5. VISUALIZZAZIONE DI STRUTTURE MOLECOLARI

Riferimenti: [Dal99] pp 303-324 escluso par. 10.3, [C04] pp 77-80.
Per approfondire: [KW98] pp. 71-86.

Gli algoritmi di cui si tratterà nel seguito si basano sul fatto che l'oggetto da visualizzare è un sistema di corpi tra cui agiscono delle forze. L'idea è quella di cercare una configurazione dei corpi in modo da minimizzare l'energia locale, cioè in modo che la somma delle forze su ogni corpo sia 0. In altre parole ancora, si può dire che si visualizza uno *stato di equilibrio*.

Queste tecniche sono molto usate anche al di fuori della rappresentazione delle strutture molecolari perché, in realtà, si prestano bene per visualizzare qualunque grafo, che scaturisca da un modello fisico oppure no.

In realtà, sono molti gli algoritmi che si basano sull'interpretazione degli archi come forze, ma tutti sono costituiti da due parti fondamentali:

- il modello: un sistema di forze che fornisce il modello fisico del grafo. Si osservi che è equivalente pensare al sistema di forze o al sistema di energia che ne deriva;
- l'algoritmo: la tecnica per trovare uno stato di equilibrio.

5.2. Algoritmi basati sulla ricerca del punto di minimo delle forze di interazione

ALGORITMO I

Vediamo ora l'algoritmo di visualizzazione più semplice: gli archi del grafo sono modellati come molle, mentre i nodi sono tutti ugualmente caricati elettricamente e quindi si respingono l'un l'altro. In particolare, quindi, su ogni nodo agiscono delle forze la cui risultante è:

$$\vec{F}(v) = \sum_{u \in V} \vec{g}_{uv} + \sum_{(u,v) \in E} \vec{f}_{uv}$$

dove $\vec{f}_{uv}$ è la forza esercitata su v dalla molla tra u e v, mentre $\vec{g}_{uv}$ è la forza di repulsione elettrica esercitata su v da ciascun altro nodo u. Possiamo specializzare la formula della forza inserendo i seguenti parametri:

- $\overrightarrow{d(u,v)}$ è la distanza euclidea tra u e v;
- $\overrightarrow{l(u,v)}$ è la lunghezza a riposo della molla tra u e v (corrispondente a forza nulla);
- k_{uv} è la costante che indica la potenza della molla: più k_{uv} è grande, più la molla tende a raggiungere la sua posizione a riposo;
- c_{uv} è la costante relativa alla forza di repulsione e ingloba anche le quantità di carica.

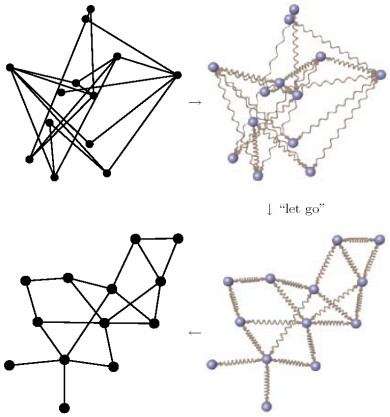
La forza $\vec{f}_{uv}$ segue la legge di Hooke, cioè è proporzionale alla differenza tra la distanza tra u e v e la lunghezza a riposo della molla: $\vec{f}_{uv} = k_{uv}(\overrightarrow{d(u,v)} - \overrightarrow{l(u,v)})$; la forza $\vec{g}_{uv}$ invece decresce come l'inverso del quadrato della distanza: $\vec{g}_{uv} = \frac{c_{uv}}{\overrightarrow{d(u,v)}^2}$.

Una volta definito il sistema di forze, bisogna cercare lo stato di equilibrio del sistema, cioè $\forall v \in V$ bisogna porre $\vec{F}(v) = 0$. Ci sono numerose tecniche numeriche per fare questo, tra cui la più semplice consiste nel posizionare a caso i nodi sul piano, calcolare per ciascuno la forza a cui sono soggetti e muovere ogni nodo nella direzione della forza di una distanza proporzionale alla forza esercitata. Questo metodo non è il più veloce ma, con un tool di animazione, permette di seguire gli spostamenti dei nodi nel modo più intutivo.

Osserviamo che questo algoritmo dà risultati molto buoni riguardo ai criteri estetici grazie alle proprietà degli stati di equilibrio delle forze che abbiamo messo in gioco: i nodi sono distribuiti

uniformemente nell'area del disegno perché la forza della molla garantisce che la distanza tra u e v tenda ad essere quella relativa alla molla a riposo – quindi non troppo grande, e la forza elettrica garantisce che i nodi non si avvicinino mai troppo. Inoltre, l'esperienza mostra che, se ci sono simmetrie nel grafo, esse vengono evidenziate. In Figura 5.1. è mostrato un esempio di funzionamento di questo metodo.

Figura 5.1.



Tuttavia, questo metodo non garantisce di giungere ad un punto di equilibrio in un certo numero di passi, e quindi non si può stimare il suo tempo di esecuzione; inoltre, non viene garantita la convergenza: a causa del fatto che lo spostamento da effettuare ad ogni iterazione è di una certa dimensione fissata, ci sono situazioni in cui un nodo "rimbalza" attorno ad un punto per un tempo infinito, senza arrivare mai ad una situazione di equilibrio (che, per esempio, si trova ad una distanza minore rispetto allo spostamento fissato). Per evitare tale situazione si tronca l'algoritmo dopo un certo tempo fissato. Questo non garantisce di arrivare ad un punto di equilibrio esatto, ma ragionevolmente vicino.

