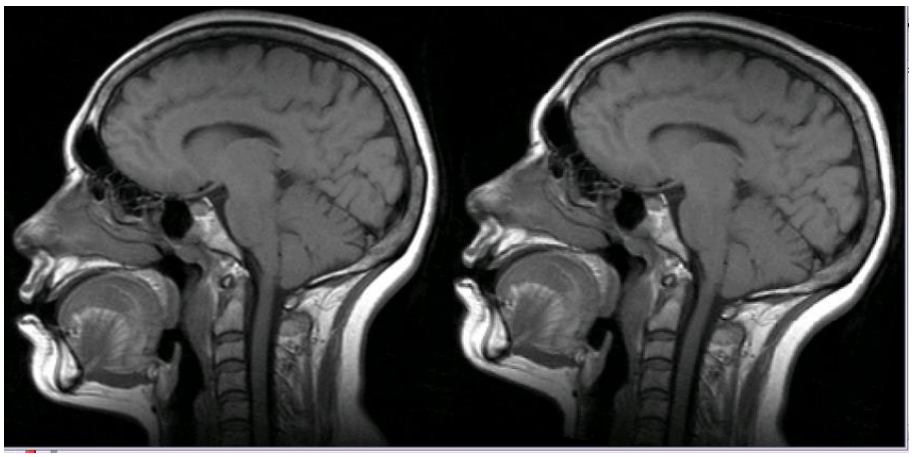
Institut Supérieur de l'Électronique et du Numérique
Tél. : +33 (0)2.98.03.84.00
Fax : +33 (0)2.98.03.84.10
20, rue Cuirassé Bretagne
CS 42807 - 29228 BREST Cedex 2 - **FRANCE**

M2 - TBM

Année scolaire 2021/2022

Rapport scientifique de TP

Recalage automatique d'images médicales



Proposé par : Mr Jean-Marie Guyader

Thématique : Technologies biomédicales de santé

DUPONT Jules — Domaine professionnel : TBM
HOCINE Elouan — Domaine professionnel : TBM

Sommaire

Introduction

Le recalage d'images est une technique dont l'utilisation est de plus en plus demandée de nos jours. La multiplication des centres d'examens, du nombre d'examens, ainsi que des différents appareils à disposition des médecins fait que le nombre d'images produites au cours d'un parcours médical a explosé. Néanmoins, il est également devenu plus difficile de concilier toutes ces sources d'informations que sont chaque image, et dans ce sens il est utile de pouvoir les superposer. Elles doivent pour cela être parfaitement alignées. Cela permet soit de réunir toutes les informations en une seule image, et ainsi de mettre en commun les avantages et informations de différentes modalités d'images (et de gommer leurs inconvénients), soit de caler des images ayant subies une évolution, qu'elle soit due à des mouvements ou bien au temps.

Cependant, cette technique, si réalisée à la main, prend beaucoup de temps, et pourrait être sujet à des erreurs. Elle devient également difficile à appréhender avec des décalages en trois dimensions. Pour ces raisons, l'automatisation du processus prend tout son sens, et est d'une grande aide aux praticiens. Pour automatiser le recalage d'images, différents procédés existent, et c'est de cela dont il va être question dans ce rapport. Tout d'abord une solution de recalage en 2D à l'aide de translations développée sous Python, puis l'intégration d'une autre mesure de dissimilarité que celle utilisée dans la partie 1, et enfin la présentation de différentes solutions utilisées dans l'industrie.

