



HAL
open science

Estimation de la qualité de grumes de bois

Florian Delconte

► **To cite this version:**

Florian Delconte. Estimation de la qualité de grumes de bois. Vision par ordinateur et reconnaissance de formes [cs.CV]. 2019. hal-03603108

HAL Id: hal-03603108

<https://hal.univ-lorraine.fr/hal-03603108>

Submitted on 9 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Stage de master informatique deuxième année
Parcours type Apprentissage, Vision, Robotique (AVR)
Rapport

Estimation de la qualité de grumes de bois

Encadré par : Phuc Ngo et Isabelle Debled-Rennesson

Équipe : ADAGIo (Algorithmique Discrète et ses Applications à la Géométrie et à l'Imagerie)

Florian Delconte
Université de Lorraine

LORIA - Département Algorithmique, Calcul, Image et Géométrie
615 Rue du Jardin-Botanique, 54506



Remerciements

Avant tout, je tenais à remercier toute l'équipe ADAGIo du LORIA dont l'implication, le soutien et la bienveillance ont été déterminants pour le bon déroulement de ce stage. Tout d'abord, je remercie Phuc Ngo, professeur agrégé à l'université de Lorraine, pour ses judicieux conseils ; Rémi Decelle, doctorant, pour m'avoir fait partager son expérience doctorale et ses précieuses expertises ; et enfin Isabelle Debled-Rennesson, directrice de recherche et professeure en informatique, pour sa disponibilité, sa patience et sa verve pédagogique. Je tiens à leur exprimer toute ma reconnaissance pour leur aide et leurs encouragements, et à leur témoigner toute ma gratitude pour m'avoir fait découvrir le milieu de la recherche avec passion et engouement.

Résumé

Il arrive que des arbres de mauvaise qualité soient abattus et acheminés en scierie. Pour filtrer et traiter différemment ces mauvais bois, un opérateur prend note des défauts visibles à l'extrémité du tronc. A partir du nombre de défauts visibles et de quelques mesures géométriques, l'opérateur estime la qualité du bois. Cependant, cette estimation mène parfois à classer un arbre de mauvaise qualité comme étant un arbre de bonne qualité. La mauvaise qualité de l'arbre n'est parfois découverte qu'à la fin du processus de sciage. Le bois est alors jeté entraînant un gâchis économique et environnemental. Nous cherchons à automatiser les démarches de l'opérateur pour réduire les erreurs liées aux classements des arbres. En mêlant des techniques de segmentation par apprentissage profond, de reconnaissance de damier dans l'espace de Hough et de construction géométrique discrète, nous avons mis en place une procédure permettant de faire automatiquement un ensemble de mesures sur des images d'extrémité de troncs. Il reste à définir comment estimer la qualité à partir de ces mesures.

Table des matières

1	Cadre de travail	1
1.1	Environnement	1
1.1.1	Laboratoires	1
1.1.2	Organisation	2
1.2	Sujet	3
1.2.1	Objets d'intérêts	3
1.2.2	Sujet	3
2	Extraction des objets d'intérêt	6
2.1	Segmentation de la grume	6
2.1.1	État de l'art	6
2.1.2	Réseaux neuronaux convolutifs	8
2.1.3	U-Net	8
2.1.4	Modèle pour la segmentation de grume	9
2.2	Segmentation de la mire	9
2.2.1	État de l'art	9
2.2.2	Segmentation avec réseau neuronal convolutif	11
2.2.3	Segmentation en utilisant la détection de coins	14
3	Conversion de pixels à centimètre	18
3.1	La méthode développée	18
3.1.1	Contour de la mire	18
3.1.2	Détections des droites de la mire	20
3.1.3	Reconstruction de la mire par segment flou	22
3.1.4	Expérimentations	24
3.1.5	Suppression du bruit	26
4	Mesures géométriques sur grume de bois	28
4.0.1	Mesures géométriques	28
5	Implémentation	31
5.1	Techonologies utilisées	31
5.2	Solution apportée	32
6	Conclusion	33
	Références	34
A	Architecture du réseau permettant de segmenter la grume	36

B	Code python pour le calcul des TP,TN,FP,FN	37
C	Exemple de segmentation de grume et de mire	38

1 Cadre de travail

1.1 Environnement

1.1.1 Laboratoires

LORIA-ADAGIo

Le LORIA est le Laboratoire Lorrain de Recherche en Informatique et ses Applications. Créé en 1997, il est membre, avec les deux autres principaux laboratoires de recherche en mathématiques et STIC (Science et Technologie de l'Information et de la Communication) de Lorraine, de la Fédération Charles Hermite. C'est une Unité Mixte de Recherche (UMR 7503), commune à plusieurs établissements : le CNRS (Centre National de la Recherche Scientifique), l'Université de Lorraine et Inria (Institut National de Recherche en Informatique et en Automatique), le laboratoire est structuré autour de 28 équipes réparties en 5 départements.



FIGURE 1 – Le Loria

L'équipe ADAGIo fait partie du département Algorithmique, Calcul, Image et Géométrie. Sous la responsabilité d'Isabelle Debled-Rennesson, ADAGIo est composée de 5 membres (4 permanents et un doctorant). L'équipe axe ses recherches principalement autour de l'algorithmique et la géométrie discrète, une branche des mathématiques qui consiste à construire des modèles discret de problème ou phénomène réel. En imagerie, le monde réel est échantillonné sur des grilles 2D ou 3D. ADAGIo s'intéresse aux objets discrets sous-jacents aux problèmes de détection, d'analyse ou de reconnaissance, qui conservent une propriété de réversibilité.

INRA

L'INRA est l'Institut national de la recherche agronomique. Fondé en 1946, c'est un organisme français de recherche en agronomie. Il s'agit du premier institut de recherche agronomique en Europe et deuxième dans le monde en nombre de publications sur les sciences animales, agricoles et végétales. L'INRA mène des recherches sur l'alimentation saine, l'agriculture durable et pour un environnement préservé. A Nancy, le centre Inra-Grand Est axe ses recherches sur les écosystèmes forestiers et sur l'ingénierie et la sécurité sanitaire des aliments.



FIGURE 2 – L'INRA

Projet TreeTrace

De plus en plus de dispositif d'imagerie sont installés en scieries, cela indique l'importance de ces données pour améliorer le rendement économique des scieries. Dans ce contexte, TreeTrace est un projet ANR qui s'intéresse à deux type d'application.

Le premier est la question du suivi des grumes depuis la forêt jusqu'à la scierie en exploitant des informations biométriques liées à la grume et des techniques de reconnaissance basées sur le traitement d'images de sections de grume. Le deuxième se penche sur la détermination de la qualité du bois à partir d'images capturées en forêt ou en scierie.

Ces deux applications partagent des points communs. Premièrement, la qualité du bois peut être évaluée à partir de dispositif d'imagerie utilisé en traçabilité, et les données servant à déterminer la qualité du bois peuvent être utilisées pour l'application de traçage. Deuxièmement, les caractéristiques extraites de la grume pour la traçabilité s'avèrent utile pour déterminer la qualité du bois. Enfin, les questions concernant le choix des capteurs d'imagerie sont commune aux deux applications.

1.1.2 Organisation

Réunions

Une réunion hebdomadaire est organisée chaque semaine, durant laquelle le personnel de l'équipe peut partager ses recherches ou présenter un article scientifique. Ces réunions ont permis de découvrir le domaine de recherche de chacun et de s'entraîner à la présentation de travaux scientifiques. De

plus, une réunion hebdomadaire, avec les encadrants ponctue chaque fin de semaine. Le but de ces réunions est de présenter l'avancée du stage, de mettre les mots sur les difficultés rencontrées et d'aiguiller les recherches afin de ne pas perdre trop de temps. Remi Decelle, travaille sur une thèse en lien avec la problématique du stage. Nous avons échangé régulièrement sur nos avancées respectives et partagé certaines connaissances. Enfin, des échanges réguliers avec l'INRA ont permis d'acquérir un savoir en Dendrologie et de formuler des demandes précises sur l'acquisition des données utilisées durant le stage.

Bureautique

Deux ordinateurs, l'un basé sous Ubuntu 18.04 et l'autre sous Debian Sretch sont utilisés. Cette configuration garantit une solution utilisable sous Linux, elle permet également de développer la solution et d'entraîner un modèle de segmentation en même temps.

- Ubuntu 18.04 : 8Giga de RAM, Intel(R) Xeon(R) CPU E5-1650 0 @ 3.20GHz
- Debian Sretch : 8Giga de RAM, Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz

1.2 Sujet

1.2.1 Objets d'intérêts

Grume

Une grume est un tronc d'arbre dont les branches ont été retirées. Avant d'être acheminée en scierie, la grume est photographiée à ses extrémités. Plusieurs espèces d'arbres sont ainsi photographiées. La couleur du bois et la forme du tronc peuvent varier d'une espèce à l'autre.

Mire

Une mire est une suite de carrés blanc et noir, dont la longueur d'un côté mesure 1cm. Disposés alternativement, ces carrés forment un damier de taille 13*2.



FIGURE 3 – Objets d'intérêts

1.2.2 Sujet

L'INRA a conçu une base de données afin que l'équipe ADAGio puisse travailler dessus. Les images sont de taille 4320 par 3240 et possèdent des caractéristiques différentes. La mire et la grume ne sont pas forcément sur le même plan. Le fond des images est changeant ; avec ou non, un environnement forestier. Des variations de luminosité sont possibles d'une image à l'autre et au sein du même objet d'intérêt. Certaines images sont très peu contrastées. Enfin, rien n'oblige que la mire soit en entier sur l'image étudiée. La figure 4 montre la diversité des images.

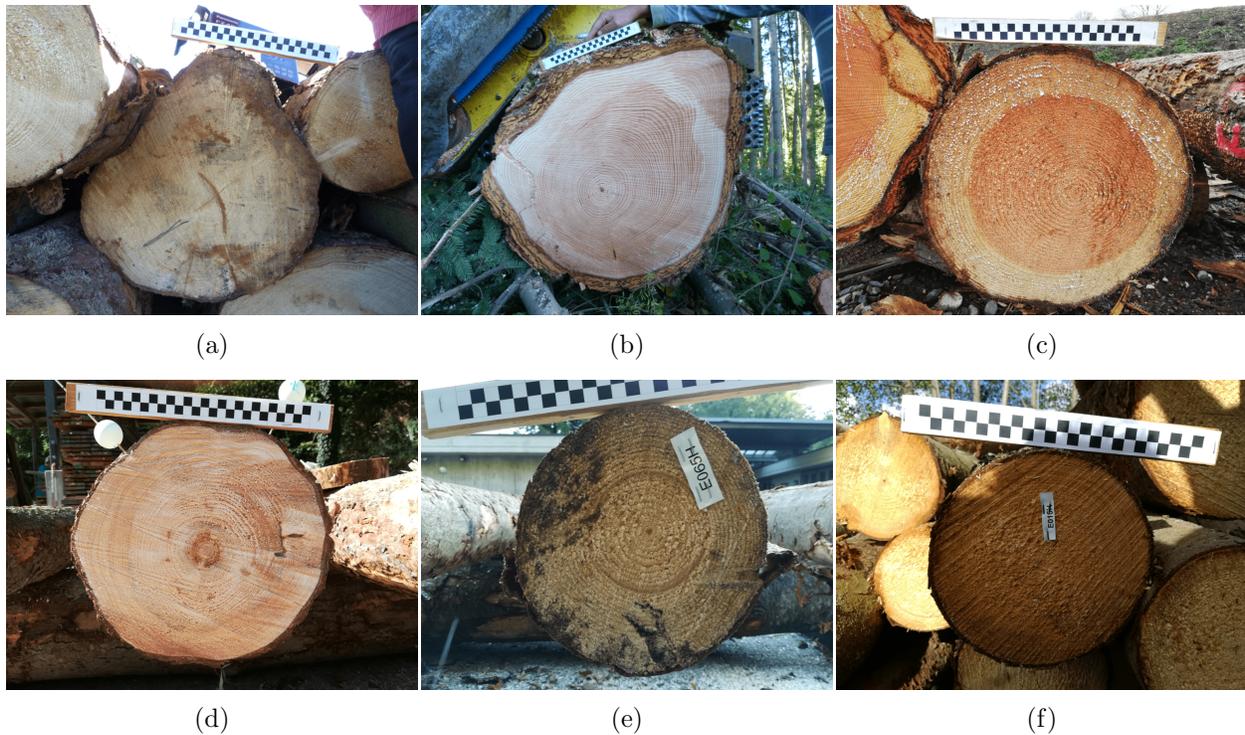


FIGURE 4 – Exemples d'images

(a) : grume de Douglas, la couleur de la mire est proche de celle du ciel . (b) : grume de Douglas dans un environnement forestier. (c) : grume de Douglas de couleur orangée.(d) : grume d'Épéca claire et une mire parfaitement visible. (e) : grume d'Épéca avec une mire partiellement visible. (f) : grume d'Épéca, un jeu d'ombre est visible sur la mire.

Le stage "Estimation de la qualité de grumes de bois", proposé par l'équipe ADAGIo s'inscrit dans le cadre du projet TreeTrace en se posant la problématique de l'estimation de qualité de tronc d'arbre à partir de la base de données fournies par l'INRA. L'objectif est d'extraire de ces images, des mesures comme le diamètre, la longueur des fissures, la surface de la grume. Pour cela nous utilisons la mire afin de calculer la taille d'un pixel en cm. On peut décomposer la problématique du stage en deux étapes (voir figure 5) :

1. Calculer la conversion de pixel à cm grâce à la mire. Nous avons développé notre propre technique de détection de damier. Cette technique se base sur la détection de droites dans l'espace de Hough et sur une reconstruction de mire avec des segments flou. Cette technique nécessite tout d'abord d'extraire la mire de l'image.
2. Utiliser la conversion calculée pour faire des mesures géométriques sur la grume. Cela nécessite en premier lieu de détecter la grume.

