Universitaire Instelling Antwerpen
Departement Wiskunde-Informatica

Het strandlijn-algoritme voor het berekenen van Voronoi-diagrammen.

Sven Maerivoet

7 December 2000

Inhoudsopgave

Hoofdstuk 1

Wat is een Voronoi-diagram ?

In dit hoofdstuk wordt kort geïllustreerd wat een Voronoi-diagram is, daarna worden enkele formele definities gegeven en tot slot vermeld ik enkele eigenschappen die voor een Voronoi-diagram gelden.

1.1 Illustratie

Voronoi-diagrammen worden in tal van toepassingen gebruikt :

- postkantorenprobleem,
- klantensectoren,
- patroonherkenning,
- trianguleren van reliëfkaarten,
- minimale kabelverbindingen,
- veilige trajectplanning voor autonome robots,
- ...

In bovenstaande toepassingen is er telkens 1 vaststaand gegeven : voor een (willekeurig) gegeven punt in het vlak zoekt men de dichtsbijzijnde site. Dit geeft aanleiding tot het verdelen van het vlak in een aantal cellen, de *Voronoi-cellen*, die elk bij een welbepaalde site horen. In zo'n cel geldt dan dat elk punt erin dichter bij de site in kwestie ligt dan bij een andere site. De verzameling van alle Voronoi-cellen van een gegeven verzameling sites wordt een Voronoi-diagram genoemd.

1.2 Definities

Gegeven is een verzameling van n sites in het Euclidisch gepunte vlak Π_0 :

$$P = \{p_1, p_2, \ldots, p_n\}. \tag{1.1}$$

Als afstandsmaat gebruiken we de Euclidische metriek :

$$d(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}. \tag{1.2}$$

Het Voronoi-diagram van P (genoteerd $Vor(P)$) is de opdeling van Π_0 in n cellen (1 cel voor elke p_i). Alle punten q in zo'n cel van een zekere p_i hebben de eigenschap dat $d(p_i, q) \leq d(p_j, q)$ voor alle $j \neq i$. Een Voronoi-cel van p_i wordt genoteerd als $V(p_i)$.

Dit gegeven, schrijven we dus :

$$Vor(P) = \bigcup_{i \in \{1,\ldots,n\}} V(p_i). \tag{1.3}$$

Als we de structuur van een enkele Voronoi-cel $V(p_i)$ beschouwen, dan merken we het volgende :

$$V(p_i) = \bigcap_{1 \leq j \leq n, j \neq i} h(p_i, p_j), \tag{1.4}$$

waarbij $h(p_i, p_j)$ het open halfvlak is dat p_i bevat en waarvan de scheiding wordt gevormd door de bissectrice van p_i en p_j.

1.3 Eigenschappen

Voor een gegeven Voronoi-diagram $Vor(P)$ gelden enkele interessante eigenschappen :

1.3.1 Formule van Euler

De formule van Euler voor een verbonden gerichte graf stelt dat :

$$n_k - n_v + n_p = 2, \tag{1.5}$$

met n_k het aantal knooppunten, n_v het aantal verbindingen en n_p het aantal vlakken (polygonen). Als we dit willen toepassen op een Voronoi-diagram moeten we er 1 knooppunt aan toevoegen, namelijk k_∞, het knooppunt dat 'op oneindig' ligt en waarmee alle open halfoneindige verbindingen verbonden zijn. Er geldt dus dat :

$$(n_k + 1) - n_v + n = 2, \tag{1.6}$$

met nu n_k het aantal punten (*vertices*) gevormd door de doorsnedes van alle n_v verbindingen en n het aantal sites in kwestie.

1.3.2 Lege cirkels

De grootste lege cirkel $C(q)$ van een punt q wordt gedefinieerd als de grootste cirkel die q als middelpunt heeft en geen enkele site van P omvat (de rand van de cirkel niet meegerekend).

> *Een punt q is een vertex van $Vor(P)$ als en slechts als de grootste lege cirkel $C(q)$ drie of meer sites op zijn rand bevat.*

> *De bissectrice van twee sites p_i en p_j bepaalt een verbinding van $Vor(P)$ als en slechts als er een punt q bestaat zodat $C(q)$ zowel p_i als p_j op zijn rand bevat maar geen enkele andere site omvat.*

Hoofdstuk 2

Berekenen van een Voronoi-diagram

In dit deel wordt zeer kort toegelicht op welke verschillende manieren een Voronoi-diagram berekend kan worden.