Partie 1 : Création d'un programme de recalage d'images sous Python

1°) Descriptif de la structure du programme

1.1 Modules utilisés

Différents modules ont été utilisés pour permettre le bon fonctionnement de ce programme. En effet, le besoin de travailler des images, et qui plus est des images médicales, force l'utilisation de bibliothèques adaptées pour faciliter le codage. Leur utilité et leur fonctionnement sont détaillés ci-dessous :

numpy : Numpy est un module python dédié à manipuler des matrices, des tableaux pluridimensionnels, et propose également de nombreuses fonctions qui permettent de faciliter les calculs autours de ces tableaux. Les images étant essentiellement des tableaux de données 2D, potentiellement en 3D pour les images médicales, il va permettre de faciliter leur manipulation.

scipy : Regroupement de bibliothèques, basé en partie sur numpy, centré autour du calcul scientifique. Il est très utilisé pour le traitement d'image, par exemple pour en faciliter l'interpolation, et offre également des possibilités de visualisation des images ainsi que la création de graphes grâce à son module matplotlib .

pydicom : Ce module permet la lecture, l'écriture et la modification de fichiers au format DICOM, un format de fichiers issu d'outils d'acquisition d'images médicales. Ce format contient un nombre élevé d'attributs, très variés dans leur sujet, et les données contenues dans les voxels.

nibabel : Comme pydicom, nibabel est une bibliothèque qui facilite la prise en main d'images médicales avec Python. Le format que nibabel permet d'ouvrir et d'éditer est le format NIfTI. Ce format a été créé pour faciliter le post-traitement des images médicales, car il présente moins d'attributs, mais conserve ceux essentiels pour le traitement d'images.

1.2 Déroulé du programme

Le programme qui a été développé suit l'algorithme proposé en figure 14 (en annexe), comme on peut le voir sur le diagramme proposé en figure 15 (en annexe). Ce programme doit permettre d'effectuer un recalage entre deux images 2D au format NIfTI, à l'aide de translations uniquement, et doit pour finir enregistrer l'image recalée au format NIfTI.

Pour faciliter la compréhension du programme, certaines étapes ont été regroupées, ou codées, dans des fonctions. Deux fonctions ont donc été codées :

- *float_Dissim*, qui se charge du calcul de la dissimilarité entre deux matrices données en entrée. La dissimilarité est une métrique traduisant à quel point 2 images sont différentes, elle sera donc nulle pour 2 images identiques. Dans ce cas, la somme des différences au carré a été choisie pour cette métrique, et elle est calculée selon la formule donnée en figure 2, et pour permettre la somme qui y est présente, une double boucle for est utilisée pour permettre de prendre en compte tous les pixels de l'images. Elle renvoie un flottant, qui est la mesure de dissimilarité C.

$$C\left(I_F, I_M\left(T_{\vec{\mu}}\right)\right) = \frac{1}{|\Omega_F|} \sum_{\vec{x} \in \Omega_F} \left(I_F(\vec{x}) - I_M\left(T_{\vec{\mu}}(\vec{x})\right)\right)^2$$

Figure 1 – Formule du calcul de la mesure de dissimilarité

- *listFl_GradDissim*, qui est chargée du calcul du gradient de dissimilarité, en prenant également en entrée 2 matrices. Le gradient va permettre de déterminer dans quelle direction et avec quelle intensité le programme doit translater l'image mobile pour qu'elle se cale sur l'image fixe. Cette fonction suit la formule présentée en figure 3, et renvoie une liste contenant les deux résultats attendus : le gradient selon l'axe des x et le gradient selon l'axe des y.

$$\frac{\partial C}{\partial \vec{\mu}} = -\frac{1}{n} \times 2 \times \sum_{\vec{x} \in \Omega_F} \left[\left(I_F(\vec{x}) - I_M(T_\mu(\vec{x}))\right) \times \frac{\partial I_M(T_\mu(\vec{x}))}{\partial T_\mu(\vec{x})} \right]$$

$$= -\frac{1}{n} \times 2 \times \sum_{\vec{x} \in \Omega_F} \left[\left(I_F(\vec{x}) - I_M(T_\mu(\vec{x}))\right) \times \begin{pmatrix} \frac{\partial I_M(T_\mu(x))}{\partial T_\mu(x)}\Big|_{\vec{x}} \\ \frac{\partial I_M(T_\mu(y))}{\partial T_\mu(y)}\Big|_{\vec{x}} \end{pmatrix} \right]$$

