Chapitre 3

Calculs de polygones de visibilité

Dans ce chapitre nous montrons comment utiliser le complexe de visibilité pour calculer le polygone de visibilité d'un point situé dans une scène composée de polygones, c'est-à-dire calculer la vue autour du point considéré comme point de vue. Nous étudions deux algorithmes et comparons leurs performances expérimentales.

Nous montrons ensuite comment calculer le polygone de visibilité d'un segment, c'est-à-dire comment calculer les régions éclairées de la scène quand le segment est considéré comme une source de lumière.

— *Attendez que je trouve à tâtons le petit bureau du fond, et je vous en mettrai plein la vue.*
— *Aïe ! L'andouille a mis son doigt dans mon œil !*
Le cas Lagaffe.

3.1 Calculs de vues autour d'un point

Soit un point p_v, dit *point de vue*, dans une scène polygonale. Le polygone de visibilité de p_v, appelé *vue autour de* p_v, contient l'ensemble des points de la scène visibles depuis p_v. C'est un polygone étoilé (figure 3.1a) autour de p_v dont les côtés sont successivement *radiaux* et *transversaux* (figure 3.1b). Les côtés radiaux sont portés par les rayons issus du point de vue p_v et passant par les sommets de polygone visibles depuis p_v ; ils sont délimités par le sommet de polygone visibles et par le point d'intersection du rayon avec l'objet vu le long de ce rayon (qui peut être dans certains cas confondu avec le sommet de polygone). Les côtés transversaux relient les extrémités des côtés radiaux et sont portés par les côtés de polygones visibles depuis p_v.

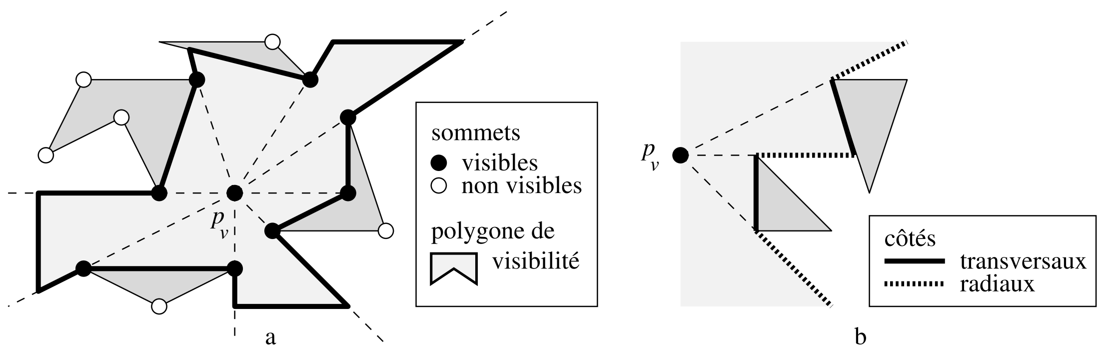


FIG. 3.1 – *Vue autour d'un point.* **a.** *Polygone de visibilité.* **b.** *Nature des côtés du polygone de visibilité.*

La vue autour de p_v calcule donc les objets vus depuis p_v, mais dans un certain ordre : les côtés radiaux apparaissent dans la frontière du polygone de visibilité selon leur ordre angulaire autour de p_v.

L'algorithme naïf pour calculer cette vue consiste à examiner tous les sommets de polygone de la scène, et pour chacun de ces sommets examiner tous les côtés de polygone pour vérifier s'il y en a un ou non qui cache le sommet de p_v. Cet algorithme s'exécute dans le cas le pire en temps $O(n^2)$ (n sommets à examiner, pour chaque sommet $O(n)$ segments à tester), et dans le meilleur des cas en temps $O(vn)$, où v est la taille de la vue, si la vérification des sommets cachés ne prend par chance qu'un temps $O(1)$.

Il faut enfin effectuer un tri angulaire de ces points. La recherche des objets stoppant les rayons radiaux peut se faire en même temps que le test de visibilité d'un sommet depuis p_v.

3.1.1 Calcul de vue par balayage

Puisque les sommets visibles doivent être ordonnés selon leur angle autour de p_v, autant se servir de cet ordre pour maintenir une cohérence dans les calculs de visibilité. C'est ce que fait l'algorithme classique de calcul de vue suivant, qui calcule une vue en temps $O(n \log n)$ grâce à un balayage rotationnel. Après avoir trié les points de la scène selon leur ordre angulaire autour de p_v, l'algorithme calcule la liste des objets intersectés (dans l'ordre) par la demi-droite issue de p_v et passant par le premier point du tri, et maintient cette liste quand la demi-droite issue de p_v passe successivement par les autres sommets de la scène dans l'ordre du tri angulaire.

La liste ne contient que les côtés susceptibles d'être vus, c'est-à-dire les côtés tels que p^+ soit au-dessus de (p_v, p). Les autres côtés sont cachés du point de vue par le polygone et ne sont pas considérés (figure 3.2). Certains sommets p incidents à de tels côtés peuvent même être déjà écartés lors du tri initial.

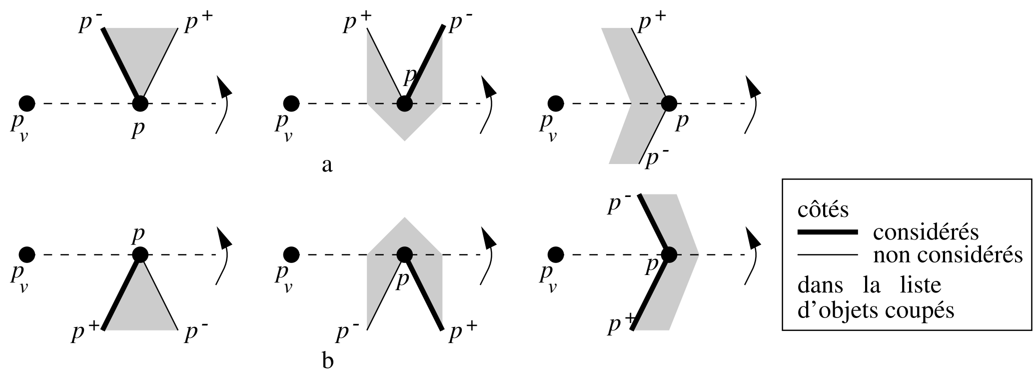


FIG. 3.2 – *Configurations lors du passage d'un sommet lors du calcul de vue.*

Une fois initialisée, la liste est maintenue à chaque fois que la demi-droite issue de p_v passe par l'un des sommets de la scène (dans l'ordre du tri) :

si $[p, p^+]$ n'est pas dans la liste (figure 3.2a)
  si p^- est au-dessus de (p_v, p)
    ajouter $[p^-, p]$ dans la liste
    si $[p^-, p]$ est le premier segment de la liste, p est vu depuis p_v
sinon (figure 3.2b)
  si p^- est au-dessus de (p_v, p)
    remplacer $[p, p^+]$ par $[p^-, p]$
    si $[p^-, p]$ est le premier segment de la liste, p est vu depuis p_v
  sinon
    enlever $[p, p^+]$ de la liste

Le tri initial se fait en temps $O(n \log n)$. La recherche dans la liste ordonnée et son maintien se fait en temps logarithmique à chaque passage de l'un des $O(n)$ points examinés. La complexité finale de l'algorithme est donc en $O(n \log n)$.

L'Utilisation de l'arrangement dual de la scène, construit lors d'une phase de prétraitement, permet de supprimer le tri initial et l'utilisation de la queue de priorité.

L'ensemble des droites passant par le point de vue p_v devient dans l'espace dual une courbe p_v^*. Comme chaque face de l'arrangement correspond à un ensemble de droites traversant la même liste d'objets, pour avoir les listes successives des objets traversés par la demi-droite issue de p_v quand celle ci pivote autour de p_v, il suffit de calculer les faces successives de l'arrangement traversées par p_v^*.

Ces faces sont trouvées en reprenant la méthode utilisée dans la construction incrémentale de l'arrangement (exposée au chapitre 1) : parcours des arêtes bordant une face pour trouver l'arête de sortie et la face suivante. La localisation de la première face (qui est l'une des faces extrêmes gauche de l'arrangement) coupée par la courbe duale se fait en temps $O(\log n)$. Trouver le premier objet vu se fait aussi en temps $O(\log n)$ en faisant une recherche dans la liste des objets associée à cette première face. À chaque nouvelle face rencontrée, la liste des objets coupés et le premier objet vu sont maintenus en temps constant : le ou les segments qui disparaissent ou apparaissent dans la liste sont ceux incidents au sommet de polygone associé à l'arête de sortie de la face ; la liste des objets vus est donnée directement par la nouvelle face traversée ; la mise à jour de l'objet vu se fait alors en temps constant en reprenant l'examen de cas de l'algorithme précédent.

Cet algorithme utilise une droite passant par p_v et non plus une demi-droite issue de p_v comme dans l'algorithme précédent, et des changements dans la liste de segments intersectés peuvent alors concerner des objets situés derrière p_v. Le balayage s'effectue alors seulement de $-\pi/2$ à $\pi/2$ et un changement derrière p_v dans la direction θ correspond à un changement dans la direction $\theta + \pi$.

Grâce à des structures persistantes, les listes d'objets associées aux faces de l'arrangement n'occupent un espace mémoire qu'en $O(n^2)$, et d'après le théorème de la zone, le calcul de la vue s'effectue en temps $O(n)$. Cet algorithme a été notamment décrit par Pocchiola [Poc90] dans le cadre de scènes composées d'objets courbes convexes où la complexité du calcul de vue devient $O(2^{\alpha(n)}n)$, où $\alpha(n)$ est la fonction inverse d'Ackerman (fonction constante en pratique).

3.1.2 Calculs de vue avec le complexe de visibilité

Comme pour l'algorithme de calcul de graphe de visibilité en temps $O(n^2)$ d'Edelsbrunner et Guibas [EG86], la complexité des deux algorithmes de calcul de vue que nous venons de décrire ne dépend que de n. Le dernier algorithme n'est efficace que dans le cas le pire, c'est-à-dire quand (presque) tous les objets de la scène sont visibles depuis le point de vue. Cependant la taille v de la vue calculée est souvent beaucoup plus petite que le nombre total d'objets ($v \ll n$). Pour accélérer le calcul et obtenir un algorithme sensible à la sortie (c'est-à-dire dépendant aussi de v), il faut aussi pour ce problème tenir compte directement des relations de visibilité pour ne traiter, si possible, que les objets visibles depuis le point de vue.

Nous utilisons dans cette partie le complexe de visibilité, qui « contient » déjà les calculs de visibilité entre objets de la scène, comme outil pour améliorer la complexité du calcul d'une vue.

