tredje kravet er gitt for at terrengmodellen skal kunne representere markante knekklinjer til objektoverflaten.

Et problem med trekantmodeller er å sikre seg mot at det oppstår inkonsistens i nettverket. Plümer og Gröger [PG97] setter opp syv aksiomer som gjelder for såkalte maps. Maps er konstruksjoner som er hentet fra grafteori og danner et matematisk grunnlag for forståelsen av oppdelingen av et plan i ikke-overlappende arealobjekter. Basert på [PG97] kan vi sette opp følgende konsistensregler for et TIN:

Knutepunkt (eng. vertices) :

A1: For hvert knutepunkt i topologibeskrivelsen svarer det bare ett punkt i geometribeskrivelsen (*referanseintegritet*). To forskjellige knutepunkt har ikke de samme koordinater (*ulikhet*).



Figur 3.1: Eksempler på trianguleringer som strider mot konsistensreglene

A2: Hvert knutepunkt har minst to kanter (*grad* $\geq$ *2).*

Kanter (eng. edges) :

A3: For hver kant finnes det bare to knutepunkt som endepunkter.

A4: Kanter er rette linjesegmenter. To kanter har ikke felles punkter unntatt i deres endepunkter (*ikke overlappende kanter*).

A5: Hver kant inngår i to flater (to trekanter eller en trekant og den ytre til trianguleringen).

Flater (eng. faces) :

A6: Hver trekant har en enkelt sykel som sitt omriss.

A7: Ingen midtpunkt på en kant tilhører den indre til en trekant.

Aksiom A1 byr på et problem dersom terrenget har vertikale vegger. Problemet stikker i at vi har forutsatt at modellen skal være 2.5D. For å omgå konsistensproblemet som skyldes vertikale vegger, kan vi innføre en liten horisontal forskyvning av noen av datapunktene. Dersom tre punkter som skal danne en trekant er kollineære, vil vi få en såkalt nulltrekant. Nulltrekanter strider mot aksiom A4. Aksiom A7 hindrer at vi får triangler som ligger innenfor et annet triangel. I figur 3.1 strider kantene $k1$ og $k2$ mot aksiom A4, kant k3 strider mot aksiom A2, kant k4 strider mot aksiom A5, kant k5 strider mot aksiom A2 og triangel T1 strider mot aksiom A7.

Et spørsmål er hvor mange av de angitte konsistensregler vi skal innføre for en triangulering. Det kan for eksempel i noen tilfeller være bekvemt å tillate nulltrekanter selv om disse strider mot aksiom A4. Spørsmålet om konsistensregler dreier seg derfor om hvilke regler som er nødvendige og hvilke regler som er ønskelige. Vi vil ikke gå nærmere inn i dette problemområdet, men viser til blant annet Cockcroft [Coc97] som gir en generell oversikt over kvalitet til geografisk informasjon og romlige integritets regler.

3.3 Metoder for å konstruere et TIN

3.3.1 Triangelsider mellom alle punkt

I den tidligste litteratur om trekantmodeller, ble det foreslått algoritmer som innebar at man måtte undersøke alle de triangelsider det er mulig å danne på grunnlag av en gitt punktsverm. Dersom vi har n datapunkter, vil vi kunne danne $(n-1)$ triangelsider med utgangspunkt i ett og samme punkt. Algoritmen ender derfor opp med en tidskompleksitet $O(n^2)$. Ett forslag gikk ut på at tidsforbruket kunne holdes under kontroll ved å dele området i m segmenter. Tidsforbruket ville da bli $O(m(n/m)^2)$ for å danne m segmenter. Dette gir likevel en kvadratisk vekstrate, bare med den forskjell at eksplosjonen i tidsforbruket kommer ved en noe høyere verdi av n.

Et problem en slik segmentering etterlater seg, er at det må etableres triangler mellom segmentene. Yoëli [Yoë77] foreslår en måte denne sammensyingen kan gjøre på. Tidskompleksiteten vil med segmentering være avhengig både av hvor effektivt de enkelte segment lar seg danne og hvor effektivt segmentene kan bli hektet sammen.

3.3.2 Delaunay-triangulering

De metoder for konstruksjon av TIN som i dag benyttes, baserer seg på den såkalte *Delaunay*-triangulering. Metoden har fått sitt navn etter B. Delaunay som beskrev den i en artikkel i 1934 [Del34]. Det er først på slutten av nittensyttiårene at metoden begynner å få en viss utbredelse innen digitale terrengmodeller. En slektning til Delaunay-triangler, er de såkalte *Thiessen*-polygoner. Thiessen-polygoner ble først presentert i en artikkel av A.H. Thiessen i 1911 [TC11]. Metoden ble seks år senere forbedret av R.E. Horton.

En Delaunay-triangulering kan etableres etter ulike prinsipper. Midtbø [Mid93] nevner følgende metoder:

Statisk triangulering (eng. static triangulation). Ved statisk triangulering gjøres ikke endringer av en trekant etter at den er dannet. Metoden er lite fleksibel, fordi dersom ny punkter legges til datasettet, må trianguleringen starte fra begynnelsen igjen. Metoden har flere varianter som

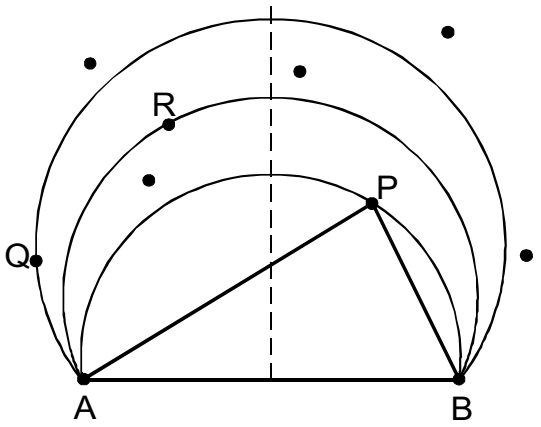
1. Radiell streife-algoritme (eng. radial sweep algorithm)
2. Rekursiv splitt-algoritme (eng. recursive split algorithm)
3. Del-og-hersk algoritme (eng. divide-and-conquer algorithm)
4. Trinn-for-trinn algoritme (eng. step-by-step algortihm)
5. Modifisert hierarkisk algoritme (eng. modified hierarchical algorithm)

Dynamisk triangulering (eng. dynamic algorithm). Dynamisk triangulering tar en rekursiv oppdeling av planet i trekanter. Den starter med noen store trekanter som dekker hele området. Disse deles så opp i mindre trekanter etter litt ulike prinsipper.

1. Inkrementell algoritme (eng. incremental algorithm)
2. Inkrementell slett-og-bygg algortime (eng. incremental delete-and-build algorithm)

Vi skal ikke beskrive alle disse metodene, men velger et par av de mest aktuelle algoritmene. Et generelt problem ved Delaunay-triangulering og operasjoner på TIN er at det kan oppstå spesialtilfeller som må takles omhyggelig for å unngå tallmessige problem; på grunn av tvetydighet og nulltrekanter. Vi vil i det etterfølgende i noen grad berøre disse problemene.

3.4 Statisk Delaunay-triangulering



Figur 3.2: Prinsippet for Delaunay-triangulering

Vi vil beskrive en statisk trianguleringsalgoritme slik den kommer til uttrykk i den terrengmodellen som er utviklet ved instititt for geodesi og fotogrammetri, NTH. Programsystemet er gitt betegnelsen IGF-TIN. Metoden er en trinn-for-trinn-algoritme.

1. Velg et vilkårlig datapunkt A og finn det punkt B som ligger nærmest A. Linjen AB vil vi benevne en *basislinje*. Med basislinje mener vi her en linje vi ønsker å danne et triangel med.

2. Finn et tredje punkt som sammen med basislinjen danner et triangel slik at betingelsene i krav 1-3 blir oppfylt. Dersom et tredje punkt ikke lar seg finne, ligger basislinjen på omrisset til det triangulerte området.

3. Dersom et tredje punkt er funnet, etableres det tilhørende trianglet. Trianglets sider legges i en *kø* (først-inn, først-ut liste), unntatt de sidene som allerede måtte befinne seg i køen. Basislinjen vil selvsagt allerede ligge i køen, men også en av de andre sidene kan befinne seg i køen.

4. Hent en triangelside fra køen, benytt denne som basislinje og gjenta punkt 2. Trianguleringen avsluttes når køen er tom.

For å kunne oppfylle kravet om at ingen triangelsider skal krysse hverandre, må det tredje punktet danne et triangel slik at ingen datapunkter ligger innenfor trianglet.

Anta at vi har et sett av triangler som består av alle de triangler som kan dannes på grunnlag av det aktuelle settet av datapunkter og basislinjen AB. Hver av disse trianglene definerer en sirkel. Vi plukker så ut den sirkel som har den minste *pilhøyden* over basislinjen. Denne sirkelen definerer et triangel som ikke har noen datapunkter innenfor seg. Dette er lett å innse ved en figurbetraktning. Med pilhøyden mener vi avstanden fra basislinjens midtpunkt til skjæringspunktet mellom det aktuelle sirkelsegmentet og basislinjens midtnormal. I figur 3.2 er punkt P det punktet som sammen med basislinjen definerer den sirkel som har den minste pilhøyden over linjen AB. Punkt Q vil for eksempel gi en sirkel med større pilhøyde over AB enn hva punkt P vil gjøre. Ved danning av de såkalte Thiessen-polygoner benyttes en prosedyre som ligner Delaunay-triangulering. Forskjellen er at ved Thiessen-polygonering velges det punktet som sammen med basislinjen definerer en sirkel som har den minste radius.

En attraktiv side ved sirkelkriteriet, er at trianglene blir tilnærmet så små som mulig og samtidig tilnærmet mest mulig likesidede. Det tredje kravet vi stilte til trianguleringen, oppfylles ikke uten videre ved en Delaunay-triangulering. Vi skal senere komme tilbake til hvordan dette kravet lar seg implementere.

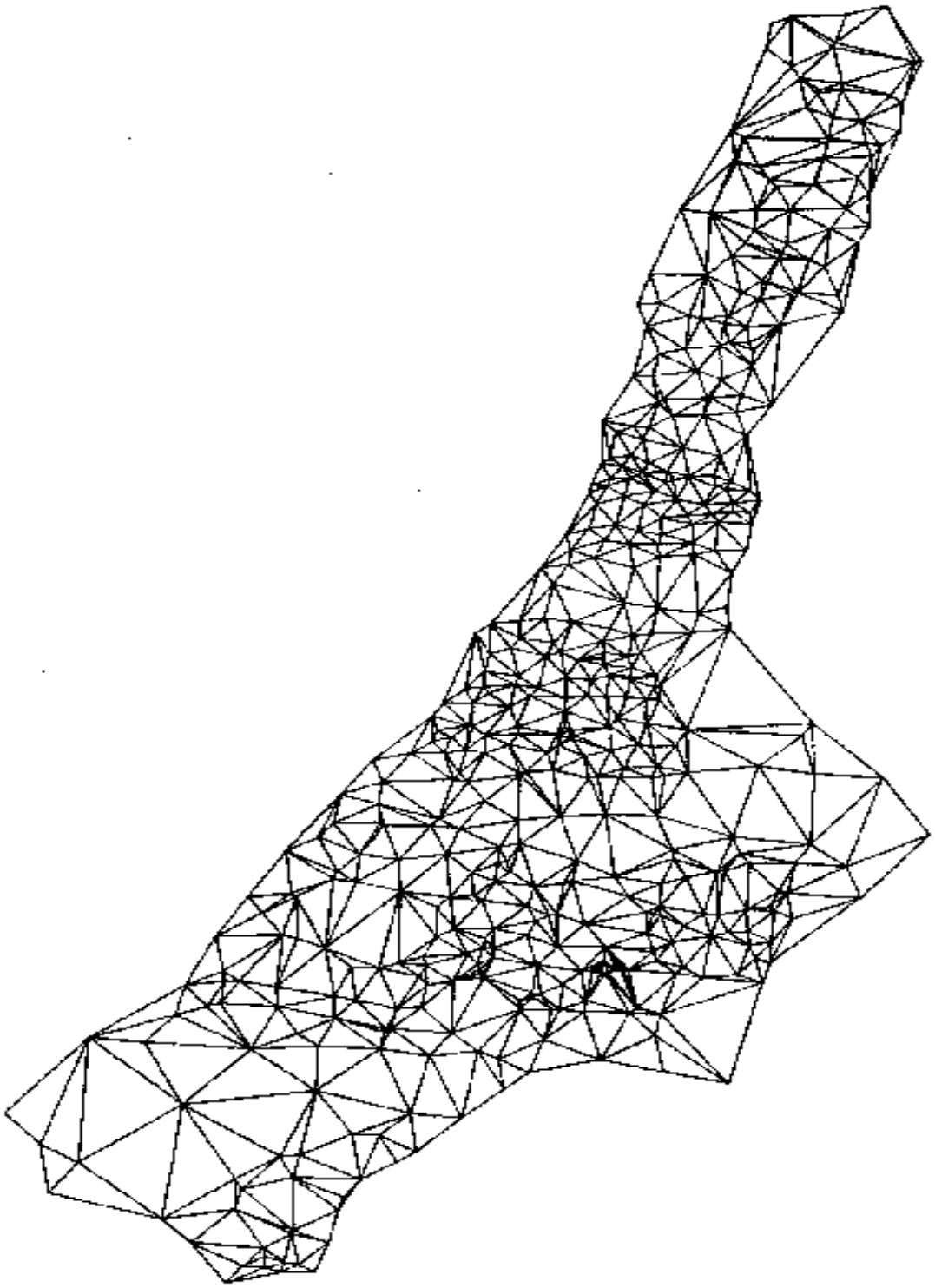
Beregne sirkelsentrum

Gitt en basislinje med endepunktene A og B og et tredje punkt C. Delaunaytrianguleringen krever at vi må bestemme den sirkel som går gjennom disse tre punktene. Beregningen blir enkel om vi tar en transformasjon til et lokalt koordinatsystem. Vi legger Y-aksen langs basislinjen AB med origo i basislinjens halveringspunkt. X-aksen orienteres slik at punkt C får positiv x-koordinat; se figur fig 3.4. Vi legger en sirkel gjennom de tre punktene A, B, C og får følgende likninger til bestemmelse av sirkelens sentrum Q:

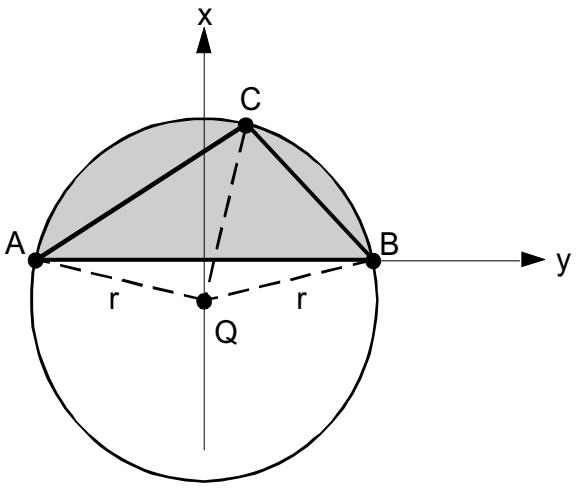
$$(x_C - x_Q)^2 + (y_C - y_Q)^2 = r^2, \tag{3.1}$$
$$(x_B - x_Q)^2 + (y_B - y_Q)^2 = r^2. \tag{3.2}$$

På grunn av den lokale transformasjonen er

$$y_Q = x_A = x_B = 0 \quad og \quad y_A = -y_B.$$



Figur 3.3: Delaunay-triangulering



Figur 3.4: Beregning av sirkelsentrum

Dette innført i likningene 3.1 og 3.2 gir følgende løsning for Q:

$$x_Q = \frac{x_C^2 + y_C^2 - y_B^2}{2x_C} \qquad forutsatt \quad x_C \neq 0, \tag{3.3}$$

$$y_Q = 0. \tag{3.4}$$

Siden vi regner x_Q med fortegn, vil den sirkel som har den minste verdi for x_Q, også ha den minste pilhøyden i forhold til basislinjen. Delaunay-trianguleringen kan derfor baseres på testing direkte mot x_Q.

Teorem 1 *Gitt en sirkel definert ved tre punkter A, B, C. Ethvert punkt C' innenfor det skraverte sirkelsegmentet vist i figur 3.4, vil sammen med A og B definere en sirkel slik at $x_{Q'} < x_Q$.*

Bevis: Ethvert punkt innenfor det aktuelle sirkelsegmentet må oppfylle betingelsen $x > 0$. Vi velger et vilkårlig punkt (x, y) på sirkelen som defineres av punktene A, B, C og flytter dette punktet et stykke ϵ langs Y-aksen til punktet $(x, y - \epsilon)$. Beviset består i å finne hvilke verdier for ϵ som gir

$$\frac{x^2 + (y-\epsilon)^2 - y_B^2}{2x} < \frac{x^2 + y^2 - y_B^2}{2x},$$

som ved litt omforming går over til

$$(y-\epsilon)^2 < y^2 \quad \text{for } x > 0;$$

som kan skrives som

$$-y < y - \epsilon < y \quad \text{for } x > 0.$$

Vårt valg av XY-system innebærer at sirkelen er symmetrisk om Y-aksen. Siden (x, y) er tilfeldig valgt på sirkelens omriss med den begrensning at $x > 0$, innebærer det angitte intervallet for $(y - \epsilon)$ at ethvert punkt C' innenfor det aktuelle sirkelsegmentet vil definere en sirkel slik at $x_{Q'} < x_Q$. □

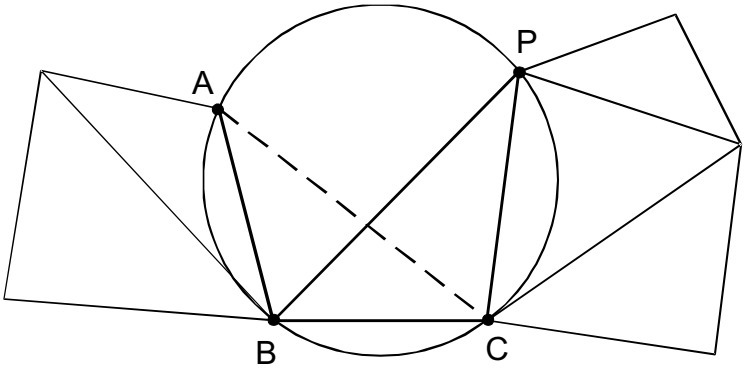
Ved teorem 1 har vi vist at Delaunaykriteriet i hovedsak garanterer at vi ikke får kryssende eller overlappende triangler, men det kan likevel oppstå situasjoner som krever forsiktighet. Som det framgår av likning 3.3, er ikke sirkelen entydig definert dersom det tredje punktet C ligger på basislinjen. På grunn av den måten Delaunay-trianguleringen velger det tredje punktet i et triangel, vil denne situasjonen normalt ikke opptre. Det blir derfor kun i helt spesielle situasjoner dette kan skje (dersom punkt C ligger like til høyre for punkt B). For å unngå numeriske problem kan vi legge inn et testkriterium som sier at sirkelen ikke forsøkes beregnet dersom

$$| x_C | < \delta,$$

hvor δ gis en liten positiv verdi.

Flere kandidater på samme sirkel

Dersom flere kandidater til tredje punkt i et triangel ligger på den samme sirkelen, må det innføres et tilleggskriterium. McCullagh [MR80] foreslår at det punktet som ligger nærmest et av basislinjens endepunkter, blir valgt. Denne framgangsmåten garanterer ikke mot kryssende triangler som vist i figur 3.5. I figuren ligger de fire punktene A, B, C og P på den samme sirkelen. Dersom vi starter med basislinjen



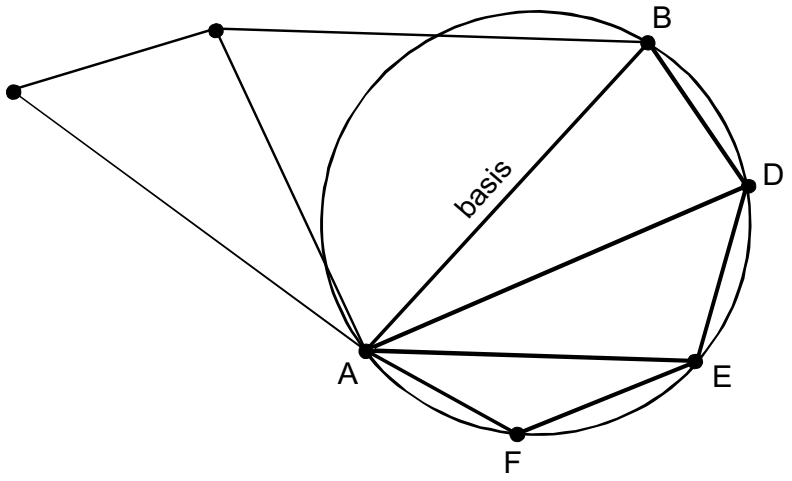
Figur 3.5: Flere kandidater ligger på den samme sirkelen. Dette innebærer risiko for kryssende triangler.

BC, har vi valget mellom punktene A og P. Anta at vi velger P, fordi avstanden CP er kortere enn avstanden CA. Deretter innføres AB som basislinje. Vi får da å velge mellom punktene P og C. Siden avstanden BC er kortere en avstanden BP, velges C. Problemet med dette valget er at triangel ABC overlapper triangel BCP. Som vi ser, er det ikke likegyldig hvordan vi formulerer tilleggskriteriet.

En metode som kan anbefales for vårt spesialtilfelle, baserer seg på et forslag av McLain [McL76]. Metoden kan beskrives slik:

1. Kandidater som ligger på den samme sirkelen, sorteres ifølge figur 3.6 i rekkefølge $D, E, F \cdots$ på grunnlag av cos til vinklene $DAB, EAB, \cdots$.

2. Adder trianglene $ABD, ADE, AEF, \cdots$ til listen.



Figur 3.6: Flere kandidater ligger på den samme sirkelen. Sorter kandidatene og adder de respektive triangler til listen.

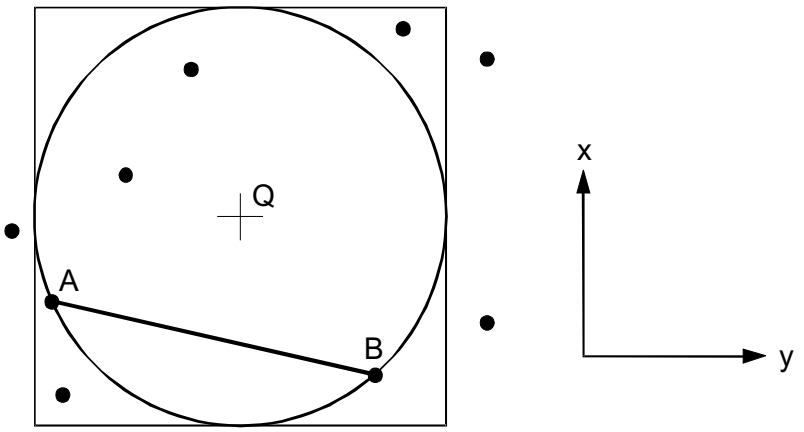
McLain's metode er illustrert i figur 3.6. Her ligger punktene A, B, D, E og F på den samme sirkelen. Fra basislinjen AB har vi derfor valget mellom tre punkter. I stedet for å velge ett av punktene, danner vi triangler med alle tre punktene. Dette vil hindre at vi får dannet kryssende triangler. For å avgjøre hvilke punkter som skal inngå i hvilke triangler, baserer vi oss på punktenes vinkelavstand fra basislinjen. Denne lar seg i vårt tilfelle enklest uttrykke ved cos til de aktuelle vinkler. Det er likegyldig hvilket av basislinjens endepunkter vi velger når cos skal beregnes. Hadde vi valgt punkt B, ville vi dannet trianglene ABF, FBE og EBD.

Strategi for å søke opp kandidater

Trianguleringsalgoritmen må benytte en fornuftig strategi for å finne den kandidat som oppfyller Delaunay-kriteriet. I IGF-TIN har vi valgt å la et sirkulært søkevindu gradvis øke i størrelse inntil en kandidat er funnet, begrenset av at vinduet må holde seg innenfor en gitt maksimalstørrelse. Algoritmen blir:

```
while (R ≤ Rmax and kandidat ikke funnet) do
begin
    :
    R := R*2; {øker søkevinduets størrelse}
    :
end;
```

Vanligvis er det raskere å søke med et rektangulært vindu som orienteres parallelt med koordinataksene, enn med et sirkulært vindu. Av denne grunn testes først



Figur 3.7: Søkevindu.

mot et omskrivende rektangel; som vist i figur 3.7. Kandidater som ligger utenfor rektanglet, vil også ligge utenfor sirkelen. Testen "innenfor sirkelbegrenses derfor til bare de punkter som ligger innenfor rektanglet. Testen "innenfor rektangulært vinduer gitt ved:

if (x > Xmax or x < Xmin or y > Ymax or y < Ymin) then utenfor:=true.

Søkevinduets maksimale størrelse er avhengig av hvor store triangler en maksimalt er villig til å akseptere. Denne størrelsen bør derfor kunne gis inn som en parameter til programmet.

Datastruktur

Vi vil benytte en side triangelstruktur og etablerer to tabeller; E-tabell og T-tabell. E-tabellen benyttes som en *kø* under etableringen av trianglene. Etter at trianglene er dannet, inngår denne tabellen som en del av datastrukturen. En triangelside er ferdigbehandlet når både Eft (sidens 1. triangel, first triangle) og Est (sidens 2. triangel, second triangle) har fått tilordnet et triangel eller eventuelt merket pp-å randentil det triangulerte område. Nye triangelsider legges til i slutten av køen og deres Est merkes "ikke behandlet". Siden neste basislinje hentes fra starten av køen, vil den øverste delen av E-tabellen etter hvert inneholde ferdig behandlede sider, og den kan om nødvendig legges ut på fil. E-tabellen kan i så fall fungere som en *sirkulær* kø. Vi skal med et eksempel vise hvordan E-tabellen kan bygges opp. Eksemplet er knyttet til figuren vist i tabell 3.1. Utviklingen av datastrukturen under oppretting av triangelstrukturen er vist i tabellene 3.3, 3.4 og 3.5.

