ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE DE MAÎTRISE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE DE LA PRODUCTION AUTOMATISÉE
M. ING.

PAR
JULIE DUFORT

ESTIMATION AUTOMATISÉE DE LA HAUTEUR DES ARBRES À PARTIR DE DONNÉES D'ALTIMÉTRIE LASER.

MONTRÉAL, DÉCEMBRE 2000

CE MÉMOIRE À ÉTÉ ÉVALUÉ PAR UN JURY COMPOSÉ DE :

- M. Richard Lepage, professeur-tuteur et professeur au département de génie de la production automatisée à l'École de technologie supérieure
- M. Benoît St-Onge, professeur-cotuteur et professeur au département de géographie à l'Université du Québec à Montréal
- M. Claude Olivier, président du jury d'évaluation et professeur au département de génie de la production automatisée à l'École de technologie supérieure
- M. Alain Croteau, professeur au département de génie de la production automatisée à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE PRÉSENTATION DEVANT CE JURY ET UN

PUBLIC

LE 6 DÉCEMBRE 2000

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ESTIMATION AUTOMATISÉE DE LA HAUTEUR DES ARBRES À PARTIR DE DONNÉES D'ALTIMÉTRIE LASER

Julie Dufort

SOMMAIRE

Ce projet étudie la possibilité de calculer la hauteur des arbres de la forêt boréale de façon précise et automatique. Les données utilisées sont des images produites à partir de mesures d'altimétrie laser, ainsi qu'une image multispectrale de la région étudiée. En séparant tout le processus de calcul de hauteurs en étapes intermédiaires, exécutées sur ordinateur de façon autonome, nous voulons aussi montrer que le processus est automatisable. Les étapes du projet sont les suivantes : l'acquisition des données, la conversion des données, l'extraction des contours des arbres, l'identification des arbres valides, l'extraction des données pour chacun des arbres et finalement le calcul de la hauteur de chacun des arbres identifiés. L'intervention humaine est cependant encore nécessaire entre l'identification de l'arbre et l'extraction de ses données, entre l'extraction des données et le calcul des hauteurs et à certains points du calcul des hauteurs. Les résultats obtenus répondent en partie aux objectifs fixés. Le filtrage donne des résultats très encourageants quant à la délimitation des couronnes, même si certaines d'elles sont encore confondues avec des voisines. De plus, pour la majorité des arbres, la hauteur calculée s'approche de la valeur réelle.

ABSTRACT

This project studies the possibility of calculating the height of trees in a boreal forest in a precise and automatic way. The used data are images produced from measurements of LIDAR, as well as multispectral image of the studied region. By separating all the process of calculation of heights in intermediate steps, executed on computer in a autonomous way, we as well want to show that it is possible to automate the process. The steps of the project are the following ones: the acquisition of the data, the conversion of the data, the extraction of the outlines of trees, the identification of the valid trees, the extraction of the data for each of the trees and finally the calculation of the height for each identified trees. The human intervention is however still necessary between the identification of the tree and the extraction of its data, between the extraction of the data and the calculation of the heights and in certain steps of the calculation of the heights. The obtained results answer well enough the fixed objectives. The filtering gave very encouraging results for the identification of crowns, even if some of them are still confused (merged) with neighbours. Also, the heights of trees were calculated near the real value for the majority of trees.

AVANT-PROPOS

Le sujet de ce mémoire m'a été proposé par Richard Lepage ing. Ph. D. professeur au département de génie de la production automatisée à l'École de technologie supérieure, ainsi que par Benoît A. St-Onge Ph.D. professeur au département de géographie de l'Université du Québec à Montréal.

La collaboration entre ces deux professeurs remonte à 1995 et elle a permis des avancements dans le domaine de la télédétection grâce à l'apport des technologies propres au génie telles que les techniques de traitement d'images.

Pour travailler sur ce projet, j'ai reçu une bourse versée conjointement par mes directeurs de maîtrise. Les fonds provenaient du CRSNG (subvention de recherche individuelle), que je tiens à remercier.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS ET SIGLES

2D : Deux dimensions

3D : Trois dimensions

CRSNG : Conseil de recherches en sciences naturelles et en génie du Canada

DoG : Différence de Gaussiennes

GPS : Global positionning system

KBV : Knowledge Based Vision (progiciel de traitement et d'analyse d'images)

LIVIA : Laboratoire d'imagerie, de vision et d'intelligence artificielle

LoG : Laplacien de Gaussienne

INTRODUCTION

L'industrie forestière est aujourd'hui confrontée à des problèmes de rupture de stock, les arbres étant de plus en plus difficile d'accès. Une grande partie du territoire de la forêt boréale (au sud) ayant été épuisée, on doit toujours aller plus loin vers le nord pour couper les arbres. Aussi, ces arbres nordiques sont plus petits et ont une croissance plus lente. Ce phénomène amène deux problématiques, une sociale et une autre financière.

Depuis plusieurs années, le débat entourant la coupe des arbres en forêt boréale est très actif. Le film « L'Erreur Boréale[1] », qui a beaucoup ravivé le débat sur le sujet, démontre bien la préoccupation des gens face à ce phénomène. Certaines personnes suggèrent qu'il est très dommageable pour la faune et la flore de faire des coupes à blanc. En effet, lors de coupes massives, les habitats de certains animaux sont détruits et par conséquent ils doivent se déplacer pour retrouver un nouvel habitat convenable. Pour cette raison, une planification des secteurs de coupe est nécessaire du point de vue écologique, afin de laisser des parties de forêt intactes en nombre et en surface suffisants pour pouvoir accueillir les animaux délogés de leur habitat par les coupes de bois. De plus, un autre problème se pose lors de la regénération de la forêt. En effet, les arbres coupés sont en majorité des épinettes et d'autres conifères. Lors du reboisement, ce sont généralement d'autres espèces d'arbres, en majorité des feuillus, qui ont tendance à croître plus rapidement. Cela entraîne des problèmes que les compagnies forestières tentent de solutionner en faisant du reboisement. Par contre, il semble qu'il faille surveiller les plantations de

---

[1] Richard Desjardins et Robert Monderie, *L'Erreur Boréale (documentaire)*, Office national du film du Canada, production Bernadette Payeur et Éric Michel, 1999, 68 minutes.

nouvelles épinettes, ce que les compagnies ne sont pas nécessairement toujours prêtes à faire en envoyant du personnel sur le terrain.

Financièrement, il est de moins en moins rentable d'aller chercher le bois toujours plus au nord. Les coûts associés à ce phénomène sont causés, entre autre, par la construction de chemins d'accès au bois. En effet, un kilomètre de chemin coûte entre 10,000$ et 30,000$. Il est donc crucial de bien connaître les ressources d'un site de coupe avant de décider de construire un chemin pour y accéder. Aussi, pour déterminer l'emplacement des sites de coupe possibles, d'autres coûts sont associés. Ainsi, l'envoi de personnel pour la mesure des arbres sur le terrain engendre des coûts assez importants. De plus, l'accès aux sites est souvent très difficile et la mesure des arbres assez imprécise et longue à effectuer. Effectivement, dans la forêt boréale, les arbres sont très rapprochés les uns des autres, ce qui rend premièrement la marche en forêt difficile et, deuxièmement, la mesure de hauteur longue à effectuer, car pour faire ces mesures, on doit reculer et viser la cîme de l'arbre avec un télémètre (appareil de triangulation). Cette tâche est d'autant plus difficile que les arbres sont rapprochés. Aussi, pour obtenir un nombre suffisant de mesures terrain afin d'évaluer les volumes de bois sur plusieurs territoires, on a besoin soit d'un personnel nombreux, soit de beaucoup de temps, ce qui résulte en des dépenses importantes.

Dans cette optique, une bonne méthode de planification de coupe et de regénération des forêts doit être adoptée. Pour ce faire, les compagnies forestières doivent se doter de tous les outils nécessaires. Nous voulons, dans ce projet, développer une méthode fiable et automatique de localisation et de mesure des arbres individuels dans la forêt boréale par altimétrie laser, également appelée « LIDAR » (Light Detection and Ranging). Cette nouvelle

méthode devrait être relativement peu coûteuse et permettre de couvrir un très grand territoire rapidement. Nous espérons, avec cette méthode, apporter une solution de rechange performante pour la planification de la coupe du bois (puisque la mesure de la hauteur des arbres est directement reliée au volume de bois) et nous espérons qu'éventuellement elle servira aussi à la surveillance des territoires en regénération. Nous espérons aussi que grâce à cette méthode, il deviendra plus facile dans le futur de faire une bonne planification des ressources et de trouver des solutions apportant un certain équilibre pour la faune et la flore.

Nous poursuivons ce document avec une revue de littérature où nous traiterons de l'état de l'art en altimétrie laser, en ressources forestières et en traitement d'images. Suivra un chapitre portant sur la problématique et les objectifs du projet. Viendra ensuite le chapitre portant sur la région d'étude et des données. Un autre chapitre couvrira la méthodologie utilisée et le dernier chapitre exposera les résultats obtenus. Tout cela sera finalement suivi d'une discussion et d'une conclusion.

CHAPITRE 1
REVUE DE LITTÉRATURE

1.1 L'altimétrie laser et ses applications en forêt

L'altimétrie laser par balayage est une nouvelle technologie pour la génération de modèles de terrain et de modèles de surface [1]. Son développement a commencé au cours des années 1970-1980. La technologie du GPS (Global Positionning System) a solutionné le problème critique du positionnement. La preuve de la grande précision de ce système a été faite entre les années 1988 et 1993 à l'Université Stuttgart. Les années suivantes ont vu le développement de plusieurs systèmes originaux de numérisation. Les applications pratiques se sont ensuite succédées.

Le développement de l'altimétrie laser a été d'ordre technologique. Les premiers modèles utilisaient des lasers pulsés utilisant l'infrarouge, qui émettaient un signal de retour clair après réflexion et diffusion sur le sol. Plus récemment, des lasers à onde continue ont aussi été utilisés. Ces lasers obtiennent des données par la mesure de la phase de l'onde réfléchie. Aussi, grâce au GPS, on obtient un bon positionnement par rapport à un référentiel externe.

L'altimétrie laser donne des résultats géométriques en terme de distance, d'altitude et de coordonnées. La précision verticale du système est de l'ordre du décimètre et plusieurs opèrent à une altitude de l'ordre de 1000 m, l'angle du faiseau étant de $\pm$ 20 à 30°. Plusieurs systèmes donnent aussi de l'information sur l'intensité du signal reçu. La fréquence de mesure est aussi d'une grande importance. Les systèmes actuels fonctionnent entre 2 et 50 KHz. Tenant

compte de la hauteur de vol, on obtient entre 1 point par 20 m^2 et 20 points par m^2 au sol. Par contre, le nombre de points au sol dépend de plusieurs facteurs, dont : la hauteur de vol, la vitesse de vol, le rythme des pulsations et l'angle du faisceau laser. L'altimètre laser ne permet pas de viser un point particulier au sol.

Les objets n'ayant pas une surface bien définie, comme les arbres, peuvent produire plusieurs réflexions enregistrables du laser. En appliquant des modèles mathématiques, on peut en dériver la surface du sol. De plus, cela permet d'avoir de l'information sur le couvert végétal.

Blair et al. [2], décrivent le principe de fonctionnement d'un altimètre laser, le LVIS (Laser Vegetation Imaging Sensor), développé par la NASA. L'appareil utilisé pour le présent projet n'est pas le LVIS, mais son fonctionnement est basé sur les mêmes principes, c'est la raison pour laquelle nous vous le présentons ici. Le LVIS est un capteur à pulsations (de moins de 10 ns) opérant à une altitude jusqu'à 10km et avec un champs de vision potentiel de 7° dans lequel les points sont espacés aléatoirement. Le balayage est effectué à l'aide d'un miroir dirigé par un galvanomètre qui contrôle le laser dans le champs de vue du téléscope. Le pas du laser peut varier de 1 à 80 m au sol et est déterminé par l'altitude de vol et par la distance focale de la lentille.

Les buts de conception du LVIS étaient d'obtenir un grand champs de balayage, d'avoir une capacité de vol à haute altitude, d'obtenir des patrons d'échantillonage variables, d'obtenir une empreinte au sol variable, d'avoir une bonne précision des mesures et un suivi continu du sol. L'empreinte du laser au sol est la zone circulaire où le laser frappe le sol. Plus la distance est grande entre l'émetteur et la zone de réception, plus l'empreinte au sol est grande. Pour obtenir un grand champs de balayage, en utilisant un télescope à grande

ouverture et une petite photodiode à avalanche, le champs de vue du télescope est balayé mécaniquement à l'aide d'un miroir muni d'un galvanomètre. Pour l'obtention de mesures précises avec une large empreinte au sol, il faut qu'une compensation soit faite pour annuler la distorsion induite par les surfaces complexes sur le retour du laser.

Le télescope recevant le signal de retour est constitué d'une lentille permettant de diriger la lumière. Un rayon de lumière est ainsi produit et ensuite capté par un miroir à balayage, situé à la sortie de la pupille. Celle-ci dirige le rayon dans un filtre passe-bande et dans un condenseur sphérique faisant le focus sur le détecteur.

Le transmetteur est une diode laser refroidie à l'eau de type oscillateur. Le laser est logé dans une cavité hermétique en aluminium qui mesure 45 X 13 X 13 cm. La fréquence de balayage peut aller jusqu'à 500 Hz, le laser émet une onde de type Gaussienne de 5 nJ, 10 ns et la longueur d'onde est de 1064 nm.

Un seul détecteur effectue les opérations suivantes dans le LVIS : discrimination du signal, mise à l'échelle, enregistrement de la forme d'onde transmise et reçue.

L'enregistrement du signal est codifié sur 8 bits et des statistiques quant à la moyemme et à l'écart type sont calculées en temps réel pour chaque pulsation. Un algorithme de détection du sol fonctionnant aussi en temps réel recherche une fenêtre de 2 km centrée automatiquement sur une localisation valable du sol trouvée précédemment.

La tâche principale du contrôleur de ce système est de récolter les données sur les formes d'onde envoyées et reçues ainsi que d'afficher l'échelle

des données et la forme d'onde pour l'évaluation en temps réel des performances des instruments de mesure. De plus, le système sauvegarde les données binaires associées aux mesures. Ce système contrôle aussi la position du miroir à balayage, l'algorithme de recherche du sol, et l'interface avec le GPS. Les données sont enregistrées sur un disque dur et toutes les variations d'amplitude du signal émis et du retour capté sont aussi enregistrées, ainsi que la distance entre les formes d'onde, les statistiques de bruit, les donneés GPS, l'altitude de vol, l'angle de balayage et l'heure. Un autre GPS au sol permettra, par la suite, de calculer la trajectoire de vol à 10 cm près en hauteur.

L'altimètre laser peut être utilisé dans différents types d'avion tels : un deux moteurs commercial ou un C-130 Hercules. L'installation et l'ajustement de l'équipement incluant un GPS est fait avant le décollage et tout est vérifié avant chaque vol. Un plan de vol est programmé dans l'ordinateur de vol des pilotes. Les données sont transférées sur un ruban magnétique de 8mm à la fin de la mission. La principale différence entre cet altimètre laser et celui utilisé pour récolter nos données est que le LVIS numérise tous les retours du laser, tandis que celui utilisé pour notre projet ne numérise que le premier et le dernier retour du laser.

Naesset [3] a essayé de calculer la hauteur moyenne des arbres d'une forêt. Il souligne que la connaissance de la hauteur moyenne des arbres d'une forêt est très importante pour l'administration adéquate des ressources forestières. Les techniques utilisées présentement (mesures sur le terrain et mesures de hauteurs par photogrammétrie) sont coûteuses et demandent beaucoup de temps. En utilisant le premier et le dernier retour de pulsation du laser, on peut estimer l'altitude du couvert végétal et du sol. En soustrayant les deux mesures, on obtient des points de hauteur des arbres directement.

L'objectif de son étude était de vérifier l'exactitude des hauteurs d'arbre déterminées à partir de données d'altimétrie laser. Il a démontré que la valeur brute moyenne estimée est souvent en dessous de la valeur réelle moyenne de hauteur de la forêt. Deux nouvelles approches pour déterminer la hauteur moyenne ont été présentées par Naesset.

Les forêt qu'il a étudiées se trouvent en Norvège dans les municipalités de Elverum (région1) et Grue (région 2). Un total de 36 sous-régions ont été choisies pour l'étude. Les arbres peuplant ces régions sont principalement des pins et des épinettes. La région 1 est dominée par les pins et la région 2, par les épinettes. L'âge des sous-régions varie entre 31 et 145 ans. Certains arbres ont été mesurés sur le terrain et choisis à l'aide de statistiques de grosseur du tronc. La valeur moyenne de hauteur pour les différentes régions a ensuite été calculée.

Le laser utilisé par Naesset est un Optech ALTM 1020 émettant à 1047 nm. Les données ont été acquises le 20 octobre 1995 à une altitude variant entre 640 m et 825 m au dessus du sol et à une vitesse approximative de 80 m/s. Plusieurs lignes de vol parallèles ont été suivies. La fréquence de pulsations était de 2 KHz avec une fréquence de balayage de 7 Hz. L'angle maximum de balayage était de 20° ce qui correspond, tenant compte de la hauteur de vol, à un régime de balayage de 460 à 600 m au sol. Le recouvrement entre les lignes de vol était de 70 à 425 m. L'empreinte du laser au sol est de 13 à 16 cm et la distance entre chacune est de 2,8 à 3,3 m. Un modèle numérique d'élévation a été produit avec ces données. De plus, avec un traitement, les données du sol et du couvert végétal ont été recueillies.

La valeur moyenne des données laser dans une sous-région a été comparée à la valeur moyenne calculée à partir des mesures prises sur le

terrain. Trois façons de calculer la hauteur moyenne, à partir des données laser, ont été utilisées. Premièrement, une moyenne arithmétique de tous les points laser (h_{h1}) a été calculée. Deuxièmement, une pondération de la moyenne des hauteurs des arbres par la hauteur des arbres individuels a été utilisée, les grands arbres ayant plus d'influence que les petits. Cette approche a été utilisée dans le calcul de la moyenne (h_{h2}). Naesset a aussi utilisé le carré des valeurs originales pondérées selon la grosseur de l'arbre pour calculer la hauteur moyenne (h_{h3}). Une autre méthode pour calculer la hauteur moyenne a été de diviser une région à l'aide d'une grille dont chaque cellule mesure 15X15 m. Seulement la valeur maximale dans chaque case de la grille a été retenue. Finalement, les points retenus dans chacune des cases de la grille ont été utilisés pour calculer une dernière moyenne. Cependant, le nombre de points utilisés dans les cases pour calculer chacune des moyennes a été utilisé pour pondérer le poids de chaque moyenne (h_{15x15}) dans la moyenne totale. La différence entre les hauteurs moyennes calculées à partir des données laser et de celles calculées à partir des mesures faites sur le terrain a été calculée (d_i). Les écart-type correspondants ont aussi été calculés.

La hauteur moyenne a été sous-estimée de 4,1 à 5,5 m avec h_{h1}. Avec h_{h2} et h_{h3} (des valeurs pondérées), la hauteur moyenne a été sous estimée de 2,1 à 3,6 m avec des déviations standard de 1,1 à 1,4 m. L'approche avec une grille a aussi donné de meilleurs résultats que h_{h1}. Dans ce cas, la hauteur moyenne différait de –0,4 à 1,9 m de la vraie valeur avec un écart-type de 1,2 à 1,3 m.

On peut constater que ces valeurs sont intéressantes et que l'utilisation de l'altimétrie laser semble justifiable, mais le but du présent projet est de calculer la hauteur des arbres individuels et non la hauteur moyenne d'un peuplement.

Flood et al. [4] abordent les applications commerciales de l'altimétrie laser. Les applications pratiques sont par exemple la topographie de terrain, la génération de modèles 3D urbains et l'étude du flot de glaciers et de leur hauteur. On peut se servir de l'altimétrie laser afin de calculer la densité des bâtiments d'une ville, suivre les catastrophes naturelles, délimiter les écozones et les ressources naturelles. La photographie aérienne ne permet pas toujours d'obtenir des résultats. Par exemple, en forêt lorsque la canopée est trop dense on ne peut percevoir le sol et on ne peut calculer exactement la hauteur des arbres. Dans ce cas, seule l'altitude de la canopée peut-être estimée et l'altimétrie laser peut être une bonne solution de rechange aux méthodes conventionnelles.

Nelson [5] tente de déterminer les caractéristiques de la canopée à l'aide de l'utilisation de l'altimétrie laser. Il a utilisé pour ce faire un laser et a survolé une forêt en Pensylvanie. Le laser a fourni des données de hauteur de la canopée et du sol. Les données ont été analysées afin de déterminer quelles caractéristiques du laser pouvaient expliquer la différence entre les données se recouvrant entre deux lignes de vol. Ils ont trouvé que la capacité du laser à pénétrer la canopée était la cause. Les pulsations du laser sont atténuées plus rapidement dans la végétation dense. De ce fait, ils ont conclu qu'une canopée dense indiquait la difficulté à obtenir des points laser pour le sol. Ils ont aussi obtenu 1 m de différence entre la hauteur moyenne mesurée par photogrammétrie comparativement à l'altimétrie laser. De ce fait, ils concluent que les modèles laser de canopée peuvent être utilisés pour la mesure d'arbres en forêt, puisque la différence est relativement faible.

Nelson et al. [6] ont utilisé un altimètre laser pour recueillir des données de hauteur de canopée dans une forêt de pins en Georgie. Ils ont essayé de prédire la biomasse de la forêt et le volume ligneux. Ils ont obtenu des résultats

à l'intérieur de 2,6 % de la valeur plancher pour la biomasse. Ils ont conclu que les profils laser peuvent être utilisés pour calculer la biomasse avec concision, mais que les variations d'un site à un autre sont considérables parce que les mesures ne comprennent pas le diamètre du tronc de l'arbre.

Nilsson [7] a aussi utilisé l'altimètre laser pour évaluer la hauteur des arbres et le volume ligneux d'une forêt. L'altimètre laser dans ce cas était installé à bord d'un hélicoptère et la forêt évaluée était une forêt de pins d'une hauteur moyenne de 12,5 m. Les résultats qui ont été obtenus étaient des hauteurs sous-estimées de 2,1 à 3,7 m.

Nelson et al. [8] ont utilisé le profil laser afin de calculer le volume et la biomasse d'une forêt tropicale. Comme ils ne traitent pas de mesure d'arbres individuels, nous n'énoncerons que leur conclusion. Ils affirment, suite à leur recherche, que l'altimétrie laser devrait être utilisée pour mesurer des forêts inaccessibles. Ce qui est aussi notre avis et la raison pour laquelle nous tenons à réussir à isoler et à mesurer les arbres individuels à partir de données d'altimétrie laser.

1.2 La connaissance des ressources forestières

St-Onge [9] mentionne l'importance de la connaissance de la moyenne de la hauteur des arbres d'une forêt pour l'administration adéquate des ressources forestières. Les techniques utilisées présentement (mesures sur le terrain et mesures de hauteurs par vision stéréoscopique) sont coûteuses et demandent beaucoup de temps. En utilisant l'altimétrie laser, on peut utiliser le premier et le dernier retour de pulsation du laser afin de connaître la hauteur du couvert

végétal et du sol. En soustrayant les deux mesures, on obtient des points de hauteur des arbres directement.

La hauteur des arbres est considérée comme la variable la plus importante, avec la densité de la forêt et le diamètre à hauteur de poitrine des arbres, dans l'estimation du volume de production de bois d'une forêt. Elle permet de déterminer aussi la quantité de lumière qui pénètre dans la forêt, variable utile à l'étude de certains habitats. La hauteur est cependant une des mesures les plus difficiles à obtenir lorsque le couvert forestier est dense. La prise de mesure au sol prend plusieurs minutes et entraîne souvent des erreurs. Une autre alternative consiste à utiliser la vision stéréoscopique, mais cette technique fonctionne bien seulement lorsqu'il y a des trouées dans le couvert végétal. Une alternative consiste à utiliser l'altimétrie laser, qui améliore considérablement les mesures de hauteur d'arbres. L'altimétrie laser permet des résolutions à l'intérieur d'un mètre. La haute densité de points permet même de reconnaître les couronnes d'arbre individuelles. L'objectif général de St-Onge était de vérifier l'utilité de combiner les données d'altimétrie laser avec des images multi-spectrales de haute résolution afin de calculer la hauteur des arbres pris individuellement.

St-Onge a utilisé les mêmes données que nous utiliserons dans ce projet (voir chapitre 2). Il a estimé la hauteur des arbres à partir du pixel (choisi à partir des données laser) ayant la plus grande valeur de hauteur et faisant partie de la couronne. Ce pixel est généralement situé au centre de la couronne ou près du centre selon qu'il s'agisse d'un feuillu ou d'un conifère.

Une régression linéaire a été calculée entre les points d'altitude du sol bruts et les points interpolés du sol. Grâce à cette régression linéaire, il a déterminé que le modèle de prédiction des hauteurs donne de bons résultats.

Ce travail fait suite aux travaux de St-Onge et l'amélioration du calcul des hauteurs des arbres individuels avec les données d'altimétrie laser sera explorée.

1.3 Traitement des images de télédétection ou d'altimétrie laser

Dans le domaine de la délimitation de couronnes d'arbres, les travaux effectués jusqu'à présent portent sur des images de forêt prises avec une caméra et non sur des images générées à partir d'interpolation de données laser 3D. Nous énumérerons quelques méthodes de délimitation des couronnes qui ont été appliquées à des images de forêt.

Gougeon [10] tente de délimiter des houppiers de façon complètement automatique. Les houppiers sont premièrement délimités en suivant les zones d'ombre qui sont éliminées par un seuillage. Ensuite les minimums locaux sont trouvés et les zones d'ombre sont suivies à partir d'un minimum jusqu'à un autre minimum. Cela permet de bien délimiter la majorité des houppiers de conifères. Cette approche n'est toutefois pas applicable à notre projet, car nos images d'altitude ne contiennent pas d'ombres causées par l'éclairage, elles proviennent de l'interpolation de données laser de hauteur.

Pollock [11] essaie lui aussi de faire la reconnaissance de houppiers à partir d'images aériennes. Sa procédure est basée sur le processus de formation des images à l'échelle des arbres individuels, sur des données d'entraînement produites par l'opérateur et sur l'exploitation de la relation spatiale entre les arbres avoisinants. Les résultats qu'il a obtenus contiennent des erreurs associées entre autre à l'éclairement dont découlent des zones d'ombre. Ce problème est beaucoup atténué avec la méthode d'altimétrie laser,

puisque l'angle de visée de l'appareil est de 10 à 15° au maximum et parce que les lignes de vol sont calculées de façon à ce qu'il y ait recouvrement. De cette façon, le points sont visés de deux angles différents, ce qui réduit l'ombrage.

Brandtberg et Walter [12] ont utilisé une approche multirésolution qui est assez intéressante et qui donne des résultats presque aussi bons que l'évaluation faite par un humain. Par contre ils travaillent eux aussi sur des images et non des données laser. Nous tenterons d'utiliser aussi une approche du genre multirésolution pour trouver le contour des arbres dans notre projet afin de faire ressortir les petits arbres aussi bien que les grands.

Les grandes différences entre les images utilisées dans les travaux présentés précédemment et nos images sont que nous n'avons que peu d'ombrages qui apparaissent sur l'image et que l'intensité des points (dans notre cas) indique la hauteur du point laser en mètres directement.

1.4 Modélisation et interpolation 3D

La prochaine section présente différentes techniques d'interpolation 3D trouvées dans la littérature. Les approches présentées sont trop étoffées pour être utilisées dans le présent projet telles quelles, mais méritent d'être présentées dans le sens qu'elles peuvent ammener des éléments de discussion et de comparaison.

Dickinson et al. [13] ont développé une approche de modélisation de la forme assez complexe. Le modèle qu'ils proposent introduit un objet hybride pour la représentation et la reconnaissance d'objets 3D à partir d'images 2D. Les objets sont construits à partir d'un groupe fini de parties volumétriques et

ces parties sont représentées comme une hiérarchie d'aspects. La forme de la surface de l'objet fait partie des aspects. De plus, dans leur modèle, ils utiliseront aussi un éventail de données. Leur hiérarchie-éventail d'aspects est composée de 3 niveaux incluant un ensemble d'aspects modélisant les volumes choisis, un ensemble de composants « surface » des aspects et un ensemble de contours entourant les surfaces. Un volume est constitué de plusieurs surfaces et chaque surface de plusieurs contours. De plus, ils disposent de l'information sur la façon dont les surfaces sont connectées pour former un volume. Dix volumes ont été utilisés pour démontrer l'approche, on retrouve le cône, l'ellipsoïde, le cube, etc. La façon ambiguë de faire le lien entre chaque niveau de la hiérarchie comprend un groupe de probabilités conditionnelles qui résulte d'une analyse statistique d'un ensemble d'images approximant les surfaces et les volumes recherchés. L'utilisation d'un éventail de grandeurs pour les volumes signifie qu'il y a une distinction importante à faire entre l'aspect original et l'aspect d'un objet d'une autre grandeur. Cela générera plus de modèles différents, diminuant du même coup l'ambiguïté du choix entre les modèles. De nouveaux objets peuvent aussi être ajoutés à la base de données d'objets. À cause de leur simplicité, les volumes ne peuvent avoir plus de 12 surfaces et chaque surface, plus de 12 contours.

Géométriquement, les modèles 3D utilisés dans cette étude sont très près des objets 3D.

À partir de leur image d'entrée, ils ont appliqué un algorithme de segmentation en régions. Les surfaces sont extraites et classiffiées en se basant sur les courbes. L'algorithme de segmentation classe les surfaces selon la dominance du type de surface. L'ordre et l'importance des surfaces dans la hiérarchie sont donc primordiales. Idéalement, il devrait y avoir une mesure de confiance associée à la classification.

Pour faire correspondre un objet réel avec un des volumes de la hiérarchie, ils effectuent une segmentation de l'image et un étiquettage de l'image originale en régions représentées par des surfaces. À partir de l'image étiquetée, on construit un graphe de topologie des régions, dans lequel les noeuds représentent les régions et les arcs représentent les frontières des régions. À partir du graphe, chaque région est caractérisée par la forme de ses contours. On fait simultanément la classification des contours résultants. Le résultat de ces opérations est un graphe de contours pour une région, où les noeuds représentent les contours et les arcs, les relations entre les contours. Quand la description de chaque région est complétée, on essaie d'associer les descriptions avec les faces dans la hiérarchie. Si on a une association, on donne une étiquette à la face, sinon on doit redescendre au niveau des contours. À partir des contours, on détermine un ensemble de surfaces possibles. Le résultat de ces opérations est un graphe des topologies de surfaces, où chaque noeud est associé à une étiquette de surface, associée à une région. Durant le processus de recherche, on effectue un assortiment des surfaces. Quand un ensemble de surfaces a été retrouvé, on utilise ces surfaces pour inférer un ou des volumes. Lorsqu'il est impossible d'identifier un volume, on peut faire bouger la caméra pour obtenir un autre point de vue sur l'objet et recommencer le processus.

Nous développerons une méthode plus simple que celle-ci pour effectuer le calcul de la hauteur de l'arbre. Le texte précédent montre une méthode possible pour apparier un volume tel un cône ou une ellipsoïde à un ensemble de données appartenant à un arbre (conifère ou feuillu).

Larsen [14] utilise une méthode d'appariement de gabarit pour identifier des couronnes d'arbres photographiés à un angle très aigu. Il a utilisé un modèle géométrique-optique pour produire une image type, un modèle. Le

modèle vu selon le même angle que l'angle de vue réel du terrain représente ce à quoi l'arbre devrait ressembler sur l'image réelle. Pour la création du modèle, il tient compte de la position du soleil et de la caméra. Ensuite il génère un modèle de sol pour tenir compte de la lumière réfléchie. Le modèle de l'arbre lui-même est très complexe, avec des variables pour créer les épines ou les feuilles selon plusieurs formules mathématiques. Ensuite une corrélation du modèle avec l'image est calculée. Aux endroits où il obtient des maximums, il prédit qu'il y a des arbres, plus le maximum est élevé, plus il est sûr qu'il y a un arbre. Parfois, plusieurs modèles d'arbres doivent être créés pour une bonne identification des arbres et cela demande beaucoup de temps de calcul. De plus, la corrélation est une technique longue à exécuter et la détermination de la grandeur de fenêtre optimale pour la corrélation est difficile à trouver.

Cette méthode est plus simple que la méthode présentée précédemment, mais nous opterons pour une méthode différente (nous ne produirons pas de modèle d'arbre aussi complexe), car nous ne travaillons pas sur des images d'illuminance et, de ce fait, les variations dans l'éclairage ne changent pas l'image obtenue. Aussi, puisque ce projet a une étendue plus grande que le seul calcul des hauteurs des arbres, nous ne voulons pas développer une méthode de calcul trop complexe.

À notre connaissance, encore aucune publication sur l'estimation des hauteurs d'arbres individuels n'a pu être recensée même si plusieurs auteurs évoquent le potentiel de l'altimétrie laser pour la mesure des arbres. Nous consacrons donc ce projet d'étude à ce sujet.

CHAPITRE 2
PROBLÉMATIQUE ET OBJECTIFS

2.1 Problématique spécifique

Ce projet se situe dans le cadre d'un plus grand projet visant à estimer le volume de bois d'une forêt. La hauteur des arbres individuels est une variable importante dans l'estimation de cette valeur. C'est pourquoi nous voulons mettre au point une méthode automatisée permettant de localiser le centre des couronnes des arbres à partir d'une image d'hauteurs et ensuite de calculer la hauteur de l'arbre correspondant.

Pour arriver à calculer la hauteur des arbres, nous disposons aussi de photographies aériennes multispectrales dans trois bandes de fréquence soient le rouge, l'infrarouge et le vert. Également, nous avons en notre possession des données d'altimètre laser, sous forme d'image, nous indiquant directement des hauteurs et nous disposons de mesures prises sur le terrain pour certains arbres afin de pouvoir vérifier que les hauteurs d'arbre calculées sont justes. À partir de ces données, nous devrons faire des opérations d'interpolation, de filtrage, de détection des contours, de localisation de l'emplacement des arbres, d'extraction des données spécifiques à chaque arbre et finalement de calcul de hauteur pour chacun des arbres. Par la suite, nous serons à même de vérifier si la méthode développée est valide en comparant avec les mesures prises sur le terrain.

2.2 Objectifs

L'objectif principal de ce projet est d'obtenir automatiquement des mesures de hauteur d'arbres aussi précises que les mesures prises sur le terrain, c'est-à-dire que l'on désire une précision autour du mètre. Pour arriver à cela, nous devrons arriver à atteindre les objectifs intermédiaires suivants :

- Identifier le contour des couronnes des arbres en séparant les couronnes entre elles.
- Identifier le centre des couronnes à partir des contours trouvés précédemment.
- Calculer la hauteur des arbres individuels à partir des données laser brutes, de la position de la couronne ainsi que de sa superficie.
- Obtenir une hauteur calculée plus près de la hauteur réelle de l'arbre que le pixel le plus haut de la couronne.
- Obtenir une hauteur d'arbre en dedans de 1 mètre de la hauteur réelle pour la majorité des arbres.

2.3 Méthodologie

La méthodologie adoptée comprendra les étapes suivantes :

- Prétraitement des données provenant de l'altimètre laser (interpolation)
- Insertion de ces données dans KBV
- Délimitation des couronnes par Laplacien de Gaussienne
- Sélection des couronnes d'arbre identifiées valides

- Extraction des données brutes 3D correspondant aux couronnes valides
- Régressions simples, polynômiales et surfaciques pour le calcul des hauteurs des arbres

CHAPITRE 3
RÉGION D'ÉTUDE ET DONNÉES

3.1 Région d'étude

La région d'étude est la Forêt d'Enseignement et de Recherche du Lac Duparquet (FERLD) située en forêt boréale mixte, localisée en Abitibi, au Québec, à environ 48° 30' N, 79° 20' O. C'est une région de 80 km^2 sous les 400 m d'altitude, la température moyenne y est de 0,8 °C et 64 jours par année il n'y a pas de gel au sol. Il tombe en moyenne 857 mm d'eau par année. Les arbres ont entre 50 et 230 ans, les principales espèces sont le peuplier faux-tremble (Populus tremuloides [Michx]), l'épinette blanche (Picea glauca [Moench], Voss.), le bouleau blanc (Betula papyrifera [Marsh.]), le sapin baumier (Abies balsamea L. [Mill]), le pin gris (Pinus banksiana [Lamb.]), le cèdre (Thuya occidentalis L.), et l'épinette noire (Picea mariana [Mill.] BSP). Le site d'étude était exploité commercialement jusqu'en 1992.

3.2 Données laser

Les données d'altimètre laser ont été acquises le 28 juin 1998 avec un appareil Optech ALTM 1020 construit en 1995 et installé sur un petit avion volant à une altitude de 700 m. Deux survols ont été effectués pour obtenir respectivement les données de sol et du couvert végétal. Les tableaux suivants résument les caractéristiques de l'acquisition des données laser.

Tableau 1 – Caractéristiques du vol

Date du survol	28 juin 1998
Avion	Piper Navajo
Altitude de vol pour la végétation et le sol	700 m
Vitesse de vol pour la végétation et le sol	65 m/s
Aire couverte	8 km^2
Largeur de la ligne de vol	0,25 km
Nombre de ligne de vol pour la végétation et le sol	10
Nombre de passages pour la végétation	2
Nombre de passages pour le sol	1
Temps de vol pour chaque passage	80 min

Tableau 2 – Caractéristiques de l'altimètre laser

Fréquence d'impulsion	4000 Hz
Fréquence de balayage	16 Hz
Énergie	140 microjoules
Longueur d'onde du laser	1047 nm
Diamètre de l'empreinte au sol	0,19 m
Mode de balayage	Zig-zag
Angle maximum de balayage	10 °
Précision en X, Y et Z approximative	+/- 15 cm
Densité moyenne de points pour la végétation	1 point par m^2
Densité moyenne de points pour le sol	1 point par 2,5 m^2
Séparation des points laser du sol vs de la végétation	ALTM de Optec

Le système de positionnement utilisé à bord de l'avion était un GPS de type «Trimble 4000 SSI kinematic» fonctionnant à une fréquence de 1 Hz. Celui

utilisé au sol était un «Trimble 4000 SSE » installé sur un point géodésique. Le système d'inertie (constitué d'un gyroscope électronique qui donne l'inclinaison de l'avion) était un «Litton».