L'avantage de l'utilisation de l'arrangement dual est d'introduire une cohérence dans le calcul de vue, cohérence qui simplifie le calcul en évitant de refaire à chaque fois des recherches. L'inconvénient de l'arrangement est qu'il fait prendre en compte dans le calcul des changements de visibilité non visibles depuis le point de vue. Ceci est du, comme nous l'avons déjà indiqué, au fait que l'arrangement représente des droites qui traversent les objets.

Nous avons aussi vu que le complexe de visibilité avait été créé pour résoudre ce problème : il ne représente plus des ensembles de droites, mais des ensembles de segments libres maximaux. L'utilisation du complexe permet alors de ne considérer que les points visibles depuis p_v.

Le calcul de la vue autour de p_v revient à calculer les changements de visibilité le long du *rayon*, et non plus de la droite, issu de p_v quand ce rayon pivote autour de p_v. Cela revient dans l'espace dual à trouver les faces successives du complexe de visibilité traversées par la courbe duale p_v^* du point de vue et à noter les arêtes séparant ces faces.

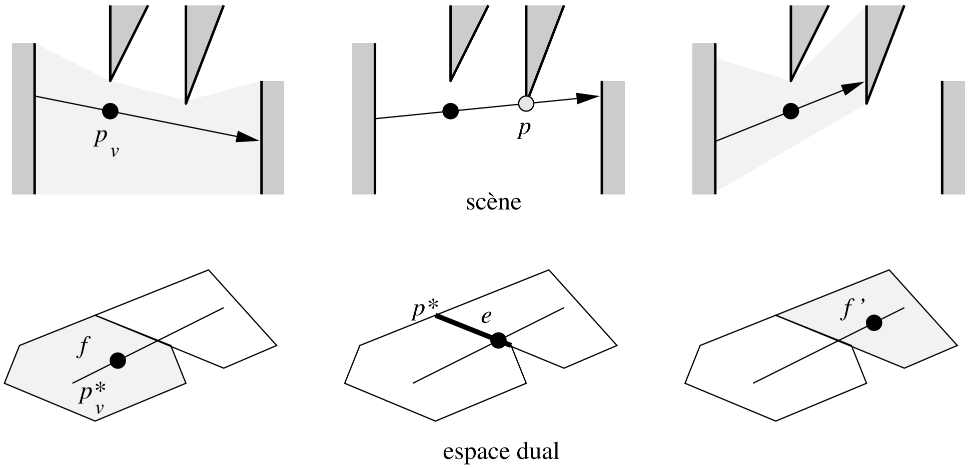


FIG. 3.3 – *Changement de visibilité autour d'un point : passage d'une face du complexe à une autre.*

En effet, lorsque le rayon pivote autour de p_v, un changement de visibilité le long de ce rayon se produit quand celui-ci quitte la face f du complexe où il se trouve pour entrer dans une autre face f'. Le rayon effectue ce franchissement de face en traversant l'arête e qui sépare les deux faces (figure 3.3) et le sommet de polygone correspondant à l'arête traversée est alors visible depuis p_v et fait partie de la vue.

L'algorithme de calcul de vue consiste à maintenir la face du complexe contenant le rayon passant par p_v quand il pivote autour de p_v depuis la direction θ_0 jusqu'à la direction $\theta_0 + \pi$ (car les changements de visibilité à l'arrière du rayon dans la direction θ sont ceux se produisant à l'avant du rayon dans la direction $\theta + \pi$) :

> calculer la face f du complexe contenant le rayon de pente θ_0 passant par p_v
> **tant que** la pente du rayon est inférieure à $\theta_0+\pi$
>   calculer l'arête de sortie de la face f
>   ajouter le sommet correspondant à l'arête dans la vue
>   calculer la nouvelle face f où se trouve le rayon

La première étape de l'algorithme consiste à trouver la face contenant le rayon de direction θ_0. Pour cela, nous considérons la décomposition en trapèzes de la scène selon

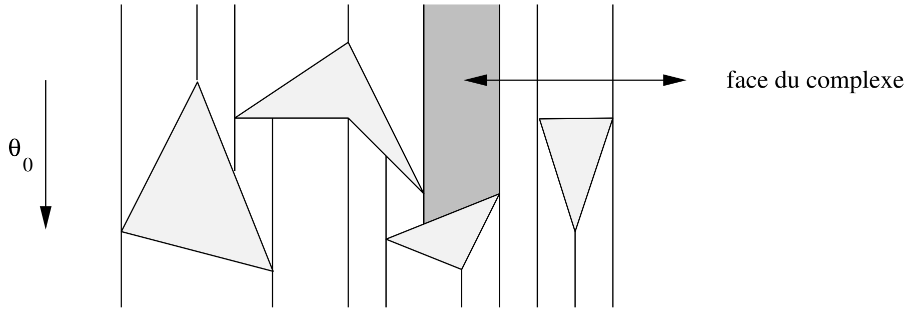


FIG. 3.4 – *Décomposition en trapèzes de la scène.*

la direction θ_0. Cette décomposition est obtenue en étendant à partir de chaque sommet de polygone un segment libre maximal de direction θ_0 (figure 3.4).

Elle correspond à une coupe du complexe de visibilité selon la direction θ_0 et chaque trapèze de cette décomposition correspond à une face du complexe présente dans la coupe. Trouver la face initiale revient donc à trouver le trapèze de la décomposition contenant le point de vue. Comme la décomposition contient $n_{sommets}+n_{polygones}+1 = O(n)$ trapèzes, en utilisant l'algorithme de localisation de point dans une subdivision du plan de Sarnak et Tarjan [ST86] (algorithme basé sur une structure d'arbre rouge–noir persistant) la recherche s'effectue en temps $O(\log n)$. La structure d'arbres persistant se construit en temps $O(n \log n)$ et en espace $O(n)$ pendant le calcul de la coupe initiale du complexe (et donc de la décomposition en trapèzes) lors de la phase d'initialisation de la construction du complexe.

Une fois la face initiale calculée, nous devons chercher à chaque étape l'arête par laquelle le rayon passant par p_v sort de la face où il se trouve ainsi que la nouvelle face où il pénètre.

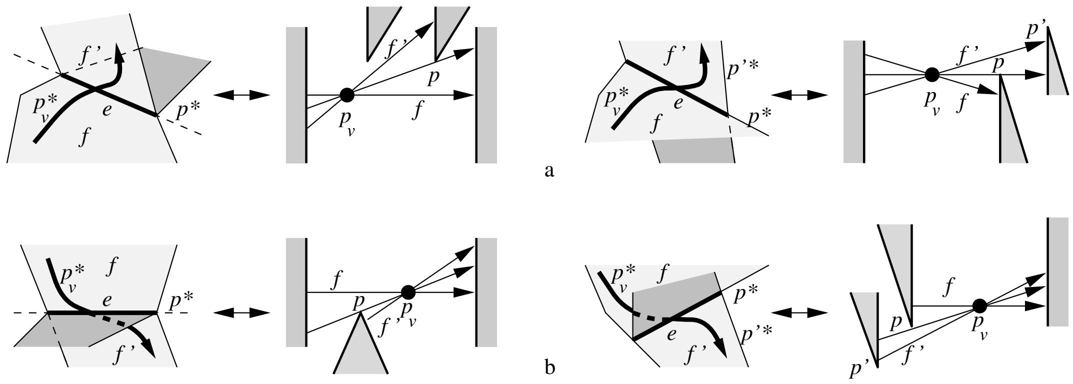


FIG. 3.5 – *Nouvelle face traversée par* p_v^*. **a.** *Sortie par une arête de la chaîne inférieure.* **b.** *Sortie par une arête de la chaîne supérieure.*

Nous avons vu qu'une face était délimitée par deux chaînes d'arêtes. Si le rayon sort de la face f où il se trouve par une arête e de la chaîne supérieure de f, soit e est une arête grasse, soit e est une arête de type μ-convexe ; dans les deux cas, la nouvelle face

f' contenant le rayon est la face incidente gauche-haute de e (figure 3.5a). Si l'arête de sortie e appartient à la chaîne inférieure de f, soit e est une arête grasse, soit e est une arête de type λ-convexe ; dans les deux cas, la nouvelle face f' contenant le rayon est la face incidente droite-basse de e (figure 3.5b).

Pour savoir si l'arête de sortie se trouve dans la chaîne inférieure ou dans la chaîne supérieure, il suffit de tester la position relative du point de vue par rapport à la droite v^* correspondant au sommet v extrême droit de la face : si p_v est au-dessus de cette droite, alors le changement de visibilité se produit à l'avant du rayon et l'arête de sortie se trouve dans la chaîne supérieure (figure 3.6a) ; si p_v est en dessous de v^*, alors le changement de visibilité se produit à l'arrière du rayon et l'arête de sortie se trouve dans la chaîne inférieure (figure 3.6b).

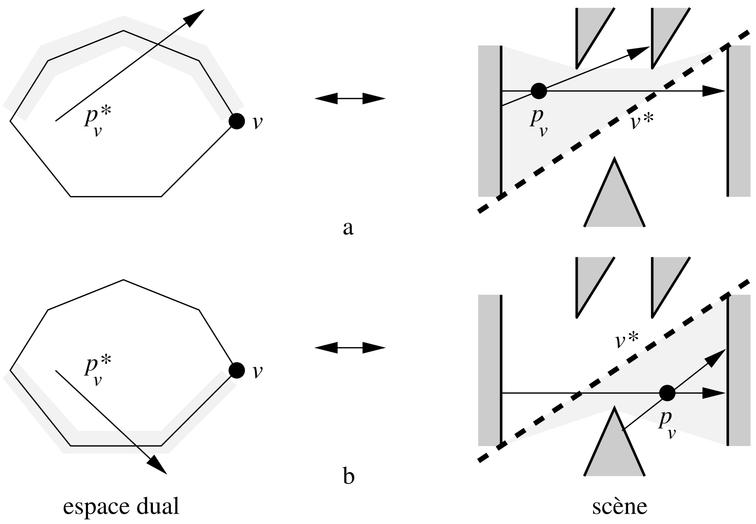


FIG. 3.6 – *Sortie de la face.* **a.** *Sortie par la chaîne inférieure.* **b.** *Sortie par la chaîne supérieure.*

Ainsi nous n'avons qu'une seule chaîne à considérer. De plus, nous remarquons aussi que le rayon passant par p_v ne peut sortir de la face par une arête grasse gauche ; nous écartons donc aussi ces arêtes de la recherche.

Soit une arête $e = (v_g, v_r)$ de la chaîne de sortie. Pour savoir si le rayon sort de la face avant, après, ou par cette arête, il suffit de tester la position de p_v par rapport aux droites v_g^* et v_r^* correspondant aux deux sommets v_g et v_r délimitant l'arête. Par exemple, si l'arête e est dans la chaîne supérieure, alors si p_v est en dessous de v_d^*, la sortie se fait par une arête située après e (figure 3.7a) ; si p_v est au-dessus de v_g^*, la sortie se fait par une arête située avant e (figure 3.7b) ; enfin si p_v est au-dessus de v_d^* et en dessous de v_g^*, la sortie se fait par l'arête e (figure 3.7c).

