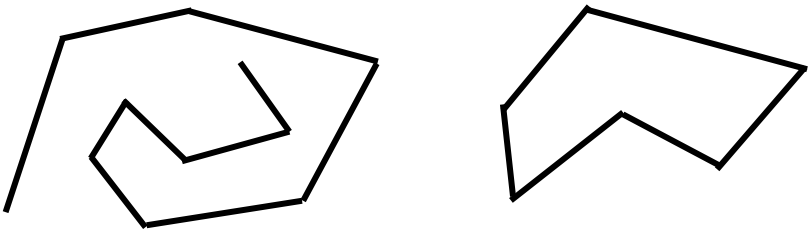
TRIANGULACIONS AL PLA

Polígon

Donats $n+1$ punts $v_0, v_{1,\ldots,} v_n$, anomenats *vèrtexs*, la seqüència de n segments $e_1=v_0v_1,\ldots,$ $e_n=v_{n-1}v_n$, anomenats *costats*, formen una *(cadena) poligonal*, que denotarem C. Identificarem la cadena C amb els vèrtexs que la defineixen i posarem $C=\{ v_0, v_{1,\ldots,} v_n \}$. Una poligonal és *tancada* si el primer i darrer vèrtex coincideixen: $v_0=v_n$.

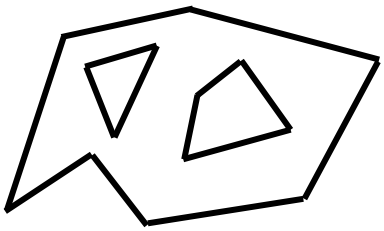
Un *polígon* és la regió del pla delimitada per una poligonal tancada, que s'anomena *frontera* del polígon. Un polígon és *simple* si la seva frontera no s'autointerseca.



Poligonal i polígon simple.

Un polígon de tres costats és un *triangle*, un de quatre costats un *quadrilàter*, ...

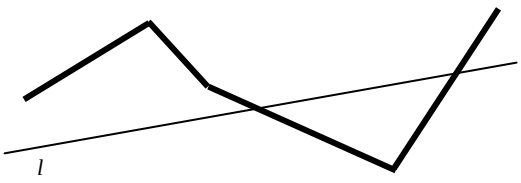
Un polígon amb *forats* és un polígon simple menys l'interior d'altres polígons simples.



Polígon amb forats.

Un vèrtex s'anomena *reflex* si l'angle que determina és estrictament més gran que π, en cas contrai el vèrtex es diu que és *convex*.

Una poligonal C és *monòtona* respecte a una recta l, si qualsevol recta ortogonal a l interseca C en un punt o en un segment. Un polígon P és *monòton* respecte a una recta l, si la seva frontera es pot partir en dues poligonals, cada una de les quals és monòtona respecte a la recta l.



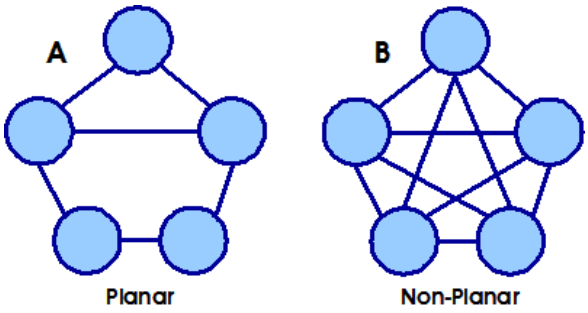
Polígonal monòtona respecte *l*.

Graf

Un *graf* (*geomètric*) $G=<V,A>$ és un objecte format per una col·lecció de punts V, anomenats *vèrtexs* o *nodes*, i per una col·lecció (possiblement buida) de corbes A, anomenades *arestes* o *arcs* , que connecten parells de punts de V.

El *grau* d'un vèrtex v és el nombre d'arestes incidents a v. El grau d'un graf és el grau màxim d'entre els graus dels seus vèrtexs. Un *camí* és una seqüència de vèrtexs v_1 , … , v_k , tal que els vèrtexs v_i i v_{i+1} defineixen una aresta. Un *circuit* és un camí que comença i acaba al mateix vèrtex.

Un graf és *planar* si es pot dibuixar sense que hi hagi encreuaments entre les arestes.

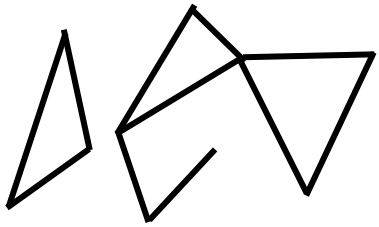


Un *graf planar rectilini* (*planar straight line graph* - PSLG) és un graf planar tal que les seves arestes són segments de recta.

Un graf és *connex* si per a qualsevol parell de vèrtexs hi ha un camí que els connecta.

Un graf $G'=<V',A'>$ és un *subgraf* del graf $G=<V,A>$ si V' és un subconjunt de V i A' és un subconjunt d'A.

Una *component connexa* d'un graf és un subgraf connex *maximal* (no existeix cap altra subgraf connex que el contingui).

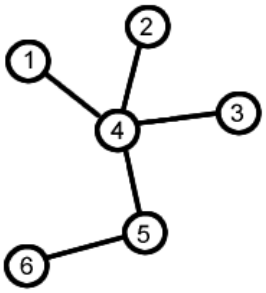


Graf amb dues components connexes.

Arbre

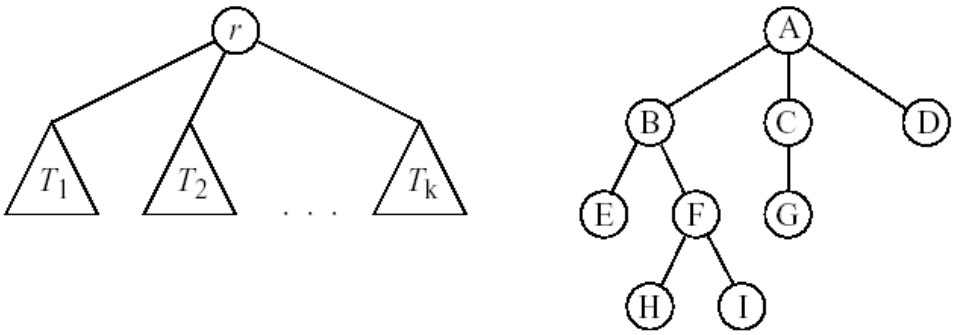
Un *arbre* és un graf connex sense circuits. Dos vèrtexs qualsevols d'un arbre estan connectats per exactament un camí. Un arbre de *n* vèrtexs te exactament *n-1* arestes.