Nelle figure 5.2 e 5.3 è presentato un esempio di funzionamento dell'algoritmo precedente: sono fissati 6 stadi del processo iterativo.

Figura 5.2.

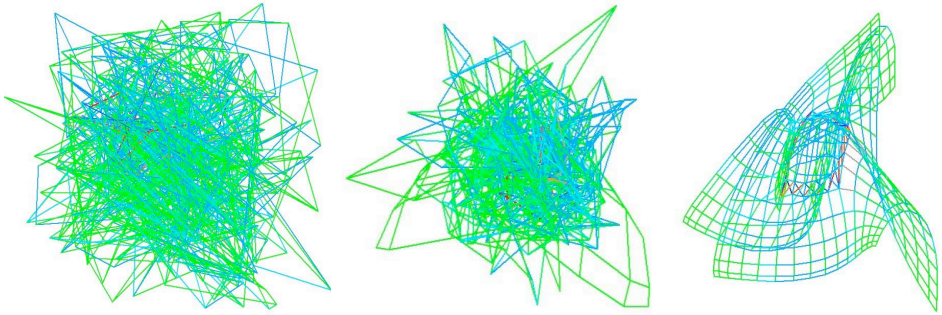
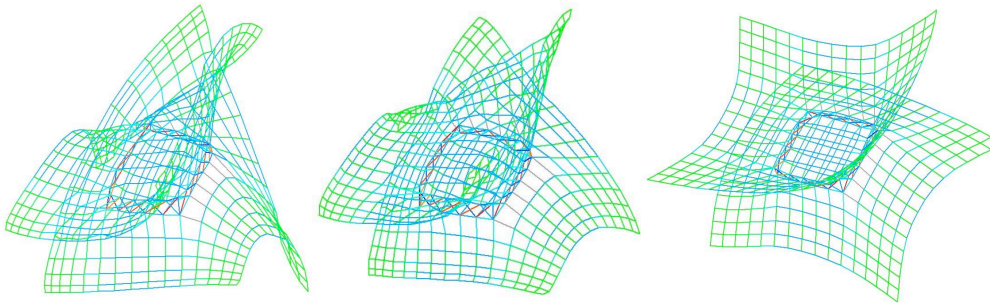


Figura 5.3.



ALGORITMO II [T60]

Questo algoritmo è uno dei primi ed è dovuto a Tutte. Esso può essere presentato come un caso particolare dell'algoritmo precedente: il modello di Tutte usa molle con lunghezza a riposo nulla ($\overrightarrow{l(u,v)}=0$), quindi la forza è proporzionale a $\overrightarrow{d(u,v)}$; il parametro della molla k_{uv} è settato ad *1* per ogni arco *(u, v)*; infine, non ci sono forze elettriche. Quindi la forza si può scrivere come:

$$\vec{F}(v) = \sum_{(u,v)\in E} \overrightarrow{d(u,v)}.$$

Tra le soluzioni di equilibrio compare anche quella banale in cui tutti i nodi sono nell'origine, ma tale soluzione non genera un buon disegno. Per evitare di imbattersi in questa soluzione, l'insieme *V* dei nodi viene partizionato in un insieme V_1 di nodi detti *fissi*, e un altro insieme V_2 di nodi detti *liberi*. Per definizione di partizione si ha: $V=V_1 \cup V_2$ e $V_1 \cap V_2=\emptyset$. E' come se i nodi fissi fossero inchiodati al piano, per cui le forze delle molle non hanno effetto su di loro. Di solito essi vengono posti sui vertici di un poligono convesso e si pone $|V_1| \geq 3$. Tutti i nodi liberi devono essere posizionati in modo tale che la risultante delle forze agenti su di essi sia nulla. Consideriamo l'equazione già citata e poniamola =0:

$$\vec{F}(v) = \sum_{(u,v)\in E} \overrightarrow{d(u,v)} = 0$$

Ora proiettiamola sugli assi cartesiani:

$$\begin{cases} F_x(v) = \sum_{(u,v)\in E} (x(u) - x(v)) = 0 \\ F_y(v) = \sum_{(u,v)\in E} (y(u) - y(v)) = 0 \end{cases}$$

La prima delle due equazioni si può riscrivere come:

$$F_x(v) = \sum_{(u,v)\in E} x(u) - \deg(v)x(v) = 0$$

e la seconda equazione è analoga.

Risolvendo le due equazioni si ottiene:

$$\begin{cases} x(v) = \dfrac{\sum_{(u,v)\in E} (x(u))}{\deg(v)} \\ y(v) = \dfrac{\sum_{(u,v)\in E} (y(u))}{\deg(v)} \end{cases}$$

Poiché *u* varia tra tutti gli adiacenti di *v*, risolvere queste equazioni equivale a posizionare ciascun nodo nel baricentro dei suoi vicini, e questa è la ragione per cui quasto metodo è detto *metodo del baricentro*.