Pour trouver l'arête de sortie, nous pouvons nous inspirer de l'algorithme de calcul de vue utilisant l'arrangement de droites : après avoir trouvé la chaîne d'arêtes de sortie, nous parcourons les arêtes de cette chaîne de la gauche vers la droite jusqu'à trouver l'arête de sortie. La recherche commence à partir de l'arête d'entrée si celle-ci

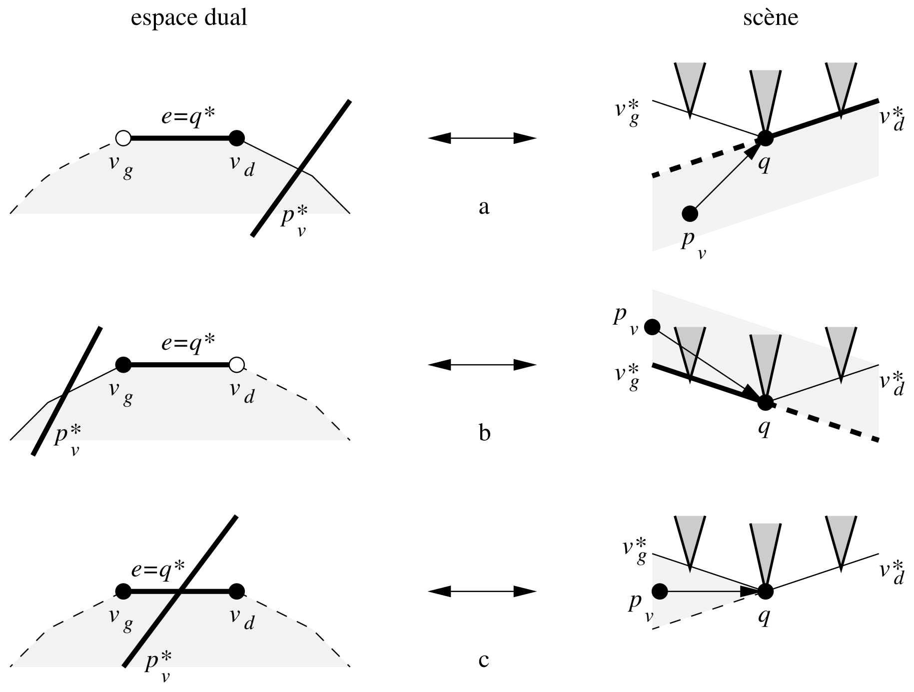


FIG. 3.7 – *Sortie d'une face par la chaîne supérieure : test d'une arête non grasse.* **a.** *Sortie après l'arête testée.* **b.** *Sortie avant l'arête testée.* **c.** *Sortie par l'arête testée.*

se trouve dans la chaîne de sortie, à partir de la première arête de la chaîne de sortie ne faisant pas partie de l'arête grasse gauche sinon.

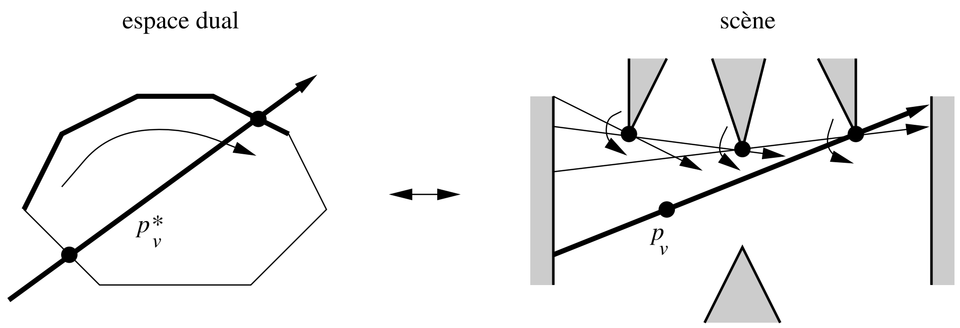


FIG. 3.8 – *Signification géométrique du parcours des arêtes d'une chaîne pour trouver la sortie.*

Dans la scène, cela revient à faire pivoter une droite autour des sommets bordant la face géométrique jusqu'à ce que cette droite passe par le point de vue (figure 3.8).

Grâce à cette méthode, nous obtenons un premier algorithme de calcul de vue :

Proposition 3.1 *Une vue autour d'un point p_v se calcule par la méthode qui vient d'être décrite en temps $\Omega(\log n + v)$ et $O(vn)$, où v est la taille de la vue calculée. De plus, il existe des situations où la borne inférieure (resp. supérieure) est atteinte.*

Démonstration. La propriété 2.13 de visibilité indirecte du chapitre précédent permet de montrer que tous les points correspondant aux arêtes balayées sont visibles du point de vue. Si aucune arête grasse n'est balayée, des techniques similaires à celles employées dans le chapitre précédent pour démontrer la complexité du balayage du graphe de visibilité permettent de montrer que le nombre d'arêtes balayées est en $O(v)$. La borne inférieure v peut même être atteinte dans certains cas.

Les arêtes grasses peuvent augmenter fortement la complexité du balayage. Ces arêtes peuvent être décomposées par les courbes duales de sommets cachés de p_v, que ce soit à leur début ou à leur fin (figure 3.9). De plus, un sommet caché de p_v peut décomposer toutes les arêtes grasses des faces traversées dans le calcul de vue. La complexité maximale en $O(vn)$ est alors obtenue comme dans l'exemple de la figure 3.10. $\square$

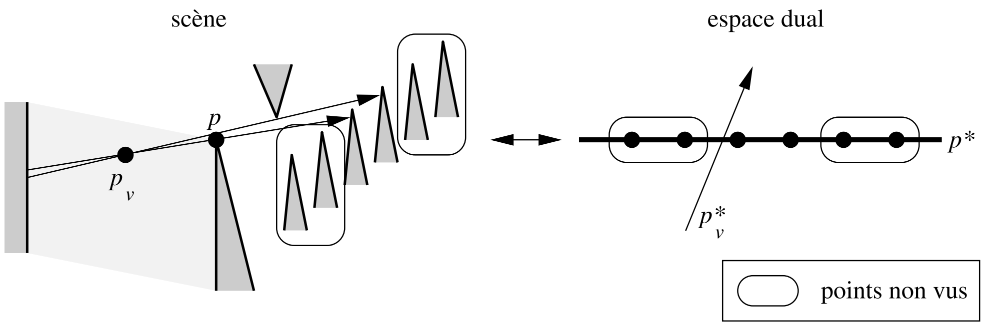


FIG. 3.9 – *Découpage d'une arête grasse par des sommets non vus.*

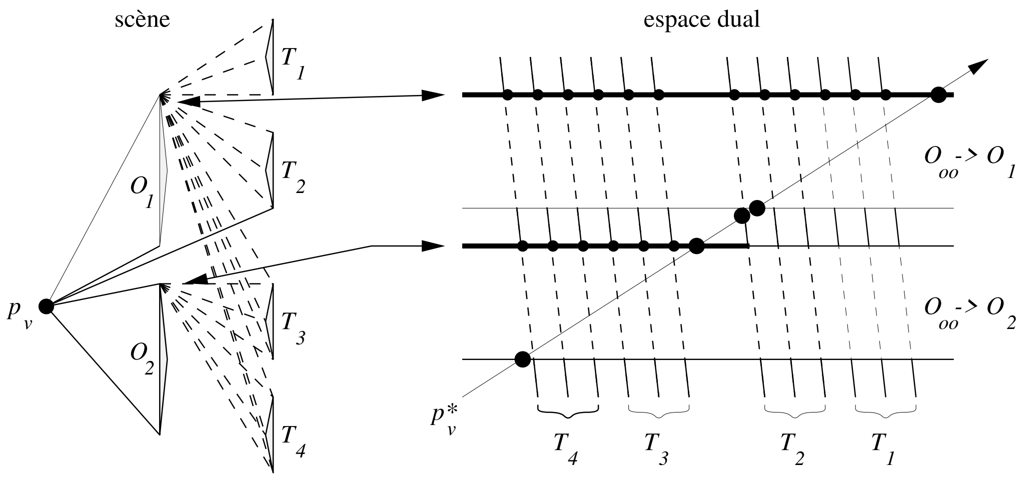


FIG. 3.10 – *Sommets non vus présents dans plusieurs arêtes grasses.*

La complexité de l'algorithme varie d'une complexité optimale à une complexité pire que celle de l'algorithme classique. Des études expérimentales montrent dans la partie suivante comment cette complexité varie en pratique.

Comme les arêtes sont ordonnées dans une chaîne d'arêtes, nous pouvons effectuer une recherche dichotomique dans ces chaînes pour trouver l'arête de sortie :

Proposition 3.2 *Une vue autour d'un point p_v peut se calculer en temps $O(v \log n)$, où v est la taille de la vue calculée, en utilisant le complexe de visibilité.*

Démonstration. Après une initialisation en temps $O(\log n)$, l'algorithme comporte v étapes. À chaque étape, une recherche dichotomique en temps $O(\log n)$ est effectuée pour trouver l'arête de sortie, soit une complexité totale en $O(v \log n)$. □

Nous pouvons construire une structure de données de recherche pour chaque chaîne d'arêtes en temps total $O(m)$. En effet, lors du balayage du complexe, les arêtes des chaînes sont examinées dans l'ordre où elles apparaîssent dans ces chaînes. Nous pouvons, en utilisant par exemple des arbres rouge–noir avec un pointeur sur le plus grand élément, insérer chaque arête en temps amorti $O(1)$ dans les chaînes correspondantes. Le temps de construction du complexe augmenté de structures de recherche reste donc inchangé (en $O(n \log n + m)$).

3.1.3 Étude expérimentale

Nous venons d'étudier deux algorithmes de calcul de vue autour d'un point de complexités théoriques différentes. Le premier, l'algorithme de *parcours*, a une complexité en $\Omega(\log n + v)$ et en $O(vn)$, et le deuxième, l'algorithme de *recherche*, a une complexité en $O(v \log n)$.

Nous étudions maintenant le comportement en pratique de chacun des algorithmes, puis nous comparons les performances expérimentales des deux algorithmes.

Protocole expérimental

Nous avons effectué des calculs de vue sur des scènes aléatoires de triangles disjoints. Chaque triangle $p_1p_2p_3$ est généré de façon aléatoire. Le premier sommet p_1 est tiré aléatoirement dans le disque de centre $(0, 0)$ et de rayon a. Le deuxième sommet p_2 est tiré aléatoirement dans le disque centré en p_1 et de rayon b. Enfin le troisième sommet du triangle est tiré aléatoirement dans le rectangle de côté $[p_1, p_2]$ et de largeur c. Les trois paramètres a, b et c permettent de faire varier les caractéristiques de la scène, notamment la taille du graphe de visibilité.