Le modèle de hauteur de la forêt a été obtenu en soustrayant les altitudes interpolées du terrain des mesures d'altitude du couvert végétal interpolées. Un maillage de type TIN (triangular irregular network) a été utilisé pour l'interpolation, ensuite les points ont été transformés en une grille avec une résolution de 50 cm. L'interpolation par TIN suppose que l'altitude varie linéairement entre les points, ce qui n'est pas nécessairement la réalité. L'influence sur la mesure de hauteur est double. D'abord, la matrice des altitudes du sol peut comporter des erreurs provenant de l'interpolation. Des buttes peuvent être tronquées et des creux peuvent être comblés s'il ne se trouve pas de point laser au centre de ces formes, mais ces erreurs sont relativement minimes. Dans le cas des données d'altitude du couvert végétal, le problème est plus grave simplement parce que le relief de la canopée est beaucoup plus complexe. Les formes générales sont toutefois reconnaissables. Par ailleurs, même s'il est vrai que l'on peut améliorer l'interpolation du sol, il est très difficile de faire mieux pour la végétation. La méthode de prédiction de hauteurs que nous proposons devrait s'avérer une méthode d'interpolation plus appropriée.

3.3 Données vidéo

L'aire couverte par les données laser avait été survolée l'année précédente (le 27 septembre 1997) à l'aide d'une caméra vidéo super VHS fonctionnant en mode agrandissement. La bande vidéo a par la suite été convertie en images multispectrales d'une résolution de 50 cm. La croissance

des arbres entre septembre 1997 et juin 1998 est minimale à cette latitude, ce qui permet de comparer directement les données laser et les images multispectrales. Le tableau suivant résume les caractéristiques des données vidéo.

Tableau 3 – Caractéristiques des données vidéo

Date du survol	27 septembre 1997
Heure du survol	11h00 – 13h00
Élévation du soleil	38° +/- 1°
Altitude de vol	1890 m
Résolution spatiale	50 cm
Superficie utile	5,9 km^2

Tableau 4 – Caractéristiques des bandes spectrales

Bande verte	520 – 600 nm
Bande rouge	630 – 690 nm
Bande infrarouge	760 – 900 nm

L'image multispectrale a été rectifiée pour une meilleure superposition avec les données 3D. Un minimum de 5 points de contrôle ont été choisis pour chaque image de 740 X 540 pixels. Les points de contrôle ont été choisis visuellement en choisissant des petites couronnes facilement identifiables sur les deux images. Les points de contrôle ont été positionnés au centre des couronnes choisies. Les images multispectrales ont été rectifiées par convolution cubique avec un polynôme d'ordre 1 (St-Onge [9]). De plus, les arbres mesurés sur le terrain ont été identifiées sur ces images, grâce aux coordonnées fournies par le GPS.

3.4 Données terrain

La hauteur de plusieurs arbres a été mesurée au sol. Deux mesures ont été prises pour chaque arbre, à partir d'endroits différents d'au moins 90°, afin d'assurer l'indépendance des mesures. Les arbres pour lesquels les mesures différaient de plus de 15 % ou 3 m ont été éliminés. Les mesures ont été prises pour des épinettes blanches et des peupliers faux-tremble principalement. Sur les 200 arbres mesurés sur le terrain, 40 ont été identifiés sur les images multispectrales jusqu'à maintenant. On a aussi pris la position de l'arbre dans l'espace avec un GPS, l'erreur maximale du GPS étant de 3 m.

CHAPITRE 4

MÉTHODOLOGIE

4.1 Environnement logiciel de développement

Dans le cadre de ce projet d'application, il a été décidé que nous travaillerions avec l'environnement KBVision[2] fonctionnant sur une plate-forme UNIX système V roulant sur des SUN Ultra 1. Ce logiciel comprend une banque de tâches allant de la conversion de format d'images jusqu'aux filtres de toutes sortes, en passant par des tâches effectuant des calculs mathématiques sur des images. De plus, KBV permet un niveau d'abstraction supérieur au simple niveau d'illuminance de chaque pixel de l'image. L'élément symbolique permet de définir des structures plus complexe comme des lignes ou des contours de façon simple et compacte. KBV permet également l'intégration de tâches définies par l'utilisateur. La programmation de ces tâches se fait en langage C ou C++.

4.2 Pré-traitements

La première étape de ce projet, appellée pré-traitement, comprendra la conversion des formats de fichiers et l'insertion de ces données dans l'environnement KBV.

---

[2] KBVision version 3.2 Amerinex Applied Imaging, 1995.

4.2.1 Conversion des fichiers PC à UNIX

Les images qui ont été utilisées dans ce projet ont été produites à l'aide d'un PC et sauvegardées en format binaire. Elles proviennent toutes de l'interpolation des données brutes d'altimétrie laser, sauf dans le cas de l'image multispectrale. L'ordre des octets n'est pas le même dans les PC et sous UNIX. La première étape consistait donc à changer l'ordre des octets dans les fichier binaires contenant les données des images que nous avons utilisées dans ce projet. Pour ce faire, un programme en langage C a été conçu. Le programme, que l'on retrouve à l'annexe 1, lit chacun des nombres en point flottant (32 bits) d'un fichier et change l'ordre des quatre octets comme illustré ci-après.

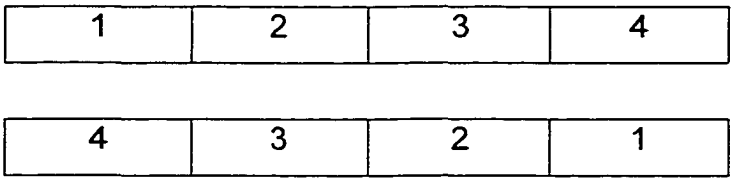


Figure 1 – Inversion de l'ordre des octets

4.2.2 Insertion des données dans KBV

Maintenant que les octets des fichiers binaires sont dans le bon ordre, nous pouvons les lire et les retransformer en images sous UNIX. Nous avons tout d'abord créé un répertoire qui contiendra toutes les images et les données servant à notre projet. Toutes ces images ont une grandeur de 1801 par 2181 pixels. Nous avons tout d'abord une image couleur multispectrale en format « TIF », puis trois images en niveaux de gris en format binaire. Les trois images en niveaux de gris sont premièrement une image représentant l'altitude du couvert végétal en mètres, deuxièmement une image représentant l'altitude du

sol en mètres et finalement une image de hauteur de la canopée résultant de la soustraction de l'altitude du sol de l'altitude du couvert végétal.

Pour importer les images de profondeur dans KBV, nous avons utilisé la tâche « ForImInt » (pour « Foreign Image Interactive ») qui permet de lire des données brutes, d'en spécifier le format et la grandeur puis de créér une image dans le format de KBV afin de pouvoir l'utiliser par la suite. L'image couleur étant en format « TIF », elle n'a pas eu à subir de transformations, KBV pouvant lire directement ce format d'images. La figure qui suit illustre les quatres images telles qu'affichées par le logiciel KBV. L'image multispectrale ne correspond qu'à une partie du territoire couvert par l'image de hauteurs. Si on superposait les rectangles encadrant les deux images (multispectrale et de hauteurs), les arbres se superposeraient parfaitement.

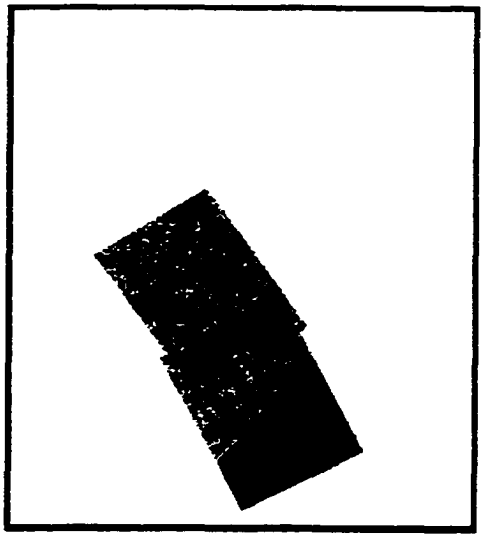


image multispectrale

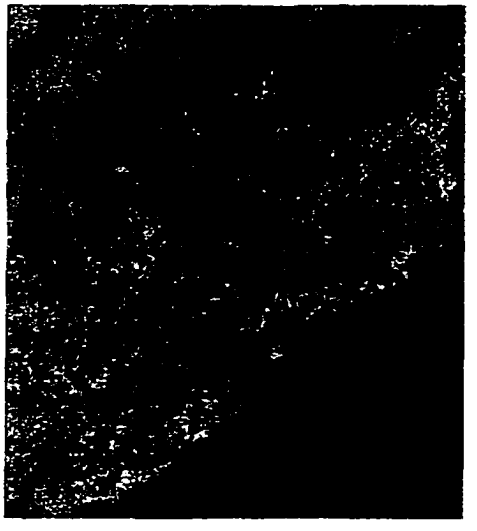
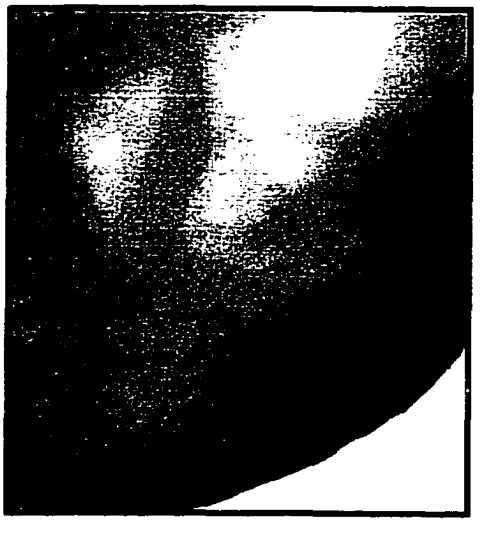


image de hauteurs



altitudes du sol

altitudes du couvert végétal

Figure 2 – Les quatre types d'images utilisées

4.2.3 Conversion du fichier des données laser brutes pour KBV

Nous avons effectué un prétraitement sur les données brutes (dont les coordonnées x, y et z étaient en format « ascii » dans un fichier texte) avant de les utiliser comme entrée de la tâche KBV qui génère les élément symboliques. Premièrement, nous avons soustrait de toutes les coordonnées (x,y) la valeur du point de géoréférence (dans notre cas le coin supérieur gauche de l'image) afin d'avoir des coordonnées référencées par rapport à un coin de l'image de coordonnées (0, 0), plutôt que par rapport à un point très loin de la région où nous travaillons. Cette valeur de référence est (623500, 5369109,5). Le programme effectuant cette conversion est listé à l'annexe 2.

4.2.4 Représentation des données brutes laser

Après avoir créé un fichier dans un format intermédiaire, nous avons programmé une tâche KBV « gentokens » permettant la génération des éléments symboliques pour représenter les données brutes laser. Nous avons déterminé une structure pour les élément symboliques détaillée ci-après.

X	Y	Delta X	Delta Y	Z

Figure 3 – Format d'un élément symbolique

- X et Y représentent les coordonnées du pixel contenant le point laser
- DeltaX et deltaY représentent le décalage du point laser dans le pixel par rapport au coin inférieur gauche du pixel
- Z représente la hauteur du point en mètres

Il faut rappeler ici que les pixels correspondent à un carré de 50 cm par 50 cm sur le terrain et que les coordonnées des données laser ont une résolution de 1 mm (avec une incertitude d'un peu plus d'une vingtaine de centimètres en z). La tâche KBV créant les éléments symboliques apparaît à l'annexe 3.

Lors de l'exécution de cette tâche, nous nous sommes aperçu d'un problème de KBV, en effet, nous travaillions sur un volume trop grand de données et KBV ne pouvait toutes les traiter. Nous avons donc dû scinder les données en deux parties, les traiter séparément et ensuite assembler les deux fichiers d'éléments symboliques (un fichier d'éléments symboliques « tokenset » est la structure contenant les élément symboliques) produits. Pour rassembler les deux fichiers d'éléments symboliques générés nous avons utilisé la tâche « AppendTks », qui, à partir d'un fichier d'éléments symboliques, ajoute le contenu d'un deuxième fichier à sa suite. Finalement, nous avons affiché le contenu du fichier d'éléments symboliques résultant. La figure 4 nous montre une grande densité de points laser représentés pas des petits « x ». Cette image sert à montrer le mouvement du faisceau laser au sol. De plus, cette image nous montre le recouvrement de lignes de vol (où il y a une densité de points plus grande). Il faut noter que les lignes du faisceau laser sont perpendiculaires aux lignes de vol de l'avion. La figure 5 montre un agrandissement d'une partie de la figure 4 et on y voit la distribution des points au sol avec plus de détails.

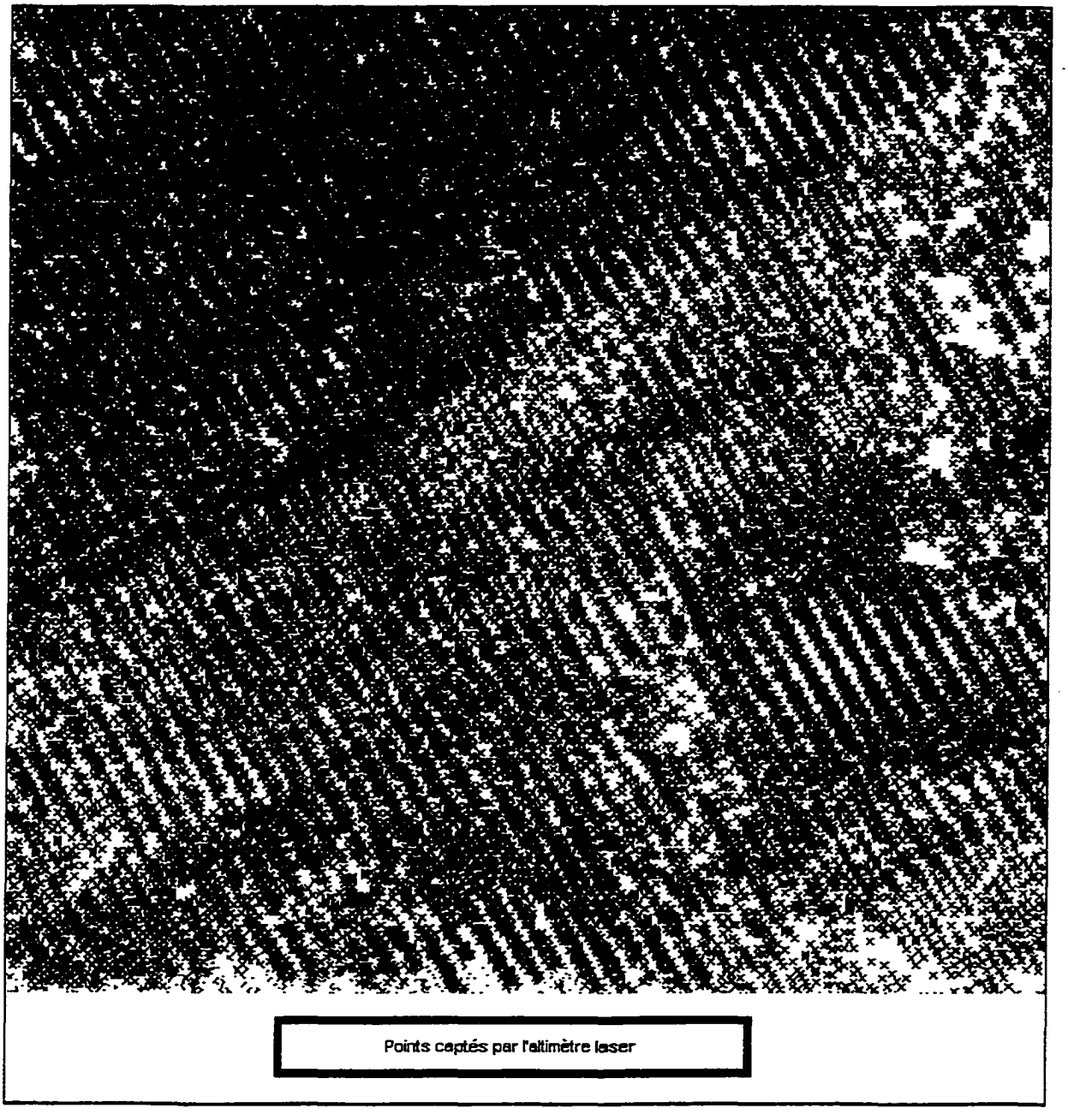


Figure 4 – Une partie des données laser brutes

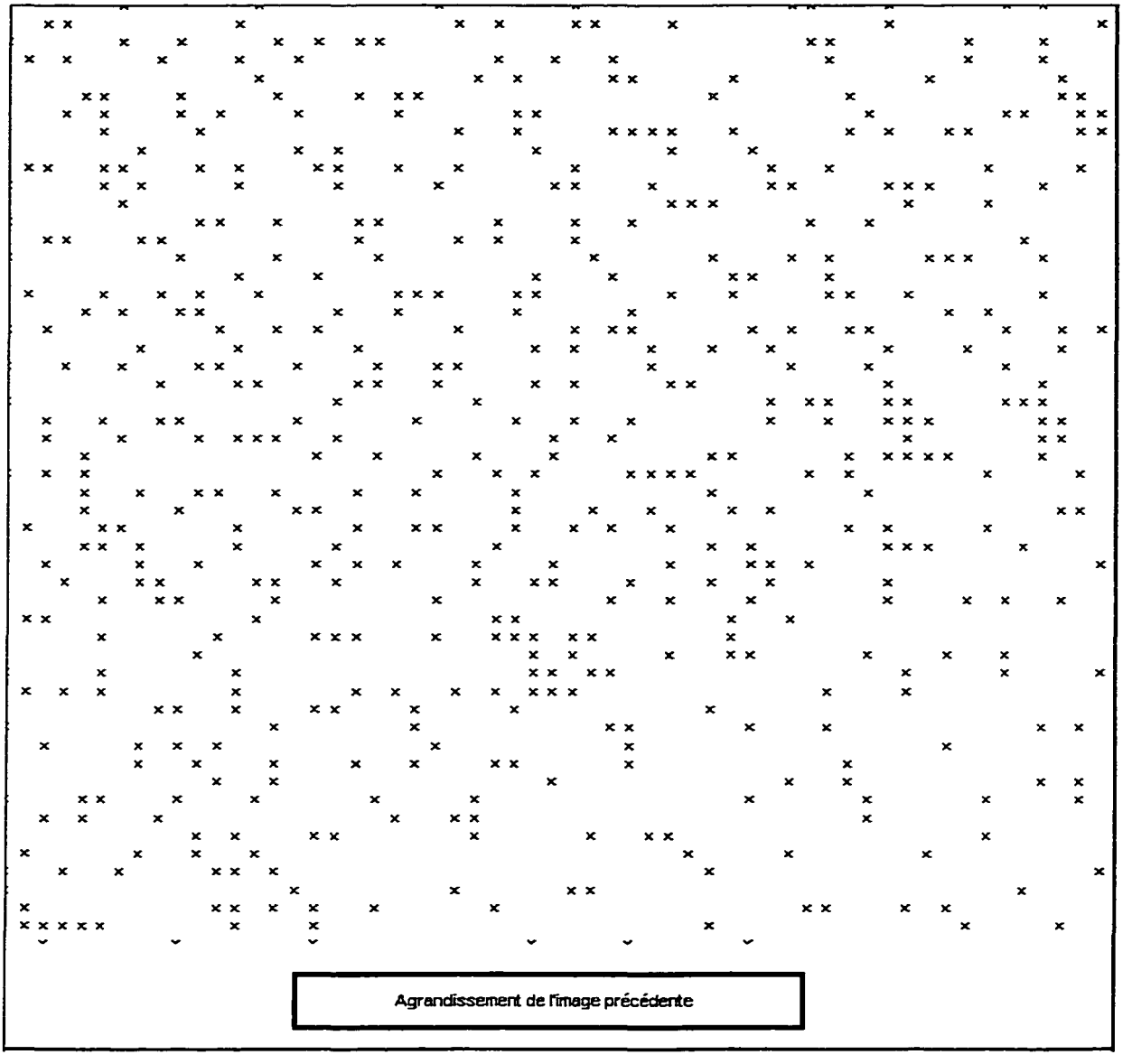


Figure 5 – Données laser brutes (agrandissement)

On peut remarquer, à la figure 4, que les points sont alignés sur une grille (correspondant à la résolution des pixels de l'image multispectrale), mais ce n'est que pour l'affichage. Dans la structure interne de l'élément symbolique, il y a un DeltaX et un DeltaY qui positionnent le point laser exactement dans le

pixel. Comme KBV affiche toujours sous forme d'images, on ne peut afficher des éléments symboliques représentés par des « x » que sur les coordonnées x et y de l'image.

4.3 Délimitation des couronnes

Cette étape du projet consiste à identifier les contours des arbres, afin de les séparer les uns des autres et de connaître l'emplacement de chacun. Cela servira dans les étapes subséquentes à extraire les données correspondant à chacun des arbres. Ces données pourront ensuite être utilisées pour le calcul de hauteurs proprement dites.

La méthode que nous utiliserons afin d'extraire les contours des arbres sera basée sur un filtre que nous choisirons selon les résultats expérimentaux. Le filtre utilisé en sera un d'extraction de contours et non un filtre d'atténuation du bruit ou de rehaussement d'image.

L'image suivante illustre la hauteur des arbres pour la région d'étude. Neuf sous-régions ont ensuite été extraites de cette grande image pour comparer entre elles différentes façons de déterminer les contours des arbres.

Figure 6 – Hauteurs des arbres pour toute la région d'étude

4.3.1 Les filtres

Nous donnons dans cette section une définition des différents filtres qui sont testés.

4.3.1.1 Filtre Laplacien de Gaussienne (LoG)

Un contour peut être défini comme une frontière, un lieu de variation. On peut donc localiser un contour par la recherche, dans un voisinage restreint, du maximum de la dérivée première ou rechercher le passage par zéro de la dérivée seconde. Dans le cas d'une image, il n'existe pas une dérivée seconde unique mais quatre dérivées partielles (selon *x*, y, *xy* et *yx*). En pratique, on a recours à l'opérateur Laplacien qui fait la somme des deux dérivées partielles principales de l'image I(i,j) :

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \qquad (1)$$

Afin de limiter les réponses dues au bruit de l'image I, cette dernière est préalablement filtrée par un filtre Gaussien G(x,y) :

$$G(x,y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{r^2}{2\sigma^2}} \qquad (2)$$

où

$$r = \sqrt{x^2 + y^2} \qquad (3)$$

Puisque le LoG bidimensionnel est un filtre à symétrie circulaire, la variable indépendante se réduit au rayon r.

En combinant les deux opérations linéaires, le filtre Gaussien et le Laplacien, on obtient le LoG :

$$\nabla^2(I*G) = (\nabla^2 I)*G = I*(\nabla^2 G) \qquad (4)$$

Les opérateurs de filtrage et de dérivation se font donc en une seule étape de calcul avec la formule suivante :

$$\nabla^2 G_\sigma(r) = \frac{-1}{2\pi\sigma^2}\left(2-\frac{r^2}{\sigma^2}\right)\exp^{\frac{r^2}{2\sigma^2}} \qquad (5)$$

Cet opérateur est communément appelé le chapeau mexicain à cause de sa forme (voir figure suivante). La largeur de la portion centrale du LoG est donnée par w :

$$w = 2\sqrt{2}\sigma \qquad (6)$$

Le Laplacien de Gaussienne a été conçu par Marr et Hildreth [15] et sert à la détection des contours. Le filtre LoG génère des contours fermés. Cette caractéristique était recherchée dans le cadre de notre projet.

Voici maintenant une description du fonctionnement du filtre LoG. Premièrement, le filtre Gaussien lisse l'image et en réduit le bruit. Les petites structures ainsi que les points générés par le bruit vont être éliminés par ce premier filtrage. Le Laplacien est ensuite effectué et est utilisé comme approximation de la dérivée seconde en deux dimensions. Les pixels ayant un gradient local maximum (passage par zéro) seront considérés comme faisant partie d'un contour. Le LoG est illustré à la figure suivante.

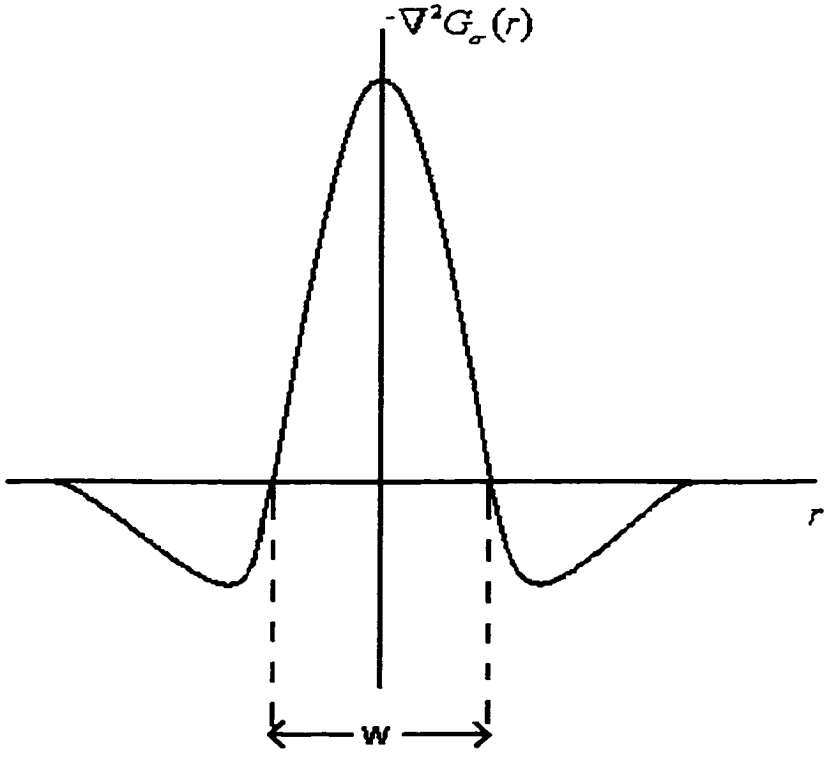


Figure 7 – Laplacien de Gaussienne

La variable w sera utilisée plus loin, elle représente la largeur de la partie centrale du Laplacien de Gaussienne.

La figure suivante illustre la réponse du Laplacien de Gaussienne lorsqu'appliqué à un saut brusque de I.

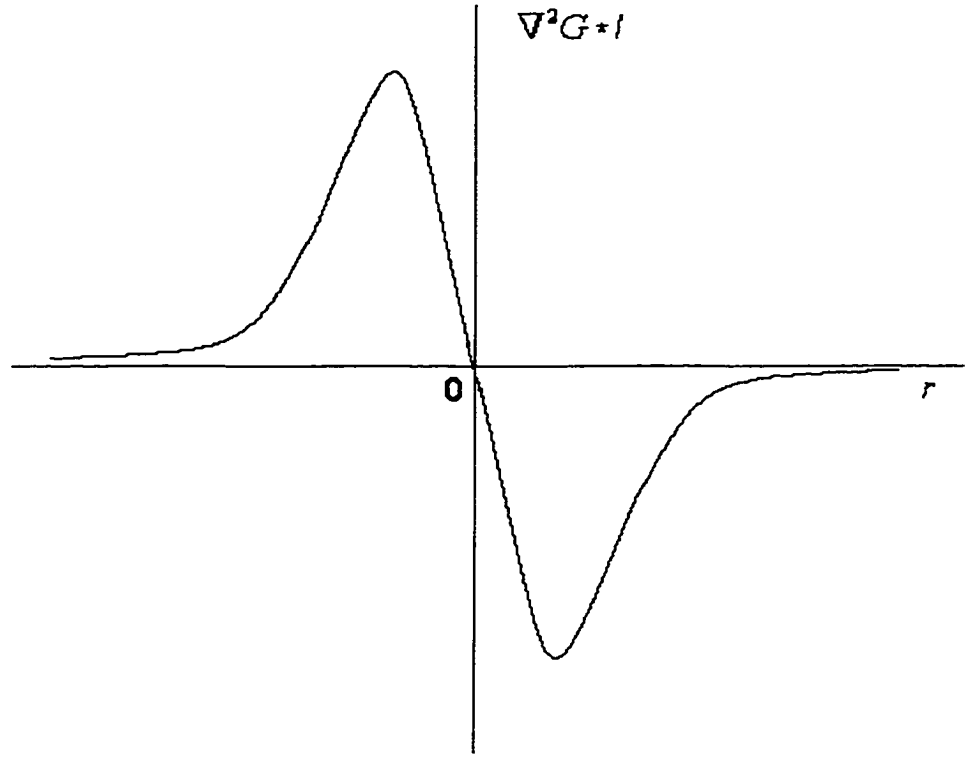


Figure 8 – Réponse au Laplacien de Gaussienne pour un échelon

Pratiquement, il y a deux façons de réaliser le LOG :

1- Calculer un masque de convolution à partir de la formule énoncée précédemment (équation 5)
2- Utiliser la Différence de Gaussiennes (qui est beaucoup plus rapide)

Voici un exemple de masque de convolution effectuant le LoG :

Tableau 5 – Masque Laplacien de Gaussienne 5x5

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

4.3.1.2 Filtre Différence de Gaussienne (DoG)

Le filtre Différence de Gaussiennes est une excellente approximation du LoG et est presque toujours utilisé pour l'implantation de ce filtre dans un logiciel parce qu'il s'exécute plus rapidement.

La différence de Gaussiennes est la différence entre deux résultats intermédiaires obtenus avec des filtres gaussiens passe-bas. Pour ces filtres, le noyau de convolution est gaussien et voici leur équations :

$$g_1(r) = \exp^{\left[\frac{r^2}{2\sigma_1^2}\right]} \qquad (7)$$

$$g_2(r) = \exp^{\left[\frac{r^2}{2\sigma_2^2}\right]} \qquad (8)$$

$$\text{où} \quad \sigma_1 > \sigma_2 \qquad (9)$$

Pour approximer le LoG avec le DoG, on évaluera les σ de la façon suivante :

$$\sigma = \sigma_1 \quad \sigma_1 = \gamma\sigma_2 \tag{10}$$

avec

$$\gamma = 1{,}6 \tag{11}$$

Marr et Hildreth ont déterminé expérimentalement, que la valeur optimale pour γ était 1,6.

Voici maintenant une image illustrant le fonctionnement du filtre DoG.

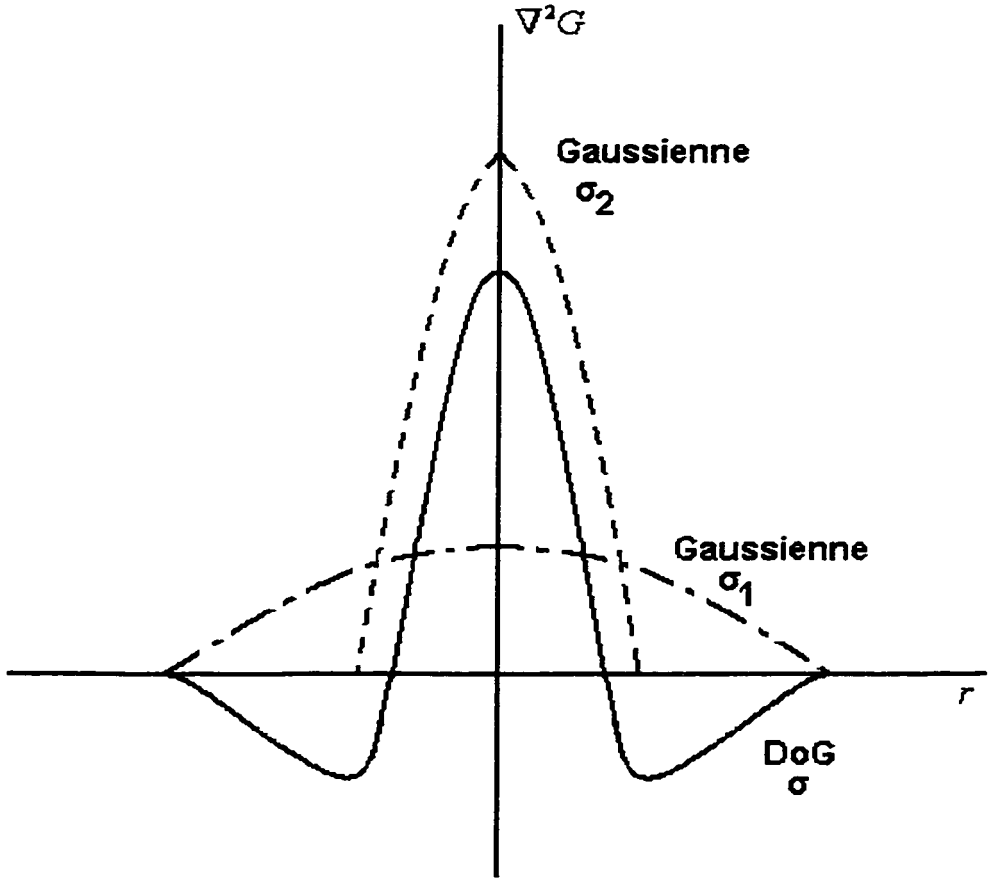


Figure 9 – Principe de fonctionnement du filtre DoG

Sur l'image suivante, on peut voir l'effet du changement de largeur (w) du filtre LoG (mis en œuvre avec le DoG) sur une sous-région d'intérêt de notre région d'étude. Plus w est grand, plus on détecte des formes grossières et arrondies. Pour le projet présenté ici, la tâche « ZeroXEdg » de KBV a été utilisée pour effectuer le filtre LoG. De plus, une opération d'ouverture morphologique a été effectuée sur le résultat du LoG afin de séparer les arbres liés par un lien mince (voir section 4.3.1.4 pour une explication du filtrage morphologique).

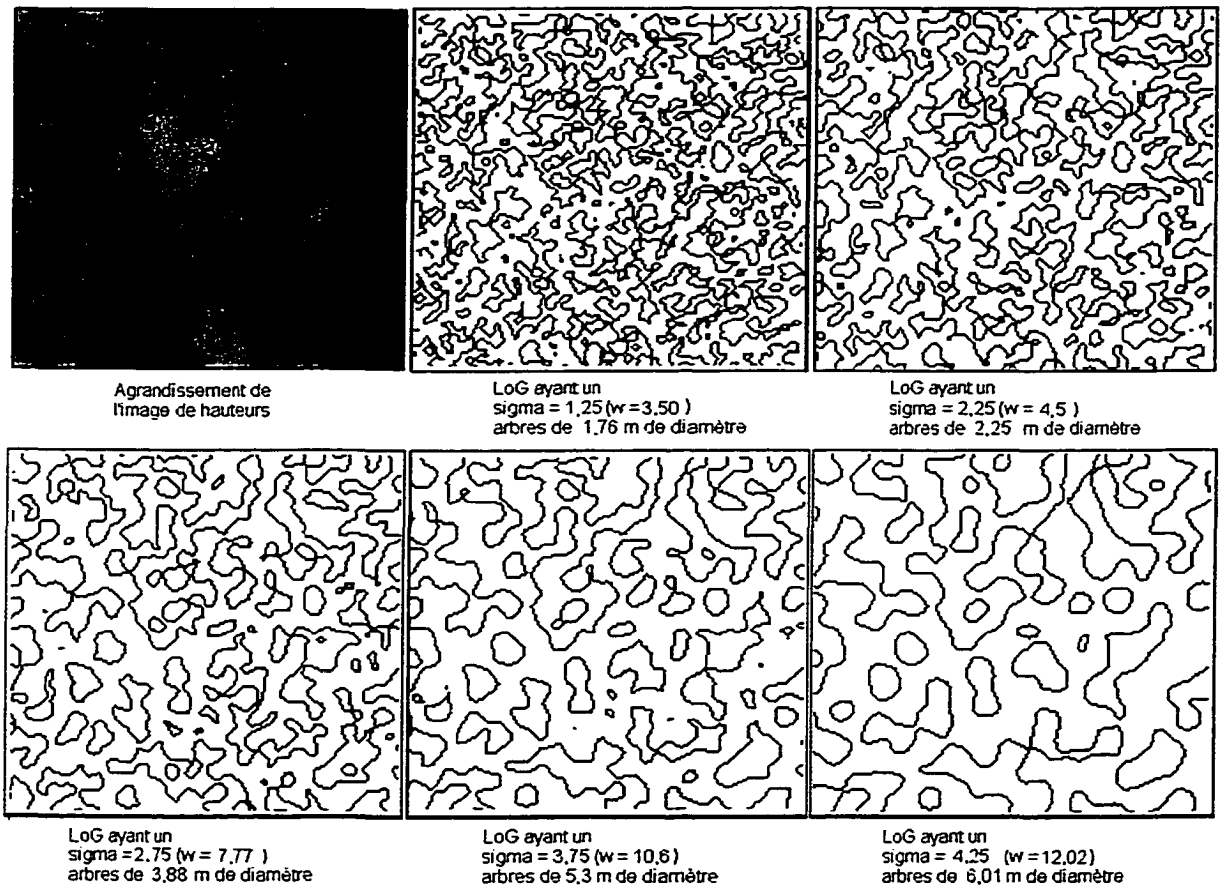


Figure 10 – Différentes grandeurs de DoG

Les résultats obtenus avec la Différence de Gaussiennes sont très prometteurs. Toutefois nous allons essayer différents filtres afin de comparer les précédents résultats avec d'autres résultats afin de vérifier si nous utilisons effectivement un filtre adéquat.