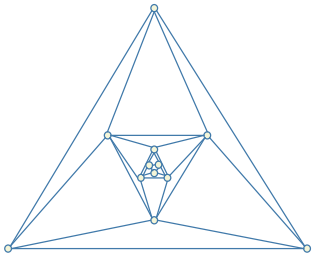
Si osservi che le coordinate dei nodi fissi sono note, e che le equazioni sono lineari; inoltre, il numero di equazioni uguaglia il numero di incognite (pari al numero di nodi liberi); infine, la matrice risultante dal sistema è diagonalmente dominante e quindi si può usare efficientemente un metodo di analisi numerica, che converge in $O(n^2)$ tempo.

E' stato dimostrato da Lipton, Rose, Tarjan [LRT79] che se il grafo è planare la matrice risultante è sparsa e le equazioni si possono risolvere in $O(n^{1.5})$.

Un lato molto interessante di questo metodo è che, se il grafo in input è planare e triconnesso, allora il disegno in output è piano e convesso, cioè ogni faccia è un poligono convesso (dimostrazione in [T60] e [T63]).

Il problema del metodo del baricentro è che esso produce spesso rappresentazioni a bassa risoluzione (v. fig. 5.4.). In effetti, è stato dimostrato che questa è una caratteristica di questo metodo: Eades e Garvan [EG96] hanno mostrato che, per ogni $n>1$ esiste un grafo per cui il metodo del baricentro produce un disegno di area esponenziale.

Figura 5.4.



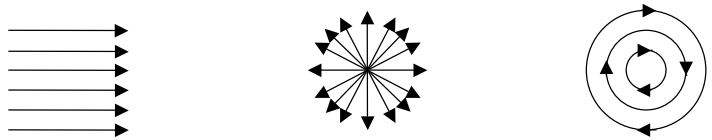
5.2. Generalizzazioni

Gli algoritmi appena visti funzionano bene per certi aspetti, ma male per altri. In particolare, difficilmente i loro output tengono conto di convenzioni e criteri estetici. Per cercare di ovviare a questi problemi sono state proposte alcune generalizzazioni.

5.2.1. Generalizzazione di Sugiyama e Misue

Sugiyama e Misue [SM95] hanno proposto delle varianti del modello, in cui alcune o tutte le molle sono magnetizzate, e c'è un campo magnetico che agisce su di esse. Il campo magnetico può essere usato per controllare l'orientamento degli archi e, quindi, per modellare e gestire dei criterici estetici.

Figura 5.5.



Sono stati considerati tre tipi di campi magnetici, quello parallelo, quello radiale e quello concentrico (vedi fgura 5.5); essi possono anche essere combinati tra loro, e a seconda del campo magnetico in cui il modello viene immerso, si forza una certa convenzione. Ad esempio, combinando un campo orizzontale ed uno verticale si forza un disegno ortogonale.

Le molle possono essere magnetizzate in diversi modi:

- unidirezionalmente (tendono ad allinearsi con il campo)
- bidirezionalmente (come prima, ma in uno qualunque dei due versi)
- nessuna magnetizzazione (il campo non influisce).

Il campo magnetico induce sulle molle magnetizzate una forza rotazionale che si combina con le forze delle molle e con le forze elettriche descritte in precedenza.

In questo modo, magnetizzando le molle in modo unidirezionale, otteniamo un disegno upward (downward, leftward o rightward) se il campo è parallelo, outward se il campo è radiale, antiorario se il campo è concentrico.

5.2.2. Generalizzazione di Davidson e Harel

Nei metodi descritti, la funzione di energia è una funzione semplice e continua delle coordinate dei nodi. Tuttavia, molti criteri estetici (come il numero di incroci) non sono continui. Includendo funzioni di energia discrete, è possibile allargare la classe dei criteri estetici da considerare.

Poiché, come abbiamo già osservato, è impossibile ottimizzare il disegno secondo tutti i criteri, quello che possiamo fare è definire su tutto il disegno una funzione energia che sia combinazione lineare di misure, che rappresentano i vari criteri estetici: $\vec{F}(v) = l_1\vec{F}_1(v) + l_2\vec{F}_2(v) + \ldots + l_k\vec{F}_k(v)$ dove le $\vec{F}_i$ sono misure dei vari criteri estetici, mentre le l_i sono delle costanti, e $\vec{F}$ che è la risultante di tutte le forze che agiscono su tutti i nodi, rappresenta la "bruttezza" del disegno: più esso è grande in modulo e meno i criteri estetici sono soddisfatti.