Un graf *G* és un arbre si i solament si és connex i l'elimiació d'una aresta qualsevol de *G* el desconecta.



Arbre.

Un arbre es un graf que es pot definir recursivament. Conté un node distingit *r*, anomenat *arrel*, i un conjunt de zero o diversos subconjunts de nodes, denotats T_1, … , T_k , cadascun dels quals és també un arbre. Aquests arbres s'anomenen *subarbres* de l'arrel.



Representació d'un arbre.

L'arrel de cada subarbre T_1, … , T_k, s'anomena *fill* de *r*, i es diu que *r* és el *pare* de cada una d'aquestes arrels. Una *fulla* en un node que no té fills.

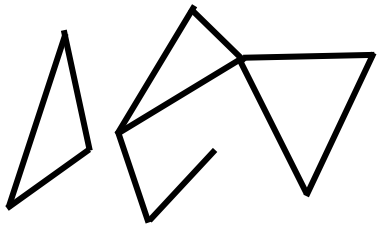
Un *camí* entre dos nodes és una seqüència de nodes u_1 , … , u_k , tal que u_i és pare de u_{i+1}. Si existeix un camí des de *u* a *v* diem que *v* és un *descendent* de *u* i que *u* és l'*ascendent* de *v*. La *longitud* d'un camí és el nombre de les seves arestes. La *profunditat* d'un node és la longitud de l'únic camí que hi ha des de l'arrel al node.

L'arrel té profunditat 0. L'*altura* d'un node és la longitud del camí més llarg que hi ha des del node a una fulla. Per tant totes les fulles tenen altura 0

Subdivisió Planar

Un graf planar determina una partició del pla, anomenada *subdivisió planar*, en regions: 0-dimensionals (*vèrtexs)*, 1-dimensionals (*arestes)*, i 2-dimensionals *(cares)*. Considerarem que els tres tipus de regions són disjuntes: una aresta no inclou els seus extrems, una cara no conté la seva frontera. Hi ha una cara no afitada que s'estén a l'infinit.

Una subdivisió és *connexa* si ho és el graf que la defineix.



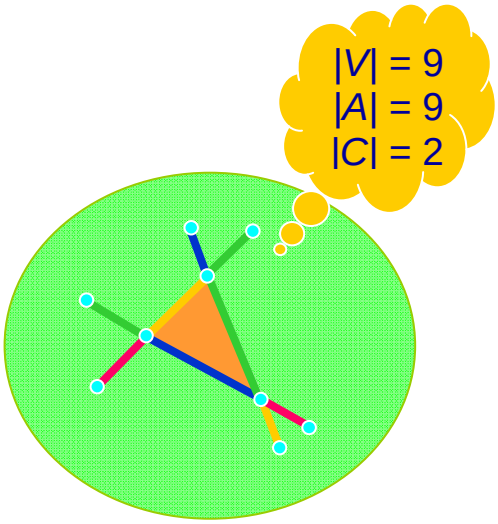
Subdivisió planar desconnexa de quatre cares.

Una subdivisió és *convexa* si totes les seves cares són conjunts convexos.

Una subdivisió es diu que és *maximal* si no es pot afegir cap aresta al graf que la determina sense que es destrueixi la planaritat.

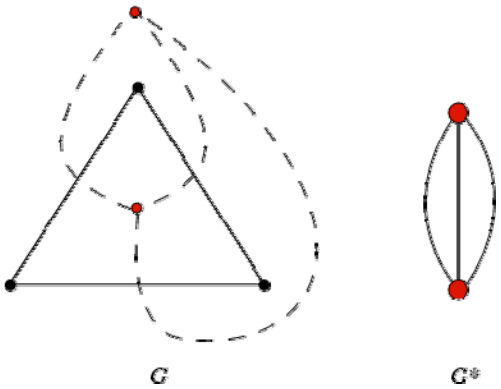
Per a tota subdivisió definida per un graf planar que té $|A|$ arestes, $|V|$ vèrtexs i $|C|$ cares es compleix: $|A| = O(|V|)$ i $|C| = O(|V|)$.

Per a tota subdivisió definida per un graf planar connex que té $|A|$ arestes, $|V|$ vèrtexs i $|C|$ cares, és vàlida la *fórmula d'Euler*: $|V|-|A|+|C| = 2$



Graf dual

Donat un graf planar G, el seu *dual* G^* és el graf que es construeix posant un vèrtex a cada cara de la subdivisió planar determinada per G (incloent-hi la cara exterior) i, si dos cares tenen una aresta en comú a, unint els vèrtexs corresponents amb una aresta a^* que talli només a a.



Representació d'una subdivisió planar

Doubly-connected Edge List (DCEL)

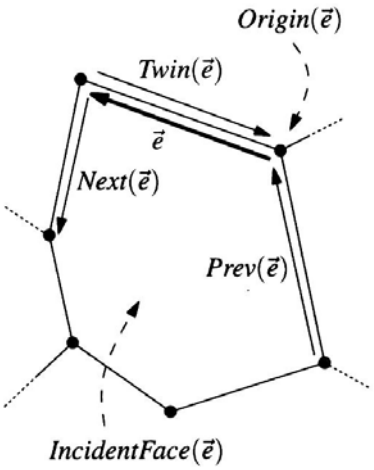
Per a simplificar la descripció de l'estructura DCEL suposarem que les cares de la subdivisió planar a representar no tenen forats. D'aquesta forma, les arestes que delimiten una cara es poden ordenar circularment. Per a veure com es pot eliminar la suposició consultar [de Berg i alt.].

L'estructura DCEL té tres conjunts de registres: *vèrtexs*, *arestes* i *cares*.

Vèrtex v: Per a cada vèrtex de la subdivisió planar s'emmagatzemen les seves coordenades, juntament amb un punter a una aresta orientada qualsevol que tingui el vèrtex com a origen v.inc edge.

Edge e: Cada aresta de la subdivisió planar es representa per dues arestes orientades, que permeten definir sense ambigüitat l'esquerra i la dreta d'una cara. Cada aresta orientada té un punter a la seva aresta orientada oposada, e.twin. Cada aresta orientada té un vèrtex e.origin i un vèrtex e.destination. Cada aresta dirigida té associades dues cares, una a l'esquerra i una altra a la dreta. S'emmagatzema un punter al vèrtex origen e.org (no cal un punter al vèrtex destí, e.dest, perquè es pot definir per e.twin.org). També s'emmagatzema un punter a la cara situada a l'esquerra de l'aresta e.left (observis que podem accedir a la cara situada a la dreta a partir de l'aresta e.twin), anomenada cara incident. També emmagatzemem les arestes orientades següent e.next i prèvia e.prev de la cara incident, en l'ordre al voltant de la cara donat pel sentit negatiu (contrari al moviment de les agulles del rellotge). Els punters next estan ordenats al voltant d'una cara en el sentit negatiu i els punters prev en sentit positiu. Els punters next i prev permeten recórrer totes les arestes d'una cara.