Ainsi, pour des fins de comparaison nous utiliserons un seuillage seul, un seuillage suivi d'un filtrage morphologique, un filtrage morphologique suivi d'un LoG et finalement le filtre de Canny. Les prochaines sous-sections expliquent les autres méthodes de filtrage qui ont été testées.

4.3.1.3 Seuillage uniquement

Puisque nous avons soustrait l'altitude du sol de celle du couvert végétal, nous pouvons penser qu'un seuillage ne laisserait que les couronnes des arbres apparaître sur l'image résultante. De plus, nous assumons que les arbres de moins de 5 mètres de hauteur ne sont pas d'intérêt. Notre seuil se situe donc à 5 mètres. Aussi, nous pouvons faire l'hypothèse que les couronnes des arbres intéressants se terminent au dessus de 5 mètres (on ne voit que la partie supérieure de la couronne sur les images d'altimétrie laser). Un exemple est donné à la figure suivante.

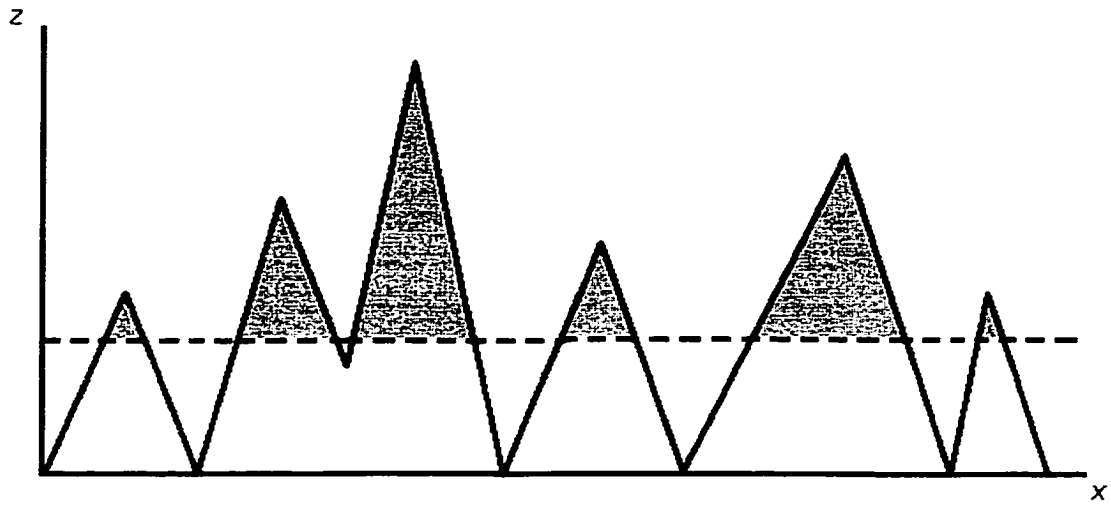


Figure 11 – Seuillage à 5 mètres

4.3.1.4 Seuillage suivi d'un filtrage morphologique

Un seuillage à 5 mètres a été appliqué sur l'image de hauteurs, suivi d'un filtre morphologique. Ces opérations vont servir de base de comparaison avec l'opération effectuée à la section précédente. Nous devrions noter une nette amélioration entre ces deux résultats, puisque le filtrage morphologique éliminera les petites structures pouvant rester après le seuillage.

Le filtre morphologique est tiré d'une approche de morphologie mathématique [17]. Il est non-linéaire et très robuste au bruit. Son principe consiste à comparer le contenu (inconnu) de l'image à traiter avec une forme géométrique dont on connait toutes les caracteristiques géometriques : l'élément structurant. Les opérations de base pour réaliser ce filtre sont la dilatation et l'érosion. L'addition de ces dernières permettent d'obtenir des

opérations d'ouverture et de fermeture, qui combinées permettent de construire des filtres morphologiques, efficaces pour l'élimination de petits détails.

Voici la définition des opérateurs de base de la morphologie mathémaique :

érosion : $$A\Theta B = \left\{x \middle| (B)_x \subseteq A\right\} \quad (12)$$

dilatation : $$A \oplus B = \left\{x \middle| (\hat{B})_x \cap A \neq 0\right\} \quad (13)$$

ouverture : $$A \circ B = (A\Theta B) \oplus B \quad (14)$$

fermeture : $$A \bullet B = (A \oplus B)\Theta B \quad (15)$$

où A est l'image et B l'élément structurant.

L'érosion a pour effet d'enlever une certaine épaisseur aux formes dans l'image, tandis que la dilatation ajoute une épaisseur. L'opération d'ouverture sépare les formes reliées par un petit segment et la fermeture rassemble les formes rapprochées.

Le filtre morphologique utilisé ici est constitué de l'ouverture suivie de la fermeture. Le contraire aurait aussi pu être utilisé.

4.3.1.5 Filtrage morphologique suivi d'un LoG

Nous avons effectué une fermeture suivie d'une ouverture, nous avons ensuite appliqué la tâche « ZeroXEdg » de KBV sur ce résultat. L'idée derrière cela est d'atténuer les petites zones d'ombre et de fort éclairage et d'ensuite effectuer le LoG sur le résultat, ce qui devrait donner des contours plus arrondis.

4.3.1.6 Filtre de Canny

Finalement, nous testerons l'efficacité du filtre de Canny, qui a été conçu dans le but de faire une bonne détection des contours tout en ayant une bonne localisation et en donnant une forte réponse.

Le filtre de Canny détecte les contours en effectuant premièrement un filtrage par convolution de l'image avec une gaussienne.

$$G(r) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{r^2}{2\sigma^2}} \tag{16}$$

Ensuite, une différenciation selon x et y est effectuée avec un filtre du type filtre de Robert, de Sobel ou de Prewitt par exemple, afin de faire ressortir de l'image des endroits où le résultat de la dérivée première est prononcé. Voici des exemples de filtres pouvant être utilisés à cette étape.

1	-1
-1	1

et

-1	1
1	-1

ou

1	0
0	-1

et

0	1
-1	0

Figure 12 – Filtres de dérivée première

L'étape suivante consiste en la suppression des non-maximas. On définit un maxima par un sommet dans la fonction gradient ou lorsque la dérivée du gradient est égale à 0. Dans cette étape on supprime les arêtes étant des non-

maximas perpendiculaires à la direction de l'arête afin de ne pas conserver des arêtes ne faisant pas partie du contour.

Finalement un seuillage par hystérésis est effectué afin de ne conserver que les arêtes faisant partie du contour. Pour effectuer un seuillage par hystérésis, on définit deux seuils, un seuil bas et un seuil haut. Si la valeur du pixel est plus grande que celle du seuil haut, le pixel est automatiquement conservé. Si la valeur du pixel est plus petite que le seuil bas, il est automatiquement rejeté. Si elle est comprise entre les deux valeurs de seuil, le pixel sera conservé seulement s'il est connecté à un autre pixel ayant une grande valeur (pixel conservé).

Le filtre de Canny est influencé principalement par trois paramètres : la grandeur du noyau de convolution (gaussienne), le seuil bas et le seuil haut. Si on augmente la grandeur du noyau de convolution, alors on réduit la sensibilité au bruit, mais on perd les détails fins et la localisation des contours devient moins bonne. Si le seuil bas est trop haut, des contours à faible contraste vont être perdus et si le seuil haut est trop bas, on obtiendra des contours douteux à la sortie. Il n'est donc pas facile d'obtenir des paramètres idéaux pour plusieurs images différentes avec ce filtre.

4.3.2 Comparaison des filtres

Pour les fins de comparaison entre les cinq méthodes, neuf sous-images de la région d'étude ont été extraites. La figure 13 les présente et les arbres dont nous connaissons les mensurations (prises sur le terrain) sont identifiés par un cercle et un numéro lorsqu'il y a plusieurs arbres sur une même image.

image 1
arbre 5 (785,505)
référence (747,460)

image 2
arbre 6 (812,618)
référence (760,560)

image 3
arbre 3 (953,595)
arbre 4 (902,546)
arbre 30 (922,544)
arbre 31 (921,558)
arbre 32 (938,603)
arbre 34 (980,584)
arbre 35 (971,574)
référence (880,510)

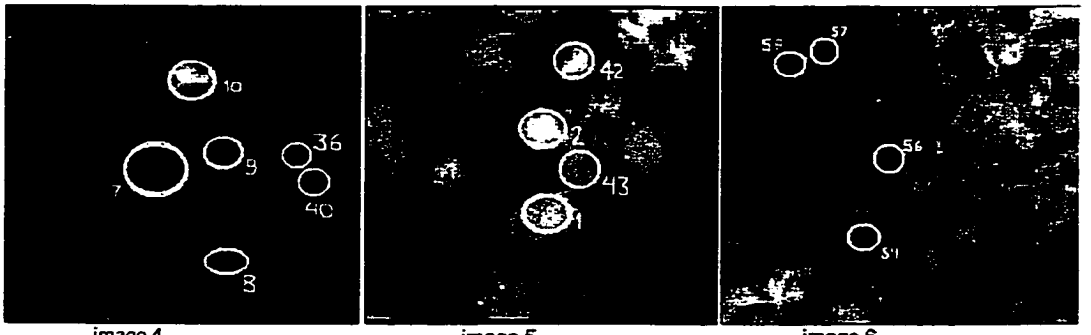


image 4
arbre 7 (875,681)
arbre 8 (891,653)
arbre 9 (893,685)
arbre 10 (883,713)
arbre 36 (911,691)
arbre 40 (916,676)
référence (830,635)

image 5
arbre 1 (1010,600)
arbre 2 (1010,629)
arbre 42 (1018,650)
arbre 43 (1016,619)
référence (960,570)

image 6
arbre 54 (717,1178)
arbre 56 (718,1207)
arbre 57 (709,1234)
arbre 58 (701,1233)
référence (680,1250)

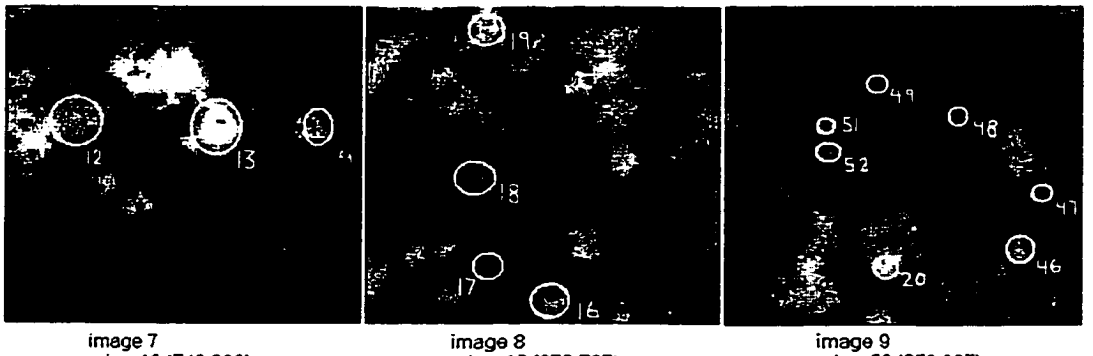


image 7
arbre 12 (749,862)
arbre 13 (789,859)
arbre 14 (818,863)
référence (738,800)

image 8
arbre 16 (672,787)
arbre 17 (655,801)
arbre 18 (653,830)
arbre 19 (653,875)
référence (620,783)

image 9
arbre 20 (656,897)
arbre 46 (693,902)
arbre 47 (698,918)
arbre 48 (676,944)
arbre 49 (650,953)
arbre 51 (639,941)
arbre 52 (640,931)
référence (610,878)

Figure 13 – Neuf sous-images avec arbres identifiés

Avant de comparer les cinq types de filtrage sur chacune des neuf sous-images, nous présenterons, à la figure 14, un schéma de l'enchaînement des tâches utilisées dans KBV pour chacun des types de filtres.

On pourra constater que le graphe est séparé en cinq parties distinctes. Chacune de ces colonnes représente un des types de filtrage présenté précédemment et produira un ensemble d'éléments symboliques différent. Le premier filtre effectue le Laplacien de Gaussienne suivi d'une croissance de région, d'un seuillage à 5 mètres et d'une ouverture morphologique. L'ouverture morphologique est effectuée afin de séparer les formes reliées ensemble par un lien mince. Un élément structurant de grandeur 3 par 3 carré (toutes les opérations morphologiques sont effectuées avec le même élément structurant) est utilisé. Toutes ces opérations sont suivies de la génération des éléments symboliques. Le deuxième filtre ne fait qu'un seuil, suivi d'une génération des élément symboliques. La troisième partie est constituée d'un seuillage à 5 mètres, suivi d'un filtrage morphologique (ouverture et fermeture) ainsi que de la génération des élément symboliques. La quatrième partie fait un filtrage morphologique (fermeture et ouverture) suivi de l'application d'un Laplacien de Gaussienne, d'un seuillage à 5 mètres et de la génération des élément symboliques. Les deux derniers filtres sont utilisés pour vérifier l'effet d'un filtrage morphologique sur notre image. Finalement, la cinquième partie ne contient que la tâche effectuant le filtre de canny (qui génère des élément symboliques en sortie) avec $\sigma = 3{,}18$ (largeur de 9 par 9 pour le filtre gaussien) pour le filtre gaussien, 1 comme seuil bas et 2 comme seuil haut.

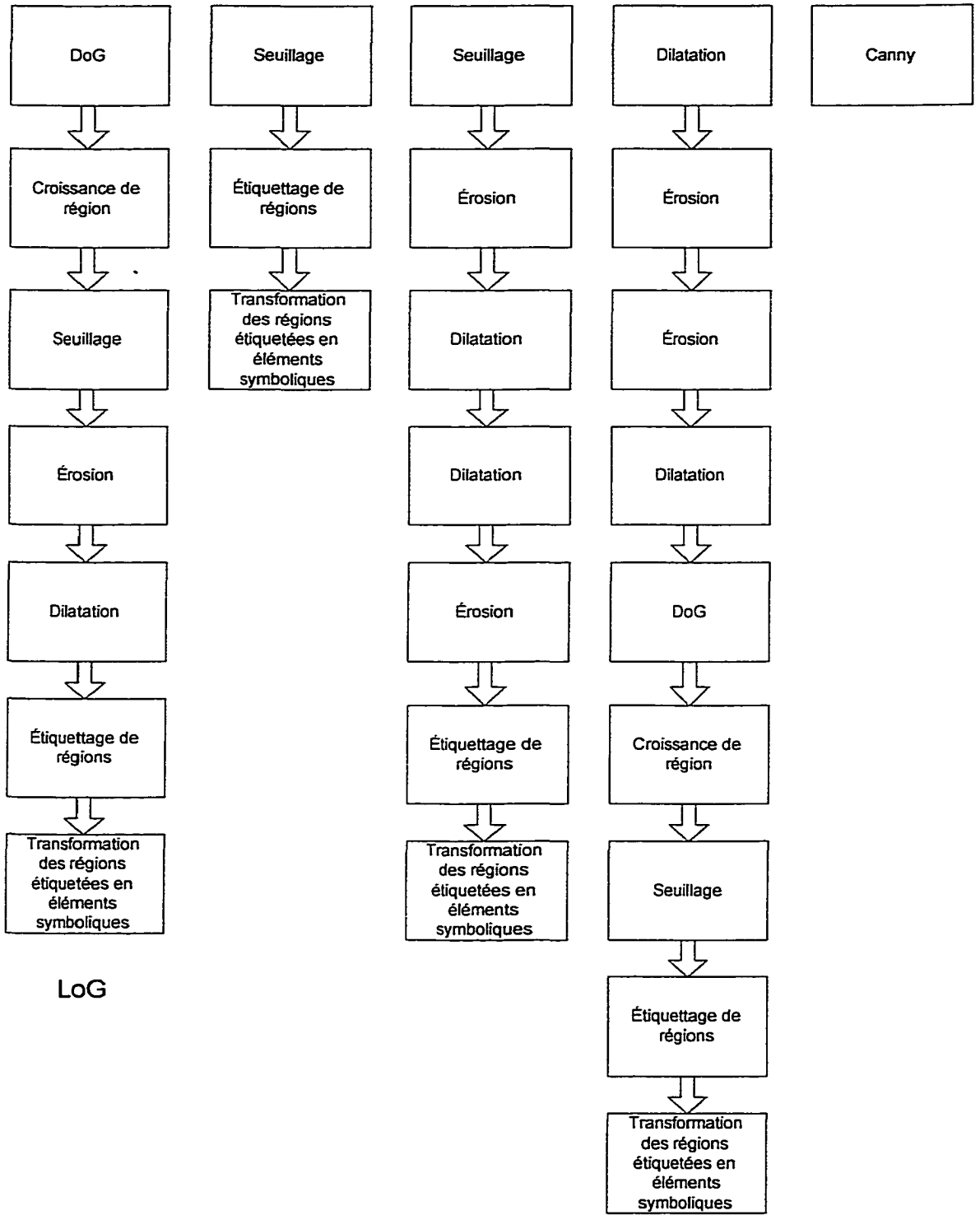


Figure 14 – Graphes de formation des élément symboliques

Voyons finalement ce que donnent les cinq types de filtrage présentés précédemment, sur une des neuf sous-images contenant des arbres mesurés sur le terrain. Les autres images sont présentées à l'annexe 4.

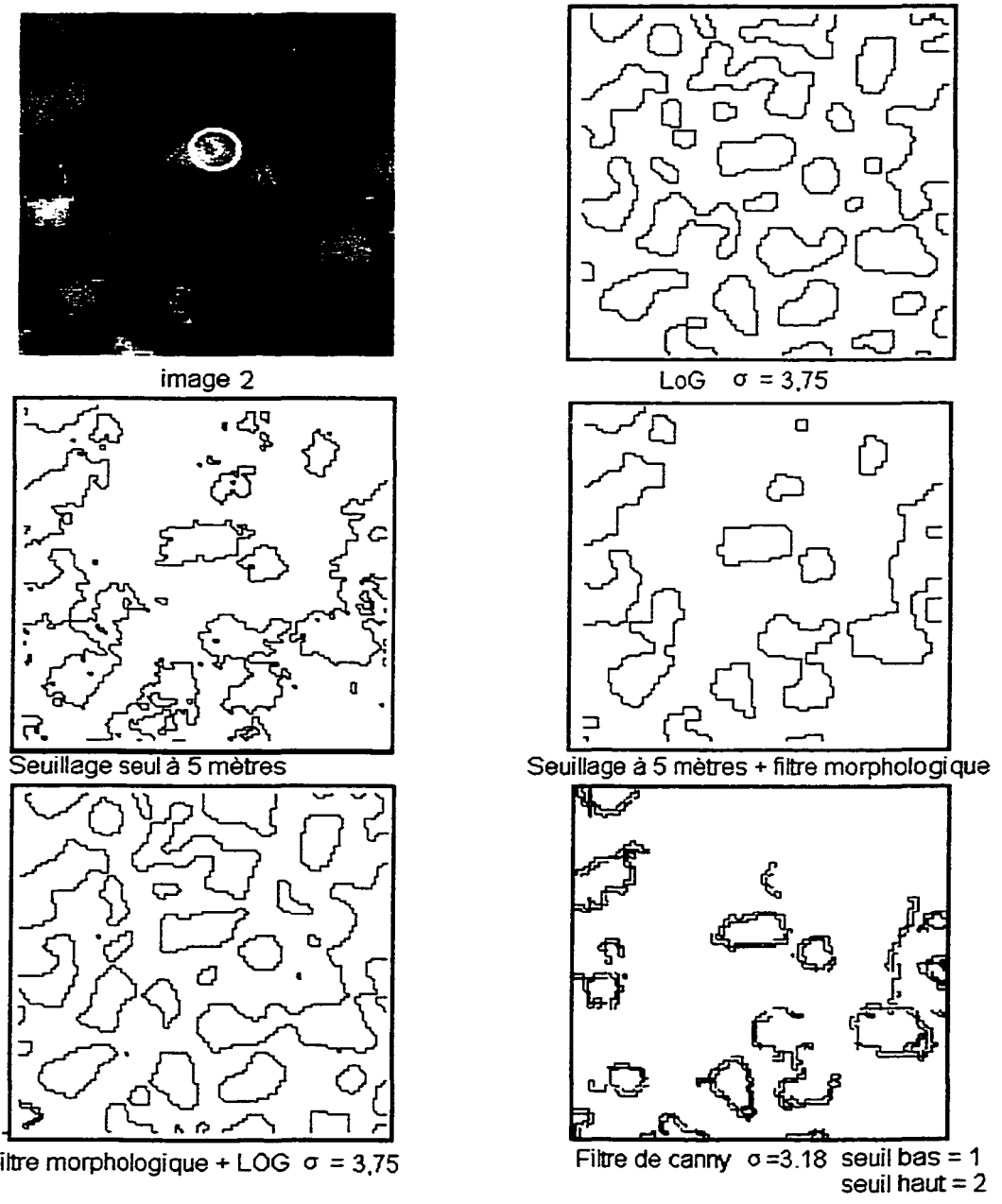


Figure 15 – Une des sous-image traitées avec les cinq types de filtrage

4.3.3 Choix d'un filtre

D'après ce que nous pouvons constater à partir de l'image 15 (voir aussi les images en annexe), le Laplacien de Gaussienne donne les meilleurs résultats. Avec un σ = 3,75, on vise surtout les arbres d'un peu plus de 5 mètres de diamètre. À plus petite échelle, nous obtenons trop de détails superflus. Les autres filtres donnent des résultats moins intéressants dans le sens où certains laissent trop de petites structures, ne donnent pas des contours fermés ou donnent des contours s'éloignant beaucoup du vrai contour des couronnes d'arbres.

4.3.4 Approche du filtre LoG « pyramidal »

Il n'est pas possible ici d'utiliser un filtre pyramidal conventionnel, car le fait de brouiller les données diminuerait grandement notre capacité à en extraire les contours. En effet, dans un filtre pyramidal conventionnel, à chaque niveau supérieur de la pyramide, la résolution de l'image est divisée par quatre et, par conséquent, la détection des contours des arbres n'en deviendrait que plus difficile, les contours étant de moins en moins bien définis. Par contre, nous pouvons utiliser plusieurs grandeurs de sigma du filtre LoG afin de simuler un filtre pyramidal. En effet, lorsque nous augmentons la valeur du sigma, cela revient à changer l'échelle à laquelle nous déterminons les contours, un peu comme dans un filtre pyramidal. De cette façon, avec un petit sigma, nous extrayons des détails plus fins et avec un grand sigma, les détails plus grossiers. Dans notre approche, nous utiliserons deux valeurs de sigma pour le Laplacien de Gaussiennes, afin d'extraire deux niveaux de détails et d'obtenir de meilleurs résultats globaux. Le premier LoG aura un σ = 2,25 et le deuxième LoG un σ = 3,75. De cette façon nous travaillerons avec deux ensembles de

données distincts sur lesquels nous effectuerons des opérations afin de définir la position des arbres, qui sera utilisée par la suite.

Pour continuer ce chapitre, nous devons maintenant introduire la notion d'éléments symboliques de KBV. Ces éléments symboliques sont au coeur des développements de ce projet.

4.4 Les éléments symboliques

4.4.1 KBV et les éléments symboliques

Un élément symbolique identifie une caractéristique d'une image ou d'une partie d'image à un niveau de représentation supérieur au pixel (plus abstrait). Il est caractérisé par un indice unique dans le fichier d'éléments symboliques. Il peut représenter un point, une ligne ou une forme géométrique quelconque. Dans notre projet, nous travaillons avec des éléments symboliques de contour. Dans ces éléments symboliques de contour, un paramètre (la constellation) permet de localiser les pixels appartenant au contour dans un rectangle frontière. Ces éléments de constellation seront caractérisés par une valeur de un ou de zéro, sous forme de chaîne de bits qui correspondent aux pixels de l'image suivant le rectangle frontière, selon que le pixel correspondant appartienne ou non au contour. Les positions ayant une valeur de « 1 » définissent le contour, tous les autres pixels ayant une valeur de « 0 ». L'élément symbolique contenant le contour est aussi caractérisé par la coordonnée du coin inférieur gauche et du coin supérieur droit du rectangle frontière. Les pixels faisant partie du contour et dont les éléments de la constellation sont « 1 » dans le cas que nous étudions, sont ceux qui faisaient partie du contour trouvé par le filtre LoG. Les pixels dans l'élément symbolique couvrent toute la surface du rectangle. Par exemple, si l'élément symbolique

est décrit par un rectangle dont les coordonnées sont (0,0) ; (3,3), on déduit qu'il contient 3x3 points, donc 9 pixels. Pour chaque pixel, l'élément symbolique contient un chiffre binaire (0 ou 1) qui indique si chacun des pixels appartient au contour.

Exemple d'une constellation 3 x 3 :

1	1	1
1	1	0
0	0	0

Figure 16 – Exemple d'une constellation

La série de bits pour cet élément symbolique serait : 0 0 0 1 1 0 1 1 1. L'ordre de balayage est du bas vers le haut et de la gauche vers la droite.

La figure 17 présente une constellation représentée telle que vue dans KBV dans la fenêtre de gauche de l'image, la fenêtre du centre montre la partie de l'image originale des hauteurs correspondantes et finalement la fenêtre de droite montre les détails internes à la constellation.

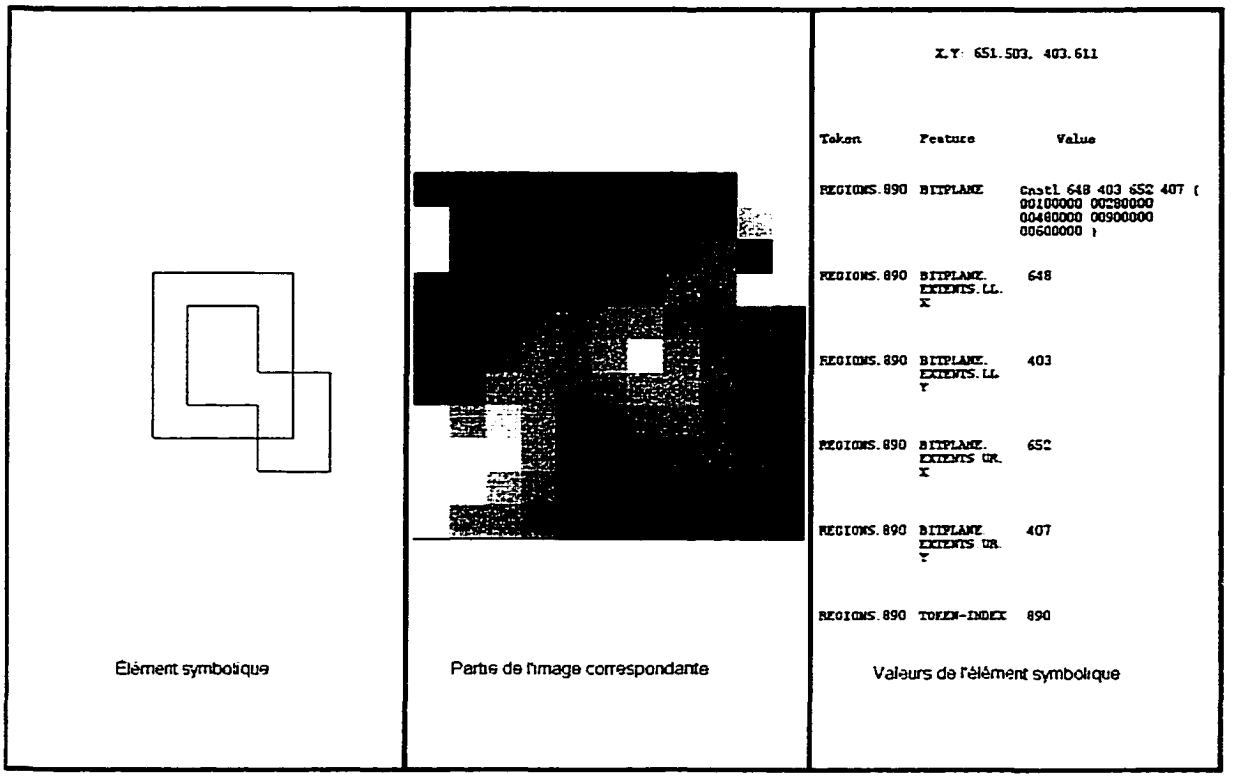


Figure 17 – Paramètre de localisation par constellation d'un élément symbolique dans KBV

4.4.2 Étapes de la création des éléments symboliques

Voici maintenant de façon plus détaillée la chaîne d'opérations effectuée afin d'isoler les arbres individuels avec le Laplacien de Gaussienne (1ère colonne du graphe de la figure 14).

Premièrement une croissance de région est effectuée afin de remplir les contours trouvés par le LoG. En effet, pour effectuer les étapes qui suivent dans KBV, nous devons créer des structures pleines et ensuite binariser l'image. Nous appliquons donc ensuite un seuillage afin de binariser le résultat précédent et, par la suite, nous effectuons un ouverture en deux étapes (érosion suivie de dilatation) car KBV n'a pas d'ouverture/fermeture travaillant en niveaux

de gris. L'ouverture est nécessaire afin de séparer les arbres adjacents reliés par de petites structures. Finalement, à partir des régions résultantes, on produit les élément symboliques avec les tâches « RlabIm » et « RlabToTks » qui étiquettent les régions et transforment ces étiquettes en éléments symboliques.

Nous pouvons voir le résultat de chacune de ces étapes intermédiaires à la figure 18 pour une sous-région quelconque :

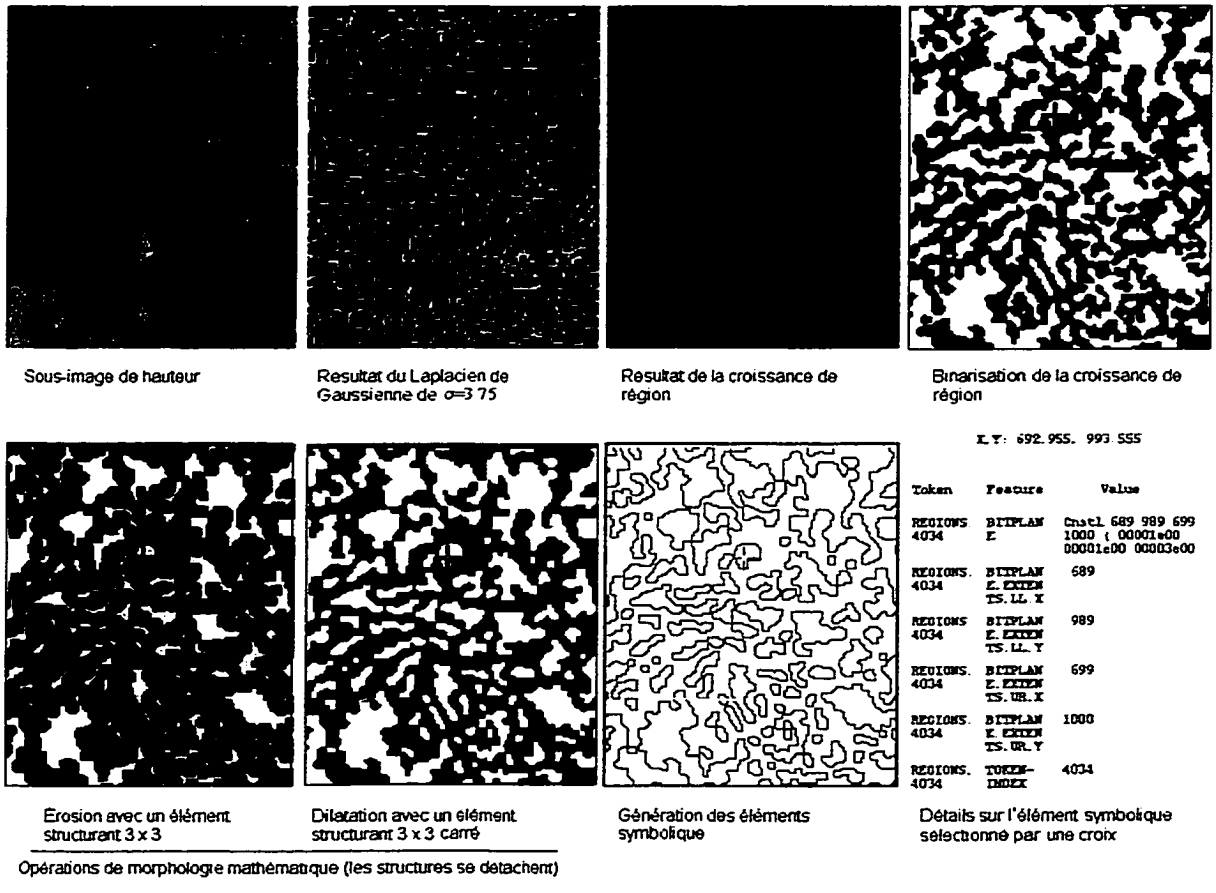


Figure 18 – Étapes menant à la création d'un élément symbolique

Les paramètres permettant de caractériser un élément symbolique sont définis au tableau 6.

Tableau 6 – Valeur des éléments symboliques

BITPLANE	Série de bits de la constellation
BITPLANE.EXTENTS.LL.X	Coordonnée X en bas à gauche
BITPLANE.EXTENTS.LL.Y	Coordonnée Y en bas à gauche
BITPLANE.EXTENTS.UR.X	Coordonnée X en haut à droite
BITPLANE.EXTENTS.UR.Y	Coordonnée Y en haut à droite
TOKEN-INDEX	# de l'élément symbolique

Le tableau 7 présente la liste des tâches KBV utilisées à chaque étape de la création des éléments symboliques ainsi qu'une description de celles-ci.

Tableau 7 – Tâches KBV utilisées

Tâche	Description
ZeroXEdg	Différence de gaussienne (8 voisins)
GrowReg	Algorithme de croissance de région
Seuilbin	Algorithme de seuillage (binarisation de la croissance de région)
ErodeGSElm	Algorithme de morphologie mathématique : érosion
DilateGSElm	Algorithme de morphologie mathématique : dilatation
RegLabIm	Algorithme qui associe un numéro à chaque région de l'image (8 voisins)
RlabToTks	Algorithme qui génère un élément symbolique pour chaque étiquette de l'image
Tksshape	Algorithme calculant différents paramètres sur des élément symboliques dont : Nombre de pixels, longueur de la constellation, largeur de la constellation, logarithme de la largeur vs la longueur, périmètre de la constellation, compacité, nombre de formes convexes, nombre de formes concaves, centroïde, axe majeur, élongation.

La tâche « Seuilbin » a été créée pour effectuer un seuillage rendant la sortie binaire. Le code source de cette tâche est montré à l'annexe 5.

4.4.3 La constellation et l'identification des arbres valides

Maintenant que nous avons des éléments symboliques localisant tous les contours fermés, nous devons éliminer les contours ne représentant pas un arbre unique. En effet, nous pouvons avoir des contours correspondant à plusieurs arbres groupés ou des contours représentant des objets autres que des arbres.

Pour déterminer si l'élément symbolique représente un arbre unique, nous devons vérifier les points suivants :

-le quadrilatère contenant la constellation doit être presque carré

-la constellation elle-même ne doit pas être allongée (par exemple d'un coin à un autre du rectangle)

-la constellation doit aussi être compacte

Une étude manuelle des constellations a révélé que la majorité des arbres uniques ont les caractéristiques suivantes (calculées par la tâche « Tksshape » de KBV).

Tableau 8 – Critères de compacité/élongation d'un arbre unique

Caractéristique	Valeur
Compacité de la constellation	>0.2
Élongation de la constellation	<0.7
Largeur du rectangle frontière de la constellation	>=4 && <=30 pour laplacien de gaussienne avec un σ = 2,25
Largeur du rectangle frontière de la constellation	>=10 && <=45 pour laplacien de gaussienne avec σ = 3,75
Seuil sur la hauteur moyenne de la constellation	1 m

La compacité est calculée en multipliant le nombre de pixels appartenant au contour par 16 et en divisant ce résultat par le périmètre du contour au carré, tandis que l'élongation se définit comme la différence entre les axes principaux de l'ellipse s'ajustant le mieux sur le contour, divisé par la somme de ces mêmes axes. Un cercle a une compacité de 1 et une élongation de 0, tandis

qu'une ellipse très allongée a une compacité près de 0 et une élongation près de 1.

La raison pour laquelle nous avons dû prendre une valeur de seuil aussi basse pour la hauteur moyenne de l'arbre (1 m) est que le LoG identifie presque toujours un contour plus grand que l'arbre lui même et de ce fait la hauteur moyenne des pixels de la région est diminuée. La valeur de 1 mètre a été trouvée expérimentalement comme étant la meilleure en combinaison avec les autres critères choisis. En effet, en seuillant à 5 mètres, plusieurs constellations contenant des arbres valides étaient éliminées.

Puisque nous utilisons deux grandeurs de sigma du filtre LoG, nous produisons et utilisons deux fichiers d'éléments symboliques distincts pour les deux grandeurs. Ainsi, nous devons traiter les informations de ces deux fichiers parallèlement. Pour ce faire, nous avons développé un programme qui trouve la position des arbres à partir des deux fichiers de constellations. Ce programme est une tâche KBV que l'on peut retrouver à l'annexe 6. Ce programme « constell.c » produit un nouveau fichier d'éléments symboliques contenant des points (éléments symboliques de type point) aux endroits où des arbres ont été trouvés, en utilisant les critères énoncés au tableau 8. Quelques fois on retrouve deux points pour le même arbre. Cela vient du fait que nous utilisons deux fichiers de constellations et que parfois, les contours des arbres trouvés, donc les constellations, ne se superposent pas exactement. Nous pouvons croire, grâce aux critères utilisés pour sélectionner les arbres à conserver, que seuls les arbres identifiés deux fois et les arbres regroupés sont mal identifiés. Nous devons ajouter que la position de l'arbre trouvée ici servira plus loin de position de départ pour le calcul de hauteur des arbres (centroïde de l'arbre).

