brève introduction à une topométrie logicielle

Claude Million

L'auteur est Ingénieur ESGT et Géomètre DPLG.
Diplômé du Centre des Hautes Etudes de la Construction et de l'Institut d'Administration des Entreprises, Docteur de l'Observatoire de Paris en Photogrammétrie.
Topographie, Photogrammétrie et Etudes de Génie Civil dans des Cabinets Privés, Construction et Travaux Publics dans l'Armée Américaine et les filiales de la Caisse Française de développement.

1 - Préambule

Le choix de ce titre est un hommage rendu à l'ouvrage de SHAMOS et PREPARATA intitulé "Computational Geometry - An introduction" qui marqua le point de départ de cette discipline connue sous le nom de Computational Geometry dont le succès a suscité la création d'une Revue du même nom qui reste florissante car toujours pleine de contributions éminentes sur le sujet.

Mais quel en est le sujet ? Le titre et le contenu nous amèneraient à le traduire par "Géométrie logicielle" puisqu'il s'agit de créer des structures de données et des algorithmes propres à permettre de poser correctement des problèmes de géométrie à des ordinateurs, la partie métrique de la résolution elle-même étant, depuis longtemps, du domaine public.

Quels types de problèmes veut-on résoudre ? Alors que le géomètre raisonne juste, sur des figures parfois fausses, le problème avec les ordinateurs est d'abord de décrire la figure dans des structures de données qui remplissent le rôle des figures du géomètre et qui, par la suite, permettent de résoudre aisément le ou les problèmes posés.

Certains comprendront peut-être mieux si on donne quelques exemples de problèmes qui ont été résolus par la géométrie logicielle :

Un problème très utile, et désormais classique, est celui qui est posé par l'utilisation d'un écran tactile sur lequel l'utilisateur pose le doigt pour demander à l'ordinateur d'identifier une parcelle sur un plan, éventuellement cadastral, afin de connaître son numéro, sa section, le nom du propriétaire, sa valeur locative etc... Ce problème a été résolu par géométrie logicielle.

Il en est de même pour la triangulation de DELAUNAY et des polygones de VORONOI utilisés, notamment, pour l'interpolation des cotes de niveau dans un semis de points, et pour le tracé automatique des courbes de niveau.

On connaissait l'ouvrage [1], sans l'avoir utilisé à d'autres fins que celles de comprendre les algorithmes utilisés dans un logiciel de triangulation de DELAUNAY, quand, dans un travail récent, on s'est aperçu que le type des raisonnements, les structures des données, et les algorithmes qu'on utilisait étaient des transpositions de celles et de ceux lus dans cet ouvrage, lesquels avaient sans doute mûri à notre insu. De fait, comme M. JOURDAIN faisait de la prose sans le savoir, on faisait de la topométrie logicielle inconsciemment.

2 - La description d'un réseau topométrique

La topométrie comporte deux aspects complémentaires : en premier lieu le lever de terrain et la rédaction des plans et fichiers décrivant l'état existant, et en second lieu l'implantation d'un projet qui commence par une opération importante qui est le report d'un projet dans le système topométrique du lever suivi de l'implantation proprement dite sur le terrain qui est un report en grandeur nature à l'échelle 1/1, et peut être considérée comme l'inverse de l'opération de lever.

De fait, on va surtout parler du premier aspect de la topométrie c'est-à-dire des opérations qui suivent le lever, mais on verra, d'abord, que la saisie des mesures et leur manière de les noter n'est pas neutre quant aux structures des données qui feront suite, la manière de décrire une figure géométrique à implanter n'est pas différente de celle qui permet de décrire le réseau de base d'un lever. On voit, aussi, bien vite que la description d'un réseau topométrique de lever ou celui servant de base à une implantation ne sont que deux aspects d'un même problème.

En outre on n'abordera que l'aspect calcul automatique des points approchés en topométrie ou en géodésie, encore que le mot calcul puisse prêter à confusion, puisque c'est l'aspect logique et topologique qui sera souligné, en laissant de côté le calcul métrique des points, la compensation qui lui fait suite, et qui ne pose pas de problèmes de la même nature de ceux qu'on veut résoudre. L'objectif est de calculer automatiquement les coordonnées des points approchés sans que l'intervention d'un calculateur soit nécessaire. Une fois les points approchés calculés, la compensation est quasiment automatique elle aussi, les programmes existants laissent à l'opérateur le soin de définir les poids des mesures, et une certaine stratégie au calculateur qui est seul à pouvoir apprécier les résultats qu'il fournira à son client.