L'extraction des objets d'intérêt (la grume et la mire) est la première étape avant tout traitement. Nous expliquons comment nous procédons à l'extraction dans la prochaine section.

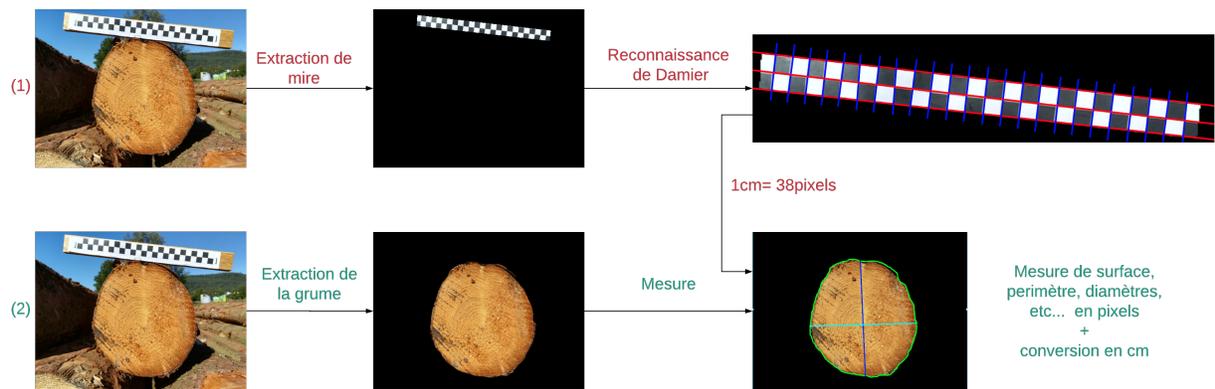


FIGURE 5 – Récapitulatif des étapes du traitement

2 Extraction des objets d'intérêt

2.1 Segmentation de la grume

2.1.1 État de l'art

La segmentation d'objet dans les images consiste à déterminer quels pixels de l'image appartiennent à l'objet d'intérêt. On distingue trois catégories d'approches différentes :

- Les approches basées régions : elles consistent à déterminer des régions de pixels ayant des propriétés communes. Par exemple, le seuillage Otsu [9] cherche le seuil qui minimise la variance intra-classe à partir de tous les seuillages possibles. Cela suppose que l'image contient seulement deux classes d'intensité (le fond et l'objet). L'élargissement de région, est une technique qui consiste à analyser les propriétés (niveau de gris, forme, couleur) des régions autour d'un ensemble de points initiaux. On regroupe ces régions pour en former de plus grosses si elles se ressemblent.
- Les approches basées contours : elles s'intéressent aux variations d'intensité liées au contour de l'objet dans l'image. Des filtres détecteurs de contours, comme Sobel [10] ou plus récemment Canny [11], sont appliqués à l'image. Ils donnent en général des contours bruités et morcelés sauf si les images sont très contrastées. Il faut alors appliquer des techniques de reconstruction par interpolation ou connaître la forme de l'objet à priori.
- Les approches morphologiques : par exemple, la ligne de partage des eaux (ou Watershed en anglais) est une approche qui consiste à considérer l'intensité d'une image comme carte topographique. On fait progressivement monter l'eau [12] dans les bassins et lorsque deux bassins, provenant de sources différentes se rejoignent, une frontière est créée.

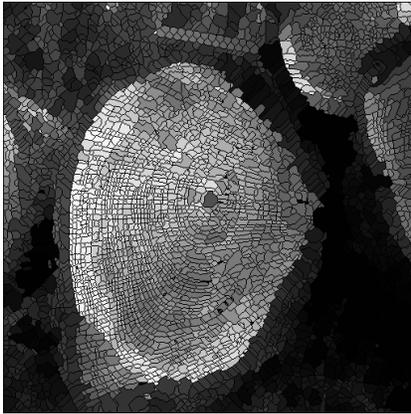
Nous avons essayé des approches comme Otsu, Watershed, Canny sur plusieurs exemples. Comme on peut le voir sur la figure 6, ces méthodes ne sont pas suffisantes pour obtenir la segmentation de la mire. Dans le cas de Otsu, le seuil trouvé automatique ne sépare pas correctement les pixels appartenant à la mire de ceux appartenant à la grume ou au fond. Dans le cas de Canny, la distinction entre les contours de la mire et ceux du fond ne se fait pas facilement. Les régions générées par Watershed sont trop nombreuses et trop petites pour être traitées efficacement, plus on augmente σ pour élargir les zones, plus la mire disparaît de la segmentation. D'autres approches existent et ne se rangent pas dans les catégories précédentes. Par exemple, les réseaux neuronaux convolutifs sont également une approche que nous avons explorée, elle est décrite dans la partie suivante.



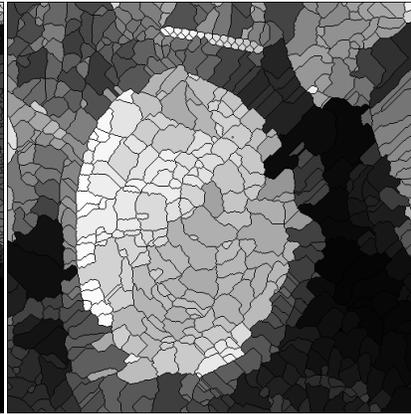
(a)



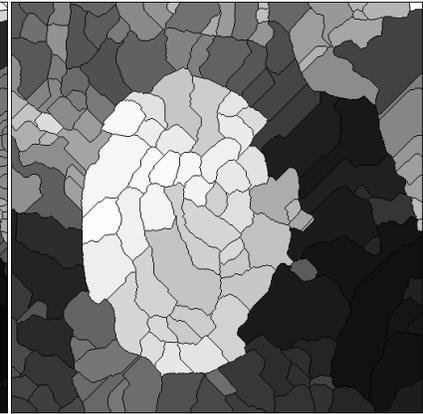
(b)



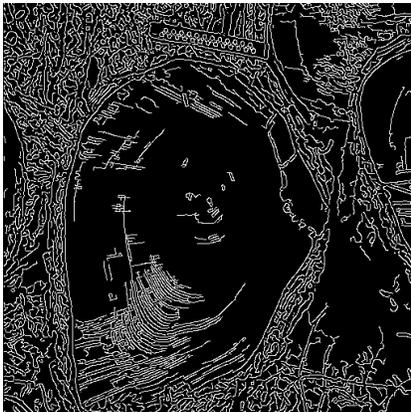
$s = 1.5$



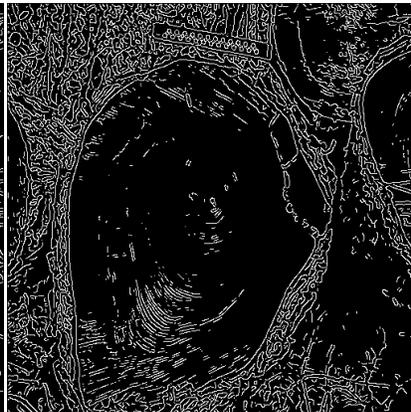
$s = 3$



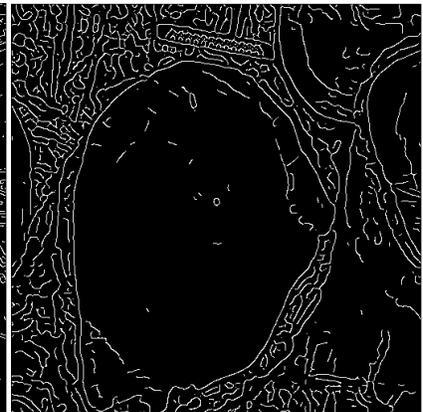
$s = 5$



$t = [0.03; 0.07], s = 1.4$



$t = [0.06; 0.065], s = 1.4$



$t = [0.06; 0.065], s = 3$

FIGURE 6 – Exemples d'images

(a) : image original. (b) : Seuillage Otsu. Deuxième ligne : Segmentation par Watershed avec un sigma de plus en plus élevé. Troisième ligne : Détecteur Canny, avec un sigma de plus en plus élevé et un seuillage de plus en plus restrictif.

2.1.2 Réseaux neuronaux convolutifs

Un réseau neuronal convolutif (ou CNN en anglais), est l'application du concept d'apprentissage profond sur les images. Dans les réseaux d'apprentissage normaux, chaque neurone d'une couche est relié à tous les neurones de la couche suivante. Si on applique le même procédé aux images (des matrices de grande taille), chaque pixel agit comme un neurone. Cela pose problème, car les millions de pixels constituant l'image sont connectés aux millions de pixels de la couche suivante. Dans ce cas, le réseau est trop complexe. On procède en convoluant l'image vers des couches de plus en plus petites. La segmentation traditionnelle d'objets dans les images peut s'avérer longue et compliquée, aujourd'hui des architectures comme U-Net sont développées [1]. Dans cette partie, nous détaillons l'utilisation que nous faisons de l'architecture U-Net pour la segmentation de la mire.

2.1.3 U-Net

L'architecture de U-Net est divisée en deux voies, une voie contractante et une voie expansive (voir figure 7). La voie contractante est constituée d'une suite de convolutions et de max-pooling. Cela permet de contracter l'information contextuelle vers des champs récepteurs de plus en plus petits. La voie expansive est constituée d'Up-conv et de convolution, cela permet de retrouver la taille originale de l'image. Pour garder l'information de localisation, après chaque Up-conv, on concatène les champs récepteurs du même étage.

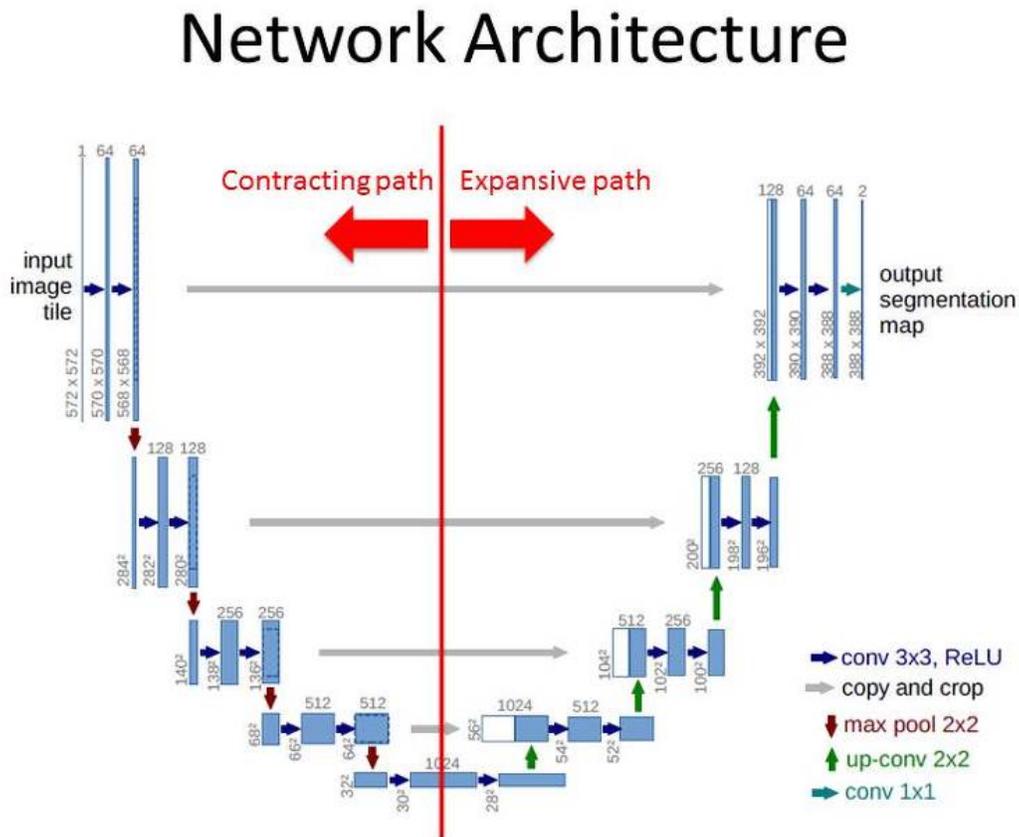


FIGURE 7 – Architecture U-Net [1]

En procédant à de l'augmentation de données les auteurs de l'article [1] arrivent à un taux de reconnaissance de 92% pour la segmentation de cellules. Avant le stage, nous ne possédions pas d'image annotées pour entraîner un modèle, nous nous basons sur U-Net car il nécessite peu d'images d'entraînement pour donner des bons résultats.

2.1.4 Modèle pour la segmentation de grume

L'utilisation de réseaux de neurones pour la reconnaissance de matériaux en bois, a déjà été explorée en [15], la précision va jusque 99.4% dans le cas de l'identification d'espèces d'arbres sur des planches en bois. Remi Decelle, travaille sur un sujet de thèse en lien avec le sujet de stage. Pour résoudre un problème lié à la localisation automatique de moelle dans les images, il a implémenté une architecture CNN inspiré d'U-Net permettant de segmenter la grume. Les changements opérés sur le modèle sont les suivants (et sont résumés en annexe A), nous appellerons ce modèle "256_G.16" :

- Les exemples sont adaptés à la reconnaissance de grume. Avec l'éditeur d'image Gimp, 30 exemples d'entraînement et 30 images de validation ont été tracés à la main.
- Les images en entrée sont de taille 256*256 et sur 3 canaux de couleurs. Cela permet de diminuer le temps d'apprentissage et de prendre davantage en compte la couleur de la grume pour régler les paramètres du réseau.
- Deux opérations de Drop out sont ajoutées à l'architecture du modèle, une après les convolutions de l'avant dernière étape dans la voie contractante, l'autre après les opérations de convolution au centre du réseau. Un Drop out permet d'ignorer certains neurones durant l'apprentissage, on l'utilise pour éviter le sur-apprentissage.
- Une opération de convolution avec une fonction d'activation sigmoïde est placée tout à la fin de la voie expansive. Cela permet d'obtenir une image en niveaux de gris représentant la probabilité d'appartenance d'un pixel à l'objet.
- Le nombre de champ récepteur est diminué à 16 afin de réduire la complexité du réseau.
- Comme dans [1], les données sont augmentées pour entraîner le modèle. Des rotations, zooms, décalages à droite et à gauche ainsi que des symétries miroirs sont appliqués aux images.