I det foreliggende eksemplet har vi på grunnlag av et tilfeldig valg plukket ut punkt a som startpunkt. Deretter finnes nærmeste punkt til a, som i dette tilfellet er punkt b. Disse to punktene innføres så i E-tabellen. Tegnet ? er i tabellen benyttet for å markere "ikke behandlet". Etter hvert som E-tabellen dannes, ser vi at den øvre

Initierer tabellen

Entry	Eft	Est	Efrom	Eto
1	?	?	a	b

Basislinje ab

Entry	Eft	Est	Efrom	Eto
1	1	?	a	b
2	1	?	a	c
3	1	?	c	b

Tabell 3.3: E-tabellen etter at den er initiert og trianglet til venstre for basislinje ab er dannet.

Basislinje ab

Entry	Eft	Est	Efrom	Eto
1	1	2	a	b
2	1	?	a	c
3	1	?	c	b
4	2	?	a	e
5	2	?	b	e

Basislinje ac

Entry	Eft	Est	Efrom	Eto
1	1	2	a	b
2	1	3	a	c
3	1	?	c	b
4	2	?	a	e
5	2	?	b	e
6	3	?	c	f
7	3	?	a	f

Tabell 3.4: E-tabellen etter at trianglet til høyre for basislinje ab og trianglet til høyre for basislinje ac er dannet.

Basislinje cb

Entry	Eft	Est	Efrom	Eto
1	1	2	a	b
2	1	3	a	c
3	1	4	c	b
4	2	?	a	e
5	2	?	b	e
6	3	?	c	f
7	3	?	a	f
8	4	?	c	d
9	4	?	b	d

Basislinje ae

Entry	Eft	Est	Efrom	Eto
1	1	2	a	b
2	1	3	a	c
3	1	4	c	b
4	2	-1	a	e
5	2	?	b	e
6	3	?	c	f
7	3	?	a	f
8	4	?	c	d
9	4	?	b	d

Tabell 3.5: E-tabellen etter at trianglet til høyre for basislinje cb og trianglet til høyre for basislinje ae er dannet.

delen av tabellen vil bestå av ferdig behandlede sider. På grunn av at denne delen av tabellen ikke behøves for den videre trekantdanningen, kan de ferdig behandlede sidene legges ut på fil. Den plassen som på denne måten frigjøres, gjør det mulig å benytte E-tabellen som en sirkulær kø.

Bare nye sider skal innføres i køen, derfor må det for hvert triangel som dannes, undersøkes om noen av triangelsidene inngår i et triangel som allerede befinner seg køen. Dette er en test som kan synes svært tidskrevende, men ved å innføre en statusvariabel for hvert datapunkt reduseres tidsforbruket betraktelig. Initielt merkes alle datapunktene "ikke-benyttet"og etter hvert som trianguleringen skrider fram merkes datapunktene benyttet". Dersom et tredje punkt er merket "ikke benyttet", vet en at de to tilhørende triangelsidene ikke finnes i E-tabellen fra før. I motsatt fall må den delen av E-tabellen som ligger i køen, undersøkes for å finne ut hvilke sider som allerede er etablert. Se algoritmen i figur 3.8.

```
if punkt benyttet tidligere then
begin
    for i:= startkø to sluttkø do
    begin
        {undersøk om de to nye triangelsider finnes i køen fra før}
    end;
else {punktet er ikke benyttet tidligere}
end;
```

Figur 3.8: Før en triangelside legges inne i E-tabellen, må det undersøkes om den allerede befinner seg der.

Innsetting av knekklinjer

Krav 3 sier at triangelsider ikke får lov til å krysse visse definerte linjer; knekklinjer. I programmet IGF-TIN takles krav 3 på følgende måte:

1. Først dannes trekantnettet uten at det tas omsyn til knekklinjene.

2. Deretter søkes etter mulige skjæringspunkt mellom knekklinjene og triangelsidene. Dersom skjæringspunkter finnes, dannes triangelnettet på nytt, men nå med en ny punktmengde $\mathbf{D}_1$ som består av alle opprinnelige datapunkter $\mathbf{D}_0$ i tillegg til alle skjæringspunkt $\mathbf{S}$; gitt ved $\mathbf{D}_1 = \mathbf{D}_0 \cup \mathbf{S}$.

Søket etter skjæringspunkt gjøres ved å spore langs knekklinjene på jakt etter mulige krysningspunkt. Sporingen blir effektiv i den datastruktur vi har valgt, fordi vi lett kan finne det neste triangel en linje går inn i når det triangel linjen kommer fra er kjent. Et springende punkt er hvor mange iterasjoner det er nødvendig å gjøre. Empiriske undersøkelser har vist at det normalt er nok med bare en iterasjon, det

vil si i alt to kjøringer, nemlig den første kjøringen med punktmengden $\mathbf{D}_0$ og den neste kjøringen med punktmengden $\mathbf{D}_1$.

Triangulere innenfor en polygon med vilkårlig form

Problemet i dette tilfellet er at triangelnettet ikke får lov til å utvikle seg fritt ved at det innføres en tilleggsbetingelse om at trianguleringen skal tilpasses en polygon av vilkårlig form. Den måten dette gjøres på i IGF-TIN er som følger:

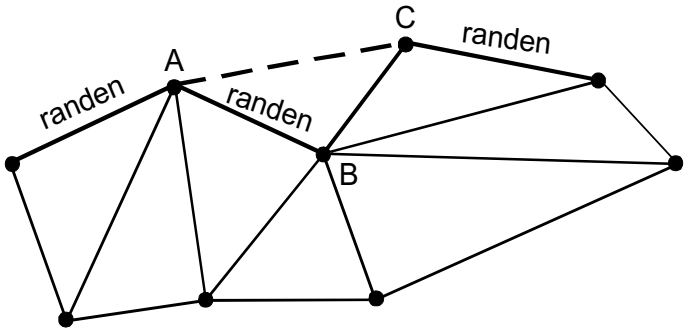
1. Trianguler på vanlig måte inntil det triangulerte området dekker hele polygonen.

2. Klipp ut de datapunkter som ligger innenfor polygonen. Dette gir oss punktmengden $\mathbf{D}_1$.

3. Finn skjæringspunktene mellom polygonen og det triangulerte området. Adder disse punktene til punktmengden $\mathbf{D}_1$. Dette gir oss en ny punktmengde $\mathbf{D}_2$.

4. Gjør en ny triangulering, men nå med $\mathbf{D}_2$. Dette vil gi en triangulering som passer innenfor den angitte polygonen. Det er nødvendig å teste på om triangler ligger utenfor områdepolygonen. For eksempel vil Delaunayprinsippet føre til at punkt C i figur 3.9 blir valgt om linjen AB benyttes som basislinje, men trianglet ABC ligger utenfor området. Den aktuelle basislinjen merkes derfor på randen". Utenfor testen kan baseres på et vilkårlig punkt innenfor trianglet.

5. Kontroller fullstendigheten av trianguleringen ved å sammenligne arealet regnet av polygonen og arealet summert over alle trekanter. Et signifikant avvik skal ikke oppstå dersom den avgrensende polygonen definerer et sammenhengende område av triangler.



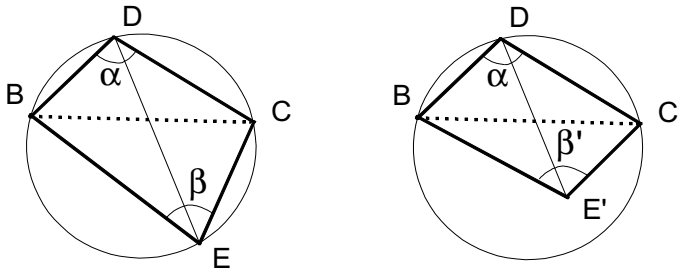
Figur 3.9: Triangler på randen krever en "utenfortest.

3.5 Dynamisk Delaunay-triangulering

Ved dynamisk triangulering vil vi formulere Delaunaykriteriet på en alternativ måte i forhold til hva vi gjorde under teorem 1 ved den statiske trianguleringen.

3.5.1 Delaunaykriteriet ved dynamisk triangulering

Definisjon 1 *Et Delaunaynettverk i to dimensjoner består av ikke-overlappende trekanter hvor ingen punkter i nettverket ligger innenfor den omskrivende sirkel til noen av trekantene.*



Figur 3.10: Maksimum vinkelsumtesten

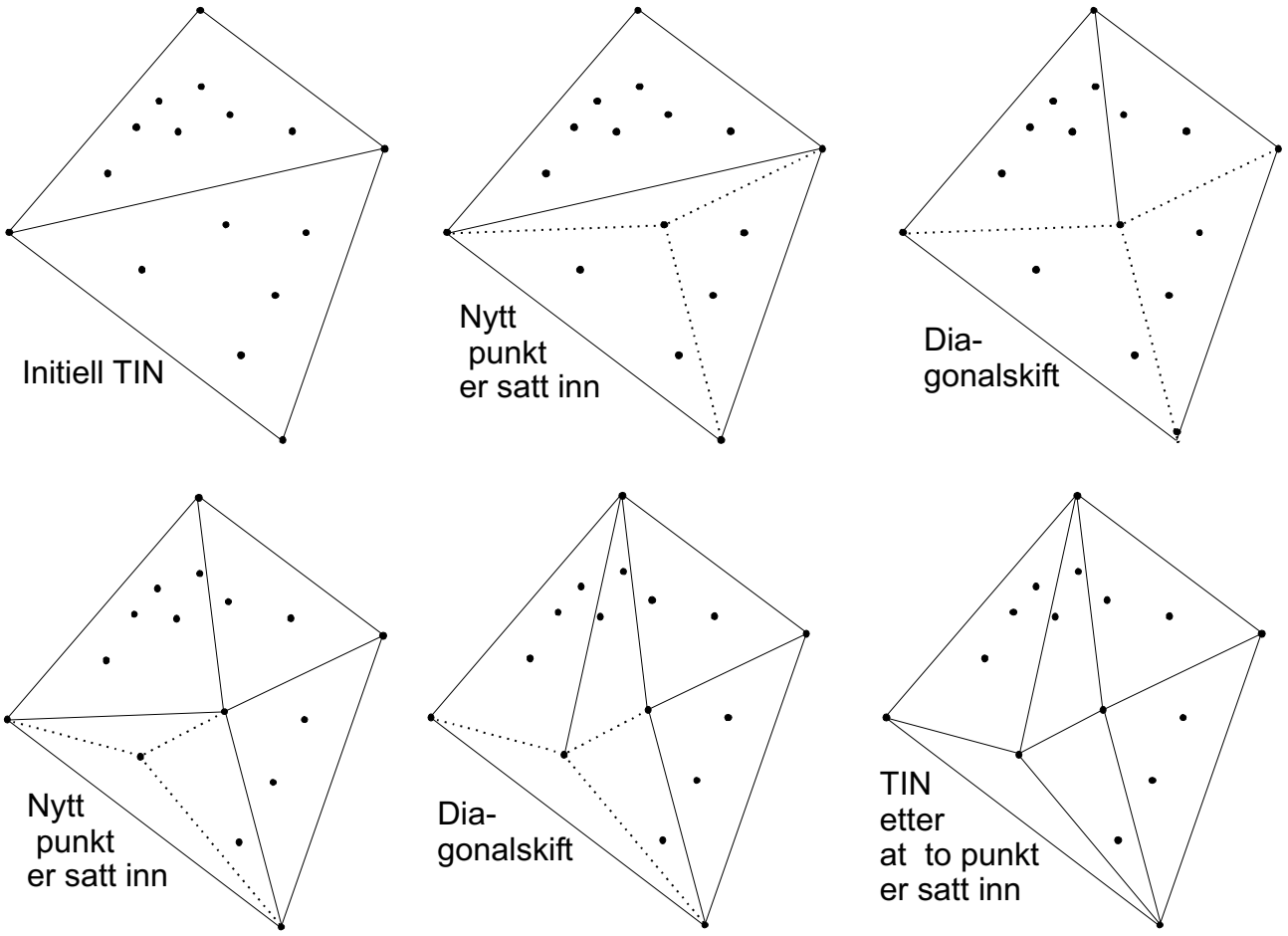
Teorem 2 *Anta en firkant som er dannet av to nabotrekanter i en Delaunaytriangulering. Da gjelder at diagonalen går mellom de to motstående hjørner i firkanten hvor summen av de innvendige vikler er større enn eller lik π.*

Bevis: Beviset relateres til en figurbetraktning. I figur 3.10 har vi gitt firkanten $\square BDCE$, diagonalen DE og sirkelen $\bigcirc BDC$. La oss anta at punkt E er plassert på $\bigcirc BDC$ og at E flyttes til et punkt E' innenfor $\bigcirc BDC$. Ifølge definisjon 1 er ikke Delaunaykriteriet oppfylt dersom E' ligger innenfor den omskrivende sirkel til trekant $\triangle BDC$; derfor kan ikke diagonalen BC gi en gyldig triangulering for $\square BDCE'$. I $\square BDCE$ er $\alpha + \beta = \pi$, siden de to vinklene spenner over buene $\frown BC$ og $\frown CB$. Dersom vi flytter punkt E til punktet E', vil vi ha at $\beta < \beta'$, derfor blir $\alpha + \beta' > \pi$. I motsatt fall dersom E' ligger utenfor $\bigcirc BDC$, vil vi ha at $\beta > \beta'$ og $\alpha + \beta' < \pi$. Delaunaykriteriet er derfor oppfylt for den diagonal i en firkant som går mellom de to hjørner hvis innvendige vinkler har en sum større enn eller lik π. Siden summen av vinklene i en firkant er 2π, vil ikke begge diagonalene gi gyldige trianguleringer, med unntak av det spesielle tilfellet at vinkelsummen er π. $\square$

Av beviset for teorem 2 følger at trianguleringen av en firkant ikke er unik dersom de motstående vinkler i firkanten har en vinkelsum lik π. Dette spesialtilfellet vil neppe by på problemer for den praktiske anvendelse av teoremet.

De innvendige vinkler i en firkant kan finnes ved innerproduktet til de aktuelle sider. Herav lar cos til vinklene seg utlede. Det lar seg vise at den diagonal som gir $\cos\alpha + \cos\beta > 0$, må byttes [Mid93], side 25.

3.5.2 Algoritme for dynamisk triangulering



Figur 3.11: Illustrasjon av dynamisk trianguleringsalgortime

Algoritmen initieres ved at vi på en eller annen måte skaffer oss noen få store Delaunaytrekanter som tilsammen dekker alle datapunktene; dvs. at alle datapunkter er tilordnet hver sin trekant. En måte å skaffe seg den initielle trianguleringen på, er å innføre fire fiktive punkter som utspenner et omskrivende rektangel til datapunktene. En annen strategi er å etablere et omskrivende polygon med basis i datapunktene. La oss anta at vi har skaffet oss en initiell triangulering der hver trekant er tilordnet sine datapunkter; altså de punktene som ligger innenfor den respektive trekanten. Dette gir oss følgende algoritme:

1. La C og T være mengder av trekanter. Initier $C = \emptyset$ og initier T med alle trekanter i den initielle trianguleringen. La L_i være mengden av datapunkter som er tilordnet trekant t_i og la L være mengden av alle L_i. Initier L_i for alle trekanter i T.

2. Velg et tilfeldig datapunkt k fra L og del punktets omskrivende trekant t_j opp i tre nye trekanter t_1, t_2 og t_3 med utgangspunkt i k og de tre hjørnene til t_j. Legg så t_1, t_2 og t_3 til C. Splitt deretter L_i i tre delmengder som blir å tilordne sin respektive trekant. Oppdater T med t_1, t_2 og t_3 og fjern t_j fra T.

3. Velg vilkårlig trekant c_i fra C og fjern c_i fra C. Benytt Delaunaykriteriet og evaluer diagonalene til alle firkanter som kan dannes med utgangspunkt i c_i og nabotrekantene til c_i. Dersom en diagonal i en firkant må byttes, legges de to nye trekantene til C og T og L oppdateres. Evalueringen av c_i terminerer nå alle naboene til c_i er evaluert, eller når det gjøres et diagonalbytte.

4. Gjenta fra trinn 3. til alle trekanter i C er evaluert.

5. Gjenta fra trinn 2. til alle datapunkter i L er inkludert i trianguleringen.

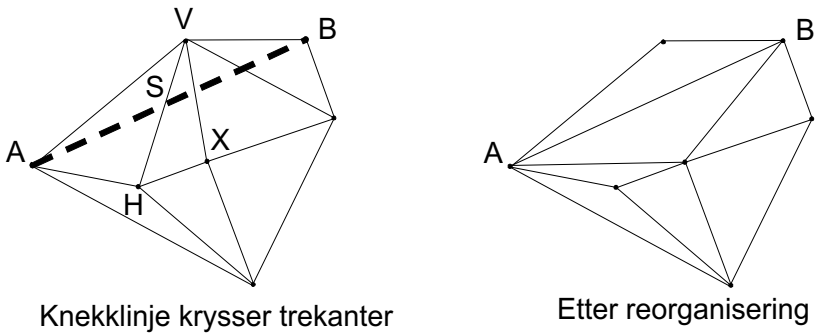
Trinn 3. i algoritmen ovenfor er noe kritisk, fordi ombyttingen av diagonalene vil forplante seg til nabotrekanter. Det lar seg imidlertid vise at denne forplantningen har svært begrenset utbredelse. I en undersøkelse av [Mid93] ble det funnet at innsetting av et datapunkt resulterte i underkant av 10 diagonalskift pr. datapunkt. [Mid93] viser at det området som blir berørt av diagonalombytting, ikke vil overstige størrelsen på det området som defineres av de omskrivende sirkler til de tre nye trekanter som dannes når den opprinnelige trekanten blir splittet. Figur 3.11 illustrerer den dynamiske trianguleringsalgoritmen.

3.5.3 Kvalifisert innsetting av punkt

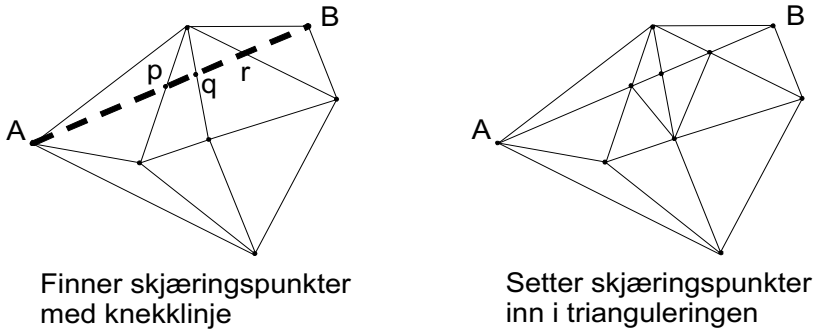
En fordel med den dynamiske trianguleringen er at vi enkelt kan implementere en generalisering av terrengmodellen. Før et punkt inkluderes i trekantstrukturen, kan vi evaluere punktets bidrag til beskrivelsen av modellen; vi foretar det vi kan kalle en kvalifisert innsetting av punkt. Evalueringen kan basere seg på avstanden mellom punktet og terrengplanet til den trekanten punktet skal inkluderes i. Skrankeverdien for å akseptere et punkt kan modelleres som en funksjon av for eksempel terrenghøyden eller avstanden til et eventuelt projeksjonssenter.

3.5.4 Innsetting av knekklinjer

Når det foreligger knekklinjer, må trianguleringen tilpasse seg disse slik at ingen trekanter krysser knekklinjene. Det finnes flere måter å innføre denne betingelsen på. I figur 3.12 er vist en metode som er foreslått av [Hel90]. En svakhet ved denne metoden er at Delaunaykriteriet ikke blir oppfylt overalt i trianguleringen, og at det kan bli dannet lange trekanter som kan gi uheldige utslag i interpolasjonen av høydeverdier. En annen måte er å beregne skjæringspunktene mellom knekklinjen og trianguleringen, som vist i figur 3.13, og foreta en lokal retriangulering ved å



Figur 3.12: Innsetting av en knekklinje ved reorganisering av trianguleringen langs knekklinja.



Figur 3.13: Innsetting av en knekklinje ved at krysningspunktene settes inn i trianguleringen.

innføre skjæringspunktene i trianguleringen. Høydeverdiene til skjæringspunktene finnes ved å interpolere langs knekklinjen.

En tredje metode er å innføre knekklinjene i den initielle trianguleringen. Knekklinjene vil da ligge fast under hele trianguleringen og trekantene vil etter hvert tilpasse seg knekklinjene. Dersom det er langt mellom punktene langs knekklinjene, kan vi interpolere fram ekstra punkter. Disse punktene vil dele knekklinjene opp i delsegmenter som blir å innføre i trianguleringen i stedet for de opprinnelige knekklinjene.

3.5.5 Sletting av punkter i en Delaunaytriangulering

I en terrengmodell har vi følgende grunnleggende operasjoner på nettverket: (1) innsetting av punkt, (2) sletting av punkt og (3) sammensying av nettverk. [Mid93] og [Mid94] beskriver algoritmer for sletting av punkter i en Delaunaytriangulering. Basisalgoritmen går i korthet ut på at den omskrivende polygonen til det punktet som skal slettes, blir oppsøkt. Deretter legges omskrivende sirkler gjennom de tre endepunktene til to påfølgende sider i polygonen. Den trekant som gir den minste omskrivende sirkel, velges. Slik fortsetter algoritmen til hele området innenfor poly-

gonen er triangulert. I de tilfeller at en innvendig vinkel i polygonen er større enn π, er det nødvendig med spesiell håndtering av triangulering [Mid93].

3.5.6 Fletting av Delaunaynettverk

[Mid93] beskriver en algoritme for å flette sammen Delaunaynettverk, som er en svært aktuell problemstilling ved store nettverk. Ved å dele opp trianguleringen i fliser, vil en rekke forespørsler mot nettverket gå raskere enn om vi opererte med hele nettverket i en enkelt tabell. Vi kan ved flisemetoden dele opp punktmengden i delmengder og triangulere hver delmengde uavhengig av de andre delemengdene. Dersom det blir behov for å ha et nettverk som spenner over flere fliser, kan de aktuelle nettverkene sys sammen til et sammenhengende nettverk.

3.6 Hierarkisk triangulering

Det finnes metoder for å etablere hierarkiske trekantstrukturer. En metode [DFP95] baserer seg på en dekomponering av den enkelte trekant. Problemet med denne metoden er at strukturen lett blir komplisert og vanskelige å vedlikeholde ved endringer i nettverket. Vi kan også etablere et hierarki av uavhengige lag av trianguleringer. Utvalget av datapunkter baserer seg her på en skrankeparameter som avtar i størrelse etter hvert som vi går mot høyere detaljeringsgrad. Den dynamiske trianguleringsalgoritmen ligger vel til rette for å bygge opp hierarkiske TIN.

Kapittel 4

Operasjoner på TIN

4.1 Noen utvalgte geometriske betraktninger

Innenfor et triangel

Det å teste om et punkt P ligger innenfor et triangel ABC kan implementeres etter ulike strategier. En mulighet er å benytte en generell "innenfor polygon"algoritme. En annen mulighet er å utnytte det forhold at P sammen med det aktuelle trianglets tre hjørner definerer tre nye triangler og analysere arealet til trianglene som vist ved algoritmen i figur 4.1.

if *(areal(A,B,P)+areal(A,C,P)+areal(B,C,P) > areal(A,B,C))* **then**
begin
    {P ligger utenfor trekant ABC}
else
    {P ligger innenfor trekant ABC}
end;

if *(areal(A,B,P) < delta)* **then** {P ligger på eller i nærheten av AB}
if *(areal(A,C,P) < delta)* **then** {P ligger på eller i nærheten av AC}
if *(areal(B,C,P) < delta)* **then** { P ligger på eller i nærheten av BC}

Figur 4.1: Test for å finne ut om et punkt ligger innenfor et triangel

En attraktiv side ved denne måten å teste på, er at vi som et biprodukt får informasjon om hvor nære P ligger trianglets sider. Dersom ett av arealene er mindre enn δ, ligger P i nærheten av den tilhørende siden, og dersom to av arealene er mindre enn δ, ligger P i nærheten av det tilhørende hjørnet.

Referansepunkt for et triangel

Som referansepunkter forutsetter IGF-TIN at det benyttes punkter som ligger innenfor de respektive triangler. For trekant ABC kan et slikt punkt finnes ved å beregne gjennomsnittsverdien for midtpunktet til to av trianglets sider. Gjennomsnittsverdien vil alltid gi et punkt som ligger innenfor trianglet. La oss anta at vi benytter sidene AB og CA. Referansepunktets koordinater blir da:

$$x_Q = \frac{x_A + 2x_B + x_C}{4} \qquad \text{og} \qquad y_Q = \frac{y_A + 2y_B + y_C}{4}$$

4.2 Interpolere z-verdien i et vilkårlig punkt

Siden planet er delt opp i triangler av vilkårlig varierende størrelse, er det ikke mulig å finne noen enkel hash-funksjon som kan identifisere hvilket triangel punktet befinner seg innenfor.

Den søkestrategi som er valgt i IGF-TIN, baserer seg på følgende algoritme:

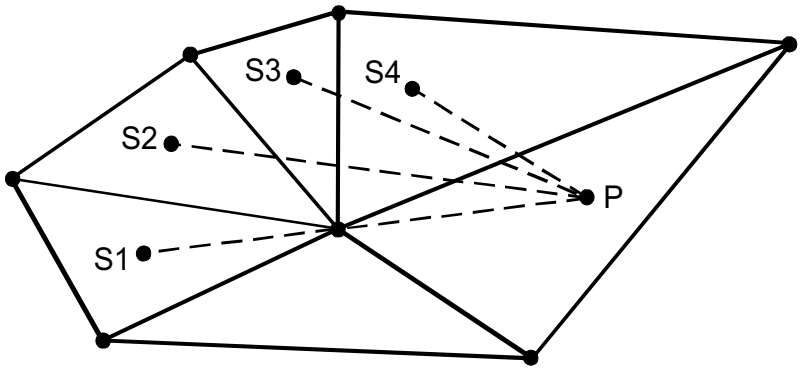
1. Etabler et referansepunkt i hvert triangel og opprett en datastruktur (f.eks. kvatre eller hash-tabell) over referansepunktene hvor referansepunktets (x, y)-koordinat er søkenøkkel og identifikatoren til det tilhørende triangel er attributt.

2. Anta at et punkt P er gitt. Finn først det referansepunktet R som ligger nærmest P. Dersom P ikke ligger innenfor det triangel som nå er funnet, gjøres et søk fra R i retning P . Det siste søket er ikke helt rett fram å implementere og trenger derfor en nærmere redegjørelse.