4.5 Les éléments symboliques et les arbres sur le terrain

À cette étape de notre projet, il est nécessaire de modifier les fichiers d'éléments symboliques pour y incorporer les informations sur les arbres dont nous avons les mesures terrain. Cette étape est nécessaire pour valider la méthode de calcul des hauteurs, mais ne fait pas partie de la méthode elle-même. Elle a été effectuée en partie manuellement (identification des constellations qui correspondent aux arbres dont nous connaissons les mesures) et en partie avec une tâche KBV « typearbr.c » présentée à l'annexe 7 (ajout de caractéristiques aux éléments symboliques « points » existants).

Pour chacun des arbres que nous avons correctement identifiés, les caractéristiques de type d'arbre et de forme ont été ajoutées au fichier d'éléments symboliques de positions d'arbres.

Le tableau 9 présente les données prises sur le terrain pour les arbres identifiés à la figure 13. Les formes d'arbres ont été évaluées lors de la mesure de l'arbre par l'observateur. Le tableau contient l'espèce et la forme de l'arbre, la hauteur réelle mesurée sur le terrain et la hauteur du point laser le plus haut (déterminé à la main par un tierce personne) ayant frappé la couronne. Toutes les mesures sont en mètres.

Tableau 9 – Données pour chacun des arbres identifiés sur le terrain

ID	Espèce	Forme	Hauteur réelle	Hauteur de point laser le plus haut
1	bouleau	sphérique	17,27	16,5
2	épinette	ogive	22,24	22,3
3			11,85222	9,31
4	épinette	ellipsoïde	23,09	22,3
5	peuplier	sphérique	23,97	22,6
6	épinette	ogive	22,54	19,7
7	bouleau	sphérique	15,80	12,5
8	saule	sphérique	6,89	4,9
9			16,4002	10,84
10	épinette	ogive	25,77	22,9
12	bouleau	ellipsoïde	17,54913	13,77
13			19,98908	19,74
14	bouleau	ellipsoïde	17,46019	15,5
16	bouleau	sphérique	18,39664	13,38
17	thuya	ogive	8,593223	7,64
18			15,39707	12,22
19	épinette	ogive	20,66693	15,4
20			24,41001	22,3
30	thuya	ogive	12,68	11,6
31	bouleau	sphérique	14,44	14,4
32	sapin baumier	ogive	12,19714	6,43
34	thuya	ogive	13,31535	8,89
35	épinette	ogive	12,99269	8,17
36			8,09164	4,7

40	bouleau	sphérique	12,42543	4,83
42			22,96	22,2
43			16,08877	12,51
46	épinette	ogive	23,96881	16,9
48			7,447307	4,55
49	sapin baumier	ogive	16,46347	10,12
51	thuya	ogive	11,399	11,42
52	épinette	ogive	16,04557	11,49
54			12,81689	7
56			8,760707	6,05
57			6,786144	4,74
58			7,796665	5,31

Nous avons choisi certains des arbres précédents pour continuer les tests en fonction du résultat du filtre LoG et de l'algorithme identifiant l'emplacement des couronnes. Les arbres éliminés sont ceux ayant été confondus avec un ou plusieurs de leurs voisins par le LoG, ou n'ayant pas été identifiés du tout. Nous avons retenu 18 arbres, soient les arbres suivants : 1, 2, 3, 8, 9, 10, 14, 16, 17, 18, 19, 34, 35, 36, 42, 43, 54, 56.

4.6 Calcul de hauteur des arbres

L'objectif lié à cette section est de prédire la hauteur du sommet de la couronne des arbres d'après celle de tous les points tombés sur la couronne.

4.6.1 Extraction des points laser bruts

Maintenant que nous avons identifié les éléments symboliques contenant les constellations qui correspondent aux arbres mesurés sur le terrain, nous pouvons extraire les points laser de données brutes correspondant à chacun des arbres dont nous avons les mesures. Pour ce faire, nous avons créé un programme que l'on retrouve à l'annexe 8 et qui n'extrait que les données laser appartenant à l'arbre voulu.

4.6.2 Méthodes de calcul de hauteur

Maintenant que nous avons les points de données brutes pour chacun des 18 arbres, nous devons calculer la hauteur de ces arbres. Nous utiliserons premièrement les trois approches suivantes :

- régression simple (droite)
- régression polynômiale (parabole)
- régression polynômiale (surface)

Pour les deux premières méthodes, les données sont recueillies de la même façon, comme indiqué à la figure 19. Nous faisons passer parmi les données laser 3D un plan dont le coin inférieur gauche est placé au centre de la surface de la couronne projetée sur le sol. Ensuite nous faisons tourner le plan autour de l'axe de rotation z et chaque fois que le plan entre en contact avec un point laser, on note cette position sur le plan. Cela résultera en une vision 2D

du profil de l'arbre, la figure 19 nous le démontre. La hauteur des points laser correspond à z et la distance entre l'origine du système d'axe (centre de l'arbre) et le point laser correspond à $r = \sqrt{x^2 + y^2}$ (ne pas confondre ici le r avec le r du LoG). Nous obtenons à la fin de cette opération un plan (r,z) contenant le profil de l'arbre.

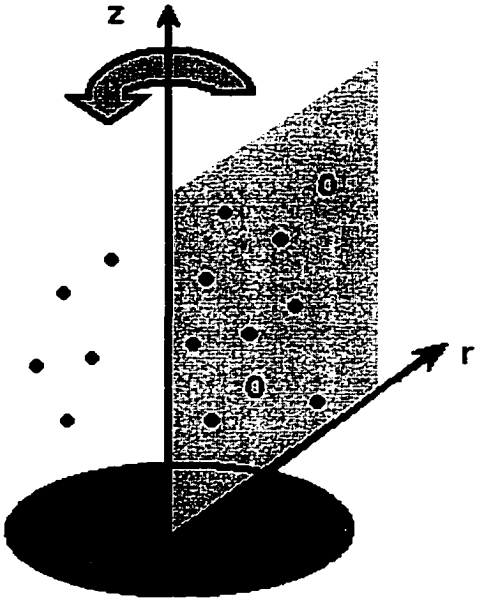


Figure 19 – Extraction des données pour régressions 2D

Pour la troisième méthode, la régression polynômiale surfacique, nous utilisons directement les données en 3D, la seule modification effectuée est le changement de référentiel. En effet, les coordonnées seront transformées pour que la référence (0,0,0) corresponde au centre de l'arbre au sol, voir la figure 20.

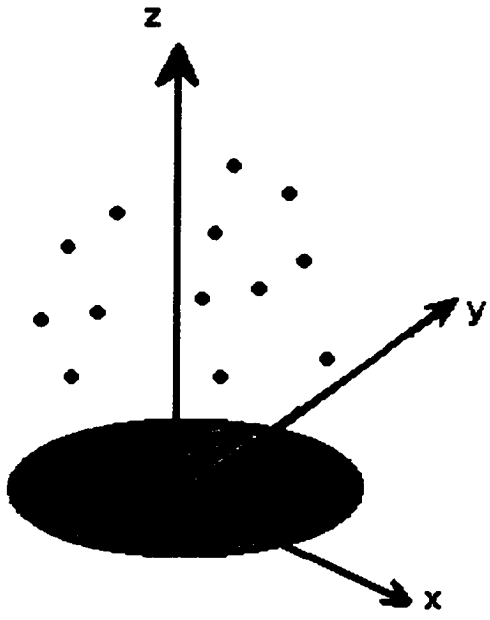


Figure 20 – Extraction des données pour régressions 3D

4.6.3 Formules de régression

4.6.3.1.Régression simple

Nous utilisons la régression linéaire simple pour prédire la hauteur de l'arbre en émettant l'hypothèse que le profil de l'arbre est une droite. Cela peut s'appliquer à un conifère qui a une forme conique en 3D. L'ordonnée à l'origine de la droite de régression simple nous donnera directement la hauteur de l'arbre pour la position de centre choisie. Pour réaliser la régression linéaire simple, nous pouvons utiliser la méthode des moindres carrés [16] où nous voulons résoudre $z = a + br$ avec $AX = K$ (voir figure 21). On trouve X avec la formule $X = (A^TA)^{-1}A^TK$ avec lequel nous obtenons la valeur des deux paramètres a et b que nous remplaçons ensuite dans l'équation $z = a+br$. De plus, nous pouvons déduire de ceci que lorsque $r = 0$, alors $z = a$ et a est la hauteur de l'arbre. Les

vecteurs r et z sont connus, puisque ce sont les données que nous avons recueillies pour chacun des arbres.

$$A=\begin{bmatrix} 1 & r_1 \\ 1 & r_2 \\ \cdots & \\ 1 & r_i \end{bmatrix} \quad X=\begin{bmatrix} a \\ b \end{bmatrix} \quad K=\begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_i \end{bmatrix} \tag{17}$$

De plus, lorsque l'on effectue la régression simple, nous effectuons un déplacement de quatre positions vers le haut, le bas, la gauche et la droite du centroïde afin de trouver la meilleure position pour le centre de l'arbre. Supposons P, la position (x,y) présumée de l'arbre et $E_{i,j}$ l'erreur d'ajustement associée à la position i,j. Nous voulons minimiser P = i, j | Min[$E_{i,j}$] pour i compris entre x ± 2 m et j compris entre y ± 2 m, où x, y, i et j sont des positions discrètes dans le nuage de points formés de pixels de ½ mètre de résolution. L'ensemble des points i, j autour du centroïde x, y d'un élément symbolique possède donc un cardinal de 9 X 9 = 81 positions. Nous ne conservons que la position de centroïde, ainsi que la droite de régression correspondante à la position où l'erreur d'ajustement de la droite est minimale. Cette position de centroïde sera aussi utilisée pour calculer la courbe de régression et la surface de régression.

4.6.3.2.Régression polynômiale

Nous utilisons la régression polynômiale pour prédire la hauteur de l'arbre en considérant que le profil de l'arbre est une courbe ayant son sommet à l'origine en x. Cela peut s'appliquer à un feuillu, qui a une forme plutôt sphérique en 3D. L'ordonnée à l'origine de la courbe de régression nous donne

directement la hauteur de l'arbre pour la position de centre choisie. Pour réaliser la régression polynômiale (qui ajuste une parabole à des données), nous pouvons aussi utiliser la méthode des moindres carrés, de la même façon que précédemment, où on veut résoudre $z = a + br + cr^2$ avec $AX = K$. Nous trouvons X avec la formule $(A^TA)^{-1}A^TK$. De plus, nous pouvons en déduire que lorsque $r = 0$, alors $z = a$ et a est la hauteur de l'arbre.

$$A=\begin{bmatrix} 1 & r_1 & r_1^2 \\ 1 & r_2 & r_2^2 \\ & \cdots & \\ 1 & r_i & r_i^2 \end{bmatrix} \quad X=\begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad K=\begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_i \end{bmatrix} \tag{18}$$

4.6.3.3.Régression polynômiale surfacique

Nous avons aussi tenté d'ajuster une surface courbe aux données laser en espérant que la surface épouserait la forme présumée de l'arbre. La valeur de la surface au point $x = 0$ et $y = 0$ nous donne dans ce cas la hauteur présumée de l'arbre. Pour réaliser la régression polynômiale surfacique (qui ajuste une surface courbe à des données), nous pouvons également utiliser la méthode des moindres carrés, toujours de la même façon, où nous voulons résoudre $z = a + bx + cy + dx^2 + ey^2 + fxy$ avec $AX = K$. Nous trouvons toujours X avec la formule $(A^TA)^{-1}A^TK$. Aussi, comme dans les cas précédents, lorsque $x = 0$ et $y = 0$, alors $z = a$ et a est la hauteur de l'arbre.

$$A=\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ 1 & x_2 & y_2 & x_2^2 & y_2^2 & x_2y_2 \\ & & \cdots & & & \\ 1 & x_i & y_i & x_i^2 & y_i^2 & x_iy_i \end{bmatrix} \quad X=\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} \quad K=\begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_i \end{bmatrix} \qquad (19)$$

4.6.4 Résultats des régressions

Voyons un exemple de résultat que nous pouvons obtenir de ces méthodes avec les données de l'arbre numéro 8. Le programme utilisé pour le reste des calculs est à l'annexe 9.

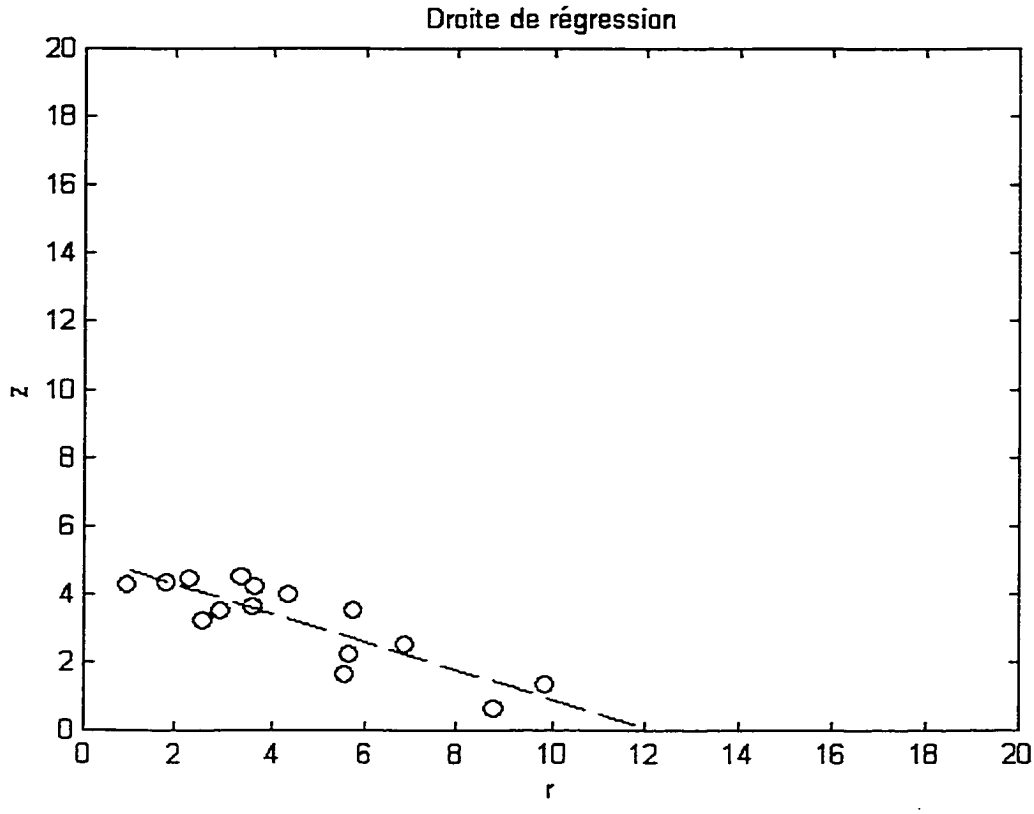


Figure 21 – Résultat droite de régression

La hauteur réelle de cet arbre est de 6,89 mètres et la droite de régression prédit une hauteur de 5,1096 mètres pour l'ordonnée à l'origine (qui est en fait équivalente au centre de l'arbre), ce qui est trop bas pour être satisfaisant. On peut voir sur la figure 21 la dispersion des données autour de la droite. Puisque la régression fait passer au mieux la droite parmi les données et non sur l'enveloppe des données, le résultat est que l'ordonnée à l'origine est plus basse qu'elle ne devrait être.

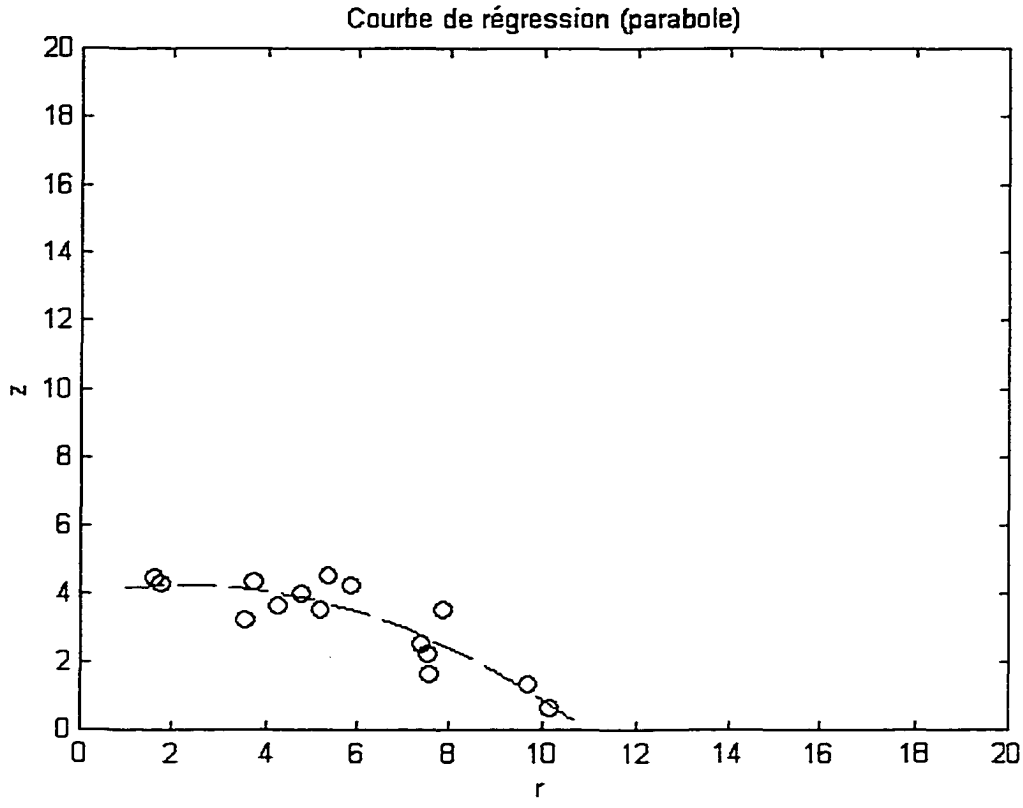


Figure 22 – Résultat parabole de régression

Sur la figure 22, nous pouvons voir le résultat de la régression polynômiale, qui donne une valeur d'ordonnée à l'origine de 3,96 mètres. Ce résultat est encore moins bon que celui de la droite de régression. Cela s'explique par le fait que la courbe passe au mieux dans les données et non sur l'enveloppe des données.

Nous pouvons dès maintenant imaginer que la surface de régression donnera aussi un résultat sous la vérité, ce qui est justement le cas (voir figure 23). La valeur de hauteur au point x = 0 et y = 0 est en effet de 4,39 mètres, plus de 2 mètres sous la vérité. Cela s'explique toujours de la même façon.

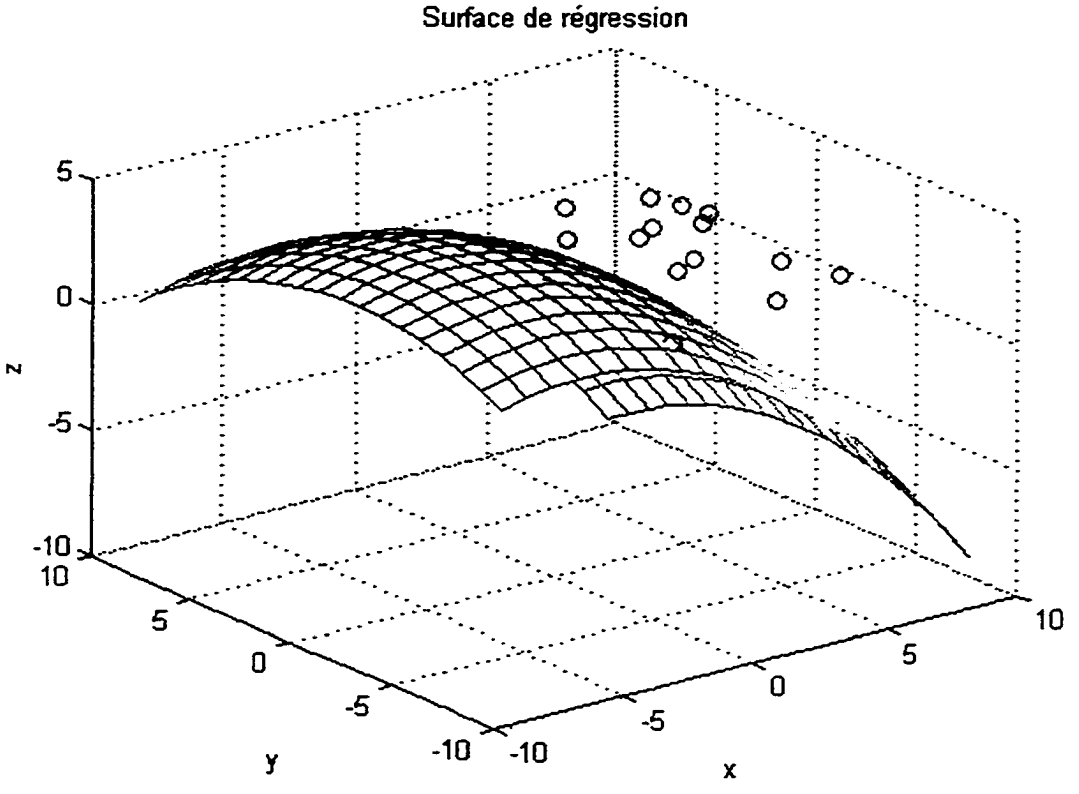


Figure 23 – Résultat surface de régression

4.6.5 Amélioration

Une amélioration des résultats précédents peut être obtenue en effectuant les opérations qui seront décrites plus bas. Posons d'abord les hypothèses.

L'erreur en x, y sur les mesures laser est d'environ le 1/1000 de la hauteur de vol, en l'occurrence 1/1000 * 700 m = 70 cm. L'erreur en z, est quant à elle, minime (moins de 20 cm). L'erreur en x,y dépend des conditions d'acquisition :

- direction du vol : les points (données), recouvrant une couronne d'arbre, ont été acquis lors de quatre passages différents (2 survols * recouvrement de deux lignes de vol)
- constellation GPS : les satellites, s'ils sont bien distribués dans l'espace au moment de survol, donnent une mesure très fiable de la coordonnée x, y, par contre si les satellites sont mal distribués dans l'espace, les erreurs de position deviennent plus importantes

Ces points comportent donc théoriquement des erreurs avoisinant 70 cm, mais pouvant se produire dans des directions différentes selon les passages. De plus, nous ne connaissons pas la distribution des erreurs, nous ne savons pas avec certitude si 70 cm est l'erreur maximale ou si ce peut être l'erreur moyenne.

Étant donné ce qui précède, nous faisons l'hypothèse que la coordonnée z peut donner une indication de la position du point x,y par rapport au centre de la couronne, si la forme de la couronne est connue (c'est le cas ici). Par exemple, une valeur élevée de z indique une position plus rapprochée du centre qu'une faible valeur de z. En effet, la forme générale des couronnes présente une décroissance continue du centre vers la périphérie de la couronne.

Ces hypothèses sont utilisées pour justifier les étapes suivantes effectuées pour chaque position de centroïde essayée (en conservant la position de centroïde donnant les meilleurs résultats pour la droite de régression):

- Trier les coordonnées z en ordre décroissant
- Trier les coordonnées r en ordre croissant
- Refaire les couples (r,z) : premier r avec premier z, ..., dernier r avec dernier z

La méthode vise à préserver les distribution des deux variables.

CHAPITRE 5
RÉSULTATS

5.1 Les hauteurs calculées

Les résultats obtenus pour chacun des 18 arbres sont montrés aux figures 25 à 42 de la section 5.3. Les résultats, obtenus avec l'amélioration de la droite et de la courbe de régression, sont satisfaisants selon les critères établis en premier lieu, soit d'obtenir des mesures meilleures que seulement le point laser le plus haut sur la couronne et d'obtenir des mesures inférieures à d'un mètre pour la majorité des arbres.

De plus, nous devrons comparer nos résultats avec ceux de St-Onge [9], qui utilisait une formule de régression prédisant la hauteur réelle de l'arbre compte tenu de la valeur de plus haut point faisant partie de la couronne.

Avant de présenter les figures, nous présenterons des tableaux compilant les résultats, ainsi que quelques statistiques sur les résultats obtenus.

Tableau 10 – Résultats expérimentaux

Arbre no.	Forme de la couronne	Hauteur réelle ; moy. des mesures terrain	Hauteur laser maximumm de la couronne	Hauteur St-Onge [9]	Hauteur régression linéaire données non triées	Hauteur régression parabolique données non triées	Hauteur régression linéaire sur les données triées	Hauteur parabole de régression sur les données triées
1	Sphère		16,5	19,25	16,71	17,59	18,51	
2	Ogive		22,3	24,53	22,75	18,03	25,26	
3	Inconnue		9,31	12,71	1,01	-4,08	12,77	
8	Sphère		4,9	8,70	5,11	3,96		5,68
9	Inconnue		12,5		10,54	3,31	11,53	9,68
10	Ogive		22,9	25,08	20,37		27,39	25,01
14	Ellipse		15,5	18,34	13,61	14,15	20,0	
16	Sphère		13,38	16,42	14,93	12,99		12,72
17	Ogive		7,64	11,19	10,13	3,81	10,13	
18	Inconnue		12,22		13,64	12,04	15,79	13,18
19	Ogive		15,4	18,25	15,39	9,26		16,69
34	Ogive		8,89		15,6	-25,97	11,33	9,91
35	Ogive		8,17		3,51	0,5	13,51	10,91
36	Inconnue		4,7		6,12	4,56	6,15	6,28
42	Inconnue		22,2	24,44	23,63	20,42	27,48	
43	Inconnue		12,51		11,87	5,1	14,02	12,57
54	Inconnue		7,0		7,61	7,23	9,12	7,56
56	Inconnue		6,05	9,74	1,46	17,39		7,42

Les résultats précédents montrent bien que la droite et la courbe de régression calculées sur les données non triées donnent des résultats contenant

un taux d'erreur élevé. En conséquence, nous retenons comme meilleure méthode la droite ou la parabole de régression ajustée sur les données triées. En comparaison, les résultats de St-Onge sont bons (sept arbres où les résultats sont meilleurs), mais une plus grande quantité d'hauteurs d'arbres (dix) ont été calculées avec plus de précision avec la méthode présentée ici.

Les arbres 34, 36 et 56 n'ont que 5, 3 et 4 points laser ayant frappé la couronne respectivement, ce qui est bien peu de points pour être certain que la valeur calculée est exacte. Aussi, il y a deux incohérences pour les arbres 9 et 10. Dans la cas de l'arbre 9, il est indiqué que le point laser le plus haut de la couronne est 12,5 mètres, alors que le point laser le plus haut est 9,25 mètres. Tandis que dans le cas de l'arbre 10, le point laser extrait le plus haut est 23,86 mètres alors que celui apparaissant sur les données de référence est 22,9 mètres. Ces problèmes particuliers peuvent altérer les résultats.

Si on ne tenait pas compte des cinq arbres contenant des irrégularités, on aurait 10/13 arbres mieux identifiés par notre méthode et 3/13 arbres mieux identifiés par la méthode St-Onge.

5.2 Les statistiques

Toutes les hauteurs d'arbres calculées sont en dedans de 70% de la hauteur réelle de l'arbre (pour tous les 18 arbres) et 83% des valeurs calculées sont en dedans de 85% de la hauteur réelle, tel que présenté à la figure suivante.

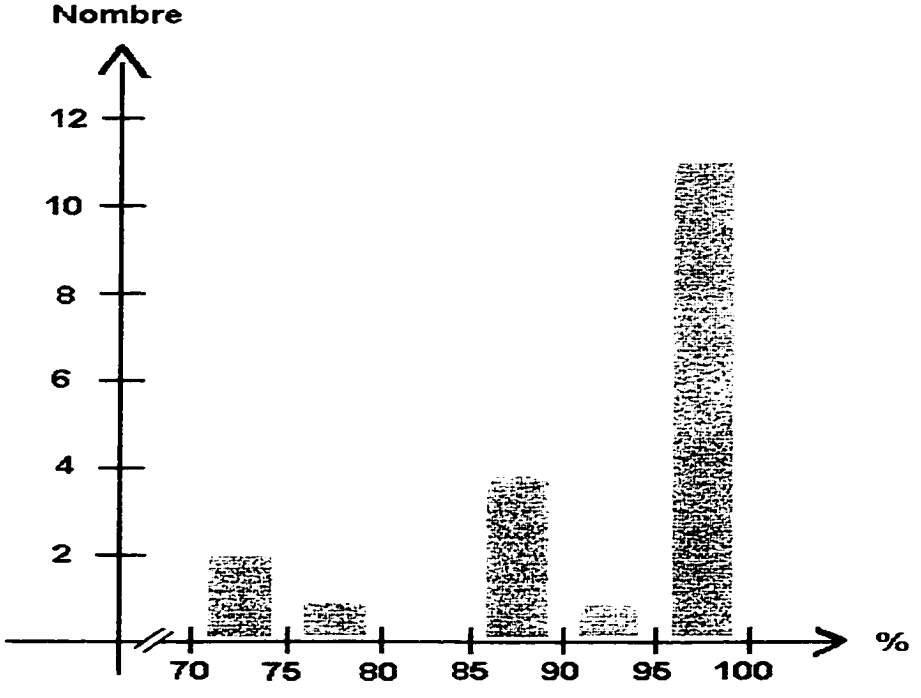


Figure 24 – Nombre d'arbres en fonction du ratio hauteur prédite sur hauteur réelle

Le pourcentage des arbres, dont la hauteur calculée est plus près de la hauteur réelle de l'arbre que le plus haut point laser la couronne, est de 83 %. Le pourcentage d'arbres dont la valeur calculée est à l'intérieur de 1 mètre de la hauteur réelle est 61 %; sur ces 11 arbres, plus de la moitié (6) ont une hauteur calculée en dedans de 50 cm de la hauteur réelle. Aussi, 72,1 % des arbres ont

une hauteur calculée en dedans de 2 mètres et tous les arbres ont une hauteur calculée en dedans de 4 mètres de la vraie hauteur. Il faut noter ici que ces statistiques incluent des arbres dont les deux mesures terrain avaient jusqu'à trois mètres d'écart entre elles.

5.3 Les figures

Voici maintenant les figures illustrant les droites et paraboles de régression ayant été utilisées pour la clacul des hauteurs des 18 arbres. Les mesures de hauteur ont déjà été présentées au tableau 10.