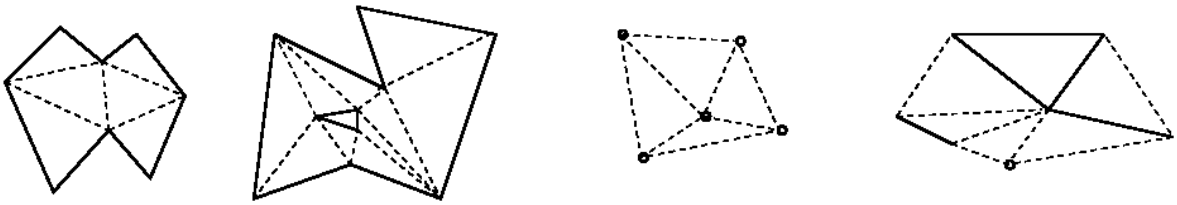
Face f: Cada cara de la subdivisió planar emmagatzema un punter a una aresta orientada per a la qual la cara sigui la cara incident f.inc edge.



Esquema de la DCEL.

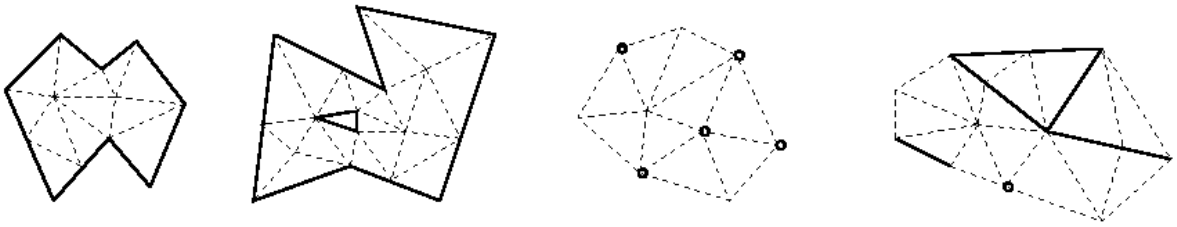
Triangulació

Una *triangulació* és una subdivisió planar d'un domini - típicament definit per un polígon simple o amb forats, un conjunt de punts, o un PSLG – que està formada per un conjunt *T* de cares triangulars, que intersequen només en les arestes comunes i en els vèrtexs.



Triangulació amb punts de Steiner

Una malla triangular és una triangulació del domini formada amb alguns vèrtexs "extres", anomenats punts de Steiner, que s'han afegit per a "optimitzar" algun criteri.

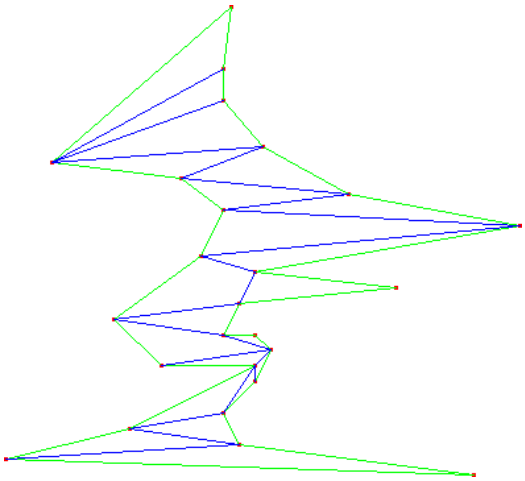


Dual d'una triangulació

Una triangulació es pot pensar com una subdivisió planar determinada per un PSLG maximal G_T, per tant podrem parlar del dual G_T^* d'una triangulació.

Triangulació d'un polígon

Donat un polígon simple *P*, una triangulació de *P* és una subdivisió del seu interior en triangles, de manera que els vèrtexs del polígon siguin vèrtexs dels triangles i viceversa.



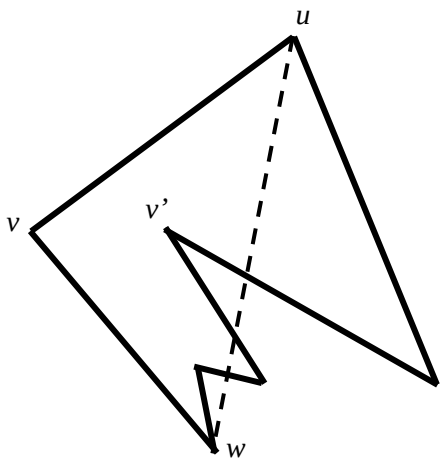
Triangulació d'un polígon.

S'anomena *diagonal* a tot costat d'un triangle que no sigui un costat de *P*.

Teorema: Tot polígon *P* de *n* costats, *n>3*, admet una diagonal.

Prova

Sigui *v* el vèrtex d'abscissa mínima i siguin *vu* i *vw* els costats adjacents a *v*. Si *uw* no interseca *P*, és una diagonal. En cas contrari, almenys hi ha d'haver un vèrtex de *P* dins del triangle *uvw*. Sigui *v'* el vèrtex interior a *uvw* més allunyat de la recta que passa per *u* i *w*. El segment *vv'* és una diagonal perquè en cas contrari qualsevol costat que talles *vv'* hauria de tenir un extrem més allunyat que *v'* de la recta per *u* i *w*.



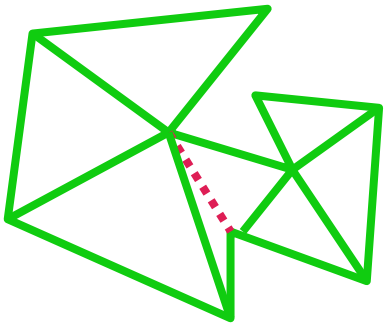
Determinació d'una diagonal.

De la demostració de l'anterior teorema es dedueix que el temps necessari per a trobar una diagonal és *O(n)*.

Teorema: Tot polígon admet una triangulació.

Prova

Partir el polígon per una diagonal i aplicar inducció.



Diagonal d'un polígon.

Teorema: Tota triangulació d'un polígon *P* de *n* costats té *n-2* triangles i utilitza *n-3* diagonals.

