

Rapport de stage de fin d'étude à l'École centrale des arts et manufactures

Maxandre Jacqueline,
sous la supervision de Stéphanie Pitre-Champagnat,
docteur en physique nucléaire,
chargée de recherche en imagerie médicale
au Centre national de la recherche scientifique (CNRS)

Septembre 2016

Résumé

Un des nouveaux enjeux de l'imagerie fonctionnelle en cancérologie est d'accéder à la description et l'évaluation de la perfusion tumorale et de son hétérogénéité. L'échographie de contraste 4D permet d'accéder à cette hétérogénéité mais demeure une technologie d'apparition récente et encore peu utilisée, et les logiciels permettant l'analyse des images produites font défaut. J'ai développé un logiciel permettant d'analyser les données 4D ; de caractériser les performances de l'échographe Aplio 500 (Toshiba Medical Systems) ; de déterminer une méthode de réduction de la taille des données 4D sauvegardées tout en optimisant l'information ; de calculer des indicateurs de l'hétérogénéité du signal dans une région d'intérêt. Une attention particulière a été portée à la facilité d'utilisation du logiciel (interface graphique intuitive) ; à la facilité de compréhension de son fonctionnement (code source compréhensible) ; de modification de son code source (programmation orientée objet). Dans ce rapport, je propose le calcul de la résolution spatiale de l'Aplio 500 suivant l'axe x au moyen du logiciel que j'ai développé.

Table des matières

I Remerciements	3
II Contexte du stage	3
1 Imagerie médicale fonctionnelle en cancérologie	4
2 Échographie de contraste	4

3	Un enjeu de l'échographie de contraste : création d'un nouveau biomarqueur du cancer	5
4	Échographie de contraste quadridimensionnelle	6
4.1	Définition et historique	6
4.2	Intérêt des quatre dimensions	6
4.3	Défis à relever	6
4.3.1	Résolution spatiale	6
4.3.2	Courbes d'intensité avec choix de l'axe des abscisses	8
4.3.3	Taille des données	8
4.3.4	Hétérogénéité de la vascularisation tumorale	8
III	Objectifs du stage	8
IV	Matériel et méthode	10
5	Dispositif expérimental	10
5.1	Présentation	10
5.2	Intérêt de l'imagerie sur fantôme expérimental	11
6	Logiciel	11
6.1	La réponse à un besoin	11
6.2	Un logiciel	11
6.2.1	Accessible	11
6.2.2	Fonctionnel	13
6.2.3	Utilisable	18
6.2.4	Robuste	19
6.3	Un code source	19
6.3.1	Accessible	19
6.3.2	Structuré	19
6.3.3	Compréhensible	26
6.3.4	Évolutif	26
6.3.5	Robuste	31
V	Résultats	31
7	Introduction	32
8	Expérience	32
9	Analyse des images	32

Table des figures

1	Illustration de la notion de résolution spatiale sur une dimension	7
2	Définition des axes	9
3	Fantômes expérimentaux utilisés	10
4	Photo du dispositif expérimental d'imagerie ultrasonore microfluidique	12
5	Définition des orientations de plans	15
6	Illustration de l'exemple de programmation orientée objet donné au paragraphe 6.3.2 page 19	25
7	Schéma de fonctionnement du programme, 1/3	27
7	Schéma de fonctionnement du programme, 2/3	28
7	Schéma de fonctionnement du programme, 3/3	29
8	Région d'intérêt sélectionnée sur l'image échographique	33
9	Courbe d'intensité de la région d'intérêt de la figure 8 page 33	34
10	Interface graphique du programme qui présente les résultats de l'analyse du graphique	35

Première partie Remerciements

- À Stéphanie Pitre-Champagnat, chargée de recherche CNRS, pour m'avoir accepté dans son équipe, et pour son guidage, sa bienveillance, sa bonne humeur et ses retours sur mon programme,
- À Virginie Grand-Perret, doctorante en imagerie médicale, pour son soutien, sa sympathie, ses retours sur mon programme et la création du dispositif expérimental,
- À Léa Chateauneuf, stagiaire en master 1, pour sa bonne humeur,
- À Bénédicte Coiffier, ingénierie biomédicale, pour son aide sur l'utilisation de l'Aplio 500 et sa sincérité,
- À Nathalie Lassau, chef de l'équipe Imagerie multimodale en cancérologie à Gustave Roussy, pour m'avoir recruté, pour son énergie et ses retours sur mon programme,
- À Baya Benatsou, technicienne d'analyse d'images, pour sa bienveillance,
- À Jean Denoyel, stagiaire en master 2, pour son partage de l'actualité vido ludique,
- À Hongchen Wang, postdoctorant en Imagerie Médicale, pour son style vestimentaire.

Deuxième partie

Contexte du stage

1 Imagerie médicale fonctionnelle en cancérologie

L'imagerie médicale permet d'évaluer l'état de santé d'un patient au moyen d'examens faiblement invasifs.¹ Son utilisation en cancérologie permet notamment de décrire les caractéristiques des tumeurs, qui sont le résultat d'un « processus pathologique où la prolifération exagérée des cellules aboutit à une sur-production tissulaire qui persiste et a tendance à s'accroître » [4].

L'imagerie *fonctionnelle* utilisée en cancérologie permet de d'étudier la fonction des tumeurs [5]. La fonction est définie comme « l'ensemble des actes accomplis par une structure organique définie en vue d'un résultat déterminé » [2] et diffère du métabolisme qui se déroule à de plus petites échelles de grandeur. Son imagerie est permise par l'utilisation d'agents de contraste qui permettent d'augmenter l'écart entre la valeur d'intensité des pixels décrivant la fonction que l'on souhaite observer, autrement dit notre *signal*, et les autres pixels décrivant le reste de la tumeur et son tissu environnant, autrement dit notre *bruit*. Dans le cadre de l'imagerie fonctionnelle de la vascularisation, l'utilisation d'agents de contraste permet de réhausser le signal vasculaire.

2 Échographie de contraste

En échographie, des solutions de microbulles sont utilisées comme agents de contraste. En France, la seule solution ayant reçu l'autorisation de mise sur le marché a le nom commercial SonoVue et est produite par la société italienne Bracco. Les microbulles ont été conçues dans l'idée de réhausser le signal ultrasonore dans le compartiment intravasculaire.

L'onde ultrasonore, engendrée par la sonde de l'échographe, est une onde de pression mécanique qui « va créer des surpressions et des dépressions par rapport à la pression d'équilibre du milieu » dans lequel elle évolue [8]. On mesure les variations maximales de pression locale lors du passage des ultrasons dans les tissus par un indice mécanique [6]. Pour des valeurs d'indice mécanique comprises entre 0,1 et 0,7, les microbulles entrent en résonance avec l'onde ultrasonore et réémettent des ondes aux fréquences variées [8].

L'échographie qui utilise des agents de contraste, appelée échographie de contraste, se sert le plus communément de l'une des ondes réémises, ou échos, pour augmenter le contraste de l'image obtenue. Si l'on considère l'ensemble

1. Un examen faiblement invasif est défini par une effraction de la peau limitée aux besoins de prise de sang et de l'injection de produits. J'ai traduit l'anglais « *minimally invasive* » à mon sens plus juste que le terme français « non-invasif » qui désigne des examens durant lesquels il y a pourtant effraction de la peau [1].

des échos des microbulles comme le signal qu'elles renvoient, alors on appelle *deuxième harmonique du signal* sa composante ayant une fréquence double de la fréquence fondamentale du signal, qui correspond à la fréquence d'émission par la sonde ultrasonore [8].

C'est le deuxième harmonique du signal qui est détecté par l'échographe Aprio 500, utilisé par l'équipe dont je fais partie, quand il fonctionne en mode d'échographie de contraste harmonique (ou *CHI*, pour *Contrast-enhanced Harmonic Imagery*). L'échographe Aprio 500 est un produit de l'entreprise japonaise Toshiba Medical Systems.

3 Un enjeu de l'échographie de contraste : création d'un nouveau biomarqueur du cancer

L'équipe à laquelle j'appartiens travaille à établir une nouvelle caractéristique de l'état d'une tumeur, autrement dit un nouveau *biomarqueur* des tumeurs, grâce aux possibilités offertes par l'échographie de contraste, qui permet de distinguer les microbulles du tissu environnant. Quand la longueur d'une tumeur cancéreuse dépasse environ un millimètre[8], elle fait croître des vaisseaux, appelés néo-vaisseaux, pour l'alimenter en nutriments et en oxygène et évacuer ses déchets. Ce processus est appelé *néo-angiogénèse* et est caractéristique des tumeurs cancéreuses [14, 10, 13, 7]. Les microbulles ont un diamètre ($2,5 \mu\text{m}$ dans le SonoVue) suffisamment petit pour rentrer dans les néo-vaisseaux, dont le diamètre est entre 10 et $20 \mu\text{m}$, mais suffisamment grand pour ne pas diffuser en dehors des vaisseaux vers les tissus environnants.

Les images d'échographie de contraste distinguent ainsi clairement les vaisseaux dans lesquels sont les microbulles, du reste du tissu. En particulier, elles permettent de distinguer les néo-vaisseaux alimentant la tumeur du reste du tissu tumoral. Elles promettent de pouvoir décrire la néo-vascularisation tumorale avec des paramètres quantitatifs. Mon équipe souhaite faire de son degré de désorganisation, appelée *hétérogénéité*, un nouveau biomarqueur d'une tumeur. Les tumeurs cancéreuses sont connues pour le développement anarchique des néo-vaisseaux, caractérisés par des vaisseaux qui fuient, sont dilatés et sont connectés entre eux de manière désorganisée [14, 15].

Le développement récent de l'immunothérapie comme traitement anti-cancer appelle au développement urgent de nouveaux biomarqueurs du cancer[20]. En effet, les médecins sont parfois dans l'incapacité de déterminer s'il faut poursuivre ou non le traitement immunothérapeutique, en l'absence d'indicateurs fiables du sens de l'évolution du cancer du patient.

4 Échographie de contraste quadridimensionnelle

4.1 Définition et historique

L'échographie quadridimensionnelle, ou « 4D », qui rajoute aux deux dimensions de l'espace et à la dimension du temps de l'échographie tridimensionnelle, ou « 2D+Temps », une troisième dimension spatiale, est apparue sur les premiers échographes à la fin des années 1990 [18]. L'échographie de contraste a commencé à être utilisée en *clinique*, c'est-à-dire sur les patients, à partir de 2004 [19]. L'échographie de contraste quadridimensionnelle est déjà possible sur l'échographe Aplio 500. C'est l'échographe que l'équipe à laquelle j'appartiens utilise. La technique n'est cependant pas encore utilisé en clinique à l'institut Gustave Roussy, où j'effectue mon stage.

4.2 Intérêt des quatre dimensions

Par rapport à l'échographie « 2D+Temps », l'échographie 4D permet d'une part des analyses plus riches des tumeurs, dans leur volume et non suivant un seul plan de coupe, et d'autre part de réduire l'influence du positionnement de la sonde par l'opérateur sur les résultats obtenus après analyse des images, source de variabilité dans les résultats appelée *variabilité inter-opérateur*.

4.3 Défis à relever

La relative jeunesse de l'échographie de contraste quadridimensionnelle pose des défis à relever avant son utilisation en clinique.