Nous utilisons la taille du graphe de visibilité pour mesurer une *densité de visibilité* de la scène, que nous notons d_v. Cette densité est calculée par la formule $d_v = (2m - 2n)/n(n - 3)$ et représente la fraction de sommets (autres que les deux sommets adjacents sur le polygone) vus en moyenne par un sommet de la scène. Elle varie théoriquement de 0 (aucun sommet vu) à 1 (tous les $n - 3$ autres sommets vus), cependant les densités maximales obtenues en pratique sont plutôt de l'ordre de 0.6.

Pour une scène donnée, nous avons tiré aléatoirement 500 points de vue dans le disque centré en $(0, 0)$ et de rayon a. Ces points de vue se situent donc en général à l'intérieur de la scène. Dans une deuxième série de tests, nous avons tiré aléatoirement 500 points de vue dans l'anneau compris entre les disques centrés en $(0, 0)$ et de rayons respectifs a et $2a$. De cette façon, les points de vue se situent en général en dehors de la scène.

Complexité mesurée

Comme pour le chapitre précédent, la complexité mesurée n'est pas le temps d'exécution des programmes, mais une complexité plus algorithmique. Pour l'algorithme de parcours, nous avons mesuré le nombre d'arêtes parcourues. Pour l'algorithme de recherche, nous avons utilisé comme structure de recherche des arbres binaires équilibrés AVL. Nous avons alors compté non pas le nombre d'arêtes testées, mais le nombre de comparaisons réellement effectuées. En effet, une arête peut donner lieu à un ou deux tests (sortie avant – sortie après). Ainsi nous mesurons bien pour les deux algorithmes, le nombre de tests effectués.

Dans les diagrammes présentant les résultats, nous avons mis en abscisse la taille des vues calculées, et en ordonnée la complexité correspondante du calcul.

Résultats

La figure 3.11 montre les résultats obtenus avec l'algorithme de parcours pour des scènes de 600 triangles, de densités respectives faible ($d_v = 0.10$), moyenne ($d_v = 0.27$) et élevée ($d_v = 0.50$). Ces résultats sont typiques de ceux obtenus avec des scènes de taille différentes et de même densité. Nous observons que la taille minimale des vues calculées à l'extérieur de la scène est plus grande que celle des vues calculées à l'intérieur. En effet un point à l'intérieur a plus de chances d'être situé très près d'un ou de plusieurs triangles qui lui cachent alors une grande partie de la scène. Nous observons aussi que la complexité du calcul de vue depuis des points situés à l'extérieur est plus élevée que celle de vues calculées à l'intérieur. Pour des points extérieurs, la situation de la figure 3.10 se produit avec la majeure partie de la scène derrière O_1, $O_2 \ldots$, alors que pour des points intérieurs, seule une fraction de la scène est située derrière ces obstacles. La complexité quadratique du découpage des arêtes grasses explique alors la complexité plus élevée pour des points extérieurs.

Nous voyons dans la figure 3.11(a) que la borne inférieure de la complexité est bien linéaire en fonction de v. En revanche, la limite supérieure est moins marquée. Pour des point de vue extérieurs (figure 3.11b), la différence entre les limites inférieure et supérieure de complexité est assez élevée et les résultats sont uniformément répartis entre ces deux limites.

La figure 3.12 montre les résultats obtenus avec l'algorithme de recherche sur les mêmes scènes et avec les mêmes points de vue. Cette fois-ci, la distribution des complexités est beaucoup moins dispersée et la dépendance de la complexité en fonction de la taille de la vue calculée apparaît clairement.

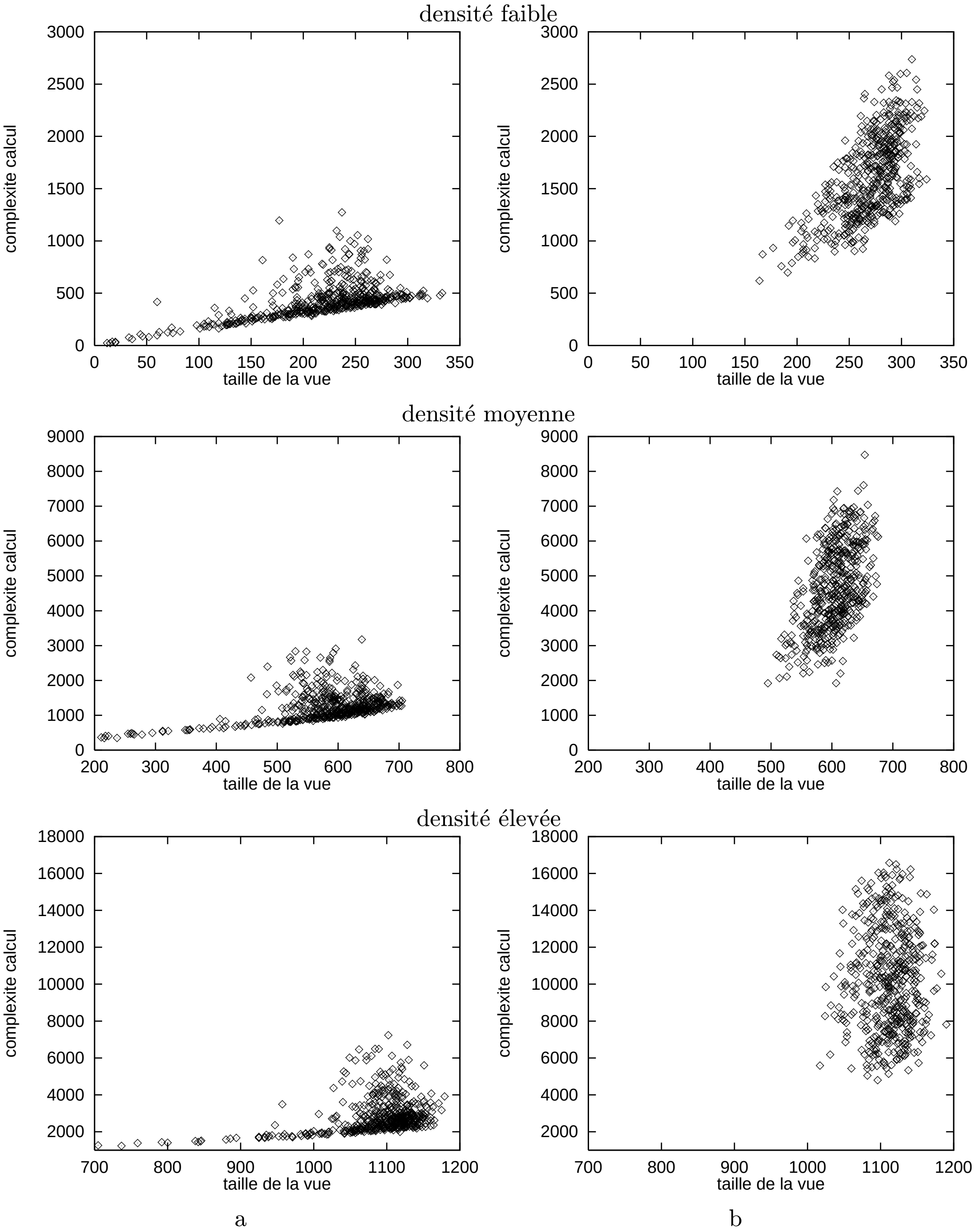


FIG. 3.11 – *Résultats de l'algorithme de parcours sur une scène de 600 triangles.* **a.** *Points de vue dans la scène.* **b.** *Points de vue à l'extérieur de la scène.*

Pour faciliter la comparaison entre les deux algorithmes, nous avons regroupé dans la figure 3.13 les résultats des deux algorithmes pour la scène de 600 triangles densité moyenne sur les mêmes diagrammes. L'algorithme de recherche semble meilleur que l'algorithme de parcours quand les points de vue sont situés à l'extérieur de la scène (figure 3.13b). En revanche, quand les points de vue sont situés à l'intérieur de la scène,

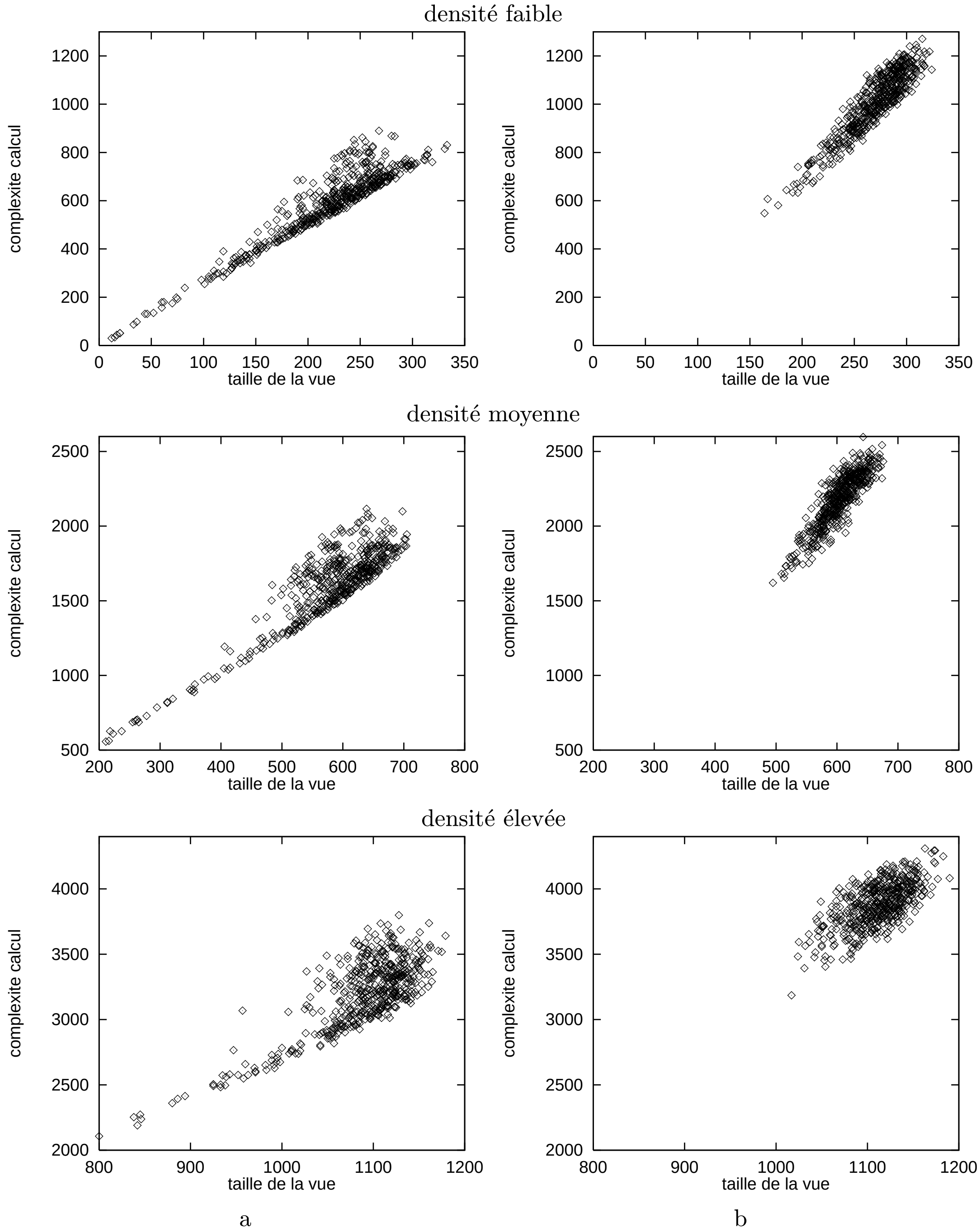
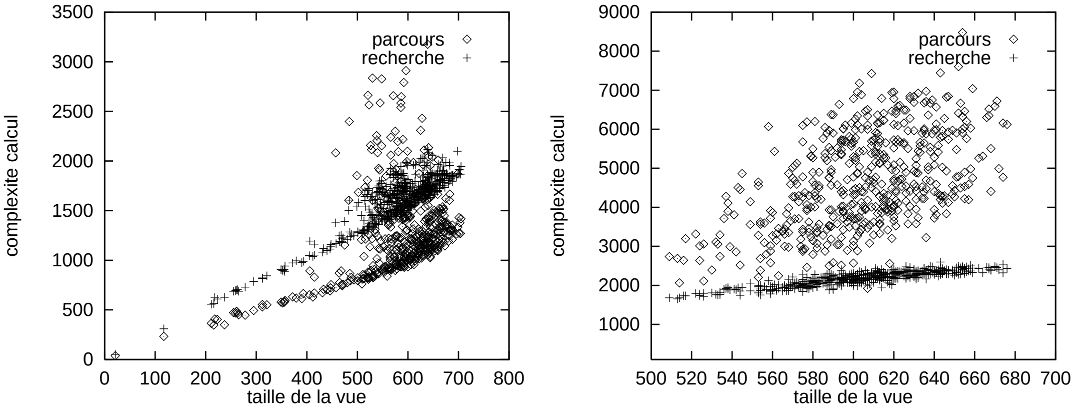