2.1 De brute aanpak

Het doel is nu : bereken het Voronoi-diagram van een verzameling sites P. Een naïeve manier om dit te doen is de volgende :

$$\forall p_i \ : \bigcap_{1 \leq j \leq n, j \neq i} h(p_i, p_j). \tag{2.1}$$

Per Voronoi-cel zijn er $O(n \log(n))$ bewerkingen nodig, voor alle n sites maakt dit dus een totale tijdscomplexiteit van $O(n^2 \log(n))$, wat niet zo denderend is (we willen namelijk het lastige kwadraat kwijt).

Maar het kan beter...

Fortune's algoritme !

2.2 Alternatieve methodes

Naast het algoritme van Fortune zijn er nog tal van andere manieren om Voronoi-diagrammen te berekenen (die dezelfde complexiteit zullen hebben).

Vermits Voronoi-diagrammen duaal zijn aan *Delaunay-triangulaties*, kan men ze ook berekenen aan de hand van een Delaunay-triangulatie van de verzameling P (of anders gezegd, men kan een Delaunay-triangulatie uitvoeren, gebaseerd op het Voronoi-diagram).

Daarnaast is er ook nog onder andere de welbekende '*verdeel-en-heers*' techniek die recursief te werk gaat door het vlak telkens in 2 stukken op te delen.

Hoofdstuk 3

Fortune's algoritme

In dit deel wordt uitgebreid besproken hoe een Voronoi-diagram concreet berekend zal worden, waar de moeilijkheid juist ligt en hoe deze wordt aangepakt. Er wordt pseudocode gegeven (alsmede de datastructuren die worden gebruikt) en tot slot wordt er iets gezegd over de complexiteit van het algoritme en gedegenereerde situaties.

3.1 Beschrijving

Dit algoritme is gebaseerd op de techniek van '*sweeping*' (vegen). Er wordt gebruik gemaakt van een horizontale rechte L die het vlak Π_0 afscant en waarbij er gepast wordt gereageerd op bepaalde gebeurtenissen gedurende de scan.

Vermits het vegen van boven naar onder gebeurt, treedt er een niet onbelangrijk probleempje op :

> Het deel van $Vor(P)$ dat boven de '*veegrechte*' ligt, is niet alleen afhankelijk van de sites die boven L liggen, *maar is ook afhankelijk van de sites die onder L liggen !*

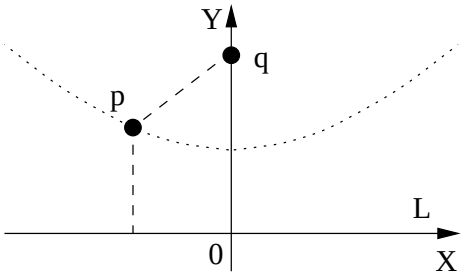
Hierbij merken we op dat voor elk punt q dat boven L ligt, er geldt dat de afstand van q tot een zekere site onder L groter dan (of gelijk aan) de afstand van q tot L zelf is.

Bijgevolg geldt dan dat de dichstbijzijnde site voor q niet onder L kan liggen als de afstand van q tot een zekere site boven L minstens zo klein is als de afstand van q tot L zelf.

Gegeven een punt q dat op de Y-as ligt en een rechte L die met de X-as samenvalt. Beschouw nu alle punten die even ver van q als van L liggen :

$$\begin{aligned} d(p,q) &= d(p,L) \qquad (3.1)\\ \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} &= \frac{ax_p + by_p + c}{\sqrt{a^2 + b^2}} \\ \sqrt{x_p^2 + (y_p - y_q)^2} &= \frac{y_p}{\sqrt{1}} \\ x_p^2 + (y_p - y_q)^2 &= y_p^2 \\ x_p^2 + y_p^2 - 2y_p y_q + y_q^2 &= y_p^2 \\ x_p^2 - 2y_p y_q + y_q^2 &= 0 \\ y_p &= \frac{x_p^2 + y_q^2}{2y_q} \end{aligned}$$

waarbij $L \leftrightarrow ax + by + c = 0$ of in dit geval $L \leftrightarrow y = 0$. Zoals in figuur 3.1 te zien is, vormen de punten p een meetkundige plaats die een *parabool* is.