4.3.1 Résolution spatiale

Sa résolution spatiale est par exemple inconnue. La résolution spatiale est la distance minimale, en unité de longueur, à laquelle on arrive à distinguer, sur une image produite par un système d'imagerie, deux objets ponctuels imaginés[3]. Si le système d'imagerie n'est pas parfaitement précis, les deux objets ponctuels n'apparaîtront pas comme deux points, mais seront plus étalés. Ils apparaîtront comme le résultat de la convolution — du filtre — de la « vraie » image des objets ponctuels par la réponse impulsionale optique du système d'imagerie, ou fonction d'étalement du point, qui caractérise l'imprécision du système d'imagerie (par exemple la réponse impulsionale optique peut être une gaussienne bidimensionnelle).

La résolution spatiale des images ultrasonores est anisotrope [9], autrement dit elle n'est pas la même dans toutes les directions. On doit caractériser la résolution spatiale de notre échographe selon chacun de ses axes pour la déterminer entièrement. Le problème devient alors de déterminer la résolution spatiale d'autant de détecteurs produisant chacun un signal unidimensionnel que l'échographe a d'axes, soit quatre. Pour déterminer la résolution spatiale suivant un

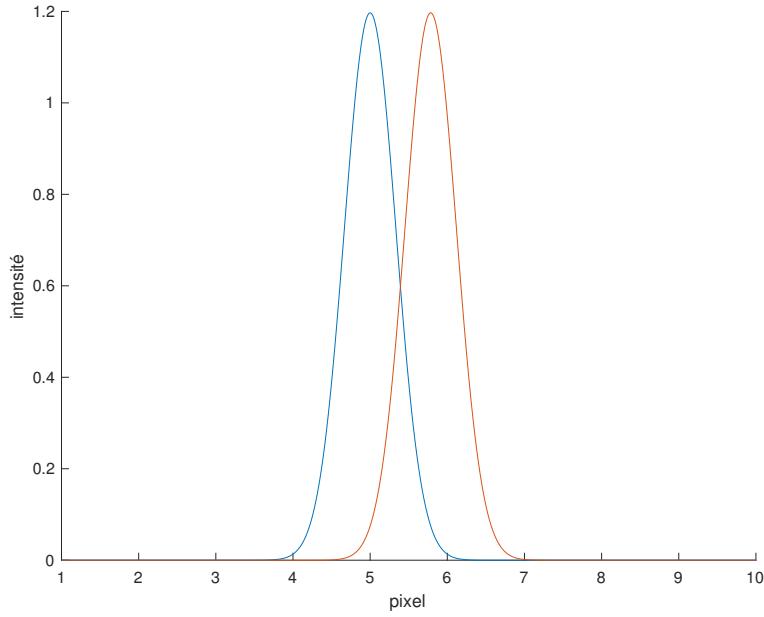


FIGURE 1 – Illustration de la notion de résolution spatiale sur une dimension. Pour illustrer la notion de résolution spatiale, on imagine que l'on a deux sources ponctuelles dont on a fait une image. On a pris l'évolution de l'intensité de l'image sur 10 pixels sur un certain axe et on en a fait le graphique ci-dessus. Les pics d'intensité correspondant aux deux sources ponctuelles sont situées à une distance égale à la largeur à mi-hauteur de leurs courbes d'intensité respectives. C'est cette distance, égale à la largeur à mi-hauteur de l'une des deux courbes, que l'on définit comme la résolution spatiale. Cette distance doit être donnée en mètre. Il est nécessaire de convertir en mètre la largeur à mi-hauteur trouvée en pixel pour déterminer la résolution spatiale.

axe fixé, on la définit comme la largeur à mi-hauteur de l'un de deux pics d'intensité identiques extraits de deux courbes d'intensités identiques produites par l'imagerie de deux sources ponctuelles identiques. On considère que la largeur à mi-hauteur de l'un des deux pics est la distance minimale à laquelle on arrive à les considérer comme provenant de deux sources ponctuelles distinctes (voir figure 1 pour l'illustration de ce choix). Cette distance doit être définie en unité de longueur (en mètre dans le système international) pour être une résolution spatiale. Il est donc nécessaire de convertir en mètre la largeur à mi-hauteur trouvée en pixel pour déterminer la résolution spatiale.

4.3.2 Courbes d'intensité avec choix de l'axe des abscisses

Le logiciel d'analyse des données quadridimensionnelles ultrasonores UltraExtendFX fourni par Toshiba Medical Systems ne permet pas de créer de graphique de l'intensité des pixels en fonction de l'axe des x , y ou z , mais seulement en fonction de l'axe du temps (pour la définition des axes, voir la figure 2 page suivante). Ce logiciel ne permet donc pas d'établir la résolution spatiale de l'échographe suivant x , y ou z en suivant la méthode décrite en 4.3.1 page 6.

4.3.3 Taille des données

Les données d'images quadridimensionnelles sont volumineuses. Prenons un exemple. On imagine qu'on choisit d'enregistrer l'entièreté d'une image quadridimensionnelle, autrement dit on a sélectionné une région d'intérêt qui englobe toute l'image. On considère qu'on fait un enregistrement sur 45 pas de temps. Les données que l'on obtient après conversion (pour les rendre lisibles) font 720 mégaoctets. Et pourtant, elles sont apparemment compressées sur 8 bits, selon une méthode que nous n'avons pas pu déterminer, peut-être au moyen d'une fonction logarithmique.²

4.3.4 Hétérogénéité de la vascularisation tumorale

L'équipe à laquelle j'appartiens cherche à caractériser l'hétérogénéité de la vascularisation tumorale pour en faire un biomarqueur du cancer. On définit une région d'intérêt comme la partie de l'image que l'on souhaite analyser. Pour caractériser l'hétérogénéité de la vascularisation tumorale, il est nécessaire d'arriver à trouver un indicateur permettant de la mesurer.

Il n'y a pas de consensus sur l'indicateur à utiliser, et des études sont menées au sein de mon équipe en parallèle de mon stage pour déterminer les indicateurs les plus pertinents. Certains sont néanmoins plus courants que d'autres, comme l'entropie de l'image (dont la définition correspond au sens strict à l'hétérogénéité), le coefficient de variation, ou encore divers indicateurs tirés de la matrice de co-occurrence des niveaux de gris de la région d'intérêt sélectionnée (voir le paragraphe 6.2.2 page 14 pour le détail des indicateurs inclus dans le logiciel développé).

Troisième partie

Objectifs du stage

L'objectif de mon stage est de développer un outil informatique permettant de :

1. analyser les données ultrasonores 4D ;

2. Les données dont nous avons mesuré la taille sont encodées en VoxelData, encodage dont Toshiba Medical Systems a la propriété.



FIGURE 2 – Définition des axes.

Les axes que l'on a défini par rapport à la sonde quadridimensionnelle ultraso-nore PLT-1204MV 12MHz.



FIGURE 3 – Fantômes expérimentaux utilisés.

De gauche à droite, les matériaux utilisés sont : Ecoflex Gel, Ecoflex 00-20, Ecoflex 00-30, Vytalox 10, Moldstar 15 slow (Smooth-On).

2. caractériser les performances de l'échographe Aplio 500 (Toshiba Medical Systems), dont sa résolution spatiale ;
3. déterminer une méthode de réduction de la taille des données 4D sauvegardées tout en optimisant l'information.

Quatrième partie

Matériel et méthode

5 Dispositif expérimental

5.1 Présentation

Un dispositif expérimental permettant entre autres de caractériser les performances de l'échographe Aplio 500 a été réalisé par Virginie Grand-Perret qui réalise son doctorat dans l'équipe à laquelle j'appartiens. Le dispositif utilise des techniques issues de l'imagerie ultrasonore et de la physique microfluidique.

Une pompe met en mouvement l'eau contenue dans un tuyau de taille micrométrique (d'environ 250 µm de diamètre) qui traverse un *fantôme expérimental*. Le fantôme est un morceau d'un matériau dont les propriétés acoustiques sont sensées être proches de celles des tissus du corps humain [11]. De nombreux matériaux ont été utilisés comme l'EcoFlex, le VytaFlex, le DragonSkin... (voir la figure 3). La pression au sein du tuyau est contrôlée au moyen d'appareils de la société française d'outillage microfluidique Fluigent.

Dans le tuyau, on injecte la solution de microbulles SonoVue au moyen d'une seringue. On image le fantôme en détectant le second harmonique du signal renvoyé par les microbulles au moyen de la sonde ultrasonore quadridimensionnelle

PLT-1204MV qui émet un signal à une fréquence de 12MHz. Une photo du dispositif expérimental est présentée à la figure 4 page suivante.

5.2 Intérêt de l'imagerie sur fantôme expérimental

Le fantôme expérimental permet de réduire les sources de variabilité de l'image acquise par rapport à de l'imagerie *in vivo* (sur souris par exemple) ou à de l'imagerie *clinique*.

On peut avoir à imager différents patients ou souris au fil du temps, ce qui ajoute aux mesures une variabilité inter-individuelle, et même si on image la même souris ou le même patient, leur constitution évolue au fil du temps, source de variabilité intra-individuelle. L'imagerie sur fantôme permet de s'affranchir de ces sources de variabilité.

Pour caractériser les performances de l'Aprio 500 il faut que les images obtenues en soient une bonne représentation, et toutes les sources de variabilité externes à l'Aprio 500 doivent être éliminées. L'imagerie sur fantôme est donc adaptée à notre objectif.

6 Logiciel

6.1 La réponse à un besoin

Le logiciel d'analyse des images produites par l'Aprio 500 appelé UltraExtendFX, conçu par Toshiba Medical Systems, ne permet pas de réaliser toutes les fonctionnalités souhaitées par mon équipe.

Il ne permet pas d'afficher de graphique selon x , y ou z (voir la sous-section 4.3.2 page 8), ni de calculer d'indicateurs de l'hétérogénéité de la région d'intérêt, ni d'analyser automatiquement un graphique pour trouver ses pics et ses distances pic à pic, ni de sous-échantillonner les fichiers d'images pour réduire leur taille mémoire.

De plus, UltraExtendFX ne nomme pas chacun des plans et des axes, ce qui rend la compréhension du positionnement dans l'espace difficile.

Le logiciel que j'ai développé propose une alternative à UltraExtendFX pour l'analyse des images quadridimensionnelle, et répond à ces manquements.

6.2 Un logiciel

6.2.1 Accessible

Une personne qui voudrait se servir de mon logiciel doit simplement aller sur https://github.com/MaxandreJ/Ultrasound_4D_Images_Analyser et cliquer sur « Clone or download » puis « Download Zip » pour télécharger mon logiciel dans sa dernière version.

Après avoir décompressé l'archive .zip, elle doit lancer la fonction lancer_programme.m dans Matlab (version 2014a ou plus récent) pour commencer à utiliser le programme.