En calcul manuel l'opérateur dresse un croquis, ou mieux une mappe qui servent à guider le calculateur dans sa stratégie d'enchaînement des calculs, si la mappe est très bien faite il peut même numériser directement les points et les entrer comme points approches sans calculs, il passe alors directement à la compensation. En revanche, en calcul automatique il faut décrire le réseau mesuré par un algorithme général, en ne se servant que des mesures, c'est-à-dire sans l'intervention de l'opérateur.

2.1 - Structure des mesures et de données

Dans les opérations anciennes, les mesures étaient lues par l'opérateur sur l'appareil, puis reportées sur un carnet pour être traitées dans des tableaux de calcul, on peut faire de même en utilisant un ordinateur et un tableur, tel n'est pas notre intention.

Maintenant, l'appareil lit lui-même les mesures et les note sur un carnet électronique, l'intérêt étant d'éliminer les erreurs de lecture et de notation ou de transcription, il ne reste, en dehors des fautes opératoires grossières du débutant, que les fautes d'identification du ou des points visés ou touchés. En général les mesures ont la forme suivante, mais dans le détail les codes dépendent des appareils de mesure et des carnets électroniques utilisés, si on utilise différents appareils dans un même lever il faudra procéder à des traductions ce qui est facile :

	Termes de la mesure				
	1	2			
Code de la mesure	origine	fin	mesure	mesures	annexes
12	F-1790-E	G-4250-è	144.8792	1.520	0.275
	Station	Point visé			

Le code de la mesure indique, par exemple, une mesure de distance zénithale, l'origine de la mesure est le numéro de la station, la fin de la mesure le numéro du point visé, suit la mesure elle-même, dans l'unité choisie.

Dans le cas présent les mesures annexes seraient la hauteur de l'appareil et la hauteur du point visé. Souvent on appelle l'origine de la mesure : station, et l'autre extrémité : point visé, même si aucune visée n'a été faite, par exemple pour une mesure de longueur ou une dénivelée, en fait c'est la signification du code de la mesure qui indique la vraie nature d'origine et de fin, et si les termes de mesures annexes ont un sens, et quel est ce sens.

La ligne représente une mesure mais, parfois, sur certains appareils l'origine de la mesure est regroupée en tête de la station avec la hauteur d'appareil ce qui évite leur répétition, mais sous-entend leur mise en commun sur les lignes qui suivent, on peut y ajouter aussi des données météorologiques, par exemple, notamment pour les mesures géodésiques. Souvent chaque donnée est individuelle et précédée d'un code.

Exemple :

Code	Hauteur d'appareil
54	1.246

Et les informations sont empilées ainsi les unes à la suite des autres. Il convient alors de regrouper toutes les informations intéressant une mesure complète sur une seule ligne, c'est-à-dire la mesure principale et les mesures annexes.

La multiplicité des solutions adoptées par les différents fabricants impose un choix unique qui est celui qu'on a montré et dans lequel seront traduits tous les enregistrements des différents appareils utilisés, les logiciels de transformation, dans la mesure où les caractères enregistrés par le carnet électronique ne sont pas des fantaisies informatiques comme les codes-barres, sont très simples ; heureusement, dans les carnets électroniques, le code ASCII semble avoir prévalu, ce qui simplifie tout.

Pour nous, après traduction, une ligne sera une mesure dont les trois données topologiques essentielles seront le code, et les termes de la mesure. Cela suffit pour la description topologique d'un réseau topométrique.

3 - Topologie du réseau

Il convient maintenant de décrire le réseau à la machine, c'est l'équivalent de l'acte de dresser le croquis du réseau ou de dessiner la mappe.

3.1 - Liste des points

Sans préjuger du fait qu'ils soient déterminables ou pas par les moyens topométriques employés, il faut dresser une liste des points triés suivant un ordre alphabétique quelconque.

