7. O método do embrulho para presente (gift wrapping)

Como é mecionado em Preparata & Shamos [11], não é conhecido nenhum algoritmo razoavelmente eficiente para construir o fecho convexo de um conjunto de pontos que não produza uma descrição da fronteira (1-esqueleto) do politopo (fecho convexo). (Lembre-se dos algoritmo que vimos para construir o fecho convexo de um conjunto de pontos no plano.)

O algoritmo que veremos é uma generalização do algoritmo de Jarvis [7] que encontrar o fecho convexo de um conjunto de pontos no plano. O método do embrulho para presente (*gift wrapping)* foi desenvolvido po Chand & Kapur [2] para construir o fecho convexo de um dado conjunto S de pontos em dimensões arbitrárias. A complexidade de tempo do algoritmo é $O(nh)$, onde h é o número de arestas de conv(S), i.e. o algoritmo é *output-sensitive.*

[**Observação.** O material desta seção foi extraído do Capítulo 3 de [6], Capítulo 3 de [11], e Capítulo 3 de [10].]

Para facilitar a exposição vamos assumir que os pontos de S estão em posição geral: não existem quatro pontos em S que sejam colineares. Intuitivamente o algoritmo simula o enrolar do conjunto S por um papel (de presente). A idéia, analogamente ao algoritmo de Jarvis, é usar as aresta das faces do fecho convexo (politopo) que já foram encontradas para encontrar outras faces (já que cada aresta é compartilhada por exatamente duas faces). Em cada passo do algoritmo temos que uma parte do fecho convexo já foi contruída. Uma face f na superfície parcial do fecho convexo é selecionada e uma aresta e de f para a qual a outra face adjacente não foi encontrada ainda também é selecionada (uma tal aresta será dita uma aresta *livre*). O plano π contendo f e rotacionada ao redor de e na direção do conjunto de pontos dado até que um ponto p é encontrado (veja Figura 10). Então conv$(e \cup \{p\})$ é uma nova face triangular do fecho convexo e o processo continua.

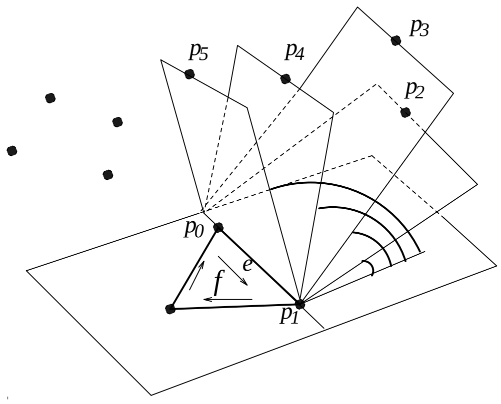

Figura 10. Ilustração de um passo do Algoritmo `Embru-lho-para-Presente`.

Seja $e = p_0p_1$ uma aresta de uma face f do fecho convexo tal que a outra face do politopo P (que representa o fecho convexo) e é incidente a e ainda não foi encontrada pelo algoritmo. Como f é uma face de P, todos os pontos do conjunto S estão em um mesmo semi-espaço definido pelo plano (orientado) π que contém a face f. A face f' adjacente a f em e é tal que todos os semiplanos determinados por e e por pontos em S estão situados em um ângulo diedral maior ou igual ao ângulo diedral formado por e, pelo plano π e pelo plano π' contendo f' (veja Figura 10). Desta forma, dentre todos os semiplanos determinados por e e por pontos em S, a face f' está contida naquele que forma o menor ângulo em relação a e e o plano π. (Esta descrição esta vaga, por favor, olhe a Figura 10 e note que os pontos de S estão todos no semi-espaço positivo determinado pelo plano orientado por $\triangle(p_0, p_1, p_2)$.) Este passo do algoritmo é chamado em [6] de passo do *embrulho-para-presente* e é o passo central do algoritmo.