Davidson e Harel [DH96], ad esempio, hanno definito F al modo seguente:

$$\vec{F} = l_1\vec{F}_1 + l_2\vec{F}_2 + l_3\vec{F}_3 + l_4\vec{F}_4$$

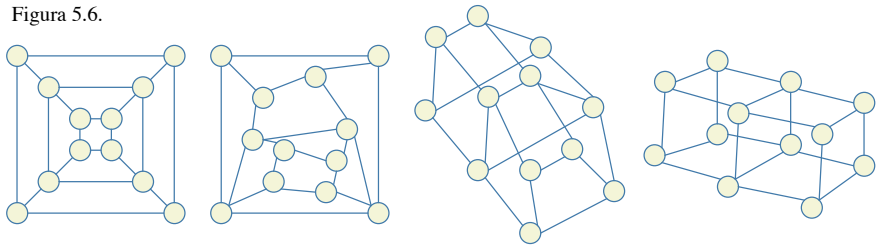
dove:

- $\vec{F}_1 = \sum_{u,v \in V} \frac{1}{\overrightarrow{d(u,v)}^2}$ è simile alla forza di repulsione elettrica, ed impedisce ai nodi di essere troppo vicini;
- $\vec{F}_2 = \sum_{u \in V} \left(\frac{1}{s(u)^2} + \frac{1}{d(u)^2} + \frac{1}{a(u)^2} + \frac{1}{b(u)^2}\right)$, dove *s(u), d(u), a(u), b(u)* sono le distanze del nodo *u* dai lati del disegno. Questo termine assicura che i nodi non vengano rappresentati troppo vicini ai bordi dello schermo;
- $\vec{F}_3 = \sum_{(u,v) \in E} \overrightarrow{d(u,v)}^2$, assicura che gli archi non divengano troppo lunghi;
- F_4 è il numero di incroci del disegno.

Modificando le costanti l_i si predilige l'uno o l'altro criterio estetico, ottenendo – a partire dallo stesso input – disegni molto diversi: se un certo l_i è grande, allora il criterio i-esimo è importante.

Come esempio, si veda la figura 5.6.

Figura 5.6.



a. b. c. d.

La parte a. mostra un grafo planare con 12 nodi ed il suo disegno qui rappresentato soddisfa parecchi criteri estetici, in particolare esso è simmetrico e piano. Il metodo di Davidson ed Harel non è in grado di produrre un tale disegno poiché tipicamente la planarità e la simmetria sono associati ad archi con lunghezze molto diverse. Tuttavia l'algoritmo produce gli altri tre disegni di figura 5.3. (da b. a d.), tutti interessanti da qualche punto di vista: la figura 5.3.b. è piano ma la lunghezza degli archi non è uniforme e il disegno non è simmetrico; questo disegno è stato prodotto con un alto valore di l_4 . Man mano che tale valore viene fatto decrescere, otteniamo la Figura 5.3.c. e poi la 5.3.d. Questi disegni hanno incroci, ma evidenziano delle simmetrie ed hanno una gradevole apparenza tridimensionale. Si osservi, infine, che la Figura 5.3.d. presenta tutti gli archi pressocché della stessa lunghezza.

Il problema di questa generalizzazione del metodo basato sulle forze è che è computazionalmente dispendioso; infatti, una volta trovato un sistema di equazioni da minimizzare (e non più da risolvere!!), dobbiamo usare delle tecniche come il simulated annealing.

Il metodo del simulated annealing è un metodo di ottimizzazione molto flessibile, originato dalla meccanica statistica negli anni '80. Esso differisce dalla tecnica *greedy* perché permette mosse "*uphill*" con energia temporaneamente maggiore per poter sfuggire dai minimi locali.

Il suo maggior punto di debolezza è che è molto lento: Davidson e Harel notano che il loro alogoritmo, basato su questa tecnica, funziona bene per grafi con al più 30 nodi e 50 archi, poi – se si vuole mantenere un tempo di computazione accetabile – la qualità della soluzione si deteriora, tanto che, per usare questo algoritmo con successo si deve partire da un disegno già vicino alla soluzione e darlo in input all'algoritmo di Davidson e Harel per migliorarlo. D'altra parte, il principale punto di forza del metodo del simulated annealing consiste nella sua capacità di gestire l'ottimizzazione in uno spazio discreto troppo grande per la ricerca esaustiva.

Il metodo del *simulated annealing* è basato sul processo fisico di raffreddare i liquidi molto lentamente (*annealing*) per raggiungere la forma cristallina, cioè quella con minimo energetico. Si noti che un raffreddamento rapido non conduce alla cristallizzazione, e quindi neanche al minimo dell'energia; invece, se il raffreddamento avviene lentamente, ad ogni temperatura si raggiunge un equilibrio termico, e il sistema obbedisce alla distribuzione di Boltzman:

$$p(E) \approx e^{-E/(kT)}$$

dove $p(E)$ è la distribuzione di probabilità dei valori di energia, come funzione della temperatura e della costante di Boltzman k.

Questo processo fisico può essere simulato con una serie di passi sequenziali:

- si parta da una configuarzione arbitraria dei nodi;
- ad ogni fissata temperatura T, si imponga il raggiungimento dell'equilibrio termico, portandosi in uno stato ad energia minima: il sistema si sposta da uno stato con energia E_1 ad uno stato con energia E_2 con probabilità $e^{-(E2-E1)/(kT)}$. Con questa regola, se $E_2 < E_1$ allora il sistema si sposta, altrimenti il passaggio è probabilstico; nel caso dell'algoritmo di Davidson e Harel E è la funzione di energia da essi determinata, ed il numero di