Prova:

Per inducció sobre *n*:

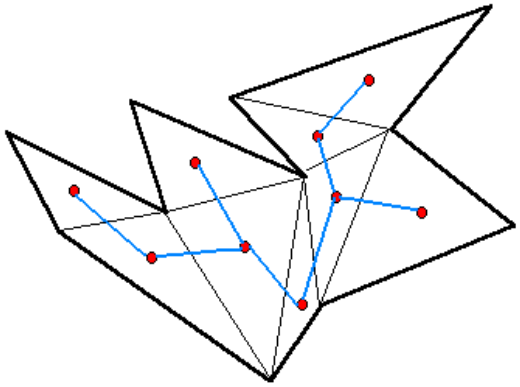
n=3: un triangle té una triangulació (ell mateix) amb un triangle i sense cap diagonal.

n>3: per a un polígon amb *n+1* vèrtexs, construir una diagonal dividint el polígon en dos polígons amb n_1 i n_2 vèrtexs, de manera que tindrem $n_1 + n_2 - 2 = n$. Triangular les dues parts del polígon. Ara hi haurà $n_1 - 3 + n_2 - 3 + 1 = n - 3$ diagonals i $n_1 - 2 + n_2 - 2 = n - 2$ triangles.

Teorema: El dual d'una triangulació d'un polígon és un arbre de grau màxim tres.

Prova

El graf dual és connex, ja que sempre podem anar des d'un triangle a un altre triangle passant per triangles que tenen una diagonal comuna. Com que una diagonal separa el polígon en dues parts, l'eliminació de l'aresta corresponent del graf dual el desconnecta. Per tant el graf dual és un arbre.



Graf dual d'una triangulació.

Algorismes de triangulació d'un polígon simple

Força bruta

Testejar tots els $O(n^2)$ parells de punts en temps O(n) per a trobar una diagonal. Repetir-ho O(n) vegades.

Temps: $O(n^4)$ i espai $O(n)$.

Búsqueda de "orelles"

Tres vèrtexs consecutius *a*, *b* i *c* d'un polígon formen una "orella" si *b* és una diagonal. Buscar una orella en temps O(n), subdividir per la diagonal de l'orella i triangular recursivament.

Temps: $O(n^2)$ i espai $O(n)$.

Òptim [Chazelle]

Temps i espai $O(n)$. Difícil (imposible ?) d'implementar.

Descomposició en polígons monòtons + triangulació polígons monòtons per escombrat

Temps $O(n \log n)$ i espai $O(n)$.

(Lecture Notes D. Mount, pp 31-35)

Algorisme de triangulació d'un polígon convex

Trivial

Unir un vèrtex qualsevol als altres vèrtexs no adjacents.

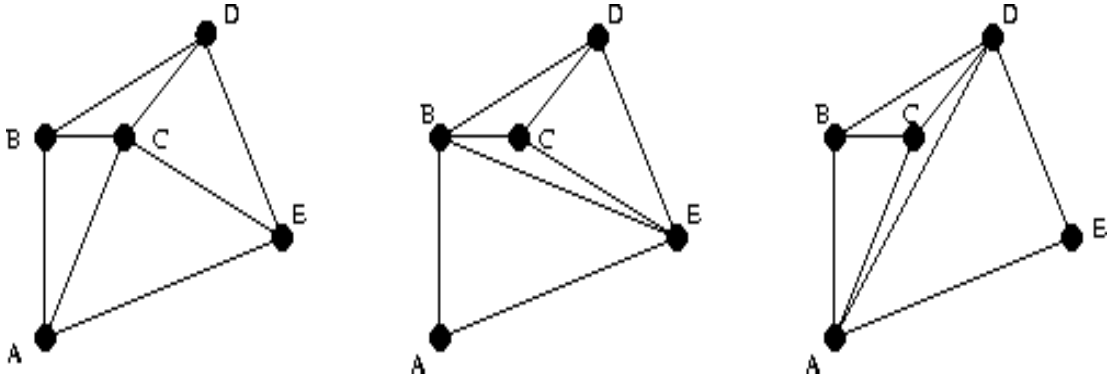
Temps i espai $O(n)$.

Triangulació d'un conjunt de punts

Una *triangulació T* d'un conjunt de punts *S* és una triangulació tal que:

- el conjunt dels vèrtexs dels triangles de *T* coincideix amb *S*,
- la unió dels triangles de *T* és l'envolopant convexa de *S*.

Un mateix conjunt de punts pot tenir diferents triangulacions:



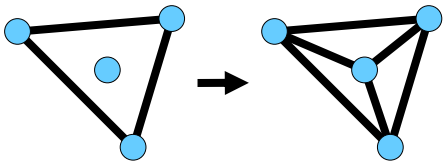
Teorema: tot conjunt de punts *S* es pot triangular

Prova

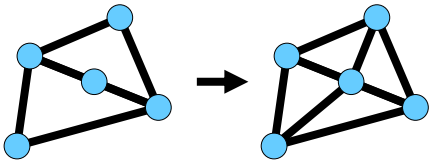
1. Construir l'envolopant convexa *CH(S)*, i connectar un vèrtex arbitrari a tots els altres.

2. Insertar els altres punts un a continuació de l'altre. Hi ha dues possibilitats:

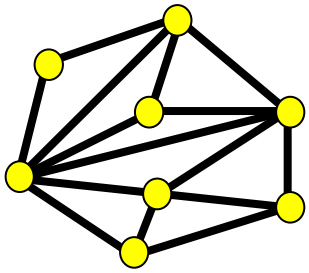
- El punt està dins d'un triangle (el triangle es parteix en tres).



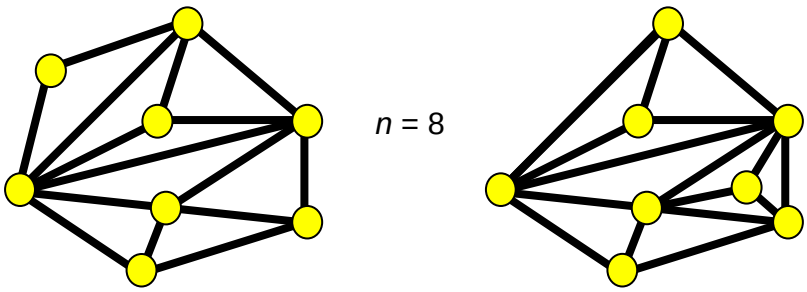
- El punt està a un costat (dos triangles es parteixen en quatre).