Le réseau a été entraîné à partir de 30 images d'entraînement et 30 images de validation. Un exemple de sortie attendue est visible en figure 8. La sortie du réseau est une image en intensité de gris. Plus le niveau de gris est élevé, plus le réseau est sûr de sa réponse. La prédiction se fait en 300 ms pour une précision de 98% et un rappel de 85%. Un exemple de prédiction se trouve également sur la figure 8.

2.2 Segmentation de la mire

2.2.1 État de l'art

Les approches de segmentations classiques ne fonctionnent pas bien pour l'extraction de la mire dans nos images, quelques exemples sont disponibles en figure 6. Cependant dans la littérature, d'autres chercheurs se sont penchés sur le problème de la segmentation de damier dans le but de

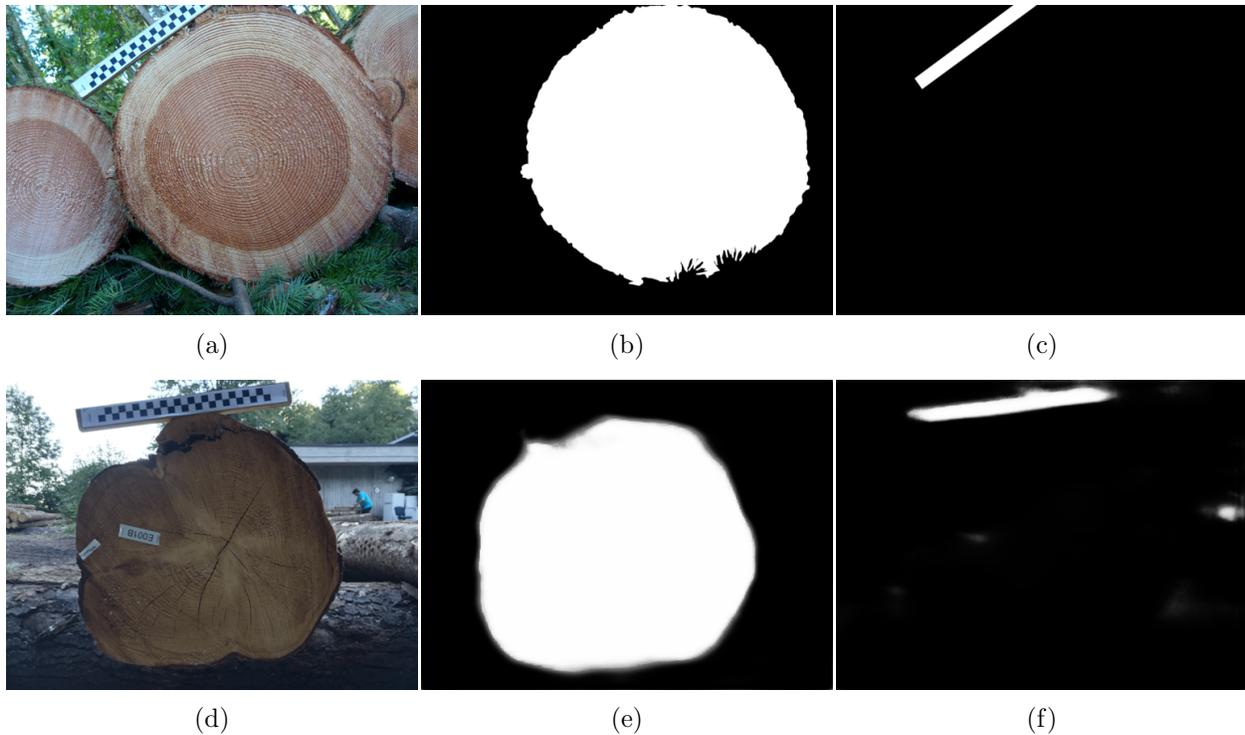


FIGURE 8 – Exemple d’entraînement et de prédiction

(a) : Exemple d’entraînement. (b) : Sortie attendue du réseau pour la segmentation de la grume . (c) : Sortie attendue du réseau pour la segmentation de la mire. (d) : Exemple de test. (e) : Prédiction de la grume. (f) : Prédiction de la mire

déterminer automatiquement les paramètres intrinsèques de caméra. Voici un résumé des approches de segmentation de damier qu’ils ont développées :

- Yu et Peng [5] cherchent un pattern composé de deux triangles dans l’image. Ce pattern ”match” avec les intersections de carré noir. Cette méthode suppose de pouvoir fixer la taille du pattern à rechercher dans l’image. Dans notre cas, la mire peut changer de résolution, ce paramètre semble difficile à fixer.
- Wang et al. [6] détectent des coins de Harris puis filtrent les mauvais candidats par un seuillage sur l’intensité des régions autour des coins. La luminosité changeante au sein de la mire peut rendre la détermination de ce seuil difficile.
- Ruffi et al. [7] binarisent l’image en utilisant une technique de seuillage adaptative puis détectent des quadrangles dans l’image résultante. Cependant, les techniques de seuillage automatique ne fonctionnent pas dans notre cas, car l’image n’est pas constituée de deux classes d’intensité.
- De la Escalera et Armingol [8] appliquent une transformation de Hough sur les coins de Harris pour détecter les lignes du damier. De la même manière que pour Wang et al, Harris ne fonctionne pas toujours dans notre cas. De plus, la détection de ligne avec Hough nécessite que les bons candidats de coin soient plus nombreux que les mauvais, ce n’est pas le cas pour nous.

Nous avons implémenté deux approches pour segmenter la mire. La première s’inspire de U-net. Nous avons repris le réseau développé pour la segmentation de la grume pour l’adapter à la segmentation de la mire. La deuxième est basée sur de la détection de coin puis sur un filtrage des mauvais candidats en étudiant l’intensité des pixels dans une région environnante.

2.2.2 Segmentation avec réseau neuronal convolutif

Comme nous travaillons sur les mêmes images que Rémi, et que son modèle de segmentation donne de bons résultats, nous avons repris son architecture afin de l’adapter à la segmentation de la mire.

DATA

La base d’apprentissage est constituée de soixante images (trente de validation et trente d’entraînement), elles ont été sélectionnées parmi les données de l’INRA. Sur ces images nous avons tracé un rectangle encadrant le damier (un exemple est disponible en figure 8) servant d’exemple à notre réseau. Ces échantillons ont été sélectionnés de sorte à avoir la mire dans différentes orientations, positions et éclairées différemment. Cela constitue la base d’apprentissage nommé BA1. Cependant, plus tard durant le stage, nous nous sommes rendu compte qu’une espèce d’arbre, le Douglas BBF dont un exemple est visible sur la figure 4, avait des résultats de segmentation systématiquement mauvais. En effet, cette espèce d’arbre possède une grume de couleur orangée. Si le réseau n’a jamais rencontré d’exemple où la grume est de couleur orange durant l’entraînement, il est possible que la segmentation de la mire en soit impactée. Pour pallier à cette faiblesse nous avons demandé à l’INRA de nous fournir davantage de données sur cette espèce d’arbre. Nous avons ajouté à la base d’entraînement 48 images de cette espèce (24 d’entraînement et 24 de validation choisies de la même manière que pour la base BA1), nous appellerons cette nouvelle base d’apprentissage BA2.

Afin de mesurer les performances de notre modèle, nous avons construit une vérité terrain de 60 images. Les images ont été sélectionnées de sorte à tromper le réseau (présence de pot de café dans l’image, fond des images bruitées par des objets rectilignes, jeu de perspective sur la mire, mire partiellement visible, etc..). Le but était de rendre la base difficile à segmenter, si le réseau a de bons résultats alors il pourra segmenter la mire dans n’importe quelle condition. De la même manière que pour la base d’entraînement les rectangles encadrant le damier ont été tracés à la main sous GIMP. Cette base s’appelle GT_HARD.

Lorsque que nous avons découvert les erreurs liées aux Douglas BFF, nous avons conçu une deuxième vérité terrain, constituée exclusivement de onze images de Douglas BBF. Le but de cette base est de tester notre hypothèse sur la prise en compte de la grume dans l’apprentissage des paramètres du réseau. Cette base se nomme : GT_BBF_Test.

Enfin, pour mesurer la performance des modèles après que nous ayons découvert la faiblesse de notre réseau dûe aux Douglas BBF, une troisième vérité terrain a été constituée. Cette vérité terrain est la concaténation de GT_HARD et GT_BBF_Test et s’appelle GT_HARDV2. Les informations concernant les différentes bases d’images sont regroupées dans le tableau ci-dessous.

Vérité terrain	GT_HARD	GT_BBF_Test	GT_HARDV2
nombre d’images	60	11	71
description	mélange de plusieurs espèce d’arbre	exclusivement Douglas BBF	GT_HARD \cup GT_BBF_Test

Base d'apprentissage	BA1	BA2
nombre d'images	60 (30+30)	108 (54 +54)
description	mélange de plusieurs espèce d'arbre	BA1 \cup images de Douglas BBF

Mesure de performance du modèle

Pour sélectionner notre modèle, nous comparons la prédiction du réseau aux images des vérités terrain. Nous calculons pour chaque paire une matrice de confusion. Une matrice de confusion est constituée de VP (vrais positif), VN (vrais négatif), FP (faux positif), FN (faux négatif) :

- Un vrai positif est un résultat où le modèle prédit correctement la classe positive.
- Un vrai négatif est un résultat où le modèle prédit correctement la classe négative.
- Un faux positif est un résultat où le modèle prédit incorrectement la classe positive.
- Un faux négatif est un résultat où le modèle prédit incorrectement la classe négative.

Le code permettant de calculer les TP, VN, FP, FN se trouve en annexe B. A partir de ces quatre valeurs on peut calculer deux mesures de performance : La précision et le rappel. La précision mesure combien de pixels appartenant à la mire dans la vérité terrain ont été correctement prédits par le modèle . Le Rappel mesure à quelle proportion la mire à été retrouvée :

$$precision = \frac{VP}{VP + FP}$$

$$rappel = \frac{VP}{VP + FN}$$

Les modèles entraînés

Le premier modèle entraîné (256_M_16) possède exactement la même architecture que 256_G_16 et sa base d'apprentissage est BA1. Nous obtenons 72% de précision et 80% de rappel. Cela confirme le fait que l'on puisse utiliser l'architecture du modèle permettant de segmenter des grumes dans le but d'entraîner un modèle à segmenter des mires. Ces valeurs de précision et rappel servent de base. Tous les modèles possédant un rappel et une précision inférieurs à ces résultats ne sont pas répertoriés ici et sont considérés comme mauvais. L'architecture 256_G_16 prend en entrée des images de taille 256 par 256. Cela est efficace pour segmenter des grumes car les grumes recouvrent une grosse partie de l'image, cependant les mires sont plus petites, lorsque l'on réduit la taille de l'image à 256 par 256 certains carrés du damier ne sont représentés que par 4 pixels.

Le deuxième modèle (512_M_16) possède les mêmes caractéristiques que le premier mais prend comme base d'entraînement des images de taille 512 par 512. Les mesures nous confirment que la taille des carrés du damier est importante pour l'apprentissage des paramètres du réseau puisque la précision et le rappel augmentent tous les deux.

La taille des images permet d’augmenter la précision et le rappel mais allonge beaucoup le temps d’apprentissage du réseau. Le troisième modèle (400_M_16) à avoir été entraîné possède la même architecture que les précédents mais est entraîné avec la base BA2 avec des images de taille 400 par 400. Le temps d’apprentissage réduit, la précision et le rappel augmentent. C’est un résultat logique puisque nous avons augmenté le nombre de données d’entraînement. Mais le résultat est tout de même notable car les images sont plus petites.

Une autre manière de réduire la durée d’apprentissage est de réduire le nombre de paramètres que le réseau doit apprendre. Pour cela on peut réduire la taille du champ récepteur. Le quatrième modèle (400_M_12) à avoir été entraîné possède 12 champs récepteurs à la place de 16, il est entraîné à partir de BA2 sur des images de taille 400 par 400. La précision et le rappel diminuent et le temps d’apprentissage est diminué de moitié.

Enfin, les carrés de la mire que nous cherchons à segmenter sont blancs et noirs. Nous avons entraîné un dernier modèle (400_M_GRAY) en prenant les images de la base BA2 en 400 par 400 convertis en niveau de gris. On obtient un meilleur rappel au prix d’une baisse de la précision.

L’ensemble des mesures de précision, rappel, temps de prédiction et temps d’apprentissage est regroupé dans la partie Résultats, ci-dessous. Plusieurs modèles sont intéressants. Celui que nous avons sélectionné est le modèle nommé 400_M_16 pour son bon pourcentage de précision/rappel et son temps de prédiction faible. Un exemple de prédiction est disponible en figure 8. Les modèles 400_M_16_GRAY et 400_M_12 sont intéressants dans le sens où leurs temps d’apprentissage est relativement faible pour un taux de précision/rappel proche du modèle sélectionné. Ils mériteraient une analyse plus approfondie.