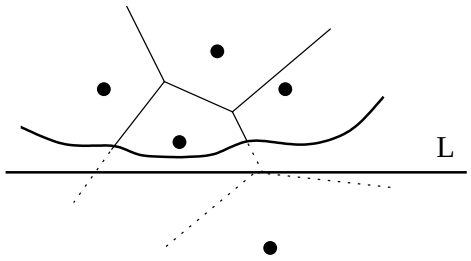


Figuur 3.1: De meetkundige plaats is een parabool.

Bijgevolg wordt dus de meetkundige plaats van alle punten q, die dichter bij een zekere site p_i (boven L) dan bij L zelf liggen, gevormd door een sequentie van *parabool-bogen* (zie figuur 3.2).

Om deze reden noemt men Fortune's algoritme soms ook het 'strandlijn-algoritme' (*beach line algorithm*).

De hypothese die we nu aanvaarden, is dat $Vor(P)$ volledig wordt bepaald door deze strandlijn. Het gedeelte boven de strandlijn ligt vast, het gedeelte tussen de strandlijn en de veegrechte L kan nog niet bepaald worden vermits het nog kan veranderen door sites die onder L liggen (en nog door het algoritme moeten verwerkt worden).



Figuur 3.2: Een strandlijn.

Merk op dat van eenzelfde parabool verschillende (gescheiden) stukken deel kunnen uitmaken van de strandlijn. En houd ook indachtig dat de snijpunten die de paraboolbogen met elkaar gemeen hebben exact op verbindingen van het Voronoidiagram liggen. **De snijpunten van de paraboolbogen volgen exact het Voronoidiagram terwijl de veegrechte van boven naar onder beweegt.**

De strandlijn wordt niet expliciet bijgehouden aangezien deze continu verandert wanneer L beweegt. In plaats daarvan houden we enkel de snijpunten van de paraboolbogen bij in een gebalanceerde binaire boom T. In de bladeren van T zitten de sites die de de paraboolbogen bepalen. De interne knooppunten van T stellen de snijpunten van de strandlijn voor. Een snijpunt wordt bewaard als een geordend paar sites $< p_i, p_j >$ waarbij p_i de parabool links van het snijpunt en p_j de parabool rechts van het snijpunt definieert.

Hoe de structuur van de strandlijn verandert, is afhankelijk van een aantal gebeurtenissen. Deze gebeurtenissen zorgen ervoor dat er nieuwe paraboolbogen verschijnen en oude paraboolbogen verdwijnen (door in te krimpen tot een punt).

3.2 De verschillende soorten gebeurtenissen

Er treden twee soorten gebeurtenissen op :

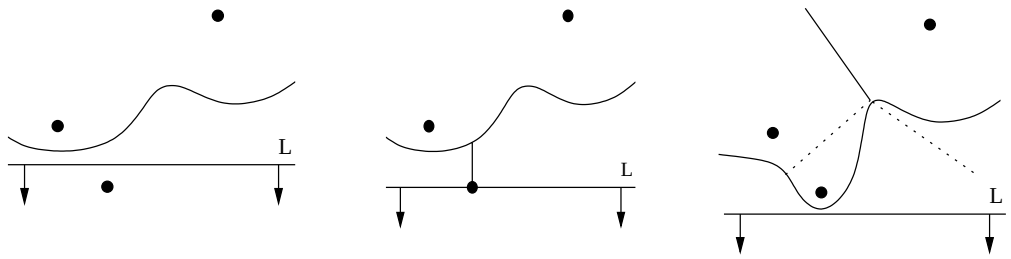
- site-gebeurtenissen
- en cirkel-gebeurtenissen.

3.2.1 Site-gebeurtenissen

Wanneer de veegrechte L een nieuwe, nog niet verwerkte, site bereikt, zal er een nieuwe paraboolboog verschijnen. Deze manifesteert zich eerst als een gede-

genereerde parabool en van zodra de veegrechte zich verder naar beneden verplaatst, verbreedt deze nieuwe parabool zich (zie figuur 3.3).

Bij een site-gebeurtenis ontstaan dus twee nieuwe snijpunten die verbindingen van het Voronoi-diagram beginnen te volgen.



Figuur 3.3: Een site-gebeurtenis.