Triangulació resultant:



El nombre *t* de triangles i el nombre *a* d'arestes d'una triangulació de *n* punts depèn del nombre de vèrtexs *h* de l'envolopant convexa: $t = 2n-h-2$; $a=3n-h-3$



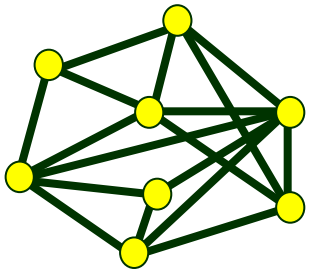
$h = 6 \rightarrow t = 8$; $a = 15$ $h = 5 \rightarrow t = 9$; $a = 16$

Algorismes de triangulació d'un conjunt de punts

Força bruta

Repetir mentre sigui possible:

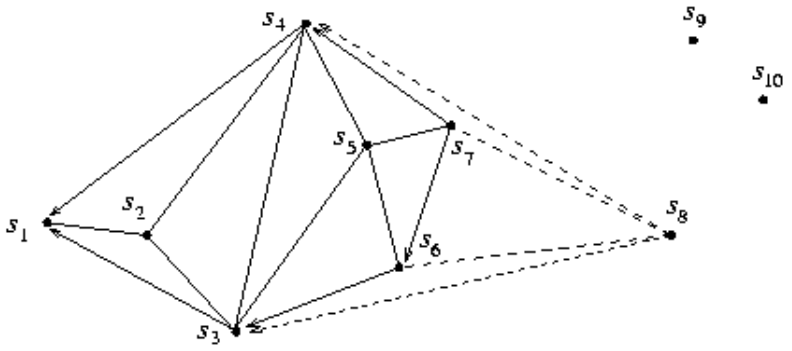
- Agafar dos punts.
- Si l'aresta que connecta els punts no interseca cap aresta seleccionada anteriorment, seleccionar-la.



Temps $O(n^3)$ i espai $O(n)$.

Algorisme d'escombrat

L'algorisme afegeix els punts de S a mesura que els va trobant una recta vertical que es va desplaçant d'esquerra a dreta. Cada vegada que s'afegeix un punt s, l'algorisme recorre l'envolopant convexa dels punts prèviament afegits, començant pel punt de l'envolopant convexa situat més a la dreta, primer en un recorregut en direcció cap amunt i després en un recorregut en direcció cap avall. Durant cada un dels dos recorreguts, afegeix a la triangulació un costat que va des de s a cada un dels punts de l'envolopant convexa trobats, fins que el signe de la pendent d'aquests costats canvia.



Algorisme d'escombrat per a calcular una triangulació.

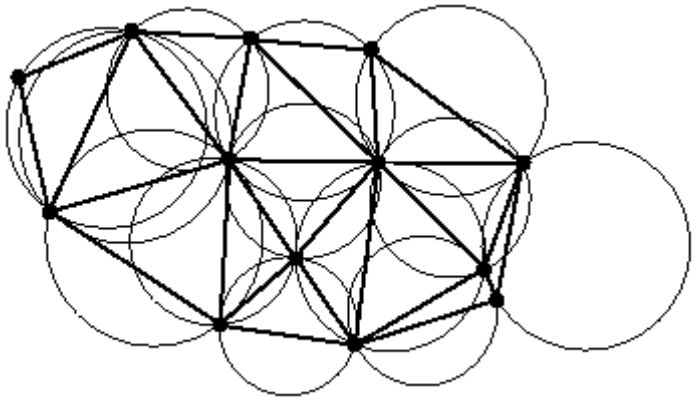
Temps $O(n \log n)$ i espai $O(n)$.

Triangulació de Delaunay d'un conjunt de punts

Un conjunt de punts *S* és *no degenerat* si no hi ha tres punts alineats, ni quatre punts que pertanyin a una mateixa circumferència. A partir d'ara suposarem que *S* és no degenerat.

Un triangle definit per tres punts de *S* és un *triangle de Delaunay* si la circumferència que el circumscriu no conté cap punt de *S* al seu interior.

La unió de tots els triangles de Delaunay forma la *triangulació de Delaunay*, *DT(S)*, de *S*. La triangulació de Delaunay és única. Tot conjunt de punts admet una triangulació de Delaunay.

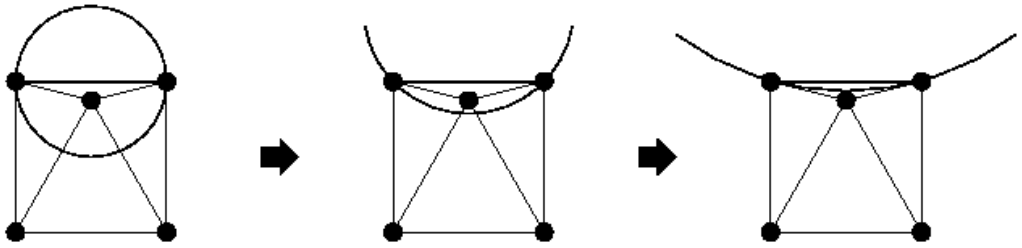


Triangulació de Delaunay. Tots els triangles són Delaunay.

Donats *p* i *q* de *S*, el segment *pq* és un *costat de Delaunay* si existeix una circumferència per *p* i *q* que no conté cap altre punt de *S*.

Teorema: Tots els costats de *DT(S)* són costats de Delaunay, i una triangulació de *S* amb tots els costats de Delaunay és la triangulació de Delaunay.

Tot costat de l'envolopant convexa *CH(S)* de *S* és un costat de Delaunay: considerar la menor circumferència que conté el costat de *CH(S)* i fer-la créixer cap a l'exterior de *CH(S)* fins que sigui buida de punts de *S*.

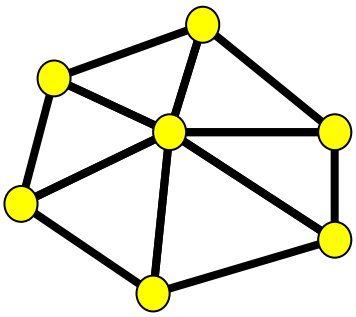


Propietats de la triangulació de Delaunay:

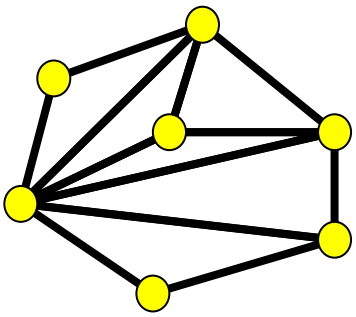
- Cada punt de *S* connecta amb els seus veïns més pròxims (el cercle més petit que conte el punt i el veí és buit).

- Maximitza l'angle mínim de la triangulació.

- Maximitza el vector d'angles de la triangulació. Sigui $\alpha(T) = (\alpha_1, ... , \alpha_{3t})$ el vector d'angles de la triangulació *T*, d'un total de *t* triangles, en ordre creixent. Una triangulació T_1 direm que és "millor" que una triangulació T_2 si $\alpha(T_1) > \alpha(T_2)$ lexicogràficament. La triangulació de Delaunay és la "millor".