Résultats

GT_HARD	256_M_16	512_M_16	400_M_16	400_M_12	400_M_16_GRAY
précision moyenne (en%)	72.89	76.36	83.06	80.40	77.97
rappel moyen (en%)	80.87	91.90	92.66	88.75	94.92

GT_BBF_Test	256_M_16	512_M_16	400_M_16	400_M_12	400_M_16_GRAY
précision moyenne (en%)	64.21	71.96	87.22	85.07	78.21
rappel moyen (en%)	31.62	76.46	92.46	89.11	94.03

GT_HARDV2	256_M_16	512_M_16	400_M_16	400_M_12	400_M_16_GRAY
précision moyenne (en%)	71.45	75.63	83.75	81.18	78.01
rappel moyen (en%)	72.67	89.32	92.63	88.81	94.77

	256_M_16	512_M_16	400_M_16	400_M_12	400_M_16_GRAY
Base d’apprentissage	BA1	BA1	BA2	BA2	BA2
durée d’entraînement (s)	—	7026	6212	3211	4287
temps de prédiction moyen (ms)	130	502	315	290	300

Les réseaux neuronaux convolutifs sont intéressants car ils permettent de segmenter des objets dans des environnements complexes en réglant minutieusement les paramètres du réseau à chaque

paires d'exemples présentés. Cependant, nous avons vu précédemment que leur apprentissage est long et que leurs performances dépendent énormément des données d'apprentissage. Des exemples de segmentation sont disponibles en annexe C. C'est pourquoi nous avons développé une technique de segmentation basée sur de la détection de coin.

2.2.3 Segmentation en utilisant la détection de coins

Dans cette partie nous nous intéressons à détecter grossièrement la mire dans l'image. Pour cela nous souhaitons détecter les coins des carrés du damier, pour construire une boîte englobante la mire. Le schéma en figure 9 illustre cette méthode.



FIGURE 9 – Illustration de la méthode de segmentation basé sur de la détection de coin.

Harris [2] et Shi-Tomasi [14]

Les coins sont des régions dans l'image avec de grosses variations d'intensité dans toutes les directions. Cette idée a été mathématisée et permet de détecter des régions plates (sans variation brutale d'intensité), des régions contenant une arête (variation d'intensité rectiligne), des régions contenant un coin (variation d'intensité dans plusieurs orientations). La formule pour calculer la variation d'intensité est la suivante :

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2$$

$w(x,y)$ est une fenêtre de taille paramétrable centrée en (x,y) . Utiliser directement cette formule s'avère très long. on utilise l'expansion de Taylor.

$$E(v, v) \approx (u, v) M \begin{pmatrix} u \\ v \end{pmatrix}$$

avec,

$$M = \sum_{x,y} w(x, y) \begin{pmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{pmatrix}$$

I_x et I_y sont les images dérivées sur x et y respectivement (calculés à partir de Sobel, dont la taille du noyaux est également à paramétrer) . Puis, à partir des valeurs propres de M, un score est calculé :

$$R = \det(M) - k(\text{trace}(M))^2$$

En fonction de la valeur de R on classe la région comme contenant une arête, un coin ou aucun des deux. Nous avons utilisé le détecteur de Harris implémenté sous OpenCv. La taille de la fenêtre définissant la région est un paramètre important car si la fenêtre est trop grande on risque de tromper le détecteur. Nous avons fixé ce paramètre à 10 en comptant le nombre de pixels autour d'un coin du damier dans une image où la mire apparaît en petit. Le paramètre servant à calculer les dérivées sur x et y de l'image a été fixé de la même manière à 7. Le détecteur Shi-Tomasi est similaire à Harris en tout point sauf pour calculer le score :

$$R = \min(\lambda_1, \lambda_2)$$

λ_1, λ_2 sont les valeurs propre de M .

OpenCV propose également une implémentation de cet algorithme. Les mêmes paramètres que pour Harris ont été utilisés. Nous avons finalement sélectionné le détecteur Shi-Tomasi car sur certaines images le détecteur Harris donne de mauvais résultats, or la détection étant grossière nous souhaitons qu'elle fonctionne sur n'importe quelle image. Un exemple est disponible en figure 10. Même en utilisant Shi-Tomasi, les coins détectés n'appartiennent pas toujours à la mire. Cela est dû au fond des images qui contient souvent du feuillage ou des intersections de grume qui forment des coins.

Filtrage des mauvais candidats

En simplifiant le problème, les carrés qui constituent la mire sont soit blancs soit noirs. Dans une image en niveau de gris cela se traduit par une valeur de 0 pour les carrés noirs et une valeur de 255 pour les carrés blancs. Les coins détectés par Shi-Tomasi, que nous voulons garder, sont ceux qui appartiennent au damier de la mire. Les régions associées à ces coins ont un ratio de pixels blancs sur noirs bien particulier. Soit, le nombre de pixels blancs est trois fois supérieur au nombre de pixels noirs (1), soit le nombre de pixels blancs est égal au nombre de pixels noirs (2). La figure 11 illustre cette simplification du problème.

En pratique, nous considérons une région rectangulaire autour des points candidats. Un histogramme d'intensité de la région est utilisé pour déterminer si la région "match" (à un seuil près) avec un des deux ratios, si ce n'est pas le cas, le coin candidat est éliminé. A la fin du filtrage nous construisons une boîte englobante de tous les points restants. Nous n'avons pas eu le temps de finir cette méthode de segmentation. Pour l'améliorer, il faudrait considérer une région circulaire, comme dans [16], plutôt que rectangulaire autour des coins candidats, car lorsque la mire n'est pas orientée à l'horizontale, les ratio (1) et (2) ne sont plus respectés. Le temps d'exécution pour retrouver la zone contenant la mire avec cette méthode est d'environ 140ms contre 315ms en moyenne pour le modèle 400_M_16, sans parler du temps d'apprentissage.

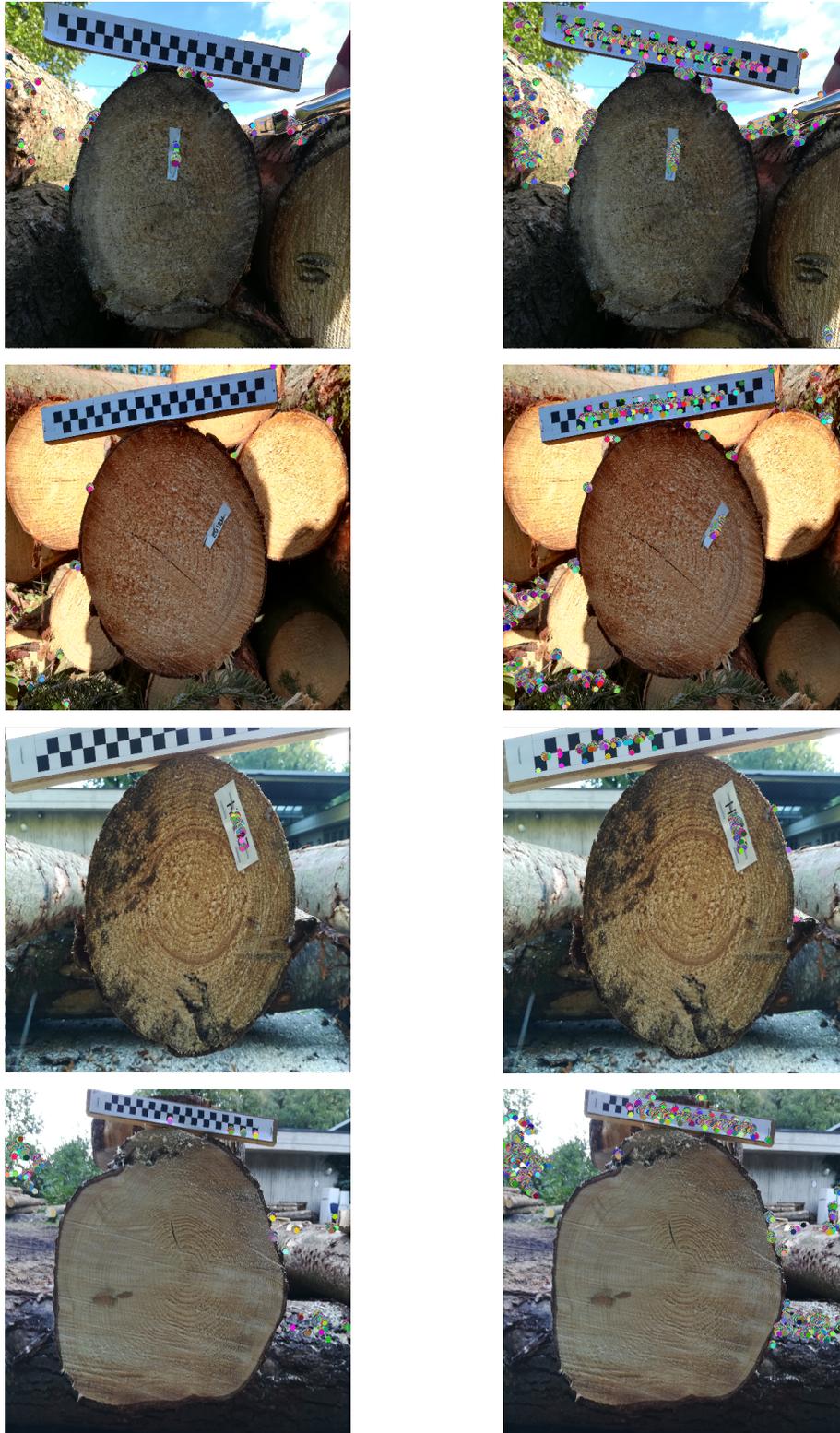
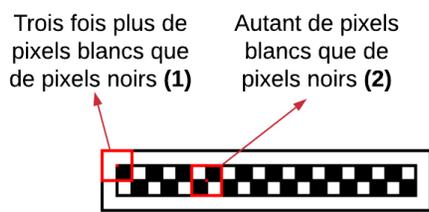
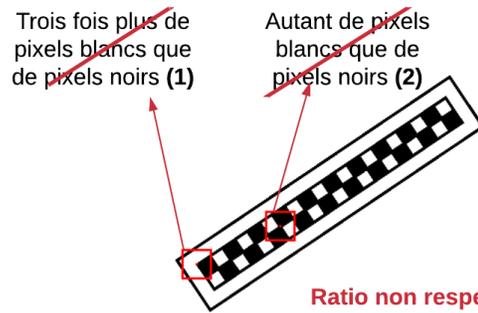


FIGURE 10 – Harris vs Shi-Tomasi

Les ronds de couleurs sont les coins détecté. Première colonne : Exemples de détection de coins avec Harris.
 Deuxième colonne : Exemples de détection de coins avec Shi-Tomasi. Les images présentées sont de taille 512*512



Ratio respecté



Ratio non respecté

FIGURE 11 – Illustration du ratio de pixels blancs et noirs autour des coins détectés par Shi-Tomasi

3 Conversion de pixels à centimètre

En calibration de caméra, on utilise des damiers pour déterminer les paramètres intrinsèques de la caméra, cette opération nécessite de calculer une conversion de pixel à cm. Matlab propose une boîte à outils pour calibrer les caméras, elle a été implémentée par J.Y Bouguet [4] et nécessite que l'utilisateur entre à la main les quatre coins du damier. Cependant, nous avons besoin d'une méthode automatique et l'information sur les coins du damier ne nous est pas disponible directement. La même version est implémentée sous OpenCV avec en plus un algorithme de détection automatique de damier. Cette algorithme prend en entrée les dimensions du damier et impose que le damier soit de taille 4 par 4 au minimum. Notre damier ne respecte pas cette condition et nous ne pouvons pas connaître la taille du damier au préalable, car celui ci peut être partiellement visible. Nous avons développé notre propre méthode de détection de mire en nous basant sur un mélange d'analyse d'images et de géométrie discrète.

3.1 La méthode développée

La segmentation de la mire nous donne une information sur l'emplacement de la zone contenant la mire dans l'image. Nous avons vu jusqu'ici deux méthodes pour obtenir cette zone. Une basée sur un réseau neuronal convolutif, et une autre qui consiste à obtenir une zone grossière autour de la mire. Nous travaillons maintenant dans les rectangles englobant ces zones. Pour obtenir la conversion de pixels à cm automatiquement, une méthode de détection de damier a été développée. Cette méthode consiste à détecter des droites appartenant à la mire dans l'image des contours. Puis nous calculons l'intersection de ces droites pour obtenir le nombre de pixels le long des côtés des carrés.

3.1.1 Contour de la mire

Le contour Canny

RVB (ou RGB en anglais) est un système de codage informatique des couleurs. Il code sur un octet chaque composante rouge, vert, bleu. Cela permet d'avoir un panel de 16777216 de possibilités théoriques de couleur. TSV ou (HSV en anglais) est un autre système de codage de couleur. Une couleur est représentée par trois composantes :

- Teinte : de 0° à 360° . Il s'agit du type de couleur.
- Saturation : de 0% à 100%. Lorsque la luminosité est à 0% on obtient une couleur sombre, et à 100% une couleur intense.
- Valeur : de 0% à 100%. Lorsque sa valeur est à 0% on obtient tout le temps du noir, et à 100% on obtient du blanc/gris en fonction de sa saturation.

Les algorithmes de détection de contours fonctionnent sur des images en niveaux de gris, c'est à dire, des images dont l'intensité peut varier de 0 à 256. C'est sur la composante rouge de la photographie que le détecteur Canny est appliqué. Ce choix a été fait arbitrairement, car aucune des composantes RVB ou TSV ne reflète mieux la mire. Le détecteur Canny est un algorithme de détection de contours. Il s'agit d'un algorithme optimal sur trois critères : bonne détection (faible taux d'erreur), bonne localisation (faible distance entre le contour détecté et le contour réel), non-multiplicité des réponses (une seule réponse par contour). On peut décomposer son fonctionnement en cinq étapes :

1. Réduction du bruit : la première étape consiste à réduire le bruit de l'image avant d'en détecter les contours. On utilise pour cela un filtrage Gaussien donné par la formule ci-dessous. Afin de diminuer le temps de calcul, on utilise en un masque de convolution pour cette étape, plus le masque utilisé est grand moins le détecteur est sensible au bruit et plus l'erreur de localisation grandit.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

2. Gradient d'intensité : après le filtrage, l'étape suivante est d'appliquer un gradient qui retourne les bords de l'objet. En général, pour cela, on utilise un opérateur de cette forme :

$$Dx = \frac{\delta I}{\delta x} = (-1 \quad 0 \quad 1)$$

$$Dy = \frac{\delta I}{\delta y} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

3. Orientation des contours : les orientations des contours sont déterminées par la formule :

$$\theta = \arctan\left(\frac{Dx}{Dy}\right)$$

On obtient une carte des intensités des contours accompagnée d'une carte qui représente l'orientation des contours.