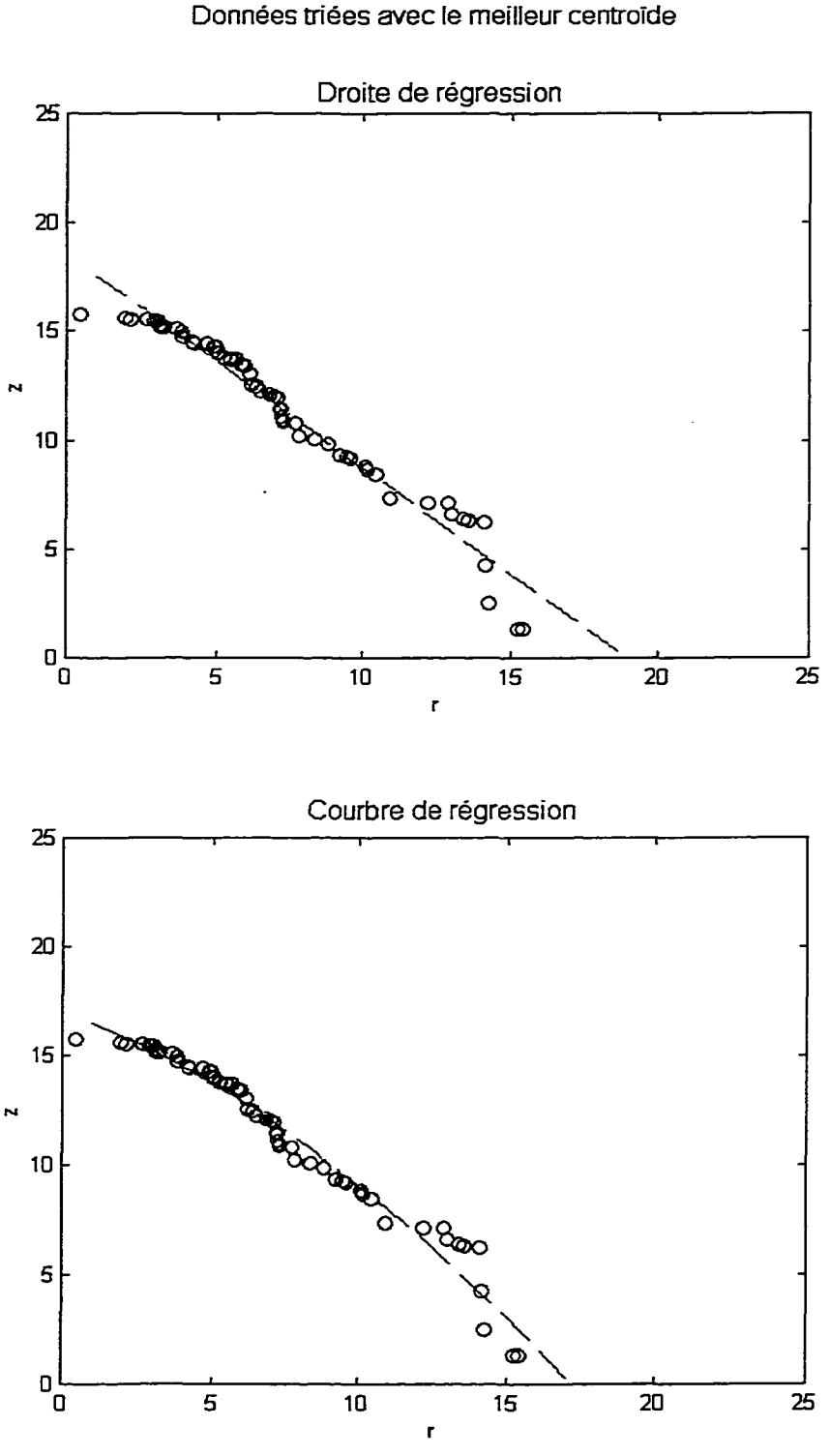


Figure 25 – Résultat des extrapolations pour l'arbre 1

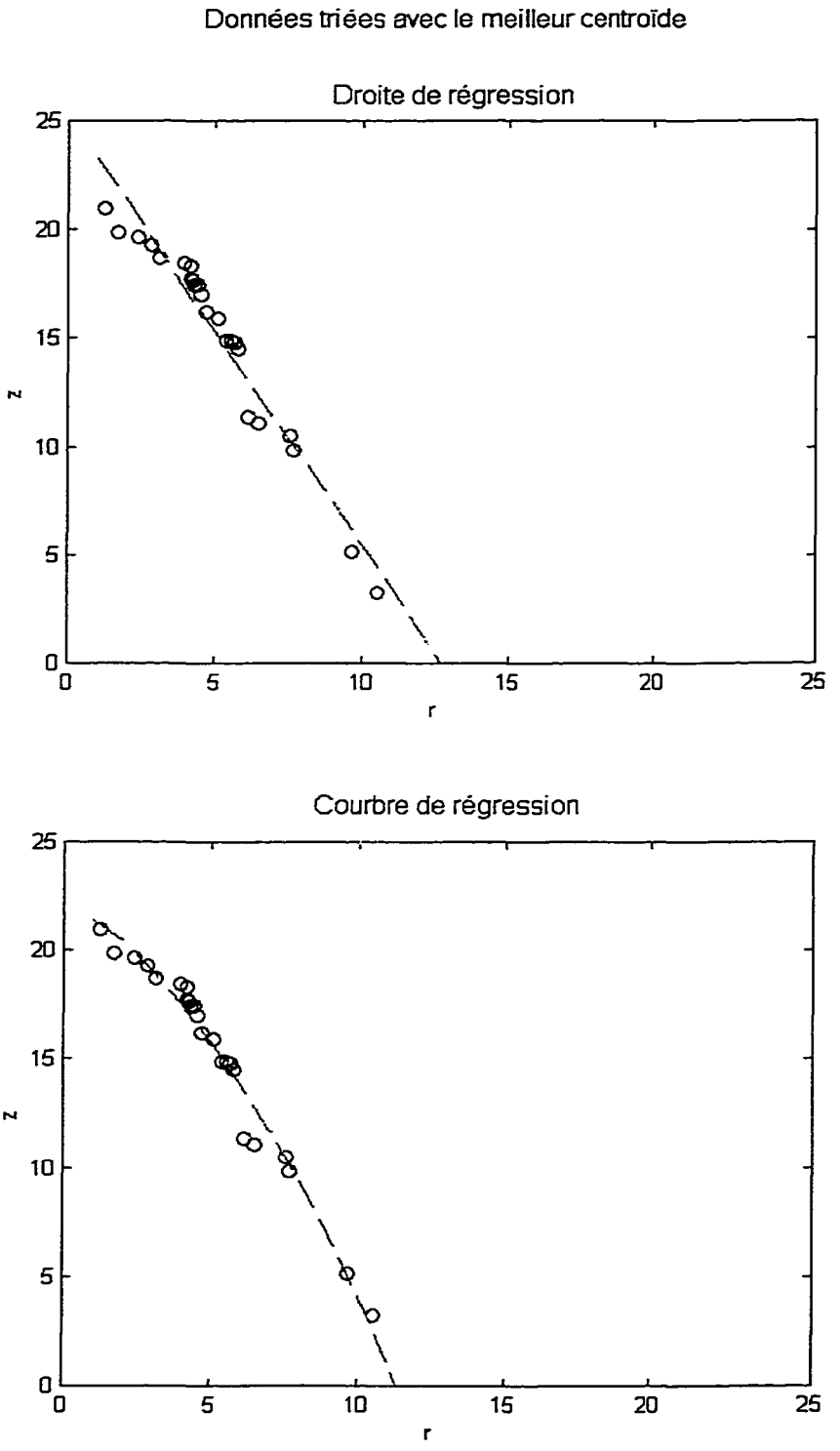


Figure 26 – Résultat des extrapolations pour l'arbre 2

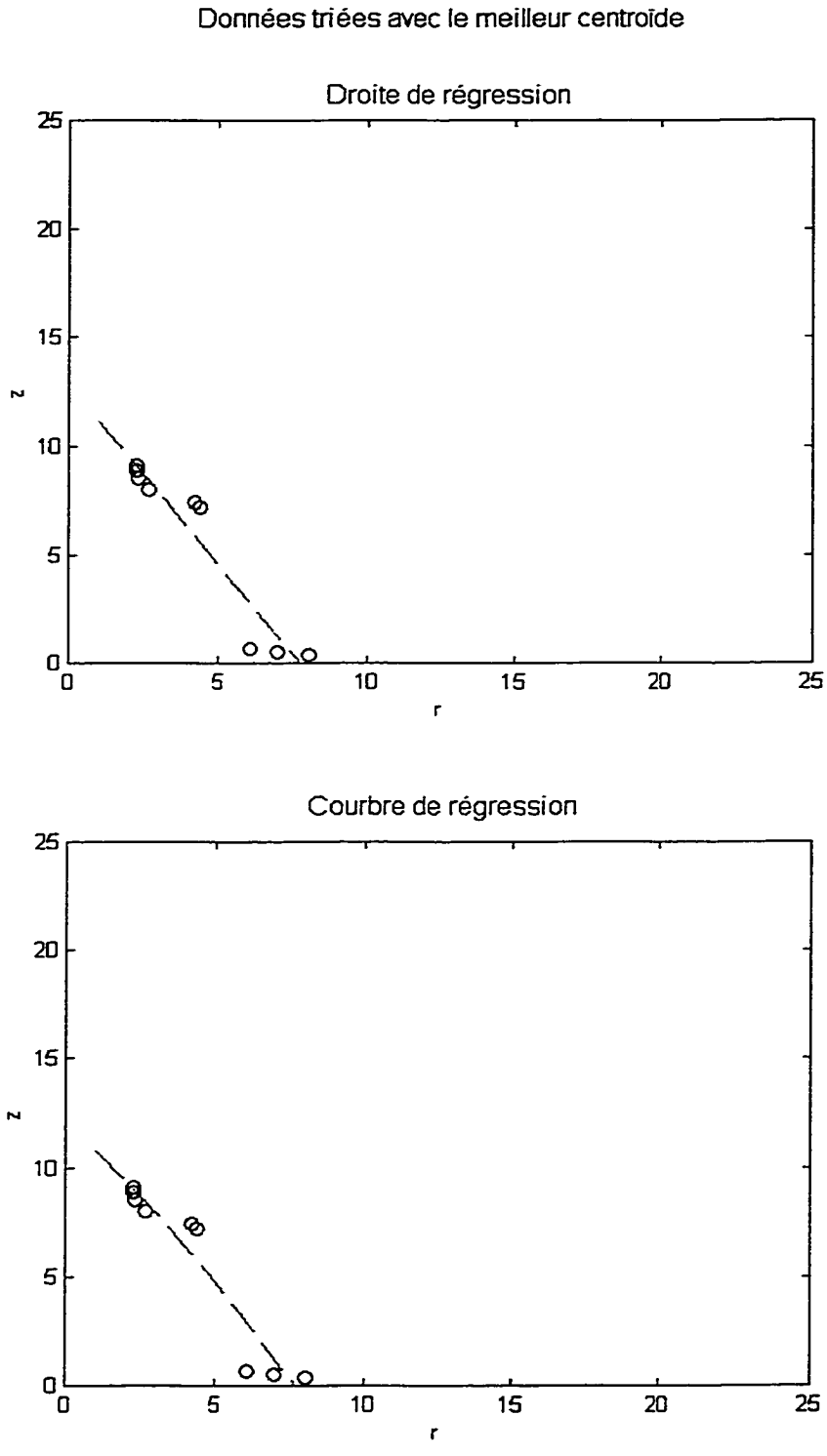


Figure 27 – Résultat des extrapolations pour l'arbre 3

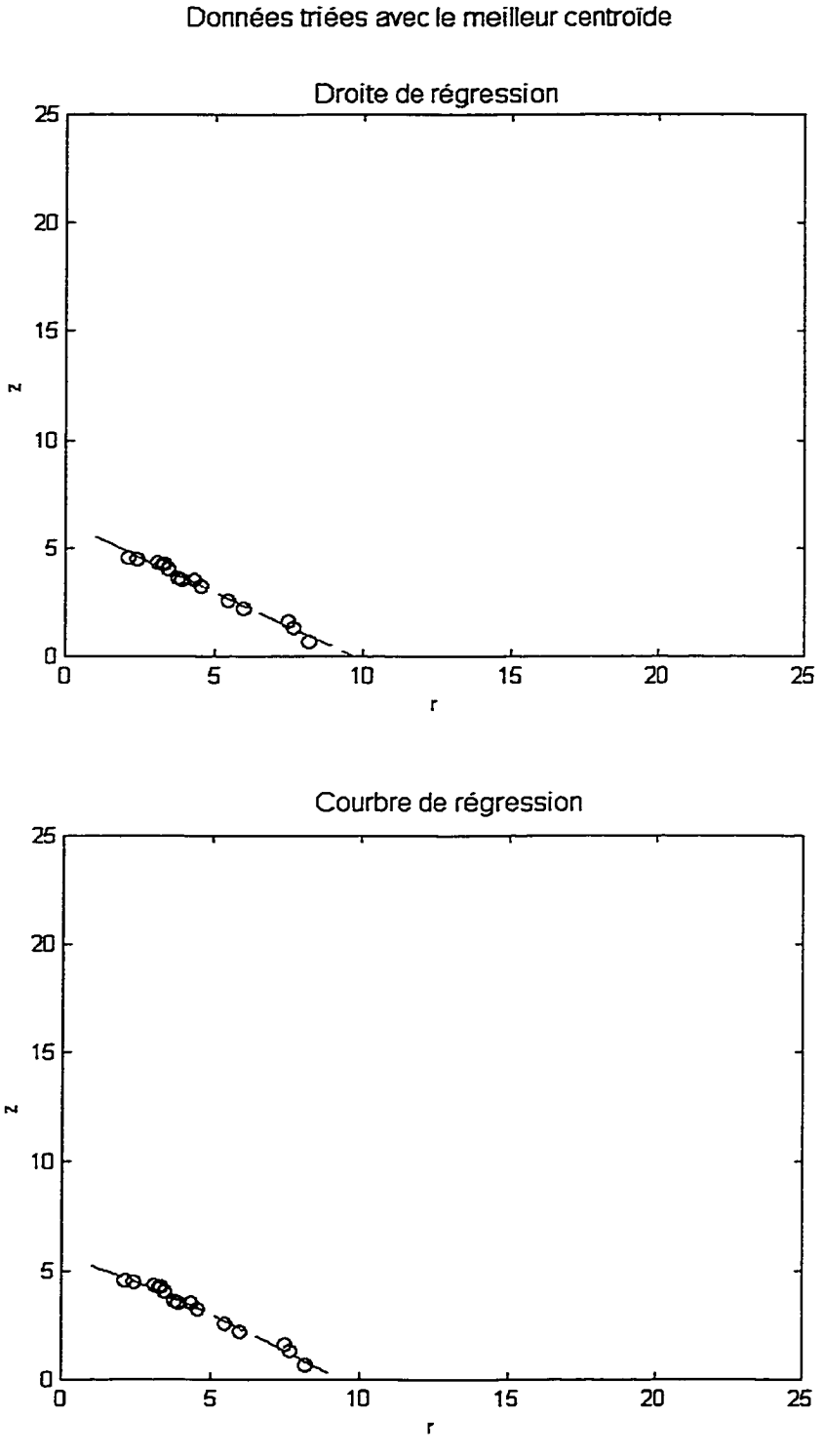


Figure 28 – Résultat des extrapolations pour l'arbre 8

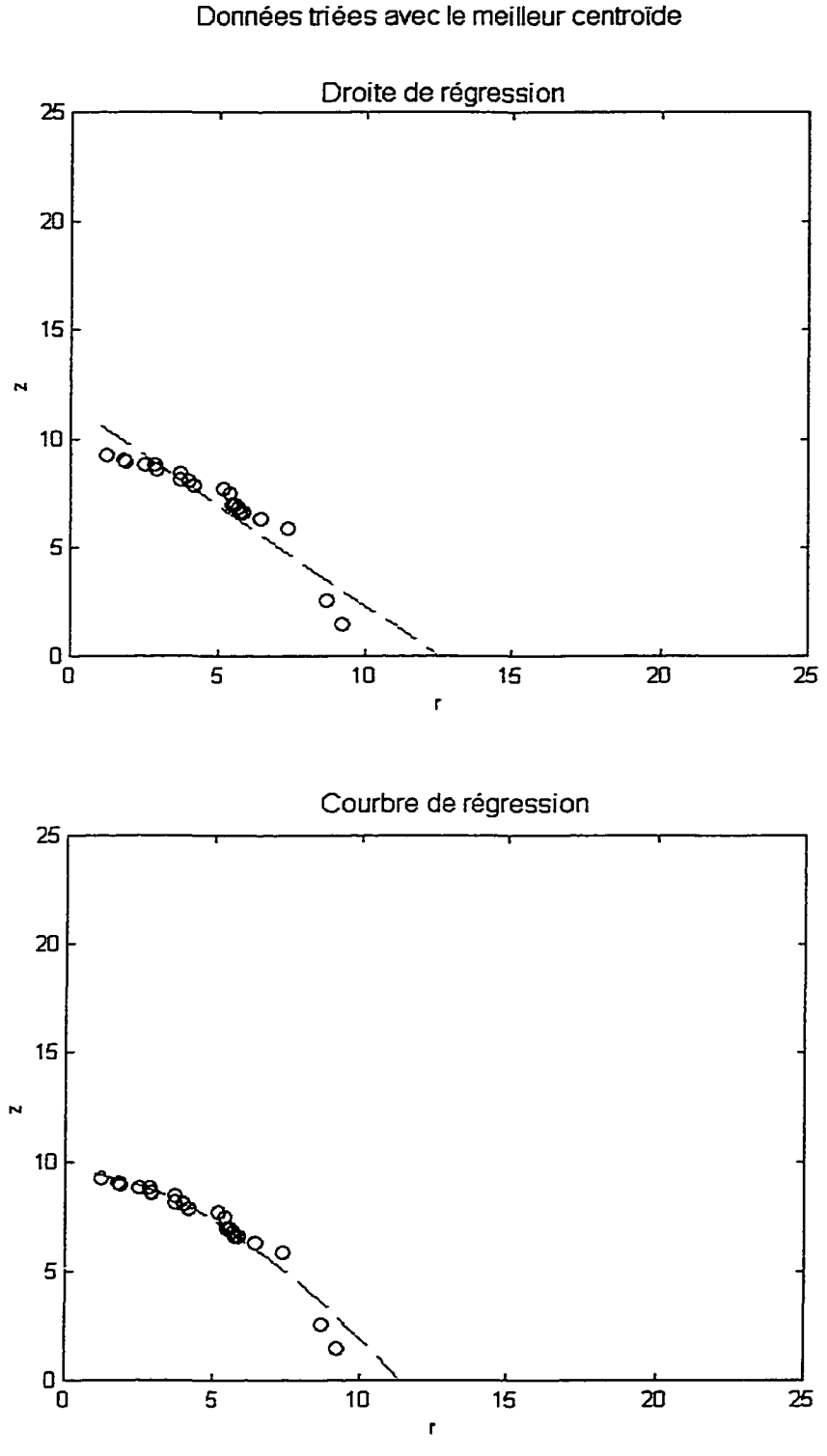


Figure 29 – Résultat des extrapolations pour l'arbre 9

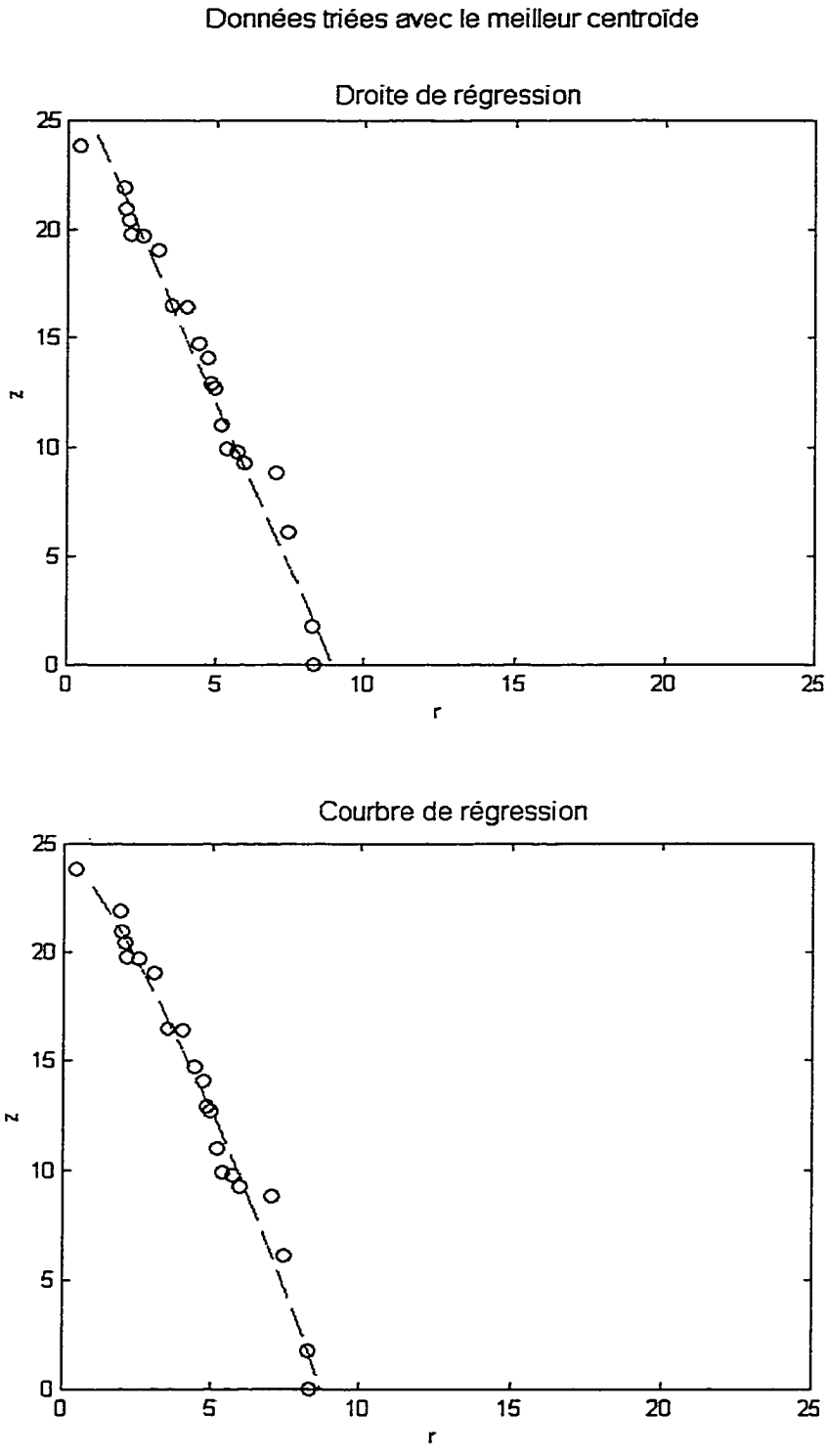


Figure 30 – Résultat des extrapolations pour l'arbre 10

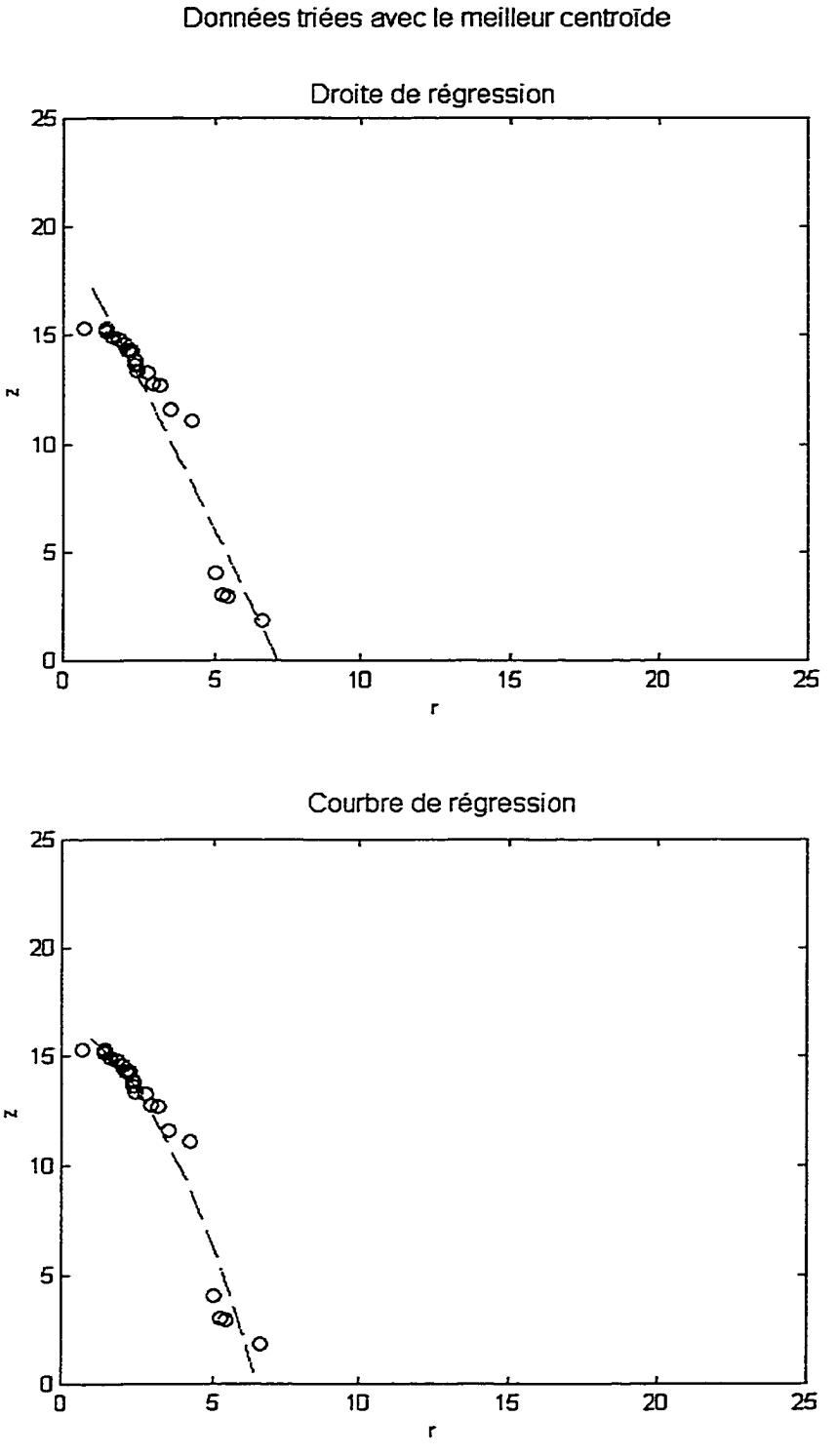


Figure 31 – Résultat des extrapolations pour l'arbre 14

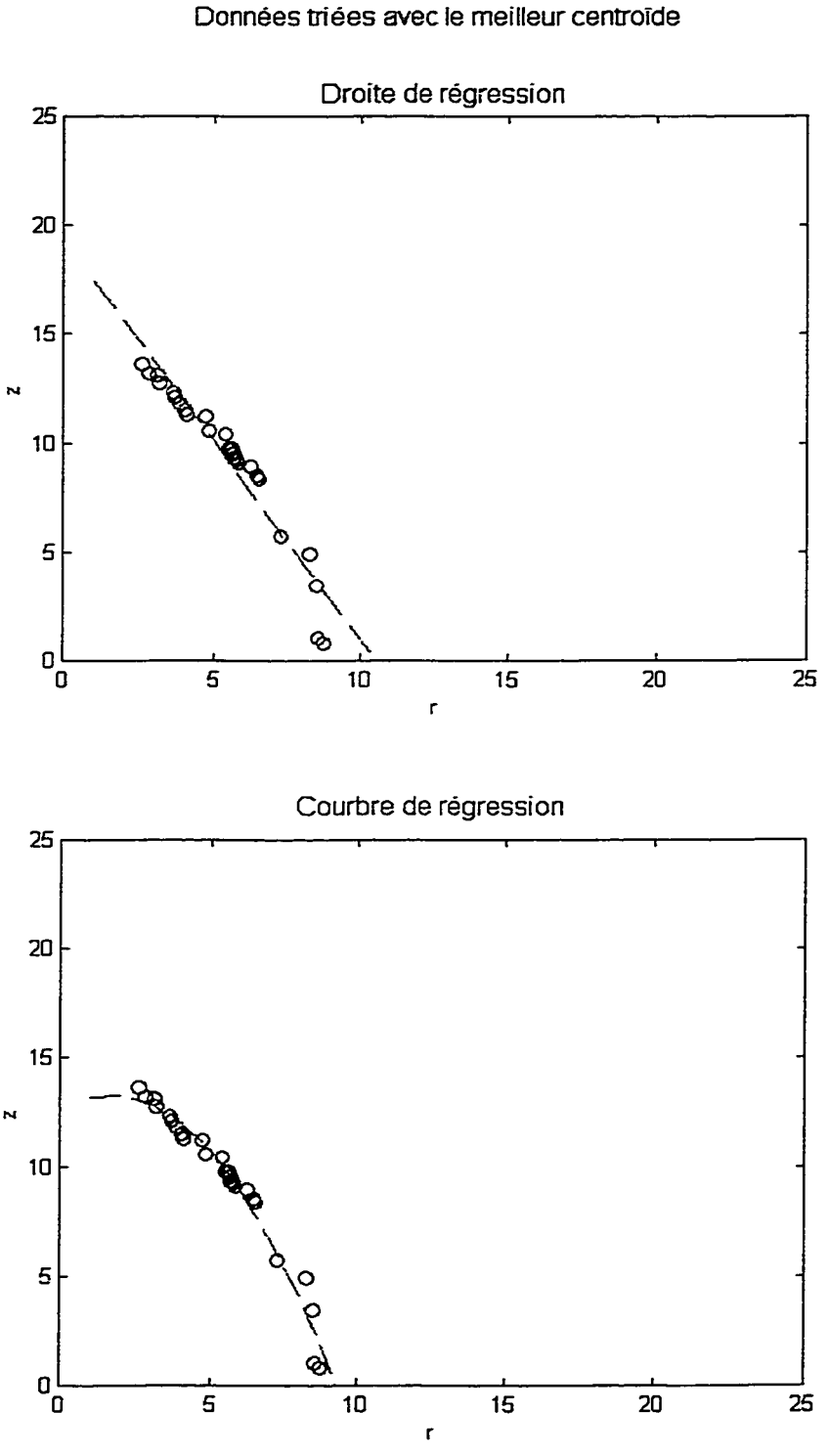


Figure 32 – Résultat des extrapolations pour l'arbre 16

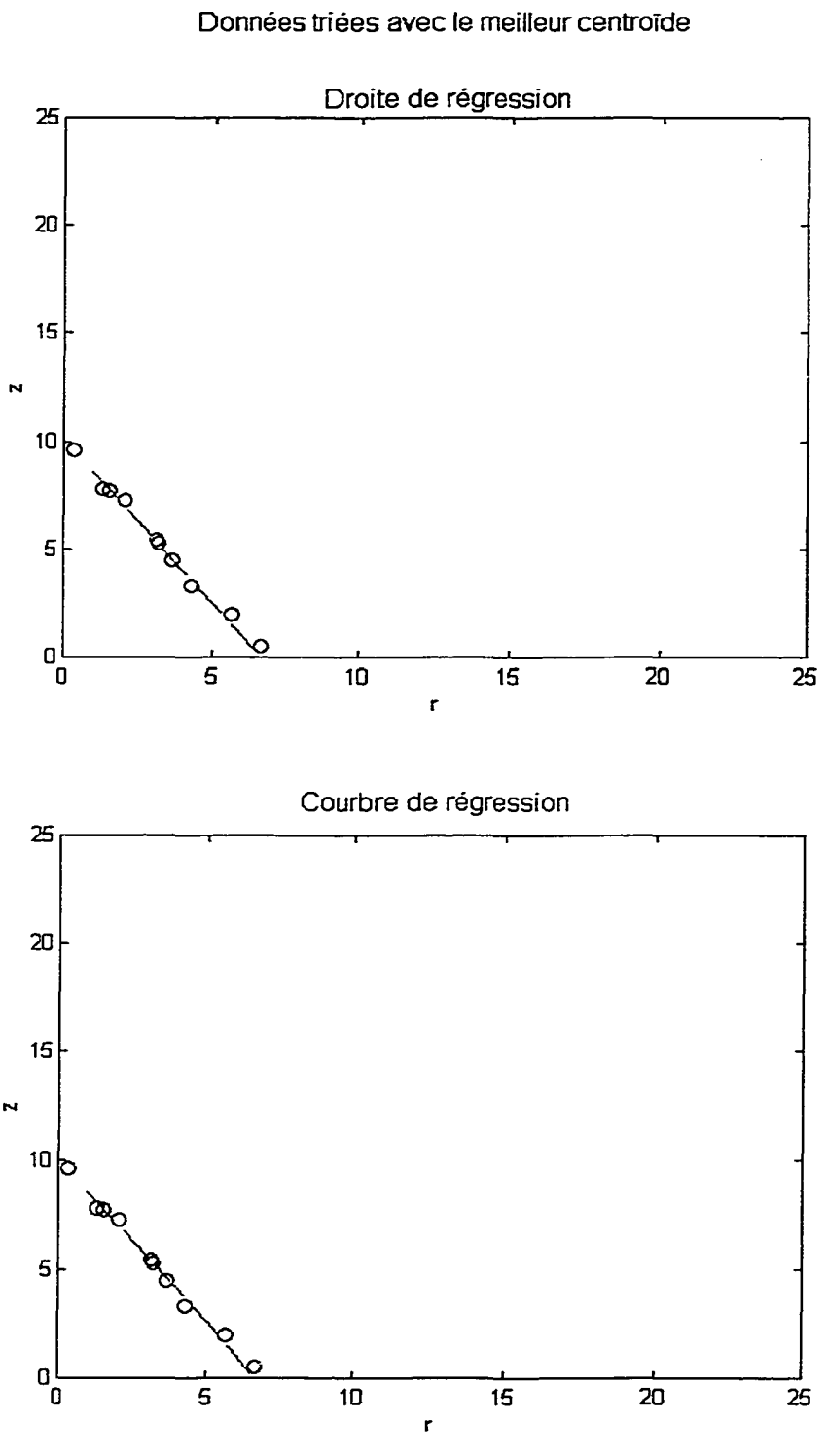


Figure 33 – Résultat des extrapolations pour l'arbre 17

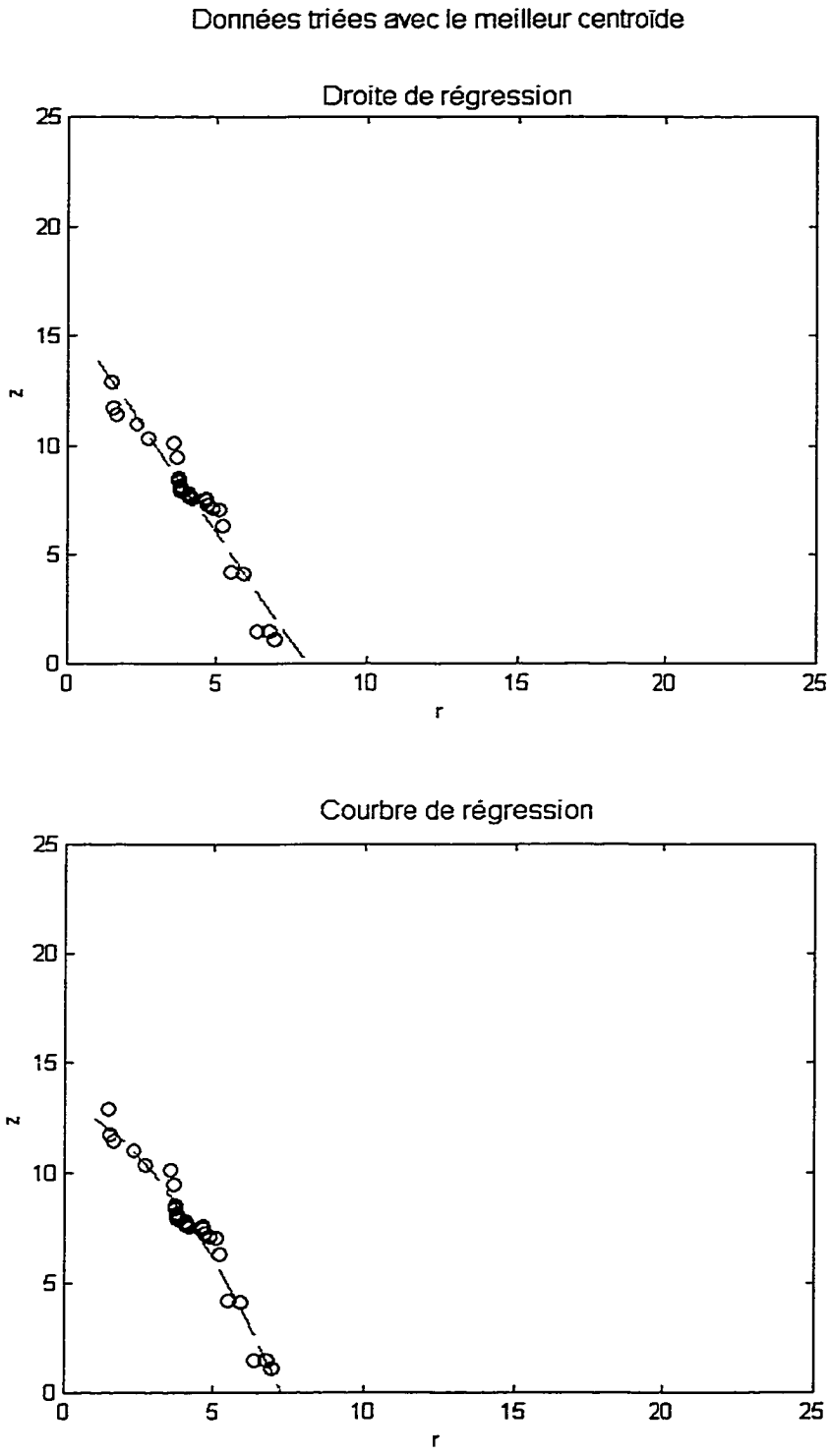


Figure 34 – Résultat des extrapolations pour l'arbre 18

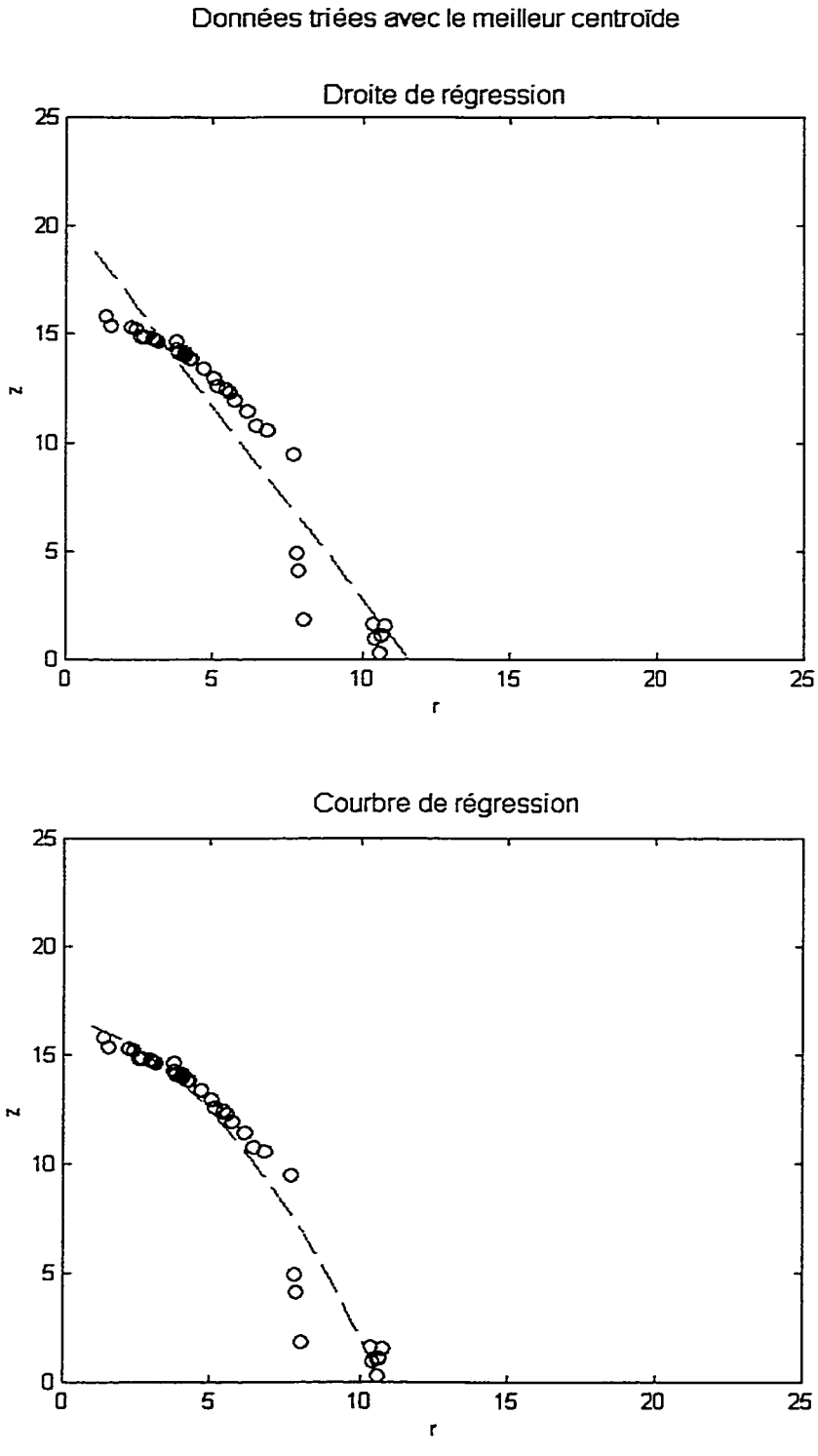


Figure 35 – Résultat des extrapolations pour l'arbre 19

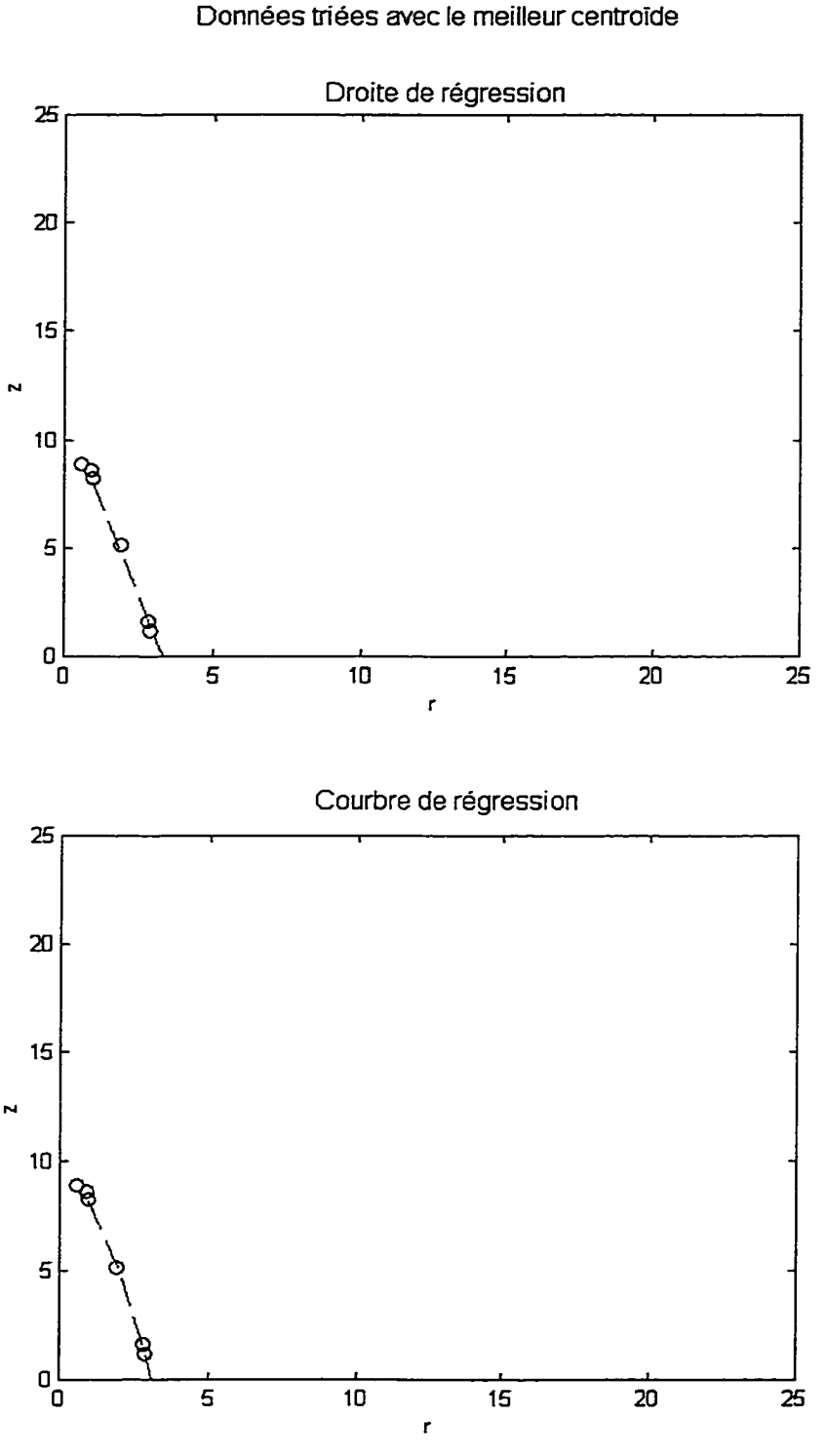


Figure 36 – Résultat des extrapolations pour l'arbre 34

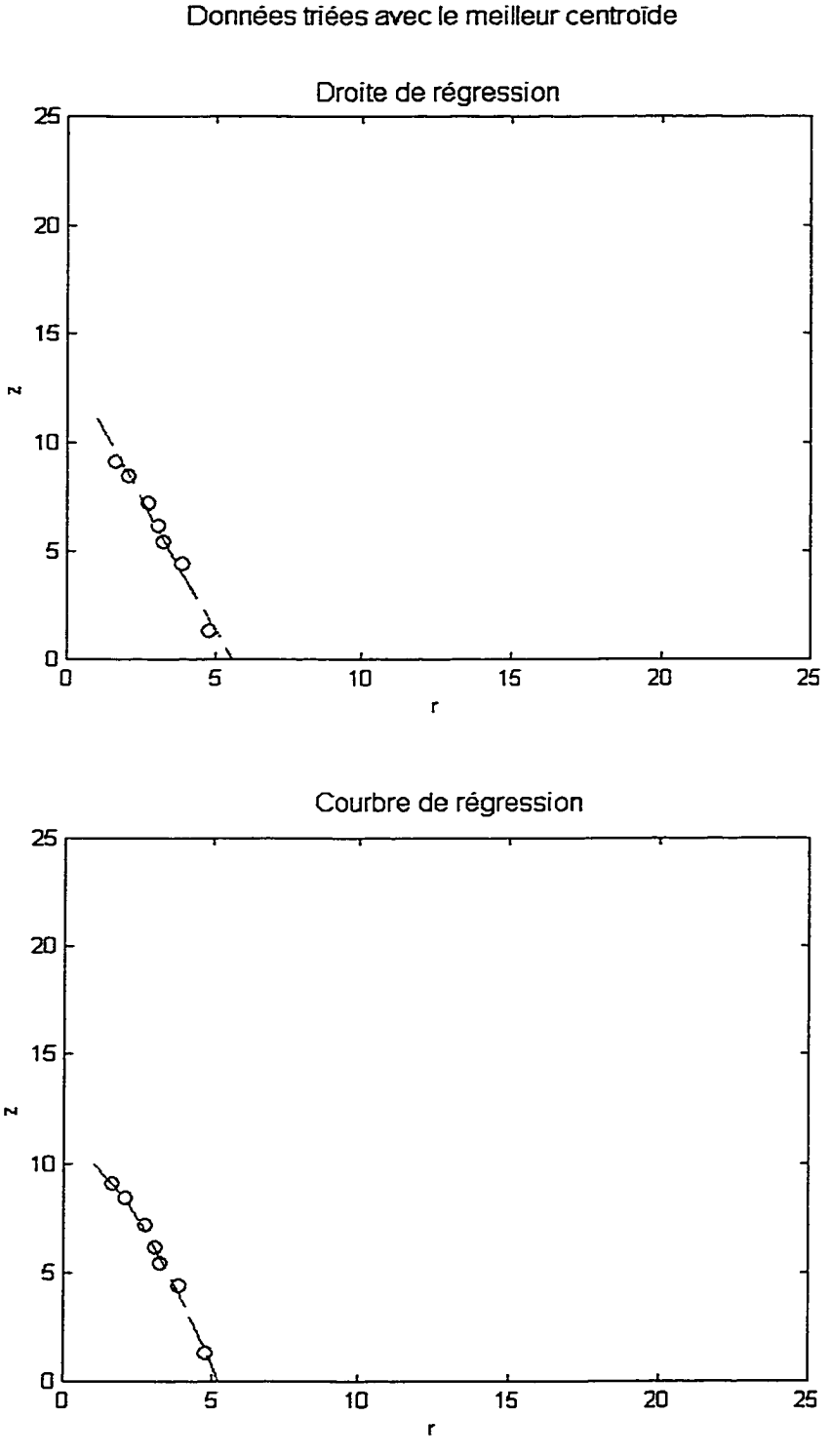


Figure 37 – Résultat des extrapolations pour l'arbre 35

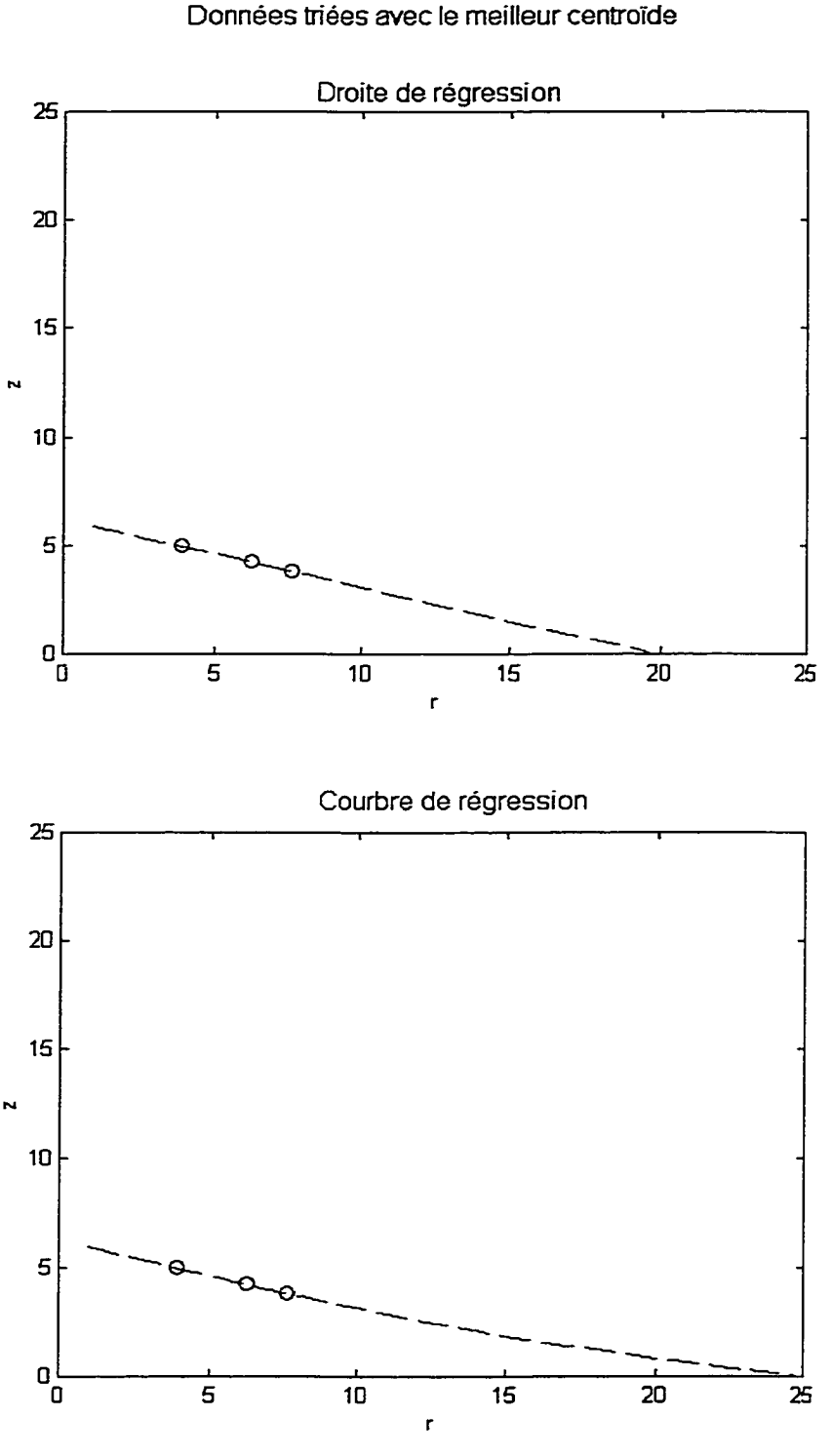


Figure 38 – Résultat des extrapolations pour l'arbre 36

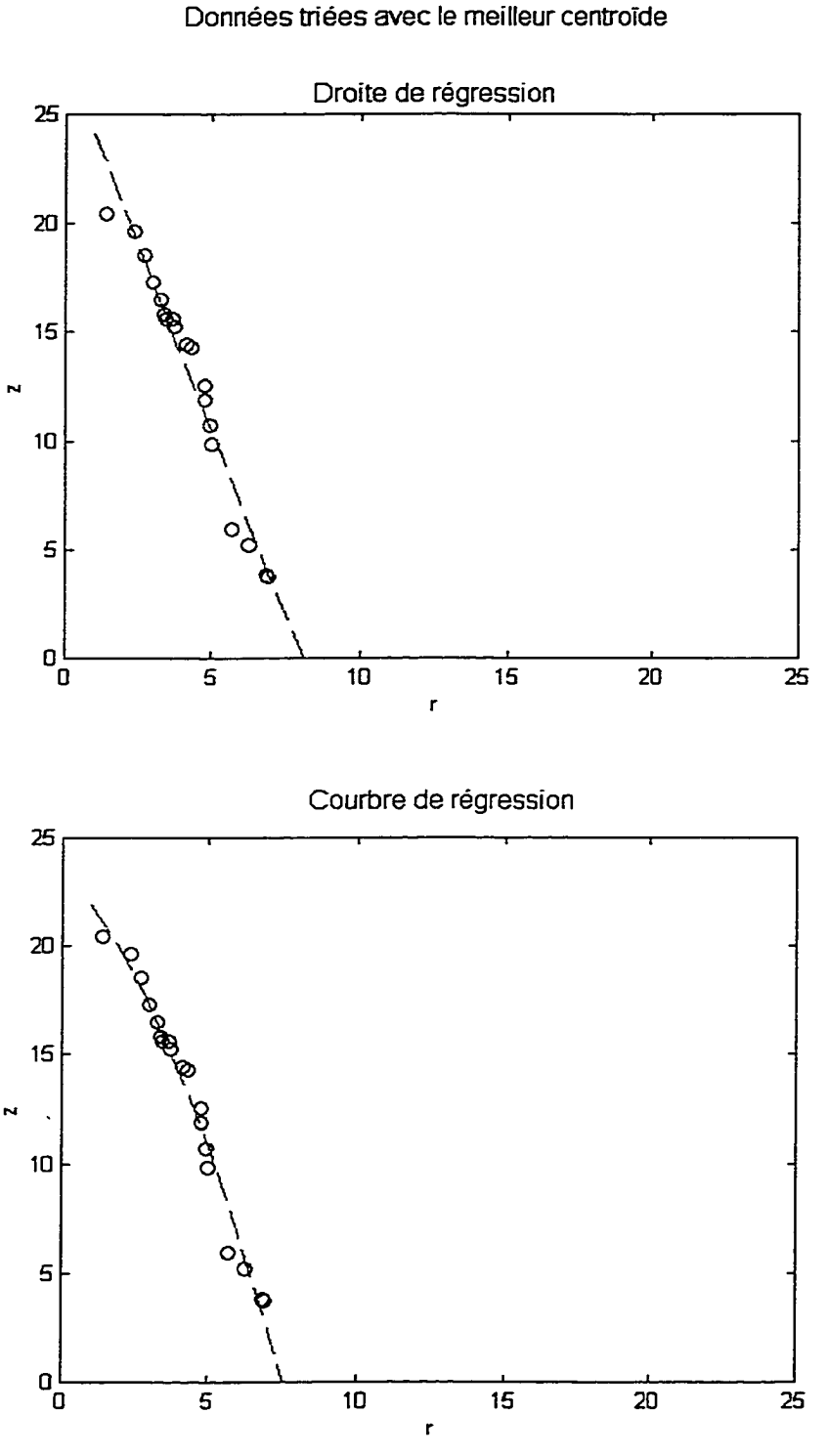


Figure 39 – Résultat des extrapolations pour l'arbre 42

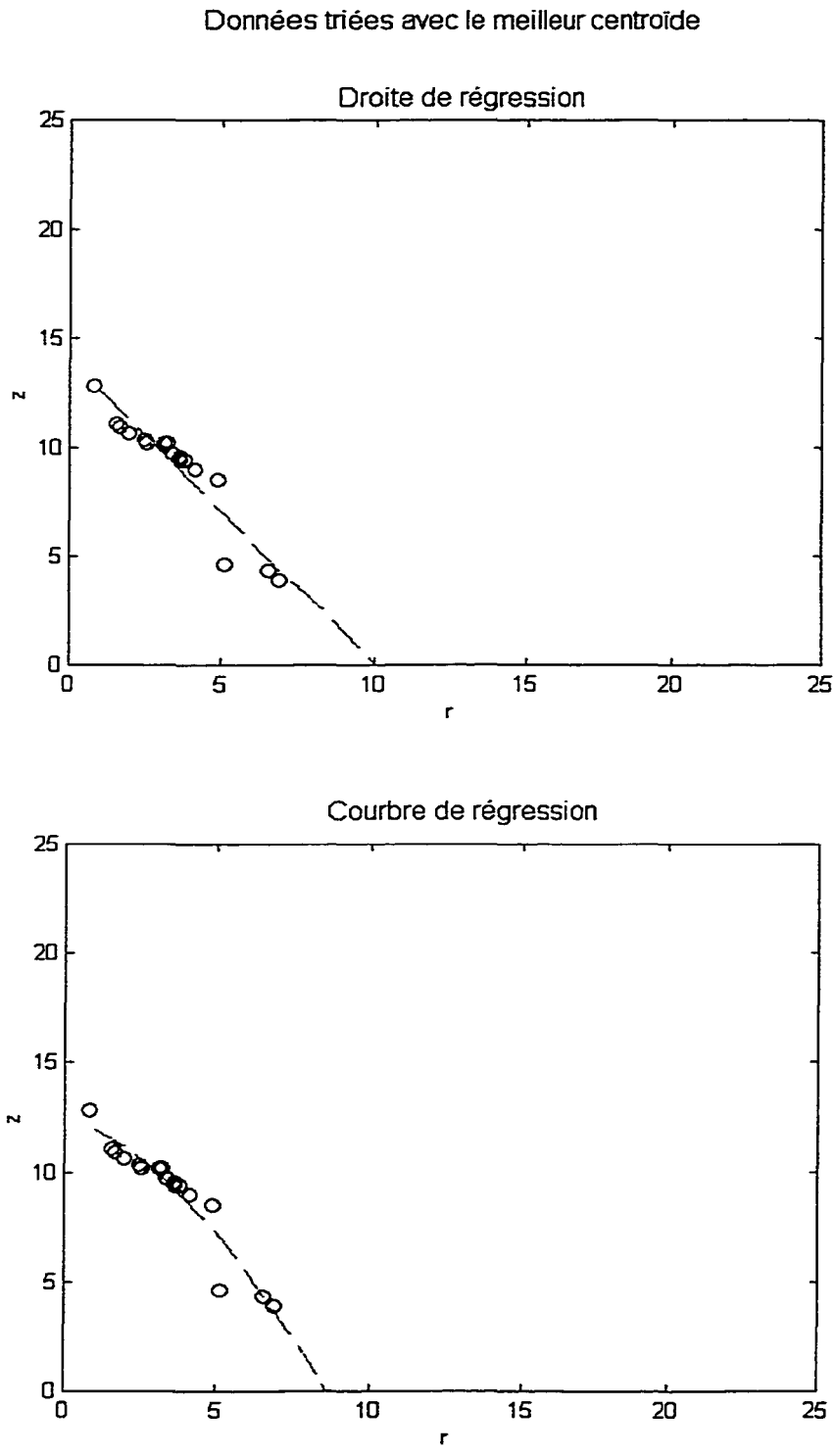


Figure 40 – Résultat des extrapolations pour l'arbre 43

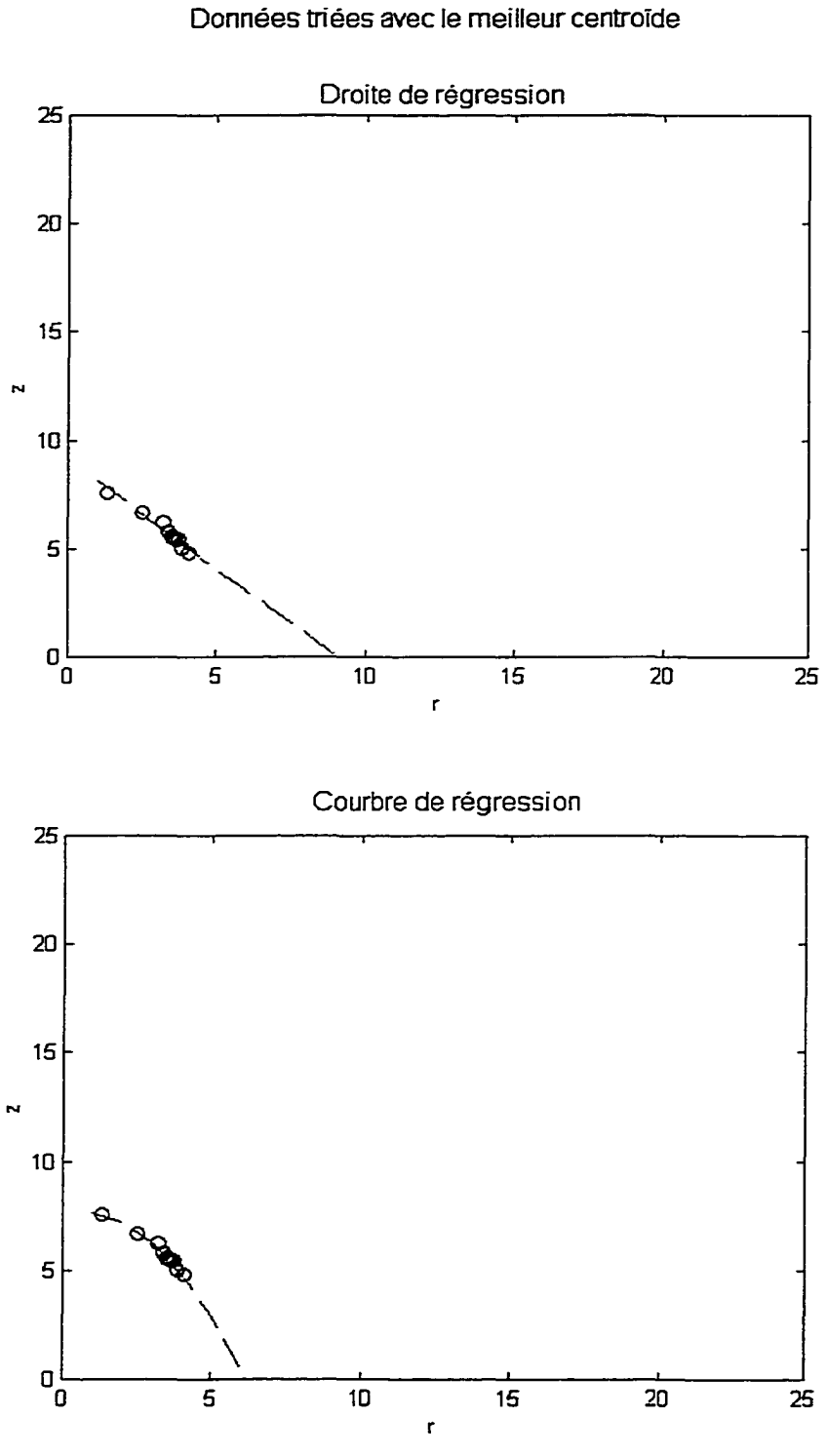


Figure 41 – Résultat des extrapolations pour l'arbre 54

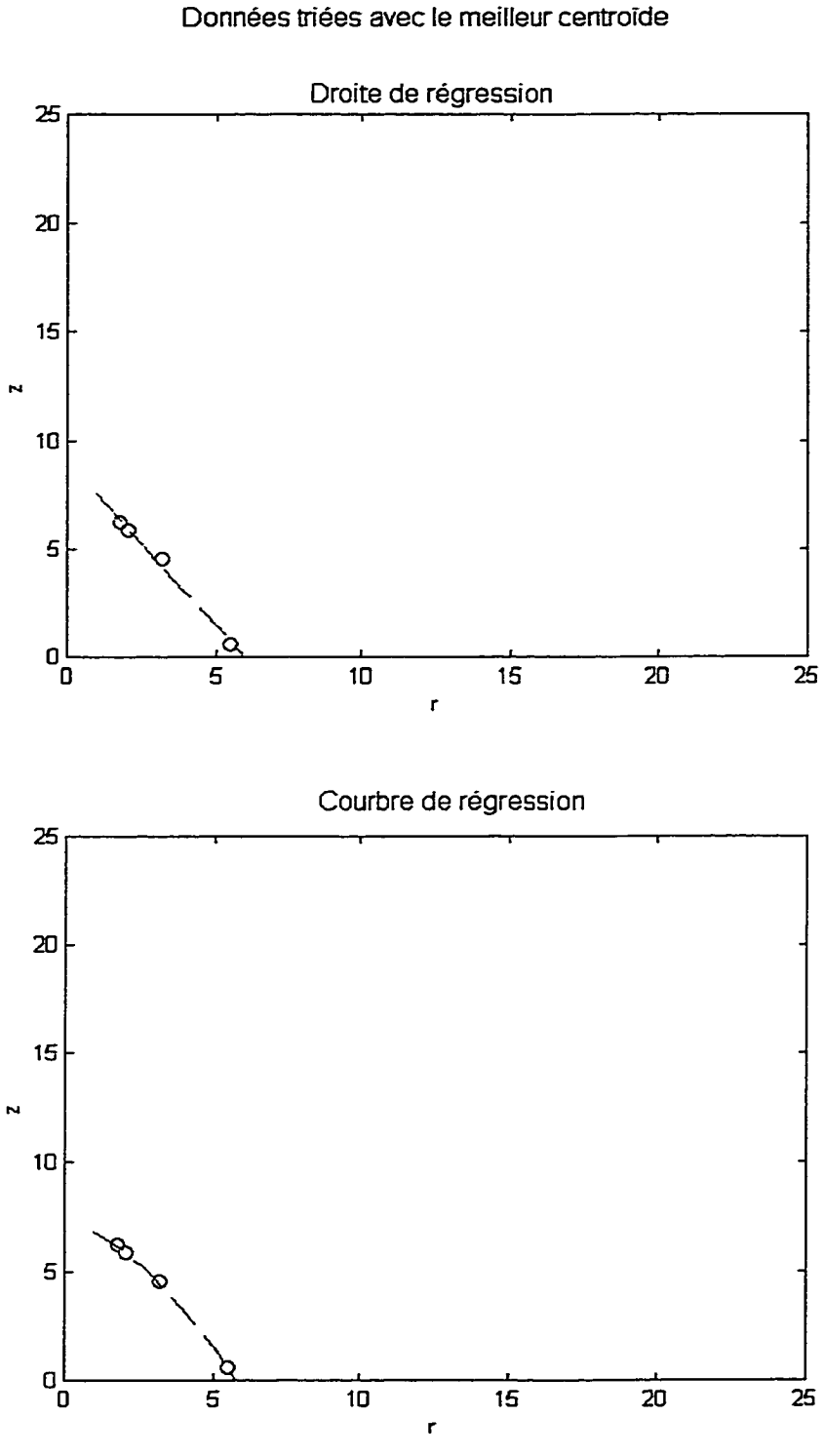


Figure 42 – Résultat des extrapolations pour l'arbre 56

DISCUSSION ET INTERPRÉTATION DU TRAVAIL

Ce projet comporte trois étapes déterminantes dont nous discuterons en détail et qui engendrent des résultats intermédiaires influençant les étapes subséquentes du projet ainsi que les résultats obtenus.

Premièrement, l'étape du filtrage, qui détermine l'emplacement du contour des couronnes d'arbres, a un effet majeur sur le reste des opérations du projet. En effet, toutes les autres étapes découlent de celle-ci et la qualité des résultats finaux en dépend directement. Cette étape pourrait encore être améliorée, il subsiste certains arbres liés entre eux, ce qui constitue le principal problème résiduel de cette étape. Pour améliorer le filtrage, il faudrait être capable de séparer les arbres selon leur contour, mais aussi selon leur hauteur moyenne, ce qui pourrait faire l'objet de projets futurs. De cette façon, deux arbres adjacents avec des hauteurs moyennes différentes devraient déterminer chacun un contour distinct, ce qui n'est pas toujours le cas présentement. Plusieurs couples d'arbres (un grand avec un petit) ont été fusionnés par le LoG, donnant une mauvaise position du centre trouvée et par conséquent un mauvais calcul de la hauteur de l'arbre.

En effet, la position initiale du centroïde de l'arbre, utilisée comme point de départ pour les calculs de régression est directement liée au résultat du LoG, ce qui pose le deuxième problème. Si le LoG assemble deux arbres et que l'algorithme de sélection d'arbres valides laisse passer ces arbres assemblés, on se retrouve avec un problème assez important où le centroïde de l'arbre recherché (habituellement le plus haut des deux arbres) est déplacé vers le deuxième arbre (plus bas). De plus, dans ce cas, nous utilisons des données appartenant à deux arbres pour calculer la valeur d'un seul, ce qui pose problème.

La troisième étape, amenant un autre problème potentiel, est le calcul de la hauteur de l'arbre lui-même. Si on avait le centroïde réel de l'arbre ainsi que des données exactes de position et de hauteur de plusieurs points laser sur la couronne de l'arbre, la droite ou la courbe de régression devrait nous donner une valeur très près de la hauteur réelle de l'arbre. Cela, parce que les conifères sont généralement de forme conique et les feuillus de forme sphériques. En théorie, l'ajustement de droites et de courbes dans le profil des arbres, sur des données parfaitement acquises, devrait donner d'excellents résultats quant à la hauteur des arbres. Les erreurs et les incertitudes sur les calculs sont ainsi dues aux erreurs de mesures de l'altimètre laser, aux erreurs de positionnement des points laser sur la couronne, et à l'erreur de positionnement du centroïde dans la couronne. De plus, il faut ajouter les cas problèmes où deux arbres sont assemblés en un seul dans un contour. Une amélioration possible des résultats de la droite de régression sur les données non-triées aurait pu consister à faire passer une droite par l'enveloppe des points. Cependant, rien n'indique que le centroïde calculé pour chaque arbre est le bon et que les erreurs sur les données ne sont pas trop grandes pour donner un mauvais calcul.

On peut donc se demander si l'objectif de calculer la hauteur en dedans du mètre était vraiment réaliste quand on sait que les erreurs de mesure sur le terrain peuvent facilement être d'un à trois mètres entre deux mesures du même arbre selon deux points différents et aussi quand les données d'altimétrie laser sont elles-même possiblement erronées de 70 cm dans le plan horizontal.