3. Anta at triangel T_1 er funnet under punkt 2. Det triangel P befinner seg innenfor, må krysse den rette linjen mellom T_1's referansepunkt S_1 og punktet P. Et problem med å søke langs denne linjen, forårsakes av spesialtilfellet at linjen går gjennom et triangelhjørne. Neste triangel vil i så fall ikke så lett la seg finne. Spesilatilfellet kan unngås ved å flytte startpunktet for søkelinjen til referansepunktet for det nabotrianglet søkelinjen går inn i. Vi får da følgende algoritme (som er illustrert i figur 4.2):

   - Finn den triangelside E_1 i triangel T_1 linjen S_1, P krysser. Dersom denne linjen krysser et av trianglets hjørner, velges *vilkårlig* en av de tilstøtende sider som E_1.
   - Finn det triangel T_2 som er nabo til T_1 og som har E_1 som felles side med T_1.
   - Gjenta søket, men nå med startpunkt for søkelinjen i referansepunkt S_2 til nabotrianglet T_2. Fordi punktene S_2 og P både ligger på den samme siden av E_1 og utenfor T_1, vil linjen S_2, P ikke kunne krysse noen av hjørnene til T_1.

- Søket fortsetter på samme måten med neste nabotriangel og terminerer når trianglet som omslutter P, er funnet.



Figur 4.2: Strategi for å finne det triangel som omslutter et punkt

4.3 Linjeberegninger

4.3.1 Profilere langs en vilkårlig polygon

Gitt en polygon som består av n rette linjestykker. Oppgaven består i å finne mulige skjæringspunkter mellom polygonen og nettverket av triangler og interpolere høyder i disse punktene. Vi tenker oss et punkt (*flytende punkt*) som flytter seg langs polygonen. Når det flytende punktet enten faller sammen med et hjørne i polygonen eller et skjæringspunkt mellom polygonen og en triangelside, interpoleres høyden i det flytende punktet. Anta at vi har en rett linje mellom de to punktene P_i og P_{i+1}. Dersom vi vet hvilket triangel punktet P_i befinner seg i, kan vi lett finne hvilken triangelside linjen P_i, P_{i+1} går ut gjennom. Hovedtilfellet er at linjen krysser et av trianglets sider. I dette tilfellet kan det entydig fastslås hvilket neste triangel linjen går inn i. Et spesialtilfelle er at linjen går ut gjennom et av trianglets hjørner. Problemet nå er at vi ikke uten videre kan fastslå neste triangel. Den løsningen som er valgt i IGF-TIN, er som følger:

1. Plasser det flytende punktet på linjen P_i, P_{i+1} i en liten avstand δ fra P_i i retning P_{i+1}.

2. Finn så det triangel det flytende punktet befinner seg innenfor. Normalt vil trianglet nå bli funnet, men en kan risikere at punktet ligger på eller så nære en triangelside at det ikke kan fastslås med sikkerhet hvilket triangel punktet ligger innenfor. En løsning kunne være å inkrementere med δ videre langs linjen inntil entydighet forelå. Dersom δ er liten og den aktuelle polygonsiden faller sammen med en triangelside, kan antall inkrement bli stort. Det er mulig

å velge en bedre løsning. Anta at valget står mellom trianglene T_1 og T_2. Vi undersøker så om polygonsiden P_i, P_{i+1} har et skjæringspunkt med T_1 eller T_2. Dersom så er tilfelle og skjæringspunktet ligger mellom det flytende punktet og P_{i+1}, flyttes det flytende punktet til dette skjæringspunktet. Skjæring med T_1 eller T_2 lar seg ikke finne om P_{i+1} ligger innenfor ett av de to trianglene. I så fall flyttes det flytende punktet til P_{i+1}.

Vår algoritme må følgelig takle tre tilfeller som gjelder det triangel det flytende punktet befinner seg innenfor.

1. Trianglet er ikke kjent
2. Trianglet lar seg ikke entydig identifisere på grunn av at det flytende punktet ligger for nære en triangelside.
3. Trianglet er kjent

4.3.2 Generere isolinjer

Generering av isolinjer faller enkelt dersom vi baserer oss på lineær interpolasjon. Under forutsetningen om lineær interpolasjon vil en isolinje aldri kunne gå ut av et triangel gjennom den samme siden den kommer inn i. Den må derfor gå ut gjennom en av de to andre sidene. Som vi ser av figur 4.3 gir lineær interpolasjon isolinjer med skarpe knekker. Dette kan kompenseres ved å etterbehandle linjene med en glattingsrutine, for eksempel Akimas metode [Aki70]. Akimas metode har den fordel at vi unngår at den glattede linjen får store slyngende utslag i forhold til den opprinnelige linjen. Dette vil hindre glattede isolinjer i å krysse hverandre.

I det spesielle tilfellet at isolinjen går gjennom et hjørne til et triangel, lønner det seg å addere til et lite beløp til det aktuelle triangelhjørnets z-koordinat for på den måten å tvinge isolinjen utenom triangelhjørnet. Dette gjør at vi omgår kompliserte tester.

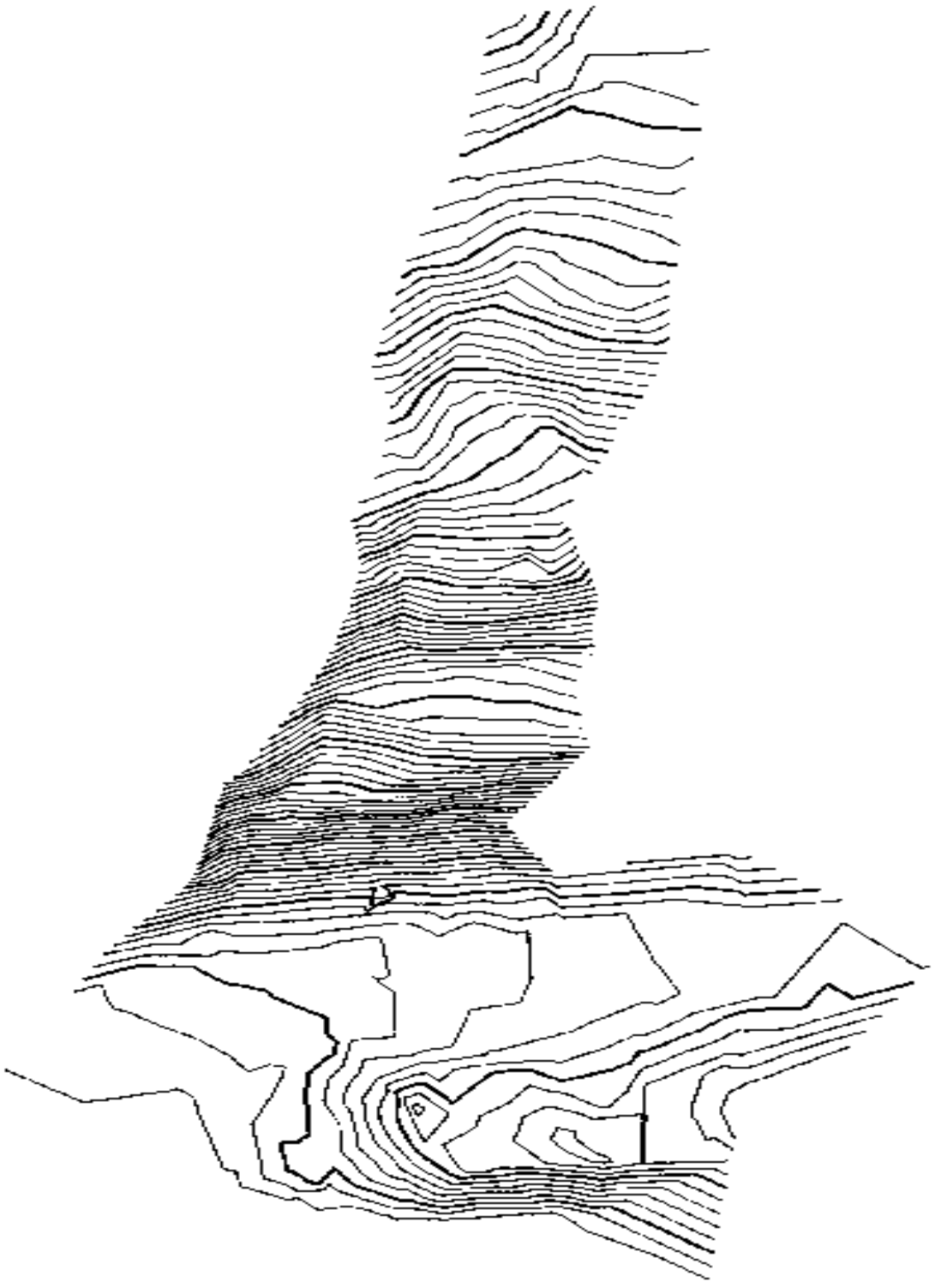
4.4 Metriske beregninger

Volumberegning

Volumet som defineres av toppflaten til et triangel og projeksjonen av trianglet på referanseflaten for høydene faller enkel dersom vi forutsetter lineær interpolasjon av z-koordinaten (terrenghøyden). Volumet v er i dette tilfellet gitt ved

$$v = g\frac{h_1 + h_2 + h_3}{3}, \tag{4.1}$$

hvor h-ene er avstander fra referanseplanet og g det horisontale arealet til trianglet. Riktigheten til denne formelen kan kontrolleres ved å splitte prismet i to volum, et



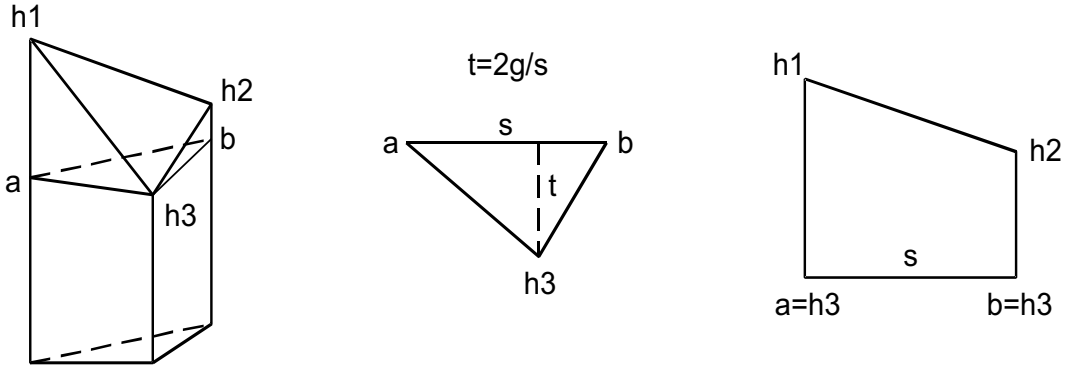
Figur 4.3: Lineær interpolasjon av isolinjer i en TIN

rett avkortet prisme og en pyramide. Delingen gjøres ved å legge et horisontalt plan gjennom det (x,y,z) punktet som har den minste h-verdi. Anta at dette er h_3. Vi har da ifølge figur 4.4

$$v = gh_3 + \frac{1}{3}\left(\frac{1}{2}s\left(h_1 - h_3 + h_2 - h_3\right)\frac{2g}{s}\right) = g\frac{h_1 + h_2 + h_3}{3},$$

som er identisk med uttrykket i likning 4.1.

Volumet av hele det triangulerte området finnes ved å summere over samtlige triangler.



Figur 4.4: Beregne volumet av et prisme med skjev toppflate

Overflateareal

Overflatearealet av terrengmodellen finnes ved å summere over samtlige trekanter. Det kan her være hensiktsmessig å benytte Heron's formel for å beregne skråarealet av de enkelte trekanter.

Kapittel 5

Kriging

Kriging er en interpolasjonsmetode som henter sitt teoretiske grunnlag fra geostatistikken. Den Sør-Afrikanske gruveingeniør D.G. Krige introduserte metoden, men metodens matematiske og statistiske grunnlag er i stor grad videreutviklet av den franske geomatematiker Georges Matheron [Mat71]. Krige tok utgangspunkt i datidens interpolasjonsmetoder og stilte følgende kritiske spørsmål til flytende flate interpolasjon:

- hvor stort bør vinduet være?
- hvilken form og orientering må vinduet ha for optimal interpolering?
- hvordan skal vektene estimeres?
- hvilken usikkerhet har de interpolerte punktene.

Disse spørsmålene ledet Krige og senere Matheron fram til en interpolasjonsmetode som baserer seg på datamaterialets statistiske egenskaper. Metoden ble først anvendt innenfor gruveindustrien, men på grunn av metodens generelle karakter, har den fått innpass i en rekke fagområder. Mange av de programpakker som i dag er på markedet for digitale terrengmodeller, har kriging som en av flere valgbare interpolasjonsmetoder.

Kriging finnes i flere varianter: enkel kriging, ordinær kriging, universell kriging, ko-kriging, blokk-kriging m.fl. Av nyere litteratur om kriging anbefales blant annet [Cre93], siden vi i denne læreboken finner en meget omfattende og grundig presentasjon av emnet. Av eldre litteratur anbefales blant annet [Ole75].

5.1 Introduksjon til det statistiske grunnlaget for kriging

En *random variabel* har en sannsynlighetsfordeling knyttet til seg. En *realisering* av en random funksjon er et gitt sett av funksjonsverdier. Dersom de betingelser

som har betydning for utfallet av en random funksjon, ikke endrer seg i løpet av forsøksperioden, kalles funksjonen *stasjonær*. En random funksjon X er *strengt stasjonær* dersom:

$$\mu(u) = E\Big[X(u)\Big] = E\Big[X(u+h)\Big] = \mu(u+h),$$

$$\sigma^2 = Var(X(u)) = Var(X(u+h)),$$

$$E\Big[X(0)X(h)\Big] = E\Big[X(u)X(u+h)\Big].$$

Altså middelverdifunksjonen og variansen er konstant samt at korrelasjonen kun er avhengig av u og h. Under bestemte vilkår kan stasjonære funksjoner være *ergodiske*. Litt forenklet kan vi si at en funksjon er ergodisk dersom dens statistiske egenskaper kan beregnes på grunnlag et sampel. Bare stasjonære funksjoner kan være ergodiske. Variansen til X i punktet u kan finnes av

$$Var(X(u)) = E\Big[\Big\{X(u) - \mu(u)\Big\}^2\Big], \tag{5.1}$$

og kovariansen til X i forhold til punktene u_j og u_{jj} kan finnes av

$$Cov(X(u_j), X(u_{jj})) = E\Big[\Big(X(u_j) - \mu(u_j)\Big)\Big(X(u_{jj}) - \mu(u_{jj})\Big)\Big]. \tag{5.2}$$

I det spesielle tilfellet at vi kan anta at $\mu(u) = 0$, kan beregningen basere seg på det forenklede uttrykket

$$Var(u) = E\Big[\Big\{X(u)\Big\}^2\Big] \text{ og } Cov(u_j, u_{jj}) = E\Big[X(u_j)X(u_{jj})\Big] \text{ for } \mu(u) = 0. \tag{5.3}$$

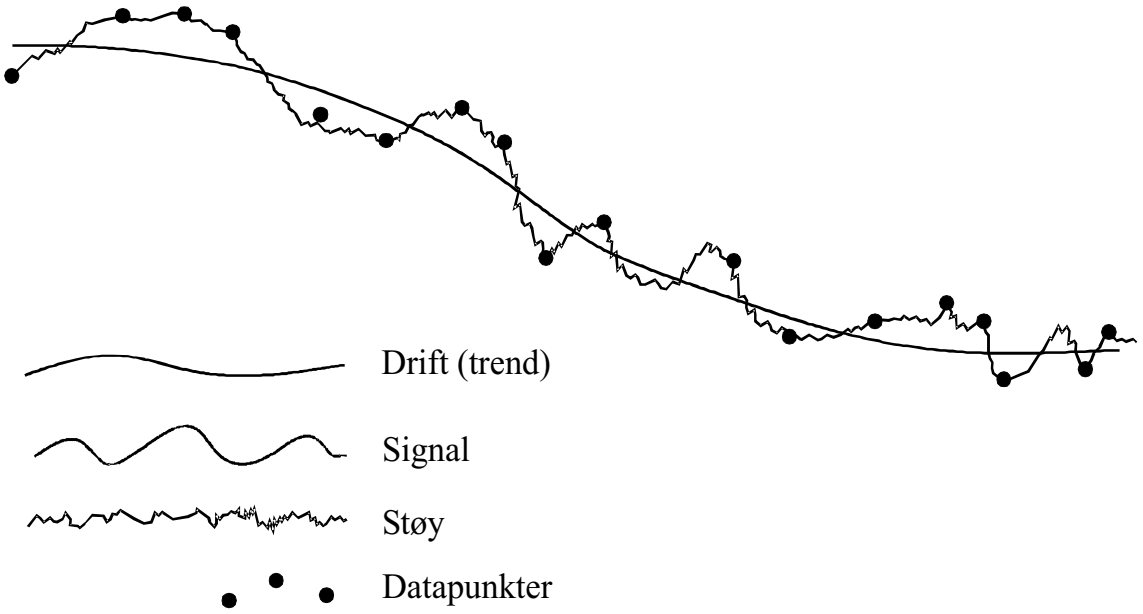
Som vi senere skal se, vil vi i forbindelse med kriging gjøre forutsetninger om middelverdifunksjon slik at vi kan basere oss på de forenklede uttrykk i likning 5.3

Et *random felt* består av bare en random funksjon, men av mere enn en uavhengig variabel. Vindhastighet er et eksempel på et firedimensjonalt random felt: tid og tre dimensjoner til et Euklidsk rom. For felt byttes konseptet *stasjonært* med *homogent* og *isotropt* (eng. isotropic). Med homogent menes at feltets statistiske egenskaper er uavhengig av posisjonen. Isotropt betyr lik i alle retninger. I et homogent felt er for eksempel middelverdi-funksjonen uavhengig av posisjonen i feltet. Dersom autokorrelasjonen (kovariansen som funksjon av avstanden mellom punkter) i tillegg er invariant under rotasjon, er ikke feltet bare homogent, men også isotropt.

For nærmer studier av stokastiske funksjoner vises for eksempel til en lærebok av D.Kannan [Kan79].

Romlig variasjon

Hvor vellykket *kriging* er som optimal interpolator, beror på hvor godt visse antagelser om den statistiske variasjon er oppfylt. Dette er viktig å ha klart for seg når metoden skal anvendes. Imidlertid viser det seg at metoden er robust overfor hvor godt de statistiske forutsetninger er oppfylt.



Figur 5.1: Hovedkomponentene til den romlige variasjon

Figur 5.1 viser hvordan kriging dekomponerer variasjonen til terrengoverflaten i de tre komponentene drift, signal og støy. Kriging er selvsagt ikke bundet til bare å beskrive terrengoverflater, men har en langt mere generell anvendelse innen GIS. Enhver kvantitativ størrelse som har en kontinuerlig utbredelse, kan interpoleres ved hjelp av kriging. La oss anta at terrengoverflaten kan oppfattes som en realisasjon av en stokastisk prosess $L(\cdot)$ og la u være en posisjon i $\mathbb{R}^n$. Den kvantitative variable $L(u)$ i posisjonen u vil vi modellere ved følgende likning:

$$L(u) = m(u) + s(u) + r(u), \tag{5.4}$$

hvor:

- $m(\cdot)$ defineres som $E[L(\cdot)]$ og er en deterministisk funksjon som beskriver den strukturelle komponenten til $L(\cdot)$. Funksjonen $m(\cdot)$ kalles *driften* eller trenden og er altså definert ved forvetningsverdien til $L(\cdot)$. Driften i punktet u er altså forventningsverdien til $L(\cdot)$ i punktet u. Siden vi oppfatter $L(\cdot)$ som en stokastisk prosess, er det meningsfylt å snakke om middelverdien eller forventningsverdien til $L(\cdot)$ i punktet u. Driften beskriver hovedvariasjonen til terrengoverflaten.

- $s(\cdot)$ er en random (stokastisk), romlig korrelert, lokalt varierende differanse i forhold til driften. Den kalles for *signalet* og er den størrelse kriging skal estimere. Signalet antas å være stasjonært og ha middelverdien null; $E[s(u)] = 0$.

- $r(\cdot)$ er en random (stokastisk), romlig ukorrelert variasjon som kalles *støyen*. Støyen klarer ikke vår interpolator å modellere, og den er derfor et uttrykk for hvor godt interpolatoren er i stand til å beskrive virkeligheten. Jo større støyen er, jo større usikkerhet vil være knyttet til de interpolerte punktene. En forutsetning for kriging er derfor at støyen er underordnet størrelsen til signalet. Vi antar at $E[r(u)] = 0$.

Likning 5.4 kan skrives som

$$l(u) = s(u) + r(u) = L(u) - m(u), \tag{5.5}$$

hvor vi forutsetter at

$$E\Big[l(u)\Big] = E\Big[s(u)\Big] = E\Big[r(u)\Big] = 0. \tag{5.6}$$

Hvor godt betingelsene i likning 5.6 er oppfylt, avhenger av om driften er omhyggelig fjernet.

5.2 Semivariogram

Vi ønsker å finne variansen til differanser mellom to posisjoner som er atskilt ved en avstandsvektor h. Variansen er gitt ved

$$2\gamma(h) = E\left[(l(u) - l(u+h))^2\right], \tag{5.7}$$

hvor $\gamma(h)$ er en funksjon som benevnes *semivariansen*. Hvorfor faktoren 2 er innført, vil senere få en forklaring.

Vi innfører følgende forkortede skrivemåter:

$$\begin{aligned} l_1 = l(u) \quad & l_2 = l(u+h), \\ s_1 = s(u) \quad & s_2 = s(u+h), \\ r_1 = r(u) \quad & r_2 = r(u+h). \end{aligned}$$

Semivariansen kan etter dette skrives som

$$2\gamma(h) = E\Big[l_1^2 + l_2^2 - 2l_1l_2\Big]. \tag{5.8}$$

Ved å benytte modellen som er gitt ved likningen 5.5, får vi

$$\begin{aligned} 2\gamma(h) &= E\Big[(s_1+r_1)^2 + (s_2+r_2)^2 - 2(s_1+r_1)(s_2+r_2)\Big] \\ &= E\Big[s_1^2 + s_2^2\Big] + E\Big[r_1^2 + r_2^2\Big] + 2E\Big[s_1(r_1-r_2) + s_2(r_2-r_1)\Big] \\ &\quad -2E\Big[s_1s_2\Big] - 2E\Big[r_1r_2\Big]. \end{aligned} \tag{5.9}$$

Under forutsetning om at vi har med et homogent random felt å gjøre, følger at

$$E\Big[s_1^2\Big] = E\Big[s_2^2\Big] = E\Big[s^2\Big], \tag{5.10}$$

$$E\Big[r_1^2\Big] = E\Big[r_2^2\Big] = E\Big[r^2\Big]. \tag{5.11}$$

I det tilfellet at $|h| = 0$, vil (u) og $(u+h)$ ikke lenger være distinkte punkt. Dersom vi ser bort fra eventuelle målefeil i dataene, vil vi derfor i tilfellet $|h| = 0$ ha at $r_1 = r_2$. Under denne forutsetningen vil kriging framstå som en eksakt interpolator. Siden vi har forutsatt at $r(\cdot)$ er en random ukorrelert stokastisk prosess, vil vi for $|h| \neq 0$ ha at både $E\Big[r_i - r_j\Big] = 0$ og $E\Big[r_i r_j\Big] = 0$. Vi innfører de nevnte forutsetninger i likning 5.9 og får

$$2\gamma(h) = 2E\Big[s^2\Big] + 2E\Big[r^2\Big] - 2E\Big[s_1 s_2\Big], \tag{5.12}$$

som viser hvorfor vi innførte faktoren 2 i likning 5.7. Ved å benytte betingelsene gitt i likning 5.6, kan vi nå finne semivariansen som en funksjon av variansen til signalet, variansen til støyen og autokorrelasjonen for signalet:

$$\gamma(h) = \begin{cases} 0 & dersom \quad \mid h \mid = 0, \\ Var(s) + Var(r) - Cov_s(h) & ellers. \end{cases} \tag{5.13}$$

Av våre forutsetninger følger at

$$\begin{aligned} \lim_{|h|\to 0} \gamma(h) &= Var(r), \\ \lim_{|h|\to \infty} \gamma(h) &= Var(s) + Var(r) = Var(l), \\ \gamma(0) &= 0. \end{aligned}$$

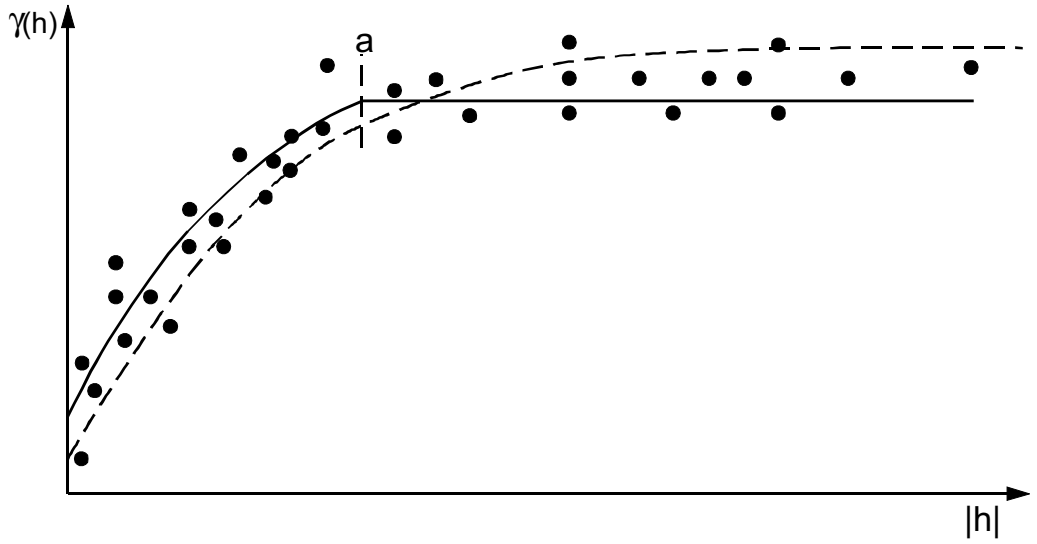
Dersom forutsetningene om et homogent, isotropt random felt er oppfylt og en eventuell drift omhyggelig fjernet, vil semivariogrammet få en karakteristisk form som vist i figur 5.2. Ved en empirisk bestemmelse av funksjonen $\gamma(h)$, er det vanlig å modellere semivariansen som diskontinuerlig i $|h| = 0$ med et sprang lik $Var(r)$. En eventuell kurvetilpassing må derfor kun gjøres mot verdier av $|h| > 0$. Semivariansen modelleres som større enn null for $|h| = 0$ i det tilfellet at vi ønsker å ta omsyn til eventuelle målefeil i det observerte datamaterialet.