3.2.2 Cirkel-gebeurtenissen

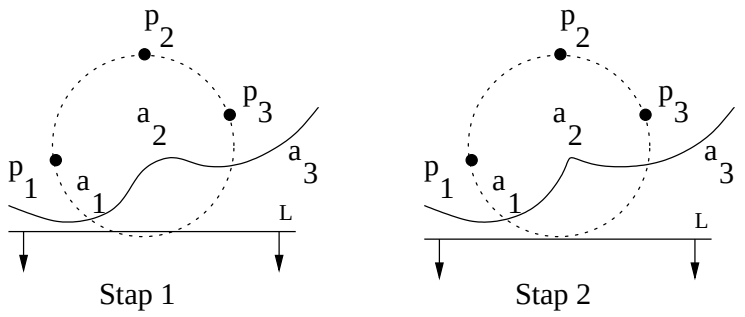
Beschouw drie opeenvolgende paraboolbogen α_1, α_2 en α_3 die gedefinieerd worden door drie verschillende sites, respectievelijk p_1, p_2 en p_3.

Op het moment dat paraboolboog α_2 verdwijnt, zullen de drie oorspronkelijke paraboolbogen door een en hetzelfde punt q gaan. Dit punt q ligt even ver van p_1, p_2 en p_3, met andere woorden, er gaat een cirkel door p_1, p_2 en p_3 met als middelpunt q. Het laagste punt op deze cirkel ligt tevens op de veegrechte L (zie figuur 3.4).

Er kan geen site in de cirkel liggen aangezien deze dan dichter bij q zou liggen dan dat q bij L ligt, wat in strijd is met het feit dat q op de strandlijn ligt. **Hieruit volgt dat q een vertex is van het Voronoi-diagram !**

Dus, van zodra een paraboolboog verdwijnt uit de strandlijn en dus twee snijpunten samenvallen, snijden er ook twee verbindingen van het Voronoi-diagram elkaar.

De gebeurtenis die optreedt wanneer de veegrechte L het laagste punt van een cirkel door drie sites (die drie opeenvolgende paraboolbogen definiëren) bereikt, wordt een cirkel-gebeurtenis genoemd.



Figuur 3.4: Een cirkel-gebeurtenis.

3.3 De gebruikte datastructuren

Vermits het doel het berekenen van het Voronoi-diagram van een verzameling P van sites p_i is, hebben we een datastructuur nodig om dit Voronoi-diagram op te slaan. Daarnaast zijn er nog twee extra datastructuren nodig (voor het veeg-algoritme) : een gebeurtenissen-wachtrij (*event-queue*) en een structuur die de status van de veeglijn weergeeft (wat ineens een voorstelling van de strandlijn is).

Het Voronoi-diagram onder constructie wordt in een dubbel-verbonden lijst van verbindingen (zie pagina 31 in [dBvKOS97]) opgeslagen. Merk op dat een Voronoi-diagram niet echt perfect in deze structuur past aangezien er halfoneindige verbindingen optreden. Dit probleem kan echter gemakkelijk omzeild worden door een grote rechthoek (*bounding box*) rond P te tekenen en de snijpunten van de verbindingen van het Voronoi-diagram met deze rechthoek te berekenen. Op deze manier ontstaan er gesloten cellen aan de randen van het Voronoi-diagram.

Zoals reeds eerder vermeld, wordt de strandlijn voorgesteld door een gebalanceerde binaire boom T. Deze voorstelling vormt dan ook de datatstructuur voor de status van de veeglijn L.

De gebeurtenissen-wachtrij wordt voorgesteld door een prioriteits-wachtrij Q (*priority-queue*). De prioriteit van een gebeurtenis wordt gevormd door z'n y-coördinaat. In deze wachtrij zitten bij het begin van het algoritme alle site-gebeurtenissen, aangezien deze op voorhand al gekend zijn (de y-coördinaat van elke site is immers ten allen tijde gekend vermits alle sites gekend zijn).

Voor een site-gebeurtenis bewaren we de site zelf in de wachtrij, voor een cirkel-gebeurtenis bewaren we het laagste punt van de cirkel en een verwijzing (*pointer*) naar het blad (in de gebalanceerde binaire boom T) dat de paraboolboog voorstelt die zal verdwijnen door de cirkel-gebeurtenis in kwestie.

Detecteren van cirkel-gebeurtenissen

In tegenstelling tot de site-gebeurtenissen, zijn de cirkel-gebeurtenissen niet op voorhand gekend.