FIG. 3.12 – *Résultats de l'algorithme de recherche* **a.** *Points de vue dans la scène.* **b.** *Points de vue à l'extérieur de la scène.*

l'algorithme de parcours semble meilleur en moyenne (figure 3.13a).

Les mesures de complexité peuvent guider le choix de l'algorithme à utiliser. Cependant, il ne faut pas oublier l'aspect concernant la taille prise en mémoire par les structures de données. L'ajout de structure de données de recherche (arbres binaires par



a b

FIG. 3.13 – *Comparaison des deux algorithmes sur une scène de densité moyenne.* **a.** *Points de vue dans la scène.* **b.** *Points de vue à l'extérieur de la scène.*

exemple) augmente d'environ 20 % la place prise en mémoire par le complexe de visibilité, ce qui peut s'avérer être beaucoup.

Ainsi, le complexe de visibilité de la scène de 600 triangles et de densité élevée, a une taille m proche du million de sommets. Il prend sur une station de travail une place mémoire d'environ 80 Mo, et d'environ 100 Mo avec des arbres de recherche. Cette place mémoire est déjà importante pour une petite scène. Pour des scènes plus complexes, la place mémoire occupée peut devenir une vraie préoccupation. Dans ce cas, un algorithme un peu moins efficace, mais tenant dans la mémoire de la machine peut être préféré si on veut faire tourner le programme sur une machine usuelle (PC par exemple) et non seulement sur des stations de calculs très hautes performances.

3.2 Calcul du polygone de visibilité d'un segment

Nous montrons dans cette section comment calculer le polygone de visibilité d'un segment. Ce polygone délimite les parties éclairées de la scène quand le segment est considéré comme un néon illuminant la scène.

Nous décrivons d'abord un algorithme de calcul du polygone de visibilité. Nous étudions ensuite la complexité de l'algorithme et de la taille du polygone même. Nous montrons enfin que l'algorithme peut aussi servir à calculer toutes les discontinuités d'éclairage dues au néon en temps quasi-optimal.

Nous supposons dans un premier temps que la ligne support du segment est située totalement en dehors de l'enveloppe convexe de la scène. Nous montrons alors comment l'algorithme et sa complexité sont modifiés lorsque le segment est situé dans un endroit quelconque de la scène.

3.2.1 Structure du polygone de visibilité

Soit un segment $s = [p, q]$, dont la droite support est située en dehors de l'enveloppe convexe de la scène. Nous considérons que s est une source lumineuse étendue (un néon par exemple) éclairant le demi-plan contenant la perpendiculaire directe du segment $[p, q]$. Le polygone de visibilité du segment s représente la partie de la scène éclairée par s, et les parties situées en dehors du polygone de visibilité sont les zones d'ombres, qui ne reçoivent aucune lumière de s (figure 3.14a).

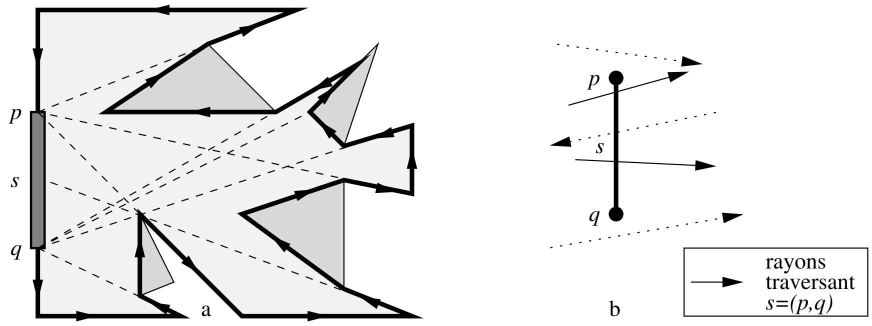


FIG. 3.14 – *Éclairage de la scène par un néon.* **a.** *Polygone de visibilité du segment* $s = (p, q)$. **b.** *Rayons traversant* s.

Un point de la scène est éclairé s'il existe un rayon partant de s et passant par ce point. La zone éclairée de la scène est donc celle couverte par l'ensemble des rayons libres maximaux passant par s. Plus précisément, nous appelons *rayon passant par s* un segment libre maximal de la scène (s n'étant pas considéré comme faisant partie de la scène) traversant s depuis son côté non éclairant vers son côté éclairant (figure 3.14b).

Dans l'espace dual, l'ensemble des rayons éclairant la scène est représenté par l'ensemble des faces (ici externes) du complexe comprises entre les courbes duales p^* et q^* associées aux extrémités de s. Ces faces, dont une partie des rayons associés traversent s, sont dites appartenant à la *zone de s*.

Examinons plus en détails la structure du polygone de visibilité. Comme pour le polygone de visibilité d'un point, les côtés du polygone de visibilité d'un segment peuvent être classés en deux catégories : côtés transversaux et côtés radiaux.

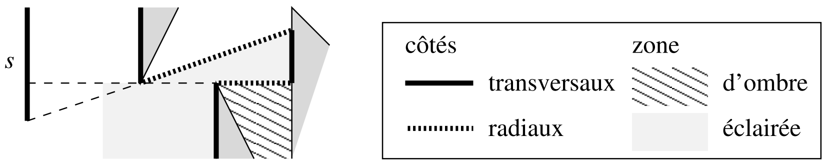


FIG. 3.15 – *Détail de la nature des côtés du polygone de visibilité de s.*

Les côtés transversaux du polygone sont les bords d'obstacle directement éclairés par le segment s. Les côtés radiaux relient les extrémités des côtés transversaux et délimitent les zones d'ombre (figure 3.15). De plus, les côtés successifs du polygone sont alternativement radiaux ou transversaux, les côtés radiaux pouvant parfois avoir leur deux extrémités confondues.

Pour calculer ce polygone, nous parcourons ses côtés transversaux successifs un par un, dans l'ordre où ils apparaissent dans sa frontière. Par exemple, les côtés du polygone de visibilité de la figure 3.14(a) seront parcourus dans l'ordre indiqué par les flèches. Ce parcours revient à visiter les faces du complexe appartenant à la zone du segment s dans un certain ordre.

3.2.2 Calcul du polygone de visibilité

Supposons que nous soyons en train de parcourir un côté transversal du polygone de visibilité porté par l'objet $O = [O_p, O_p^+]$ de la scène, en considérant qu'il est éclairé par les rayons de la face f du complexe de visibilité traversant s.

Si l'extrémité O_p de O n'est pas visible depuis s, alors la limite de la partie éclairée de O est donnée par l'intersection de O et du rayon d de pente maximale parmi les rayons de f traversant s. Ce rayon est « bloqué en haut » par l'extrémité O'_p d'un objet O' et « bloqué en bas » soit par l'extrémité q de s (figure 3.16a), soit par un objet de la scène (figure 3.16b).

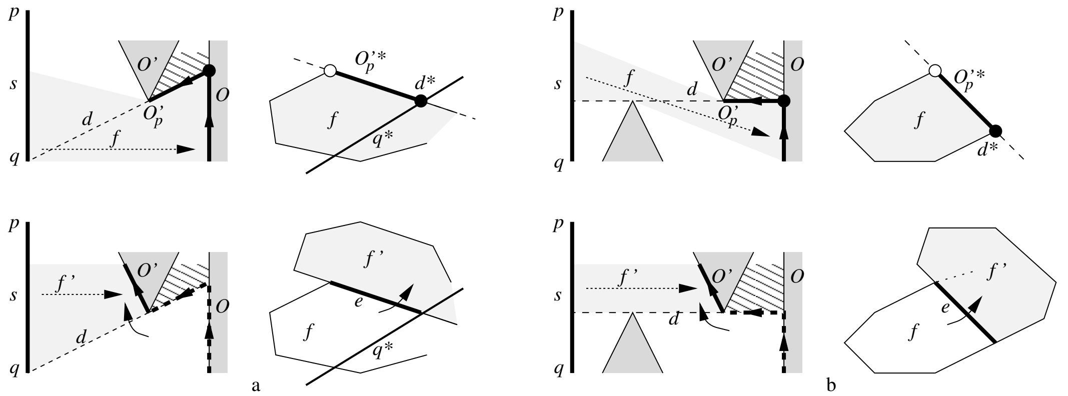


FIG. 3.16 – *Limite d'éclairage sur un côté transversal quand son extrémité n'est pas éclairée.* **a.** *Éclairage limité par q.* **b** *Éclairage limité par un objet de la scène.*

Dans le complexe de visibilité, nous avons deux configurations correspondantes. Soit la courbe duale q^* coupe la face f en une arête e non grasse de la chaîne supérieure de f, auquel cas nous notons d^* le sommet intersection entre l'arête e et q^*. Soit, dans le cas contraire, la dernière arête e de la chaîne supérieure de f n'est pas une arête grasse, et nous notons alors d^* le sommet droit de cette arête. Alors, l'extrémité de la partie éclairée de O est l'intersection de O et du rayon d associé au sommet d^*. Ce rayon passe par O'_p, sommet associé à la courbe duale support de l'arête e. De plus, en notant f' la face incidente gauche-haute de e, le prochain côté transversal du polygone de visibilité est porté par O', objet droit de l'étiquette (O_∞, O') de f', et ce côté est éclairé par les rayons de f' passant par s.