Det kan velges flere modeller for å tilpasse en glatt kurve til en punktsverm i et semivariogram. Vi skal angi tre slike modeller.

a) Sfærisk modell

$$\gamma(h) = \begin{cases} c_0 + c_1 \left[\frac{3|h|}{2a} - \frac{1}{2}(\frac{|h|}{a})^3\right] & dersom \;\; 0 < |h| < a, \\ c_0 + c_1 & dersom \;\; |h| \geq a, \\ 0 & dersom \;\; |h| = 0. \end{cases}$$

Den sfæriske modellen lager en knekk i punktet $|h| = a$ og gir en horisontal linje for $a \geq |h|$, som illustrert ved den heltrukne linjen i fig 5.2. Burrough [Bur87] benytter denne modellen.



Figur 5.2: Eksempel på et typisk semivariogram

b) Eksponentiell modell

$$\gamma(h) = \begin{cases} c_0 + c_1 \left[1 - exp^{\frac{-|h|}{a}}\right] & dersom \ \ |h| > 0, \\ 0 & dersom \ \ |h| = 0. \end{cases}$$

Den eksponentielle modellen gir en kontinuerlig kurve for $|h| > 0$ og vil flate av og bli tilnærmet horisontal når $|h|$ blir stor nok. Dette er illustrert ved den stiplete kurven i fig 5.2.

c) Lineær modell

$$\gamma(h) = \begin{cases} c_0 + c_1|h| & dersom \ \ |h| > 0, \\ 0 & dersom \ \ |h| = 0. \end{cases}$$

Den lineære modellen er enkel, men kan være akseptabel for et mindre intervall av $|h|$, altså for et intervall $[\min < |h| < \max]$.

I noen tilfeller kan $Var(r)$ være så dominerende at den empiriske semivariansen ikke viser noen tendens til å øke med $|h|$. I dette tilfellet bør det derfor ikke utføres kurvetilpassing i semivariogrammet, men $L(u)$ bør estimeres ved middeltallsberegning uten å ta romlig korrelasjon i betraktning.

Burrough [Bur87] peker på at i praksis vil kravet om et isotropt felt ikke alltid være like godt oppfylt. Dette gjør at semivariogrammet ka få en form som er retningsbestemt. Han viser at denne variasjonen kan modelleres innenfor et enkelt semivariogram ved at anta at semivariogrammet i nord-syd retning er en strukket"versjon av semivariogrammet i øst-vest retning. Den teknikken han referer til,

finner først den retning Φ som gir maksimum variasjon. Deretter finnes semivariogrammet i retningen $\Phi + \pi/2$. På grunnlag av disse to semivariogrammene, bestemmes koeffisientene i en strekkslik at semivariogrammet i en vilkårlig retning lar seg beregne.

5.3 Kovariansfunksjon

Under forutsetning om random homogent felt lar kovariansfunksjonen (autokorrelasjonen) $Cov_l(h)$ for $l(\cdot)$ seg uttrykke ved semivariansen til $l(\cdot)$. Siden vi ifølge likning 5.6 forutsetter at $E[l(\cdot)] = 0$, kan kovariansen til $l(\cdot)$ skrives som

$$Cov_l(h) = E\Big[l(u)l(u+h)\Big] = E\Big[l_1 l_2\Big]. \tag{5.14}$$

Ved å benytte modellen i likning 5.5, får vi at uttrykket for kovariansen går over til

$$\begin{aligned} Cov_l(h) &= E\Big[(s_1 + r_1)(s_2 + r_2)\Big] \\ &= E\Big[s_1 s_2\Big] + E\Big[s_1 r_2\Big] + E\Big[s_2 r_1\Big] + E\Big[r_1 r_2\Big], \end{aligned}$$

som under forutsetning om random homogent felt går over til

$$Cov_l(h) = E\Big[s_1 s_2\Big] = Cov_s(h). \tag{5.15}$$

Ifølge likning 5.13 kan vi finne kovariansen til signalet ved $Cov_s(h) = Var(s) + Var(r) - \gamma(h) = Var(l) - \gamma(h)$, som innført i likning 5.15 gir oss

$$Cov_l(h) = \begin{cases} Var(l) & dersom \;\; |h| = 0, \\ Var(l) - \gamma(h) & ellers. \end{cases} \tag{5.16}$$

5.4 Enkel kriging

Enkel kriging (eng. simple kriging) [Cre93] (side 110) baserer seg på at driften $m(\cdot)$ er kjent. Etter at driften er fjernet, sitter vi tilbake med følgende vektor av gitte datapunkter:

$$\mathbf{l} = \begin{pmatrix} l(u_1) \\ l(u_2) \\ \vdots \\ l(u_n) \end{pmatrix} = \begin{pmatrix} l_1 \\ l_2 \\ \vdots \\ l_n \end{pmatrix} \tag{5.17}$$

Oppgaven er nå å finne et estimat for høyden i punktet u_0.

Estimatet for høyden i punktet u_0 benevner vi $\hat{s}(u_0)$. I utgangspunktet benytter vi den røffe tilnærmelse at estimatet søkes ved en lineær kombinasjon av elementene i datavektoren $\mathbf{l}$. Som vi skal vise, vil vi forfine modellen ved at de ulike datapunktene

skal inngå i den lineære kombinasjonen med vekter som fastsettes på grunnlag av deres romlige korrelasjonen (semivarians). Estimatet i punktet u_0 er gitt ved

$$\hat{s}(u_0) = \mathbf{a}^t\mathbf{l}, \tag{5.18}$$

hvor

$$\mathbf{a}^t = [a_1 \;\; a_2 \;\; \cdots \;\; a_n]. \tag{5.19}$$

Problemet er å bestemme de ukjente koeffisientene slik at vi finner et godt estimat for høyden i det punktet som skal interpoleres.

Avviket mellom estimatet og den virkelige verdien i u_0 er gitt ved

$$v = s(u_0) - \hat{s}(u_0) = s_0 - \hat{s}_0 = s_0 - \mathbf{a}^t\,\mathbf{l},$$

eller

$$v = [1 - \mathbf{a}^t] \begin{bmatrix} s_0 \\ \mathbf{l} \end{bmatrix} \tag{5.20}$$

Den første innskytelse er kanskje å bestemme vektoren $\mathbf{a}$ slik at tallverdien til v blir så liten som mulig. Det en imidlertid gjør, er å basere minimaliseringen på $E\Big[v^2\Big]$ ved å søke den vektor $\mathbf{a}$ som gir $E[v^2] = \min$. Ifølge vår modell i likning 5.20 har vi

$$E\Big[v^2\Big] = \sigma_v^2 = E\Big[(s_0 - a_1l_1 - a_2l_2 - \cdots - a_nl_n)^2\Big].$$

Ved å regne ut kvadratet i uttrykket ovenfor, får vi

$$E\Big[v^2\Big] = E\Big[\begin{array}{ccccc} s_0^2 & -s_0a_1l_1 & -s_0a_2l_2 & -\cdots & -s_0a_nl_n \\ -a_1l_1s_0 & +a_1^2l_1^2 & +a_1l_1a_2l_2 & +\cdots & +a_1l_1a_nl_n \\ -a_2l_2s_0 & +a_2l_2a_1l_1 & +a_2^2l_2^2 & +\cdots & +a_2l_2a_nl_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_nl_ns_0 & +a_nl_na_1l_1 & +a_nl_na_2l_2 & +\cdots & +a_n^2l_n^2 \end{array}\Big]. \tag{5.21}$$

Av våre forutsetninger om homogent felt følger at $Var(l(u_i)) = Var(l(u_j))$. Siden vi forutsetter at $E[l(u)] = 0$, har vi

$$E\Big[l_1^2\Big] = E\Big[l_2^2\Big] = \cdots = E\Big[l_n^2\Big] = Var(l) = \sigma_l^2.$$

Vi innfører følgende betegnelser:

$$\sigma_{l_il_j} = Cov(l_il_j) = E\Big[l_il_j\Big] = \sigma_{l_jl_i},$$

$$\sigma_{sl_i} = Cov(s_0l_i) = E\Big[s_0l_i\Big] = \sigma_{l_is}.$$

På grunnlag av likning 5.21 danner vi varians/kovariansmatrisen til de random variable s_0 og $\mathbf{l}$, gitt ved

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_s^2 & \sigma_{sl} \\ \sigma_{ls} & \mathbf{\Sigma}_{ll} \end{bmatrix} = \left[\begin{array}{c|cccc} \sigma_s^2 & \sigma_{sl_1} & \sigma_{sl_1} & \cdots & \sigma_{sl_n} \\ \hline \sigma_{l_1s} & \sigma_l^2 & \sigma_{l_1l_2} & \cdots & \sigma_{l_1l_n} \\ \sigma_{l_2s} & \sigma_{l_2l_1} & \sigma_l^2 & \cdots & \sigma_{l_2l_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{l_ns} & \sigma_{l_nl_1} & \sigma_{l_nl_2} & \cdots & \sigma_l^2 \end{array}\right] \tag{5.22}$$

I likning 5.22 er matrisen $\mathbf{\Sigma}$ en $(n+1) \times (n+1)$ kvadratisk matrise som består av:

- σ_s^2 som er variansen i det punktet vi skal estimere høyden for.
- σ_{sl} en $1 \times n$ matrise som består av autokovarianser mellom det punktet høyden skal estimeres for og samtlige punkter i datavektoren $\mathbf{l}$. Autokovariansene kan finnes med utgangspunkt i likning 5.16.
- $\sigma_{ls} = \sigma_{sl}^t$ en $n \times 1$ vektor.
- $\mathbf{\Sigma}_{ll}$ en symmetrisk, $n \times n$ kvadratisk matrise bestående av autokovarianser for datavektoren $\mathbf{l}$ med variansen σ_l^2 på diagonalen. Autokovariansene kan finnes med utgangspunkt i likning 5.16.

Med utgangspunkt i likning 5.21 kan variansen til v skrives på matriseform som følger:

$$\sigma_v^2 = \mathbf{j}^t \mathbf{\Sigma}\, \mathbf{j} = \left[1 - \mathbf{a}^t\right] \mathbf{\Sigma} \begin{bmatrix} 1 \\ -\mathbf{a} \end{bmatrix} \tag{5.23}$$

Ved å benytte likningene 5.22 og 5.23 får vi:

$$\sigma_v^2 = \sigma_s^2 - \mathbf{a}^t \sigma_{ls} - \sigma_{sl}\, \mathbf{a} + \mathbf{a}^t \mathbf{\Sigma}_{ll}\, \mathbf{a},$$

som kan forenkles ved å innføre at $\sigma_{sl}\, \mathbf{a} = \mathbf{a}^t \sigma_{sl}^t = \mathbf{a}^t \sigma_{ls}$. Vi har da

$$\sigma_v^2 = \sigma_s^2 - 2\mathbf{a}^t \sigma_{ls} + \mathbf{a}^t \mathbf{\Sigma}_{ll}\, \mathbf{a}. \tag{5.24}$$

For å finne en minimum interpolasjonsfeil velger vi en vektor $\mathbf{a}$ slik at

$$\frac{\partial \sigma_v^2}{\partial \mathbf{a}} = 0.$$

Det totale differensial av σ_v^2 fås ved å differensiere likning 5.24:

$$\begin{aligned} d\sigma_v^2 &= -2d\mathbf{a}^t \cdot \sigma_{ls} + d\mathbf{a}^t \cdot \mathbf{\Sigma}_{ll}\mathbf{a} + \mathbf{a}^t \mathbf{\Sigma}_{ll} \cdot d\mathbf{a} \\ &= -2\sigma_{ls}^t \cdot d\mathbf{a} + (\mathbf{\Sigma}_{ll}\mathbf{a})^t \cdot d\mathbf{a} + \mathbf{a}^t \mathbf{\Sigma}_{ll} \cdot d\mathbf{a} \\ &= -2\sigma_{sl} \cdot d\mathbf{a} + \mathbf{a}^t \mathbf{\Sigma}_{ll}^t \cdot d\mathbf{a} + \mathbf{a}^t \mathbf{\Sigma}_{ll} \cdot d\mathbf{a}. \end{aligned}$$

Siden $\mathbf{\Sigma}_{ll}$ er en symmetrisk, kvadratisk matrise, er $\mathbf{\Sigma}_{ll}^t = \mathbf{\Sigma}_{ll}$. Dette innført gir oss følgende likning til bestemmelse av $\mathbf{a}$:

$$\frac{\partial \sigma_v^2}{\partial \mathbf{a}} = -2\sigma_{sl} + 2\mathbf{a}^t \mathbf{\Sigma}_{ll} = 0. \tag{5.25}$$

Likning 5.25 gir oss denne løsningen for vektoren $\mathbf{a}$:

$$\begin{aligned} \mathbf{a}^t \mathbf{\Sigma}_{ll} &= \sigma_{sl}, \\ (\mathbf{a}^t \mathbf{\Sigma}_{ll})^t &= \sigma_{sl}^t, \\ \mathbf{\Sigma}_{ll}^t \mathbf{a} &= \sigma_{ls}, \\ \mathbf{\Sigma}_{ll} \mathbf{a} &= \sigma_{ls}, \\ \mathbf{a} &= \mathbf{\Sigma}_{ll}^{-1} \sigma_{ls}. \end{aligned} \tag{5.26}$$

Vi kan nå finne en verdi for estimatet $\hat{s}_0$ ved å kombinere likningene 5.26 og 5.18:

$$\begin{aligned} \hat{s}_0 &= (\mathbf{\Sigma}_{ll}^{-1}\sigma_{ls})^t\mathbf{l} \\ &= \sigma_{ls}^t(\mathbf{\Sigma}_{ll}^{-1})^t\mathbf{l} \\ &= \sigma_{sl}\mathbf{\Sigma}_{ll}^{-1}\,\mathbf{l}. \end{aligned} \tag{5.27}$$

Et pluss ved kriging er at vi kan få utledet et uttrykk for usikkerheten til interpolasjonen. Vi finner dette med utgangpunkt i likning 5.24 og likning 5.26

$$\begin{aligned} \sigma_{\hat{s}}^2 = \sigma_v^2 &= \sigma_s^2 + \mathbf{a}^t(\mathbf{\Sigma}_{ll}\mathbf{a} - 2\sigma_{ls}) \\ &= \sigma_s^2 + \mathbf{a}^t(\mathbf{\Sigma}_{ll}\mathbf{\Sigma}_{ll}^{-1}\sigma_{ls} - 2\sigma_{ls}) \\ &= \sigma_s^2 - \mathbf{a}^t\sigma_{ls} \\ &= Cov_s(0) - \mathbf{a}^t\sigma_{ls}. \end{aligned} \tag{5.28}$$

Vi summerer opp de løsninger vi har kommet fram til:

$$\hat{s}_0 = \mathbf{a}^t\mathbf{l}$$

$$\sigma_v^2 = \sigma_s^2 - \mathbf{a}^t\sigma_{ls}$$

hvor :

$$\mathbf{a} = \mathbf{\Sigma}_{ll}^{-1}\sigma_{ls} \qquad \sigma_{ls} = \begin{bmatrix} \sigma_{l_1 s} \\ \sigma_{l_2 s} \\ \vdots \\ \sigma_{l_n s} \end{bmatrix} \quad \mathbf{\Sigma_{ll}} = \begin{bmatrix} \sigma_l^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_n} \\ \sigma_{l_2 l_1} & \sigma_l^2 & \cdots & \sigma_{l_2 l_n} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{l_n l_1} & \sigma_{l_n l_2} & \cdots & \sigma_l^2 \end{bmatrix}$$

Vi har nå vist hvordan enkel kriging kan beregnes med utgangspunkt i kovariansfunksjonen $Cov_l(h)$. Det lar seg vise at under forutsetning om random homogent felt kan vi i likning 5.27 erstatte kovariansene med semivarianser, se [Cre93] (side 122-123).

5.5 Universell kriging

Universell kriging (eng. universal kriging) [Cre93] (side 151-183) forsøker å overvinne problemet fra enkel kriging med at driften må være kjent på forhånd. Selv om vi i universell kriging betrakter driften som ukjent, er likevel metoden ikke rett fram, fordi universell kriging forutsetter at vi må kjenne semivariogrammet eller kovariansfunksjonen for residualene $l(\cdot)$ mellom $m(\cdot)$ og $L(\cdot)$. Det forholder seg slik at dersom vi kjenner $m(\cdot)$ lar $\gamma_l(h)$ seg beregne. På den annen side dersom vi kjenner $\gamma_l(h)$ lar $m(\cdot)$ seg beregne. Vi kan derfor si at universell kriging på en måte fører til et resonnement som biter seg selv i halen."Vi vil i den videre framstilling bygge på [Ole75] og utlede kriginglikningene med utgangspunkt i kovariansfunksjonen til $l(\cdot)$.

5.5.1 Utledning av likningene for universell kriging

Definisjon 2 *En regionalisert variabel er enhver numerisk funksjon som har en romlig fordeling som varierer fra sted til sted med tilstrekkelig kontinuitet, men hvis endring ikke lar seg representere ved noen beregnbar funksjon.*

Denne definisjonen karakteriserer mange størrelser som beskriver geografiske forekomster. Anta at vi har funksjonen $h = L(u)$ som på grunnlag av noen kjente datapunkter gir oss høyden til terrengoverflaten i en vilkårlig (x, y)-posisjon u. Siden terrengoverflaten har et stokastisk forløp, kan vi aldri ut fra spredte målinger beregne korrekt verdi for høyder i alle mellomliggende punkter. Funksjonen $L(u)$ kan vi derfor betrakte som en regionalisert variabel.

Definisjon 3 *Anta en regionalisert variabel $L(\cdot)$ med driften $m(\cdot)$. Driften defineres ved*

$$m(u) = E\Big[L(u)\Big],$$

hvor $E[\cdot]$ betyr forventningsverdien.

Definisjon 4 *Anta en regionalisert variabel $L(\cdot)$. Residualene til $L(\cdot)$ defineres ved*

$$l(u) = L(u) - m(u),$$

hvor $m(u)$ er driften til $L(\cdot)$ i punktet u. Kovariansfunksjonen til residualene benevnes $cov_l(u_j, u_{jj})$.

En viktig egenskap ved residualene er at de har forventningen null.

Korollar 1 *La $l(\cdot)$ være residualene for en regionalisert variabel. Da har vi at*

$$E\Big[l(u)\Big] = 0.$$

Bevis: Ved definisjon 4 har vi

$$E\Big[l(u)\Big] = E\Big[L(u) - m(u)\Big] = E\Big[L(u)\Big] - E\Big[m(u)\Big],$$

men for en gitt posisjon u er $m(u)$ ingen variabel, men en konstant. Derfor er forventningen til driften lik driften selv. Av definisjonen på driften følger at $E[L(u)]$ også er driften, derfor blir differansen lik null. □

Definisjon 5 *Anta en regionalisert variabel $L(\cdot)$. Vi definerer estimatet $\hat{L}(\cdot)$ for $L(\cdot)$ i punktet u ved en lineær kombinasjon av omkringliggende datapunkter; gitt ved*

$$\hat{L}(u) = \sum_{j=1}^{k} \lambda_j L(u_j),$$

hvor λ-ene er vektene i den lineære kombinasjonen.

Definisjon 6 *Optimalisering av $\hat{L}(u)$ vil bli gjort ved å innføre følgende betingelser:*

$$E\Big[\hat{L}(u_0) - L(u_0)\Big] = 0 \tag{5.29}$$

$$E\Big[(\hat{L}(u_0) - L(u_0))^2\Big] = \min \quad \textit{med omsyn på } \lambda_j. \tag{5.30}$$

Ifølge definisjon 6 har vi gjennom likning 5.29 innført betingelsen om at vi skal ha en forventningsrett estimator. Det å få innført dette kravet i tillegg til minimaliseringen, vil føre oss fram til et minimaliseringsproblem med sidevilkår. Det å innføre sidevilkåret, vil kreve en del matematiske utledninger. Disse vil nå bli gjennomgått.

Definisjon 7 *Driften til den regionaliserte variable $L(\cdot)$ i punktet u defineres analytisk ved*

$$m(u) = \sum_{i=0}^{n} a_i f^i(u).$$

Det er vanlig i definisjon 7 å uttrykke driften ved de første leddene i det bikubiske polynom. I dette tilfellet vil a-ene representere koeffisientene i polynomet, som i utgangspunktet er ukjente, og $f(u)$-ene vil være funksjoner av koordinatene til u, som for eksempel $1, x, y, xy, x^2, y^2$ osv.

Korollar 2 *La*

$$m(u) = \sum_{i=0}^{n} a_i f^i(u)$$

være uttrykket for driften til den regionaliserte variable i u. Da, dersom λ_j er vektene i estimatet for $\hat{L}(u_0)$, er

$$E\Big[\hat{L}(u_0)\Big] = \sum_{i=0}^{n}\sum_{j=1}^{k} a_i \lambda_j f^i(u_j).$$

Bevis: Ved definisjon 5 har vi at

$$E\Big[\hat{L}(u_0)\Big] = E\Big[\sum_{j=1}^{k} \lambda_j L(u_j)\Big].$$

Vi kan bytte om forventning og summering og fjerne konstanten λ_j fra forventningen og får

$$E\Big[\hat{L}(u_0)\Big] = \sum_{j=1}^{k} \lambda_j E\Big[L(u_j)\Big].$$

I henhold til definisjon 3 er forventningen $E\Big[L(u_j)\Big]$ lik driften i u_j, følgelig

$$E\Big[\hat{L}(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \sum_{i=0}^{n} a_i f^i(u_j).$$

Men λ_j kan plasseres i den andre summasjonen fordi den er uavhengig av i. Ved å bytte om summasjonene og snu produktet $\lambda_j a_i$, får vi

$$E\Big[\hat{L}(u_0)\Big] = \sum_{i=0}^{n} \sum_{j=1}^{k} a_i \lambda_j f^i(u_j).$$

□

Det neste teoremet formulerer kravet om forventningsrett estimator.

Teorem 3 *La estimatet og driften for $L(\cdot)$ i u_0 være gitt ved henholdsvis definisjon 5 og definisjon 7. Da gjelder at*

$$\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0) \quad \textit{for} \quad i = 0, 1, \ldots, n$$

hvis og bare hvis $E\Big[\hat{L}(u_0) - L(u_0) = 0\Big]$.

Bevis: La oss anta at $E[\hat{L}(u_0) - L(u_0) = 0]$. Ved vår hypotese har vi at

$$E\Big[\hat{L}(u_0)\Big] = E\Big[L(u_0)\Big],$$

som omformet på basis av uttrykket for $E\Big[\hat{L}(u_0)\Big]$ i korollar 2 går over til

$$\sum_{i=0}^{n} \sum_{j=1}^{k} a_i \lambda_j f^i(u_j) = E\Big[L(u_0)\Big].$$

Vi kan ta a_i ut fra den andre summasjonen. Dette gir oss

$$\sum_{i=0}^{n} a_i \sum_{j=1}^{k} \lambda_j f^i(u_j) = E\Big[L(u_0)\Big].$$

Ved definisjon 3 er forventningsverdien til den regionaliserte variable lik driften. Ved å innføre det analytiske uttrykket for driften fra definisjon 7, får vi

$$\sum_{i=0}^{n} a_i \sum_{j=1}^{k} \lambda_j f^i(u_j) = \sum_{i=0}^{n} a_i f^i(u_0).$$

Derfor har vi at

$$\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0) \quad \text{for } i = 0, 1, \ldots, n,$$

under den forutsetning at $E[\hat{L}(u_0) - L(u_0)] = 0$. Vi har nå bevist det første hvis i teoremet. Det gjenstår derfor å bevise at det andre hvis også holder. La

$$\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0) \quad \text{for } i = 0, 1, \ldots, n.$$

Ved å multiplisere begge sider av likningen med a_i, får vi

$$a_i \sum_{j=1}^{k} \lambda_j f^i(u_j) = a_i f^i(u_0) \quad \text{for } i = 0, 1, \ldots, n,$$

som kan skrives som

$$\sum_{i=0}^{n} a_i \sum_{j=1}^{k} \lambda_j f^i(u_j) = \sum_{i=0}^{n} a_i f^i(u_0).$$

Ved korollar 2 kan vi omforme venstre side av likningen slik at vi får

$$E\Big[\hat{L}(u_0)\Big] = \sum_{i=0}^{n} a_i f^i(u_0).$$

Høyre side av likningen er ifølge definisjon 7 driften i u_0. Ved så å benytte definisjon 3, vil omforming av høyre side av likningen resultere i

$$E\Big[\hat{L}(u_0)\Big] = E\Big[L(u_0)\Big]$$

eller

$$E\Big[\hat{L}(u_0) - L(u_0)\Big] = 0.$$

Herav følger det at

$$\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0) \quad \text{for } i = 0, 1, \ldots, n$$

impliserer

$$E\Big[\hat{L}(u_0) - L(u_0)\Big] = 0.$$

□

Teorem 3 er et viktig resultat, siden det gir oss en alternativ måte å formulere kravet om at differansen mellom estimert verdi i et punkt og den verdien som kan måles i punktet, ikke skal vise noen tendens til forskyvning i den ene eller den andre retning.

Dersom vi modellerer driften i u_0 ved en konstant; gitt ved $m(u_0) = a_0 f^0(u_0)$ hvor $f^0(u_0) = 1$, vil vi ifølge teorem 3 ha at vilkåret for forventningsrett estimator blir $\sum_{j=1}^{k} \lambda_j = 1$. Summen av vektene må altså i dette tilfellet være lik 1. Dersom vi benytter denne modellen for driften, kalles kriging for ordinær kriging. De neste korollar vil bli benyttet i formuleringen av $E[(\hat{L}(u_0) - L(u_0))^2]$.