L'étude a quand même fourni une base de comparaison avec la méthode de régression fournie par St-Onge [9] qui donne des résultats valables compte tenu de la simplicité de la formule. Par contre, le travail présenté ici est d'une grande utilité, car l'identification des contours des couronnes avec le LoG n'avait

pas encore été explorée et donne des résultats des plus prometteurs. Même si on devait repenser la méthode proposée ici de calcul des hauteur ou que l'on utilisait celle de St-Onge, parce que plus simple, nous pourrions tout de même utiliser les trois quarts du travail effectué dans le cadre de ce projet pour aller extraire automatiquement le pixel le plus haut de la couronne afin de l'utiliser dans la formule de prédiction de hauteur proposée par St-Onge.

CONCLUSION

Avant de pouvoir penser à une mise en œuvre commerciale, les méthodes proposées demandent à être perfectionnées. Par contre elles sont très prometteuses dans le sens de l'automatisation du processus et du même coup de l'économie de temps et d'argent pour déterminer la hauteur des arbres individuels et la biomasse de forêts.

Les principales étapes de ce projet consistaient à faire d'abord la délimitation des couronnes des arbres à l'aide du LoG. Par la suite, nous avons effectué l'identification des arbres valides et du même coup du centroïde de ces arbres. Ensuite nous nous sommes attardé au problème de calcul de hauteurs qui s'est avéré peu concluant face aux attentes que nous avions au départ (nous n'avons d'ailleurs pas présenté toutes les méthodes de calcul de hauteurs essayées dans le cadre de ce travail).

Nous pouvons dire que nous avons atteint notre objectif de délimitation automatique des couronnes d'arbre, nous avons aussi réussi l'extraction automatique du point laser le plus haut d'une couronne d'arbre. Globalement, nous pouvons conclure que les résultats sont prometteurs quant aux perspectives futures d'automatisation des mesures en forêts boréales.

RÉFÉRENCES BIBLIOGRAPHIQUES

[1] Ackermann, F., 1999. *Airborne laser scanning - present and future expectations*, ISPRS Journal of Photogrammetry and Remote Sensing, 54, 64-67.

[2] Blair, J. B., Rabine, D. L., and Hofton, M. A., 1999. *The laser vegetation imaging sensor: a medium-altitude, digitisation-only, airborne laser altimeter for mapping vegetation and topography*, ISPRS Journal of Photogrammetry and Remote Sensing, 54, 115-122.

[3] Naesset, E., 1997. *Determination of mean tree height of forest stands using airborne laser scanner data*, ISPRS Journal of Photogrammetry and Remote Sensing, 52, 49-56.

[4] Flood, M. and Gutelius, B., 1997. *Commercial Implications of Topographic Terrain Mapping Using Scanning Airborne Laser Radar*, Photogrammetric Engineering and Remote Sensing, 63, 327-329.

[5] Nelson, R, Krabill, W., and Maclean, G. A., 1984. *Determining forest canopy characteristics using airbone laser data*, Remote Sensing of Environment , 15, 201-212.

[6] Nelson, R., Krabill, W., and Tonelli, J., 1988. *Estimating forest biomass and volume using airborne laser data*, Remote Sensing of Environment , 24, 247-267.

[7] Nilsson, M., 1996. *Estimation of tree heights and stand volume using an airborne lidar system*, Remote Sensing of Environment , 56, 1-7.

[8] Nelson, R., Oderwald, R., and Gregoire, T. G., 1997. *Separating the ground and airborne laser sampling phases to estimate tropical forest basal area*, volume and biomass, Remote Sensing of Environment , 60, 311-326.

[9] St-Onge, B. A., 1999. *Estimating individual tree heights of the boreal forest using airborne laser altimetry and digital videography*, ISPRS Workshop on "Mapping surface structure and topography by airborne and spaceborne lasers", Commission III, Working Group 3 , LaJolla, California, 9-11 November 1999.

[10] Gougeon, F. A., *Automatic individual tree crown delineation using a valley-following algorithm and a rule-based system*, International Forum on automated

interpretation of high spatial resolution digital imagery for forestry, février 1998, Pacific forestry center, Victoria, Colombie-Britanique, pp. 11-23.

[11] Polloc, R., *Individual tree recognition based on a synthetic tree crown image model*, International Forum on automated interpretation of high spatial resolution digital imagery for forestry, février 1998, Pacific forestry center, Victoria, Colombie-Britanique, pp. 25-34.

[12] Brandtberg, T., Walter F., *An algorithm for delineation of individual tree crown in high spatial resolution aerial images using curved edge segments at multiple scales*, International Forum on automated interpretation of high spatial resolution digital imagery for forestry, février 1998, Pacific forestry center, Victoria, Colombie-Britanique, pp. 41-54.

[13] Dickinson, S. J., Metaxas, D., Pentland, A., *Role of model-based segmentation in the recovery of volumetric parts from range data*, IEEE Transactions on Pattern Analysis and Machine Intelligence, v 19 n 3, Mar 1997, 259-267

[14] Larsen, M., *Finding an optimal match window for spruce top detection based on an optical tree model*, International Forum on automated interpretation of high spatial resolution digital imagery for forestry, février 1998, Pacific forestry center, Victoria, Colombie-Britanique, pp. 55-66.

[15] Marr, D., Hildreth, E. C., *Theory of edge detection*, Proc. R. Soc. Lond. B, 207:187-217, 1980.

[16] Leroux, Pierre, *Algèbre Linéaire une approche matricielle*, Modulo Éditeur, Mont-Royal, 1983, 500 p.

[17] Gonzalez R. G., Richard W. E., *Digital Image Processing*, Addison-Wesley Publishing Company, 1992, 716 p.