On a utilisé l'alphabet ASCII étendu qui comporte 256 lettres, dont certaines sont muettes, c'est-à-dire n'apparaissent pas à l'écran, il est facile de limiter cet alphabet aux "lettres" (lettres, chiffres, tirets, points, accents, signes) qu'on veut utiliser, tout autre alphabet ferait l'affaire, pourvu qu'à chaque signe soit affecté un "chiffre" (ici un octet) afin de permettre un classement par tri biunivoque : rang → nom, nom → rang, de façon à ce que chaque point ait un nom, et un rang dans le classement des noms.

Ceci se fait à partir de l'ensemble des mesures, à partir des deux termes, en triant les origines et les fins en vrac, sans faire de distinction particulière entre les points et les stations, le programme de tri supprime automatiquement les doublons (A,B,C,D,E,...).

La liste des rangs des points (i,j,k,l,m,n) sert de base à tous les tableaux de connectivité.

4 - Les tableaux de connectivité

On a vu que chaque mesure était reconnue par un code, numérique dans notre cas, mais rien n'empêchait, s'il en était besoin d'imaginer un autre code qui soit plus explicite pour l'opérateur, on peut aussi envisager des numéros de codes qui facilitent la recherche ultérieure, des nombres premiers par exemple.

A chaque type de mesure correspondra un tableau logique, une matrice carrée, de connectivité.

Les tableaux de connectivité, même s'ils ne sont que virtuels, on verra plus loin comment, sont de type booléen c'est-à-dire que chaque case renseignée du tableau ne comporte qu'une information du type : connecté ou non connecté, et rien d'autre, c'est-à-dire, pour rester concret : A a reçu une visée venant de B ou, à l'inverse, A n'a pas reçu une visée venant de B et rien de plus.

L'usage a amené à ne considérer pour l'instant que neuf types de mesures topométriques, le mot "mesures" étant entendu dans son sens "très" large, car certaines contraintes, comme, par exemple, la reprise de station ou centrage forcé, ont été considérées comme des mesures, on a suivi en cela les principes des compensations par les moindre carrés par la méthode des équations d'observation qui traitent, numériquement, certaines conditions ou contraintes comme des mesures, c'est-à-dire qu'on écrit une équation d'observation pour exprimer la contrainte, et on la traite comme une observation en lui donnant un poids approprié.

Or en "Computationnal Geometry" on fait de même, toute information, par exemple que trois points sont alignés, est traitée comme une mesure.

4.1 - Détail des tableaux

Chaque type de mesure a ses caractéristiques propres qu'on va évoquer :

De fait on a trouvé deux types de mesures : dans l'une appelée type visée les deux termes ne sont pas interchangeables : un des termes est la station, l'autre est le point visé. Leurs tableaux représentatifs sont carrés.

En revanche, dans le type appelé distance, les deux termes sont interchangeables, leurs tableaux représentatifs sont des triangles, si on connaît la distance A B on connaît aussi la distance B A, et cette information n'a pas à être dupliquée. De même pour le tableau des dénivelées (den(..)), il est facile de ne noter que den(A,B) car on a la relation métrique (par opposition à logique) :

den(B,A) = -den(A,B).

Prenons quelques exemples : un tableau de type visée, soit le tableau de connectivité des mesures d'angles horizontaux, on note :

ah(i,j) = vrai si du point A de rang i on a visé un point B de rang j, et ah(i,j) = faux dans le cas contraire.

Les termes diagonaux de la matrice n'ayant aucun sens, le point i ne pouvant se viser lui-même, on les a utilisés pour faire savoir au logiciel si on connaît, ou si on ne connaît pas le Go de la station.

ah(i,i) = vrai, indique par conséquent qu'au point A de rang i on connaît le Go de la station.

De cette manière si on a : ah(i,j) = vrai et ah(i,i) = vrai, alors le gisement A → B est connu. Les termes si, et, alors étant les synonymes explicites de ceux utilisés en informatique et de leurs équivalents de l'algèbre de BOOLE, on les soulignera pour bien marquer qu'ils représentent des opérations booléennes.

On passe du logique : ah(i,j) = vrai et ah(i,i) = vrai au métrique G(A,B)= 344.2458 grades (Ces chiffres étant évidemment des illustrations arbitraires).