Korollar 3 *La $L(\cdot)$ være en regionalisert variabel med driften $m(\cdot)$ og la $cov_l(u_j, u_{jj})$ være kovariansfunksjonen for residualene til $L(\cdot)$. Da er*

$$E\Big[L(u_j)L(u_{jj})\Big] = cov_l(u_j, u_{jj}) + m(u_j)m(u_{jj}).$$

Bevis: Ved definisjon 4 har vi

$$E\Big[L(u_j)L(u_{jj})\Big] = E\Big[\Big\{l(u_j) + m(u_j)\Big\}\Big\{l_{jj} + m(u_{jj})\Big\}\Big].$$

Dersom vi regner ut produktet og tar forventningen til hvert ledd, får vi

$$\begin{aligned} E\Big[L(u_j)L(u_{jj})\Big] &= E\Big[l(u_j)l(u_{jj})\Big] + E\Big[m(u_j)m(u_{jj})\Big] + \\ &\quad E\Big[m(u_j)l(u_{jj})\Big] + E\Big[m(u_{jj})l(u_j)\Big]. \end{aligned}$$

Siden forventningen til driften er driften selv, kan den tas ut av forventningsklammen. Dette gir oss

$$\begin{aligned} E\Big[L(u_j)L(u_{jj})\Big] &= E\Big[l(u_j)l(u_{jj})\Big] + m(u_j)m(u_{jj}) + \\ &\quad m(u_j)E\Big[l(u_{jj})\Big] + m(u_{jj})E\Big[l(u_j)\Big], \end{aligned}$$

men ifølge korollar 1 er $E[l(u)] = 0$. De to siste leddene i likningen faller derfor bort og det første leddet blir av samme grunn lik kovariansen til residualene. Herav får vi

$$E\Big[L(u_j)L(u_{jj})\Big] = cov(u_j, u_{jj}) + m(u_j)m(u_{jj}).$$

□

Korollar 4 *Anta en regionalisert variabel $L(\cdot)$. La $m(u_0)$ være driften i u_0 og la estimatet for $L(\cdot)$ i u_0 være gitt ved definisjon 5. Kovariansfunksjonen for residualene $l(\cdot)$ eksisterer og benevnes $cov_l(u_j, u_{jj})$. Da gjelder*

$$E\Big[\{\hat{L}(u_0)\}^2\Big] = \sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) + \{m(u_0)\}^2.$$

Bevis: Ved definisjon 5 har vi at

$$E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] = E\Big[\sum_{j=1}^{k} \lambda_j L(u_j) \sum_{jj=1}^{k} \lambda_{jj}\, L(u_{jj})\Big].$$

Ved å introdusere $\lambda_j L(u_j)$ i den andre summasjonen og bytte om summasjon og forventning, får vi

$$E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] = \sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj} E\Big[L(u_j)L(u_{jj})\Big].$$

Ved korollar 3,

$$E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] = \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,cov(u_j,u_{jj}) + \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,m(u_j)m(u_{jj}).$$

Ved å anvende uttrykket for driften i definisjon 7, får vi

$$\begin{aligned} E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] &= \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,cov(u_j,u_{jj}) \\ &\quad + \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\sum_{i=0}^{n}a_i f^i(u_j)\sum_{ii=0}^{n}a_{ii}f^{ii}(u_{jj}). \end{aligned}$$

Ved å skille leddene som inneholder j fra leddene som inneholder jj og endre rekkefølgen på summasjonen, får vi

$$\begin{aligned} E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] &= \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,cov(u_j,u_{jj}) + \\ &\quad \sum_{i=0}^{n}a_i\sum_{j=1}^{k}\lambda_j f^i(u_j)\sum_{ii=0}^{n}a_{ii}\sum_{jj=1}^{k}\lambda_{jj}\,f^{ii}(u_{jj}). \end{aligned}$$

Fra teorem 3 har vi at $\sum_{j=1}^{k}\lambda_j f^i(u_j) = f^i(u_0)$. Dette innført, gir

$$E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] = \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,cov(u_j,u_{jj}) + \sum_{i=0}^{n}a_i f^i(u_0)\sum_{ii=0}^{n}a_{ii}f^{ii}(u_0).$$

Ved å benytte uttrykket for driften i definisjon 7, går de to siste leddene over til $\{m(u_0)\}^2$. Dette gir oss

$$E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] = \sum_{j=1}^{k}\sum_{jj=1}^{k}\lambda_j\lambda_{jj}\,cov(u_j,u_{jj}) + \{m(u_0)\}^2.$$

□

Korollar 5 *Anta en regionalisert variabel $L(\cdot)$ med residualene $l(\cdot)$. La $m(\cdot)$ være driften for den regionaliserte variable og la $cov_l(u_j,u_{jj})$ være kovariansfunksjonen for residualene. Da gjelder*

$$E\Big[\{L(u_0)\}^2\Big] = cov(u_0,u_0) + \{m(u_0)\}^2.$$

Bevis: Fra definisjon 4

$$E\Big[\{L(u_0)\}^2\Big] = E\Big[\Big\{l(u_0)+m(u_0)\Big\}^2\Big].$$

Ved å multiplisere ut og ta forventningen til hvert av leddene, får vi

$$E\Big[\{L(u_0)\}^2\Big] = E\Big[l(u_0)l(u_0)\Big] + 2E\Big[l(u_0)m(u_0)\Big] + E\Big[\{m(u_0)\}^2\Big],$$

men siden forventningen til driften er en konstant, kan $m(u_0)$ settes utenfor forventningsklammen. Dette gir

$$E\Big[\{L(u_0)\}^2\Big] = E\Big[l(u_0)l(u_0)\Big] + 2m(u_0)E\Big[l(u_0)\Big] + \{m(u_0)\}^2.$$

Siden vi ifølge korollar 1 har vist at forventningen til residualene er lik null, blir det andre leddet lik null og det første leddet blir kovariansen til residualene. Dette gir oss

$$E\Big[\{L(u_0)\}^2\Big] = cov_l(u_0, u_0) + \{m(u_0)\}^2.$$

□

Korollar 6 *Anta en regionalisert variabel $L(\cdot)$. La $m(u_0)$ være driften i u_0 og la estimatet for $L(\cdot)$ i u_0 være gitt ved definisjon 5. Kovariansfunksjonen for residualene $l(\cdot)$ eksisterer og benevnes $cov_l(u_j, u_{jj})$. Da gjelder*

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \, cov(u_j, u_0) + \{m(u_0)\}^2.$$

Bevis: Ved å erstatte $\hat{L}(u_0)$ med definisjon 5 og $L(u_0)$ med definisjon 4, får vi

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = E\Bigg[\sum_{j=1}^{k} \lambda_j L(u_j)\Big\{l(u_0) + m(u_0)\Big\}\Bigg].$$

Ved å erstatte $L(u_j)$ med definisjon 4, får vi

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = E\Bigg[\Big\{\sum_{j=1}^{k} \lambda_j\Big(l(u_j) + m(u_j)\Big)\Big\}\Big\{l(u_0) + m(u_0)\Big\}\Bigg],$$

som ved utmultiplikasjon går over til

$$\begin{aligned} E\Big[\hat{L}(u_0)L(u_0)\Big] \;=\; & E\Bigg[\sum_{j=1}^{k} \lambda_j\Big\{ l(u_j)l(u_0) \\ & + m(u_0)l(u_j) + m(u_j)l(u_0) + m(u_0)m(u_j)\Big\}\Bigg]. \end{aligned}$$

Vi kan bytte om forventning og summasjon og siden λ_j og $m(u)$ er konstanter som kan settes utenfor forventningsklammen, får vi

$$\begin{aligned} E\Big[\hat{L}(u_0)L(u_0)\Big] \;=\; & \sum_{j=1}^{k} \lambda_j\Big\{E\Big[l(u_j)l(u_0)\Big] \\ & + m(u_0)E\Big[l(u_j)\Big] + m(u_j)E\Big[l(u_0)\Big] + m(u_0)m(u_j)\Big\}. \end{aligned}$$

Ved korollar 1 har vi vist at forventningen til residualene er null. Det andre og tredje leddet faller derfor bort. Av samme grunn er $E[l(u_j)l(u_0)] = cov_l(u_j, u_0)$. Dette gir oss

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \Big\{cov_l(u_j, u_0) + m(u_0)m(u_j)\Big\}.$$

Ved å multiplisere ut får vi

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \, cov_l(u_j, u_0) + \sum_{j=1}^{k} \lambda_j \, m(u_0)m(u_j).$$

I det siste leddet kan $m(u_0)$ tas ut av summasjonen og $m(u_j)$ kan erstattes med det analytiske uttrykket for driften gitt i definisjon 7. Dette gir oss

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \, cov_l(u_j, u_0) + m(u_0) \sum_{j=1}^{k} \lambda_j \sum_{i=0}^{n} a_i f^i(u_j).$$

Ved teorem 3 har vi at $\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0)$. Dette innført gir oss

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \, cov_l(u_j, u_0) + m(u_0) \sum_{i=0}^{n} a_i f^i(u_0),$$

men ifølge definisjon 7 er den siste summasjonen lik det analytiske uttrykket for driften i u_0. Vi har derfor

$$E\Big[\hat{L}(u_0)L(u_0)\Big] = \sum_{j=1}^{k} \lambda_j \, cov_l(u_j, u_0) + \{m(u_0)\}^2.$$

□

Vi har nå kommet så langt med utviklingen av det uttrykket vi hele tiden har hatt for øye, nemlig $E[\{\hat{L}(u_0) - L(u_0)\}^2]$, at vi kan sette opp det teorem som vil danne basis for berekningen av de ukjente vektene λ_j i estimatet $\hat{L}(u_0)$.

Teorem 4 *Anta en regionalisert variabel $L(\cdot)$ og la estimatet for $L(\cdot)$ i u_0 være gitt ved definisjon 5. Dersom det eksisterer en kovariansfunksjon $cov_l(u_j, u_{jj})$ for residualene $l(\cdot)$, som vi for enkelhetsskyld vil skrive som $cov(u_j, u_{jj})$, har vi at*

$$\begin{aligned} var\Big(\hat{L}(u_0) - L(u_0)\Big) \;&=\; cov(u_0, u_0) - 2\sum_{j=1}^{k} \lambda_j \, cov(u_j, u_0) \\ &\quad + \sum_{j=1}^{k} \sum_{jj=1}^{k} \lambda_j \lambda_{jj} \, cov(u_j, u_{jj}), \end{aligned} \tag{5.31}$$

hvor $var(\cdot)$ betegner variansen.

Bevis: Gjennom likning 5.29 i vårt optimaliseringskrav i definisjon 6 har vi innført betingelsen om at $E[\hat{L}(u_0) - L(u_0)] = 0$. Dette gjør at variansen til residualene kan uttrykkes ved

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = E\Big[\Big\{\hat{L}(u_0) - L(u_0)\Big\}^2\Big].$$

Ved å multiplisere ut, får vi

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = E\Big[\Big\{\hat{L}(u_0)\Big\}^2\Big] + E\Big[\Big\{L(u_0)\Big\}^2\Big] - 2E\Big[\hat{L}(u_0)L(u_0)\Big\}\Big].$$

De foregående korollar 4,5 og 6 kommer oss nå til nytte. Disse innført gir oss

$$\begin{aligned} var\Big(\hat{L}(u_0) - L(u_0)\Big) &= \sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) + \{m(u_0)\}^2 \\ &\quad + cov(u_0, u_0) + \{m(u_0)\}^2 \\ &\quad - 2\sum_{j=1}^{k} \lambda_j\, cov(u_j, u_0) - 2\{m(u_0)\}^2. \end{aligned}$$

Herav får vi

$$\begin{aligned} var\Big(\hat{L}(u_0) - L(u_0)\Big) &= cov(u_0, u_0) - 2\sum_{j=1}^{k} \lambda_j\, cov(u_j, u_0) \\ &\quad + \sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}). \end{aligned}$$

□

Ved teorem 4 har vi kommet fram til en differensierbar formulering av det opprinnelige optimaliseringskravet i likning 5.30 gitt i definisjon 6. Vi står nå overfor et minimaliseringsproblem med sidevilkår. Sidevilkåret ble opprinnelig formulert ved likning 5.29 gitt i definisjon 6 og videreutviklet ved teorem 3. Den optimale estimator $\hat{L}(\cdot)$ finner vi følgelig ved å minimalisere funksjonen gitt ved teorem 4, likning 5.31 med omsyn på λ_j under sidevilkåret

$$\sum_{j=1}^{k} \lambda_j f^i(u_j) = f^i(u_0) \quad \text{for } i = 0, 1, \ldots, n.$$

Vi benytter Lagrange's multiplikatormetode og setter opp hovedfunksjonen

$$\begin{aligned} H &= cov(u_0, u_0) - 2\sum_{j=1}^{k} \lambda_j\, cov(u_j, u_0) + \sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) \\ &\quad -2\sum_{i=0}^{n} \mu_i \Bigg[\sum_{j=1}^{k} \lambda_j f^i(u_j) - f^i(u_0)\Bigg], \end{aligned}$$

hvor μ_i er Lagrange's multiplikator. De partiellderiverte med omsyn på de ukjente er:

$$\frac{\partial H}{\partial \lambda_j} = -2cov(u_j, u_0) + 2\sum_{jj=1}^{k} \lambda_{jj}\, cov(u_j, u_{jj}) - 2\sum_{i=0}^{n} \mu_i f^i(u_j), \tag{5.32}$$

for $j = 1, 2, \cdots, k$. Løsningen for de ukjente k λ-ene og $(n+1)$ μ-ene vil komme fra følgende likningssystem:

$$\begin{aligned}
\sum_{jj=1}^{k} \lambda_{jj}\, cov(u_1, u_{jj}) - \sum_{i=0}^{n} \mu_i f^i(u_1) &= cov(u_1, u_0), \\
\sum_{jj=1}^{k} \lambda_{jj}\, cov(u_2, u_{jj}) - \sum_{i=0}^{n} \mu_i f^i(u_2) &= cov(u_2, u_0), \\
\cdots\cdots\cdots\cdots\cdots\cdots & \quad \cdots\cdots \\
\sum_{jj=1}^{k} \lambda_{jj}\, cov(u_k, u_{jj}) - \sum_{i=0}^{n} \mu_i f^i(u_k) &= cov(u_k, u_0), \\
\sum_{j=1}^{k} \lambda_j f^0(u_j) &= f^0(u_0), \\
\sum_{j=1}^{k} \lambda_j f^1(u_j) &= f^1(u_0), \\
\cdots\cdots\cdots\cdots\cdots\cdots & \quad \cdots\cdots \\
\sum_{j=1}^{k} \lambda_j f^n(u_j) &= f^n(u_0).
\end{aligned}$$

I mange tilfeller vil vi velge å modellerere driften ved de første leddene i det bikubiske polynom. Leddet $f^0(u)$ vil derfor anta verdien 1. Vi innfører denne forutsetningen og skriver likningssystemet på matriseform

$$\mathbf{AU} = \mathbf{E} \tag{5.33}$$

hvor matrisen $\mathbf{A}$ er gitt ved

$$\begin{pmatrix}
cov(u_1,u_1) & cov(u_1,u_2) & \cdots & cov(u_1,u_k) & 1 & f^1(u_1) & f^2(u_1) & \cdots & f^n(u_1) \\
cov(u_2,u_1) & cov(u_2,u_2) & \cdots & cov(u_2,u_k) & 1 & f^1(u_2) & f^2(u_2) & \cdots & f^n(u_2) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
cov(u_k,u_1) & cov(u_k,u_2) & \cdots & cov(u_k,u_k) & 1 & f^1(u_k) & f^2(u_k) & \cdots & f^n(u_k) \\
1 & 1 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 \\
f^1(u_1) & f^1(u_2) & \cdots & f^1(u_k) & 0 & 0 & 0 & \cdots & 0 \\
f^2(u_1) & f^2(u_2) & \cdots & f^2(u_k) & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
f^n(u_1) & f^n(u_2) & \cdots & f^n(u_k) & 0 & 0 & 0 & \cdots & 0
\end{pmatrix}$$

og

$$\mathbf{U} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_k \\ -\mu_0 \\ -\mu_1 \\ \vdots \\ -\mu_n \end{pmatrix} \qquad \mathbf{E} = \begin{pmatrix} cov(u_1, u_0) \\ cov(u_2, u_0) \\ \vdots \\ cov(u_k, u_0) \\ 1 \\ f^1(x_0) \\ f^2(x_0) \\ \vdots \\ f^n(x_0) \end{pmatrix}$$

hvor $f^0(u) = 1$. Vi ser at likning 5.33 har samme struktur som likningene fra enkel kriging, med unntak av at likningen fra universell kriging er noe utvidet i forhold til likningen fra enkel kriging.

Nøyaktigheten til det krigede punktet kan finnes med utgangspunkt i neste teorem.

Teorem 5 *La $L(\hat{u}_0)$ være den optimale estimator for en regionalisert variabel i punktet u_0, og la den regionaliserte variable ha kovariansen $cov(u_j, u_{jj})$ for sine residualer. Da gjelder*

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = cov(u_0, u_0) - \mathbf{U}^t\mathbf{E}. \tag{5.34}$$

Bevis: Siden λ_j må tilfredsstille $\frac{\partial H}{\partial \lambda_j} = 0$ i likning 5.32, har vi

$$\sum_{jj=1}^{k} \lambda_{jj}\, cov(u_j, u_{jj}) = cov(u_j, u_0) + \sum_{i=0}^{n} \mu_i f^i(u_j),$$

for $j = 1, 2, \cdots, k$. Vi kan multiplisere begge sider av likningen med λ_j og summere de k-likningene. Dette gir oss

$$\sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) = \sum_{j=1}^{k} \lambda_j cov(u_j, u_0) + \sum_{j=1}^{k} \lambda_j \sum_{i=0}^{n} \mu_i f^i(u_j).$$

I det siste leddet kan vi introdusere λ_j i den andre summasjonen, snu rekkefølgen i summasjonen og ta μ_j ut av den andre summasjonen:

$$\sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) = \sum_{j=1}^{k} \lambda_j cov(u_j, u_0) + \sum_{i=0}^{n} \mu_i \sum_{j=1}^{k} \lambda_j f^i(u_j),$$

men fra teorem 3 har vi at den andre summasjonen kan erstattes med $f^i(u_0)$. Dette gir

$$\sum_{j=1}^{k}\sum_{jj=1}^{k} \lambda_j\lambda_{jj}\, cov(u_j, u_{jj}) = \sum_{j=1}^{k} \lambda_j cov(u_j, u_0) + \sum_{i=0}^{n} \mu_i f^i(u_0),$$

men fra teorem 4, likning 5.31 finner vi et uttrykk vi kan føre inn i den venstre side av likningen

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) \quad - \quad cov(u_0,u_0) + 2\sum_{j=1}^{k}\lambda_j\, cov(u_j,u_0) = \sum_{j=1}^{k}\lambda_j cov(u_j,u_0) + \sum_{i=0}^{n}\mu_i f^i(u_0),$$

som etter sammentrekking gir

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = cov(u_0,u_0) - \sum_{j=1}^{k}\lambda_j cov(u_j,u_0) + \sum_{i=0}^{n}\mu_i f^i(u_0).$$

Ved å anvende definisjonen av matrisen **U** og **E**, får vi

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = cov(u_0,u_0) - \mathbf{U}^t\mathbf{E}.$$

□

Som vi ser har likning 5.34 samme form som likning 5.28 fra enkel kriging.

Det lar seg vise at under forutsetning av at den regionaliserte variable statistisk sett er et homogent felt, vil i likning 5.33 kovariansene kunne byttes ut med semivarianser, se for eksempel [Ole75] side 73. Kriginglikningene kan derfor skrives som

$$\mathbf{CV} = \mathbf{F} \tag{5.35}$$

hvor

$$\mathbf{C} = \begin{pmatrix} \gamma(u_1,u_1) & \gamma(u_1,u_2) & \cdots & \gamma(u_1,u_k) & 1 & f^1(u_1) & f^2(u_1) & \cdots & f^n(u_1) \\ \gamma(u_2,u_1) & \gamma(u_2,u_2) & \cdots & \gamma(u_2,u_k) & 1 & f^1(u_2) & f^2(u_2) & \cdots & f^n(u_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \gamma(u_k,u_1) & \gamma(u_k,u_2) & \cdots & \gamma(u_k,u_k) & 1 & f^1(u_k) & f^2(u_k) & \cdots & f^n(u_k) \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 \\ f^1(u_1) & f^1(u_2) & \cdots & f^1(u_k) & 0 & 0 & 0 & \cdots & 0 \\ f^2(u_1) & f^2(u_2) & \cdots & f^2(u_k) & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f^n(u_1) & f^n(u_2) & \cdots & f^n(u_k) & 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

og

$$\mathbf{V} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_k \\ \mu_0 \\ \mu_1 \\ \vdots \\ \mu_n \end{pmatrix} \qquad \mathbf{F} = \begin{pmatrix} \gamma(u_1, u_0) \\ \gamma(u_2, u_0) \\ \vdots \\ \gamma(u_k, u_0) \\ 1 \\ f^1(x_0) \\ f^2(x_0) \\ \vdots \\ f^n(x_0) \end{pmatrix}$$

hvor $f^0(u) = 1$.

Det lar seg vise, [Ole75] side 74, at nøyaktigheten til den estimerte verdien til den regionaliserte variable kan finnes med utgangspunkt i

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = \mathbf{V}^t\mathbf{F}. \tag{5.36}$$

5.5.2 Algoritme for universell kriging

Vi antar at vi har skaffet oss autokorrelasjonen eller semivariogrammet for residualene $l(\cdot)$ og at vi har valgt et analytisk uttrykk for driften $m(\cdot)$, for eksempel

$$m(u) = a_0 + a_1x + a_2y + a_3xy + a_4x^2y + a_5y^2x + a_6x^2y^2,$$

hvor vi antar at $u = (x, y)$ er en posisjon i planet $\mathbb{R}^2$.

Kovariansfunksjonen er kjent

Vi får følgende algoritme i det tilfellet at kovariansfunksjonen er kjent:

1. Definer korrelasjonsfunksjonen for residualene $l(\cdot)$ på grunnlag av de gitte datapunkter.
2. Definer matrisene $\mathbf{A}$ og $\mathbf{E}$ med utgangspunkt i korrelasjonsfunksjonen og det analytiske uttrykket for driften.
3. Beregn løsningsvektoren $\mathbf{U}$ i likningssystemt $\mathbf{AU} = \mathbf{E}$.
4. Beregn estimert høyde i u_0 ved

   $$\hat{L}(u_0) = \sum_{j=1}^{k} \lambda_j L(u_j),$$

   hvor u_j er posisjonen til k datapunkter i et område (u_0, r) rundt det punktet høyden skal beregnes for.
5. Beregn variansen i det interpolerte punktet ved

   $$var\Big(\hat{L}(u_0) - L(u_0)\Big) = cov(u_0, u_0) - \mathbf{U}^t\mathbf{E}.$$

Semivariogrammet er kjent

Dersom semivariogrammet er kjent, får vi en liknende algoritme som i det tilfellet at kovariansfunksjonen er kjent.

1. Definer semivariogrammet for residualene $l(\cdot)$ på grunnlag av de gitte datapunkter.
2. Definer matrisene $\mathbf{C}$ og $\mathbf{F}$ med utgangspunkt i semivariogrammet og det analytiske uttrykket for driften.
3. Beregn løsningsvektoren $\mathbf{V}$ i likningssystemt $\mathbf{CV} = \mathbf{F}$.
4. Beregn estimert høyde i u_0 ved

$$\hat{L}(u_0) = \sum_{j=1}^{k} \lambda_j L(u_j),$$

   hvor u_j er posisjonen til k datapunkter i et område (u_0, r) rundt det punktet høyden skal beregnes for.
5. Beregn variansen i det interpolerte punktet ved

$$var\Big(\hat{L}(u_0) - L(u_0)\Big) = \mathbf{V}^t\mathbf{F}.$$

Bestemme krigingnaboskapet

Hvor stort område rundt u_0 vi skal plukke datapunkter fra, kan bestemmes på flere måter. En måte er å analysere vektene λ_j og si at punkter som får tilordnet en liten verdi for λ_j, ligger så langt unna u_0 at deres bidrag til den estimerte høyden kan neglisjeres. En annen betraktningsmåte er å sette r i (u_0, r) lik den avstanden der semivariogrammet begynner å flate ut. Et annet hensyn som må telle ved valg av datapunkter, er at vi bør tilstrebe en jevn spredning av datapunkter i alle sektorer rundt u_0, for eksempel åtte sektorer og beholde de nærmeste punktene i hver sektor. Dette vil hindre at vi havner i situasjoner med ekstrapolasjon.

5.5.3 Bestemme semivariogrammet til residualene

Problemet med universell kriging er å bestemme semivariogrammet, eventuelt autokorrelasjonen, til residualene. Dette krever at vi kjenner driften, men i universell kriging er jo denne å betrakte som ukjent. Et framgangsmåte som kan benyttes, men som har svakheter ved seg, er å estimere driften på grunnlag en minste kvadraters flatetilpassing, for eksempel flytende flate. Dersom vi kan anta at dette gir oss et estimat for $m(\cdot)$, kan vi beregne residualene ved $l(u) = L(u) - m(u)$ og herav sette opp semivariogrammet. Problemet knyttet til bestemmelse av semivariogrammet for residualene, er meget komplekst og vi må her vise til litteraturen, for eksempel [Ole75] og [Cre93].

5.6 Regneeksempler

Den enkleste form for kriging bygger på en modell der vi antar at en eventuell drift (trend) er fjernet fra datamaterialet, uttrykt ved likningen $l(u) = L(u) - m(u)$, hvor $L(u)$ betraktes som en realisasjon av en stokastiske prosessen i punktet u og hvor $m(u)$ er driften i u. Ved universell kriging betraktes driften som ukjent.