Bo

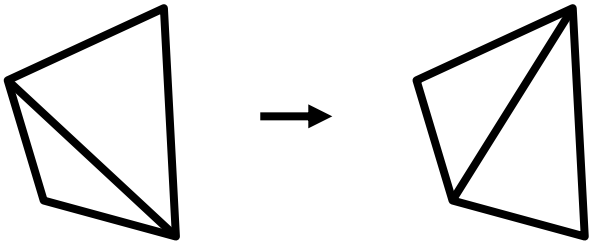


Dolent

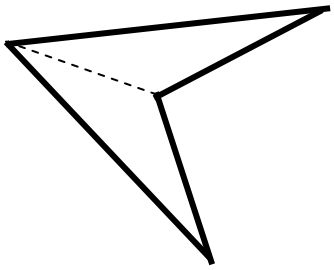
Intuïtivament, *DT(S)* d'entre totes les possibles de *S* és la que té els triangles més equiangulars.

Flip

En un quadrilàter convex, "flipar" una diagonal consisteix en substituir-la per l'altra diagonal:

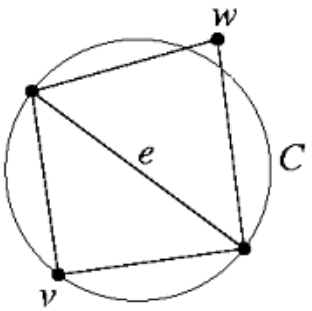


Dos triangles adjacents d'una triangulació *T* de *S* defineixen un quadrilàter. Si aquest quadrilàter és convex, les dues diagonals són flipables entre elles.



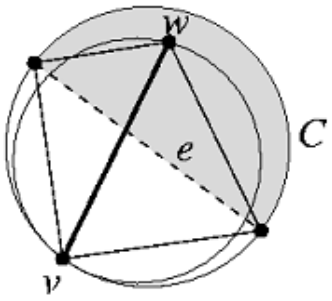
Diagonal no flipable.

Un costat *c* d'una triangulació *T* de *S*, que no sigui de *CH(S),* és *localment Delaunay* si existeix una circumferència que passa pels vèrtexs de *c* que no conté cap dels dos vèrtexs oposats a *c* en els dos triangles adjacents que comparteixen *c*. Tots els costats de *CH(S)* es consideren localment Delaunay.



El costat *e* és localment Delaunay.

Teorema: tot costat *c* d'una triangulació *T* de *S* és localment Delaunay o, en cas contrari, és flipable i el nou costat creat pel flip és localment Delaunay.



Costat *e* no localment Delaunay; al flipar-lo passa a ser localment Delaunay.

Convé observar que un costat localment Delaunay no és necessàriament un costat Delaunay, no obstant es verifica el següent:

Teorema: Si una triangulació *T* de *S* que té tots els costats localment Delaunay, llavors tots els costats de *T* també són Delaunay.

Com a conseqüència del Teorema anterior, si una triangulació conté un costat que no és Delaunay, ha de contenir un costat que no és localment Delaunay, i per tant pot ser flipat. Aquesta és la base de l'anomenat algorisme de flipat que fa servir flips per a convertir una triangulació arbitrària *T* de *S* en la triangulació de Delaunay *DT(S)*.

Algorismes de càlcul de la triangulació de Delaunay

Força bruta

Repetir mentre sigui possible:

- Agafar tres punts.
- Si la circumferència que passa pels tres punts no conté cap altre punt de *S* seleccionar el triangle determinat pels tres punts.

Temps: $O(n^4)$ i espai $O(n)$.

Millora local per flipat [Lawson]

L'algorisme de flipat parteix d'una triangulació qualsevol de *S* i aplica una seqüència de "bons" flips fins a obtenir *DT(S)*.

L'algorisme fa servir una cua que conté els costats de la triangulació que han de ser tractats. Per a evitar duplicats a la cua es marquen els costats que ja estan guardats.

Donat un costat *e* d'una triangulació de *S* que no pertany a l'envolopant convexa de *S* (*costat intern*), denotarem Q_e el quadrilàter format pels dos triangles que comparteixen el costat *e*.

Calcular una triangulació inicial de *S*
Guardar els costats interns a una cua
Mentre la cua no sigui buida **fer**
    Agafar el primer costat *e* de la cua
    Si *e* no és localment Delaunay **llavors**
        Flipar *e* amb l'altra diagonal de Q_e i afegir els costats de Q_e a la cua
    Fisi
Fimentre

Temps $O(n^2)$ i espai $O(n)$. Es comporta força be a la pràctica.

Dividir i vèncer

- Dividir *S* en dos subconjunts, amb aproximadament *n/2* punts cadascun, per mitjà d'una recta vertical.
- Construir recursivament DT a cada subconjunt.
- Ajuntar les dues triangulacions en una sola triangulació.

Temps: $O(n \log n)$ i espai $O(n)$.

Inserció incremental

- Insertar els punts d'un en un.
- Mantenir la triangulació de Delaunay de tots els punts insertats (mitjançant: flips [Lawson], retriangulació polígon [Watson]).

Temps $O(n^2)$ i espai $O(n)$.

Inserció incremental aleatori

Veure detalls a curs Dave Mount

Temps $O(n \log n)$ i espai $O(n)$.

Implementacions eficients a *Triangle*, a la WEB de J. Shewchuk:

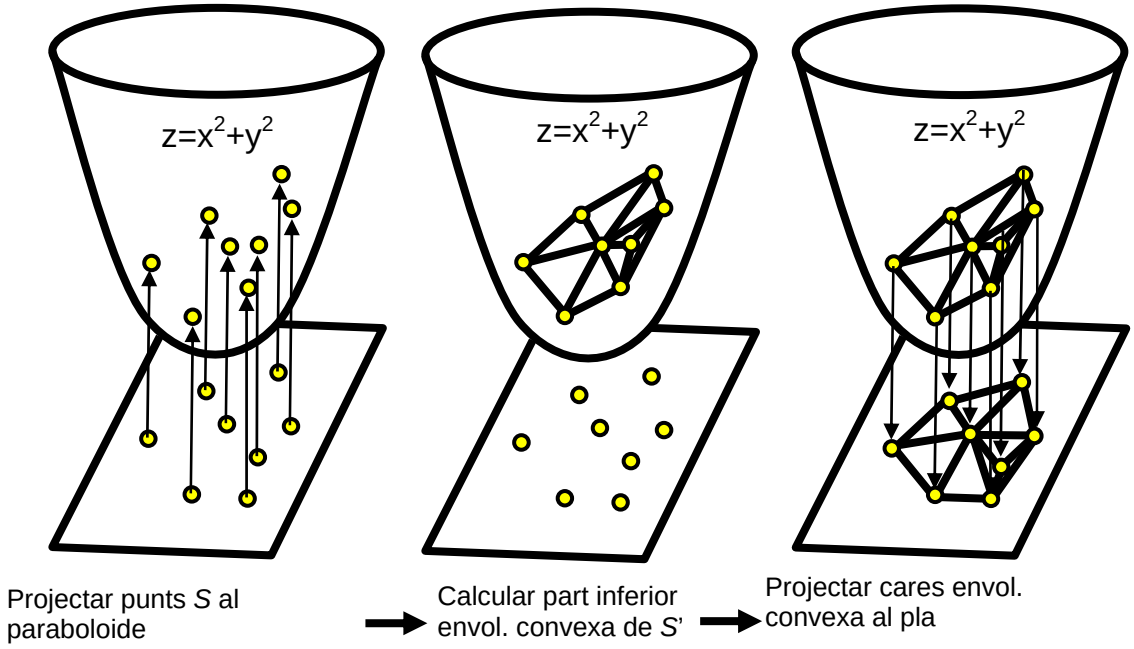
http://www-2.cs.cmu.edu/~quake/triangle.html

Relació entre triangulació de Delaunay al pla i envolopant convexa a l'espai

Envolopant convexa a l'espai

L'envolopant convexa d'un conjunt de punts a l'espai es defineix de forma semblant a com s'ha fet al pla. Es pot calcular en temps *O(n log n)* i espai *O(n)*.

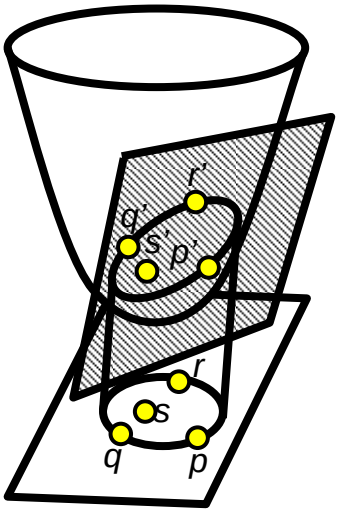
Transformació geomètrica:



Teorema: La triangulació obtinguda al pla és la triangulació de Delaunay

Prova

- La intersecció del paraboloide amb un pla és una el·lipse que es projectar al pla en una circumferència.



- El punt *s* és interior a la circumferència per *p, q, r* si i solament si *s*' està situat sota del pla que passa per *p*', *q*', *r*'.

- *p, q, r* $\in$ *S* determinen un triangle de Delaunay si i solament si *p*', *q*', *r*' determinen una cara de l'envolopant convexa de *S'*.

El mètode descrit ens proporciona una altra alternativa per a calcular la triangulació de Delaunay en temps *O(n log n)* i espai *O(n).*

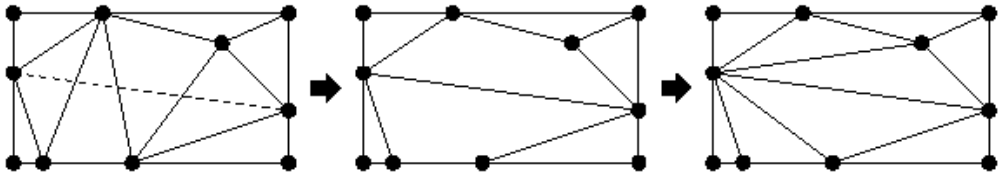
Triangulació restringida d'un PSLG

Una triangulació restringida d'un PSLG *G* és una triangulació dels vèrtexs de *G* tal que les arestes de *G* són costats de la triangulació.

Teorema: Tot PSLG *G* es pot triangular

Prova

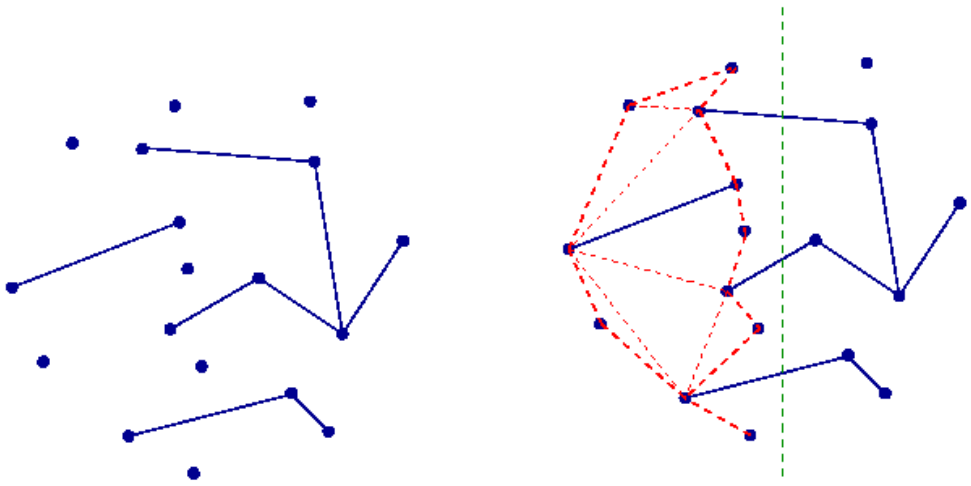
Començar amb una triangulació inicial dels vèrtexs de *G*. A continuació examinar cada aresta *e* de *G* per a veure si forma part de la triangulació. Si *e* no forma part de la triangulació, eliminar tots els costats de la triangulació que tallen *e*, insertar *e* com a nou costat de la triangulació, i retriangular els dos polígons resultants (un a cada costat de *e*) com s'indica a la figura següent:



Inserció d'un segment de *G* a la triangulació.

Algorisme d'Escombrat

L'algorisme per a calcular una triangulació restringida de *G* és una mica més complicat que l'algorisme d'escombrat que construeix una triangulació d'un conjunt de punts *S*. Mentre que aquest manté l'envolopant convexa dels punts de *S* que queden a l'esquerra de la recta d'escombrat, en el cas restringit cal mantenir una envolopant convexa pels vèrtexs de *G* situats l'esquerra de la recta d'escombrat, per a cada interval de la recta d'escombrat determinat per dos segments de *G* intersecats consecutivament.



Algorisme d'escombrat.