ANNEXE 1

Programme d'inversion d'octets

Programme en langage C, inversant l'ordre des octets de nombres réels 32 bits, dans des fichiers binaires d'images.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void main(int argc, char * argv[])
{
 FILE * flin, *flout, *f1, *f2;
 char datain[4], dataout[4];
 float data[1],zero[1],datasmall[1];
 int i,j,numread=0;

 float *fdata;

 if (argc<3)
 {
  printf("Ce programme necessite 2 arguments : fichier_entree fichier_sortie\n");
  exit(0);
 }
 zero[0]=0.0;

/*Ici j'ecris les bytes inverses*/
 flin = fopen(argv[1],"rb");
 flout = fopen(argv[2],"wb");
 fseek(flin,0,SEEK_SET);
 i=0;
 while(i<706400)
 {
  i++;
  strcpy(datain,"");
  strcpy(dataout,"");

   fread(datain,4,1,flin);  /*format IEEE 4 byte real ici => float*/

  fdata=&datain;
   if((fdata[0]<10000)&& (fdata[0]>-1000))
  {
   dataout[0]=datain[3];
   dataout[1]=datain[2];
   dataout[2]=datain[1];
   dataout[3]=datain[0];
   fwrite(dataout,4,1,flout);
  }
  else
  {
   fwrite(zero,4,1,flout);
  }

 }
 fclose(flin);
 fclose(flout);
}
```

ANNEXE 2

Programme de préformattage des données

Programme en langage C préformattant les données brutes laser 3D. Soustraction du point de référence (623500, 539109.5), afin d'obtenir des coordonnées référencées à (0,0).

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void main(int argc, char * argv[])
{
 FILE * flin, *flout;
 char datain[100],*data1,*data2,*data3;
 double data1f,data2f,data3f,data2max=0,dx,dy;
 int x,y;
 long int i;

 if (argc<3)
 {
  printf("Ce programme necessite 2 arguments : fichier_entree fichier_sortie\n");
  exit(0);
 }

 flin = fopen(argv[1],"rb");
 flout = fopen(argv[2],"wb");
 fseek(flin,0,SEEK_SET);
 fseek(flout,0,SEEK_SET);

 while(fgets(datain,100,flin)!=NULL)
 {
  data1=strtok(datain,",");
  data2=strtok(NULL,",");
  data3=strtok(NULL,",");
  data1f=atof(data1);
  data2f=atof(data2);
  data3f=atof(data3);
  data1f=data1f-623500;
  data2f=data2f-5369109.5;

  x=(int)(data1f*2);
  y=(int)(data2f*2);
  dx=(data1f*2)-x;
  dy=(data2f*2)-y;

  fprintf(flout,"%d,",x);
  fprintf(flout,"%d,",y);
  fprintf(flout,"%.2f,",dx);
  fprintf(flout,"%.2f,",dy);
  fprintf(flout,"%.2f\n",data3f);
  }
 fclose(flin);
 fclose(flout);
}
```

ANNEXE 3

Tâche KBV « gentokens »

Tâche KBV, programmée en C, produisant les éléments symboliques (1 point par point laser) à partir du fichier créé avec le programme de l'annexe 2.

```
#include <KBVstd.h>
#include <Tcb.h>
#include <laf.h>
#include <lsr.h>
#include <lsrFuns.h>
#include <ErrStatus.h>

ERR_MODULE("gentokens");

gentokens(TCBPTR tcbptr)
{

 TCB_DOCUMENTATION ("Genere les tokens des donnees 3D.") ;

 TCB_PARAM_IN (infile, KBV_STRING,
            (Documentation: "Fichier entree",
          Default     : "" ));
 TCB_PARAM_IN (outfile, KBV_STRING,
            (Documentation: "Fichier sortie",
          Default     : "" ));

 int Tokenindex,i=0;
 KBV_STRING Tokensetname = "Tproj01";
 KBV_FPOINT fpoint;
 FILE * flin;
 char datain[100],*data1,*data2,*data3,*data4,*data5;
 short data1f,data2f;
 double data3f,data4f,data5f;

 ERR_FUNCTION ("gentokens");

 lsrDefineTokenset("Tproj01");
 lsrAddFeature("Tproj01", "BITPLANE", KBV_FPOINT_DT);
 lsrAddFeature("Tproj01", "DELTAX", KBV_FLOAT_DT);
 lsrAddFeature("Tproj01", "DELTAY", KBV_FLOAT_DT);
 lsrAddFeature("Tproj01", "HAUTEUR", KBV_FLOAT_DT);

 flin = fopen(infile,"rb");
 fseek(flin,0,SEEK_SET);

 while(fgets(datain,90,flin)!=NULL)
 {

  data1=strtok(datain,",");
  data2=strtok(NULL,",");
  data3=strtok(NULL,",");
  data4=strtok(NULL,",");
  data5=strtok(NULL,"\n");

  data1f=(short)atoi(data1);
  data2f=(short)atoi(data2);
  data3f=atof(data3);
  data4f=atof(data4);
  data5f=atof(data5);

  fpoint.x=data1f;
  fpoint.y=data2f;

  lsrDefineTokenNI("Tproj01", i);
```

```
    IsrPutFeatureNIN("Tproj01",i,"BITPLANE",KBV_FPOINT_DT,fpoint);
    IsrPutFeatureNIN("Tproj01",i,"DELTAX",KBV_FLOAT_DT,data3f);
    IsrPutFeatureNIN("Tproj01",i,"DELTAY",KBV_FLOAT_DT,data4f);
    IsrPutFeatureNIN("Tproj01",i,"HAUTEUR",KBV_FLOAT_DT,data5f);

    i++;

  }
  fclose(flin);

  IsrWriteTokenset(outfile,"Tproj01");

  ERR_RETURN KBV_SUCCESS;

  return;

}
```

ANNEXE 4

Sous-images traitées avec les 5 types de filtrages

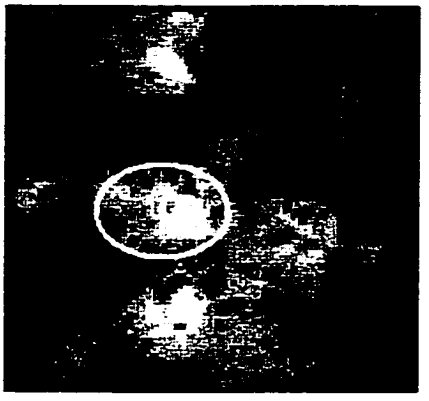
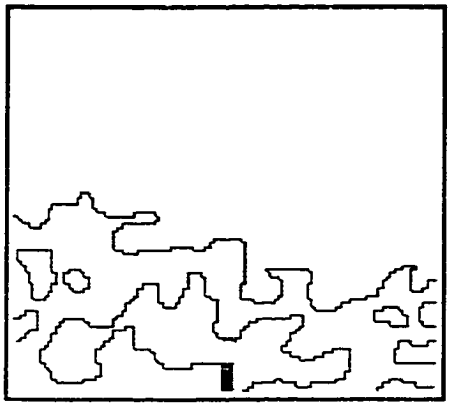
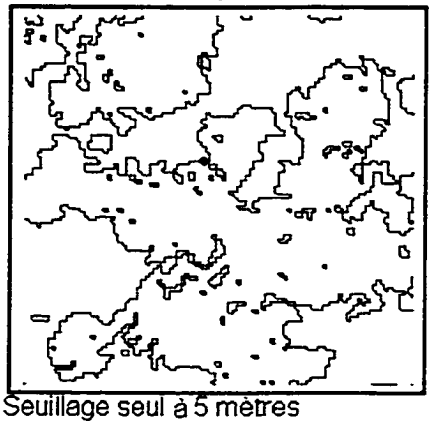


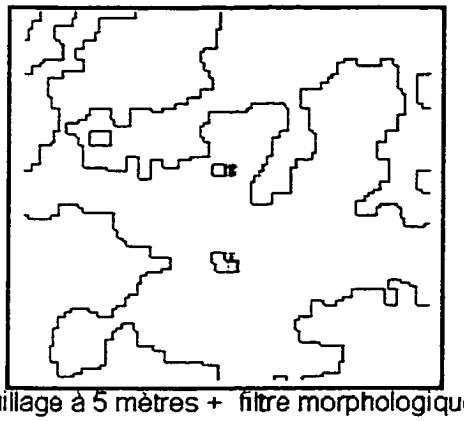
image 1



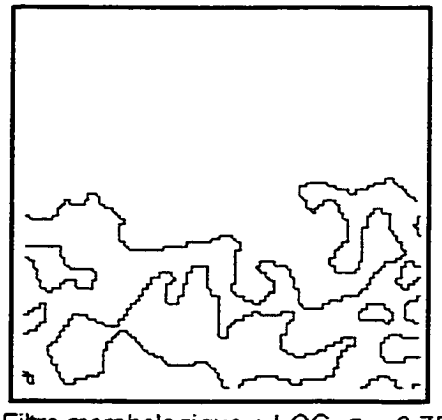
LoG $\sigma = 3{,}75$



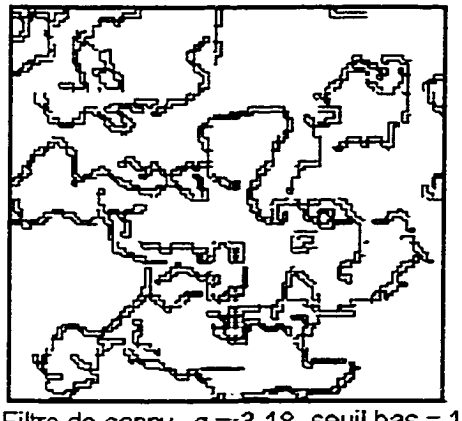
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique

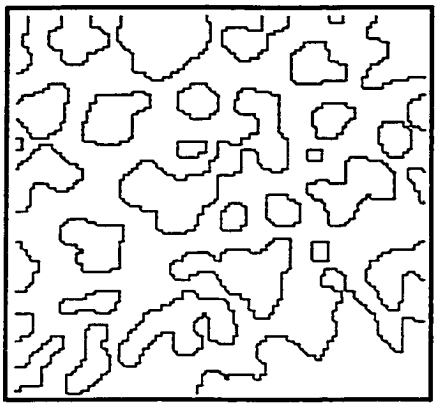


Filtre morphologique + LOG $\sigma = 3{,}75$

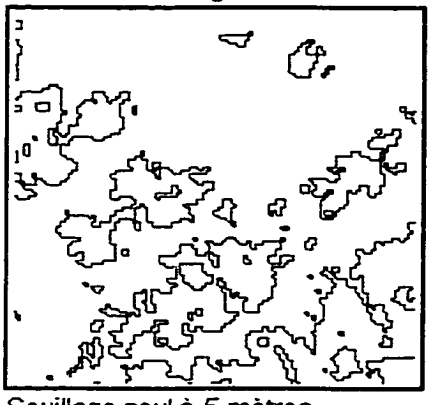


Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

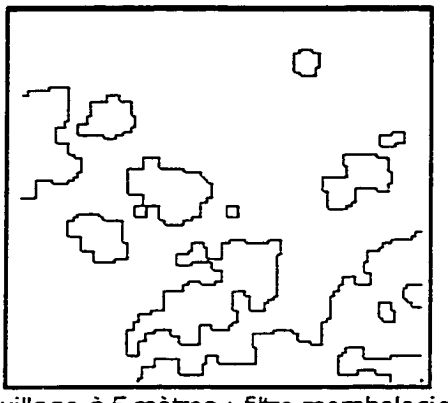
image 3



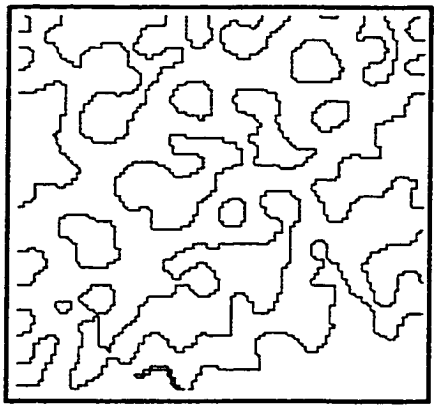
LoG $\sigma = 3{,}75$



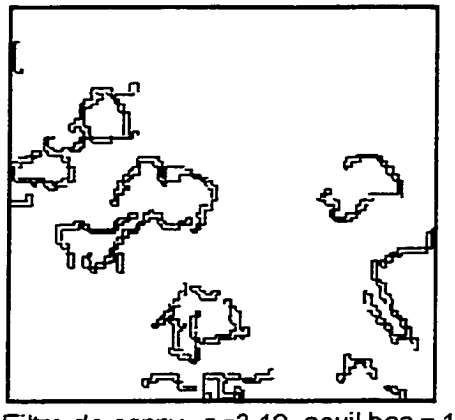
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique



Filtre morphologique + LOG $\sigma = 3{,}75$



Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

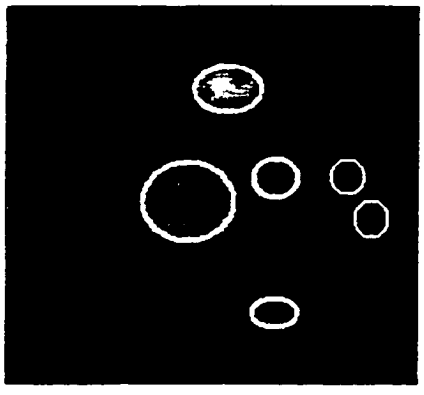
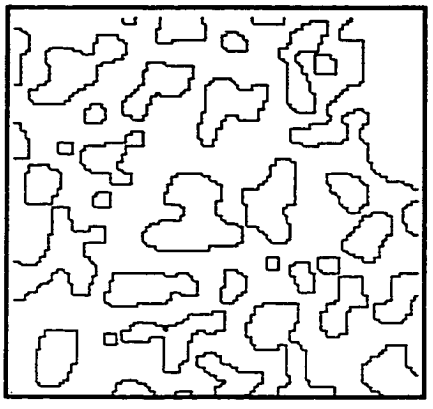
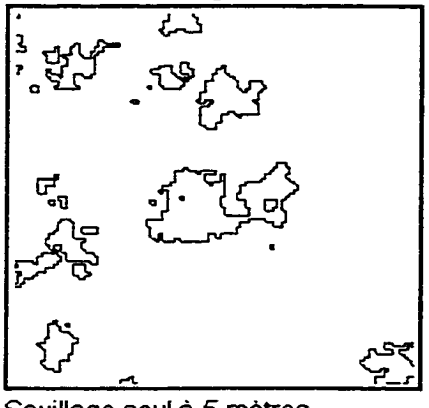


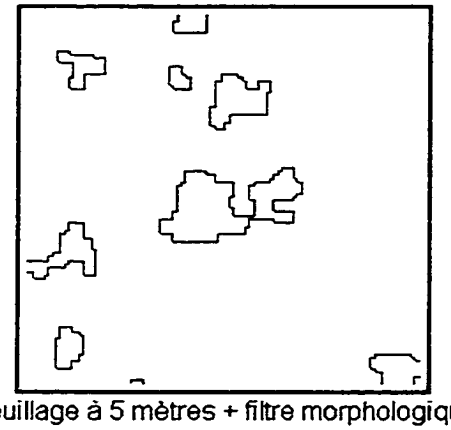
image 4



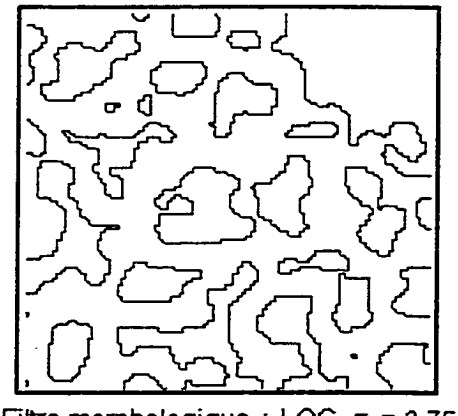
LoG σ = 3,75



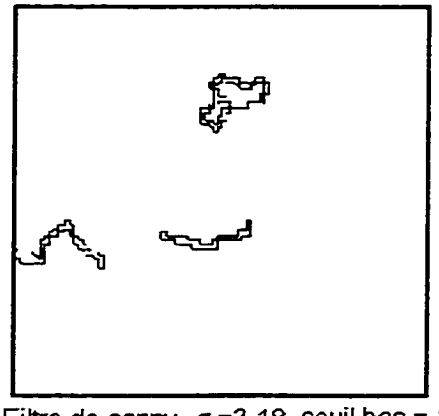
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique

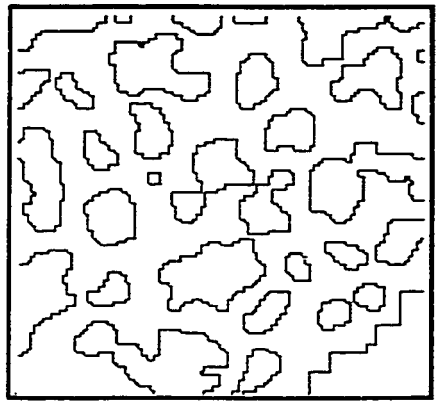


Filtre morphologique + LOG σ = 3,75

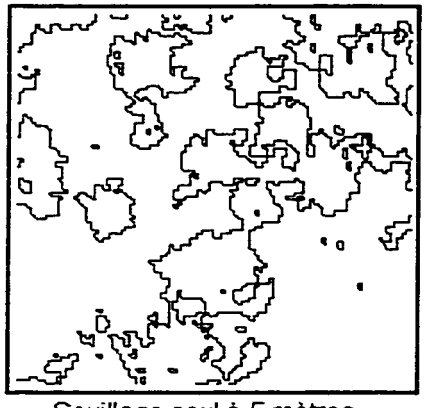


Filtre de canny σ=3.18 seuil bas = 1
seuil haut = 2

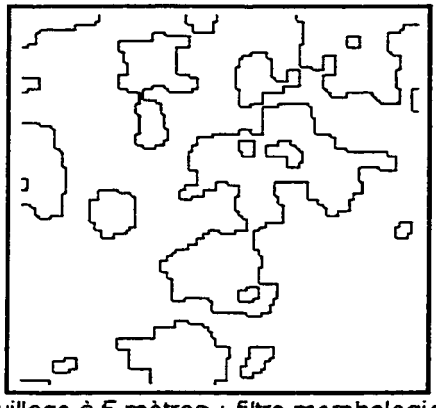
image 5



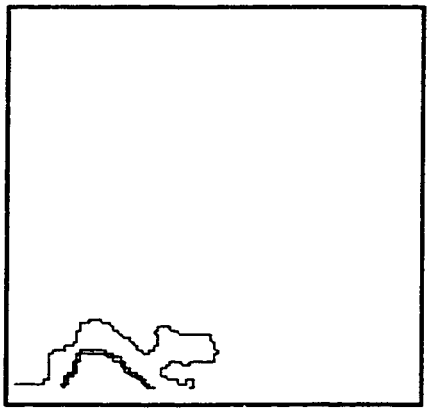
LoG $\sigma = 3,75$



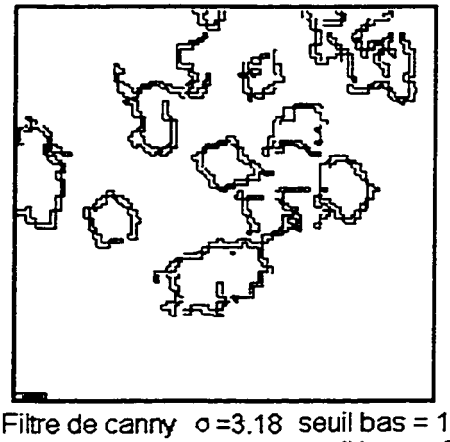
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique

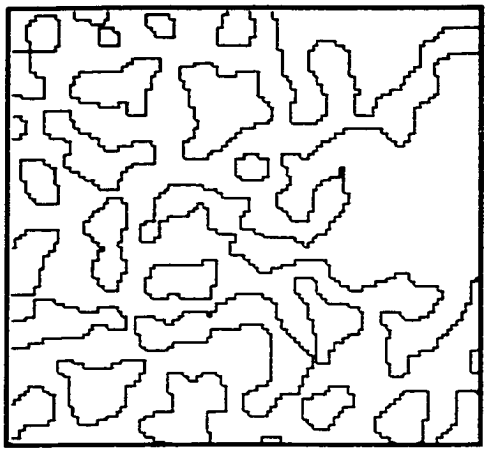


Filtre morphologique + LOG $\sigma = 3,75$

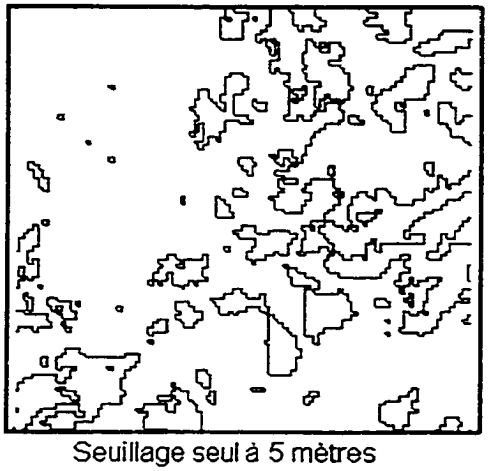


Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

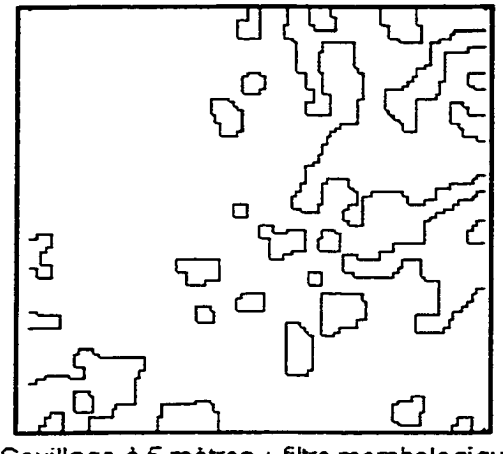
image 6



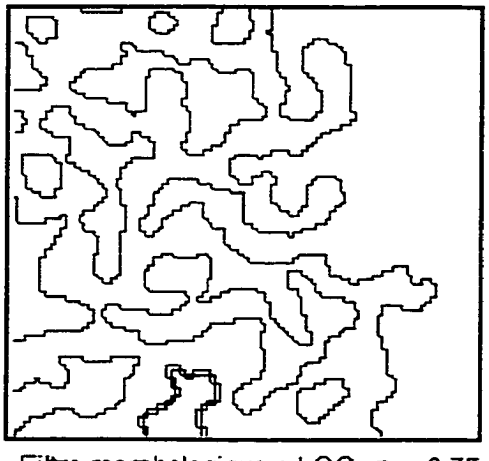
LoG $\sigma = 3,75$



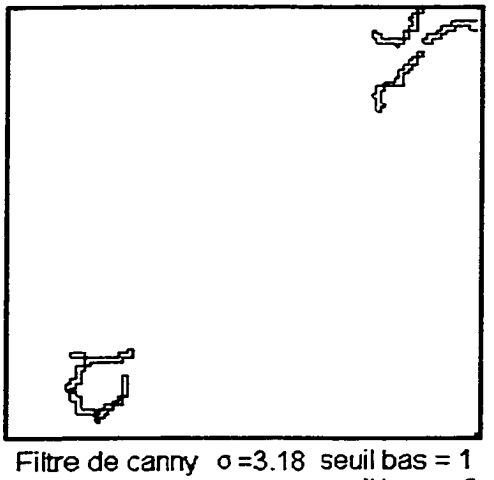
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique



Filtre morphologique + LOG $\sigma = 3,75$



Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

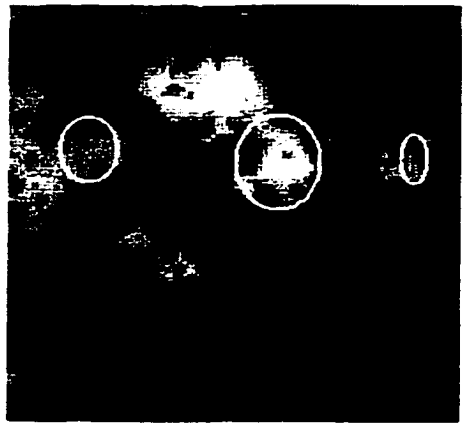
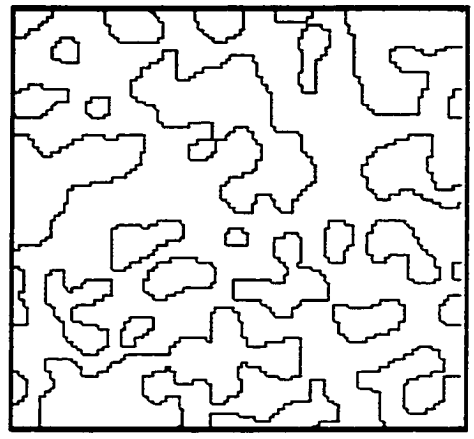
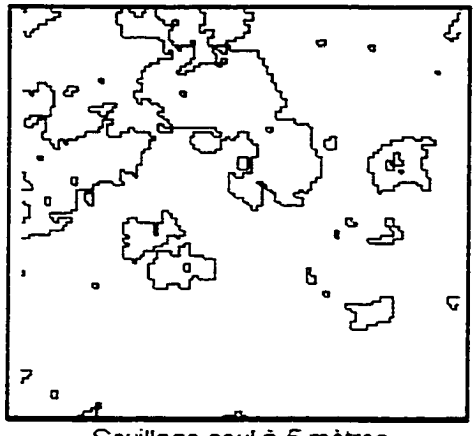


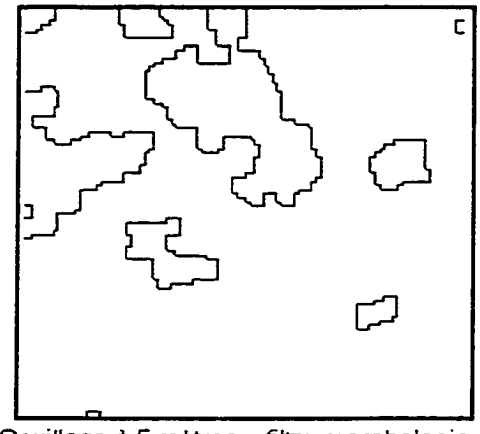
image 7



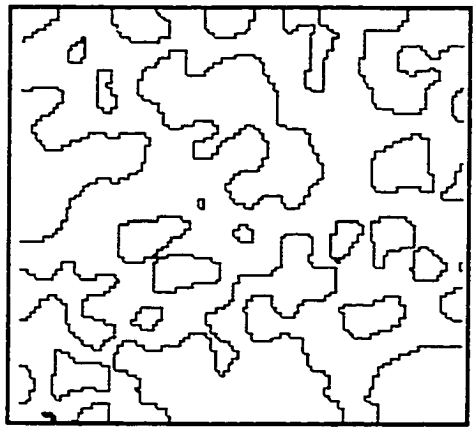
LoG $\sigma = 3,75$



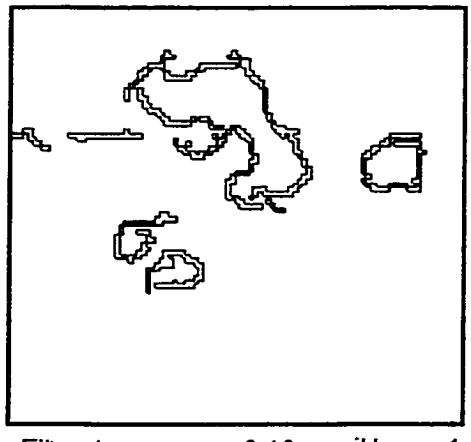
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique



Filtre morphologique + LOG $\sigma = 3,75$



Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

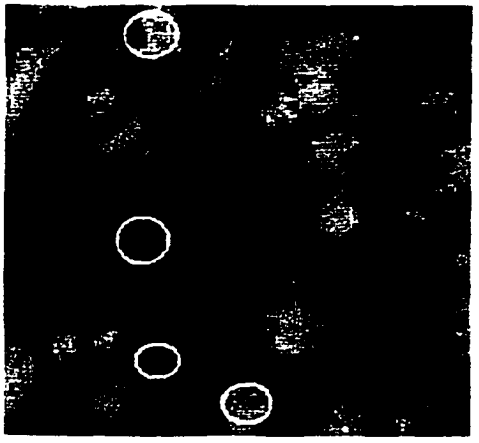
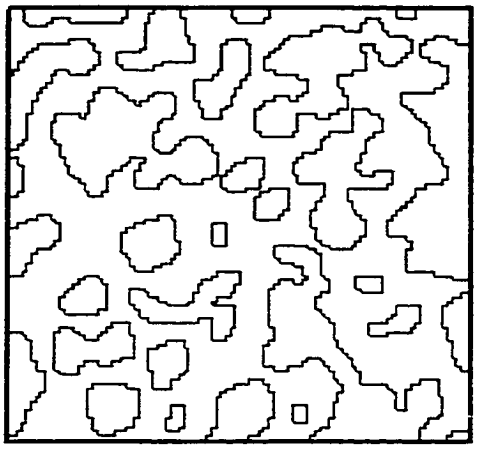
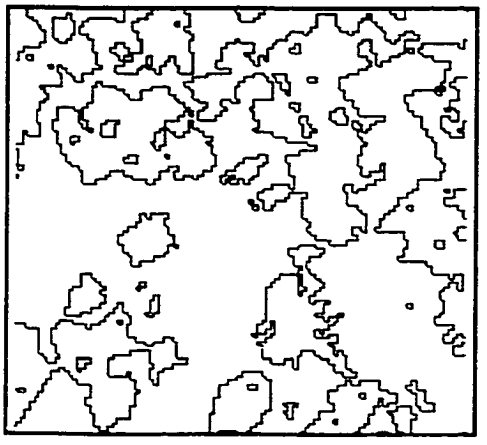


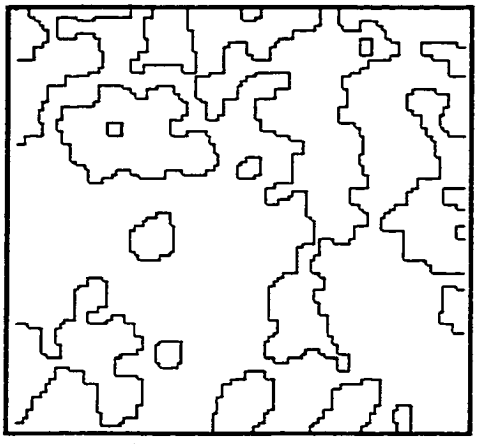
image 8



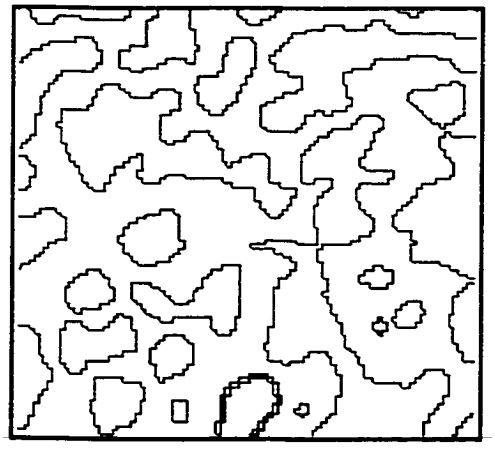
LoG $\sigma = 3,75$



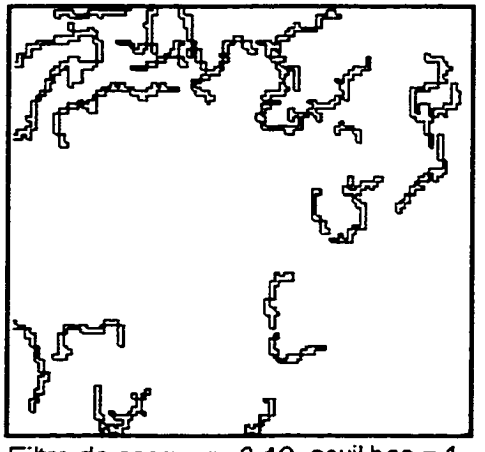
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique



Filtre morphologique + LOG $\sigma = 3,75$



Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

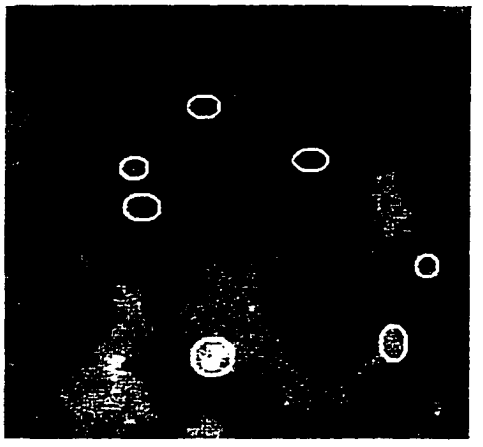
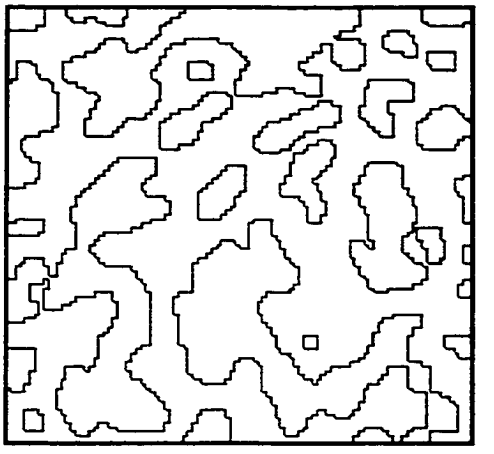
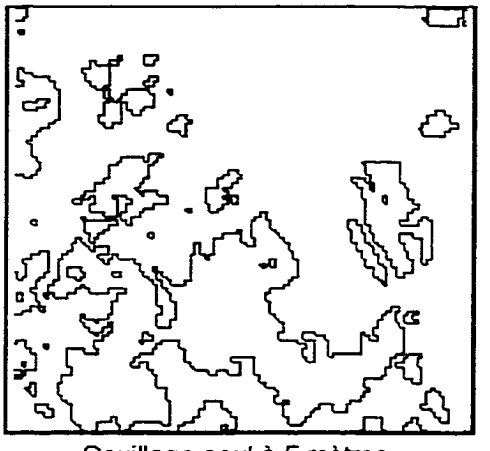


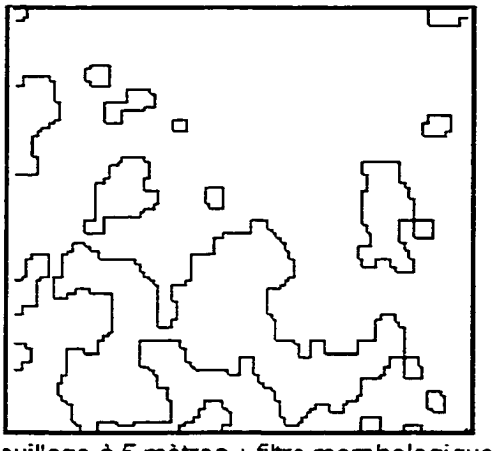
image 9



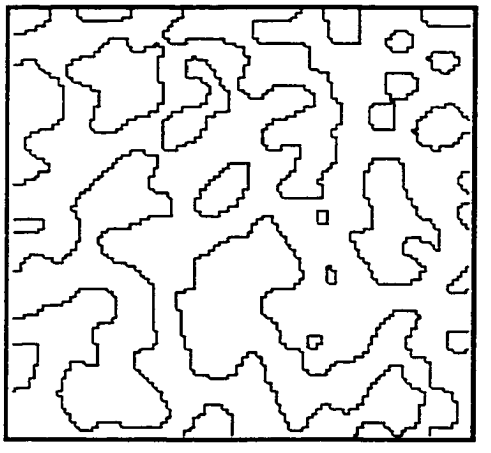
LoG $\sigma = 3,75$



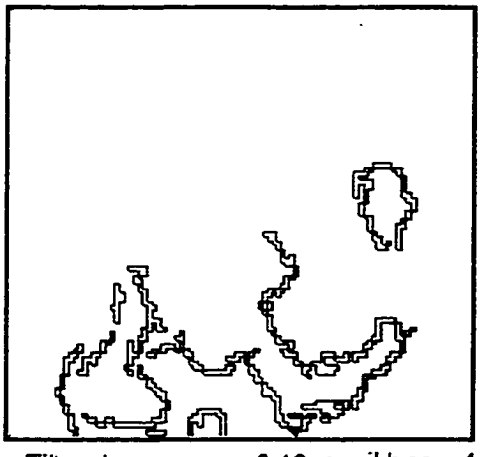
Seuillage seul à 5 mètres



Seuillage à 5 mètres + filtre morphologique



Filtre morphologique + LOG $\sigma = 3,75$



Filtre de canny $\sigma = 3.18$ seuil bas = 1
seuil haut = 2

ANNEXE 5

Tâche KBV « seuilbin »

Tâche KBV, programmée en C, permettant de faire un seuillage binaire.

```
#include <KBVstd.h>
#include <KBVUtil.h>
#include <Tcb.h>
#include <Iaf.h>
#include <Isr.h>
#include <IsrFuns.h>
#include <ErrStatus.h>

ERR_MODULE("seuilbin");

seuilbin(TCBPTR tcbptr)
{
/*Paramètres d'entrée*/
  TCB_DOCUMENTATION ("Seuil une image.") ;
    TCB_PARAM_IN (infile, KBV_STRING,
                (Documentation: "Fichier entree",
              Default     : "" ));
  TCB_PARAM_IN (seuil, KBV_INTEGER,
                (Documentation: "seuil",
                Default     : ""));
   IMAGE_PTR ImagePtr;
  KBV_INTEGER xsize, ysize;
  int i,j,type;
  int pixvalue;
  float pixvalue1;

  ImagePtr=IafReadImage(*infile);  /*infile);*/

/*On trouve les grandeurs de l'image */
  xsize=IafGetI_XSize(ImagePtr);
  ysize=IafGetI_YSize(ImagePtr);
  type=IafGetI_Datatype(ImagePtr);

  ERR_FUNCTION ("seuilbin");

/* Si le type de donnee est 4 (integer) pour tous les pixels, si le pixel est plus grand que le sueil ...la nouvelle valeur =1
sinon =0*/
  if(type==4){
  for(i=0;i<xsize;i++){
    for(j=0;j<ysize;j++){
      pixvalue=IafGetPE_ii(ImagePtr,i,j);
      if(pixvalue>*seuil)
        pixvalue=1;
      else
           pixvalue=0;
      IafSetPE_ii(ImagePtr,i,j,pixvalue);
    }
  }
  }

/*On écrit la nouvelle image de sortie */
  IafWriteImage(ImagePtr,"seuilout.im");
  ERR_RETURN KBV_SUCCESS;
  return;
}
```

ANNEXE 6

Tâche KBV « constell »