We weten dat een cirkel-gebeurtenis wordt veroorzaakt door het verdwijnen van een paraboolboog uit de strandlijn. Veronderstel nu dat er drie opeenvolgende paraboolbogen van de strandlijn worden beschouwd en dat de middelste paraboolboog verdwijnt door een cirkel-gebeurtenis (merk op dat de oorzaak-gevolg relatie hier omkeerbaar is). Het moment waarop dit gebeurt is wanneer de drie sites, die deze drie paraboolbogen definiëren, een cirkel definiëren waarvan het laagste punt op de veegrechte ligt.

Dus, voor elke drie opeenvolgende paraboolbogen op de strandlijn zal de overeenkomende cirkel-gebeurtenis in de gebeurtenissen-wachtrij Q zitten als de cirkel de veegrechte snijdt. Als de cirkel volledig boven de veegrechte ligt dan werd de cirkel-gebeurtenis reeds afgehandeld. Indien er nog een of meerdere andere sites in de cirkel liggen, dan moet de gebeurtenis *niet* afgehandeld worden.

Wanneer nu een tripel opeenvolgende paraboolbogen wordt vernietigd dan verwijderen we de overeenkomende cirkel-gebeurtenis uit Q als ze voorkomt in Q. Veronderstel nu dat een paraboolboog α uit de strandlijn verdwijnt (of ze wordt gesplitst) en we alle cirkel-gebeurtenissen waarin α betrokken is willen verwijderen. We zoeken daarvoor eerst in T naar een blad μ dat de paraboolboog α definieert. In dit blad zit een verwijzing naar de cirkel-gebeurtenis in Q die vervolgens uit Q verwijderd wordt. De andere twee cirkel-gebeurtenissen waarin α betrokken is, kunnen gevonden worden door respectievelijk het vorige en het volgende blad in T te beschouwen (T is immers een gebalanceerde binaire boom). Ook deze twee bladeren hebben verwijzingen naar knooppunten in Q en die moeten dus ook verwijderd worden.

3.4 Het algoritme in pseudocode

Het algoritme bestaat uit de hoofdroutine VORONOIDIAGRAM(P) die twee subroutines BEHANDELSITEGEBEURTENIS(p_i) en BEHANDELCIRKELGEBEURTENIS(p_l) oproept :

3.4.1 VoronoiDiagram(P)

Invoer : een verzameling $P = \{p_1, p_1, \ldots p_n\}$.

Uitvoer : $Vor(P)$ in een rechthoek (*bounding box*), voorgesteld als een dubbel-verbonden lijst van verbindingen.

Pseudocode :

1. Initialiseer de gebeurtenissen-wachtrij Q met alle site-gebeurtenissen.
2. **Zolang** (Q niet leeg is) **Doe**
3. Beschouw de gebeurtenis met de grootste y-coördinaat in Q.
4. **Als** deze gebeurtenis een site-gebeurtenis is (die optreedt bij site p_i),
5. **Dan** BEHANDELSITEGEBEURTENIS(p_i),
6. **Anders** BEHANDELCIRKELGEBEURTENIS(p_l).
7. Verwijder de gebeurtenis uit Q.
8. De interne knooppunten van de gebalanceerde binaire boom T komen overeen met de halfoneindige verbindingen aan de randen van het Voronoi-diagram. Bereken een rechthoek (*bounding box*) die het Voronoi-diagram volledig omvat en bereken de snijpunten met de verbindingen in het Voronoi-diagram (en verander de datastructuur overeenkomstig).
9. Doorloop nu de datastructuur van het Voronoi-diagram en pas de cellen aan de randen aan.

In stap 6 is p_l het laagste punt op de cirkel dat de gebeurtenis veroorzaakt.

3.4.2 BehandelSiteGebeurtenis(p_i)

Bij deze soort gebeurtenissen worden nieuwe paraboolbogen aan de strandlijn toegevoegd.

Pseudocode :

1. Zoek in de gebalanceerde binaire boom T naar de paraboolboog α die verticaal boven p_i ligt en verwijder uit Q alle cirkel-gebeurtenissen waarin α betrokken is.