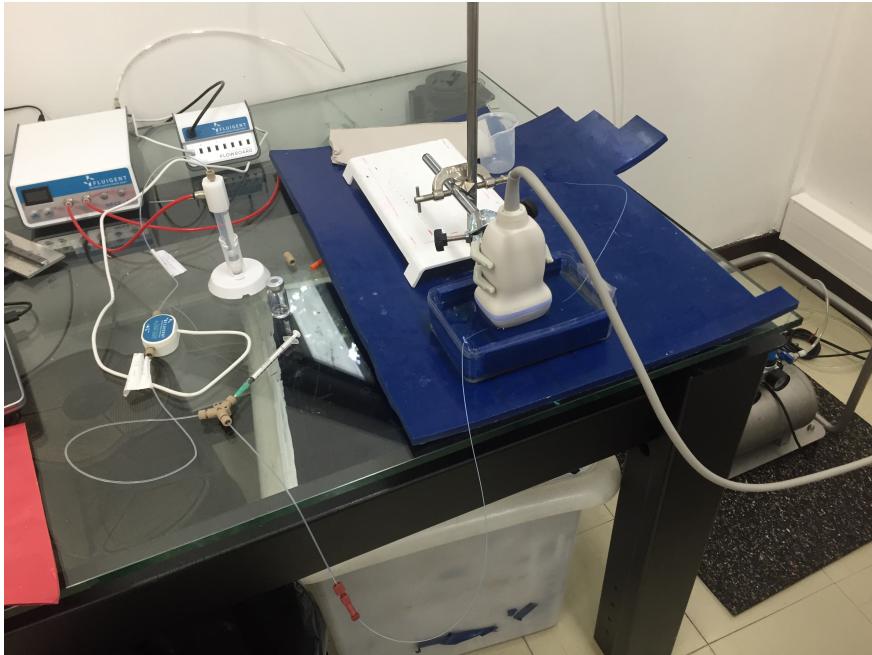


FIGURE 4 – Photo du dispositif expérimental d'imagerie ultrasonore microfluide.

À droite sur la table, on peut voir la sonde échographique 4D PLT-1204MV 12MHz de Toshiba Medical Systems, reliée par un cable (à l'échographe, invisible sur cette photo), immergée dans de l'eau pour une bonne conduction des ondes ultrasonores, qui surplombe le fantôme expérimental (caché), dans lequel arrive et duquel repart un tuyau d'environ 250 µm de diamètre, à l'intérieur duquel circule de l'eau et à certains moments des microbulles.

Sur le sol à droite, on peut voir la pompe qui fournit la pression nécessaire pour déplacer le fluide contenu dans le tuyau. À gauche sur la table, on trouve un flacon de microbulles en solution de la marque SonoVue et une seringue d'injection, par laquelle on injecte les microbulles dans le tube micrométrique. En haut à gauche sur la table, on trouve le dispositif de la marque d'outillage microfluide Fluigent qui permet de précisément contrôler la pression dans le tube au moyen d'un ordinateur, dont on aperçoit le coin à l'extrême gauche de la photo, sur la table.

6.2.2 Fonctionnel

Chargement d'images

Capacités Le logiciel peut charger des fichiers au format DICOM encodés en RawData ou en VoxelData (formats propres à Toshiba Medical Systems), ou au format .mat (format de matrice Matlab).

Les fichiers au format DICOM doivent préalablement avoir été enregistrés depuis l'échographe Aplio 500, puis convertis par le logiciel RawDataExport v1.11 version août 2016 (ou plus récent) avant d'être ouverts. La conversion vers un encodage en VoxelData est le seul qui permet de ne pas avoir de distorsions dans les images converties mais nécessite d'être effectuée sur un ordinateur doté d'une carte graphique particulière (probablement conçue avec une architecture de calcul parallèle, par exemple CUDA pour les cartes graphiques de la marque Nvidia).

Sur mon logiciel, pour charger des images enregistrées au format DICOM, il faut sélectionner le dossier qui contient l'ensemble des fichiers convertis et le fichier texte PatientInfo qui contient l'ensemble des indications de lecture de ces fichiers.

Les fichiers au format .mat peuvent être soit chargés individuellement, soit chargés par dossier contenant un fichier par pas de temps enregistré.

Manquements Il est pour l'instant impossible de charger des images « 2D+Temps » capturées sur l'Apilio 500, à cause de l'impossibilité de convertir de tels fichiers sur le logiciel RawDataExport pour les rendre lisibles par Matlab.

De plus, quel que soit l'encodage choisi, les images semblent compressées en intensité par une compression logarithmique. L'intervalle des valeurs d'intensité semble correspondre aux valeurs d'un entier non-signé codé sur 8 bits, soit de 0 à 256. La compression semble être effectuée par le logiciel RawDataExport puisque le logiciel UltraExtendFX, qui ne requiert pas l'utilisation du logiciel RawDataExport, affiche des valeurs d'intensité sur une gamme plus large (d'ordre de grandeur 10^{-4}). La compression logarithmique des valeurs d'intensité peut être source d'erreur de diagnostic [16] et il est impératif de pouvoir accéder à des données non-compressées.

Des discussions avec Toshiba Medical Systems sont en cours pour résoudre ces problèmes.

Le chargement d'images quadridimensionnelle encodées en VoxelData et contenant plus de 45 pas de temps est très long et ralentit considérablement Matlab. Plusieurs solutions peuvent être trouvées à ce problème mais par manque de temps je n'ai pas pu les implémenter. On pourrait :

1. Ne plus charger l'entièreté des images dans une variable de l'espace de travail, mais seulement charger l'image que l'on souhaite afficher au moment où on souhaite l'afficher. C'est ce concept qui est utilisé dans les bases de données « NoSQL » comme MongoDB qui sont conçues pour gérer de vastes quantités de données. Si l'on fait ce choix, le passage entre les images d'un examen sera néanmoins ralenti.

2. Augmenter l'allocation de mémoire virtuelle sur le système d'exploitation utilisé (c'est le plus simple mais cela ne marchera peut-être pas).
3. Réécrire le programme dans un langage plus robuste, comme C# ou C++, mais je ne suis pas sûr que cela fonctionne. Réécrire le programme en Python ne changera probablement rien à ce problème.

Dans l'immédiat, je recommande simplement de ne pas charger d'examen contenant plus de 45 pas de temps.

Choix d'une image Le choix d'une image peut se faire en entrant ses coordonnées sur les deux axes non-inclus dans les axes de l'image affichée ou en naviguant entre les images par commande au clavier (voir 6.2.3 page 18).

L'orientation du plan de l'image affichée est également choisie par commande au clavier (voir 6.2.3 page 18). On distingue quatre orientations principales de plans décrites à la figure 5 page suivante, et trois orientations supplémentaires, respectivement définies par les axes X et Temps, Y et Temps, et Z et Temps.

Sélection de région d'intérêt Il est possible de sélectionner une région d'intérêt rectangulaire en entrant les coordonnées des coins opposés du rectangle dans l'interface, ou en la sélectionnant sur l'image avec la souris (voir 6.2.3 page 19).

Il est également possible de sélectionner une région d'intérêt polygonale en la sélectionnant sur l'image avec la souris (voir 6.2.3 page 19).

Calcul d'indicateurs de l'hétérogénéité de la région d'intérêt On permet de calculer les indicateurs suivants de l'hétérogénéité de la région d'intérêt :

- l'entropie globale de la région d'intérêt est une mesure de la quantité d'information contenue dans la région d'intérêt (on doit à Claude Shannon cette définition de l'entropie [17]). Elle vaut 0 lorsque la région d'intérêt a une seule intensité de gris et augmente avec le nombre d'intensités de gris présentes dans l'image. Elle donne une information sur la texture de la région d'intérêt : plus l'image est hétérogène, plus l'entropie est forte. Mon programme utilise la définition suivante de l'entropie adaptée à une image en niveau de gris [12] :

$$-\sum_{I \in \text{IntervalleIntensitésRégionIntérêt}} \frac{\text{NombreDePixels}_I}{\text{NombreTotalPixels}} * \log_2\left(\frac{\text{NombreDePixels}_I}{\text{NombreTotalPixels}}\right)$$

où :

IntervalleIntensitésRégionIntérêt correspond à l'intervalle des intensités de la région d'intérêt (si l'image est codée avec des entiers non-signés sur 8 bits, cet intervalle est [0; 256]) ;

NombreDePixels_I correspond au nombre de pixels dans la région d'intérêt dont la valeur d'intensité est *I* (valeur que l'on retrouve dans l'histogramme non-pondéré de la région d'intérêt) ;

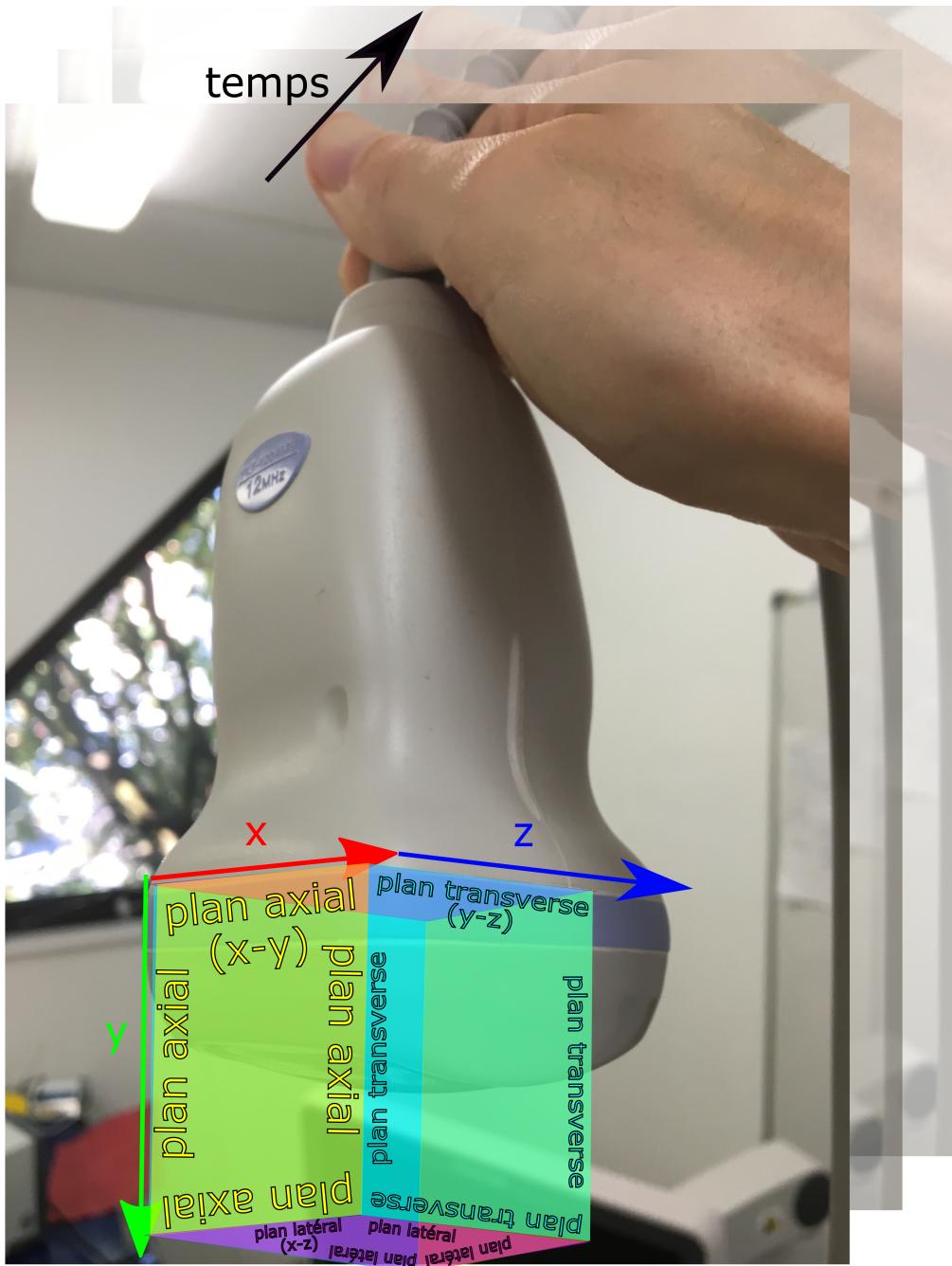


FIGURE 5 – Les orientations de plans que l'on a défini à partir des axes de la sonde.