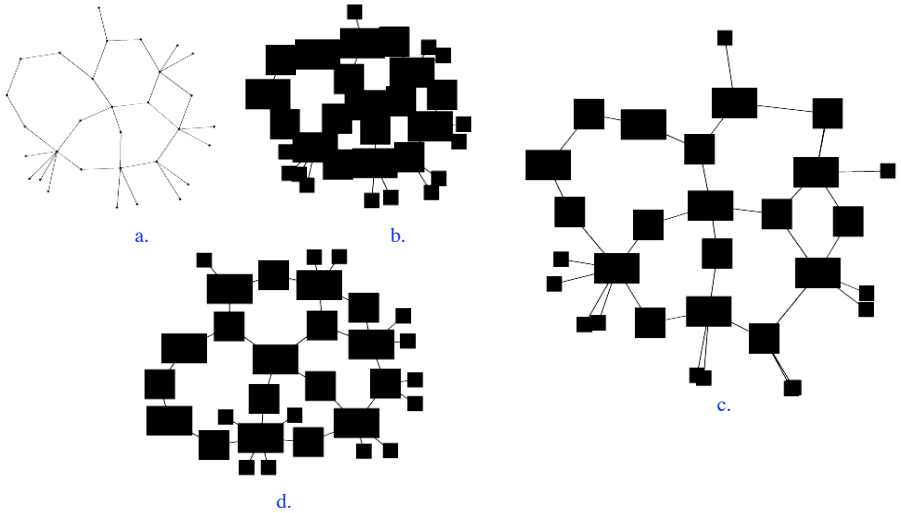
iterazioni richieste per raggiungere il minimo energetico (o una sua buona approssimazione) è circa 30 per ogni temperatura fissata;
- la funzione di raffreddamento è definita dalla seguente formula: $T_{p+1}=\gamma\ T_p$, dove γ appartiene all'intervallo [0.6, 0.95].

La complessità computazionale dell'algoritmo è $O(n^2m)$.

5.2.3. Generalizzazione di Harel e Koren

In questa generalizzazione si affronta il problema di rappresentare non più un grafo con nodi puntiformi, ma un grafo i cui nodi siano dotati di area non nulla. E' facile vedere (Figura 5.7.a. e .b) che non è sufficiente disegnare il grafo con nodi puntiformi e poi sostituirli, ad esempio, con dei rettangoli poiché si genera un numero enorme di sovrapposizioni. Ne' è una buona soluzione quella di espandere il disegno in modo che i nodi rettangolari abbiano più spazio, poiché in ogni caso si spreca molto spazio senza necessariamente risolvere il problema delle sovrapposizioni (vedi Figura 5.7.c). Applicando, invece, un metodo basato sulle forze, il risultato è soddisfacente (vedi Figura 5.7.d.). Tale metodo richiede anch'esso di modificare il sistema di forze utilizzando campi ellittici (per un approfondimento, si consulti [HK02]).

Figura 5.7.



Si conclude con un'osservazione sulla lunghezza degli archi: nel caso in cui i nodi siano puntiformi, ottenere un disegno con gli archi abbastanza corti è auspicabile per non avere sprechi di spazio. Nel caso in cui, invece, i nodi, siano dotati di area non nulla, non ha sempre senso minimizzare la lunghezza degli archi, come si evince dalla Figura 5.8, in cui accorciare gli archi fa perdere la simmetria del disegno, e dalla Figura 5.9, in cui si introducono addirittura degli incroci. Per ovviare a questo problema, questa generalizzazione procede in varie fasi: all'inizio consente archi lunghi; poi, tramite una ricerca binaria, li accorcia via via fino a quando ciò sia possibile, cioè fino a quando non vengano introdotte situazioni indesiderate.

Figura 5.8.

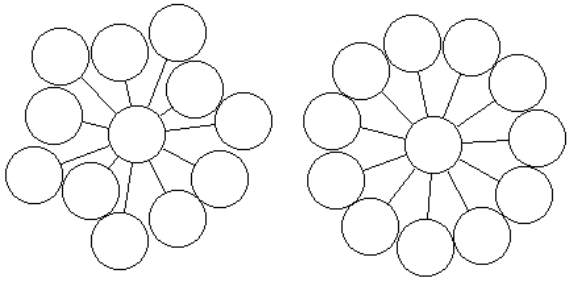
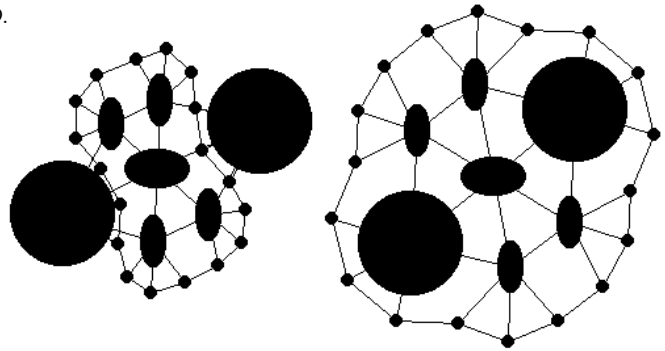


Figura 5.9.



5.3. Vincoli

Un'altra modifica all'idea di base che si può efficacemente fare è quella di introdurre dei vincoli. I più comuni sono quelli di fissare alcuni nodi su dei punti, su una retta, o all'interno di una regione.

E' abbastanza semplice generalizzare i metodi numerici affinché rispettino il vincolo che un nodo rimanga entro certi spazi (basta limitarne i movimenti).