Si l'extrémité O_p de O est visible depuis s, alors l'extrémité de la partie éclairée de

O est O_p. Pour trouver le prochain segment transversal du polygone de visibilité, nous considérons le rayon de pente minimale passant par O_p, traversant s et appartenant à f.

Si le rayon est de type ν-droit en O_p, c'est-à-dire si le rayon est stoppé en O_p par le polygone dont O_p est un sommet, alors le côté suivant du polygone de visibilité est porté par O', côté précédent de O du polygone. Le côté radial est dans ce cas dégénéré : ses deux extrémités sont confondues (et égales à O_p). Le rayon peut être bloqué en bas soit par q (figure 3.17a), soit par un objet de la scène (figure 3.17b).

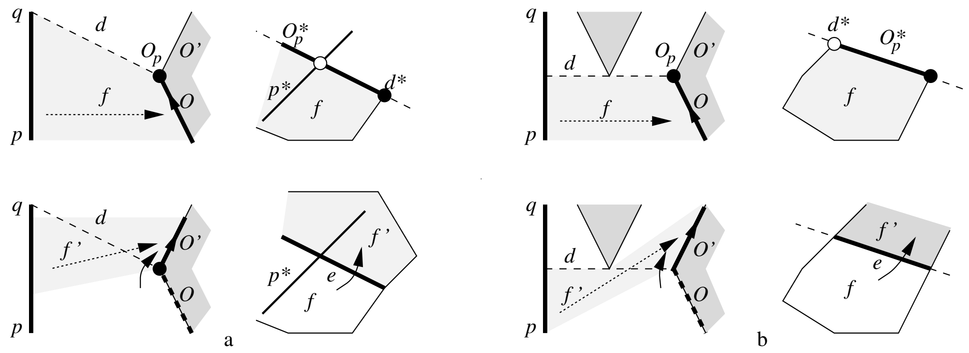


FIG. 3.17 – *Situation où le côté radial du polygone de visibilité est dégénéré.* **a.** *Côté bloqué par q.* **b.** *Côté bloqué par un objet de la scène.*

Dans le complexe de visibilité, la dernière arête e de la chaîne supérieure est de type ν-droit. Le rayon d associé au sommet droit d^* de e est considéré comme support du côté radial dégénéré du polygone de visibilité. En notant f' la face incidente gauche-haute de e, le prochain côté transversal du polygone de visibilité est O', objet droit de l'étiquette (O_∞, O') de f' (et aussi côté précédent de O sur le polygone), éclairé par les rayons de f' passant par s.

Enfin, si le rayon est de type λ-convexe, alors O crée derrière lui une zone d'ombre dont la limite est donnée par le rayon de pente minimale passant par O_p, appartenant à f et traversant s. Ce rayon est bloqué soit par q (figure 3.18a), soit par un objet de la scène (figure 3.18b). Ce rayon est stoppé par un objet O' qui porte le prochain côté du polygone de visibilité et la première extrémité de ce nouveau côté est l'intersection entre O' et le rayon.

Dans le complexe de visibilité, la dernière arête de la chaîne supérieure de f est une arête grasse dont la première sous-arête est de type λ-convexe. Si p^* coupe une sous-arête de cette arête grasse, soit e cette sous-arête et soit d^* le sommet intersection entre e et p^* ; sinon, soit e la première sous-arête de cette arête grasse et soit d^* son sommet gauche. En notant f' la face incidente gauche-haute de e, le prochain côté transversal du polygone de visibilité est porté par O', objet droit de l'étiquette de f', et son extrémité droite est le point d'intersection entre O' et d, le rayon associé au sommet d^*.

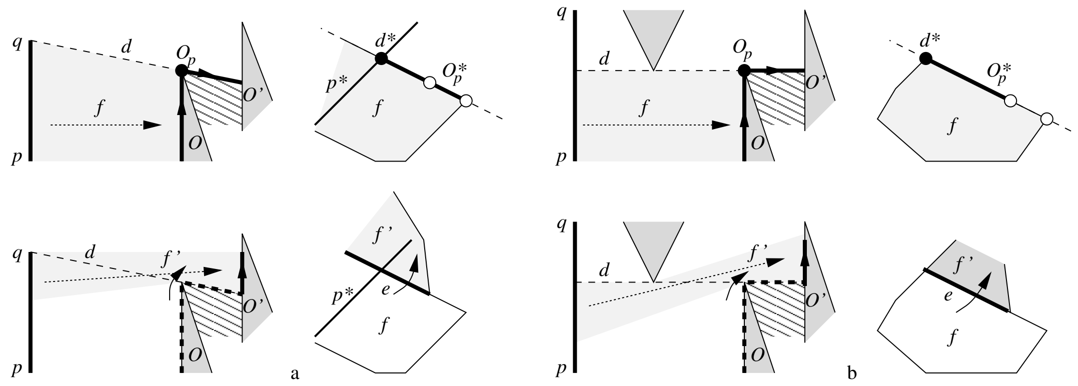


FIG. 3.18 – *Zone d'ombre délimitée par un objet dont son extrémité est éclairée.* **a.** *Éclairage limité par q.* **b.** *Éclairage limité par un objet de la scène.*

La méthode que nous venons de décrire permet de passer d'un côté transversal à un autre. Nous montrons dans la partie suivante la correction de l'algorithme et étudions sa complexité. Nous résumons maintenant l'algorithme de façon complète, en indiquant comment synchroniser le calcul des vues de p et q avec le parcours des faces du complexe. La première face f_d (resp. dernière face f_a) examinée par l'algorithme est la face du complexe de visibilité contenant le rayon (p, q) (resp. (q, p)). Comme nous avons supposé que la droite support de s était en dehors de l'enveloppe convexe de la scène, f_d (resp. f_a) est la face semi-infinie inférieure (resp. supérieure) du complexe de visibilité. L'algorithme complet s'écrit alors :

$f = f_d$
e_p=première arête du complexe coupée par p^*
e_q=première arête du complexe coupée par q^*
tant que $f \neq f_a$
  si e_p est incidente à f **et** est une arête grasse
    $e = e_p$
    si e est de type ν-droit
      $v =$ sommet droit de e
    sinon
      $v = e \cap p^*$
  sinon
  si e_q est incidente à f **et** n'est pas une arête grasse
    $e = e_q$
    $v = e \cap q^*$
  sinon
  si la dernière arête de la chaîne supérieure de f est une arête grasse
    $e =$ première sous-arête de l'arête grasse
    si e est de type ν-droit
      $v =$ sommet droit de e
    sinon
      $v =$ sommet gauche de e

sinon
  e = dernière arête de la chaîne supérieure de f
  v = sommet droit de e
le côté radial est porté par la droite associée à v
si e_p est incidente à f
  e_p = prochaine arête coupée par p^*
si e_q est incidente à f
  e_q = prochaine arête coupée par q^*
f =face incidente gauche-haute de e

3.2.3 Correction et complexité de l'algorithme

L'algorithme que nous venons de définir parcourt les faces du complexe appartenant à la zone de s selon un certain ordre : à une face f de la zone de s, nous associons une unique face f', que nous appelons *face suivante de f selon s*. Un examen des relations d'incidence entre f' et f montre aussi que pour toute face f', il existe une unique face f telle que f' soit la face suivante de f selon s ; f est alors dite *face précédente de f' selon s*.

Toute face du complexe comprise dans la zone de s et différente de f_a (resp. f_d) a une unique face suivante (resp. précédente) selon s. Cela montre que si nous partons de f_d et visitons à chaque fois la face suivante selon s, alors nous arrivons à f_a en ayant traversé chaque face du complexe comprise dans la zone de s une fois et une seule. L'algorithme est donc correct.

Considérons l'ensemble des arêtes e traversées par l'algorithme lors du parcours des face de la zone de s, c'est-à-dire les arêtes séparant une face de la zone de s de sa face suivante selon s. Pour chacune de ces arêtes, nous avons retenu un sommet d^* qui est soit une extrémité propre de l'arête, soit une intersection entre l'arête et une des deux courbes duales p^* ou q^*. Ces sommets permettent de définir dans la scène les côtés radiaux du polygone de visibilité de s. Chacun de ces sommets n'est visité qu'une fois par l'algorithme.

Dans la scène, les rayons correspondant à ces sommets font soit partie de la vue de p (resp. q), soit partie de l'ensemble des arêtes du graphe de visibilité qui (une fois étendues) traversent s. En notant m_s (resp. m_p, resp. m_q) le nombre d'arêtes du graphe de visibilité traversant s (resp. la taille de la vue autour de p, resp. q), alors la taille $Visi(s)$ du polygone de visibilité vérifie $Visi(s) = O(m_s + m_p + m_q)$.

Cependant, la taille du polygone peut être plus petite. La figure 3.19 montre des situations où des arêtes de la vue de p (resp. des arêtes du graphe de visibilité coupant s) dont les sommets correspondant dans le complexe ne sont incidents à aucune arête traversée par l'algorithme.

Précisons maintenant la complexité de l'algorithme. En associant pour chaque face un pointeur sur la première sous-arête de l'arête grasse droite de sa chaîne supérieure, le passage d'une face à la face suivante se fait en temps constant si l'arête de passage e n'est coupée ni par p^* ni par q^*.

Pour rester dans la zone de s, nous devons rester entre les courbes duales p^* et q^*

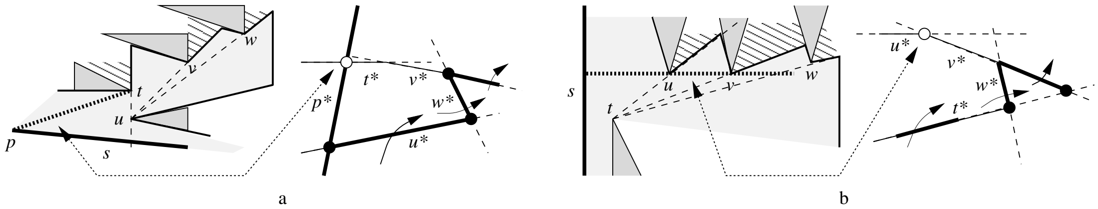


FIG. 3.19 – *Arêtes non visitées dans le calcul du polygone de visibilité.* **a.** *Arête de la vue de p.* **b.** *Arête du graphe de visibilité.*

qui peuvent restreindre les faces visitées. Pour cela, nous calculons au fur et à mesure la vue autour de ces deux points, c'est-à-dire les arêtes du complexes coupées par les courbes duales. Le temps total de calcul de ces faces successives est donc le temps de calcul des deux vues autour de ces points.