Les tableaux de connectivité des distances zénithales sont de même nature, on les notera dz(i,j), comme précédemment i et j sont les deux termes de la mesure i est la station j le point visé. Noter que dz(i,j) = vrai implique que la mesure de la distance zénithale A → B est faite, mais aussi qu'on a mesuré la hauteur de la station et la hauteur du voyant du point visé ; car attention, s'il manque une seule de ces trois mesure alors dz(i,j) = faux.

Prenons un exemple de tableau de type distance pour montrer les possibilités de gagner des positions mémoires : les distances inclinées mesurées di et les distances horizontales qui sont le plus souvent calculées dh, sont deux tableaux triangulaires, car si di(i,j) ou dh(i,j) = vrai alors di(j,i) ou dj(j,i) = vrai, comme il est inutile de noter deux fois la même chose, et pour gagner de la place on les notent dans le même tableau d(i,j), avec la condition suivante : Si i<j alors c'est dh(i,j) = vrai ou faux ; Sinon c'est di(i,j) = vrai ou faux.

Dans ce cas dh(i,j) et di(i,j) sont des fonctions booléennes de recherche dans un tableau, et non plus le tableau carré de données lui-même, qui, dans ce cas, est d(i,j) ; des fonctions booléennes n'ont pour résultat que vrai ou faux, si bien qu'en cours de raisonnement on peut ne pas faire de différence entre les valeurs d'un tableau carré et le résultat d'une fonction de recherche dont l'utilité et les notations sont identiques. Attention ceci est vrai en analyse, et faux en programmation où les notations sont, évidemment, différentes.

Ceci va nous permettre d'aller plus loin, constatant que dans les grands réseaux les matrices de connectivité sont creuses, et même, très creuses, on peut décider de les remplacer par des listes simplement composées des rangs des termes de la mesure (rang de la station et rang du point visé par exemple) et les noter en un seul terme appelé "Nombre" :

Nombre = 1000*rang (station) + rang (du point visé), la décomposition se faisant par les fonctions inverses suivantes :

rang(station) = entier(Nombre divisé par 1000),

rang(point visé) = Modulo 1000 (Nombre).

Nombre peut être trié dans sa liste pour faciliter sa recherche dichotomique. A la fin de la recherche, dans une liste relativement courte, si on a trouvé le nombre désignant les deux termes de la mesure, la fonction de recherche prendra la valeur : "vrai", dans le cas contraire, et par défaut, elle prendra la valeur "faux".

On remarque encore que la nature informatique même des tableau de connectivité tableau, liste, est sans importance pour le raisonnement logiciel, alors qu'il est, évidemment, de première importance pour la programmation.

Le tableau des dénivelées dn(i,j) est très comparable au tableau des longueurs, il mélange les connectivités de deux types valeurs : des mesures de dénivelées directes, faites au niveau par exemple, et des dénivelées calculées à partir des distances linéaires et des distances zénithales, en outre si deux altitudes sont connues par tout moyen la dénivelée entre ces deux points est connue et ce fait doit être noté, car l'expérience montre que certains excellents programmes sont limités faute d'avoir pris ce fait élémentaire en compte.

Ce détail nous amène à souligner qu'il est nécessaire, en topométrie logicielle, de noter dans des tableaux des informations qui semblent triviales ou redondantes en calcul manuel. Par exemple avant de calculer un point, il faut s'assurer si cela est bien nécessaire, c'est-à-dire vérifier qu'il n'est pas... connu. Le renseignement du tableau des dénivelées est un second exemple de cette nécessité de noter des informations apparemment évidentes et qui ne le sont certainement pas d'un point de vue strictement logiciel.

5 - Tableaux des résultats

Certains tableaux correspondent à des résultats et non à des mesures, il n'expriment pas une relation entre deux points, mais donnent une information sur un seul point, ils restent donc unidimensionnels. Ce ne sont pas de tableaux de connectivité mais des tableaux indiquant la situation des points connus et inconnus.

Par exemple, on exprime qu'un point de rang i est connu ou inconnu dans ses coordonnées planimétriques par l'expression booléenne :

xy(i) = vrai ou xy(i) = faux.

On écrit de même que l'altitude du point de rang i est connue ou inconnue par :

z(i) = vrai ou z(i) = faux.