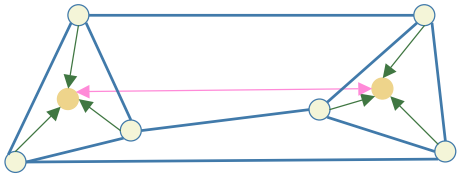
Bloccare dei nodi consiste nell'assegnare un prefissato sottodisegno ad un certo sottografo; il metodo del baricentro, con il suo insieme di punti fissati, si può vedere come un caso particolare di questo vincolo, in cui il sottodisegno è un poligono convesso. In questo caso, il sottodisegno va trattato come un corpo rigido – anziché come un punto – a cui siano applicate le forze relative al metodo usato; ne segue che il sottodisegno può essere ruotato o traslato, ma non dilatato.

Obbligare un certo arco a giacere su una retta si può simulare con un campo magnetico omogeneo nella direzione di quella retta.

Un altro vincolo interessante è quello di clusterizzare un insieme di nodi. Questa operazione si può fare come segue (vedi Fgura 5.10):

- per ogni insieme C di nodi da clusterizzare, si aggiunga un nodo ausiliario v_c che funga da "attrattore";
- si aggiungano delle forze attrattive tra v_c e ciascun nodo di C;
- si aggiungano forze repulsive tra coppie di attrattori e tra attrattori e nodi di cluster diversi.

Figura 5.10.



RIFERIMENTI BIBLIOGRAFICI

[C04] C. Chen: *Information Visualization beyond the horizon* – 2nd Edition, Springer 2004.

[Dal99] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis*: Graph Drawing – Algorithms for the visualization of graphs*, Prentice Hall, 1999.

[EG96] P. Eades, P. Garvan. Drawing Stressed Planar Graphs in Three Dimensions. *Proc. GD '95*, Lecture Notes in Computer Science 1027, 1996.

[HK02] D. Harel, Y. Koren. Drawing graphs with non uniform vertices. *Proc. Working Conference on Advanced Visual Interfaces (AVI'02),* 157--166. ACM Press, 2002.

[KW98] M. Kaufmann, D. Wagner (Eds.): *Drawing Graphs – Methods and Models*. Lecture Notes in Computer Science 2025, Springer 1998.

[LRT79] R.J. Lipton, D.J. Rose, R.E. Tarjan. Generalized Nested dissection. *SIAM J. Numer. Anal.* 16(2), 346-358, 1979.

[SM95] K. Sugiyama, K. Misue. A Simple and Unified Method for Drawing Graphs: Magnetic-Spring Algorithm. *Proc. GD '94* Lecture Notes in Computer Science 894, 364-375, 1995.

[T60] W.T. Tutte. Convex Representations of Graphs. Proc. London Math. Society 10(3), 304-320, 1960.

[T63] W.T. Tutte. How to Draw a Graph. Proc. London Math. Society 13(3), 743-768, 1963.

6. VISUALIZZAZIONE DI SUPERFICI TRIDIMENSIONALI

6.1. Il problema della triangolazione

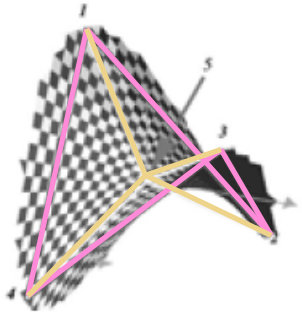
Riferimenti: [MBR01] pp 87-92.

E' ben noto che se vogliamo rappresentare una curva o una retta in uno spazio bidimensionale, a causa del fatto che ogni supporto grafico è discretizzato, dobbiamo avvalerci di una sua approssimazione tramite segmenti orizzontali o verticali.

Allo stesso modo, quando ci troviamo nello spazio a 3 dimensioni, per rappresentare una superficie, dobbiamo necessariamente usare una sua approssimazione discretizzata, cioè una forma poliedrica. La forma di rappresentazione più semplice che si adotta per descrivere modelli geometrici solidi è quella dei poliedri con facce triangolari. Il motivo di questa scelta risiede in una serie di ragioni tecniche e di proprietà geometriche, di cui ne elenchiamo alcune.

1. Dal punto di vista geometrico, questa scelta presenta un vantaggio fondamentale: poiché per 3 punti passa uno ed un solo piano, una faccia triangolare è univocamente definita dai suoi tre vertici.
2. La faccia triangolare evita che si debbano fare verifiche sulla complanarità dei suoi vertici, infatti – per la stessa ragione precedente – 3 punti sono sempre complanari, ma se abbiamo 4 o più punti che vogliamo definiscano una faccia, non è detto che essi giacciano sullo stesso piano e definiscano, quindi, una faccia ammissibile.
3. Infine, se la superficie non è estremamente regolare (vedi Figura 6.1.), usando facce quadrangolari non si riesce sempre ad approssimarla bene, mentre migliore è l'approssimazione ottenuta tramite facce triangolari.

Figura 6.1.