Comme la droite support de s est située en dehors de l'enveloppe convexe de la scène, la face f_d de départ (resp. f_a d'arrivée) du parcours de faces est en fait la face semi-infinie délimitant le bas (resp. le haut) du complexe de visibilité, et est accessible directement.

Selon la méthode utilisée pour calculer la vue autour de p et q, le calcul du polygone de visibilité du segment s se fait donc en temps $\Omega(Visi(s)_s + \log n + m_p + m_q)$ et $O(Visi(s)_s + (m_p + m_q)n)$ (méthode de parcours), ou en temps $O(Visi(s)_s + (m_p + m_q)\log n)$, où $Visi(s)_s$ désigne le nombre de côtés radiaux du polygone de visibilité portés par une arête du graphe de visibilité. Avec la première méthode, la borne optimale $O(\log n + Visi(s))$ peut être atteinte dans certains cas.

3.2.4 Limites de discontinuité de l'éclairage

Les côtés radiaux du polygone de visibilité calculés correspondent aux limites d'ombre portées par les objets. Cependant, ces limites d'ombre ne sont que locales. Pour un point O_p, différentes faces f peuvent avoir une arête portée par la courbe duale O_p^* dans leur chaîne supérieure, et plusieurs de ces arêtes peuvent être traversées pendant le parcours des faces de la zone de s. Le polygone de visibilité n'est donc pas forcément un polygone simple et les limites d'ombre indiquées peuvent n'être alors que des limites de pénombre.

La figure 3.20 nous montre un exemple de scène illuminée par le segment s. Le polygone de visibilité correspondant n'est pas un polygone simple (figure 3.20(a), le sens des flèches et la numérotation indiquent le sens de parcours des côtés du polygone), et le parcours des faces dans le complexe (figure 3.20b) nous montre effectivement que plusieurs limites d'ombres locales sont considérées au voisinage d'un même point (points 3,4,5,6 de la figure 3.20c).

Pour avoir la vraie limite d'ombre derrière un objet O, il faut prendre la limite de plus grande pente (ou de plus petite pente selon le cas) parmi les « limites locales d'ombres » (figure 3.20c). La structure de polygone est alors perdue.

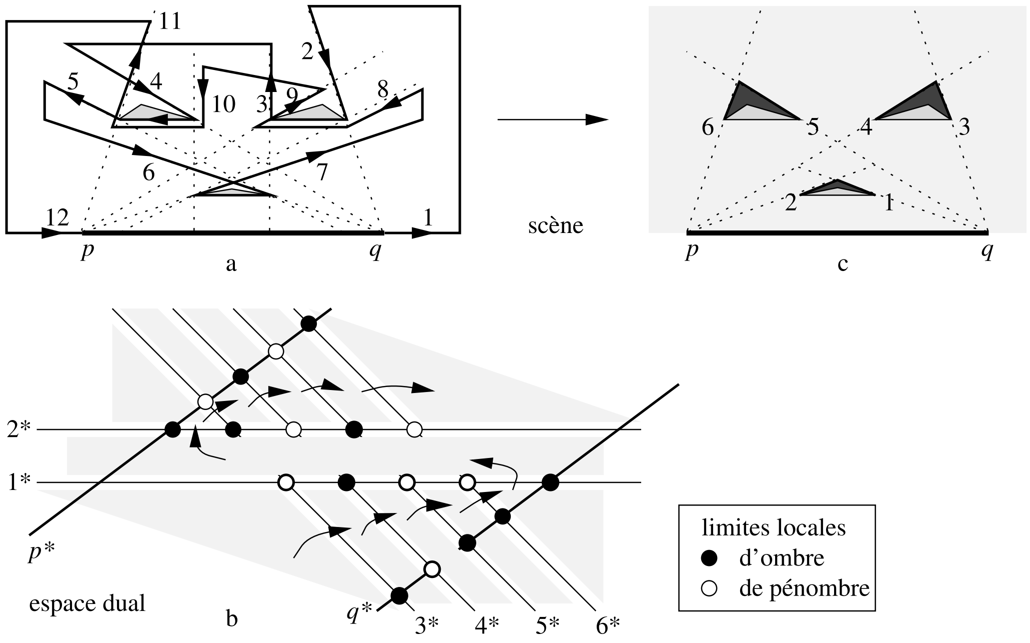


FIG. 3.20 – **a.** *Polygone de visibilité non simple de la scène.* **b.** *Parcours correspondant des faces du complexe.* **c.** *Limites d'ombre résultantes dans la scène.*

Pour certaines applications (calcul d'illumination globale par exemple), il est intéressant d'avoir en plus des limites d'ombre les limites de pénombre. Nous venons de voir que les limites d'ombre calculées ne sont que locales et peuvent n'être alors que des limites de pénombre. Cependant, il existe aussi des « vraies » limites de pénombre : ces limites séparent deux zones qui sont toutes deux (localement) éclairées, mais dont la partie de s qui les éclaire est différente.

D'une manière générale, les limites d'éclairage (ombre et pénombre) sont données par les arêtes qui relient p (resp. q) aux points de sa vue et par les arêtes (étendues) du graphe de visibilité qui coupent le segment s. Notre algorithme calcule déjà la vue autour de p et q. Nous devons donc visiter en plus toutes les arêtes du graphe coupant s et non plus seulement celles qui portent un côté du polygone de visibilité. Nous vérifions facilement que les sommets du complexe associés à ces arêtes sont toujours incidents à une chaîne supérieure d'arêtes d'une face de la zone de s.

Comme notre algorithme parcourt chaque face f du complexe de la zone de s, il suffit de visiter tous les sommets extrémité des arêtes de la chaîne supérieure de f. Ainsi, chaque limite de zone ou de pénombre est reportée, et l'est au plus deux fois. Cela permet de calculer l'éclairage de la scène en temps optimal $O(\log n + limvisi(s))$, où $limvisi(s)$ est le nombre de limites d'ombre et de pénombre.

3.2.5 Segment compris dans la scène

Nous avons supposé que la droite support de s était en dehors de l'enveloppe convexe de la scène. Ainsi tous les rayons éclairant la scène provenaient en fait de l'objet

O_∞ situé à l'infini. De cette façon, aucun événement de visibilité ne se produisait derrière s. Si s est situé n'importe où dans (l'espace libre de) la scène, alors les faces traversées peuvent être délimitées par des changements de visibilité situés derrière s. Ces changements ne modifient pas le polygone de visibilité de s (s n'éclaire que d'un côté) mais peuvent compliquer l'algorithme et augmenter le temps de calcul.

Examinons d'abord le cas où le segment éclairant s est situé hors de l'enveloppe convexe de la scène, sans que sa droite support ait nécessairement une intersection vide avec l'enveloppe convexe. Dans ce cas, des objets de la scène peuvent être situés « derrière » s, c'est-à-dire dans la partie non éclairée, et des changements de visibilité dans la scène peuvent alors survenir « derrière » s. Mais dans ces situations, le devant de s est vide : les rayons correspondant à ces changement de visibilité traversent s pour se perdre ensuite vers O_∞. Nous pouvons alors restreindre le balayage du polygone de visibilité aux directions où les objets de la scène sont situés devant s.

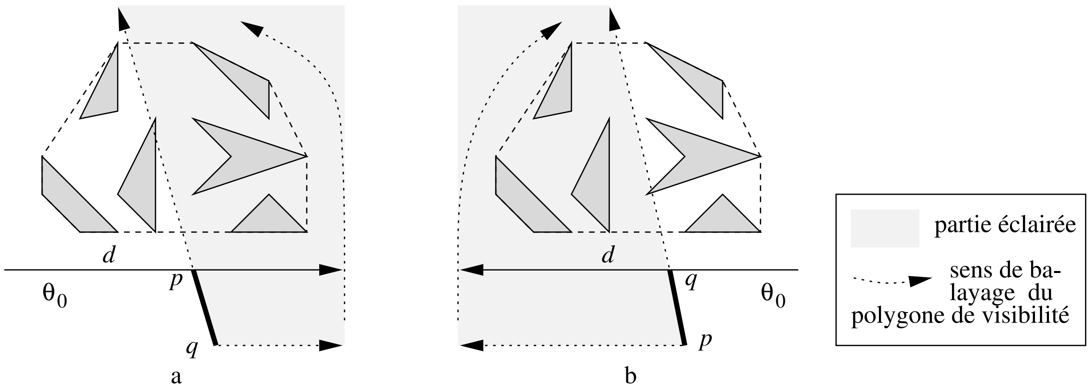


FIG. 3.21 – *Calcul du polygone de visibilité quand le segment est en dehors de l'enveloppe convexe de la scène.* **a.** *Segment en dessous.* **b.** *Segment au-dessus.*

Comme le segment s est situé en dehors de l'enveloppe convexe de la scène, il existe une droite passant par une des extrémité de s et séparant s de la scène. Si cette droite passe par p, alors les rayons de même direction que la droite et passant par s appartiennent à la la face semi-infinie inférieure du complexe. Nous prenons alors cette face comme face de départ f_d dans l'algorithme de calcul de polygone de visibilité (figure 3.21a). La face d'arrivée f_a de l'algorithme, face contenant le rayon (p, q), n'est plus la face semi-infinie supérieure du complexe. Nous rajoutons alors dans l'algorithme un test pour vérifier si la face en train d'être parcourue contient le rayon (p, q). Ce test est effectuable en temps constant une fois connues les arêtes de sorties de p^* et q^*.

Si la droite séparatrice passe par q, alors les rayons de même direction que la droite et passant par s sont dans la face semi-infinie supérieure du complexe (figure 3.21b). Pour simplifier les calculs, nous utilisons l'algorithme qui parcourt les faces de la zone de s en sens inverse de celui de l'algorithme que nous avons détaillé (les deux algorithmes sont similaires). Nous prenons alors la face semi-infinie supérieure comme face de départ et nous rajoutons un test pour vérifier si la face traversée n'est pas la face contenant le rayon (q, p).

Dans le cas où le segment s est en position quelconque en dehors de l'enveloppe convexe de la scène, les modifications apportées à l'algorithme de calcul de polygone de visibilité sont mineures : seules les faces de départ et d'arrivée sont modifiées. Il n'en va pas de même si le segment est situé à l'intérieur de la scène : lors du balayage, nous rencontrons des situations où des objets sont situés de part et d'autre du segment s, et où des changements de visibilité se produisent derrière s et ne doivent pas être pris en compte dans le calcul du polygone de visibilité.