2. Vervang in T het blad dat α voorstelt met een subboom die drie bladeren heeft. In het middelste blad wordt de nieuwe site p_i opgeslagen en in de andere twee bladeren wordt de site p_j opgeslagen die oorspronkelijk bij α bewaard was. Bewaar in beide bladeren ook respectievelijk de geordende paren $< p_j, p_i >$ en $< p_i, p_j >$ die de nieuwe snijpunten voorstellen (T *moet eventueel opnieuw gebalanceerd worden*).

3. Maak in de datastructuur van het Voronoi-diagram ook plaats voor de twee halve verbindingen die $V(p_i)$ en $V(p_j)$ scheiden. Deze twee verbindingen zullen door de snijpunten van de paraboolbogen gevolgd worden.

4. Kijk de tripels na van drie opeenvolgende paraboolbogen waarin één van de drie nieuwe paraboolbogen optreedt. Voeg de overeenkomstige cirkel-gebeurtenis toe aan Q indien de gevormde cirkel de veeglijn L snijdt en de cirkel-gebeurtenis nog niet in Q zit.

3.4.3 BehandelCirkelGebeurtenis(p_l)

Bij deze soort gebeurtenissen worden paraboolbogen uit de strandlijn verwijderd.

Pseudocode :

1. Zoek in de gebalanceerde binaire boom T naar de paraboolboog α die verticaal boven p_l ligt *en zal verdwijnen*. Verwijder uit Q alle cirkel-gebeurtenissen waarin α betrokken is.

2. Verwijder uit T het blad dat α voorstelt. Wijzig overeenkomstig hiermee de tupels die de snijpunten van de paraboolbogen voorstellen in de interne knooppunten (T *moet eventueel opnieuw gebalanceerd worden*).

3. Voeg het middelpunt van de cirkel (die de gebeurtenis veroorzaakt) als een vertex toe aan de datastructuur van het Voronoi-diagram en creëer twee halve verbindingen die overeenkomen met het nieuwe snijpunt van het Voronoi-diagram (en verander ook de verwijzingen).

4. Kijk de tripels na van drie opeenvolgende paraboolbogen die ontstaan door de verdwijning van α. Voeg de overeenkomstige cirkel-gebeurtenis toe aan Q indien de gevormde cirkel de veeglijn L snijdt en de cirkel-gebeurtenis nog niet in Q zit.

3.5 De complexiteit van het algoritme

Het algoritme van Fortune heeft een tijdscomplexiteit van :

$$O(n \log(n))$$

en een ruimtecomplexiteit van :

$$O(n).$$

3.6 Gedegenereerde situaties

Vermits het algoritme de gebeurtenissen van boven naar onder afhandelt, kan er degeneratie optreden wanneer twee of meer gebeurtenissen op dezelfde horizontale rechte liggen (wat voorkomt indien er bijvoorbeeld twee sites zijn met dezelfde y-coördinaat). Deze gebeurtenissen kunnen in willekeurige volgorde afgehandeld worden indien de x-coördinaten van beide sites verschillend zijn.

Indien er nu bijvoorbeeld cirkel-gebeurtenissen samenvallen dan kan het gebeuren dat het middelpunt van zo'n cirkel een grote multipliciteit heeft (indien er bijvoorbeeld vier sites op dezelfde cirkel liggen). In plaats van specifieke code te schrijven die deze degeneratie opvangt, laten we het algoritme gewoon zijn werk doen. Het resultaat zal een verbinding opleveren van lengte nul. Nadat het algoritme uitgevoerd is, kunnen we achteraf nog altijd deze soort verbindingen met de hand verwijderen.

Een analoge redering geldt in het geval bij het ontstaan van een nieuwe paraboolboog p_l samenvalt met een zekere p_i.

Bibliografie

[dBvKOS97] M. de Berg, M. van Kreveld, M. Overmars, en O. Schwarzkopf. Voronoi Diagrams. Uit *Computational Geometry : Algorithms and Applications*. Springer-Verlag, 1997.

[Pen00] Rudi Penne. *Algoritmen in meetkundige toepassingen*. UIA - Universitaire Instelling Antwerpen, 2000.

[VVMR92] R. Verhulst, W. Vanneste, C. De Munter, en F. Van Roey. Meetkundige plaatsen. Uit *Analytische meetkunde*. Standaard Educatieve Uitgeverij Antwerpen, 3e editie, 1992.

[YAN98] *Voronoi Diagram*, 1998. Webextract van Congjun Yang.