Temps *O(n log n)* i espai *O(n).*

Triangulació restringida de Delaunay d'un PSLG

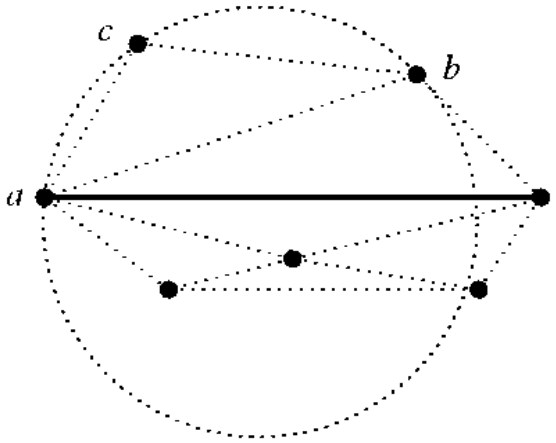
Un PSLG *G* és *no degenerat* si no hi ha tres vèrtexs de *G* alineats i quatre que pertanyin a una mateixa circumferència. A partir d'ara suposarem que *G* és no degenerat.

Direm que dos vèrtexs *p* i *q* de *G* són *mútuament visibles* si:

- el segment *pq* no conté cap vèrtex de *G* al seu interior;
- l'interior del segment *pq* no interseca cap aresta de *G*.

Un triangle definit per tres vèrtexs no alineats de *G* és un *triangle restringit de Delaunay* si la circumferència que el circumscriu no conté cap vèrtex de *G* que sigui visible des de l'interior del triangle.

La unió de tots els triangles restringits de Delaunay forma la *triangulació restringida de Delaunay*, C*DT(G)*, de *G*. La triangulació restringida de Delaunay *CDT(G)* és única.



Triangulació restringida de Delaunay. El triangle *abc* és restringit Delaunay.

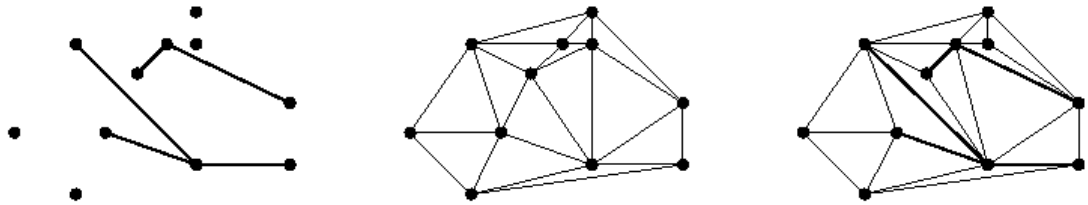
Donats *p* i *q* de *G*, el segment *pq* és un *costat restringit de Delaunay* si *p* i *q* són mútuament visibles i si existeix una circumferència per *p* i *q* que no conté cap altre vèrtex de *G* visible des de l'interior de *pq*. Les arestes de *G* es consideren costats restringits de Delaunay**.**

Teorema: Tots els costats de C*DT(G)* són costats restringits de Delaunay, i una triangulació de *G* amb tots els costats restringits de Delaunay és la triangulació restringida de Delaunay.

Tot costat de l'envolopant convexa dels vèrtexs de *G* és un costat restringit de Delaunay.

A una triangulació restringida de Delaunay, a diferència del que passa amb una triangulació de Delaunay, i poden aparèixer triangles amb un angle petit.

En general, la triangulació restringida de Delaunay d'un PSLG i la triangulació de Delaunay dels vèrtexs del PSLG no coincideixen.



a) PSLG; b) Triangulació Delaunay de V; c) Triangulació restringida Delaunay de G

Algorismes de càlcul de la triangulació restringida de Delaunay

Força bruta

Repetir mentre sigui possible:

- Agafar tres vèrtexs *p, q* i *r* de *G*.
- Si la circumferència que passa pels tres punts no conté cap altre punt de *G* visible des de l'interior del triangle *pqr*, seleccionar el triangle.

Temps $O(n^4)$ i espai $O(n)$.

Flipat

Es semblant al algorisme que fa servir flips per a calcular la triangulació de Delaunay.

Temps $O(n^2)$ i espai $O(n)$.

Dividir i vèncer

- Dividir *G* en dos PSLGs per mitjà d'una recta vertical.
- Construir recursivament CDT per a cada subgraf.
- Ajuntar les dues triangulacions en una sola triangulació

Temps: $O(n \log n)$ i espai $O(n)$.

La CDT d'un polígon simple es pot tractar com la d'un PSLG, removent les arestes de la triangulació exteriors al polígon.

Si el polígon és convex hi ha un algorisme lineal per a calcular el CDT [Aggarwal].

Hi ha un algorisme aleatori que construeix el CDT d'un polígon simple en temps lineal [Klein,Lingas].

Triangulació conforme d'un PSLG

Donat un PSLG $G=<V,A>$, i un conjunt de punts $W \supset V$, una triangulació de W és conforme a G si tota aresta d'A és pot representar com la unió de costats de la triangulació.

Teorema: Per tot PSLG $G=<V,A>$ existeix un conjunt de punts W la triangulació de Delaunay del qual és conforme a G, de manera que els vèrtexs de W-V pertanyen a les arestes d'A.

Calcular una triangulació conforme Delaunay d'un PSLG $G=<V,A>$, es redueix al problema de trobar un conjunt de punts V' de manera que la triangulació de Delaunay de $W=V \cup V'$ sigui conforme a G.

Algorismes de càlcul d'una triangulació conforme Delaunay d'un PSLG

Edelsbrunner-Tan

Demostren que el nombre màxim de punts a afegir és afitat superiorment per $O(l^2n)$, essent n i l el nombre de vèrtexs i arestes de G.

Temps: $O(l^2n+ n^2)$.

Referències

Llibres:

de Berg, van Kreveld, Overmars, and Schwarzkopf
Computational Geometry (Second Edition),, 2000.
http://www.cs.uu.nl/geobook/

J. O'Rourke,
Computational Geometry in C (Second Edition),
Cambridge University Press, 1998.
http://cs.smith.edu/~orourke/books/compgeom.html

Articles:

Marshall Bern and David Eppstein.
Mesh generation and optimal triangulation.
Computing in Euclidean Geometry, World Scientific, 1995.
http://www.ics.uci.edu/~eppstein/pubs/BerEpp-CEG-95.pdf

Marshall Bern and Paul Plassmann.
Mesh generation.
Handbook of Computational Geometry, Elsevier, 2000.
http://citeseer.nj.nec.com/bern00mesh.html

Cursos:

Dave Mount
Computational Geometry
http://www.cs.umd.edu/~mount/754/

Jonathan Shewchuk
Meshing and Triangulation in Graphics, Engineering, and Modeling
http://www.cs.berkeley.edu/~jrs/mesh/