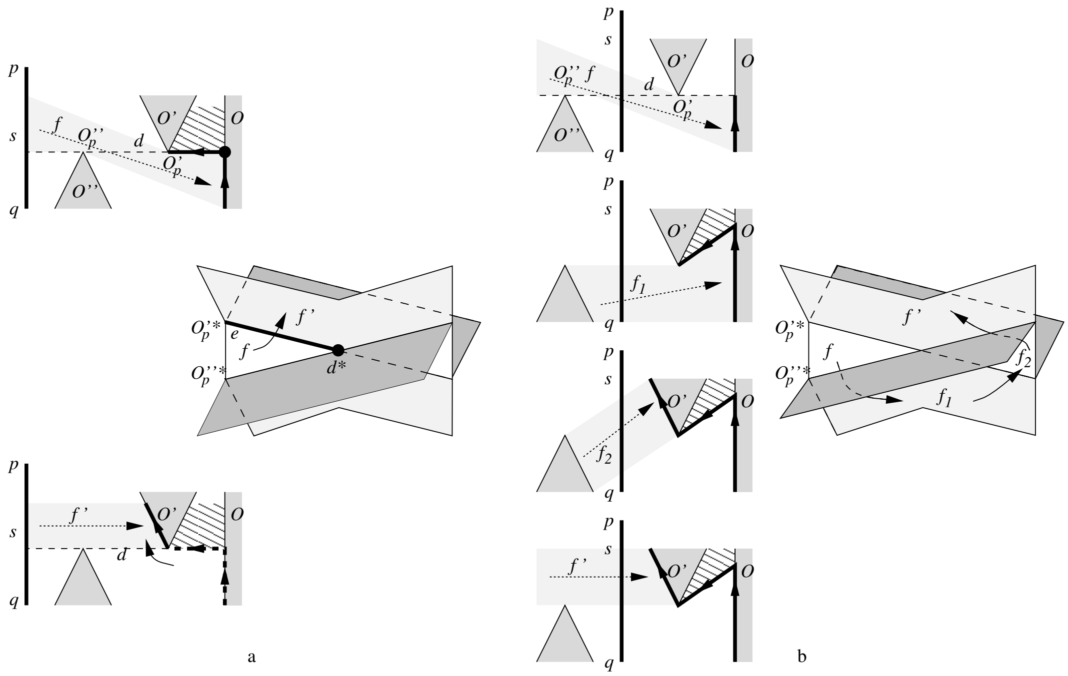


FIG. 3.22 – *Détermination de la zone d'ombre créée par le bas d'un objet.* **a.** *Changement de visibilité devant le néon.* **b.** *Changement de visibilité derrière le néon.*

Ainsi, la situation de la figure 3.16(b) où nous cherchions la limite supérieure d'éclairage sur un segment n'est plus qu'un cas particulier de la situation générale où nous cherchons la limite de l'ombre créée par le bas de l'objet O' sur l'objet O (figure 3.22). S'il y a un objet O'' entre le néon s et O', alors O'' cache le néon et la limite d'ombre est donnée par le rayon passant par (O''_p, O'_p), extrémités respectives de O'' et O' (figure 3.22a). Nous passons alors de la face f qui éclairait l'objet O à la face f' qui éclaire O' en considérant la dernière arête e de la chaîne supérieure de f : f' est la face incidente gauche-haute de e.

Si maintenant le segment s est situé entre O'' et O', bien que le complexe de visibilité de la scène soit toujours le même, nous devons agir autrement. En effet, O'' ne cache plus le néon et nous devons passer de f à f_1 pour chercher la limite d'ombre, où cette fois-ci f_1 est la face incidente droite-basse de la dernière arête de la chaîne inférieure de f. La figure 3.22(b) montre les faces alors parcourues dans le complexe avant de traverser f'.

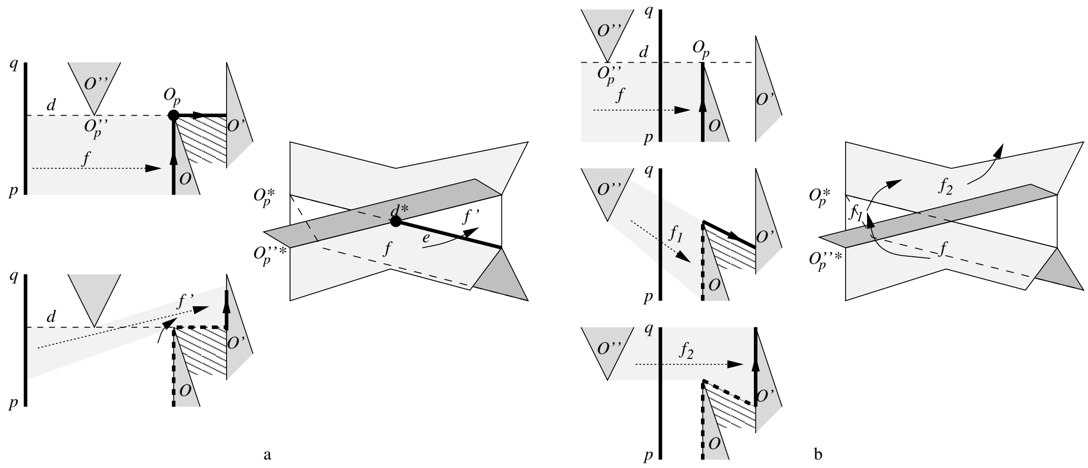


FIG. 3.23 – *Détermination de la zone d'ombre créée par le haut d'un objet.* **a.** *Changement de visibilité devant le néon.* **b.** *Changement de visibilité derrière le néon.*

Les mêmes précautions doivent être prises avec la situation de la figure 3.18(b) où nous cherchions la limite inférieure d'éclairage d'un segment : elle n'est plus qu'un cas particulier de la situation générale où nous cherchons la limite de l'ombre créée par le haut de l'objet O sur l'objet O' (figure 3.23). S'il y a un objet O'' entre le néon et O, alors O'' cache le néon et la limite d'ombre est donnée par le rayon passant par (O''_p, O_p), extrémités respectives de O'' et O (figure 3.23a). Nous passons alors de la face f qui éclairait l'objet O à la face f' qui éclaire O' en considérant la première sous-arête e de l'arête grasse droite de la chaîne supérieure de f : f' est la face incidente gauche-haute de e.

Si maintenant le segment s est situé entre O'' et O', comme pour la situation décrite précédemment, nous devons aussi agir autrement. O'' ne cache plus le néon et nous devons passer de f à f_1 pour chercher la limite d'ombre, où cette fois-ci f_1 est la face incidente droite-haute de l'arête précédent la première sous-arête grasse droite de la chaîne inférieure de f. La figure 3.23(b) montre les faces alors parcourues dans le complexe. Nous remarquons aussi que dans ce cas la face f' n'est pas parcourue.

Ceci est du au fait que les propriétés du chemin de parcours des faces de la zone de s étudiées quand s était en dehors de la scène ne sont plus valables. En particulier, la figure 3.24 montre un cas où deux faces f_1 et f_2 ont la même face suivante selon s. L'algorithme étendu de calcul de polygone de visibilité ne parcourt donc plus toutes les faces de la zone de s. Cependant les faces oubliées n'ont pas d'incidence sur le calcul du polygone de visibilité. La figure 3.23 permet de voir que la face f' de la première situation (figure 3.23a) est coupée en deux faces dans la deuxième situation (figure 3.23b) : les faces f' (similaire à la première situation) et f_2. Ce découpage est du à un obstacle O'' situé derrière le néon, obstacle qui n'a aucune influence sur l'éclairage de la scène : il ne détermine aucune limite d'ombre ou de pénombre.

La figure 3.24 montre aussi que f_2 n'a pas vraiment de face précédente selon s. L'algorithme modifié de calcul de polygone de visibilité parcourt correctement toutes les faces utiles au calcul d'éclairage de la scène et ne se perd pas dans des faces inutiles.

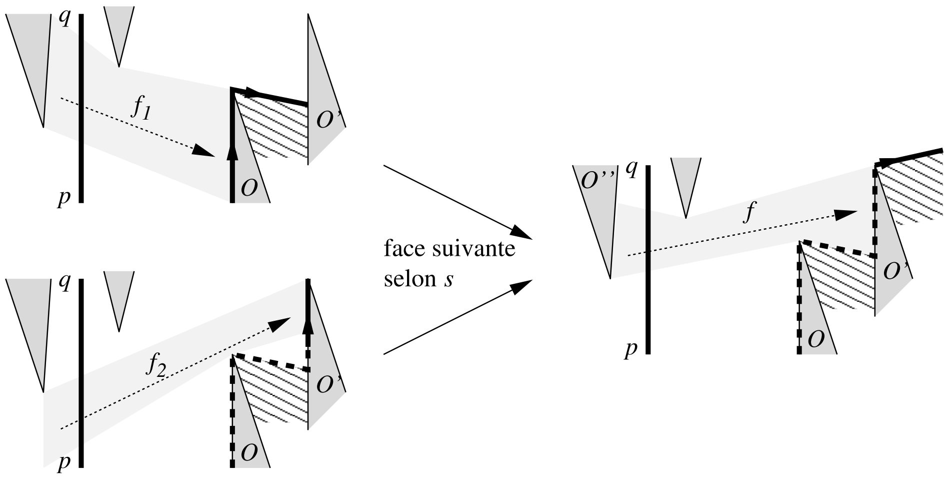


FIG. 3.24 – *Exemple où deux faces ont la même face suivante selon s.*

Il faut simplement calculer la face f_d de départ contenant le rayon (q, p). Cette face doit cette fois-ci réellement calculée : ce n'est plus une des deux faces semi-infinies du complexe. Elle peut cependant être trouvée simplement en calculant la vue autour de p et q.

L'algorithme de calcul du polygone de visibilité d'un segment s se complique fortement quand s est situé dans la scène : quand un changement de visibilité se produit, il faut vérifier s'il se produit dans la partie éclairée de la scène et agir en conséquence sinon. La complexité de l'algorithme s'en ressent. Elle est toujours en $O(m_s + (m_p + m_q) \log n)$, où m_s est toujours le nombre d'arêtes (étendues) du graphe de visibilité coupant s, mais parmi les arêtes coupant s seules celles situées dans la partie éclairée de la scène interviennent dans les limites d'éclairage.

L'algorithme perd aussi la propriété de parcours des faces de la zone de s : toutes les faces de la zone ne sont plus parcourues. Il faut alors calculer le polygone de visibilité de $[p, q]$ et de $[q, p]$ pour visiter toutes les faces de la zone.