Dalle precedenti osservazioni deduciamo che, data una superficie, al fine di rappresentarla, è essenziale determinarne una sua triangolazione. Innanzi tutto, sorge il problema che, scomponendo una superficie in triangoli, bisogna decidere quale sia il numero giusto di triangoli: è chiaro che aumentando il numero di triangoli si ottiene una maggiore *accuratezza* (si prenda ad esempio la Figura 6.1.: aggiungendo punti da triangolare, la superficie risulta molto meglio approssimata), mentre diminuendolo si ottiene una maggiore *efficienza*, sia nella memorizzazione che nella gestione. Occorre quindi trovare un equilibrio tra accuratezza ed efficienza.

Il procedimento per scomporre in triangoli una superficie si chiama *tassellazione* o *triangolazione*, ed è di questo che ci occuperemo nel seguito.

6.1.1. Come calcolare i punti della triangolazione

Abbiamo detto che tre punti determinano univocamente un triangolo, quindi il primo passo necessario per ottenere una triangolazione consiste nel calcolare i vertici che definiscono tale triangolazione.

Se la superficie è descritta da una funzione $z=f(x,y)$, per costruire una triangolazione si possono individuare tutti i punti sulla superficie che abbiano le loro coordinate x e y tra loro a distanza costante in tutte le direzioni. Per ottenere questo risultato, è sufficiente decidere la distanza d e poi individuare tutti i punti $P_{i,j}=(x_i, y_j, z_{i,j})$ tali che $y_i=id$ e $y_j=jd$.

Questo metodo può andare bene se la superficie ha un andamento regolare, ovvero se la curvatura è abbastanza uniforme. Ma se la superficie ha regioni a curvatura elevata e regioni a curvatura bassa, allora non ha molto senso suddividere tutta la superficie allo stesso modo: sarebbe più proficuo infittire i triangoli nelle regioni a curvatura alta e diradarli nelle altre.

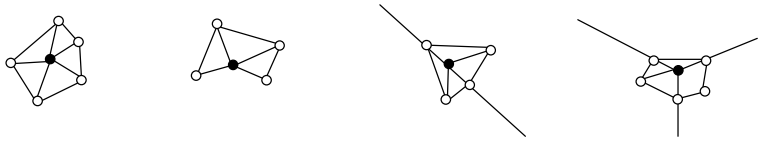
Per fare ciò, determiniamo il grado di curvatura valutando le derivate prima e seconda della superficie in prossimità di un determinato punto (questo per tutti i punti). Vista la difficoltà numerica di calcolare le derivate (limite del rapporto incrementale), ci limitiamo ad approssimarle, calcolando il rapporto incrementale nelle due direzioni x ed y. Se il rapporto incrementale è cresciuto o diminuito più di una certa soglia nel passare da un punto ad un altro, allora deduciamo che la curvatura è abbastanza alta e sarebbe bene introdurre un nuovo punto sulla superficie per costruire un triangolo più piccolo; altrimenti la triangolazione può essere lasciata invariata.

Laddove la curvatura invece è praticamente inesistente, potremmo eliminare alcuni punti, senza perdere informazioni. Questa eliminazione prende il nome di *decimazione*. Un metodo molto usato (detto *metodo di Schroeder*) classifica gli spigoli del poliedro da decimare per individuare quelli rilevanti per l'aspetto, quelli cioè la cui rimozione cambierebbe totalmente la forma del poliedro (ad esempio il bordo di un parallelepipedo). Per decidere se uno spigolo è rilevante per l'aspetto si può valutare se l'angolo che le due facce ad esso adiacenti formano superi una certa soglia fissata. Se non la supera, e quindi lo spigolo potrebbe teoricamente essere eliminato, occorre classificare i punti che lo generano e decidere la loro eventuale eliminazione.

Le classi possibili di punti sono 4:

1. punti interni generici, detti *punti semplici* (vedi Figura 6.2.a);
2. punti che si trovano sul bordo della superficie, detti *punti contorno* (vedi Figura 6.2.b);
3. punti che si trovano su uno spigolo rilevante per l'aspetto, detti *punti rilevanti* (vedi Figura 6.2.c.);
4. punti comuni a 3 o più spigoli rilevanti per l'aspetto, detti *punti vertice* (vedi Figura 6.2.d).

Figura 6.2.



a. b. c. d.

Consideriamo le 4 classi una ad una e studiamo in quale caso il punto si possa eliminare.

1. I punti semplici si possono eliminare se non danno un contributo troppo significativo: lo eliminiamo se la sua distanza dalla superficie approssimante i vertici di tutti i triangoli che ammettono quel punto come vertice è inferiore ad una certa soglia. Infatti, in tal caso, la rimozione del nodo comporterebbe una limitata deformazione della curvatura della superficie in prossimità del punto (vedi Figura 6.3.a).
2. Un punto contorno si può eliminare se la sua distanza dalla retta congiungente gli altri punti contorno ad esso adiacenti è inferiore ad una certa soglia (vedi Figura 6.3.b).
3. Analogamente si decide dell'eliminazione di un punto rilevante, considerando i suoi adiacenti punti rilevanti.
4. Infine, un punto vertice non può mai essere eliminato.

Figura 6.3.

a. b.