4. Suppression des non-maxima : la carte des gradients indique une intensité en chaque point de l'image. Une forte intensité indique une forte probabilité d'appartenir au contour. Seuls les points correspondant aux maxima locaux sont gardés pour l'étape prochaine. Un maximum local est présent sur les extremas du gradient (où la dérivée s'annule).
5. Seuillage par hystérésis des contours : cette étape nécessite deux seuils, un haut et un bas, qui seront comparés à l'intensité du gradient de chaque point. Si l'intensité du gradient de chaque point est inférieure au seuil bas, le point n'est pas accepté. Si elle est supérieure au seuil haut, le point est considéré comme appartenant au contour. Si elle est entre les deux, le point est accepté s'il est connecté à un point déjà accepté.

Matlab dispose d'une interface permettant d'utiliser le détecteur Canny. Le paramètre σ vaut par défaut $\sqrt{2}$ et les seuils haut/bas de la dernière étape sont calculés automatiquement. Le détecteur Canny est appliqué sur une boîte englobant la segmentation. Un schéma récapitulatif des étapes pour obtenir la zone Canny est disponible en figure 12.

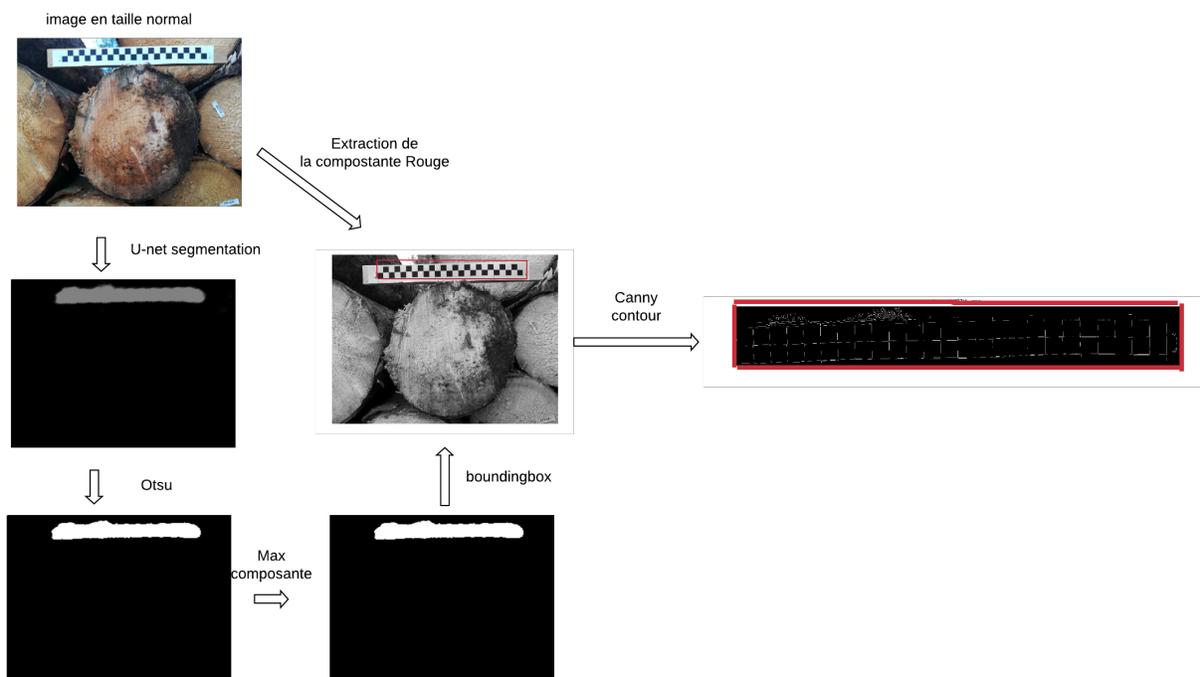


FIGURE 12 – Pipeline pour obtenir le contour de la mire par Canny

Le contour Canny nous donne une image binaire. Les pixels blancs sont les contours détectés et le reste de l'image est noir, des exemples de contours détectés sont disponibles dans les figures 19 et 12 . On cherche à partir de ces contour à retrouver les droites constituant la mire.

3.1.2 Détections des droites de la mire

Espace de Hough

La transformée de Hough [3] permet de détecter des droites. La méthode proposée par Hough cherche à détecter l'alignement d'un ensemble de points dans une image binaire. Pour chaque pixel (x, y) dans l'espace primal cartésien on définit une sinusoïde dans l'espace dual.

$$\rho = \cos(\theta)x + \sin(\theta)y$$

L'espace dual est défini en coordonnées polaire et il est constitué de cellules appelés "accumulateur". La transformée de Hough propose de remplir cette grille d'accumulation de sorte que les cases qui contiennent un gros score sont celles dont les coordonnées correspondent aux plus longues droites dans l'espace primal. Un schéma pour résumer la construction de l'espace de Hough se trouve en figure 13.

Il existe de nombreuses implémentations de la transformée de Hough. Pour notre détection des droites de la mire nous avons utilisé celle de Matlab. Une capture d'écran de la transformée de Hough associée aux pixels nettoyés du contour Canny se trouve également en figure 15. On y voit un ensemble de sinusoïdes se recouper en plusieurs points formant un pattern spécifique à notre mire. Un niveau de gris est associé à chaque valeur de l'espace accumulateur. Plus la valeur de

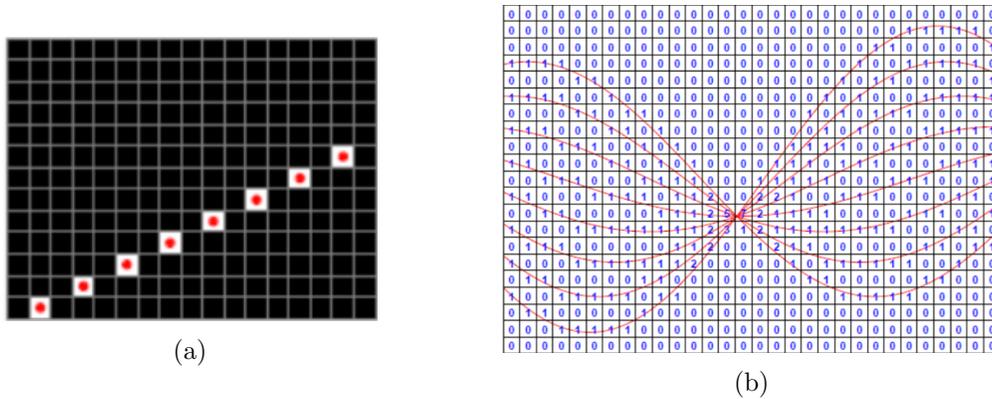


FIGURE 13 – Espace de Hough

(a) : Espace primal, 8 pixels blanc (schéma issu du cours TAI, M2 AVR). (b) : Espace dual, 8 sinusôides (schéma issu du cours TAI, M2 AVR).

l'accumulateur est élevée, plus le niveau de gris est important. Les zones blanches sont les points correspondant aux droites les plus longues de notre image des contours.

Droites de Hough

Les droites horizontales sont les trois droites qui relient les carrés noirs du plus long côté de la mire. Les droites verticales sont les droites perpendiculaires aux droites horizontales qui relient les deux carrés noirs tout le long de la largeur.

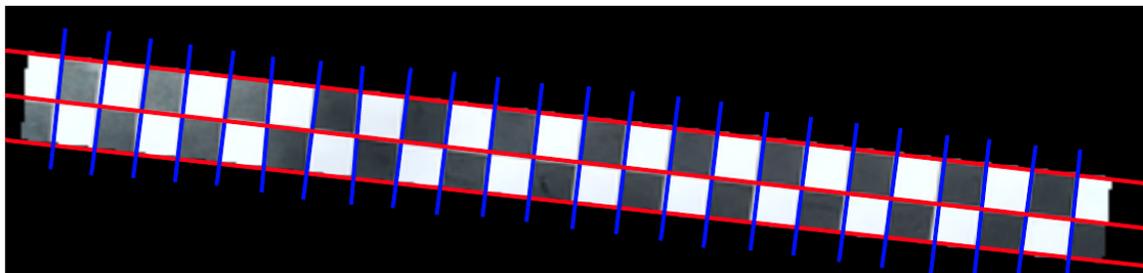
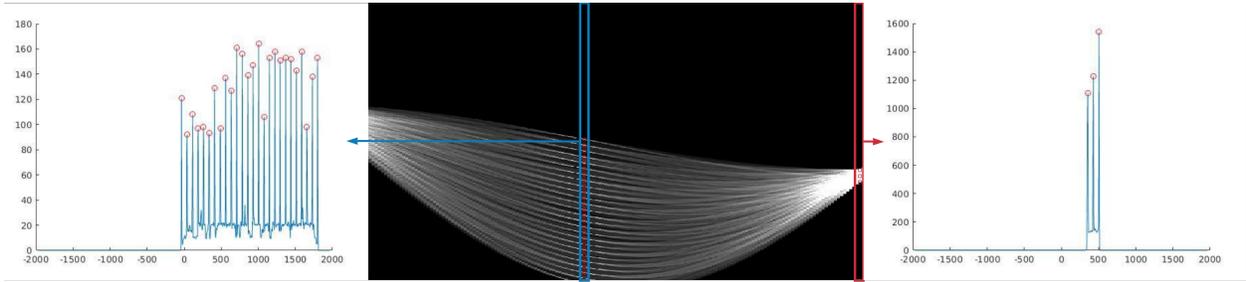


FIGURE 14 – Illustration des droites horizontales et verticales
En bleu : les droites verticales. En rouge : les droites horizontales.

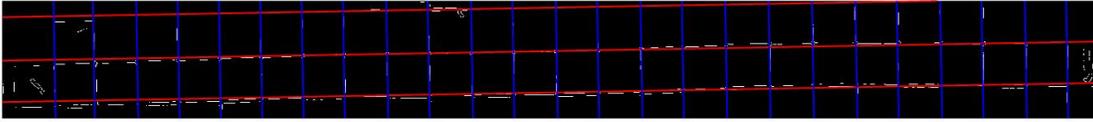
Nous faisons l'hypothèse que l'accumulateur ayant reçu le plus de vote dans l'espace de Hough (donc le point le plus blanc), correspond forcément aux paramètres (ρ_h, θ_h) d'une des trois droites horizontales de notre mire. Partant de cette hypothèse, nous avons déjà l'orientation θ_{h1} et la distance à l'origine ρ_{h1} d'une des droites horizontales. Les deux autres droites horizontales sont parallèles à la première, donc leurs θ_h respectif est égale à θ_{h1} , pour obtenir leurs ρ_h , il suffit de trouver, dans le vecteur correspondant à θ_{h1} de l'espace de Hough, les deux autre maximum locaux. Cette opération est représentée en rouge (à droite) dans la figure 15.

Les droites verticales sont perpendiculaires aux droites horizontales. De ce fait, leurs θ_v respectif vaut $\theta_{h1} + 90^\circ$. De la même manière que pour les droites horizontales, on recherche dans le vecteur correspondant, les maximums locaux. Cette opération est représentée en bleu (à gauche) dans la figure 15.

On obtient ainsi les paramètres (ρ, θ) des droites orthogonales appartenant à la mire. Un visuel de ces droites est disponible en 15. Cependant, lorsque la mire réelle (l'objet photographié) est un peu déformée, les droites construites à partir de ces paramètres ne représentent pas bien la mire. Pour prendre en compte la déformation de la mire, nous construisons des segments flous à partir de pixels proches des droites, puis nous calculons leurs intersections.



(a)



(b)

FIGURE 15

(a) : Espace de Hough calculé sur l'image des contours. (b) : En rouge et bleu, les droites détectées

3.1.3 Reconstruction de la mire par segment flou

Définition d'une droite discrète [17]

Une droite discrète de paramètre (a, b, μ) et d'épaisseur arithmétique μ est définie comme l'ensemble des points entiers (x, y) vérifiant la double inégalité :

$$\mu \leq ax - by < \mu + \omega$$

avec a, b, μ, ω dans \mathbb{Z} ; a, b sont premier entre eux et $\frac{a}{b}$ pente de la droite. On note $D(a, b, \mu, \omega)$.

Définition d'un segment flou [13]

Soit S_f une suite de point dans \mathbb{Z}^2 . Une droites discrète $D(a, b, \mu, \omega)$ est dite englobante pour S_f si tous les points de S_f appartiennent à D .

Une droite englobante $D(a, b, \mu, \omega)$ d'une suite de points entiers S_f est dite optimale si son épaisseur

verticale est égale à l'épaisseur verticale de l'enveloppe convexe de S_f .

Soit S_f une suite de points entiers, S_f est un segment flou d'épaisseur v si et seulement si sa droite optimale englobante a une épaisseur verticale inférieure ou égale à v .

Nous construisons un segment flou pour chaque droite obtenue dans l'espace de Hough. La propriété optimale des segments flous nous permet d'être sûr que le rectangle qui englobe les pixels est d'épaisseur minimal. Nous calculons ensuite l'intersection de deux segments flous comme l'intersection de leurs droites centrales.

Création des segments flous

Un segment flou s'obtient à partir d'une suite de point entier. Pour obtenir cette suite, nous avons mis au point un algorithme dont l'idée est d'associer à chaque cellule de la discrétisation de la droite étudiée, un unique pixel de l'image des contours. Cet unique pixel doit être le plus proche possible de la droite étudiée. Pour cela nous déplaçons un segment de droite discrète, orthogonale à la droite étudiée, le long de la discrétisation. Un exemple du résultat de cet algorithme est disponible en figure 16.