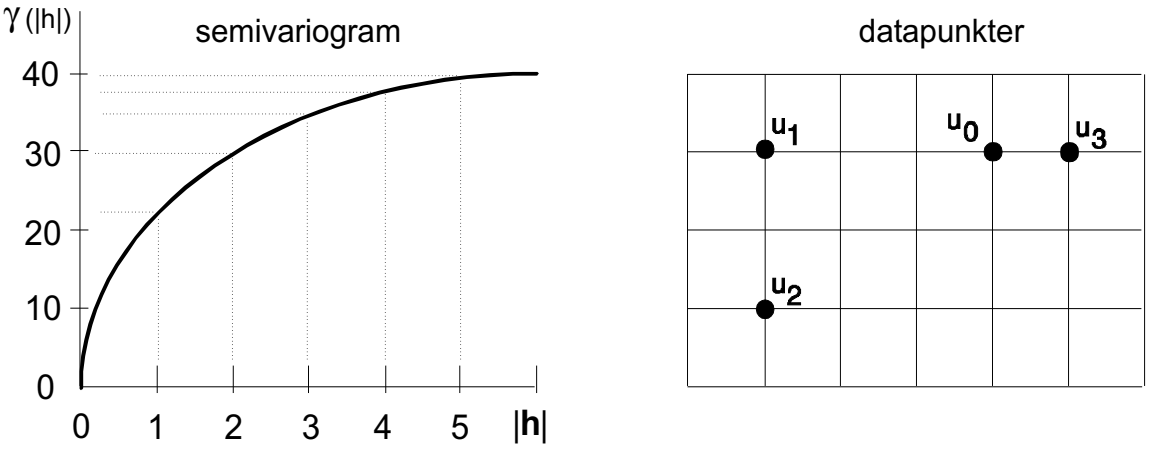
(a) Forklar hva kriging kan benyttes til innenfor terrengmodellering.

(b) Hva forutsetter kriging om forventningsverdien $E[l(\cdot)]$?

(c) Hva kalles funksjonen $\gamma(h)$ gitt ved $2\gamma(h) = E\left[(l(u) - l(u+h))^2\right]$ og hva uttrykker den?

(d) Forklar betydningen av følgende størrelser ved enkel kriging:

$$p(\mathbf{l}; u_0) = \mathbf{a}^t\mathbf{l} \quad \text{hvor } \mathbf{a}^t = [a_1 \;\; a_2 \;\; \cdots \;\; a_n].$$



Figur 5.3: Informasjon for kriging. Semivariansen er gitt i enheten m^2. Målestokken på $|h|$-aksen i semivariogrammet er den samme som målestokken i skissen over datapunkter til høyre i figuren.

(e) Ved enkel kriging kan $\mathbf{a}$ kan finnes av likningen

$$\mathbf{a} = \mathbf{\Sigma}^{-1}\mathbf{g} = \begin{bmatrix} \gamma(u_1-u_1) & \gamma(u_1-u_2) & \cdots & \gamma(u_1-u_n) \\ \gamma(u_2-u_1) & \gamma(u_2-u_2) & \cdots & \gamma(u_2-u_n) \\ \vdots & \vdots & \vdots & \vdots \\ \gamma(u_n-u_1) & \gamma(u_n-u_2) & \cdots & \gamma(u_n-u_n) \end{bmatrix}^{-1} \cdot \begin{bmatrix} \gamma(u_0-u_1) \\ \gamma(u_0-u_2) \\ \vdots \\ \gamma(u_0-u_n) \end{bmatrix}$$

Benytt informasjonen i figur 5.6 til å beregne vektoren $\mathbf{a}$.

(f) Vi antar at driften er beregnet til 23.0m og at måleverdien i punktene u_1, u_2, u_3 er henholdsvis 23.5m, 22.1m og 23.2m. Fyll ut vektoren $\mathbf{l}$ i uttrykket $p(\mathbf{l}; u_0) = \mathbf{a}^t\mathbf{l}$.

(g) Anta enkel kriging og beregn estimert verdi for høyden i u_0 og angi usikkerheten til den estimerte verdien.

(h) Anta universell kriging og at driften modelleres ved det første leddet i det bikubiske polynomet. Beregn estimert verdi for høyden i u_0 og angi usikkerheten til den estimerte verdien.

(i) Anta universell kriging og at driften modelleres ved de tre første leddene i det bikubiske polynomet. Beregn estimert verdi for høyden i u_0 og angi usikkerheten til den estimerte verdien.

(Svar a) Kriging er en interpolasjonsmetode. På grunnlag av gitte høyder i et utvalg av punkter estimeres høyden i mellomliggende punkter. Kriging anses for å være en nøyaktig interpolator. En fordel med kriging framfor andre interpolasjonsmetoder, er at vi kan anslå en verdi for usikkerheten til de interpolerte høydeverdiene.

(Svar b) $E[l(\cdot)] = 0$.

(Svar c) $\gamma(\cdot)$ kalles semivariansen, og den er et mål for graden av samvariasjon i l-verdi (høydeverdi) som funksjon av avstanden $|\ \mathbf{h}|$ mellom to datapunkter. Liten semivarians betyr stor samvariasjon.

(svar d) $p(\mathbf{l}; u_0)$ betyr predikert (estimert, interpolert) verdi i punktet u_0 basert på data gitt ved vektoren $\mathbf{l}$. Den predikerte verdien som settes lik $\mathbf{a}^t\mathbf{l}$, er en lineær kombinasjon av de gitte høydeverdier hvor a-ene er vekter.

(Svar e) Vi avleser avstanden mellom punktene i høyre del av figuren. I kurven til venstre finner vi den korresponderende semivariansen. Dette gir oss

$$\mathbf{a} = \mathbf{\Sigma}^{-1}\mathbf{g} = \begin{bmatrix} 0 & 30 & 38 \\ 30 & 0 & 39 \\ 38 & 39 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 35 \\ 36 \\ 22 \end{bmatrix} = \begin{bmatrix} 0.291 \\ 0.281 \\ 0.699 \end{bmatrix}$$

(Svar f) Vi reduserer måleverdiene med driften og får

$$\mathbf{l} = \begin{bmatrix} 0.5 \\ -0.9 \\ 0.2 \end{bmatrix}$$

(Svar g) Vi finner estimert verdi for $l(\cdot)$ i u_0 ved

$$\hat{l}(u_0) = \mathbf{a}^t\,\mathbf{l} = \begin{bmatrix} 0.291 & 0.281 & 0.699 \end{bmatrix} \begin{bmatrix} 0.5 \\ -0.9 \\ 0.2 \end{bmatrix} = 0.326.$$

Høyden i det interpolerte punktet finnes nå ved

$$\hat{L}(u_0) = \hat{l}(u_0) + m(u_0) = 0.326 + 23.0 = 23.3m.$$

Dersom vi ikke hadde tatt omsyn til driften, men beregnet estimert verdi direkte mot $L(u)$, ville vi fått verdien 29.3m, som sannsynligvis er et dårlig estimat. Usikkerheten i det estimerte punktet finnes ved likning 5.36, som i vår notasjon blir å skrive som

$$var\Big(\hat{l}(u_0) - l(u_0)\Big) = \mathbf{a}^t\mathbf{g} = 35.7m^2,$$

som gir et standardavvik i det interpolerte punktet på $\pm 6.0m$.

(Svar h) Vi baserer oss på likning 5.35 som gir følgende likningssystem som må løses

$$\mathbf{CV} = \mathbf{F}.$$

Av forutsetningen blir trenden å modellere ved $a_0 f^0(u) = a_0 \cdot 1$. Dette gir oss

$$\mathbf{V} = \mathbf{C}^{-1}\mathbf{F} = \begin{bmatrix} 0 & 30 & 38 & 1 \\ 30 & 0 & 39 & 1 \\ 38 & 39 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 35 \\ 36 \\ 22 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.208 \\ 0.195 \\ 0.597 \\ 6.478 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \mu_0 \end{bmatrix}$$

Som vi ser er summen av λ_j=1.0, som er et resultat av kravet om forventningsrett estimator. Estimert høyde finnes nå ved

$$\hat{L}(u_0) = \sum_{j=1}^{3} \lambda_j L(u_j) = \begin{bmatrix} 0.208 & 0.195 & 0.597 \end{bmatrix} \begin{bmatrix} 23.5 \\ 22.1 \\ 23.2 \end{bmatrix} = 23.0m$$

Variansen finnes ved likning 5.36

$$\begin{aligned} var\Big(\hat{L}(u_0) - L(u_0)\Big) &= \mathbf{V}^t\mathbf{F} \\ &= \begin{bmatrix} 0.208 & 0.195 & 0.597 & 6.478 \end{bmatrix} \begin{bmatrix} 35 \\ 36 \\ 22 \\ 1 \end{bmatrix} \\ &= 33.9m^2, \end{aligned}$$

som gir standardavviket $\pm 5.8m$.

(Svar i) Løsningen på oppgaven blir tilsvarende som under foregående spørsmål. Av forutsetningen blir trenden nå å modellere ved $a_0 + a_1 x + a_2 y$. Koordinatene til punktene avleses i figuren til $(x_0, y_0) = (3, 4)$, $(x_1, y_1) = (3, 1)$, $(x_2, y_2) = (1, 1)$ og $(x_3, y_3) = (3, 5)$. Dette gir oss

$$\mathbf{V} = \mathbf{C}^{-1}\mathbf{F} = \begin{bmatrix} 0 & 30 & 38 & 1 & 3 & 1 \\ 30 & 0 & 39 & 1 & 1 & 1 \\ 38 & 39 & 0 & 1 & 3 & 5 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 3 & 1 & 3 & 0 & 0 & 0 \\ 1 & 1 & 5 & 0 & 0 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 35 \\ 36 \\ 22 \\ 1 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 0.25 \\ 0.00 \\ 0.75 \\ -5.88 \\ 3.63 \\ 1.50 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \mu_0 \\ \mu_1 \\ \mu_2 \end{bmatrix}$$

Estimert høyde finnes ved

$$\hat{L}(u_0) = \sum_{j=1}^{3} \lambda_j L(u_j) = \begin{bmatrix} 0.25 & 0.00 & 0.75 \end{bmatrix} \begin{bmatrix} 23.5 \\ 22.1 \\ 23.2 \end{bmatrix} = 23.3m.$$

Variansen finnes ved likning 5.36

$$\begin{aligned} var\Big(\hat{L}(u_0) - L(u_0)\Big) &= \mathbf{V}^t\mathbf{F} \\ &= \begin{bmatrix} 0.25 & 0.00 & 0.75 & -5.88 & 3.63 & 1.50 \end{bmatrix} \begin{bmatrix} 35 \\ 36 \\ 22 \\ 1 \\ 3 \\ 4 \end{bmatrix} \\ &= 36.3m^2, \end{aligned}$$

som gir standardavviket $\pm 6.0m$.

Del III

Visualisering av terrengoverflater

Kapittel 6

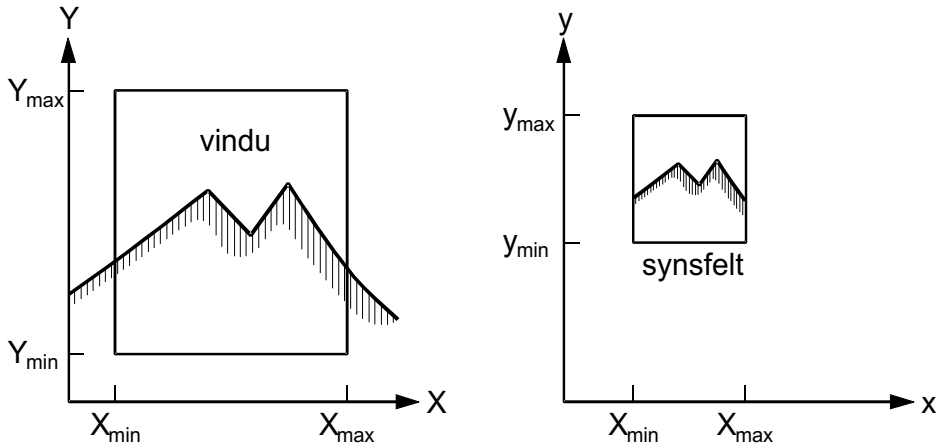
2D-grafikk

Framstillingen i dette kapitlet bygger blant annet på lærebøker i grafisk databehandling av Bøe og Hov Moen [BM79], Newmann og Sproull [NS84] samt Hearn og Baker [HB86].

6.1 Vindusteknikk

Innen grafisk databehandling brukes begrepene *vindu* (eng. window) og *synsfelt* (eng. viewport). Med vindu mener vi det utsnittet av objektet vi ønsker å betrakte, og med synsfelt forstår vi det området på visningsmediet vinduet transformeres til. Vi vil i det etterfølgende benevne vindus-koordinatene med (X, Y) og synsfelt-koordinatene med (x, y).

I figur 6.1 transformeres et punkt (X,Y) i vinduet til et punkt x, y i synsfeltet. 2D-transformasjon fra vindu til synsfelt gjøres vanligvis ved hjelp av en skalering og



Figur 6.1: Transformasjon fra vindu til synsfelt

en translasjon. Transformasjonen baseres derfor gjerne på likningene

$$\frac{X - X_{min}}{X_{max} - X_{min}} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{6.1}$$

og

$$\frac{Y - Y_{min}}{Y_{max} - Y_{min}} = \frac{y - y_{min}}{y_{max} - y_{min}} \tag{6.2}$$

Vi kan omskrive likningene 6.1 og 6.2 og får eksplisitte uttrykk for vinduskoordinatene

$$x = \frac{x_{max} - x_{min}}{X_{max} - X_{min}}(X - X_{min}) \ + \ x_{min},$$

$$y = \frac{y_{max} - y_{min}}{Y_{max} - Y_{min}}(Y - Y_{min}) \ + \ y_{min}.$$

Vindu-til-synsfelt transformasjonen skrives mere kompakt som

$$x \ = \ S_x(X - X_{min}) \ + \ x_{min}, \tag{6.3}$$

$$y \ = \ S_y(Y - Y_{min}) \ + \ y_{min}. \tag{6.4}$$

Dersom $S_x \neq S_y$, vil bildet få ulik målestokk i de to akseretningene. Videre ser vi at vindu-synsfelt transformasjonen er formulert slik at den kun inneholder lineær skalering og translasjon. For kartografiske formål forutsetter dette at vinduskoordinatene til terrengoverflaten er transformert fra jordkula til en kartprojeksjon.

Gografiske informasjonssystem inneholder gjerne mulighet til å velge mellom ulike kartprojeksjoner. Når koordinatene er lagret i en database i et globalt referansesystem, som for eksempel geografiske koordinater (ϕ, λ), kan utføres en projeksjon til plane koordinater i det øyeblikk opptegningen foretas. Transformasjon fra vindu til synsfelt blir med geografiske koordinater litt mere komplisert enn de enkle formler basert på plane koordinater. Vi må her vise til litteraturen om kartprojeksjoner og geodetiske datum.

6.2 Klipping

For å kunne avbilde et objekt eller et terrengområde i en bestemt målestokk innenfor et synsfelt med en begrenset størrelse, blir det som regel nødvendig å foreta en klipping. Det er vanlig å gi synsfeltet en rektangulær form. Dette har noe å gjøre med den fysiske utformingen til grafisk utstyr (grafiske skjermer og tegnefolier er rektangulære). Dessuten er det fordelaktig rent beregningsmessig å klippe mot et rektangulært synsfelt framfor å klippe mot en polygon med vilkårlig form.

Klippingen utføres ofte mot vinduet. Alternativt kan terrengkoordinatene først transformeres til synsfeltkoordinater og så klippes mot synsfeltets områdebegrensning. Fordelen med å klippe mot vinduet er at vi slipper å beregne synsfeltkoordinater for punkter som blir liggende utenfor synsfeltet. For geografiske informasjonssystem er

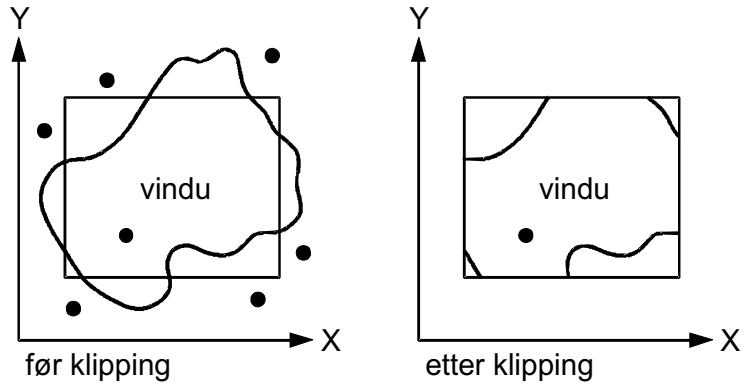
det helt nødvendig å kunne foreta en viss grov-klipping før finjusteringen mot vinduet (eller synsfeltet). Har vi for eksempel en database over hele Norge med informasjon tilsvarende M711 serien (M 1:50.000) og vi definerer et vindu over Trondheim, må vi på en effektiv måte kunne sile ut de data det er aktuelt å beregne klipping for. I den følgende diskusjonen vil vi forutsettes at klippingen utføres i forhold til vinduet og at koordinatene er gitt i en kartprojeksjon.

6.2.1 Punkter

Klipping av punkter mot vinduet kan enkelt baseres på følgende ulikheter

$$X_{min} \leq X \leq X_{max} \qquad \text{og} \qquad Y_{min} \leq Y \leq Y_{max}, \tag{6.5}$$

se illustrasjon i figur 6.2.



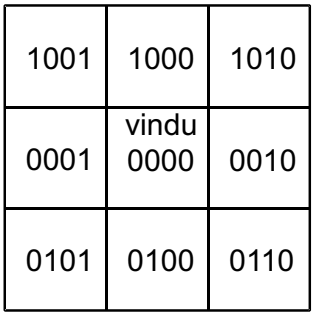
Figur 6.2: Klipping

6.2.2 Linjer

Klipping av linjer mot vinduet er mer ressurskrevende enn klipping av punkter. Dette er fordi skjæringspunkter må beregnes for alle linjer som krysser vinduets ytterbegrensning. For å unngå å regne mulige skjæringspunkter for samtlige linjer, innretter en seg slik at det ikke forsøkes regnet skjæringspunkter for linjer som opplagt ikke kan krysse vinduets ytterbegrensning. En veletablert teknikk på dette området er *Cohen-Sutherland* algoritmen.

Cohen og Sutherland benytter en kode som identifiserer hvilket område et punkt tilhører. Koden framkommer ved å dele planet i 9 felter, som gis binære koder slik som vist i figur 6.3. Hvert felt får tilordnet en 4 biters kode etter følgende strategi

- første bit: punktet ligger til venstre for vinduet,
- andre bit: punktet ligger til høyre for vinduet,



Figur 6.3: Cohen-Sutherland kodingsskjema

- tredje bit: punktet ligger under vinduet,
- fjerde bit: punktet ligger over vinduet.

Følgelig har vi at dersom den 4-biters koden er null for begge endepunktene, ligger linjesegmentet i sin helhet innenfor vinduet.

Ved å uføre en logisk *and* operasjon på begge endepunkter, vil vi kunne avgjøre om linjesegmentet i sin helhet ligger utenfor vinduet. Dersom resultatet av *and* operasjonen ikke er 0000, ligger nemlig linjesegmentet i sin helhet utenfor vinduet. Operasjonen *and* sammenligner to variable bit for bit. Dersom bit i har verdien 1 i begge variable, blir resultatet 1. Ellers blir resultatet 0. For eksempel har vi at

$$and(1010, 0101) = 0000 \qquad \text{og} \qquad and(1010, 0110) = 0010.$$

I det første tilfellet ligger linjens endepunkter i øvre-høyre hjørne og nedre-venstre hjørne. Følgelig vil linjen kunne krysse vinduet. I det andre tilfellet ligger begge endepunkter til høyre for vinduet og kryssing av vinduet er utelukket.

Linjer som ikke blir eliminert av disse to testene, må vi beregne skjæringspunkter for. Det er flere måter skjæringspunktet kan beregnes på. Vi kan for eksempel benytte den tidligere omtalte punkt-vektor metoden, homogene koordinater eller likningene

$$y = ax + b \qquad \text{eller} \qquad x = cy + d.$$

I det siste tilfellet velges den ene eller den andre av de to likningene avhengig av den aktuelle vinduskantens orientering i forhold til aksesystemet.

Det finnes også en iterativ metode som kan være aktuell, den såkalte *midtpunkt-metoden*. Først finnes linjesegmentets midtpunkt. Deretter beregnes avstanden fra midtpunktet til vinduets sidekanter. Dersom en avstand er mindre enn en gitt toleranse, er et krysningspunkt funnet. I motsatt fall gjentas prosedyren med den halvdelen av linjesegmentet som krysser vinduet (i noen tilfeller kan dette gjelde begge halvdelene). Prosedyren terminerer når skjæringspunktet er funnet med tilstrekkelig nøyaktighet.

Kapittel 7

3D-grafikk

Framstillingen i dette kapitlet bygger blant annet på lærebøker i grafisk databehandling av Bøe og Hov Moen [BM79], Newmann og Sproull [NS84], Hearn og Baker [HB86] samt Rankin [Ran89].

7.1 Sentralperspektivisk avbildning

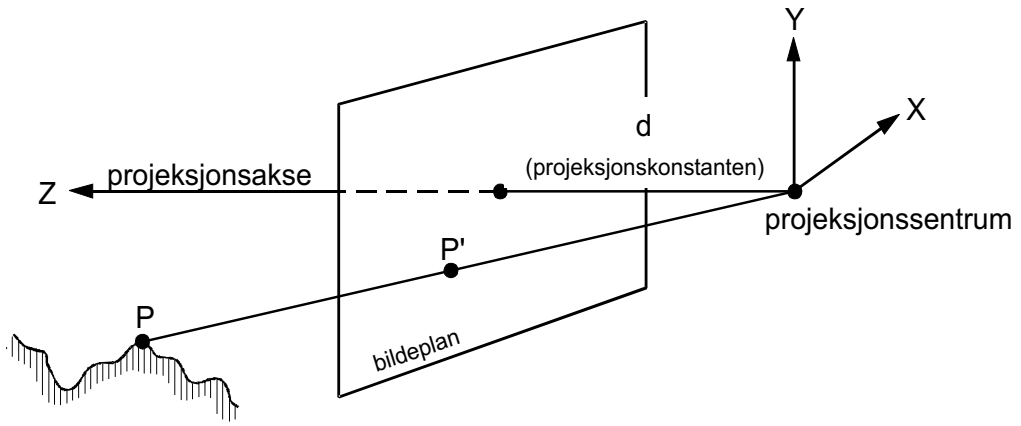
Den sentralperspektiviske projeksjonen har begynt å få en økende utbredelse innen geografiske informasjonssystemer. Dette skyldes blant annet at terrengformer kan være lettere å forstå i et perspektivbilde enn i et kart. Vi skal likevel være klar over at perspektivbilder har noen ulemper i forhold til kartprojeksjoner. Blant annet vil målestokken i et perspektivbilde variere i bildet og oppstikkende objekter kan skjule objekter som ligger bak i senen. Dessuten er det vanskelig å sammenligne høydedifferanser i et perspektivbilde.

7.1.1 Grunnbegreper om den perspektiviske transformasjonen

Vi finner det praktisk å velge et venstrehåndssystem som koordinatsystem. Sentralprojeksjonen krever at vi velger et *projeksjonssentrum* og et *bildeplan*. Normalen fra projeksjonssentret til bildeplanet kalles *projeksjonsaksen*. Som vi ser av figur 7.1 legges origo i projeksjonssentret og z-aksen langs projeksjonsaksen. Følgelig vil bildeplanet være parallelt med xy-planet. Avstanden fra projeksjonssentret til bildeplanet er den størrelse som tilsvarer brennvidden (kamerakonstanten) til et fotografiapparat. Denne størrelsen vil vi kalle *projeksjonskonstanten* og benevne den d.

Av figur 7.1 finner vi at linjen mellom projeksjonssentret og punkt P skjærer bildeplanet i punkt P'. Koordinatene til P' finner vi ved å betrakte likedannede trekanter. Dette gir oss

$$x' = x\frac{d}{z} \tag{7.1}$$



Figur 7.1: Sentralperspektivisk avbildning

$$y' = y\frac{d}{z} \tag{7.2}$$

$$z' = d. \tag{7.3}$$

Ved å benytte homogene koordinater kan P' sine koordinater skrives som

$$(x_h, y_h, z_h, w) \qquad \text{hvor} \quad w = \frac{z}{d}$$

Den perspektiviske transformasjonen på matriseform blir da

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.4}$$

Herav finnes bildekoordinatene ved å dividere med w

$$x' = \frac{x_h}{w} \qquad y' = \frac{y_h}{w} \qquad z' = \frac{z_h}{w} = d.$$

Siden matrisen $\mathbf{P}$ kun er avhengig av projeksjonskonstanten d, vil $\mathbf{P}$ derfor kunne utledes en gang for alle når d er gitt. Ved å benytte homogene koordinater har vi klart å beskrive den perspektiviske avbildningen ved hjelp av en matrisemultiplikasjon. Dette gjør at den perspektiviske transformasjonen kan kjedes sammen med 3D-rotasjoner, translasjoner og skaleringer.

Målestokken til bildet kan endres ved å velge en ny verdi for projeksjonskonstanten eller ved å flytte projeksjonssentret, men de to metodene er ikke ekvivalente. La oss anta at et objekt skal avbildes i en bestemt målestokk. Dersom d gjøres mindre, må projeksjonssentret flyttes nærmere objektet for at bildemålestokken fortsatt skal bli den samme. Konsekvensen av dette er at projeksjonsstrålene mot objektet vil danne større vinkler med projeksjonsaksen. Jo lengre unna objektet projeksjonssentret flyttes, jo større må d gjøres for at bildemålestokken skal holde seg konstant.

Følgelig vil projeksjonsstrålene nærme seg å bli parallelle når avstanden til objektet går mot uendelig.

Den sentralperspektiviske projeksjon kan oppfattes som en skaleringstransformasjon der skaleringsfaktoren er lineært avhengig av dybden (z-verdien) til punktet som skal transformeres. Objekter som befinner seg i et plan vinkelrett på projeksjonsaksen, vil derfor avbildes i konstant målestokk i bildeplanet. Bortsett fra dette spesialtilfellet vil vi ha at målestokken i bildeplanet er variabel. Dette er en ulempe ved projeksjonen, siden det blir noe vanskelig for en observatør å bedømme størrelsesforholdet mellom ulike objekter. Objekter som ligger nære projeksjonssentret vil avbildes i større målestokk enn objekter som ligger legger unna projeksjonssentret. Som vi vet er et av sportsfiskerens triks å holde fisken på strak arm rettet mot fotografen. Fiskens z-koordinat vil da minimeres i forhold til z-koordinaten til personen som holder fangsten, og følgelig vil bildet gi inntrykk av at fisken er større enn den i virkeligheten er.