6.1.2. Come calcolare la triangolazione, dato l'insieme di punti; la triangolazione di Delunay e il diagramma di Voronoi

Nel paragrafo precedente, abbiamo discusso su come si generino i vertici di una triangolazione. Inoltre, dato un insieme di punti oppure una triangolazione già definita, abbiamo descritto come aggiungere e rimuovere punti. In ciascuno di questi casi, ci troviamo di fronte alla necessità di costruire una triangolazione, o dall'inizio, oppure da una triangolazione parziale non più valida.

Descriviamo quindi un metodo per costruire una triangolazione, dato un insieme di punti, visto che sono innumerevoli le triangolazioni che possono scaturire da un insieme di punti qualsiasi.

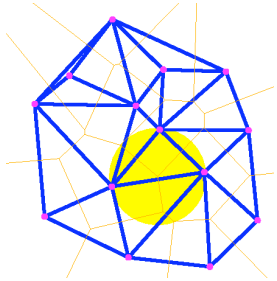
La prima osservazione da fare è che sarebbe opportuno definire dei triangoli il più possibile regolari e tra punti che siano abbastanza vicini tra loro. Tra i tanti metodi, ci concentriamo sulla *triangolazione di Delaunay*, che ha appunto questo requisito. La triangolazione di Delaunay ha la proprietà che il cerchio circoscritto a un qualunque triangolo non contiene alcun punto dell'insieme. Questa triangolazione viene spesso messa in relazione con il *diagramma di Voronoi* costruito sull'insieme dei punti, poiché ne costituisce l'opearzione duale.

Definizione. Il diagramma di Voronoi di un insieme di punti, detti *vertici,* è una partizione del piano tale che ogni *cella di Voronoi* di un vertice P è il luogo dei punti del piano che sono più vicini a P che a qualunque altro vertice. Vedi Figura 6.4. Il punto P si chiama *generatore* della cella di Voronoi. Ciascun bordo di una cella di Voronoi è un segmento che biseca il segmento che congiunge P con il vertice della cella adiacente. Ciascuna intersezione di bordi di una cella di Voronoi (cioè ciascun vertice del poligono che delimita la cella di Voronoi) appartiene almeno a tre celle di Voronoi, ed è il centro del cerchio passante per i generatori di queste tre celle. Questi tre vertici formano un *triangolo di Delaunay*.

Dalla definizione precedente segue che una triangolazione di Delaunay si può determinare calcolando prima il diagramma di Voronoi e deducendo quindi la triangolazione.

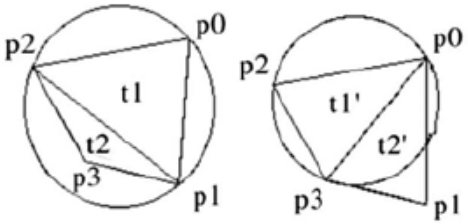
Un altro metodo – dovuto a Sibson, detto *algoritmo di flipping* - calcola invece direttamente la triangolazione di Delaunay, ma ha bisogno come punto di partenza di una triangolazione qualunque, per poi trasformarla in una di Delaunay.

Figura 6.4.



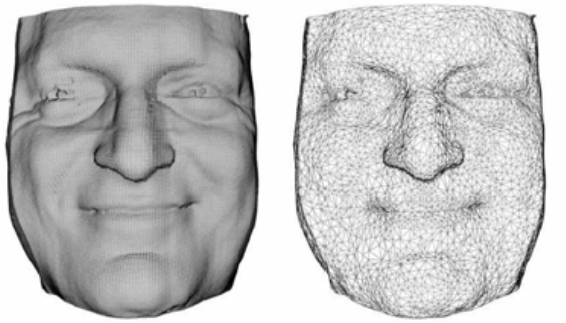
Si considerino tutte le coppie di triangoli adiacenti $t_1=P_0P_1P_2$ e $t_2=P_3P_2P_1$. Se il cerchio circoscritto al triangolo t_1 contiene internamente il vertice P_3, oppure il cerchio circoscritto al triangolo t_2 contiene internamente il punto P_0, allora questi due triangoli non costituiscono una triangolazione di Delaunay valida e devono essere ridefiniti come $t'_1=P_0P_2P_3$ e $t'_2=P_0P_1P_3$ (vedi Figura 6.5). E' possibile dimostrare che l'insieme di queste trasformazioni produce una triangolazione di Delaunay valida.

Figura 6.5.



In Figura 6.6. è mostrato un esempio di funzionamento dell'intero processo di triangolazione (determinazione dei punti, decimazione e determinazione della triangolazione di Delaunay).

Figura 6.6.



6.2. Modelli di Illuminazione

Riferimenti: [MBR01] pp 254-259, 262-266; [AB97] § 9.1, 9.2 e 9.4.

6.2.1. Modelli di illuminazione locali: modello di Lambert, luce ambientale, modello di Phong, attenuazione

6.2.2. Ombreggiature: flat shading, ombreggiatura di Gourard, ombreggiatura di Phong

6.2.3. Ombre proiettate e sorgenti estese

RIFERIMENTI BIBLIOGRAFICI

[MBR01] D. Marini, M. Bertolo, A. Rizzi. *Comunicazione Visiva Digitale – Fondamenti di eidomatica*. Addison-Wesley, 2001.

[AB97] G. Attardi, A. Bernasconi. *Fondamenti di Computer Graphics*, 1997. http://medialab.di.unipi.it/web/IUM/Fondamenti/cap9.htm