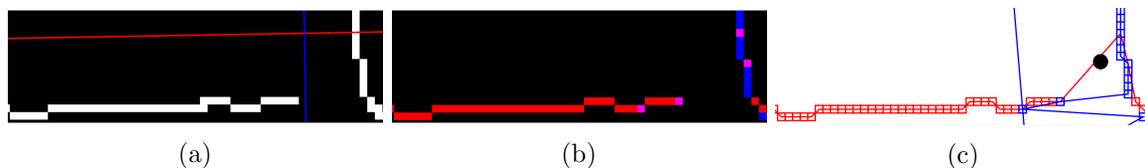


FIGURE 16 – Droites de Hough vs Pixels de Hough

(a) : zoom sur une zone de l'image des contours, le point d'intersection entre droite horizontales et droite verticale est peu précis. (b) : En rouge, les pixels associés à la droites horizontales, en bleu le pixel associé a la droite verticale et en rose les pixels associés aux deux. (c) : point d'intersection de segment flou, les droites limites ne sont pas visible à cause du zoom sur le point d'intersection.

On construit pour chaque droite, son segment flou, on peut voir sur la figure 17 un exemple de mire reconstituée à partir de segments flous. On sait que les côtés des carrés mesurent un cm en réalité. Ce que l'on cherche à obtenir, c'est le nombre de pixels le long de ces côtés. On peut l'obtenir de deux façons avec notre reconstitution de la mire :

- La distance verticale D_v , distance moyenne entre points d'intersection des segments flous verticaux avec les segments flous horizontaux.
- La distance horizontale D_h , distance moyenne entre points d'intersection des segments flous horizontaux avec les segments flous verticaux.

Pour calculer une intersection de segments flous, on considère la droite centrale du segment flou comme sur le schéma en figure 17. On obtient finalement deux mesures de la conversion de pixel à cm pour chaque image : les distances verticales D_v et les distances horizontales D_h . Nous détaillons dans la partie suivante les tests que nous avons faits pour déterminer laquelle de ces deux mesures correspond le mieux à l'échelle réelle.

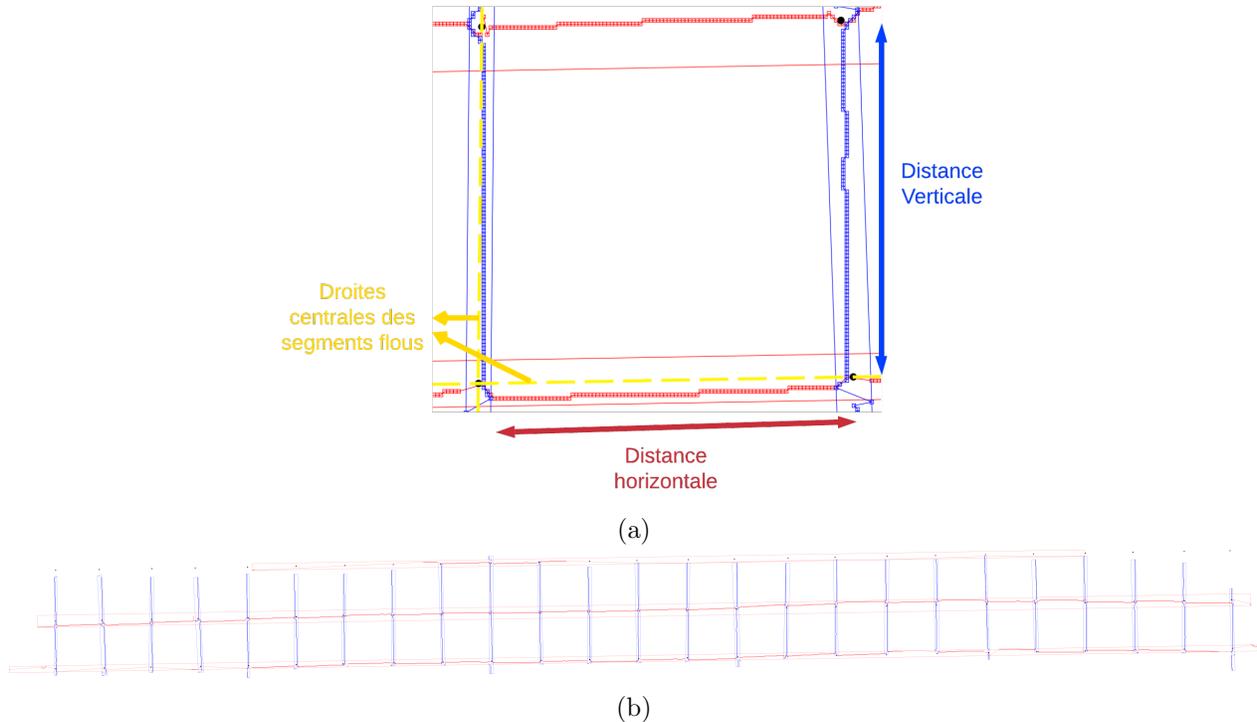


FIGURE 17 – Exemple de segment flou

(a) : Zoom sur un carré de la reconstitution de la mire. (b) : Reconstitution de la mire par segments flous.

3.1.4 Expérimentations

Vérité terrain

L'INRA a mesuré le nombre de pixels le long des côtés des carrés de la mire sur plusieurs images. Le logiciel qui a été utilisé pour cela est ImageJ. Les mesures sont prises à la main, des erreurs liées au choix de l'emplacement des coins des carrés pour mesurer l'échelle sont possibles. Obtenir une erreur nulle est impossible. D'une part, nous comparons D_v et D_h à l'échelle mesurée par l'INRA pour déterminer à quel point notre méthode est précise, d'autre part nous souhaitons déterminer laquelle des deux mesures, D_v et D_h , représente le mieux l'échelle mesurée par l'INRA. Nous possédons quatre bases :

- Douglas BBF (8 images) : Les grumes de cette base ont une couleur orangée et le blanc de la mire est très proche de la couleur du ciel.
- Douglas BESLE (11 images) : Les grumes de cette base sont généralement plus grosses. L'opérateur doit s'éloigner pour prendre la photographie. Ce faisant, la mire apparaît en plus petit dans l'image.
- Corcieux Lumix (99 images) : La couleur des grumes de cette base varie du jaune pâle au gris clair. Le fond des images est essentiellement obstrué par d'autres grumes.
- Fva Huawei (100 images) : La couleur des grumes varie du rose très clair à du jaune grisé. Le fond de l'image est un environnement forestier. Dans la majeure partie des photos, la grume est isolée des autres.

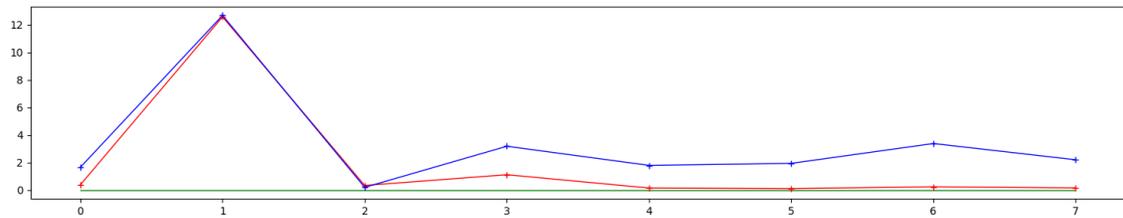
La précision de notre méthode nous est donnée par l'écart entre l'échelle calculée et l'échelle de vérité terrain. Nous calculons une moyenne et un écart type de l'erreur commise sur chaque base.

Les résultats sont regroupés dans le tableau ci-dessous. La mire est toujours la même sur les quatres bases, cependant l'environnement lumineux et le fond de l'image changent entre les bases. Nous avons séparé les tests pour déterminer si notre détection est résiliente à ces changements.

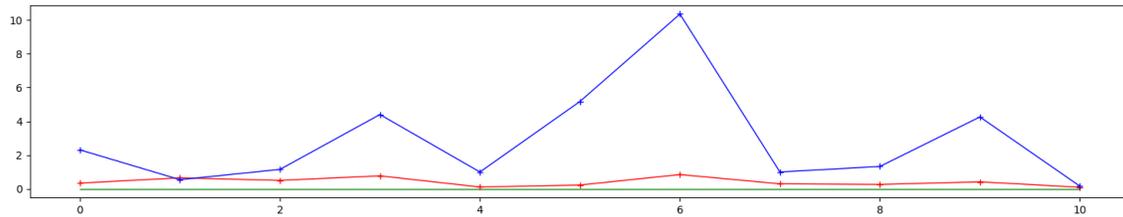
Résultats

Distances Horizontales	Douglas BBF	Douglas BESLE	Corcieux Lumix	Fva Huawei
taux d'erreur	0.025	0.007	0.0106	0.0115
écart type	4.049	0.242	1.055	1.106

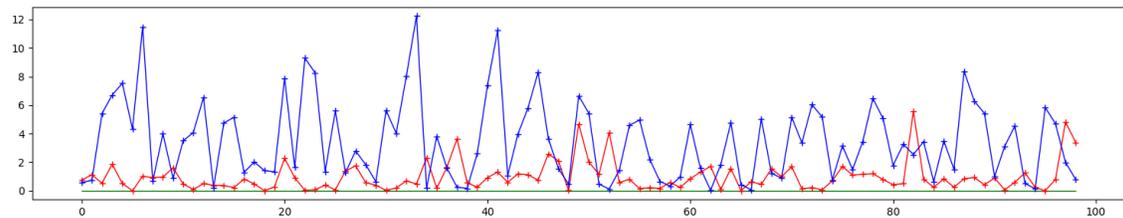
Distances Verticales	Douglas BBF	Douglas BESLE	Corcieux Lumix	Fva Huawei
taux erreur	0.046	0.0572	0.050	0.0574
écart type	3.630	3.021	2.842	2.244



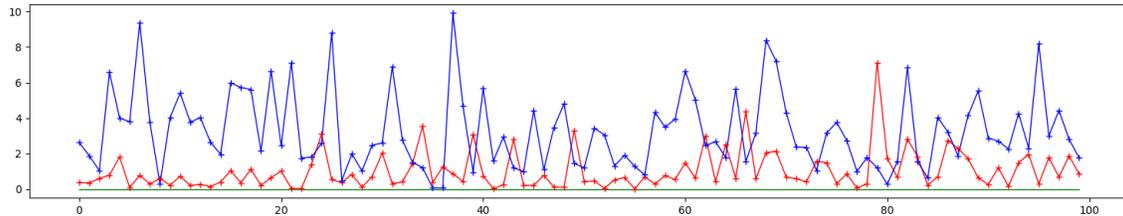
Douglas BBF



Douglas Besle



Corcieux Lumix



Fva Huawei

FIGURE 18 – Erreur en pixel par cm sur les différente base
En bleu, les distances verticales. En rouge, les distances horizontales. Chaque point représente l'erreur pour une image.

Interprétation

Pour les distances horizontales, on constate que l'erreur en nombre de pixel par cm et l'écart type sont de l'ordre de 1% sur chaque base, sauf pour la base Douglas BBF qui possède une erreur de 2.5%. La base Douglas BBF, est constituée seulement de 8 images. Si on analyse le graphique en figure 18, on comprend que le taux d'erreur est élevé à cause d'un seul échantillon. Le damier de cette image n'est pas droit, cela joue sur la détection des droites dans l'espace de Hough. Pour les distances verticales, on a une erreur de l'ordre de 5% et un écart type plus élevé. L'erreur plus élevée pour les distances verticales est dû à la forme de la mire. En effet, les distances verticales sont calculées avec l'intersection entre les segments flous verticaux avec les segments flous horizontaux, sauf que la mire n'est reconstituée qu'à partir de trois segments flous horizontaux. Si un des trois segments est faussé, l'ensemble des distances verticales l'est également.

Sur la figure 18, on voit l'erreur commise par notre méthode (en nombre de pixels par cm) pour chaque échantillon des bases fournies par l'INRA. En rouge, les erreurs sur les distances horizontales et en bleu, les erreurs sur les distances verticales. L'erreur sur les distances horizontales est plus faible en moyenne que pour les distances verticales. Nous utiliserons les distances horizontales pour obtenir la taille d'un pixel en cm.

Les erreurs semblent à peu près équivalentes sur les différentes bases, cela laisse sous-entendre que notre méthode pour détecter les damiers est assez résiliente pour pouvoir être utilisée sur d'autres images du même projet.

Nous avons également mesuré le temps d'exécution de notre méthode. Notre méthode met environ 348 secondes à traiter une base de données de 100 images. 164 secondes sont nécessaires à la segmentation, la construction des droites de Hough et l'extraction des pixels associés nécessite 128 secondes, le calcul de l'intersection des segments flous demande 56s.

3.1.5 Suppression du bruit

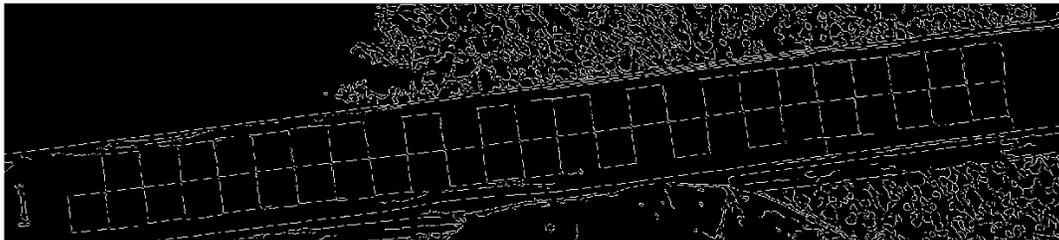
Les méthodes de segmentation de la mire que nous avons mises en place sont peu précises, en appliquant Canny sur ces régions résultantes, on capture quelque fois les contours du fond de l'image. Du fait que le fond de l'image est constitué de branchages et feuillages, ce contour est dense comparé à celui de la mire, nous appelons ce mauvais contour du "bruit". Un exemple est visible en figure 19.