$NombreTotalPixels$ correspond au nombre de pixels dans la région d'intérêt.

- le coefficient de variation défini par :

$$\frac{EcartTypeIntensitésPixelsRégionIntérêt}{MoyenneIntensitésPixelsRégionIntérêt}$$

où :

$EcartTypeIntensitésPixelsRégionIntérêt$ correspond à l'écart type des valeurs d'intensités des pixels dans la région d'intérêt ;

$MoyenneIntensitésPixelsRégionIntérêt$ correspond à la moyenne des valeurs d'intensités des pixels dans la région d'intérêt.

- quatre statistiques sur la matrice de co-occurrence des niveaux de gris de la région d'intérêt, nommées *corrélation*, *énergie*, *homogénéité* et *contraste* dont on peut trouver la définition au bas de la page de documentation de Matlab accessible par le lien suivant : <http://fr.mathworks.com/help/images/ref/graycoprops.html>. Ces statistiques donnent des informations sur la périodicité des niveaux d'intensité de gris dans la région d'intérêt.

Affichage de graphique On peut afficher un graphique à partir des valeurs dans la région d'intérêt sélectionnée. Si on a choisi une région d'intérêt rectangulaire, on peut choisir l'axe des abscisses du graphique comme l'un des deux axes de l'image affichée. Pour toutes les formes de région d'intérêt, on peut aussi choisir l'axe des abscisses comme l'un des deux axes non présent dans l'image affichée. Si on a choisi une région d'intérêt rectangulaire, on peut moyenner les valeurs de la région d'intérêt selon un des deux axes de l'image affichée, qui ne doit pas être l'axe choisi comme axe des abscisses. Par exemple, si on a choisi la région d'intérêt rectangulaire suivante sur une image orientée axialement (soit une orientation suivant les axes x et y) :

$$\begin{matrix} 15 & 30 & 15 & 30 \\ 10 & 20 & 10 & 20 \\ 5 & 10 & 5 & 10 \end{matrix}$$

et que l'on choisit d'afficher le graphique avec comme axe des abscisses x et comme axe de moyennage y alors le graphique nous permettra de visualiser la fonction suivante :

$$xDébutRégionIntérêt \rightarrow \frac{5 + 10 + 15}{3} = 10$$

$$xDébutRégionIntérêt + 1 \rightarrow 20$$

$$xDébutRégionIntérêt + 2 \rightarrow 10$$

$$xDébutRégionIntérêt + 3 \rightarrow 20$$

avec $xDébutRégionIntérêt$ la coordonnée en x du premier pixel (par rapport à x) de la région d'intérêt.

Si on a choisi une région d'intérêt rectangulaire, on peut également choisir de ne pas faire de moyennage. On affichera alors toutes les courbes que l'on peut engendrer dans la région d'intérêt avec un axe des abscisses fixé. En reprenant le même exemple, en choisissant l'axe x comme abscisse, on affichera alors trois courbes d'ordonnées suivantes : $[5; 10; 5; 10]$, $[10; 20; 10; 20]$ et $[15; 30; 15; 30]$, et d'abscisses identiques : $[xDébut; xDébut + 1; xDébut + 2; xDébut + 3]$ avec $xDébut$ qui correspond au $xDébutRégionIntérêt$ précédemment défini.

Quelle que soit la forme de la région d'intérêt, on peut aussi moyenner les valeurs d'intensité sur les deux axes de l'image affichée. Cela est nécessaire quand on a choisi un axe des abscisses (par exemple le temps) qui n'est pas l'un des deux axes de l'image affichée (par exemple x et y). Par exemple, si la région d'intérêt vaut les valeurs précédemment définies au temps 1, et les valeurs suivantes au temps 2 (toujours sur une image orientée axialement) :

$$\begin{matrix} 30 & 60 & 30 & 60 \\ 20 & 40 & 20 & 40 \\ 10 & 20 & 10 & 20 \end{matrix}$$

et qu'on veut afficher le graphique avec comme axe des abscisses le temps et comme axe de moyennage x et y , alors le graphique nous permettra de visualiser la fonction suivante :

$$1 \rightarrow \frac{5 + 10 + 15 + 10 + 20 + 30 + 5 + 10 + 15 + 10 + 20 + 30}{12} = 15$$

$$2 \rightarrow 30$$

Si la région d'intérêt est de forme polygonale, la matrice de la région d'intérêt sera toujours de forme rectangulaire, mais les valeurs qui ne sont pas incluses dans le polygone vaudront la valeur nulle (*NaN* sur Matlab).

Analyse du graphique Le logiciel permet une détection automatique des pics, autrement dit des maximums locaux, du graphique. L'utilisateur choisit le nombre de pics que le logiciel doit détecter, et peut aussi décider de lisser le graphique par la méthode des moyennes mobiles avant de les détecter.

La méthode des moyennes mobiles consiste à lisser la courbe originale en remplaçant chaque point de la courbe par la moyenne des ordonnées de la courbe dans un certain voisinage du point. Par exemple, si on définit la taille de voisinage à trois, l'ordonnée d'un point de la courbe lissée sera égale à la moyenne de l'ordonnée du point à gauche de ce point, du point lui-même, et du point à droite de ce point (soit trois points).

Sous-échantillonnage des fichiers Le logiciel permet de sous-échantillonner les fichiers chargés, et de les enregistrer dans leur version sous-échantillonnée.

Les courbes classiques obtenues en imagerie ultrasonore de contraste sont appelées *courbes de réhaussement* en fonction du temps. Une région d'intérêt est définie dans une tumeur où les microbulles arriveront une fois qu'elles seront injectées, puis qu'elles quitteront progressivement. Le graphique engendré par

le choix du temps comme axe des abscisses et le moyennage sur les deux axes de l'image dans laquelle on a choisi la région d'intérêt produira une courbe avec un fort pic de signal au moment de l'arrivée des microbulles, puis une décroissance pendant qu'elles quittent la région.

On permet à l'utilisateur de choisir un certain pas de temps en fonction du temps où le pic de signal est atteint (par exemple 1,5x le temps où le pic est atteint) à partir duquel on sous-échantillonera les volumes capturés par l'échographe (on en enregistrera par exemple plus qu'un sur dix). Ce procédé permet de réduire la taille des données quadridimensionnelles, dont la grandeur pose problème (voir 4.3.3 page 8).

Enregistrement des images, du graphique ou de l'interface On laisse la possibilité à l'utilisateur d'enregistrer l'image échographique affichée, le graphique affiché, ou l'interface graphique entière dans différents format d'image, dont des formats d'image matricielles (.png, .bmp, .jpg) et un format d'image vectoriel (.eps). Nous avons pu offrir cette possibilité à l'utilisateur en utilisant la bibliothèque *export_fig* développé par almany (<https://fr.mathworks.com/matlabcentral/fileexchange/23629-export-fig>).

6.2.3 Utilisable

Interface graphique On a développé une interface graphique pour le programme grâce à l'interface graphique de création d'interface graphique appelée GUIDE sous Matlab.

Textes d'aide Chaque fonctionnalité du programme est expliquée par des bulles d'aide qui se trouvent près de l'endroit sur l'interface graphique où la fonctionnalité va être utilisée. La bulle d'aide s'affiche dès que la souris survole une petite icône en forme de point d'interrogation blanc entouré d'un carré bleu layette.

Une aide générale est également accessible en cliquant sur le menu « Aide » du programme.

Commandes au clavier Les flèches multidirectionnelles permettent de glisser entre les plans de l'image, ce qui permet de rapidement passer d'un plan à l'autre. Les flèches gauche et droite permettent de glisser selon le premier axe mentionné dans le titre de l'image, tandis que les flèches bas et haut permettent de glisser selon le deuxième axe mentionné.

Les chiffres du clavier ou du pavé numérique permettent d'alterner entre les différents plans de l'image. On a les associations suivantes (voir figure 5 page 15 pour la définition des plans dans l'espace) :

- 0 : plan axial (Y en ordonnées et X en abscisses) ;
- 1 : plan latéral (Z en ordonnées et X en abscisses) ;
- 2 : plan transverse (Z en ordonnées et Y en abscisses) ;

- 3** : plan x-temps (X en ordonnées et le temps en abscisses) ;
- 4** : plan y-temps (Y en ordonnées et le temps en abscisses) ;
- 5** : plan z-temps (Z en ordonnées et le temps en abscisses).

Commandes à la souris Des menus contextuels peuvent apparaître en effectuant un clic droit sur l'image (pour l'enregistrer ou sélectionner une région d'intérêt) ou en effectuant un clic droit sur le graphique (pour l'enregistrer).

Une région d'intérêt rectangulaire est sélectionnable par un cliquer-glisser. Une région d'intérêt polygonale est sélectionnable par autant de clics que de points du polygone, le dernier clic devant être double et sur un point déjà défini.

6.2.4 Robuste

Des messages d'erreur sont affichés sous la forme de boîtes d'avertissement en cas d'action illégale. Un message décrit ce que l'utilisateur doit faire pour éviter l'erreur.

6.3 Un code source

6.3.1 Accessible

Une personne qui voudrait poursuivre le développement du logiciel doit :

1. créer un compte sur Github <https://github.com/> ;
2. m'envoyer un courrier électronique sur l'adresse maxandre.jacqueline@gmail.com pour que j'ajoute son compte à la liste des collaborateurs de mon dépôt Git ;
3. télécharger GitHub Desktop sur le lien suivant : <https://desktop.github.com/> ;
4. aller sur https://github.com/MaxandreJ/Ultrasound_4D/Images_Analyser et cliquer sur « Clone or download » puis « Open in Desktop » ;
5. choisir le dossier où elle veut enregistrer le code source de mon logiciel.

Un nouveau répertoire sera créé sur son ordinateur, il contiendra toutes les données de mon programme. Une fois que des modifications seront faites, il faudra écrire un message décrivant ces modifications puis cliquer sur « Commit » dans GitHub Desktop, puis cliquer sur « Sync ».