Tâche KBV, programmée en langage C, permettant de trouver la position des arbres à partir de constellations.

```
#include <KBVstd.h>
#include <KBVUtil.h>
#include <Tcb.h>
#include <Iaf.h>
#include <Isr.h>
#include <IsrFuns.h>
#include <ErrStatus.h>

ERR_MODULE("constell");

constell(TCBPTR tcbptr)
{

 TCB_DOCUMENTATION ("Recherche une constellation.") ;

 KBV_STRING tokensetname,tokensetname1;
 KBV_INDEX tokensetindex,tokensetindex1;
 KBV_CONSTELLATION * cptr;
 FEATURE_INFO * featptr, * featptr1, * featptr2, * featptr3, * featptr4, * featptr5, * featptr6, * featptr7;
 FEATURE_DESCR * descrptr,* descrptr1,* descrptr2,* descrptr3,* descrptr4,* descrptr5,* descrptr6,* descrptr7;
 KBV_LOGICAL tf;
 int i,j,flag,k;
 float elong,compact,centrx,centry,tempx,tempy;
 int height,width;
 short posx,posy;
 IMAGE_PTR ImagePtr;
 KBV_INTEGER xsize, ysize;
 int ii,jj,type,pixvalue,comptpixel;
 /*FunGetP_F GetP_f;*/
 float pixmoy;
 KBV_STRING infile="hauteurout01.im";

 KBV_FPOINT fpoint;

 tokensetname=IsrReadTokenset("/home/vision/foret/regionsed5-9.tks");
 tokensetindex=IsrGetTokensetIndex(tokensetname);
 ImagePtr=IafReadImage("/home/vision/foret/hauteurout01.im");
/* GetP_F=IafGetPfun_gf("/home/vision/foret/hauteurout01.im",NEAREST);  */
 xsize=IafGetI_XSize(ImagePtr);
 ysize-IafGetI_YSize(ImagePtr);
 type=IafGetI_Datatype(ImagePtr);

 IsrDefineTokenset("posarbre");
 IsrAddFeature("posarbre", "BITPLANE", KBV_FPOINT_DT);
 IsrAddFeature("posarbre", "CENTRX", KBV_FLOAT_DT);
 IsrAddFeature("posarbre", "CENTRY", KBV_FLOAT_DT);
 IsrAddFeature("posarbre", "HEIGHT", KBV_INTEGER_DT);
 IsrAddFeature("posarbre", "WIDTH", KBV_INTEGER_DT);

 j=0;
 for(i=1;i<12911;i++){

 comptpixel=0;
 pixmoy=0;
 descrptr=IsrGetFeatureDescr(tokensetname,"BITPLANE");
 featptr=IsrGetFeatureID(i,descrptr);
 cptr= (KBV_CONSTELLATION *) featptr->Value.PVal;
 for(ii=cptr->extents.ll.y;ii<=cptr->extents.ur.y;ii++)
 for(jj=cptr->extents.ll.x;jj<=cptr->extents.ur.x;jj++)
 {
```

```
  tf=IsrCnstlGetPixel(cptr,jj,ii);
  if(tf==1)
  {
    comptpixel=comptpixel+1;
    pixmoy=pixmoy+IafGetPE_ff(ImagePtr,jj,ii);
  }
}

if(comptpixel!=0)
  pixmoy=pixmoy/comptpixel;

descrptr=IsrGetFeatureDescr(tokensetname,"ELONGATION");
featptr=IsrGetFeatureNIN(tokensetname,i,"ELONGATION");
if(featptr->Type!=5)
break;
elong=featptr->Value.FVal;
descrptr1=IsrGetFeatureDescr(tokensetname,"COMPACTNESS");
featptr1=IsrGetFeatureNIN(tokensetname,i,"COMPACTNESS");
compact=featptr1->Value.FVal;
descrptr2=IsrGetFeatureDescr(tokensetname,"CENTROID.X");
featptr2=IsrGetFeatureNIN(tokensetname,i,"CENTROID.X");
centrx=featptr2->Value.FVal;
descrptr3=IsrGetFeatureDescr(tokensetname,"CENTROID.Y");
featptr3=IsrGetFeatureNIN(tokensetname,i,"CENTROID.Y");
centry=featptr3->Value.FVal;
descrptr4=IsrGetFeatureDescr(tokensetname,"HEIGHT");
featptr4=IsrGetFeatureNIN(tokensetname,i,"HEIGHT");
height=featptr4->Value.IVal;
descrptr5=IsrGetFeatureDescr(tokensetname,"WIDTH");
featptr5=IsrGetFeatureNIN(tokensetname,i,"WIDTH");
width=featptr5->Value.IVal;

 fpoint.x=(short)centrx;
 fpoint.y=(short)centry;

if (compact> 0.20 && elong <0.7 && height>=4 && height<=30 && pixmoy>=1)
{
  IsrDefineTokenNI("posarbre",j);
  IsrPutFeatureNIN("posarbre",j,"BITPLANE",KBV_FPOINT_DT,fpoint);
  IsrPutFeatureNIN("posarbre",j,"CENTRX",KBV_FLOAT_DT,centrx);
  IsrPutFeatureNIN("posarbre",j,"CENTRY",KBV_FLOAT_DT,centry);
  IsrPutFeatureNIN("posarbre",j,"HEIGHT",KBV_FLOAT_DT,height);
  IsrPutFeatureNIN("posarbre",j,"WIDTH",KBV_FLOAT_DT,width);
  j++;
}

}

IsrDeleteTokenset(tokensetname);
tokensetname1=IsrReadTokenset("/home/vision/foret/regionsed9-15.tks");
tokensetindex1=IsrGetTokensetIndex(tokensetname1);


for(i=1;i<5678;i++){

comptpixel=0;
pixmoy=0;

descrptr=IsrGetFeatureDescr(tokensetname1,"BITPLANE");
featptr=IsrGetFeatureID(i,descrptr);
cptr= (KBV_CONSTELLATION *) featptr->Value.PVal;
for(ii=cptr->extents.ll.y;ii<=cptr->extents.ur.y;ii++)
for(jj=cptr->extents.ll.x;jj<=cptr->extents.ur.x;jj++)
{
  tf=IsrCnstlGetPixel(cptr,jj,ii);
  if(tf==1)
```

```
  {
   if(ii<ysize && jj<xsize)
   {
   comptpixel=comptpixel+1;
   pixmoy=pixmoy+IafGetPE_ff(ImagePtr,jj,ii);
   }
  }
}

if(comptpixel!=0)
  pixmoy=pixmoy/comptpixel;

descrptr=IsrGetFeatureDescr(tokensetname1,"ELONGATION");
featptr=IsrGetFeatureNIN(tokensetname1,i,"ELONGATION");
if(featptr->Type!=5)
break;
elong=featptr->Value.FVal;
descrptr1=IsrGetFeatureDescr(tokensetname1,"COMPACTNESS");
featptr1=IsrGetFeatureNIN(tokensetname1,i,"COMPACTNESS");
compact=featptr1->Value.FVal;
descrptr2=IsrGetFeatureDescr(tokensetname1,"CENTROID.X");
featptr2=IsrGetFeatureNIN(tokensetname1,i,"CENTROID.X");
centrx=featptr2->Value.FVal;
descrptr3=IsrGetFeatureDescr(tokensetname1,"CENTROID.Y");
featptr3=IsrGetFeatureNIN(tokensetname1,i,"CENTROID.Y");
centry=featptr3->Value.FVal;
descrptr4=IsrGetFeatureDescr(tokensetname1,"HEIGHT");
featptr4=IsrGetFeatureNIN(tokensetname1,i,"HEIGHT");
height=featptr4->Value.IVal;
descrptr5=IsrGetFeatureDescr(tokensetname1,"WIDTH");
featptr5=IsrGetFeatureNIN(tokensetname1,i,"WIDTH");
width=featptr5->Value.IVal;

 fpoint.x=(short)centrx;
 fpoint.y=(short)centry;

if (compact> 0.2 && elong <0.7 && height>=10 && height<=45 && pixmoy>=1)
{
 flag=0;
 for(k=0;k<j;k++){
   descrptr6=IsrGetFeatureDescr("posarbre","CENTRX");
   featptr6=IsrGetFeatureNIN("posarbre",k,"CENTRX");
   tempx=featptr6->Value.IVal;
   posx=(short)tempx;
   descrptr7=IsrGetFeatureDescr("posarbre","CENTRY");
   featptr7=IsrGetFeatureNIN("posarbre",k,"CENTRY");
   tempy=featptr7->Value.IVal;
   posy=(short)tempy;
   if((abs(posx-centrx)+abs(posy-centry))<5){
        flag=1;
        break;}
   }
   if(flag==0){
 IsrDefineTokenNI("posarbre",j);
 IsrPutFeatureNIN("posarbre",j,"BITPLANE",KBV_FPOINT_DT,fpoint);
 IsrPutFeatureNIN("posarbre",j,"CENTRX",KBV_FLOAT_DT,centrx);
 IsrPutFeatureNIN("posarbre",j,"CENTRY",KBV_FLOAT_DT,centry);
 IsrPutFeatureNIN("posarbre",j,"HEIGHT",KBV_FLOAT_DT,height);
 IsrPutFeatureNIN("posarbre",j,"WIDTH",KBV_FLOAT_DT,width);
 j++;}

}

}
```

```
  IsrWriteTokenset("posarbre05.tks","posarbre");
  ERR_FUNCTION ("constell");
  ERR_RETURN KBV_SUCCESS;
  return;
}
```

ANNEXE 7

Tâche KBV « typearbr »

Programme permettant l'ajout de données (des arbres mesurés sur le terrain) aux éléments symboliques de positions d'arbres.

```
#include <KBVstd.h>
#include <KBVUtil.h>
#include <Tcb.h>
#include <Iaf.h>
#include <Isr.h>
#include <IsrFuns.h>
#include <ErrStatus.h>

ERR_MODULE("typearbr");

typearbr(TCBPTR tcbptr)
{

 TCB_DOCUMENTATION ("Recherche une constellation.") ;

 KBV_STRING tokensetname;
 KBV_INDEX tokensetindex;
 KBV_CONSTELLATION * cptr;
 FEATURE_INFO * featptr;
 FEATURE_DESCR * descrptr;
 KBV_LOGICAL tf;
 int i,j;

 tokensetname=IsrReadTokenset("posarbre.tks");


 /*descrptr=IsrGetFeatureDescr(tokensetname,"BITPLANE");
 featptr=IsrGetFeatureID(12,descrptr);
 cptr= (KBV_CONSTELLATION *) featptr->Value.PVal;
 for(i=cptr->extents.ll.y;i<=cptr->extents.ur.y;i++)
 for(j=cptr->extents.ll.x;j<=cptr->extents.ur.x;j++)
  tf=IsrCnstlGetPixel(cptr,j,i);*/


 /*IsrDefineTokenset("posarbre");*/
 IsrAddFeature("posarbre", "TYPEARBRE", KBV_INTEGER_DT);
 IsrAddFeature("posarbre", "FORME", KBV_INTEGER_DT);


  IsrPutFeatureNIN("posarbre",3250,"TYPEARBRE",KBV_INTEGER_DT,5);
  IsrPutFeatureNIN("posarbre",3250,"FORME",KBV_INTEGER_DT,4);

  IsrPutFeatureNIN("posarbre",434,"TYPEARBRE",KBV_INTEGER_DT,3);
  IsrPutFeatureNIN("posarbre",434,"FORME",KBV_INTEGER_DT,2);

  IsrPutFeatureNIN("posarbre",3232,"TYPEARBRE",KBV_INTEGER_DT,3);
  IsrPutFeatureNIN("posarbre",3232,"FORME",KBV_INTEGER_DT,3);

  IsrPutFeatureNIN("posarbre",422,"TYPEARBRE",KBV_INTEGER_DT,3);
  IsrPutFeatureNIN("posarbre",422,"FORME",KBV_INTEGER_DT,2);

  IsrPutFeatureNIN("posarbre",477,"TYPEARBRE",KBV_INTEGER_DT,9);
  IsrPutFeatureNIN("posarbre",477,"FORME",KBV_INTEGER_DT,4);

  IsrPutFeatureNIN("posarbre",3287,"TYPEARBRE",KBV_INTEGER_DT,3);
  IsrPutFeatureNIN("posarbre",3287,"FORME",KBV_INTEGER_DT,2);
```

```
 lsrWriteTokenset("posarbre.tks","posarbre");
 ERR_FUNCTION ("typearbr");
 ERR_RETURN KBV_SUCCESS;
 return;
}
```

ANNEXE 8

Tâche KBV « points »

Programme pour l'extraction des données brutes correspondantes à l'intérieur du contour de la constellation de l'arbre trouvé .

```
#include <KBVstd.h>
#include <Tcb.h>
#include <Iaf.h>
#include <Isr.h>
#include <IsrFuns.h>
#include <ErrStatus.h>

ERR_MODULE("points");

points(TCBPTR tcbptr)
{
  int larg, longu;
  KBV_FPOINT fpoint;
  FILE * flin, *flout;
  short data1f,data2f;
  double data3f,data4f,data5f;
  KBV_STRING tokensetname, tokensetname1, tokensetname2;
  KBV_FPOINT * cptr;
  KBV_CONSTELLATION * cptr1;
  FEATURE_INFO * featptr, * featptr1, * featptr2, * featptr3, * featptra;
  FEATURE_DESCR * descrptr,* descrptr1,* descrptr2,* descrptr3, * descrptra;
  int i,tokenconstellno,tokenconstell;
  float deltax,deltay,hauteur;
  int coordminx, coordminy, coordmaxx, coordmaxy;
  IMAGE_PTR im;
  float pixel,centroidx,centroidy;

/*************************************************************************/
  flout = fopen("arbre9.txt","wb");
  tokenconstellno = 5;
  tokenconstell = 2193;
/*************************************************************************/
  coordminx = 0;
  coordminy = 0;
  coordmaxx = 0;
  coordmaxy = 0;
  fseek(flout,0,SEEK_SET);

  ERR_FUNCTION ("points");

  tokensetname=IsrReadTokenset("/home/vision/foret/tottoken.tks");
  if(tokenconstellno==5)
    { tokensetname1=IsrReadTokenset("/home/vision/foret/regionsed5-9.tks");}
  if(tokenconstellno==9)
    { tokensetname1=IsrReadTokenset("/home/vision/foret/regionsed9-15.tks");}
  im=IafReadImage("/home/vision/foret/solout01.im");
    descrptra=IsrGetFeatureDescr(tokensetname1,"CENTROID.X");
    featptra=IsrGetFeatureNIN(tokensetname1,tokenconstell,"CENTROID.X");
    centroidx=featptra->Value.FVal;

    descrptra=IsrGetFeatureDescr(tokensetname1,"CENTROID.Y");
    featptra=IsrGetFeatureNIN(tokensetname1,tokenconstell,"CENTROID.Y");
    centroidy=featptra->Value.FVal;

    descrptra=IsrGetFeatureDescr(tokensetname1,"BITPLANE");
    featptra=IsrGetFeatureNIN(tokensetname1,tokenconstell,"BITPLANE");
    cptr1= (KBV_CONSTELLATION *) featptra->Value.PVal;
  coordminx=cptr1->extents.ll.x;
  coordmaxx=cptr1->extents.ur.x;
```

```
coordminy=cptr1->extents.ll.y;
coordmaxy=cptr1->extents.ur.y;

fprintf(flout,"%.2f,",centroidx);
fprintf(flout,"%.2f\n",centroidy);

for(i=1;i<536897;i++){

descrptr=IsrGetFeatureDescr(tokensetname,"BITPLANE");
featptr=IsrGetFeatureNIN(tokensetname,i,"BITPLANE");
cptr= (KBV_FPOINT *) featptr->Value.PVal;

larg=cptr->x;
longu=cptr->y;

descrptr1=IsrGetFeatureDescr(tokensetname,"DELTAX");
featptr1=IsrGetFeatureNIN(tokensetname,i,"DELTAX");
deltax=featptr1->Value.FVal;

descrptr2=IsrGetFeatureDescr(tokensetname,"DELTAY");
featptr2=IsrGetFeatureNIN(tokensetname,i,"DELTAY");
deltay=featptr2->Value.FVal;

descrptr3=IsrGetFeatureDescr(tokensetname,"HAUTEUR");
featptr3=IsrGetFeatureNIN(tokensetname,i,"HAUTEUR");
hauteur=featptr3->Value.FVal;

 if((larg>=coordminx && larg<=coordmaxx)&&(longu>=coordminy && longu<=coordmaxy))
 {
  if(IsrCnstlGetPixel(cptr1,larg,longu)){

                fprintf(flout,"%d,",larg);
                fprintf(flout,"%d,",longu);
                fprintf(flout,"%.2f,",deltax);
                fprintf(flout,"%.2f,",deltay);
                fprintf(flout,"%.2f,",hauteur);
                pixel=IafGetPX_gf(im,larg,longu);
                fprintf(flout,"%.2f\n",pixel);
  }
 }
}

IsrDeleteTokenset(tokensetname);

fclose(flout);

ERR_RETURN KBV_SUCCESS;

return;

}
```

ANNEXE 9

Programme Matlab

Programme Matlab pour le calcul de hauteur des arbres (les données pour chacun des arbres sont incluses dans ce programme).

```
clear;
clc;
m=4;  %deplacement du centroid

noarbre=8; %1 2 3 8 9 10 14 16 17 18 19 34 35 36 42 43 54 56 numero de l'arbre que l'on veut traiter

coupedonnees=0; %si on veut couper les donnees a un certain niveau
hautcoup=3; %hauteur ou on veut couper les donnees (2 => 1/2)

coupelarg=0;
largcoup=8;

coupelargtriees=0;  %couper les donnees triees en largeur
largcouptriees=7;
couper=coupelargtriees;
largcouper=largcouptriees;

printallrz=0; % pour imprimer tous les graphes r vs z (printallrz=1)

r=0;
x=0;
y=0;
z=0;

%arbre 1
if noarbre==1
x=[995.78 995.8 997.68 997.74 998.1 998.14 999.76 1000.08 1000.76 1000.84 1001.38 1001.78 1001.92 1002.38
1002.62 1003.28 1003.3 1004.5 1005.22 1005.36 1005.88 1005.9 1005.9 1006 1006.38 1007.32 1008.02 1008.06
1008.18 1008.36 1008.64 1009.34 1009.92 1010.24 1010.48 1010.76 1010.8 1010.98 1010.98 1012.2 1012.28 1012.42
1012.76 1013 1013.6 1013.86 1013.98 1014.22 1014.3 1014.4 1015.14 1015.3 1015.74 1016.38 1016.86 1017.42
1018.02 1018.12 1018.42 1019.36];
y=[602.7 603.68 600.22 600.16 603.4 600.64 596.22 599.54 603.04 604.54 595.72 596.04 603.18 603.12 599.4 603.52
599.48 597.96 605.6 606.92 596.26 595.34 606.22 605.68 599.48 602.16 602.64 601.94 605.56 609.84 611.56 611.14
610.8 597.94 601.4 607.62 609.14 605.46 609.92 597.88 607.98 606.38 603.82 601.78 606.62 603.74 604.58 610.24
610.98 598.68 602.28 607.7 609.42 606.8 606.84 608.46 605.88 603.4 602.78 604.82];
z=[6.3 1.35 6.63 11.94 12.47 7.09 9.28 11.41 10.89 10.81 2.46 8.78 11.47 9.18 8.4 14.72 7.33 12.01 12.24 11.08 8.77
8.68 11.97 14.43 10.16 14.28 13.69 15.13 13.97 12.1 12.57 9.25 1.3 14.94 15.49 13.98 13.81 15.65 9.83 13.39 14.28
15.27 14.51 15.57 15.74 15.21 13.47 4.22 14.42 10.03 13.42 15.49 15.2 10.83 13.69 6.22 15.51 13.03 7.13 6.41];
centrxi=1008;
centryi=602;
end
%arbre2
if noarbre==2
x=[1003.9 1004.36 1006.36 1006.5 1006.86 1008.5 1008.52 1008.78 1009.62 1009.66 1010.14 1010.24 1010.26
1010.66 1011.54 1011.66 1011.94 1011.98 1012.2 1012.34 1012.74 1014.14 1014.2 1016.42 1018.12];
y=[630.04 634.16 634.68 624.48 629.92 630.88 626.42 624.5 631.64 634.66 635.76 635.86 627.4 632.94 627.48 633.64
630.3 627.72 631.94 635.24 631.88 631.54 629.88 627.34 629.2];
z=[14.79 14.49 17 5.14 19.29 17.76 16.2 3.24 20.99 19.9 17.46 18.71 14.85 17.65 14.85 19.65 18.51 10.52 18.34 15.89
17.42 11.06 18.72 11.34 9.84];
centrxi=1009;
centryi=632;
end
%arbre =3
if noarbre==3
x=[948.38 949.02 950.38 951.16 952.54 952.9 953.22 953.48 954.88];
y=[595.44 599.08 601.2 595.1 597.74 600.26 603.26 602 603.46];
z=[8.52 7.4 7.97 8.88 9.08 0.69 7.15 0.4 0.53];
centrxi=951;
centryi=597;
```

```
end
%arbre8
if noarbre==8
x=[887.82 889.78 890.1 891.58 892.06 892.14 892.74 893.4 893.98 894.76 895.16 895.46 897.36 898.88 899.76];
y=[653.34 649.48 655.26 657.24 652.24 651.54 654.46 655.96 653.44 657.2 651.06 651.72 657.62 660.9 654.2];
z=[3.61 2.2 4.26 4.45 3.53 4.22 4.36 3.21 4.51 4.01 3.52 1.63 2.54 1.33 0.64];
centrxi=893;
centryi=654;
end
%arbre9
if noarbre==9
x=[886.14 887.28 888.2 888.28 889.64 890.22 890.96 892.32 892.58 892.6 893.32 893.36 893.52 894.36 895.06 895.14
895.22 895.7 895.8 897.82 898.38 898.94];
y=[685.48 687.98 681.44 689.02 684.16 685.22 685.6 681.02 686.78 690.66 690.64 690.28 682.62 683.3 687.1 690.64
686.76 686.88 692.82 692.16 688.7 695.12];
z=[6.59 6.28 5.85 7.86 6.6 8.43 7.46 7.69 9.05 8.6 8.8 8.14 -0.79 8.94 8.77 6.84 6.99 9.25 8.04 2.56 6.97 1.45];
centrxi=893;
centryi=687;
end
%arbre10
if noarbre==10
x=[877.26 878.1 878.26 878.36 878.72 879.26 879.86 880 880.72 880.84 880.9 881.2 881.48 881.96 882.52 882.74
882.86 884.56 884.76 884.96 886 887.06 887.86 890.78];
y=[715.88 715.5 709.54 712.32 702.36 712.36 718.54 711.84 708.9 717.94 714.4 708.96 715.16 711.86 715.94 709.66
714.94 711.08 712.5 709.98 714.84 710.22 715.04 713.52];
z=[8.78 19.71 14.7 -0.9 1.78 14.04 12.65 19.78 -0.93 11.02 23.86 12.93 21.92 16.48 20.47 6.12 16.41 9.89 20.96 9.27
0.01 9.74 19.08 -0.47];
centrxi=882;
centryi=714;
end
%arbre14
if noarbre==14
x=[811.02 812.92 813.16 813.28 813.54 813.6 813.66 813.68 813.96 814.36 814.56 814.82 814.9 815.2 815.42 815.88
815.88 817.02 818.24 820.52 822.84 823.18 823.68 823.72 824.4 825.44 829.38 829.42 830.78];
y=[863.12 866.2 859.22 864.14 865.12 863.34 861.82 860.3 860.96 862.92 865.14 863.48 862.76 861.16 862.06 861.46
866.58 858.96 862.6 859.06 868.42 867.08 868.42 867.12 853.84 855.76 856.5 855.9 855.6];
z=[12.79 11.62 14.57 14.92 14.8 -2.21 -1.63 12.69 13.88 15.32 0.38 -2.1 13.28 15.3 15.16 1.36 11.1 14.3 14.31 13.63
13.31 0.45 0.04 0.01 1.83 2.92 4 3 0.41];
centrxi=817;
centryi=861;
end
%arbre16
if noarbre==16
x=[665.34 665.44 665.7 666.14 666.74 667.34 667.38 667.84 668.12 669.24 669.58 670.3 670.36 670.38 671.92 672.06
672.44 672.8 673.04 673.52 674.44 674.5 674.9 675.5 676 676.78];
y=[791.24 790.82 793.1 788.04 786.02 787.82 787.6 789.76 793.12 783.62 783.82 783.26 789.48 785.46 786.16 783.44
796.5 795.14 794.4 793.32 794.28 793.18 790.88 790.2 794.98 789.02];
z=[8.97 8.48 9.21 9.34 9.76 10.38 9.53 8.39 11.78 4.92 11.24 5.71 13.67 12.1 9.72 3.41 12.33 1.06 11.5 9.07 10.54
12.76 13.15 13.19 11.28 0.77];
centrxi=670;
centryi=790;
end
%arbre17
if noarbre==17
x=[650.68 651.5 653.02 653.98 655.4 655.48 657.32 657.46 657.94 658.26 658.32];
y=[804.02 797.06 800.24 797.2 804.82 804.82 802.04 804.04 801.1 805.48 802.8];
z=[5.41 9.61 7.71 -1.9 7.8 7.23 3.3 4.44 5.26 1.97 0.51];
centrxi=654;
centryi=802;
end
%arbre18
if noarbre==18
x=[643.02 643.4 645.14 646.98 647.32 649.04 649.3 649.4 650.5 650.78 651.06 651.16 651.16 651.22 651.26 651.4
651.42 651.5 651.64 651.68 651.72 651.76 651.78 651.8 651.82 651.86 652.02 653.26 653.3 653.72 655.18 655.54];
y=[828.22 828.42 827.96 824.34 829.54 831.3 825.9 834.04 834.66 834.68 831.1 828 834.56 833.98 834.76 833.96
830.08 833.52 833.3 833.82 833.32 833.32 833.36 833.08 833.36 833.34 834.7 827.34 827.54 830.34 828.1 827.76];
```

```
z=[4.19 4.14 8.06 6.31 11.76 8.17 10.35 8.33 1.49 7.1 11.02 8.5 9.49 7.97 7.9 1.09 12.93 7.75 7.65 7.58 7.66 7.58 7.63
7.62 7.57 7.49 7.05 8.41 10.09 11.43 1.46 7.27];
centrxi=650;
centryi=829;
end
%arbre19
if noarbre==19
x=[644.3 644.34 646.22 646.24 648.28 648.34 648.92 649.06 649.72 649.74 650.1 650.4 650.96 651.08 651.46 651.58
651.62 652.64 652.74 652.9 653.18 653.54 653.58 653.88 654.76 654.92 654.98 655.14 655.48 655.8 657.02 657.12
657.68 658.52];
y=[868.04 873.34 868.1 869.78 869.42 872.8 872.04 880.62 871.52 873.12 879.84 879.98 877.02 880.88 878.52 870.36
876.66 877.64 879.86 873.38 884.12 871.18 874.16 872.78 876.64 873.82 883.42 880.84 879.92 880.4 880.3 873.12
877.06 875.68];
z=[1.53 15.32 1.85 13.83 1.12 14.79 13.4 14 1.61 14.7 14.89 4.93 14.18 13.86 12.94 12.59 12.49 10.55 14.67 14.89
14.15 0.32 11.42 14.32 15.82 11.93 10.77 15.4 15.22 0.92 9.49 14.69 12.31 4.07];
centrxi=652;
centryi=876;
end
%arbre34
if noarbre==34
x=[981.62 981.9 982.74 982.82 982.92 983.9];
y=[588.58 586.1 588.34 586.2 588.98 589.44];
z=[8.18 8.55 8.85 5.12 1.59 1.18];
centrxi=982;
centryi=587;
end
%arbre35
if noarbre==35
x=[969.34 970.14 970.44 970.9 972.16 973.78 974.6];
y=[574.9 579.22 575.92 578 575.38 574.16 578.32];
z=[7.19 5.43 9.07 4.39 6.15 1.35 8.45];
centrxi=972;
centryi=578;
end
%arbre36
if noarbre==36
x=[909.48 911.42 913.14];
y=[691.98 688.26 690.42];
z=[4.22 4.95 3.81];
centrxi=913;
centryi=689;
end
%arbre42
if noarbre==42
x=[1014.48 1014.56 1015 1017.02 1017.94 1018.06 1018.24 1018.7 1019.14 1019.14 1019.28 1019.5 1020.04 1020.26
1020.58 1021.02 1021.16 1022.1 1022.42 1023.9];
y=[651.94 649.62 649.82 659.94 657.86 652.08 655 649.72 655.7 658.92 655.34 651.38 651.48 656.48 649.48 655.6
651.7 655.64 657.24 651.24];
z=[15.23 -0.29 15.84 3.81 12.52 20.44 16.52 9.83 15.61 11.85 18.53 15.58 14.29 19.62 3.73 17.28 14.42 5.23 5.93
10.68];
centrxi=1019;
centryi=653;
end
%arbre43
if noarbre==43
x=[1015.4 1017.2 1017.38 1017.42 1018.96 1019.4 1019.46 1019.74 1019.82 1019.96 1020.5 1021.44 1021.92 1022.24
1022.7 1022.9 1024.04 1025.52];
y=[619.66 621.54 615.74 618.96 622.32 617.06 615.24 619.34 622.3 616.7 619.68 618.34 621.92 618.66 615.76 624.32
617.9 614.54];
z=[10.19 12.85 8.51 10.22 10.62 10.89 10.22 8.93 10.36 4.61 9.37 9.71 9.4 9.73 11.1 13.87 9.56 4.36];
centrxi=1020;
centryi=618;
end
%arbre54
if noarbre==54
x=[715.04 715.64 716.12 716.56 716.66 716.72 716.72 716.74 716.74 716.76 716.8 716.82 716.88 717.5];
```

```
y=[1178.34 1180.14 1180.92 1180.06 1180 1180.06 1180.46 1179.94 1180.22 1180.08 1180.02 1180.26 1180.06
1176.62];
z=[4.95 6.67 5.59 5.39 5.76 5.76 4.77 6.22 5.47 5.76 5.51 5.41 5.51 7.56];
centrxi=718;
centryi=1178;
end
%arbre56
if noarbre==56
x=[722.44 724.78 725.78 726.72];
y=[1201.34 1205.44 1200.28 1201.72];
z=[4.51 5.83 0.58 6.21];
centrxi=724;
centryi=1203;
end

d=0;
d=size(x);
%Trouve le max des donnees en z
maxi=0;
for l=1:d(2)
        if z(l)>maxi
                maxi=z(l);
        end
end
mm=1;
%coupe les donnees en hauteur selon une hauteur minimum voulue en pourcentage de la hauteur
if coupedonnees==1
        for l=1:d(2)
                if z(l)>(maxi/hautcoup)
                        xx(mm)=x(l);
                        yy(mm)=y(l);
                        zz(mm)=z(l);
                        mm=mm+1;
                end
        end
        x=0;
        y=0;
        z=0;
        x=xx;
        y=yy;
        z=zz;
end
d=size(x);

%coupe les donnees en largeur selon un rayon maximum voulu
mm=1;
if coupelarg==1
        for l=1:d(2)
    if x(l)<(centrxi+largcoup)
      if x(l)>(centrxi-largcoup)
        if y(l)<(centryi+largcoup)
          if y(l)>(centryi-largcoup)
                xx(mm)=x(l);
                yy(mm)=y(l);
                zz(mm)=z(l);
            mm=mm+1;
          end
        end
      end
    end
end
x=0;
y=0;
z=0;
x=xx;
y=yy;
```

```
z=zz;
end
d=size(x);

%dessine les donnees en 3D
%figure(1)
%plot3(x(1)-centrxi,y(1)-centryi,z(1),'o');
%hold on;
%for l=2:d(2)
%   plot3(x(l)-centrxi,y(l)-centryi,z(l),'o');
%end
%hold off;
%xlabel('x');
%ylabel('y');
%zlabel('z');
%title('Donnees laser');


centrx=centrxi;
centry=centryi;
SSE=1000;
SSEtrie=1000;
SSEtrie2=1000;
SSEtrie3=1000;
hmax=0;
pp=9;

%calcul de la droite de regression de degre 1
for j=-m:m,
  for k=-m:m,
    for i=1:d(2),
        r(i)=sqrt((x(i)-centrx)*(x(i)-centrx)+(y(i)-centry)*(y(i)-centry));
    end

    %DEBUT
    %calcul de la droite de regression avec les donnees triees ayant le meilleur centroide
    rtrie=sort(r);
    dtrie=size(z);
    ztrie=sort(z);
    for i=1:dtrie(2),
        ztrie1(i)=ztrie(dtrie(2)-i+1);
    end


    mmmm=1;
    if couper==1
        for l=1:dtrie(2)
                if rtrie(l)<largcouper
                        rrrrr(mmmm)=rtrie(l);
                        zzzzz(mmmm)=ztrie1(l);
                        mmmm=mmmm+1;
                end
        end
        rtrie=0;
        ztrie=0;
        rtrie=rrrrr;
        ztrie1=zzzzz;
    end
    dtrie=size(rtrie);


    %degre1
    [ptrie,strie]=polyfit(rtrie,ztrie1,1);
    xitrie=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25];
    yitrie=0;
    yitrie=polyval(ptrie,xitrie);
```

```
hauttrie=polyval(ptrie,0);
if strie.normr<SSEtrie %sauvegarde de la meilleure combinaison
    rftrie=rtrie;
    ztrie1f=ztrie1;
    cxtrie=centrx;
    cytrie=centry;
    yitrief=yitrie;
    xitrief=xitrie;
    SSEtrie=strie.normr;
    hautftrie=hauttrie;
    [ptrie2,strie2]=polyfit(rtrie,ztrie1,2);
    xitrie2=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25];
    yitrie2=0;
    yitrie2=polyval(ptrie2,xitrie2);
    hautftrie2=polyval(ptrie2,0);
end
%degre2
[ptrie3,strie3]=polyfit(rtrie,ztrie1,2);
xitrie3=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25];
yitrie3=0;
yitrie3=polyval(ptrie3,xitrie3);
hauttrie3=polyval(ptrie3,0);
if strie3.normr<SSEtrie3 %sauvegarde de la meilleure combianaison
    ztrie1f3=ztrie1;
    rftrie3=rtrie;
    cxtrie3=centrx;
    cytrie3=centry;
    SSEtrie3=strie3.normr;
    xitrie3f=xitrie3;
    yitrie3f=yitrie3;
    hautftrie3=hauttrie3;
    end %FIN


    [p,s]=polyfit(r,z,1);
    xi=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20];
    yi=0;
    yi=polyval(p,xi);
    haut=polyval(p,0);
    if haut>hmax
            hmax=haut;
    end
    zmoy=0;
    rmoy=0;
    for l=1:d(2)
            if z(l)~=maxi
                    zmoy=zmoy+z(l);
                    rmoy=rmoy+r(l);
            end
            if z(l)==maxi
                    rmaxi=r(l);
            end
    end
    zmoy=zmoy/(d(2)-1);
    rmoy=rmoy/(d(2)-1);

%calcul de la hauteur calculee avec la droite passant par le centroid des points sur
%le graphique 2D et le point le plus haut
jm=(maxi-zmoy)/(rmaxi-rmoy);
jb=zmoy-jm*rmoy;
%sauvegarde des donnees pour la meilleure droite trouvee
    if s.normr<SSE
            cx=centrx;
            cy=centry;
            pf=p;
            sf=s;
```

```
                SSE=s.normr;
                rf=r;
                yif=yi;
                hautf=haut;
                maxif=maxi;
                rmaxif=rmaxi;
                zmoyf=zmoy;
                rmoyf=rmoy;
                jbf=jb;
                jmf=jm;
        end
    centrx=centrxi+j;
    centry=centryi+k;
    %impression de chacune des droites pour toutes les positions de centroid essayees
    if printallrz==1
                figure(pp)
                plot(r,z,'o',xi,yi,'--');
                        title('Regression');
                axis([0 20 0 20]);
                pp=pp+1;
                xlabel(s.normr);
        end
  end
end

%LES MEILLEURS RESULTATS SONT DONNES PAR LES DEUX FIGURES SUIVANTES
%figure(2);
%plot(rftrie,ztrie1f,'o',xitrief,yitrief,'--');
%title('Donnees triees avec le meilleur centroid / droite de regression');
%axis([0 25 0 25]);
%xlabel('r');
%ylabel('z');
%hautftrie
%cxtrie , cytrie
%figure(3);
%plot(rftrie,ztrie1f,'o',xitrie2,yitrie2,'--');
%title('Donnees triees avec le meilleur centroid pour la droite / courbe de regression');
%axis([0 25 0 25]);
%xlabel('r');
%ylabel('z');
%hautftrie2
%FIN DU COMMENTAIRES

%figure(4);
%plot(rftrie3,ztrie1f3,'o',xitrie3f,yitrie3f,'--');
%title('Donnees triees avec le meilleur centroid pour la courbre de regression');
%axis([0 25 0 25]);
%xlabel('r');
%ylabel('z');
%hautftrie3
%cxtrie3 , cytrie3

%impression de la droite de regression
figure(5)
plot(rf,z,'o',xi,yif,'--');
axis([0 20 0 20]);
xlabel('r')
ylabel('z')
title('Droite de regression');

%impression des donnees de hauteur et de centroid trouve
hautf
%hmax
%SSE
%cx , cy
%jbf   %hauteur centroid et point le plus haut
```

```
%rmaxif
%maxif
%rmoyf
%zmoyf


%calcul de la parabole de regression de degre 2
centrx2=centrxi;
centry2=centryi;
SSE2=1000;
hmax2=0;
%degre 2
for j=-m:m,
  for k=-m:m,
        for i=1:d(2),
              r2(i)=sqrt((x(i)-centrx2)*(x(i)-centrx2)+(y(i)-centry2)*(y(i)-centry2));
        end

        [p2,s2]=polyfit(r2,z,2);
        xi2=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20];
        yi2=0;
        yi2=polyval(p2,xi2);
        haut2=polyval(p2,0);
        if haut2>hmax2
        hmax2=haut2;
  end
if s2.normr<SSE2
cx2=centrx2;
cy2=centry2;
pf2=p2;
sf2=s2;
SSE2=s2.normr;
rf2=r2;
yif2=yi2;
hautf2=haut2;
maxif2=maxi;
end
centrx2=centrxi+j;
centry2=centryi+k;
 if printallrz==1
        figure(pp)
        plot(r2,z,'o',xi2,yi2,'--');
        title('Regression');
        axis([0 20 0 20]);
        pp=pp+1;
        xlabel(s2.normr);
end
end
end

%impression de la parabole de regression
figure(6)
plot(rf2,z,'o',xi2,yif2,'--');
axis([0 20 0 20]);
xlabel('r')
ylabel('z')
title('Courbe de regression (parabole)');

hautf2
%hmax2
%SSE2
%cx2
%cy2


%calcul de la surface de regression de degre 3
```

```
maxhaut=0;
iii=1;
for mmm=-m:m
  for nnn=-m:m
    for iii=1:d(2)
        centrxii=centrxi+mmm;
        centryii=centryi+nnn;
        A(iii,1)=1;
        xtemp=x(iii)-centrxii;
        ytemp=y(iii)-centryii;
        A(iii,2)=xtemp;
        A(iii,3)=ytemp;
        A(iii,4)=xtemp*xtemp;
        A(iii,5)=ytemp*ytemp;
        A(iii,6)=xtemp*ytemp;
        K(iii)=z(iii);
        iii=iii+1;
    end
        temp=A'*A;
        temp1=inv(temp);
        temp2=A'*K';
        temp3=temp1*temp2;
    if temp3(1)>maxhaut
      maxhaut=temp3(1);
      centrxiif=centrxii;
      centryiif=centryii;
    end
  end
end

maxhaut
%centrxiif
%centryiif

for ix=-9:9
  for iy=-9:9
    zii(ix+10,iy+10)=temp3(1)+temp3(2)*ix+temp3(3)*iy+temp3(4)*ix*ix+temp3(5)*iy*iy+temp3(6)*ix*iy;
    xii(ix+10,iy+10)=ix;
    yii(ix+10,iy+10)=iy;
  end
end
%impression de la surface de regression
figure(7);
mesh(xii,yii,zii);
hold on;
for ppp=1:d(2)
  plot3(x(ppp)-centrxiif,y(ppp)-centryiif,z(ppp),'o');
end
hold off;
title('Surface de regression');
xlabel('x');
ylabel('y');
zlabel('z');


rsort=sort(r);
r2sort=sort(r2);
d=size(z);
zsort=sort(z);
for i=1:d(2),
  zsort1(i)=zsort(d(2)-i+1);
end
```

```
%coupe les donnees triees selon un rayon r
mmm=1;
if coupelargtriees==1
        for l=1:d(2)
         if rsort(l)<largcouptriees
                rrr(mmm)=rsort(l);
                zzzz(mmm)=zsort1(l);
                mmm=mmm+1;
        end
        end
        rsort=0;
        zsort1=0;
        rsort=rrr;
        zsort1=zzzz;
end
d=size(rsort);

%centroid trouve a partir du degre 1
[psort,ssort]=polyfit(rsort,zsort1,1);
xisort=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25];
yisort=0;
yisort=polyval(psort,xisort);
hautsort=polyval(psort,0);
%impression de la droite de regression sur les donnees triees
%figure(8);
%plot(rsort,zsort1,'o',xisort,yisort,'-');
%title('Donnees triees');
%axis([0 25 0 25]);
%xlabel('r');
%ylabel('z');


[psort1,ssort1]=polyfit(rsort,zsort1,2);
xisort1=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25];
yisort1=0;
yisort1=polyval(psort1,xisort1);
hautsort1=polyval(psort1,0);
%impression de la courbe de regression sur les donnees triees
%figure(9);
%plot(rsort,zsort1,'o',xisort1,yisort1,'-');
%title('Donnees triees');
%axis([0 25 0 25]);

%hautsort
%hautsort1
```