Il existe d'autres tableaux unidimensionnels dont les informations permettent d'accélérer la recherche (voir 6 - Moteur d'inférence).

Par exemple st(i) = vrai indique que le point de rang i a été stationné, de même pv(i)= nombre de visées, indique de combien d'autres points le point de rang i a été visé, de même la diagonale de d est utilisée car d(i,i) = nombre de mesures de longueur ayant i pour terme.

On verra que ces tableaux unidimensionnels servent d'accélérateurs de recherches car, par exemple, si pv(i) < 2 on ne cherchera pas si une intersection du point de rang i est possible, de même, si st(i) = faux on ne cherchera pas si le calcul du Go du point de rang i est possible etc... et cela sans entrer dans un niveau de recherche élevé (nous sommes dans une recherche à un seul paramètre).

6 - Le moteur d'inférence

On a pris l'habitude de désigner sous ce nom bien pompeux le logiciel qui utilise ces informations pour les transformer en décisions d'accomplir des actions possibles.

Si conditions... alors FAIRE

On passe alors du domaine topologique, en minuscules, au domaine métrique, en majuscules, dont on parlera peu puisqu'il est évident que la topométrie logicielle ne peut être abordée que lorsque la topométrie tout court est totalement assimilée.

Les notations que nous avons explicitées vont nous permettre de donner quelques exemples notés en pseudo-code.

Pour des facilités mnémoniques on a attribué la lettre minuscule i qui désigne le rang du point A (lettre majuscule) au point inconnu, les autres lettres (j, k, l, m,...) sont affectées à des points inconnus.

Le moteur d'inférence est des plus simples, son schéma est le suivant :

Si Condition 1 et Condition 2 et condition n faire

CALCULER (Calcul topométrique traditionnel)

Si opération de calcul et de transfert des résultats dans les tableaux s'est bien passée alors

Début

Noter que les résultats désirés sont connus.

AFFICHER CE QU'ON VIENT DE FAIRE

Fin

Bien noter que le calcul topométrique peut être aussi bien un calcul géodésique traditionnel, on ne préjuge en rien de la nature des calculs réalisés, ni du référentiel, ni du type de coordonnées déterminé.

Il faut rappeler un point important : dans une chaîne de condition Si...et...et , dès qu'une des conditions est calculée ou constatée fausse le calcul logique se débranche, car toutes les conditions doivent être remplies sans exception, on a donc intérêt à placer en premières conditions celles qui ont le plus grand nombre de chances de provoquer un rejet, et de placer en amont les conditions qui n'impliquent qu'une recherche dans un seul tableau logique, ou mieux, une seule liste logique. Mais attention, la condition ou a la propriété inverse, il faut placer en tête la condition la plus probable car le débranchement a lieu dès qu'une des conditions proposées est réalisée.

Et correspond à une multiplication logique si un des termes est nul le résultat est nul ; en revanche, ou ne correspond pas à une addition logique, si une seule des conditions ou est vraie l'ensemble est vrai. Donc attention aux parenthèses dans les relations où se mêlent les deux termes.

6.1 - Calcul principaux

On appelle ainsi les sous-programmes qui décident directement de la possibilité ou de l'impossibilité de réalisation des calculs de coordonnées de points inconnus.

Prenons le plus simple et le premier de tous, le rayonnement.

On vérifie dans l'ordre :

A - Sur un seul indice i

1°/Que le point qu'on veut calculer n'est pas connu (Condition triviale absolument essentielle si on ne veut pas tourner en rond).

2°/ Que ce point A de rang i est un point visé.

3°/ Que ce point A de rang i est un des termes d'une mesure de longueur.

B - Sur deux indices i, et j :

4°/ Que le point B de rang j est connu en coordonnées planimétriques.

5°/ Que le point B de rang j est une station.

6°/ Qu'on connaît le Go du point B de rang j.

7°/ Qu'il existe une visée de j → i.

8°/ Qu'il existe une mesure de longueur j == i.