6.3.2 Structuré

En objets

Définition des concepts de la programmation orientée objet Mon code est constitué d'objets logiciels en interaction, à la manière du monde réel où des objets physiques sont en interaction. Voici la liste des concepts de programmation orientée objet nécessaire à la compréhension de mon programme :

- un *objet* est un élément du logiciel (par exemple la voiture de Virginie) ;
- une *classe* est un type d'objet (par exemple les voitures) ;
- une *propriété* d'une classe ou d'un objet est une caractéristique qui permet de le définir (par exemple, la couleur) — cette caractéristique peut-être sous la forme d'un autre objet ;
- une *méthode* est une action que peut faire un objet (par exemple, rouler) ;
- une *instance* d'une classe est un objet appartenant à une classe (par exemple, la voiture de Virginie appartient à la classe des voitures), souvent instance et objet sont synonymes.
- l'*instanciation* correspond à la définition de l'appartenance d'un objet à une classe ;
- un *constructeur* est la méthode d'une classe dont le résultat retourné est une instance de cette classe ;
- une *classe concrète* est une classe qui peut avoir des instances (par exemple, les voitures) ;
- une *classe abstraite* est une classe qui ne peut pas avoir d'instance (par exemple, les véhicules) ;
- une *implémentation* correspond à du code qui effectue une certaine fonction ;
- une *méthode abstraite* est une méthode non-implémentée (dont le code n'est pas écrit) d'une classe abstraite qui doit être redéfinie dans une classe concrète par une méthode concrète de même nom pour être utilisable ;
- une *méthode concrète* est une méthode implémentée (dont le code est écrit) ;
- une *méthode statique* est une méthode qui est liée à une classe elle-même et non à ses instances ;
- l'*héritage* est l'appartenance d'une classe concrète à une classe abstraite (par exemple les voitures héritent des véhicules) — si une classe concrète hérite d'une classe abstraite, elle inclut dans sa définition toutes les propriétés et méthodes de la classe abstraite de laquelle elle hérite (par exemple, si les véhicules ont la méthode « rouler », les voitures aussi) ;
- le *polymorphisme* est la capacité d'une méthode abstraite à être redéfinie en fonction de la classe concrète à laquelle appartient l'instance depuis laquelle elle est appelée.

Exemples Prenons un exemple : *une personne rentre dans une voiture rouge qui fait 4 mètres de long*.

En utilisant le lexique de la programmation orientée objet, on dira que l'objet personne, qui est une instance de la classe Personne, utilise sa méthode « rentrer » en prenant pour argument une voiture, qui est une instance de la classe Voiture. Les objets de la classe Voiture ont tous une propriété « couleur », et cette voiture, autrement dit cette instance, a cette propriété qui vaut 'rouge'. La classe Voiture est une classe concrète, c'est-à-dire qu'elle peut avoir des instances, qui hérite d'une classe abstraite, c'est-à-dire qui ne peut pas avoir

d'instance, Véhicule. La classe Véhicule a une propriété longueur_en_mètre que par conséquent toutes les instances des classes concrètes qui héritent de la classe abstraite Véhicule ont également. En particulier, notre voiture a une propriété longueur_en_mètre qui vaut 4.

On remarque que quand on dit « une personne » (ou « une voiture »), on définit implicitement l'appartenance de l'objet personne à la classe d'objet, autrement dit le type d'objet, Personne. Cette relation d'appartenance, implicite dans la langue, permet à la personne d'être décrite par les caractéristiques (i.e. propriétés) et la rend capable d'effectuer les actions (i.e. méthodes) que tous les membres (i.e. objets ou instances) de la catégorie (i.e. classe) des personnes, qu'on a appelé Personne. Mathématiquement, on crée cette relation d'appartenance de la manière suivante : soit personne appartenant à Personne, ou soit $\text{personne} \in \text{Personne}$. Cette opération de création d'un objet ou instance appartenant à une classe s'appelle en informatique une *instanciation*. On instancie une instance d'une classe en utilisant le constructeur de la classe, qui a généralement le même nom que celle-ci. L'instanciation de l'objet personne comme instance de la classe Personne s'écrit sur Matlab au moyen du constructeur de personne appelé Personne. Cela donne le code suivant :

```
personne = Personne;
```

Et la phrase entière s'écrit :

```
personne = Personne;
voiture = Voiture('rouge', 4);
personne.rentrer(voiture);
```

en ayant défini les classes suivantes dans le chemin courant de Matlab :

— la classe Personne :

```
classdef Personne
methods
    function rentrer(voiture)
        %code ici
    end
end
```

— la classe Véhicule :

```
classdef (Abstract) Vehicule
properties
    longueur_en_metre
end
```

— la classe Voiture :

```
classdef Voiture < Vehicule
```

```

properties
    couleur
end

methods
    function Voiture(couleur, longueur)
        % le constructeur
        soi.couleur = couleur;
        soi.longueur = longueur;
    end
end

```

Prenons un second exemple pour introduire une autre notion. Imaginons qu'on veuille programmer la phrase : *on change la couleur de la voiture en bleu.*

Dans cette phrase, on remarque que l'agent qui change la couleur de la voiture n'est pas identifié. Ce qui compte, c'est que les objets « voiture » peuvent voir leur couleur changer. Dans cette phrase, l'action de changement de voiture est davantage liée au type d'objet concerné (Voiture) qu'à l'agent qui les effectue. On modélise cette possibilité de changement de couleur de voiture par la création d'une *méthode statique* au sein de la classe Voiture, que l'on va appeler quand on voudra changer la couleur d'une instance « voiture ». Le changement de couleur de la voiture s'effectue sur Matlab en exécutant le code suivant :

```
Voiture.changer_couleur(voiture, 'bleu');
```

après avoir ajouté à la classe Voiture la méthode statique changer_couleur de cette manière :

```

classdef Voiture
    %code non-recopié ici
    methods (static)
        function changer_couleur(voiture, 'bleu')
            voiture.couleur = 'bleu';
        end
    end
end

```

Prenons un troisième exemple pour introduire une autre notion. Programmons la phrase : *la porte de la voiture est jaune.* Pour cela, il faut ajouter aux propriétés de la voiture la propriété porte, qu'on va elle même définir comme une instance d'une classe Porte qui a une propriété couleur. On programme la phrase en instanciant une porte dans les propriétés de la voiture, et en mettant la bonne couleur voulue à la porte. On doit exécuter le code suivant :

```
voiture.porte = Porte;
voiture.porte.couleur = 'jaune';
```

après avoir ajouté la classe voiture la propriété porte :

```

classdef Voiture
    properties
        porte
        %code non-recopié ici
    end

    % code non-recopié ici
end

```

et après avoir défini une classe Porte ainsi :

```

classdef Porte
    properties
        couleur
    end
end

```

Enfin, prenons un quatrième exemple pour présenter nos dernières notions. Programmons la phrase : *notre voiture et un camion roulent*. On observe que notre voiture et le camion sont tous deux des véhicules et font une action qui est désigné par le même concept « rouler ». Cependant, la manière dont la voiture et le camion roulent est différente (la voiture a probablement besoin d'un moteur moins puissant que le camion, et des pneus moins résistants...).

On modélise cette situation dans nos classes en introduisant les concepts de *méthode abstraite* et de *polymorphisme*. On va redéfinir la classe Véhicule en lui ajoutant une méthode abstraite « rouler », dont on ne donnera que le nom dans la classe Véhicule, et qui sera implémentée, autrement dit son code sera écrit, seulement dans les classes concrètes qui héritent de Véhicule, soient Voiture et Camion. On dit que la méthode abstraite « rouler » est polymorphe car sa forme finale, autrement dit le code qu'elle exécute, dépend de la classe concrète auquel appartient l'objet à partir duquelle elle est appelée.

Le polymorphisme permet de réduire le nombre de structures de contrôle « if...then...else » dans le programme. Il n'est pas nécessaire de savoir si le véhicule que l'on manipule est une voiture ou un camion pour le faire rouler : il suffit d'appeler sa méthode rouler, qui appellera automatiquement la bonne implémentation (celle de la voiture ou du camion).

La phrase « *notre voiture et un camion roulent* » est programmée par le code suivant qui utilise la méthode abstraite polymorphe « rouler » :

```

camion = Camion;
voiture.rouler();
camion.rouler();

```

Au préalable, on a ajouté à la classe Véhicule la méthode abstraite « rouler » :

```

classdef (Abstract) Vehicule
    % code non-recopié ici
    methods (Abstract)
        rouler()

```

```
    end  
end
```

et on l'a implémenté dans les classes concrètes qui héritent de Véhicule, « Voiture » :

```
classdef Voiture < Vehicule  
    %code non-recopié ici  
    methods  
        function rouler(soi)  
            %implémentation de la méthode rouler  
            %pour une voiture ici  
        end  
    end
```

et « Camion » :

```
classdef Camion < Vehicule  
    methods  
        function rouler(soi)  
            %implémentation de la méthode rouler  
            %pour un camion ici  
        end  
    end
```

Exemple d'illustration schématique inspirée du formalisme UML

L'Unified Modelling Language (UML) donne des règles pour illustrer un programme écrit suivant le paradigme orienté objet. Chaque classe est illustrée par une boîte, qui contient au sommet le nom de la classe, puis ses propriétés, puis ses méthodes. Une classe abstraite est dénotée par un nom écrit en italique, une méthode statique par un nom souligné, et une méthode abstraite par un nom en italique.

Si un objet A contient dans ses propriétés un objet B (dans notre exemple une voiture contient dans ses propriétés une porte), alors la boîte de la classe de l'objet A aura un trait commençant par un losange au niveau de l'objet A pour finir par une flèche simple (un V) au niveau de la boîte de la classe de l'objet B. Si les objets A et B se contiennent l'un l'autre dans leurs propriétés, alors le trait qui les relie est terminé par deux flèches. Le nombre d'instances de l'objet B que peut contenir en propriété un objet A sera dénoté par un nombre, appelé *cardinalité*, écrit à côté de la flèche.

Si une classe A hérite d'une classe B (dans notre exemple la classe Voiture hérite de la classe Véhicule) alors la boîte de la classe de l'objet A aura un trait qui fini par une flèche fermée (en V fermé) au niveau de la boîte de la classe de l'objet B. On a fait un diagramme correspondant aux phrases données au paragraphe 6.3.2 pour expliciter les notations inspirées du formalisme UML que je vais utiliser pour décrire les classes de mon logiciel, sur la figure 6 page suivante .

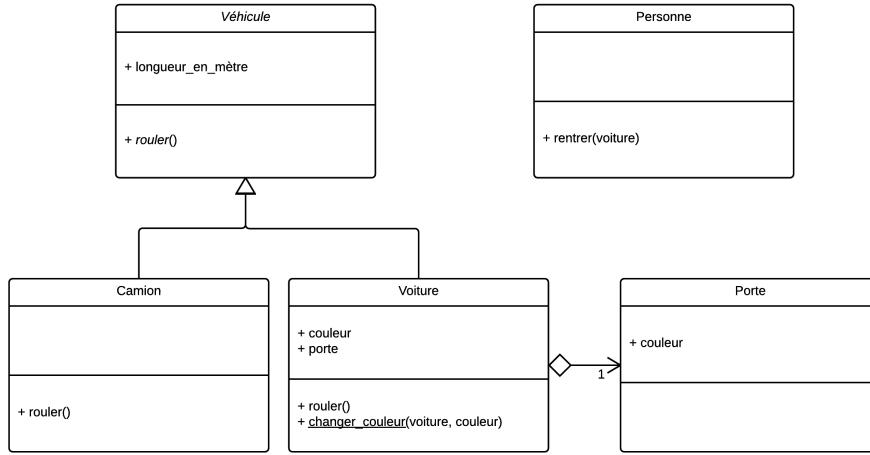


FIGURE 6 – Illustration de l'exemple de programmation orientée objet donné au paragraphe 6.3.2 page 19

Selon le patron de conception Modèle-Vue-Contrôleur Un patron de conception correspond à une certaine organisation d'objets au rôle prédéterminé, dans un programme. Le patron de conception utilisé dans mon programme s'appelle « Modèle-Vue-Contrôleur ». Ce patron de conception permet de séparer les traitements des données qui se font dans l'instance de la classe Modèle (et les instances des classes qu'elle contient dans ses propriétés), tandis que la gestion de l'interface se fait dans l'instance de la classe Vue, et que l'instance la classe Contrôleur appelle tantôt les méthodes du modèle, tantôt les méthodes de la vue, et se pose en chef d'orchestre du programme.