7.1.2 Perspektiv med vilkårlig ytre orientering

Vi skal vise hvilke transformasjoner som må utføres for å finne den perspektiviske avbildningen fra et vilkårlig projeksjonssenter med en vilkårlig orientering av projeksjonsaksen; se figur 7.2. Som vist i figur 7.2 legges projeksjonssentret i punkt Q og projeksjonsaksens retning defineres ved å forlange at den skal peke mot punkt N. Dette fastlegger hva vi kan kalle projeksjonens *ytre orientering*. Den *indre orientering* defineres ved projeksjonskonstanten.

Det viser seg praktisk å operere med følgende tre koordinatsystemer:

Terrengkoordinatsystemet defineres som et høyrehåndssystem der aksene betegnes X, Y, Z.

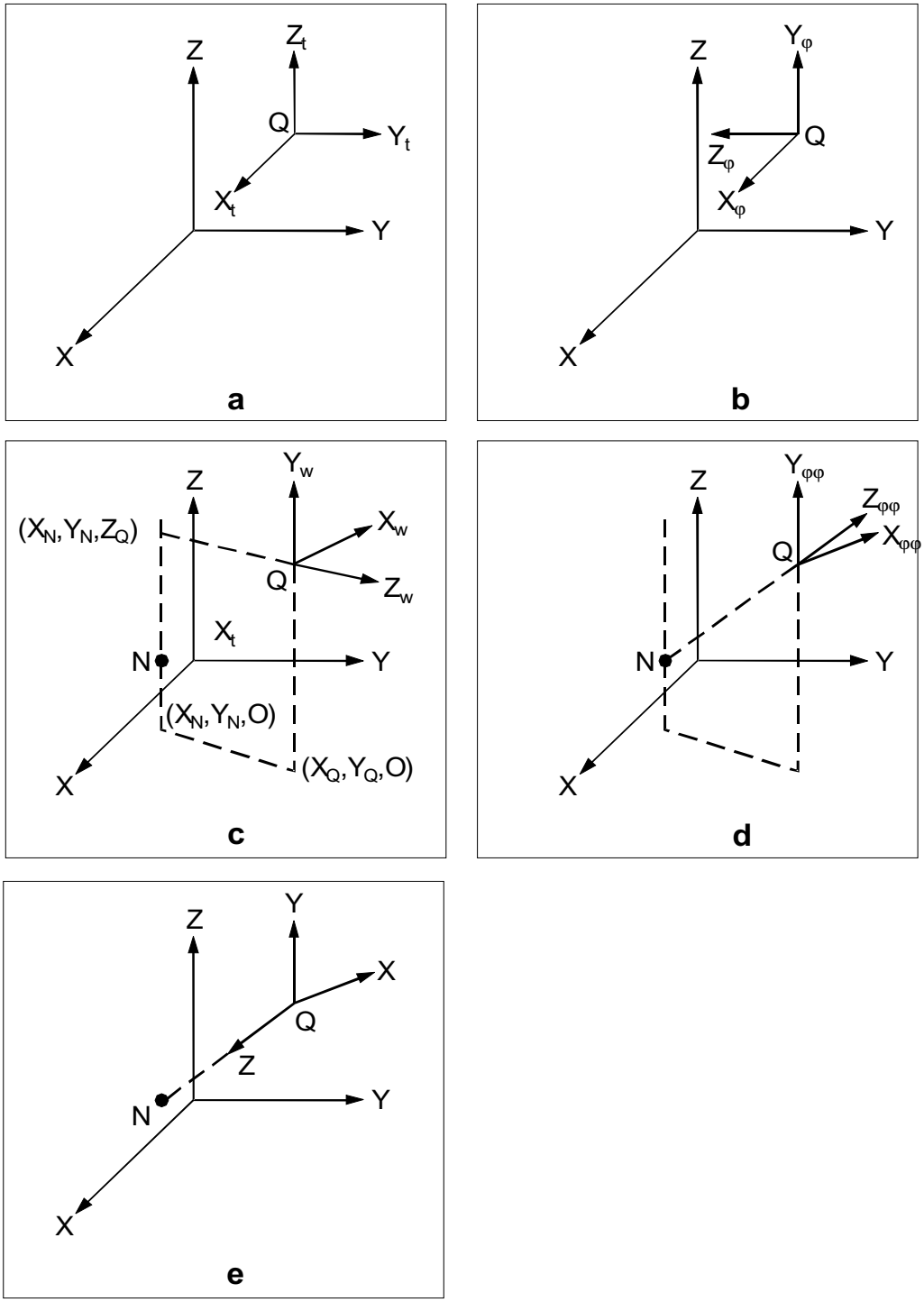
Modellkoordinatsystemet defineres som et venstrehåndssystem der aksene betegnes x, y, z med origo i projeksjonssentret.

Bildekoordinatsystemet vil vi også oppfatte som 3D siden alle punkter vil få tilordnet en z-verdi. Som vi senere skal se, vil z-verdien riktignok bli lik for samtlige punkter. Dette er grunnen til at bildekoordinatsystemet ofte betraktes som et 2D-koordinatsystem. Bildekoordinater vil vi angi med x', y', z'.

Formlene for den perspektiviske avbildningen forutsetter at z-aksen til modellkoordinatsystemet er sammenfallende med projeksjonsaksen. Vi må derfor finne koordinatene til objektet i modellkoordinatsystemet. Dette løses ved å utføre en serie transformasjoner.

Translasjon til projeksjonssentret

Vi starter med et koordinatsystem som er sammenfallende med aksene til terrengkoordinatsystemet. Koordinatsystemet parallellforskyves så til projeksjonssentret, gitt



Figur 7.2: Prinsipp for å finne bildekoordinatene til en sentralperspektivisk avbildning med vilkårlig ytre orientering. a) Parallellforskyv koordinatsystemet til punkt Q, som er det valgte projeksjonssentrum. b) Roter om x-aksen. c) Roter om y-aksen. d) Roter på nytt om x-aksen slik at z-aksen peker mot punkt N. e) Snu z-aksens retning.

ved translasjonen

$$\begin{bmatrix} x_t \\ y_t \\ z_t \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -X_Q \\ 0 & 1 & 0 & -Y_Q \\ 0 & 0 & 1 & -Z_Q \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{T}_1 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{7.5}$$

Roter deretter om x-aksen

Vi roterer deretter modellkoordinatsystemet om $x-$aksen slik at $y-$aksen peker i samme retning som terrengkoordinatsystemet's $Z-$akse. Dette er en φ-rotasjon som ifølge likning 1.3 kan skrives som

$$\begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a & -b \\ 0 & b & a \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}$$

Til bestemmelse av φ velger vi et punkt med koordinatene

$$(x_t, y_t, z_t) = (0, 0, z_1).$$

I det roterte koordinatsystemet er dette punktets koordinater

$$(x_\varphi, y_\varphi, z_\varphi) = (0, z_1, 0).$$

Dette innført gir oss

$$\begin{bmatrix} 0 \\ z_1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a & -b \\ 0 & b & a \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ z_1 \end{bmatrix}$$

som har løsningen

$$a = 0 \qquad \text{og} \qquad b = -1.$$

Vi innfører homogene koordinater og skriver rotasjonsmatrisen som

$$\begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{T}_2 \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.6}$$

Roter deretter om y-aksen

Neste gang roteres modellkoordinatsystemet om $y-$aksen slik at den negative $z-$aksen går gjennom punktet (X_N, Y_N, Z_Q). Dette er en ω-rotasjon som ifølge likning 1.4 er gitt ved

$$\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix} = \begin{bmatrix} a & 0 & b \\ 0 & 1 & 0 \\ -b & 0 & a \end{bmatrix} \begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \end{bmatrix}$$

Til bestemmelse av ω har vi at

$$\begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \end{bmatrix} = \mathbf{T}_2\mathbf{T}_1 \begin{bmatrix} X_N \\ Y_N \\ Z_Q \end{bmatrix} = \begin{bmatrix} X_N - X_Q \\ 0 \\ -(Y_N - Y_Q) \end{bmatrix}$$

I det roterte koordinatsystemet er dette punktets koordinater

$$(x_\omega, y_\omega, z_\omega) = (0, 0, -S_2), \text{ hvor} \quad S_2 = \sqrt{(X_N - X_Q)^2 + (Y_N - Y_Q)^2}.$$

Dette innført gir oss

$$\begin{bmatrix} 0 \\ 0 \\ -S_2 \end{bmatrix} = \begin{bmatrix} a & 0 & b \\ 0 & 1 & 0 \\ -b & 0 & a \end{bmatrix} \begin{bmatrix} X_N - X_Q \\ 0 \\ -(Y_N - Y_Q) \end{bmatrix}$$

som har løsningen

$$a = \frac{Y_N - Y_Q}{S_2} \qquad \text{og} \qquad b = \frac{X_N - X_Q}{S_2} \tag{7.7}$$

Vi benytter homogene koordinater og får følgende rotasjonsmatrise:

$$\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \\ 1 \end{bmatrix} = \begin{bmatrix} a & 0 & b & 0 \\ 0 & 1 & 0 & 0 \\ -b & 0 & a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \\ 1 \end{bmatrix} = \mathbf{T}_3 \begin{bmatrix} x_\varphi \\ y_\varphi \\ z_\varphi \\ 1 \end{bmatrix} \tag{7.8}$$

Roter på nytt om x-aksen

Vi tar så en ny rotasjon om $x-$aksen, men denne gangen slik at slik at den negative $z-$aksen peker mot punkt N. Rotasjonen er en φ-rotasjon som ifølge likning 1.3 er gitt ved

$$\begin{bmatrix} x_{\varphi\varphi} \\ y_{\varphi\varphi} \\ z_{\varphi\varphi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a & -b \\ 0 & b & a \end{bmatrix} \begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix}$$

Til bestemmelse av φ har vi at

$$\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix} = \mathbf{T}_3\mathbf{T}_2\mathbf{T}_1 \begin{bmatrix} X_N \\ Y_N \\ Z_N \end{bmatrix} = \begin{bmatrix} 0 \\ Z_N - Z_Q \\ -S_2 \end{bmatrix}$$

I det roterte koordinatsystemet er dette punktets koordinater

$$(x_{\varphi\varphi}, y_{\varphi\varphi}, z_{\varphi\varphi}) = (0, 0, -S_3),$$

hvor

$$S_3 = \sqrt{(X_N - X_Q)^2 + (Y_N - Y_Q)^2 + (Z_N - Z_Q)^2}.$$

Dette innført gir oss

$$\begin{bmatrix} 0 \\ 0 \\ -S_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a & -b \\ 0 & b & a \end{bmatrix} \begin{bmatrix} 0 \\ Z_N - Z_Q \\ -S_2 \end{bmatrix}$$

som har løsningen

$$a = \frac{S_2}{S_3} \qquad \text{og} \qquad b = -\frac{Z_N - Z_Q}{S_3} \tag{7.9}$$

Vi får da følgende rotasjonsmatrise for de homogene koordinater:

$$\begin{bmatrix} x_{\varphi\varphi} \\ y_{\varphi\varphi} \\ z_{\varphi\varphi} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & a & -b & 0 \\ 0 & b & a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \\ 1 \end{bmatrix} = \mathbf{T}_4 \begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \\ 1 \end{bmatrix} \tag{7.10}$$

Snu z-aksen

Det er praktisk å innrette seg slik at den positive z-aksen peker langs projeksjonsaksen. Dette oppnår vi ved å multiplisere $z_{\varphi\varphi}$ med -1. Denne transformasjonen innebærer at vi går over fra et høyre- til et venstrehåndssystem. Transformasjonen er gitt ved

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\varphi\varphi} \\ y_{\varphi\varphi} \\ z_{\varphi\varphi} \\ 1 \end{bmatrix} = \mathbf{T}_5 \begin{bmatrix} x_{\varphi\varphi} \\ y_{\varphi\varphi} \\ z_{\varphi\varphi} \\ 1 \end{bmatrix} \tag{7.11}$$

Perspektivisk transformasjon

Vi har nå endelig kommet så langt at vi kan utføre den perspektiviske transformasjonen gitt ved likning 7.4.

Sammenkjeding av transformasjonene

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} = \mathbf{P}\mathbf{T}_5\mathbf{T}_4\mathbf{T}_3\mathbf{T}_2\mathbf{T}_1 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{F} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{7.12}$$

Ved å kjede sammen matrisemultiplikasjonene som vist i likning 7.12, kan vi utføre den sentralperspektiviske transformasjonen ved kun å multiplisere terrengkoordinatene med matrisen $\mathbf{F}$. Dette gir oss riktignok de homogene bildekoordinater, slik at

en eventuell overgang til ordinære bildekoordinater må finnes ved å dividere med w. Bildekoordinatene er følgelig

$$x' = \frac{x_h}{w} \qquad y' = \frac{y_h}{w} \qquad z' = \frac{z_h}{w} \tag{7.13}$$

Eksempel

Vi skal gjøre et regneeksempel for å vise at formelverket, som nettopp er utviklet, virker. La oss velge projeksjonssentret i punkt $(6, 8, 7.5)$ og forlange at projeksjonsaksen skal peke mot punkt $(0, 0, 0)$. Videre settes projeksjonskonstanten til 0.10.

Dette gir oss følgende matriser:

$$\mathbf{T}_1 = \begin{bmatrix} 1 & 0 & 0 & -6 \\ 0 & 1 & 0 & -8 \\ 0 & 0 & 1 & -7.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_3 = \begin{bmatrix} -0.8 & 0 & -0.6 & 0 \\ 0 & 1 & 0 & 0 \\ 0.6 & 0 & -0.8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{hvor} \quad \begin{cases} S_2 = \sqrt{36 + 64} = 10 \\ a = (0 - 8)/(10) = -0.8 \\ b = (0 - 6)/(10) = -0.6 \end{cases}$$

$$\mathbf{T}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.8 & -0.6 & 0 \\ 0 & 0.6 & 0.8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{hvor} \quad \begin{cases} S_2 = \sqrt{36 + 64 + 56.25} = 12.5 \\ a = 10/12.5 = 0.8 \\ b = -(0 - 7.5)/(12.5) = 0.6 \end{cases}$$

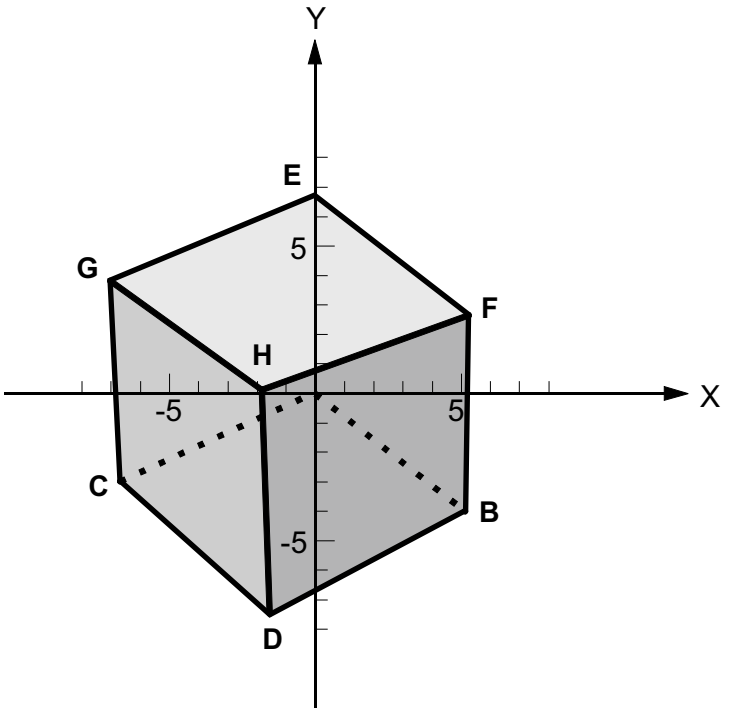
$$\mathbf{T}_5 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 10 & 0 \end{bmatrix}$$

Vi multipliserer sammen matrisene (husk å gjøre dette i riktig rekkefølge) og får

$$\mathbf{F} = \begin{bmatrix} -0.80 & 0.60 & 0 & 0 \\ -0.36 & -0.48 & 0.80 & 0 \\ -0.48 & -0.64 & -0.60 & 12.5 \\ -4.80 & -6.40 & -6.00 & 125.0 \end{bmatrix}$$

Det perspektiviske sambandet mellom terrengkoordinater og bildekoordinater er nå utledet. La oss beregne det perspektiviske bildet av en terning som har sidekant med lengde lik 1. Resultatet av beregningen er vist i figur 7.3. Her har vi satt opp en tabell over terningens terrengkoordinater og bildekoordinater, samt tegnet opp det perspektiviske bildet av terningen på grunnlag av bildekoordinatene. Som vi ser, har bildekoordinatene så små tallverdier at en vindu-synsfelt transformasjon av bildekoordinatene er aktuelt før opptegning på grafisk skjerm.

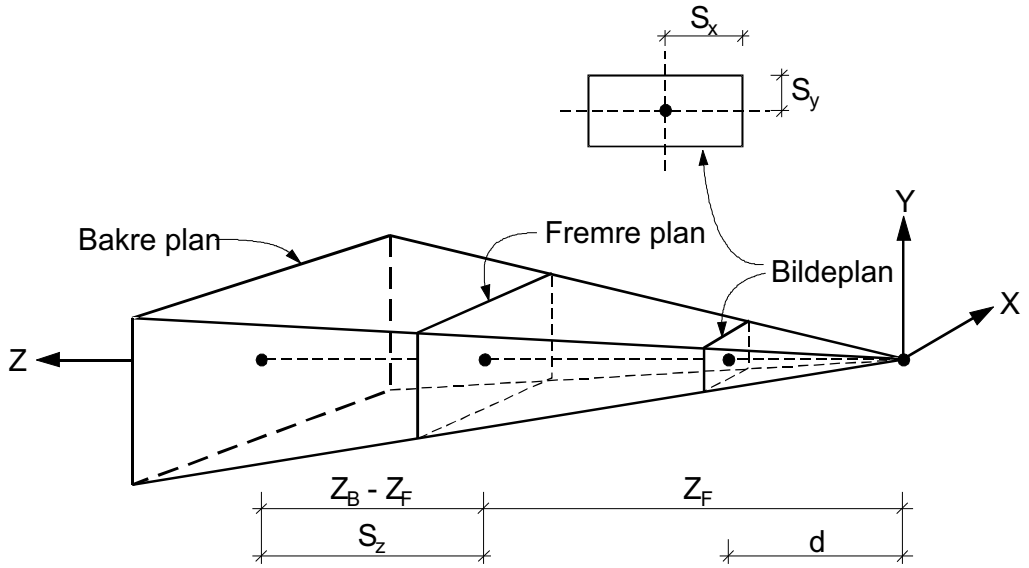
	terreng			bilde		
	X	Y	Z	x'	y'	z'
A	0	0	0	0.0000	0.0000	0.10
B	0	1	0	0.0051	-0.0040	0.10
C	1	0	0	-0.0067	-0.0030	0.10
D	1	1	0	-0.0018	-0.0074	0.10
E	0	0	1	0.0000	0.0067	0.10
F	0	1	1	0.0053	0.0028	0.10
G	1	0	1	-0.0070	0.0039	0.10
H	1	1	1	-0.0019	-0.0004	0.10



Figur 7.3: Eksempel på perspektivisk avbildning av en terning

7.2 Klipping mot synsfeltpyramiden

Prinsippet for 3D-klipping framkommer ved å generalisere prinsippet for 2D-klipping. På grunn av overgangen til 3 dimensjoner, utføres klippingen mot et 3D-synsfelt, som vi heretter vil benevne synsfeltpyramiden. Som det framgår av figur 7.4, defineres synsfeltpyramiden av 6 plan, hvorav 4 plan danner veggene i pyramiden. De to øvrige plan står vinkelrett på projeksjonsaksen og angir intervallet for z-verdiene (dybden). Disse to planene vil vi kalle fremre- og bakre plan og vi benevner deres dybdeverdier med henholdsvis z_F og z_B.



Figur 7.4: Prinsippet for 3D-klipping.

En god grunn for å innføre objektenes dybde i klippekriteriet, er at objekter som ligger svært nære projeksjonssentret kan skjule fjerntliggende objekter som vi er interessert i. Forstyrrende objekter av denne typen kan fjernes ved å klippe mot fremre plan. En annen viktig funksjon til fremre plan er at det siler ut objekter som ligger på baksiden av projeksjonssentret. En avbildning av disse objektene vil sannsynligvis ødelegge bildet. Ved å klippe mot bakre plan, vil vi unngå å avbilde objekter som ligger så langt unna at de ikke er av interesse. På denne måten oppnår vi å redusere informasjonen i bildet, vi reduserer med andre ord bildets entropi.

Vi går tilbake til figur 7.4 og ser at toppvinkelen til synsfeltpyramiden er definert ved projeksjonskonstanten og den rektangulære avgrensningen (2D-synsfeltet) i bildeplanet. Ved å utnytte dette forholdet, kan klippingen mot synsfeltpyramidens sidevegger gjøres meget rasjonelt. Anta et vilkårlig punkt P med koordinatene x_p, y_p, z_p. Spørsmålet om P ligger utenfor synsfeltpyramidens sidevegger, kan besvares ved å basere seg på beregning av tg til vinkelen mellom projeksjonsaksen og linjen fra projeksjonssentret til P. Vi har da betingelsene for at P ligger innenfor

synsfeltpyramidens sidevegger

$$-\frac{S_x}{d} \leq \frac{x_p}{z_p} \leq \frac{S_x}{d}$$

og

$$-\frac{S_y}{d} \leq \frac{y_p}{z_p} \leq \frac{S_y}{d}$$

hvor S_x og S_y definerer størrelsen på det rektangulære synsfeltet. Etter litt omforming går klippebetingelsene over til

$$-\frac{z_p}{d} \leq \frac{1}{S_x} x_p \leq \frac{z_p}{d} \qquad \text{og} \qquad -\frac{z_p}{d} \leq \frac{1}{S_y} y_p \leq \frac{z_p}{d} \tag{7.14}$$

Divisjon av modellkoordinatene med S_x og S_y vil føre til en normalisering av bildekoordinatene; i det de skalerte modellkoordinater ved overgang til bildekoordinater vil gi verdier i intervallet $[-1, 1]$ for punkter som befinner seg innenfor synsfeltet. Vi gjenkjenner størrelsen z_p/d som skaleringsfaktoren til de homogene bildekoordinatene i likning 7.4. Dette betyr at vi kan formulere klippingen direkte mot de homogene bildekoordinater ved å kjede inn en skalering med S_x og S_y i transformasjonen fra terrengkoordinater til homogene bildekoordinater. Den tilhørende matrisen blir vist senere. Klippebetingelsen basert på de skalerte homogene koordinater blir følgelig

$$-w \leq x_{h'} \leq w \qquad \text{og} \qquad -w \leq y_{h'} \leq w. \tag{7.15}$$

Vi skal så se på klippebetingelsene for fremre- og bakre plan. Ved overgang fra homogene bildekoordinater til ordinære koordinater vil dybdeinformasjonen går tapt, fordi alle z-verdier transformeres til verdien d. Dette betyr at klippingen må basere seg på de homogene bildekoordinatene (modellkoordinatene). Klippebetingelsen blir

$$wz_F \leq z_h \leq wz_B.$$

Multiplikasjon med w på høyre side av ulikheten kan elimineres ved å gå over til normaliserte z-koordinater. Vi innfører

$$S_z = z_B - z_F$$

og får

$$w\frac{1}{S_z} z_F \leq \frac{1}{S_z} z_h \leq w.$$

Skaleringen kan vi kjede inn i transformasjonen på samme måte som ved klippingen mot synsfeltpyramidens sidekanter. Klippekriteriet i forhold til de skalerte homogene koordinater blir da

$$wz_{F'} \leq z_{h'} \leq w \qquad \text{hvor} \qquad z_{F'} = \frac{1}{S_z} z_F \tag{7.16}$$

Dersom vi hadde redusert z-verdiene slik at fremre plan fikk $z = 0$, ville venstre side av klippebetingelsen blitt enklere, nemlig lik null. Ulempen ved dette er de numeriske problem som kan oppstå for punkter som ligger i nærheten av fremre plan.

Vi har nå utledet klippebetingelsene for synsfeltpyramidens 6 plan basert på normaliserte bildekoordinater. Normaliseringen er gitt ved

$$\begin{bmatrix} x_{h'} \\ y_{h'} \\ z_{h'} \\ w \end{bmatrix} = \begin{bmatrix} 1/S_x & 0 & 0 & 0 \\ 0 & 1/S_y & 0 & 0 \\ 0 & 0 & 1/S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} = \mathbf{N} \begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} \tag{7.17}$$

Ved å kjede matrisen $\mathbf{N}$ sammen med matrisen $\mathbf{F}$ i likning 7.12 får vi :

$$\begin{bmatrix} x'_h \\ y'_h \\ z'_h \\ w \end{bmatrix} = \mathbf{NF} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{F'} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{7.18}$$

7.2.1 Klipping av linjer

Vi kan utvide 2D-konseptet til Cohen-Sutherland til et 3D-konsept. Dette oppnås ved å addere til 2 ekstra biter til områdekoden. De to nye bitene sier hvordan et punkt ligger i forhold til fremre- og bakre plan. Utvidelsen fra 4 til 6 biter medfører ingen endring i strategien for testing. Vi har fortsatt at et punkt befinner seg innenfor klippevolumet dersom koden er 000000 og at en linje i sin helhet ligger utenfor volumet dersom en logisk *and* operasjon på endepunktene gir et resultat forskjellig fra null.

Dersom en linje er kandidat til å krysse synsfeltpyramiden, beregnes et eventuelt skjæringspunkt. Denne beregningen kan basere seg på skjæringsberegninger med homogene koordinater. Disse teknikkene er behandlet i kapitlet om homogene koordinater. Det er viktig å huske at vi gjør klippingen mot et volum. Først etter at klippingen er utført, transformeres de homogene koordinater til bildekoordinater ved å dividere med w. Deretter transformeres så de normaliserte bildekoordinater til det aktuelle synsfeltet ved vindu-synsfelt transformasjonen.

7.2.2 Oppsummering av 3D-klippeprosess

Vi skal kort oppsummere gangen i en 3D-klippeprosess.

1. Definisjon av synsfeltpyramiden. Oppstilling av likninger for pyramidens 6 plan.

2. Transformer fra terrengkoordinater til normaliserte homogene bildekoordinater.

3. Utfør klipping mot synsfeltpyramiden basert på de normaliserte homogene bildekoordinater.

4. Beregn ordinære bildekoordinater ved å dividere med w.

5. Til slutt utføres en vindu-synsfelt transformasjon.

Det er å bemerke at vi ikke har lagt inn en prosess for fjerning av skjulte objekter. I mange tilfeller er dette både ønskelig og helt nødvendig for å få et bilde som blir forståelig. Dette vil bli behandlet i et senere kapittel.