Nous souhaitons retirer ce bruit car il peut nuire à la détection des droites de la mire. En analysant la forme du bruit, nous avons remarqué que ce bruit est complètement désordonné, anarchique, comparé aux contours de la mire. Nous avons défini la notion de pixel dense : un pixel dense est un pixel appartenant au contour de l'image dont le ratio de pixels blancs dans une fenêtre autour de celui-ci dépasse un seuil. Nous avons fixé ce seuil de sorte à supprimer le moins de pixel parmi la mire. Or les pixels les plus denses parmi la mire sont ceux situés à l'intersection des carrés noirs. Le motif formé par le contour à cette intersection est toujours en forme de croix. Le nombre de pixel parmi cette croix est une borne supérieure du seuil permettant de considérer un pixel comme étant dense.

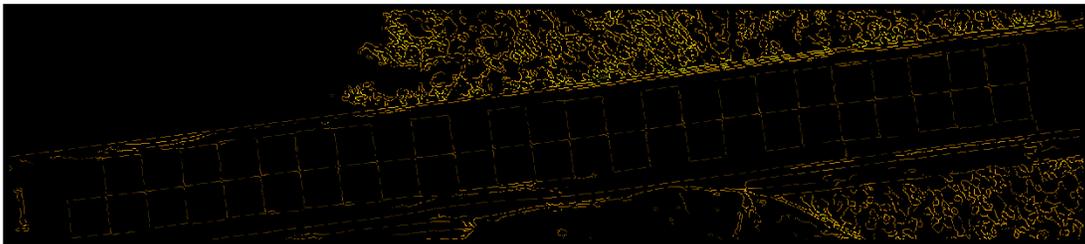
Si on applique la suppression du bruit à toutes les images, y compris celles où la segmentation est précise, on risque de supprimer des pixels appartenant à la mire. Nous appliquons la suppression du bruit seulement aux images dont le pourcentage de pixels denses est supérieur à 40%.



(a)



(b)



(c)

FIGURE 19 – Densité de de pixel

(a) : Composante Rouge de l'image. (b) : résultat du détecteur Canny sur la zone segmentée. (c) : map de densité, plus le jaune est intense, plus le pixel est dense.

4 Mesures géométriques sur grume de bois

Nous avons vu précédemment, l'utilisation d'un CNN pour extraire les grumes de bois dans les images et une méthode permettant de calculer la taille d'un pixel en cm. Nous travaillons maintenant sur les régions de grume afin de faire des mesures géométriques.

4.0.1 Mesures géométriques

Plusieurs mesures ont été demandées par l'INRA : le périmètre de la grume, sa surface, son diamètre, l'écart entre le centre géométrique et le centre des cernes de la grume (la moelle). Nous avons développé un script prenant en entrée, la région de la grume, l'échelle de conversion pixel à cm et le centre de la moelle qui permet de renvoyer à l'utilisateur les mesures souhaités.

La sortie du CNN nous permet d'obtenir une image en niveau de gris représentant la probabilité d'appartenance d'un pixel à la grume. Nous appliquons Otsu dessus afin d'obtenir une image binaire : Le fond et la grume. A la suite de ce seuillage plusieurs composantes connexes apparaissent dans l'image, on sélectionne celle qui possède le plus grand nombre de pixels en considérant qu'il s'agit forcément de la grume.

Périmètre

OpenCV dispose d'une fonction permettant d'extraire les contours d'une image binaire. Cette fonction utilise l'algorithme inventé par Satoshi Suzuki en [18]. En sortie de cet algorithme on obtient un contour sous forme de liste de coordonnées entières. Dans le cas d'un contour 4-connexe (en considérant que les voisins NORD,EST,SUD,OUEST) on obtient le périmètre de la grume par le nombre de pixels appartenant aux contours multiplié par l'échelle de conversion de pixels à cm. Dans le cas d'un contour 8 connexes (en considérant les voisins diagonaux), on calcule le périmètre de la même manière que pour le contour 4-connexes sauf lorsque deux pixels sont voisins par une des directions diagonales, on multiplie la taille du pixel en cm par $\sqrt{2}$. en figure 20 il y a un exemple de contour 4 et 8 connexe.



FIGURE 20 – Contours
(a) : exemple de contour 4 connexe. (b) : exemple de contour 8 connexe.

Surface

La surface d'une forme dans une image binaire est le nombre de pixels constituant la forme. On compte simplement les pixels dont l'intensité est différente de zéro. On obtient la surface en cm^2 en multipliant le nombre de pixels constituant la forme par l'échelle de conversion pixel à cm au carré.

Diamètre

Le diamètre d'une grume est le plus grand segment passant par le point de la moelle (M) qui intersecte le contour de la grume. Pour l'obtenir, on considère deux points I, J , positionnés sur le même point du contour. On déplace J le long du contour de sorte à aligner (I, J, M) . Ces trois points sont alignés si la pente formée par le segment (IM) est égale à la pente formée par le segment (MJ) :

$$\frac{y_1 - y_2}{x_1 - x_2} = \frac{y_2 - y_3}{x_2 - x_3}$$

$(x_1, y_1), (x_2, y_2), (x_3, y_3)$ les indices respectifs de I, M et J . On applique le produit en croix :

$$((y_1 - y_2) * (x_2 - x_3)) = ((y_2 - y_3) * (x_2 - x_1))$$

Si cette condition est vraie (J' sur le schéma), on retient l'alignement, on incrémente I , et on recommence à déplacer J le long du contour. Parmi tous les alignements de (I, J, M) on sélectionne le plus grand, en calculant la distance euclidienne en I et J .

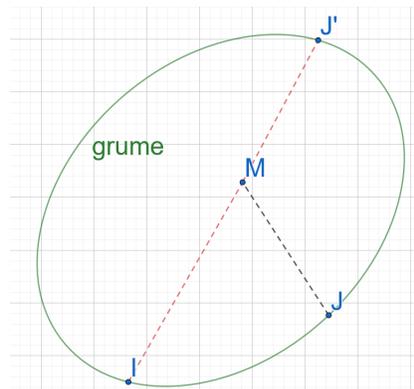


FIGURE 21 – Schéma du calcul du diamètre

En vert : le contour de la grume. En pointillé rouge : IM et MJ' ont la même pente

Normal au diamètre

L'équation de la normale s'obtient à partir de l'équation du diamètre en inversant la pente et en centrant le segment sur la moelle. On cherche l'intersection de la normale avec le contour pour obtenir sa longueur.

Écart entre centre géométrique et centre de la moelle

Le centre de la moelle nous est donné en entrée du script, le centre géométrique s'obtient en calculant le barycentre de la forme. On utilise la distance euclidienne pour mesurer l'écart entre ces deux centres. En figure 24 nous avons rassemblé plusieurs visuels des mesures.

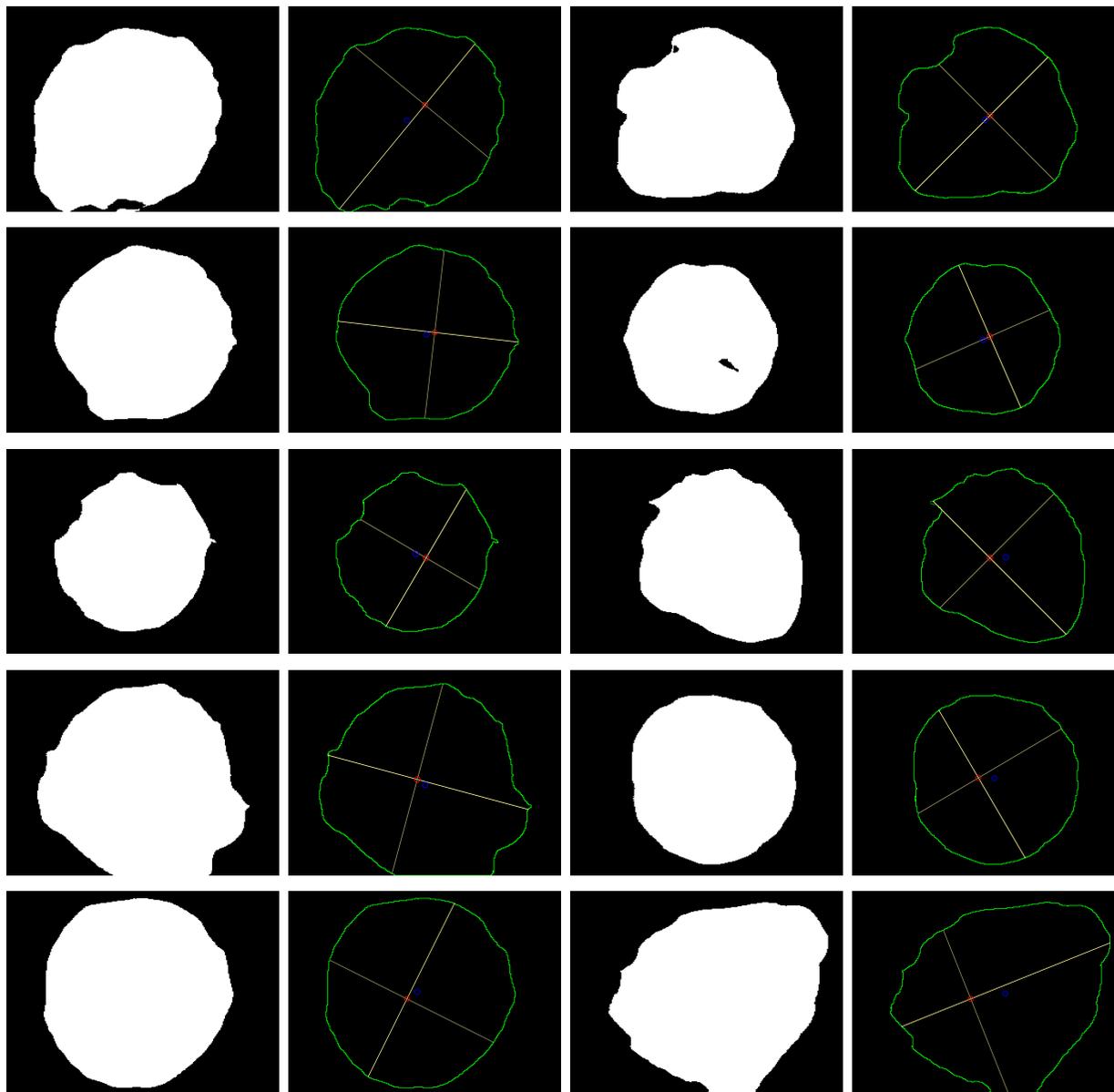


FIGURE 22

première et troisième colonne : segmentation de la grume, deuxième et quatrième colonne : visuel des mesures : En vert le contour (élargi en utilisant Gimp), en gris clair le diamètre, en brun la normal au diamètre, en rouge le centre de la moelle, et en bleu le centre géométrique

5 Implémentation

5.1 Technologies utilisées

Python

Python est un langage interprété orienté objet qui supporte l'héritage multiple. Il se mani assez facilement comparé à d'autre langage. En forçant une tablature propre, python à l'avantage d'être très lisible. Ce langage est de plus en plus utilisé en entreprise comme en recherche, la communauté est donc diversifiée et très active.

Numpy est une extension de Python qui permet de faire des calculs efficaces sur des matrices. Par exemple, on peut faire des opérations logiques entre matrice. Nous avons utilisé cette librairie pour faire des mesures de précision et de rappel sur la segmentation des objets d'intérêt.

Tensorflow est un outil d'apprentissage automatique développé par l'équipe Google Brain. Le code source est libre depuis 2015. Initialement, le but de Tensorflow était d'optimiser les calculs numériques complexes, mais aujourd'hui il est très connu pour résoudre des problèmes de deepLearning.

Keras est une librairie Python qui encapsule l'accès aux fonctions proposées par plusieurs librairies de machine learning, en particulier TensorFlow. Nous utilisons Tensorflow et Keras pour la segmentation des objets d'intérêt.

Matlab

Matlab est un langage de script émulé dans un environnement de travail du même nom. Il est utilisé à des fins de calcul numérique. Il offre un aspect pratique au calcul en permettant de manipuler facilement des matrices.

Image Processing toolbox propose un ensemble complet d'algorithmes standard en traitement d'images. On peut faire facilement des opérations de segmentation d'image, de réduction de bruit, etc... Cette toolbox nous permet, de faire un ensemble de post-traitement sur la segmentation de la mire et de construire l'espace de Hough.

C++

C++ est un langage de programmation compilé permettant la programmation sous de multiples paradigmes. Il fait partie des langages de programmation les plus utilisés actuellement, il possède une communauté très active, plus encore que Python. Il est compatible avec la plupart des librairies qui fonctionnent en C.

DGTal est un projet collaboratif qui vise à développer des algorithmes et des outils géométriques génériques et efficaces. Il se présente comme une librairie c++ open Source. Nous utilisons cette librairie pour reconstruire la mire avec des segments flous.

OpenCV (pour Open Computer Vision) est une librairie graphique libre, initialement développée par Intel, spécialisée dans le traitement d’images en temps réel. Des algorithmes d’extraction de contour et de recherche de composantes connexes sont utilisés pour faire des mesures sur la grume

5.2 Solution apportée

A l’issue de ce stage, deux scripts et un plugin ImageJ ont été fournis à l’équipe :

- Un script bash permettant de calculer la conversion de pixel à cm sur un ensemble d’images. Il prend en entrée un répertoire d’images et exécute trois sous scripts. Le premier est en python, il permet de faire la segmentation de la mire. Le deuxième est fait sous matlab, il utilise la segmentation de la mire et les images de la base de données pour extraire les pixels appartenant à la mire. Le troisième est codé en C++, il utilise les librairies DGTal et OpenCV pour reconstruire la mire avec des segments flous et calcule l’échelle de conversion de pixels à cm. Le schéma 23 résume le fonctionnement de ce script.
- Un script C++ permettant de faire des mesures sur les grumes de bois. Il prend en entrée une image, son échelle de conversion de pixels à cm, et les coordonnées de la moelle pour calculer un ensemble de mesures.
- ImageJ est un logiciel open source de traitement et analyse d’image. Un plugin ImageJ a été développé permettant de prendre en compte l’échelle de conversion de pixels à cm lorsque des mesures sont effectuées avec les outils du logiciel.