Le modèle a la particularité d'être observé par la vue, ce qui veut dire qu'en cas de changement des propriétés observables du modèle, la vue pourra effectuer des actions (par exemple changer l'affichage). Il est important de noter que la vue est ici appelée implicitement : jamais le modèle n'appelle directement la vue. Cela permet de désolidariser le traitement des données de l'affichage, ce qui sépare le code en différents morceaux (ce qui le simplifie et permet une collaboration plus simple entre deux développeurs dont l'un peut s'occuper de l'affichage et l'autre des traitements) et permet si on le souhaite de changer l'affichage du programme sans toucher à ses traitements. La relation qui lie le modèle à la vue est elle-même un « sous » patron de conception, appelé patron de conception observateur.

Schéma de la structure de mon programme Selon le même formalisme que celui présenté au paragraphe 6.3.2 page précédente, j'ai illustré l'organisation des classes de mon programme par le schéma donné à la figure 7 page 29. Par rapport à ce que l'on a déjà présenté, on a ajouté deux dessins de bon-

hommes que l'on trouve dans les diagrammes UML de scénarios d'utilisation. Le bonhomme situé tout en haut du schéma est l'utilisateur, tandis que le bonhomme situé un peu plus bas est le testeur. Ces deux hommes correspondent aux deux scénarios d'utilisation envisagés qui sont l'utilisation simple du programme, et le test. Les boîtes simples correspondent à des fonctions.

6.3.3 Compréhensible

Lisible Les variables ont des noms explicites qui permettent de comprendre facilement ce qu'elles contiennent. Les différents mots constituant les variables sont séparées par des tirets bas. Par exemple, la variable *pic_choisi* contient le numéro du pic choisi (la numérotation des pics se fait de gauche à droite sur le graphique).

Les noms des classes commencent par des majuscules.

Commenté Le code est commenté à toutes les échelles. Les classes ont un commentaire qui explique leur raison d'être, les méthodes l'action qu'elles effectuent, et les propriétés la caractéristique qu'elles définissent, quand celle-ci n'est pas évidente d'après leur nom. L'intérieur des méthodes est elle-aussi commentée.

Ces commentaires permettent de comprendre l'utilité des différents composants du programme, et permet une modification facilitée par une compréhension claire de son fonctionnement.

Documenté Ce document et ses nombreux schémas permettront d'aider celui qui voudrait comprendre ou modifier le code source.

6.3.4 Évolutif

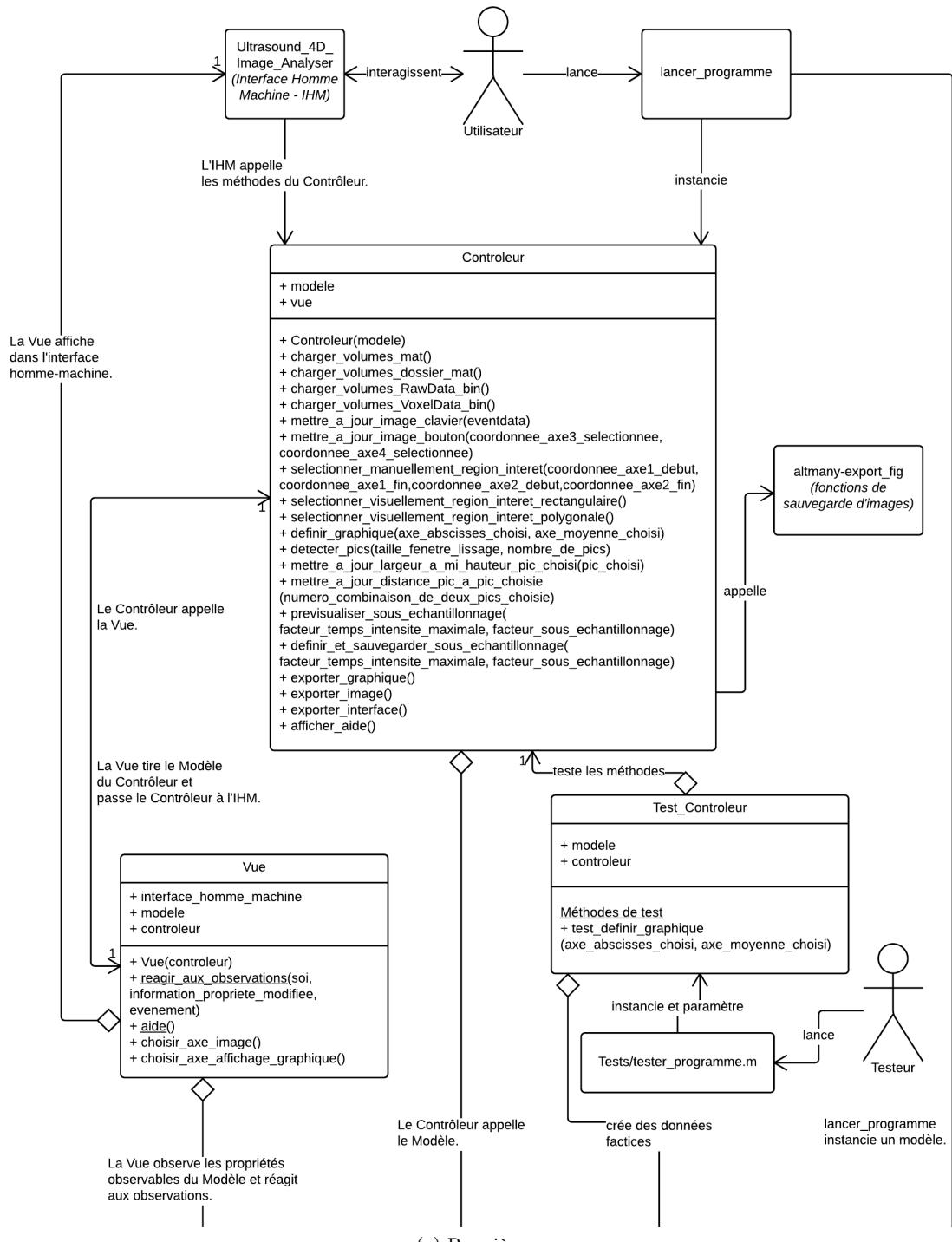
L'écriture du code qui suit un paradigme de programmation orientée objet et le patron de conception modèle-vue-contrôleur permet un important découpage du code en différents modules, ce qui permet d'ajouter de nouvelles fonctionnalités au programme sans risquer d'entrainer des bogues sur d'autres parties du programme (dites erreurs de regression).

Le patron de conception modèle-vue-contrôleur permet de changer l'interface graphique sans avoir à toucher au modèle qui s'occupe du traitement de données, et vice versa.

Le polymorphisme de certaines méthodes permet de limiter la nécessité d'utiliser des structures de contrôles « if...then...else ».

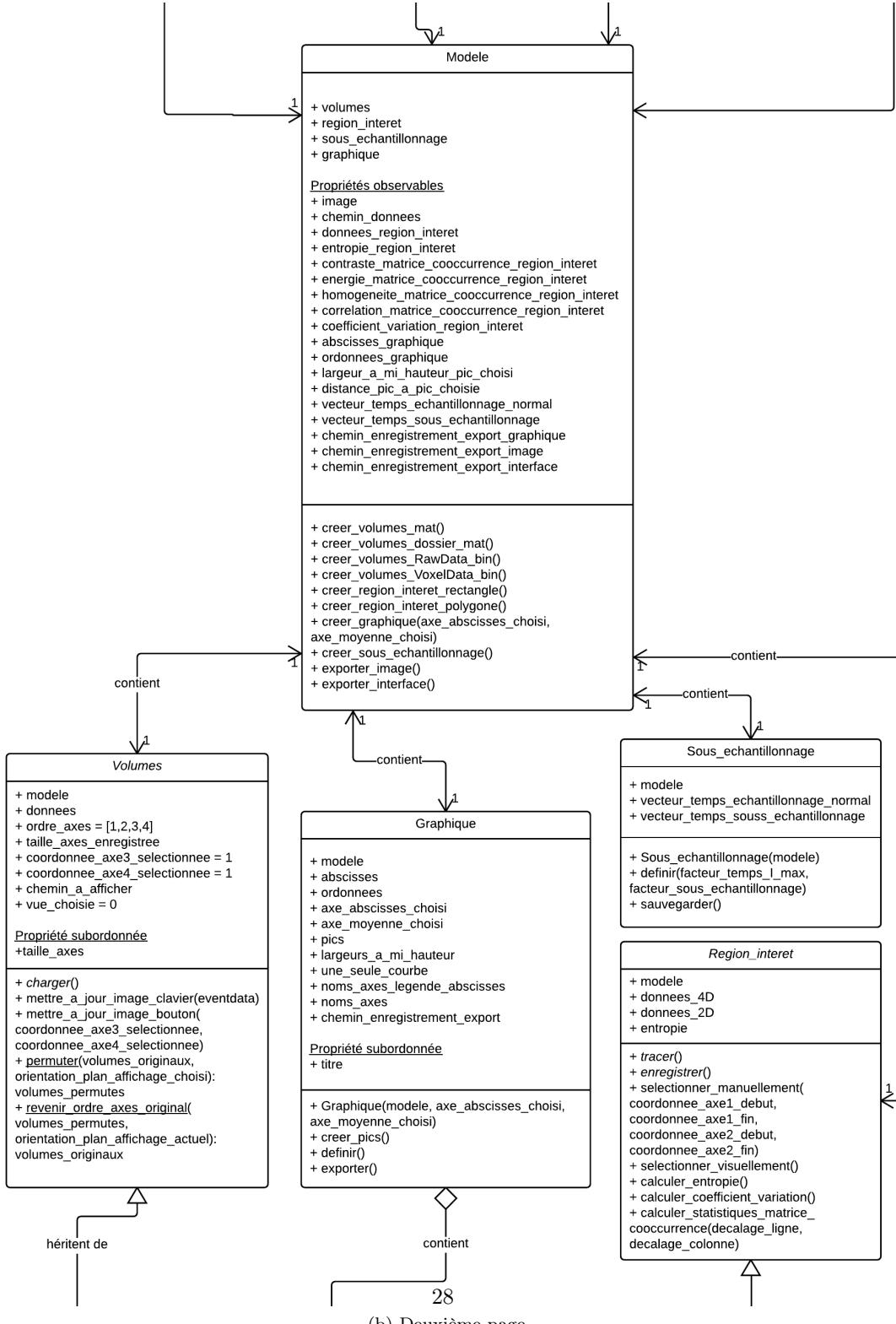
On détaille ci-dessous un exemple d'un ajout de fonctionnalité pour mieux comprendre comment l'évolution du code est possible.