Como começar o algoritmo já que no início não temos nenhuma face de conv(S)? (Veja [6], pg. 63.) Procederemos de maneira análoga ao Algoritmo de Jarvis. Começamos por determinar um ponto extremo p_0 de S. Seja p_0 o ponto de S com z-coordenada mínima (em caso de empate escolha o ponto com menor y-coordenada e depois, se persistir o empate, escolha o de menor x-coordenada—na realidade, se tivermos empates o nosso trabalho fica, de certa forma, mais fácil). Consideremos agora a reta r que é paralela ao eixo das abscissas e contém o ponto p_0 e consideremos ainda o semi-plano horizontal π contendo p_0 e orientado positivamente na direção do eixo y. Giramos este semi-plano no sentido de y para z até encontrarmos outro ponto extremo p_1 de S. Agora, giramos o novo plano π em torno da aresta $e = p_0p_1$ até encontrar um outro ponto extremo p_2 de S. O triângulo $\triangle(p_0, p_1, p_2)$ será a nossa face inicial. (O que foi descrito aqui nada mais é do que o passo embrulho para presente aplicado duas vezes.)

Abaixo encontra-se uma descrição do Algoritmo `Embrulho-para-Presente`. O algoritmo começa com uma face inicial (gerada como foi descrito acima), e para cada aresta adjacente a esta face o algoritmo encontra a face adjacente. No próximo passo o algoritmo passa a examinar uma outra face e termina quando todas as faces foram geradas. Quando uma nova face f do fecho convexo é gerada pelo passo embrulho-para-presente, suas arestas são examinadas para verificar se elas pertencem a uma face já criada. Neste caso, as duas faces incidentes a esta aresta já estão determinadas e a aresta não precisará mais ser examinada pelo passo embrulho-para-presente. Se só uma das faces da aresta foi determinada chamamos a aresta de *livre*. Cada face recém-criada é colocada em uma fila Q. A cada passo do algoritmo, uma face é removida da fila e o passo embrulho-para-presente é aplicado a cada uma de suas arestas livres. O algoritmo pára quando a fila está vazia (em outra palavras, não existem mais arestas livres a serem examinadas).

Algoritmo `Embrulho-Para-Presente`(S)
`Entrada:` Um conjunto finito S de pontos em E^3.
`Saída:` O fecho convexo dos pontos em S.
1. $Q \leftarrow \emptyset$; [Fila com as faces encontradas.]
2. $f \leftarrow$ uma face incial de conv(S);
3. Insira f na fila Q e na winged-edge estrutura T;
4. **while** $Q \neq \emptyset$ **do**
5. $\quad f \leftarrow$ primeira face na fila Q;

6. **para** cada aresta livre e de f **do**
   [Passo embrulho-para-presente.]
7. $f' \leftarrow$ face do fecho convexo que compartilha e com f;
8. Insira f' na fila Q;
9. Insira f' na estrutura winged-edge T, conectando-a com as faces já geradas que lhe são adjacentes e determinando suas arestas livres.
10. **return** T.

Os passos dominantes do algoritmo são os Passos 5 e 9. O Passo 5 (embrulho-para-presente) pode ser realizado em tempo $O(n)$. Logo, se o número de arestas do fecho convexo de S é h, temos que este passo gasta tempo $O(hn)$ durante toda a execução do algoritmo. No Passo 9, cada vez que uma face é encontrada, é necessário percorrermos a estrutura de dados para verificar se as arestas desta face já foram criadas. Para cada aresta o tempo gasto com esta verificação é $O(n)$. Como cada aresta é gerada exatamente duas vezes temos que o tempo total gasto por este passo durante a execução do algoritmo é (hn). A primitiva usada pelo algoritmo é `Teste-de-Orientação` (ou `Volume6`).

Teorema 3. *O Algoritmo `Embrulho-para-Presente` constrói o fecho convexo de um conjunto de n pontos em E^3 em tempo $O(hn)$, onde h é o número de arestas do fecho convexo.*

Note que se a descrição gerada pelo algoritmo para o fecho convexo contém faces coplanares isto não é inconveniente algum. A partir desta representação podemos obter uma nova representação sem faces coplanares em tempo linear.