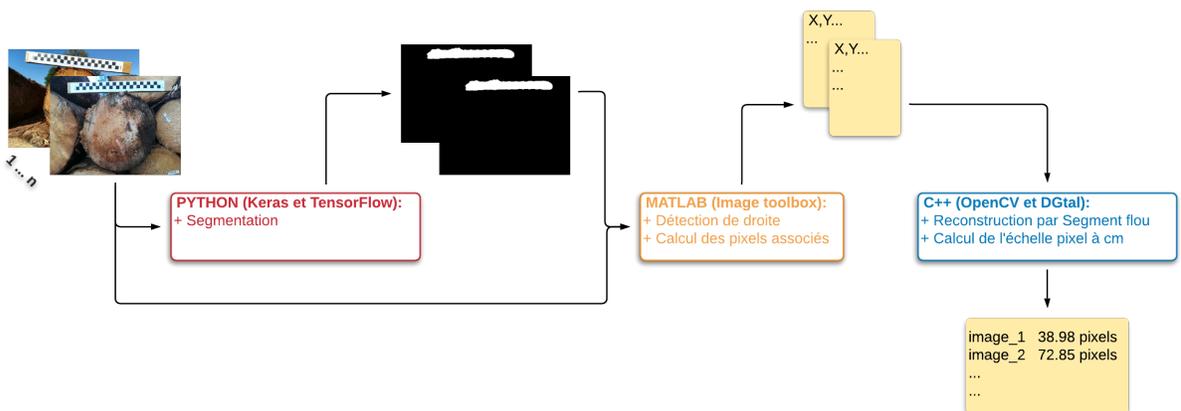


FIGURE 23 – Fonctionnement du script bash pour la conversion de pixel à cm

6 Conclusion

A travers ce stage, nous avons mis en place une chaîne complète pour convertir la taille d'un pixel en cm à partir d'images contenant une mire : segmentation des images, reconnaissance de damier, reconstruction de la mire. Nous avons utilisé une approche basée sur de la détection de droite dans l'espace de Hough et de la géométrie discrète pour reconstruire la mire.

Les résultats montrent que l'approche est utilisable dans ce contexte avec un taux d'erreur de l'ordre de 1% et un temps d'exécution de 348 secondes pour le traitement de 100 images. Un script permettant de faire des mesures sur grumes à été développé à la suite de ces résultats.

Cependant, plusieurs points peuvent encore être améliorés. D'une part, la méthode de segmentation basée sur un réseau neuronal convolutif coûte environ 315ms par prédiction. Par conséquent un développement plus approfondi de la méthode de segmentation basé sur de la détection de coin pourrait réduire le temps de segmentation. D'autre part, les script pour faire des mesures sur la grume traite des images individuellement, il serait intéressant d'intégrer le script permettant de faire des mesures sur la grume au reste de la chaîne. Ainsi, des statistiques sur les caractéristiques géométriques de la grume seraient possibles.

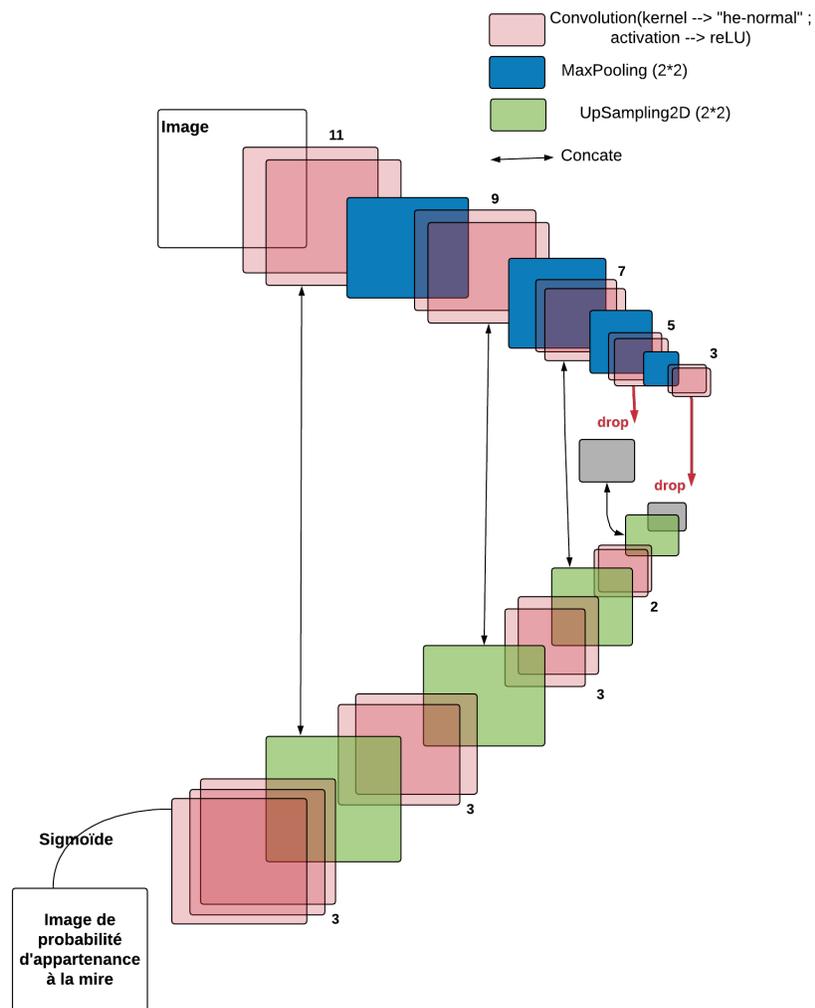
Dans le cadre des métiers du bois, il est nécessaire de déterminer préalablement et avec le plus de précision possible la qualité du bois utilisé. Ce contrôle qualité est actuellement réalisé "à l'oeil" (on détermine la qualité du bois à partir de quelques mesures géométriques visibles à l'extrémité de la grume). Par conséquent, aussi laborieuse et rigoureuse qu'apparaisse cette expertise, on ne peut que déplorer un surplus non négligeable de bois gâché. Ces mesures approximatives pourraient être automatisées par la chaîne que nous avons mis en place, ce qui affinerait l'estimation de la qualité. De ce fait, ce projet présenterait un double avantage : Non seulement un avantage en terme d'efficacité (estimation précoce de la qualité du bois en amont de la chaîne de production), mais aussi en terme écologique (réduction du gaspillage inhérent à la mauvaise estimation de la qualité de la grume ; ce qui entraîne une réduction de l'impact de l'homme sur son environnement).

Bibliographie

- [1] O. RONNEBERGER, P. FISCHER, T. BROX. U-Net : Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention*, pages 234-241, 2015.
- [2] C. HARRIS, M. STEPHENS. A combined corner and edge detector. *In Proc. of Fourth Alvey Vision Conference*, pages 147-151, 1988.
- [3] P.V.C. HOUGH. Machine analysis of bubble chamber pictures. *2nd International Conference on High-Energy Accelerators*, 1959.
- [4] J. Y. BOUGUET. A Release of a Camera Calibration Toolbox for Matlab. web : http://www.vision.caltech.edu/bouguetj/calib_doc/ ,2008.
- [5] C. YU, Q. PENG. Robust recognition of checkerboard pattern for camera calibration. *Optical Engineering*, volume 45, 2006.
- [6] Zhongge WANG,Z. WANG, Y. WU. Recognition and location of the internal corners of planar checkerboard calibration pattern image. *Proceedings of the 29th Chinese Control Conference*, 2010.
- [7] M. RUFLI, D. SCARAMUZZA, R. SIEGWART. Automatic detection of checkerboards on blurred and distorted images. *International Conference on Intelligent Robots and Systems*.2008.
- [8] A. DE LA ESCALERA, J.-M. ARMINGOL. Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration *Sensors 10*, 2010.
- [9] N. OTSU. A Threshold Selection Method from Gray-Level Histograms. *Transactions on Systems, Man, and Cybernetics* , volume 9, pages 62-66 1979.
- [10] H. FREEMAN, E. SOBEL. An isotropic 3*3 gradient operator in Machine Vision for Three-Dimensional Scenes. *Machine Vision for Three-Dimensional Scenes*, pages 376–379, 1990.
- [11] J. CANNY. A Computational Approach to Edge Detection. *Transactions on Pattern Analysis and Machine Intelligence*, pages 679-698, 1986.
- [12] J. COUSTY, G. BERTRAND, L. NAJMAN, M. COUPRIE. Watershed Cuts : Minimum Spanning Forests and the Drop of Water Principle. *Transactions on Pattern Analysis and Machine Intelligence*, volume 31, pages 1362 - 1374, 2009.
- [13] I. DEBLED-RENNESON. Éléments de Géométrie Discrète Vers une Étude des Structures Discrètes Bruitées. Habilitation à diriger des recherches, pages 61-69, 2007.
- [14] J. SHI, C. TOMASI. Good features to Track. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1994.
- [15] D. SHUSTROV. Species identification of wooden material using convolutional neural network. Thèse d'état, 2018.
- [16] Y. LIU, S. LIU, Y. CAO, Z. WANG. Automatic chessboard corner detection method. *IET Image Processing*, volume 10, pages 16-23, 2015.

- [17] J.-P. REVELLES, J. FRANCON. Structure des droites discrètes. *Journées mathématique et informatiques*,Marseille-Luminy, 1989.
- [18] S. SUZUKI. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, volume 30, pages 32-46, 1985.

A Architecture du réseau permettant de segmenter la grume



B Code python pour le calcul des TP,TN,FP,FN

```
%%  
%detection : numpy array  
%truth : numpy array  
%%  
def confusion_matrix(detections , truth):  
    TP = 0  
    FN = 0  
    FP = 0  
    TN = 0  
  
    positive_count = np.count_nonzero(detections)  
    negative_count = truth.size - positive_count  
  
    temp1=np.bitwise_and(truth , detections)  
    TP = np.count_nonzero(temp1)  
    FP = positive_count - TP  
    temp2=np.bitwise_not(detections)  
    temp2=np.bitwise_and(truth , temp2)  
    FN = np.count_nonzero(temp2)  
    TN = negative_count - FN  
    return TP,FP,FN,TN
```

C Exemple de segmentation de grume et de mire

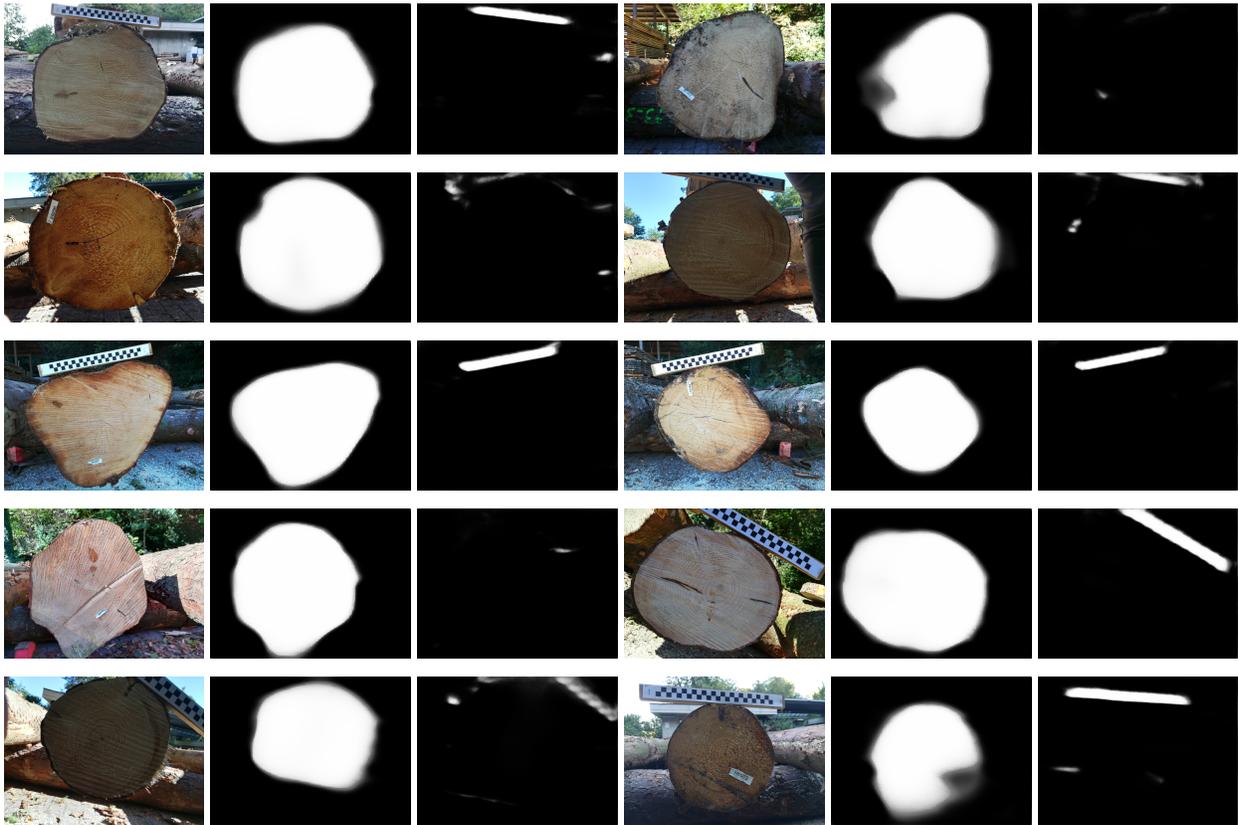


FIGURE 24

Première et quatrième colonne : image en entrée du CNN. Deuxième et cinquième colonne : segmentation de la grume. Troisième et sixième colonne : segmentation de la mire