Exemple d'ajout de fonctionnalité : chargement d'un autre format de fichier Pour permettre de charger un nouveau format de fichier, par exemple



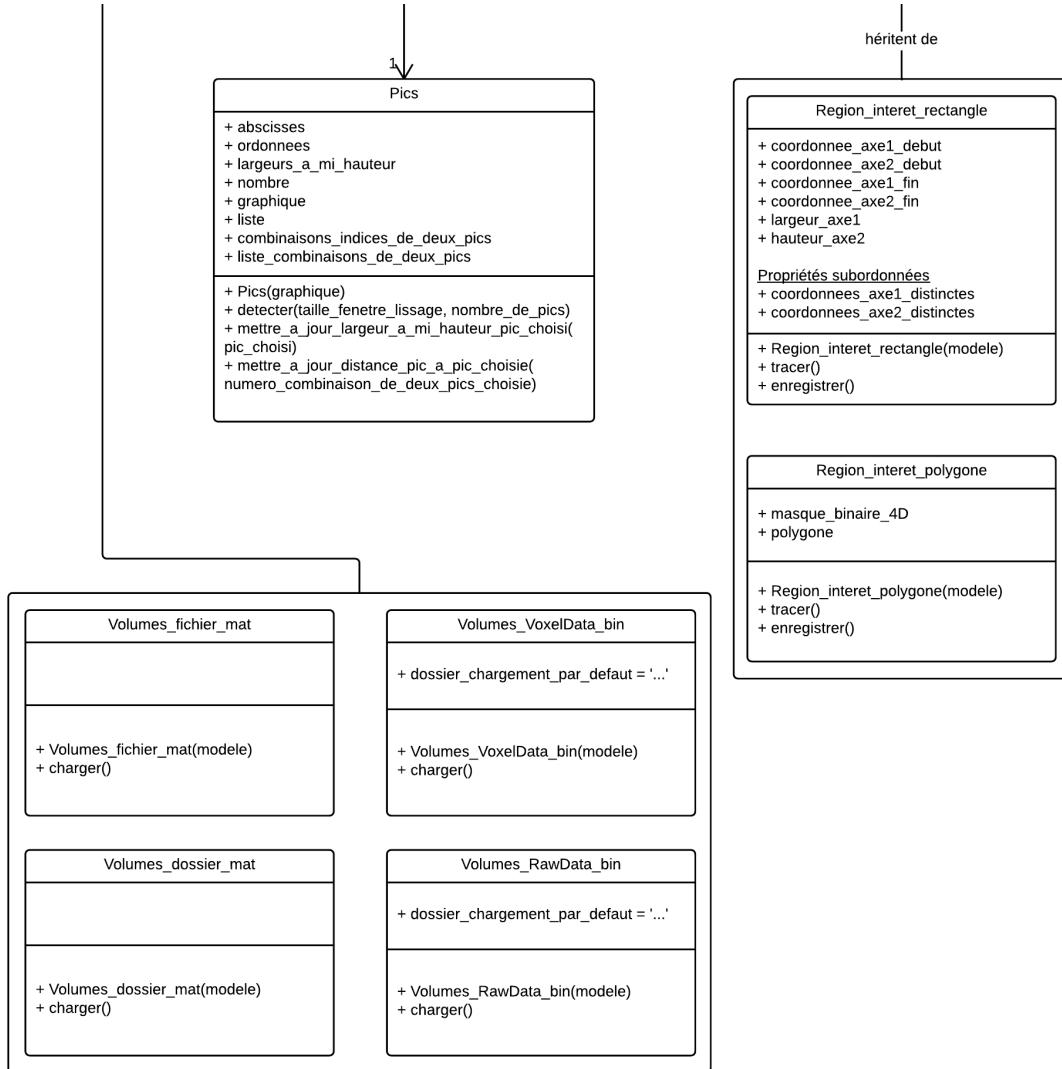
(a) Première page

FIGURE 7 – Schéma de fonctionnement du programme, 1/3
27



(b) Deuxième page

FIGURE 7 – Schéma de fonctionnement du programme, 2/3



(c) Troisième page

FIGURE 7 – Schéma de fonctionnement du programme (3/3) d'analyse d'images ultrasonore quadridimensionnelles "Ultrasound_4D_Image_Analyser" que j'ai développé, inspiré du formalisme des diagrammes de classes et des scénarios d'utilisations de l'Unified Modeling Language (UML)

le format correspondant aux données « 2D+Temps » ou un format correspondant aux données provenant d'un autre échographe, il faudrait suivre les étapes suivantes :

1. créer une classe appelée Volumes_ \$format, où \$format doit être remplacé par le nom relatif au format du type de fichier que l'on veut pouvoir charger ;

2. faire hériter cette classe de la classe Volumes en mettant devant son nom (sur la première ligne de la classe) le code :

```
< Volumes
```

3. créer un constructeur de cette classe qui lui permet de connaître le modèle, en ajoutant les lignes de code suivantes dans la classe :

```
methods ( Access = ?Modele)
% Seul le modèle peut appeler cette méthode
function soi = Volumes_ $format(modele)
    soi . modele = modele;
end
end
```

4. définir une méthode « charger » (implémentation de la méthode abstraite « charger » de la classe abstraite Volumes) dans cette classe et écrire son code permettant de charger le type fichier voulu. Cette méthode doit se finir par la ligne suivante :

```
soi . donnees = variable _ contenant _ les _ donnees _ a _ charger;
```

5. ajouter une méthode créer_volumes_ \$format dans la classe Modèle, qui appelle le constructeur de la classe Volumes_ \$format pour charger une instance de cette classe dans la propriété volumes du modèle. Son code doit être :

```
methods
    function creer_volumes_ $format(soi)
        soi . volumes = Volumes_dossier_mat(soi);
    end
end
```

6. ajouter une méthode dans la classe Contrôleur qui va appeler la fonction précédente du modèle et qui va lancer le chargement. Son code doit être :

```
methods
    function charger_volumes_ $format(soi)
        soi . modele . creer_volumes_ $format;
        soi . modele . volumes . charger;
    end
end
```

7. ajouter le nom du nouveau format chargeable dans la liste des formats de l'interface grâce à l'outil GUIDE (ouvrir le dossier InterfaceHommeMachine puis faire un clic droit sur Ultrasound_4D/Images_Analyser.fig puis cliquer sur GUIDE) ;
8. ajouter les lignes suivantes dans la fonction *chargement_callback* du fichier InterfaceHommeMachine/ Ultrasound_4D/Images_Analyser.m :

```

switch choix_chargement
%code non-recopié ici
case 5
    format = '$format';
end

switch format
%code non-recopié ici
case '$format',
    handles.controleur.charger_volumes_$format;
end

```

6.3.5 Robuste

Gestion des erreurs Les erreurs, parfois appelées « exceptions », sont gérées au moyen de blocs *try...catch*. Le code est exécuté dans la partie *try*, puis si une erreur est levée (soit par une fonction appelée, soit par le code lui-même), alors l'exécution passe au bloc *catch*. Si le nom de l'erreur est reconnu dans le bloc *catch*, alors un code spécifique dit de gestion d'erreur s'exécute. Par exemple, on affiche un message d'erreur pour que l'utilisateur puisse corriger son erreur.

Exemple de test unitaire Un test unitaire permet de tester si une fonction ou une méthode donne pour une entrée donnée le résultat auquel on s'attend. J'ai écrit un test unitaire partiel pour la méthode *définir_graphique* de la classe *graphique*, qui s'occupe du calcul des ordonnées et des abscisses du graphique. Mon test vérifie que pour une région d'intérêt polygonale ne contenant que des valeur d'intensité 0, le graphique doit avoir pour ordonnées un vecteur de 0. Le test peut être lancé en exécutant la fonction *Tests/lancer_tests.m*.

Si j'avais été plus rigoureux, il aurait fallu faire beaucoup plus de tests, de sorte à au moins tester 60% des méthodes que j'ai écrites. Malheureusement, j'ai manqué de temps pour faire cela, et j'ai privilégié les autres aspects du logiciel présentés dans cette partie. Le test que j'ai écrit dans le dossier *Tests* permet néanmoins à un développeur qui reprendrait mon code de pouvoir prendre modèle sur celui-ci pour en écrire d'autres.

Cinquième partie

Résultats

7 Introduction

Comme on l'a présenté à la sous-section 4.3.1 page 6, on souhaite calculer la résolution spatiale de l'Apilo 500 suivant chacun des axes selon lesquels il fait des images, décrits à la figure 2 page 9. Stéphanie, Virginie et moi avons réalisé une expérience dont nous avons analysé les résultats pour déterminer la résolution spatiale selon l'axe x de l'échographe. Malheureusement, les sources de signal étaient encore un peu trop grandes pour que l'on puisse les considérer comme ponctuelles dans la définition de la résolution spatiale. Néanmoins, on suppose que notre résultat est une bonne approximation de la résolution spatiale selon l'axe x .

8 Expérience

Un fantôme en matériau DragonSkin traversé par deux tubes d'environ 275 μm de diamètre a été réalisé par Virginie Grand-Perret. Les deux tubes sont espacés d'environ 5500 μm . Elle a injecté des microbulles dans les deux tubes tout en imageant le fantôme en échographie de contraste avec l'Apilo 500, selon le dispositif présenté à la section 5 page 10.

9 Analyse des images

Après avoir enregistré les images capturées par l'Apilo 500, on les convertit en VoxelData, puis on les charge dans le logiciel Ultrasound_4D/Images_Analyser que j'ai développé. On sélectionne une région d'intérêt dans une image orientée axialement (plan $x-y$). L'image correspond à l'intensité du deuxième harmonique du signal ultrasonore renvoyé, comme on a l'expliqué à la section 2 page 4. La région d'intérêt définit la partie de l'image où se trouve le signal provenant des microbulles contenues dans les deux tuyaux. L'image et la région d'intérêt qu'on a définie sont montrées à la figure 8 page suivante.

On décide ensuite d'afficher le graphique de la courbe d'intensité de la région d'intérêt dont l'axe des abscisses est x et l'axe de moyennage y . On détecte les deux pics du graphique, comme on le voit à la figure 9 page 34.

On calcule ensuite la largeur à mi-hauteur de chacun des pics en pixels, et la distance qui les sépare en pixels, comme on le voit à la figure 10 page 35.

On a que la largeur à mi-hauteur du premier pic (à l'abscisse 118) est de 8,74583 pixels et que la largeur à mi-hauteur du deuxième pic (à l'abscisse 150) est de 9,58443 pixels. En moyenne, la largeur à mi-hauteur des deux pics est donc de $\frac{8,74583+9,58443}{2} = 9,15$ pixels.