7.3 Fjerning av skjulte flater og skjulte linjer

For å lage mest mulig realistiske 2D-bilder av 3D-objekter, er det som regel nødvendig å fjerne skjulte linjer og skjulte flater. I de tilfeller vi ønsker å illudere gjennomsiktige flater, er det imidlertid ønskelig å tegne skjulte linjer og flater, men da med endret visuell synlighet (for eksempel stiplet strek eller svakere kontrast mellom flatene). Generelt vil algoritmer for fjerning av skjulte linjer og flater være ressurskrevende, men dersom en kan gjøre forutsetninger om formen til det objektet som skal avbildes, kan dette utnyttes til å redusere tidsforbruket. Algoritmene kan grovt sett klassifiseres i to hovedgrupper

- objekt-orienterte algoritmer
- bilde-orienterte algoritmer

De objekt-orienterte algoritmer gjør berekningene i et 3D-koordinatsystem, mens de bilde-orienterte algoritmer opererer mot bildeplanets pixler. Stort sett vil algoritmer for fjerning av skjulte linjer være objekt-orienterte mens algoritmer for fjerning av skjulte flater gjerne er bilde-orienterte. Selv om det er store forskjeller på de ulike algoritmer for fjerning av skjulte linjer og skjulte flater, utnyttes ofte sortering og koherens (beslektede naboer) for å redusere prosesseringstida.

7.3.1 Fjerne baksider

En enkel objekt-orientert metode for å fjerne plan som vender baksiden mot projeksjonssentret, er å teste på fortegnet til innerproduktet mellom planets likning og projeksjonssentret. Se likning 1.29. Ved bruk av denne metoden innretter en seg slik at punkter på framsiden av planet gir positivt innerprodukt mens punkter på baksiden av planet gir negativt innerprodukt. Baksiden defineres som den siden av planet som vender inn mot objektet. Matematisk kan dette gjøres ved å velge høvelig fortegn til skaleringsfaktoren k i likning 1.33.

Baksiden vil ikke være synlig fra utsiden av objektet. Følgelig har vi at dersom projeksjonssentret ligger på baksiden av planet, vil planet uten videre kunne fjernes. Omlag halvparten av objektets overflate vil på denne måten bli eliminert.

7.3.2 Dybde-buffer metode

Dybde-buffer metoden er en bilde-orientert algoritme. Metoden benytter to array som vi kaller *dybde* og $farge$. Arrayene gis lengde lik antall pixel i bildeplanet. Initielt settes

$$dybde(x, y) = \infty \qquad \text{og} \qquad farge(x, y) = bakgrunn.$$

Et høvelig antall punkter (avhengig av antall pixel) på objektoverflaten transformeres til bildeplanet.

Anta at punkt i, som har dybde z_i of farge f_i, transformeres til pixel (x, y). Dersom $dybde(x, y)$ er større enn z_i, settes

$$dybde(x, y) = z_i \qquad \text{og} \qquad farge(x, y) = f_i,$$

ellers endres ikke innholdet i *dybde* og $farge$. Riktigheten av dette er lett å innse. Dersom to områder på objektet transformeres til samme pixel i bildeplanet, vil det området som ligger nærmest projeksjonssentret (det punktet som har den minste dybden) være synlig og vil følgelig skjule det andre området.

7.3.3 Dybde-sorterings metode

Det er mulig å bruke både bilde-orienterte og objekt-orienterte operasjoner i algoritmer for å fjerne skjulte flater. *Dybde-sortering* er en slik algoritme, og den har følgende to basisfunksjoner:

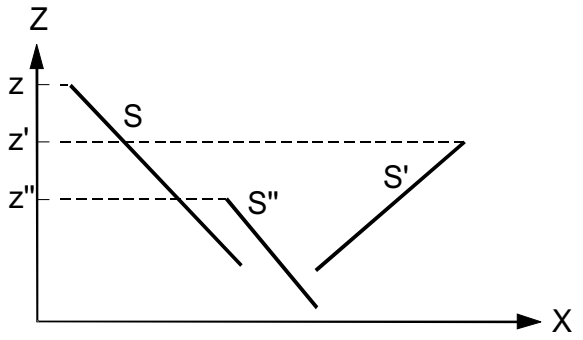
1. Sorter flater etter minkende dybde.

2. Overfør flatene til bildeplanet (den grafiske skjermen) i rekkefølge fra flate med største til minste dybde.

Metoden blir ofte kalt *malerens algoritme*. Dette skyldes at en maler gjerne starter med bakgrunnen for til slutt å legge på objektene nærmest betraktningsstedet.

Som grunnlag for sorteringen benyttes dybden til det punktet på en flate som har størst dybde. Sortering basert på bare dette grunnlaget vil imidlertid føre til komplikasjoner. La oss anta at to flater overlapper hverandre i $z-$retning (dybde). Dersom disse flaten også overlapper hverandre i xy-planet (bildeplanet), kan det føre til at flatene må bytte rekkefølge i den sorterte listen. Som det framgår av figur 7.5, har flate S større dybde enn flate S''. Flatene vil derfor ha rekkefølgen S, S'' i den sorterte listen. Men fordi S delvis skjuler S'', må rekkefølgen endres til S'', S. Vi må derfor i algoritmen innføre noen tester i tillegg til sorteringen. Dersom noen av de følgende tester er sanne, behøves ingen ombytting av rekkefølgen til flatene. Testene er listet i prioritert rekkefølge. Så snart en test er sann, kan testingen avsluttes.

1. De omskrivende rektangler til de to flatene overlapper ikke i $xy-$planet.



Figur 7.5: Tre flater som etter dybdesortering får rekkefølgen S, S', S''. Rekkefølgen skal være S', S'', S

2. Den flaten som har størst dybde ligger i sin helhet på baksida av den andre flaten i forhold til projeksjonssentret.

3. Den flaten som har minst dybde ligger i sin helhet på framsida av den andre flaten i forhold til projeksjonssentret.

4. Projeksjonen av de to flatene i bildeplanet overlapper ikke.

For testene 2) og 3) kan vi benytte planligningene på samme måte som i avsnitt 7.3.1 da vi fant om punkter lå på baksiden av et plan. Dersom for eksempel samtlige punkter til plan S ligger på baksiden av plan S', ligger S i sin helhet på baksiden av S'.

Metoden ser ut til å passe svært bra for trekantmodeller (digital terrengmodell). Ved å innføre de restriksjoner som gjelder for trianglenes plassering i forhold til hverandre (de skal ligge på terrengoverflate), kan testene 1) til 4) forenkles. To triangler vil vel aldri kunne overlappe hverandre både i dybde og i $xy-$planet. Test 4) ser derfor ut til å være unødvendig i dette spesielle tilfellet.

7.3.4 Fjerning av skjulte linjer

Når bare omrisset av et objekt skal vises, er det aktuelt å fjerne skjulte linjer. En direkte metode for å fjerne skjulte linjer er å sammenligne hver enkelt linje mot samtlige flater i senen. Dette blir som å klippe mot et vindu med vilkårlig form, unntatt at vi nå klipper vekk de deler som er skjult av flatene. Ved å benytte koherens metoder kan en unngå å teste hver eneste posisjon på linjene. Vi beregner for eksempel skjæringspunktene mellom linjen og den aktuelle flatens begrensningslinje i projeksjonsplanet. Dersom begge disse skjæringspunktene har større dybde enn flaten i disse punktene, er linjesegmentet mellom skjæringspunktene i sin helhet skjult av flaten. Dersom linjen har større dybde enn flaten i det ene skjæringspunktet og mindre dybde enn flaten i det andre skjæringspunktet, vil linjen delvis

være skjult av flaten. I så fall beregnes skjæringspunktet mellom linjen og flaten i objekt-koordinatsystemet og den skjulte delen av linjen klippes vekk.

Kapittel 8

Kartografisk visualisering av modellen

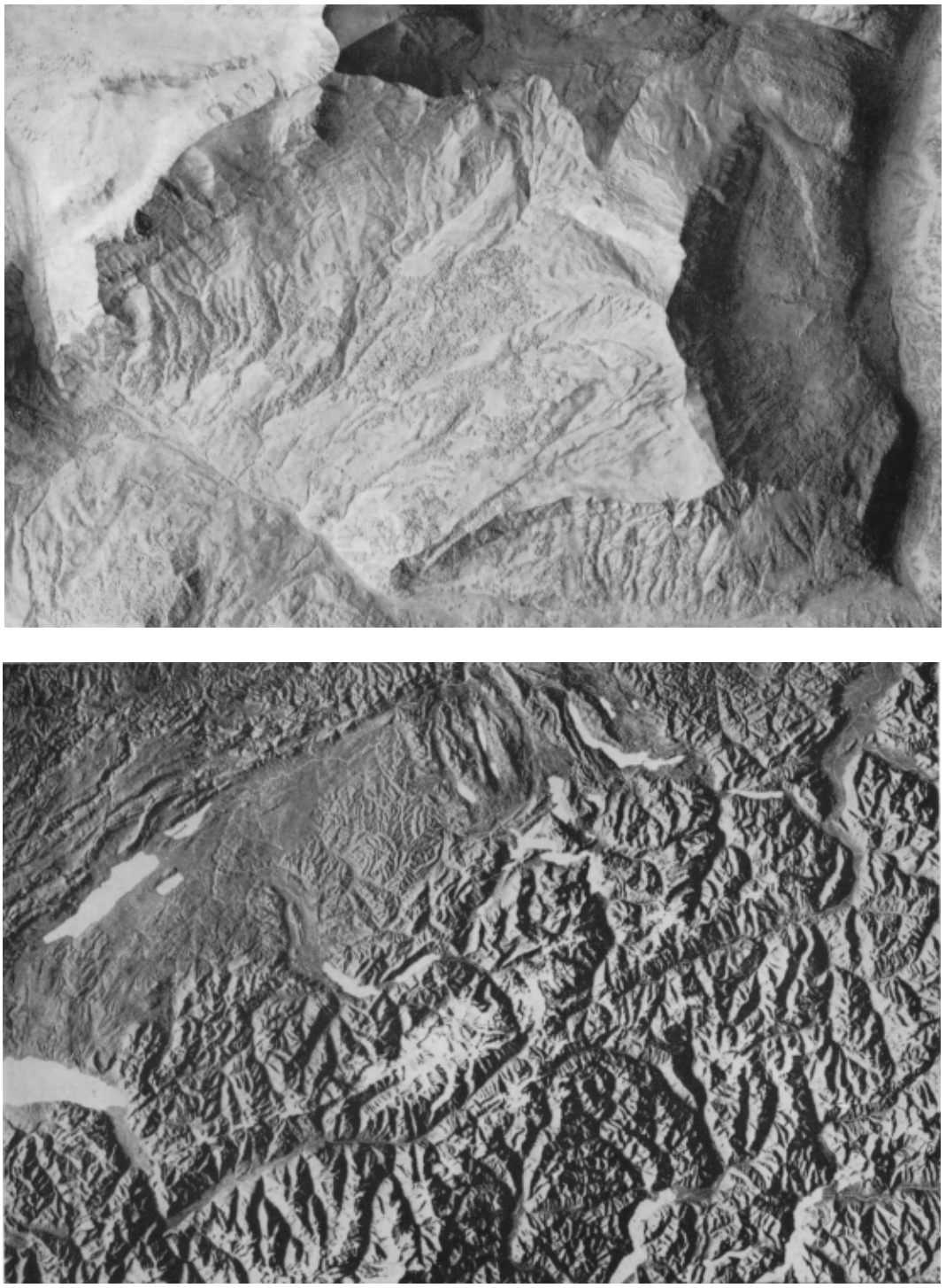
I kartografien har vi lange tradisjoner i å benytte ulike grafiske virkemidler for å vise den tredje dimensjonen til et geografisk objekt i et 2D-bilde. For en mere utdypende beskrivelse av tradisjonelle kartografiske framstillingsmåter vises til kompendiet [Bjø97]

8.1 Kartografisk skyggelegging

Noen norske kartserier har de senere årene fått påført automatisk generert skrålysskygge, for eksempel N250 fra Statens Kartverk. Fargelagte høydesjikt i kombinasjon med skrålysskygge kan gi gode 3D-effekter, se eksempelkart hos [Imh65]. I Sveits har man lange tradisjoner i skyggelegging av kart og de manuelle teknikker ble i sin tid utviklet til et høyt nivå; se figur 8.1 og legg merke til at skyggeleggingen klarer å framstille små topografiske variasjoner.

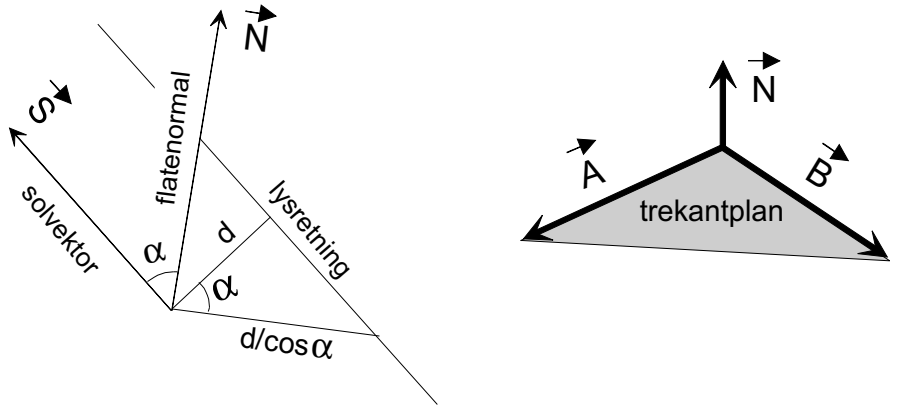
Skyggelegging i kartografien baserer seg på at gråtonen i kartet blir bestemt som en funksjon av vinkelen mellom lysretningen og terrengoverflaten. For å oppnå optimal 3D-effekt er det imidlertid i visse situasjoner nødvendig å fravike hovedprinsippet noe. Små justeringer av lysretningen kan gi betydelige utslag på 3D-effekten. Dette gjør at optimal 3D-effekt krever en viss individuell tilpasning av lysretningen for hvert enkelt kartbilde. Enkelte forsøk tyder på at det kan lønne seg å beregne gråtonen som et gjennomsnitt fra flere lysretninger. Dette gjør det vanskelig å finne automatiserte rutiner som gir en optimal skyggelegging. Vi vil ikke gå nærmere inn på denne optimaliseringsproblematikken, men presentere formelverket for enkel beregning av gråtonen i et skyggekart.

Vi antar at solen ligger så langt unna at de stråler som treffer terrenget er parallelle. Dette betyr at vi kan operere med den samme solvektoren for hele kartet. Figur 8.2 illustrerer at den lysenergien som treffer et rettlinjet snitt av terrengoverflaten, fordeler seg over et område viss størrelse kan uttrykkes ved cos til vinkelen mellom



Figur 8.1: Eksempler på kartografisk skrålysskygge, fra [Imh65] side 198-199

lysretningen og normalen på snittet. Vi tenker oss at terrengmodellen deles opp i rekke små fasetter. Gråtonen for en fasett framkommer ved en beregning av cos til vinkelen mellom flatenormalen og solvektoren.



Figur 8.2: Utledning av gråtonen ved skrålysskygge som en funksjon av vinkelen mellom flatenormalen og solvektoren. Når snittlinjen står skrått på lysretningen, blir lysenergien å fordele på et større område enn når snittlinjen står vinkelrett på lysretningen.

Vi benevner solvektoren med $\overrightarrow{S}$ og flatenormalen med $\overrightarrow{N}$, se illustrasjonen i figur 8.2. Vinkelen α mellom de to vektorene finnes ved deres innerprodukt, gitt ved

$$\overrightarrow{N} \cdot \overrightarrow{S} = \mid \overrightarrow{N} \mid\mid \overrightarrow{S} \mid \cos\alpha. \tag{8.1}$$

Siden vi kan forutsette parallelle lysstråler, kan vi benytte den samme solvektoren for hele kartet. Ved å normalisere lengden av solvektoren kan vi redusere antall aritmetiske operasjoner for det samlede antall fasetter, altså ved å velge $\overrightarrow{S}$ slik at $s_x^2+s_y^2+s_h^2 = 1$. Flatenormalen $\overrightarrow{N} = [n_x, n_y, n_h]$ finner vi ved kryssproduktet mellom to vektorer $\overrightarrow{A}$ og $\overrightarrow{B}$ som ligger i flatens plan, gitt ved

$$\overrightarrow{N} = \overrightarrow{A} \times \overrightarrow{B}. \tag{8.2}$$

Vi definerer

$$\overrightarrow{A} = [a_1, a_2, a_3], \tag{8.3}$$
$$\overrightarrow{B} = [b_1, b_2, b_3], \tag{8.4}$$
$$\overrightarrow{N} = [n_1, n_2, n_3], \tag{8.5}$$

og stiller opp følgende to likninger i vektorenes innerprodukt til bestemmelsen av $\overrightarrow{N}$:

$$\overrightarrow{A} \cdot \overrightarrow{N} = a_1n_1 + a_2n_2 + a_3n_3 = 0, \tag{8.6}$$
$$\overrightarrow{B} \cdot \overrightarrow{N} = b_1n_1 + b_2n_2 + b_3n_3 = 0. \tag{8.7}$$

Siden lengden av $\overrightarrow{N}$ kan velges fritt, gir dette oss en frihetsgrad. Vi multipliserer likningene 8.6 og 8.7 med henholdsvis b_3 og $-a_3$, summerer og får

$$(a_1b_3 - a_3b_1)n_1 + (a_2b_3 - a_3b_2)n_2 = 0,$$

som tilfredsstilles av

$$n_1 = a_2b_3 - a_3b_2, \tag{8.8}$$
$$n_2 = a_3b_1 - a_1b_3. \tag{8.9}$$

Ved innsetting får vi

$$n_3 = a_1b_2 - a_2b_1. \tag{8.10}$$

Gråtonen i kartet beregnes som en funksjon av $\cos\alpha$, gitt ved

$$g = \begin{cases} 0 & dersom\ \cos\alpha < 0, \\ g_{\max}\cos\alpha & ellers, \end{cases} \tag{8.11}$$

hvor $g_{\max}$ er en faktor som skalerer intervallet for $\cos\alpha$ til det aktuelle intervallet for gråtonene, for eksempel $[0, 255]$. Det kan også være aktuelt å innføre en ekponentiell funksjon f i $\cos\alpha$ for å finjustere gråtonene i forhold til øyets perseptuelle egenskaper; $g = g_{\max}f(\cos\alpha)$. Det forholder seg nemlig slik at om lysenergien som reflekteres fra en flate endres, gir ikke dette en lineær virkning på vår oppfattelse av gråverdien til flaten.

I det spesielle tilfellet at vi har en rutemodell som er parallell med koordinataksene, kan vi utlede et forenklet uttrykk for $\cos\alpha$. Vi velger vektorparet $\overrightarrow{A}$ og $\overrightarrow{B}$ slik at

$$\overrightarrow{A} = [a_1, a_2, a_3] = [a_x, a_y, a_h] = [1, 0, \Delta h_x], \tag{8.12}$$
$$\overrightarrow{B} = [b_1, b_2, b_3] = [b_x, b_y, b_h] = [0, 1, \Delta h_y]. \tag{8.13}$$

Vektorkomponentene Δh_x og Δh_y, som er høydedifferanser i henholdsvis X- og Y-retning, kan beregnes ved et gjennomsnitt for den aktuelle ruten. Ved å innføre vektorkomponentene fra likningene 8.12 og 8.13, blir normalvektoren

$$\overrightarrow{N} = [n_1, n_2, n_3] = [-\Delta h_x, -\Delta h_y, 1]. \tag{8.14}$$

Verdien for $\overrightarrow{N}$ fra likning 8.14 satt inn i likning 8.1 gir

$$\cos\alpha = \frac{-s_x\Delta h_x - s_y\Delta h_y + s_h}{\sqrt{\Delta h_x^2 + \Delta h_y^2 + 1}} \quad \text{for } |\overrightarrow{S}| = s_x^2 + s_y^2 + s_h^2 = 1. \tag{8.15}$$

8.2 Naturtro bilder

For å oppnå naturtro bilder, må vi blant annet ta omsyn til terrengoverflatens refleksjonsegenskaper. Skyggemodellen blir derfor komplisert og ressurskrevende å produsere. Naturtro bilder har noe for seg ved estetisk vurdering av planlagte byggverk.

8.3 Fargelagte høydesjikt

Fargelagte høydesjikt er en koropletkartframstilling av terrengets høydeforhold. Det finnes ulike prinsipper for valg av fargeskala og valg av klassegrenser for de ulike høydesjiktene; se [Imh65].

8.4 Fysiske modeller

Innen verkstedindustrien benyttes fresemaskiner for å skjære ut modeller av 3D-objekter. I de senere årene er det utviklet en metode som baserer seg på at en veske størkner når den blir belyst. Ved å sveipe over vesken med en tynn laserstråle, kan vi få vesken til å størkne langs en profillinje til 3D-objektet. Ved så å trekke profillinjen litt opp fra vesken, kan neste profillinje lages. På denne måten trekkes 3D-modellen opp fra veskekaret. Metoden kalles stereolitografi og benyttes innen verkstedindustrien.

8.5 Perspektivmodeller

Som grunnlag for å lage perspektivbilder av terrengmodeller kan vi benytte isolinjer, parallelle profiler eller linjemønstre som framkommer ved trekantnett eller rutenettet. Vi kan også benytte gråtonemodellen fra kartografisk skrålysskygge. Gråtonemodellen kan kombineres med fargelagte høydesjikt, men for at fargesjiktene ikke skal ødelegge inntrykket av skyggeleggingen, må det benyttes pastellaktige farger, se eksempelkartene hos [Imh65].

Bibliografi

[Aki70] Hiroshi Akima. A new method of interpolation and smooth curve fitting based on local procedures. *Journal of the Association for Computing Machinery*, 17(4):589–602, October 1970.

[Bjø88] Jan T. Bjørke. Quadtrees and triangulation in digital elevation models. In *International Archives of Photogrammetry and Remote Sensing*, pages 38–44. International Society for Photogrammetry and Remote Sensing, Committee of the 16th International Congress of ISPRS, 1988. Volume 27, part B4, Commision IV.

[Bjø97] Jan T. Bjørke. Digital kartografi, del 1: kartografisk kommunikasjon. Kompendium, NTNU, Institutt for kart og oppmåling, 1997.

[BM79] Ketil Bø and Sigmund Hov Moen. *Grafisk databehandling*. Tapir, 1979.

[Bur87] P. A. Burrough. *Principles of Geographical Information Systems for Land Resources Assesment*. Oxford University Press, New York, 1987.

[Coc97] Sophie Cockcroft. A taxonomy of spatial data integrity constraints. *Geoinformatica*, 1(4):327–343, 1997.

[Cre93] Noel A. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, Inc., New York, revised edition, 1993.

[Del34] B. Delaunay. Sur la sphere vide. *Bulletin of the Academy of Sciences of the USSR*, pages 793–800, 1934.

[DFP95] Leila De Floriani and Enrico Puppo. Hierarchical triangulation for multiresolution surface description. *ACM Transactions on Graphics*, 14(4):363–411, October 1995.

[HB86] Donald Hearn and M. Pauline Baker. *Computer Graphics*. Prentice-Hall International, USA, 1986.

[Hel90] Martin Heller. Triangulation algorithms for adaptive terrain modelling. In *Proceedings of the 4th International Symposium on Spatial Data Handling*, pages 163–174, July 1990.

[Imh65] Eduard Imhof. *Kartographische Geländedarstellung*. Walter de Greyter & Co., Berlin, 1965. Boken finnes i en engelsk oversettelse.

[Kan79] D. Kannan. *An Introduction to Stochastic Processes*. ., North Holland , New York, 1979. ISBN 0-444-00301-0.

[Mat71] G. Matheron. The theory of regionalized variables and its applications. Les Cahiers du centre de morphologie mathematique de Fontainebleu. Ecole National Superieure des Mines de Paris, 1971.

[Max46] E.A. Maxwell. *Methods of Plane Projective Geometry Based on the use of General Homogeneous Coordinates*. Cambridge Univ.Press, Cambridge, 1946.

[Max51] E.A. Maxwell. *General Homogeneous Coordinates in Space of Three Dimensions*. Cambridge Univ.Press, Cambridge, 1951.

[McL76] D.H. McLain. Two dimensional interpolation from random data. *The Computer Journal*, 19(2):178–181, 1976.

[Mid93] Terje Midtbø. *Spatial Modelling by Delaunay Networks of Two and Three Dimensions*. PhD thesis, The Norwegian Institute of Technology, Department of Surveying and Mapping, N-7034 Trondheim, February 1993.

[Mid94] Terje Midtbø. Removing points from a delaunay triangulation. In Thomas C. Waugh and Richard G. Healy, editors, *SDH 94, Sixth International Symposium on Spatial Data Handling*, pages 739–750, Edinburg, Scotland, UK, September 1994. International Geographical Union Commision on Geographic Information Systems.

[MR80] M.J. McCullagh and C.G. Ross. Delaunay triangulation of a random data set for isarithmic mapping. *The Cartographic Journal*, 17(2):93–99, des 1980.

[NS84] W.M. Newman and R.F. Sproull. *Principles of Interactive Computer Graphics*. McGraw-Hill, Inc., 1984. ISBN 0-07-046338-7.

[Ole75] Ricardo A. Olea. Optimum mapping techniques using regionalized variable theory. Series on Spatial Analysis. Kansas Geological Survey, Lawrence, 1975.

[PG97] Lutz Plümer and Gerhard Gröger. Achieving integrity in geographic information systems - maps and nested maps. *Geoinformatica*, 1(4):345–367, 1997.

[Ran89] John R. Rankin. *Computer Graphics Software Construction*. Prentice Hall of Australia Pty Ltd., Australia, 1989.

[TC11] A. J. Thiessen and Alter J. C. Precipitation averages for large areas. *Monthly Weather Review*, 39:1082–1084, 1911.

[Yoë77] P. Yoëli. Computer executed interpolation of contours into arrays of randomly distributed height-points. *The Cartographic Journal*, 14(2):103–108, des 1977.