Ce qui donne en pseudo-code : Cherche-Rayonnement

```
Pour i = 1 à nombre de points faire
    Si xy(i) = faux et pv(i) = vrai et d(i,i) = vrai alors
        Pour j = 1 à nombre de points faire
            Si xy(j) = vrai et st(j) = vrai et ah(j,j) = vrai et
            ah(j,i) = vrai et dh(i,j) = vrai alors
                CALCULER LE POINT A RAYONNE A PARTIR DE B
                PORTER LES RESULTATS DANS LES TABLEAUX
            Si l'opération s'est bien déroulée alors
                xy(i) = vrai, ah(j,i) = faux, dh(i,j) = faux...
            Ceci pour éviter tout effet de circularité.
```

Un autre exemple de la recherche de la possibilité de calcul d'un point inconnu, le relèvement.

Il s'agit toujours de la partie topologique du programme ; la métrique est connue de tous.

Toutefois, dans le cas du relèvement comme dans d'autres, le programme de calcul métrique lui-même doit être précédé d'un contrôle métrique de la possibilité de calcul, en vérifiant, par exemple, que le point inconnu et les trois points connus ne sont pas sur un même cercle.

Dans les cas semblables, lorsque la vérification montre que le calcul métrique serait trop imprécis, le programme de calcul est interrompu, puis le programme principal (topologique) doit être repris, à l'endroit où il s'était débranché, pour poursuivre la recherche topologique sur d'autres points et dans d'autres configurations.

Sous-programme Cherche-Relèvement (En pseudocode)

Légende des variables

npcxy : nombre de points connus en planimétrie.

St(i) : nombre de visées issues de la station i si le point i n'est pas une station st(i) = 0.

pv(i) : nombre de visées convergentes vers le point i, si le point n'est pas visé pv(i)=0

xy(i) : vrai, le point i est connu en planimétrie, faux il ne l'est pas.

ah(i,j) : vrai il existe une visée de la station i vers le point j, faux l'inverse.

ah(i,i) : vrai, on connaît le Go de la station i, faux l'inverse

```
Début
Si npcxy > 2 alors :
  Faire i de 1 à nombre de points
  début
    Si st(i) > 2 et xy(i) = Faux alors :
      Faire j de 1 à nombre de points
      début
        Si (i<>j) et xy(j) = vrai et ah(i,j) = vrai alors :
          Faire k de j+1 à nombre de points
          début
            Si (k<>i) et xy(k) = vrai et ah(i,k) = vrai alors
              Faire l de k+1 à nombre de points
              début
                Si (l<>i) et xy(l) = vrai et ah(i,l) alors :
                Début
                  Opération métrique Relèvement(i, j, k, l)
                  Opération métrique Go(i)
                  xy(i) = vrai ah(i,i) = vrai
                  st(i) = st(i) - 3
                  pv(j) = pv(j) - 1
                  pv(k) = pv(k) - 1
                  pv(l) = pv(l) - 1
                  Si z(i) = faux alors
                    Si z(j) = vrai et dz(i,j) = vrai alors
                      Opération altitude (i par j) ;
                      z(i) = vrai ;
                    Sinon Si z(k) = vrai et dz(i,k) = vrai
                      alors opération altitude (i par k) ;
                      z(i) = vrai ;
                      Sinon Si z(l) = vrai et
                        dz(i,l) = vrai ; alors
                        Opération (i par l) ;
                        z(i) = vrai ;
                  ah(i,j) = faux ; ah(i,k) = faux ; ah(i,l) = faux
                  Fait = vrai Retour au programme appelant
                Fin(si)
              Fin(l)
            Fin(k)
          Fin(j)
        Fin(i)
      Fin
```

```
{                                                  }n
{    (                                        } j=n }
{    {    {                              } k=n}     }
{    {    {    {                  } l=n }     }     }
{ ?  { ?  { ?  { ? Relèvement (i, j, k, l) ? } ?  } ?  } ?  } ?
{    {    {    { l=k+1            }     }     }     }
{    {    { k=j+1                       }     }     }
{    { j=1                                    }     }
{ i=1                                               }
```

Squelette de l'algorithme de recherche d'un relèvement

n : Nombre de points

Les petites parenthèses superposées remplacent une grande parenthèse enveloppante.

BIBLIOGRAPHIE

[1] 1985 F.P. PREPARATA et M.I. SHAMOS Computational Geometry-An - Introduction : Springer-Verlag.

[2] 1994 C. MILLION : Le calcul automatique des points approchés : Revue Geotop n°132 - Mars