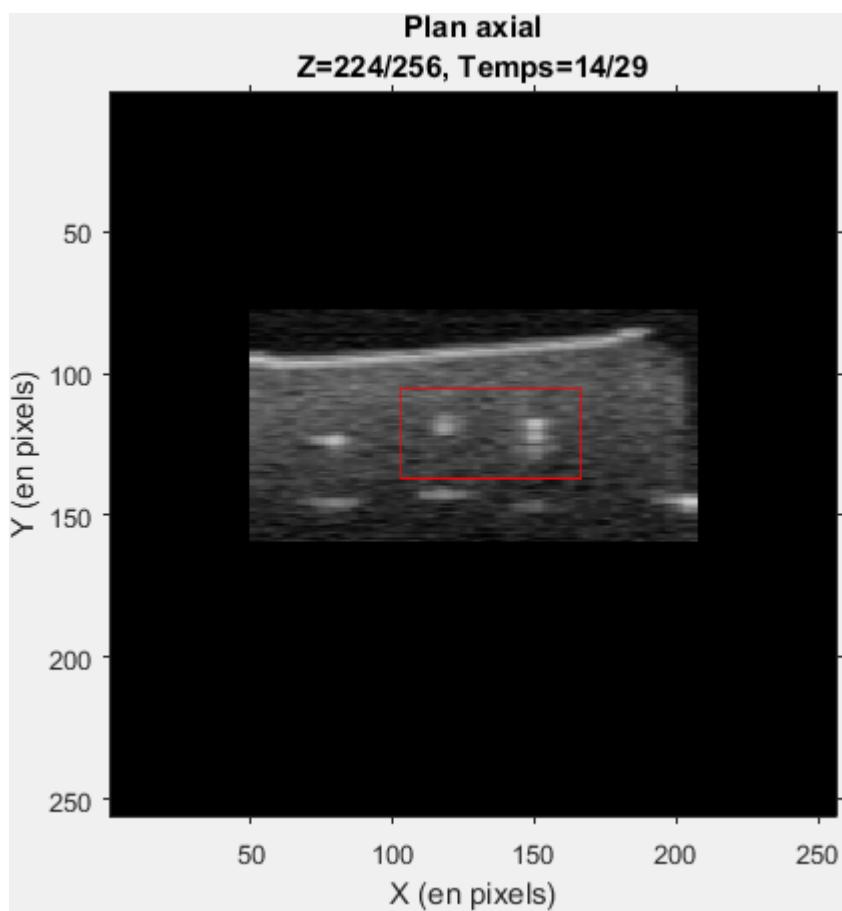


FIGURE 8 – Région d'intérêt sélectionnée sur l'image échographique.
La région d'intérêt est en rouge, et à l'intérieur les deux points blancs correspondent au deuxième harmonique du signal envoyé par les microbulles situées à l'intérieur des tuyaux.

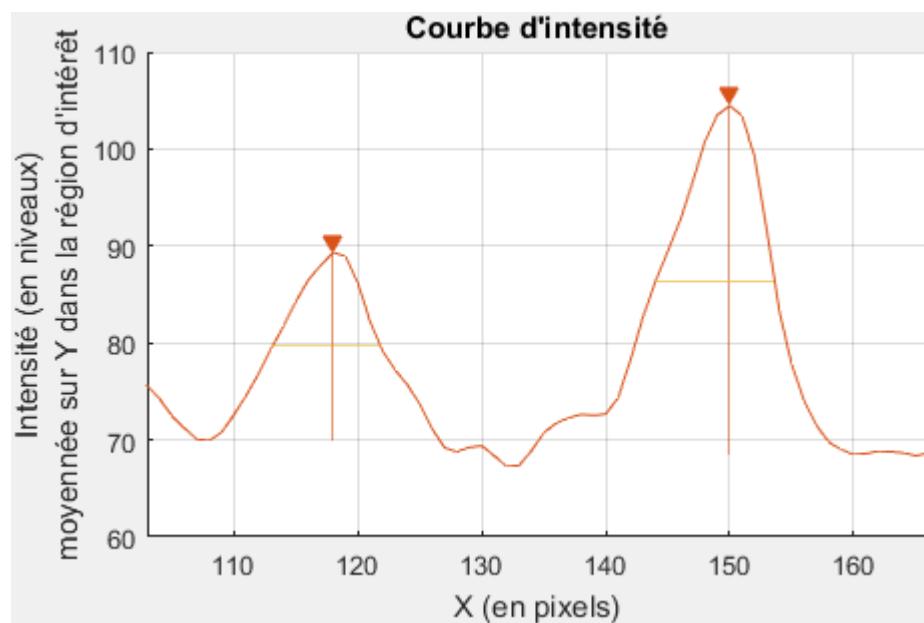


FIGURE 9 – Courbe d'intensité de la région d'intérêt de la figure dont l'axe des abscisses est x et l'axe de moyennage y .

Les pics (maximums locaux) sont marqués par des triangles oranges et la largeur à mi-hauteur de chacun des pics correspond à la ligne horizontale jaune qui se trouve en dessous. Le trait vertical orange qui prend son origine au pic correspond à la hauteur du pic par rapport au signal de base.

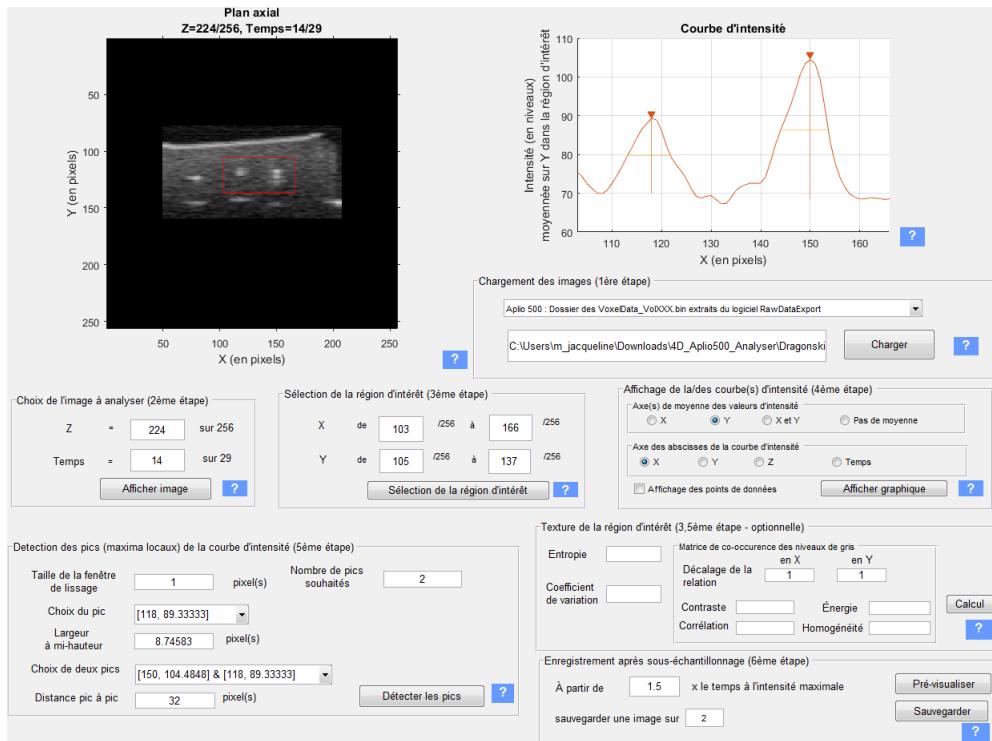


FIGURE 10 – Interface graphique du programme qui présente les résultats de l'analyse du graphique, dont la largeur à mi-hauteur du pic situé à l'abscisse 118, et la distance à mi-hauteur entre les deux pics.

La distance entre les deux pics est de 32 pixels. Comme on sait que les deux tubes sont espacés de 5500 µm, on en déduit que chaque pixel fait $\frac{5500}{32} = 172$ µm. Ainsi, d'après la définition donnée dans la section 4.3.1 page 6, la résolution spatiale de l'Aplio 500 suivant l'axe x est de $9,15 * 172 = 1570$ µm, soit 1,57 millimètre.

Références

- [1] Examen non-invasif, wikipédia. URL https://fr.wikipedia.org/wiki/Examen_non_invasif.
- [2] Fonction, trésor de la langue française informatisé. URL <http://www.cnrtl.fr/definition/fonction>.
- [3] Image resolution, chapitre spatial resolution. URL https://en.wikipedia.org/wiki/Image_resolution#Spatial_resolution.
- [4] Tumeur, dictionnaire en ligne larousse. URL <http://www.larousse.fr/dictionnaires/francais/tumeur/80202>.
- [5] Quelle est la différence entre imagerie fonctionnelle et imagerie anatomique ?, 2007. URL http://www.petscan.info/index.php%3Foption%3Dcom_content%26view%3Darticle%26id%3D45:fondionnelleVsAnatomique%26catid%3D31:general%26Itemid%3D46.
- [6] Philippe ARBEILLE. Risques d'effets biologiques par echographie, doppler pulse et couleur. In *Société Francophone pour l'Application des Ultrasons à la Médecine et à la Biologie (SFAUMB)*, 2003.
- [7] Gabriele Bergers and Laura E Benjamin. Tumorigenesis and the angiogenic switch. *Nature reviews cancer*, 3(6) :401–410, 2003.
- [8] Laure Boyer. *Imagerie multiparamétrique en échographie de contraste (DCE US) pour caractériser la vascularisation tumorale et son hétérogénéité : de la modélisation numérique à l'expérimentation préclinique*. PhD thesis, 2016.
- [9] Elisabeth BRUSSEAU, Ghada SAID, and Patrick CLARYSSE. Modèle numérique d'estimation 2d de la déformation des tissus mous biologiques à partir d'images échographiques radiofréquence : Résultats sur simulations numériques et données expérimentales. In *20° Colloque sur le traitement du signal et des images, FRA, 2005*. GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2005.
- [10] Peter Carmeliet and Rakesh K Jain. Molecular mechanisms and clinical applications of angiogenesis. *Nature*, 473(7347) :298–307, 2011.
- [11] Martin O Culjat, David Goldenberg, Priyamvada Tewari, and Rahul S Singh. A review of tissue substitutes for ultrasound imaging. *Ultrasound in medicine & biology*, 36(6) :861–873, 2010.
- [12] Woods Gonzalez and Richard E Woods. Eddins, digital image processing using matlab. *Third New Jersey : Prentice Hall*, 2004.

- [13] Fabian Kiessling, Daniel Razansky, and Frauke Alves. Anatomical and microstructural imaging of angiogenesis. *European journal of nuclear medicine and molecular imaging*, 37(1) :4–19, 2010.
- [14] Vinay Kumar, Abul K Abbas, and Jon C Aster. *Robbins basic pathology*. Elsevier Health Sciences, 2012.
- [15] Joanne R Less, Thomas C Skalak, Eva M Sevick, and Rakesh K Jain. Microvascular architecture in a mammary carcinoma : branching patterns and vessel dimensions. *Cancer research*, 51(1) :265–273, 1991.
- [16] P Peronneau, N Lassau, I Leguerney, A Roche, and D Cosgrove. Contrast ultrasonography : necessity of linear data processing for the quantification of tumor vascularization. *Ultraschall in der Medizin-European Journal of Ultrasound*, 31(04) :370–378, 2010.
- [17] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1) :3–55, 2001.
- [18] Susan Raatz Stephenson. 3d and 4d sonography history and theory. *Journal of Diagnostic Medical Sonography*, 21(5) :392–399, 2005.
- [19] François Tranquart, Jean-Michel Correas, and Ayache Bouakaz. *Echographie de contraste : méthodologie et applications cliniques*. Springer Science & Business Media, 2007.
- [20] Jianda Yuan, Priti S Hegde, Raphael Clynes, Periklis G Foukas, Alexandre Harari, Thomas O Kleen, Pia Kvistborg, Cristina Maccalli, Holden T Maecker, David B Page, et al. Novel technologies and emerging biomarkers for personalized cancer immunotherapy. *Journal for immunotherapy of cancer*, 4(1) :1, 2016.