Figure 2 – Formule du calcul du gradient de la mesure de dissimilarité

Le programme commence donc par une phase d'import des modules nécessaires, puis de définition des fonctions. Il effectue ensuite un nettoyage du dossier permettant le stockage des images afin de ne pas avoir d'images provenant d'anciennes exécutions du programme. Vient ensuite la phase d'ouverture des fichiers NIfTI, celui de l'image fixe (l'objectif à atteindre) et de l'image mobile (l'image décalée). Ces fichiers sont ouverts, puis les données des voxels sont converties en matrices numpy. Enfin, le début du programme finit par une phase d'initialisation des paramètres (Nombre d'itérations, pas des itérations) ainsi que des variables nécessaires à son bon fonctionnement.

La boucle principale du programme commence alors, effectuant le nombre d'itérations définies au début. La phase d'échantillonnage n'est pas effectuée car, considérant la taille réduite de l'image, la capacité de calcul d'un ordinateur moyen est suffisante pour considérer tous les pixels de l'image et non juste une partie. Vient ensuite l'interpolation de l'image mobile, en prenant en compte la valeur actuelle de la transformation. Cette partie est nécessaire, car si non effectuée, le programme pourrait tenter d'accéder à des valeurs de voxel qui n'ont plus de sens en prenant en compte les translations. Pour cela une fonction de la bibliothèque ndimage de scipy est utilisée : la fonction *interpolation.shift*.

On calcule ensuite la mesure de dissimilarité en entrant la matrice de l'image fixe, et la matrice de l'image interpolée, dans la fonction *float_Dissim.* On calcule également avec ces 2 mêmes matrices la valeur des gradients x et y, grâce à *listFl_GradDissim*. Ces valeurs sont stockées pour être ensuite utilisées dans le calcul du nouveau vecteur de transformation, qui se calcule en se basant sur l'état précédent, et dont l'amplitude va dépendre du pas de transformation qui va venir pondérer la valeur du gradient, comme on peut le voir dans la formule suivante (avec μ le vecteur contenant les paramètres de la transformation, et a le pas et d le gradient) :

$$\vec{\mu}_{k+1} = \vec{\mu}_k + a_k \times \vec{d}_k$$

Ce nouveau vecteur sera celui utilisé pour interpoler l'image à la prochaine boucle. Pour finir la boucle, un affichage des métriques est réalisé, l'image mobile est enregistrée avec son vecteur de translation actuel dans le dossier *Images*, et le compteur est mis à jour.

Une fois le nombre d'itérations prévues réalisé, la boucle s'arrête, et on enregistre l'image obtenue avec la dernière mise à jour du vecteur de transformation au format NIfTI. On affiche également la courbe de l'évolution de la dissimilarité (comme en figure 3) pour permettre de rendre compte de l'évolution de cette dernière.

1.3 Choix des paramètres

L'une des principales difficultés du recalage d'images est la définition des paramètres. Ces derniers se multiplient rapidement selon le type d'algorithme que l'on développe, et ont également tendance à être dépendants les uns des autres. De plus, des paramètres très efficaces pour une modalité d'image sont susceptibles de ne pas marcher pour une autre. La robustesse de l'algorithme est donc mise à rude épreuve, et trouver des paramètres qui fonctionnent relativement bien dans toutes les situations est difficile. D'autant plus que la seule façon de les améliorer est bien souvent d'adapter empiriquement ces derniers, et notamment le pas de recalage.

Pour permettre une meilleure compréhension du réglage du pas, plusieurs régalages du paramètre ont été testés successivement avec le même nombre d'itérations. Les résultats sont visibles à la figure 3.

Cette figure permet de relever plusieurs résultats d'intérêt : alors qu'un trop grand pas fait osciller de façon trop importante la mesure, et rend la tâche impossible à l'algorithme de recalage (comme le pas de 0.1), un pas trop faible permet certes de décaler l'image avec douceur, mais requiert alors un grand nombre d'itérations et prend donc beaucoup de temps.

La solution choisie pour l'algorithme a donc été de prendre un pas de 0.0005, qui minimise les deux problèmes ci-dessus.

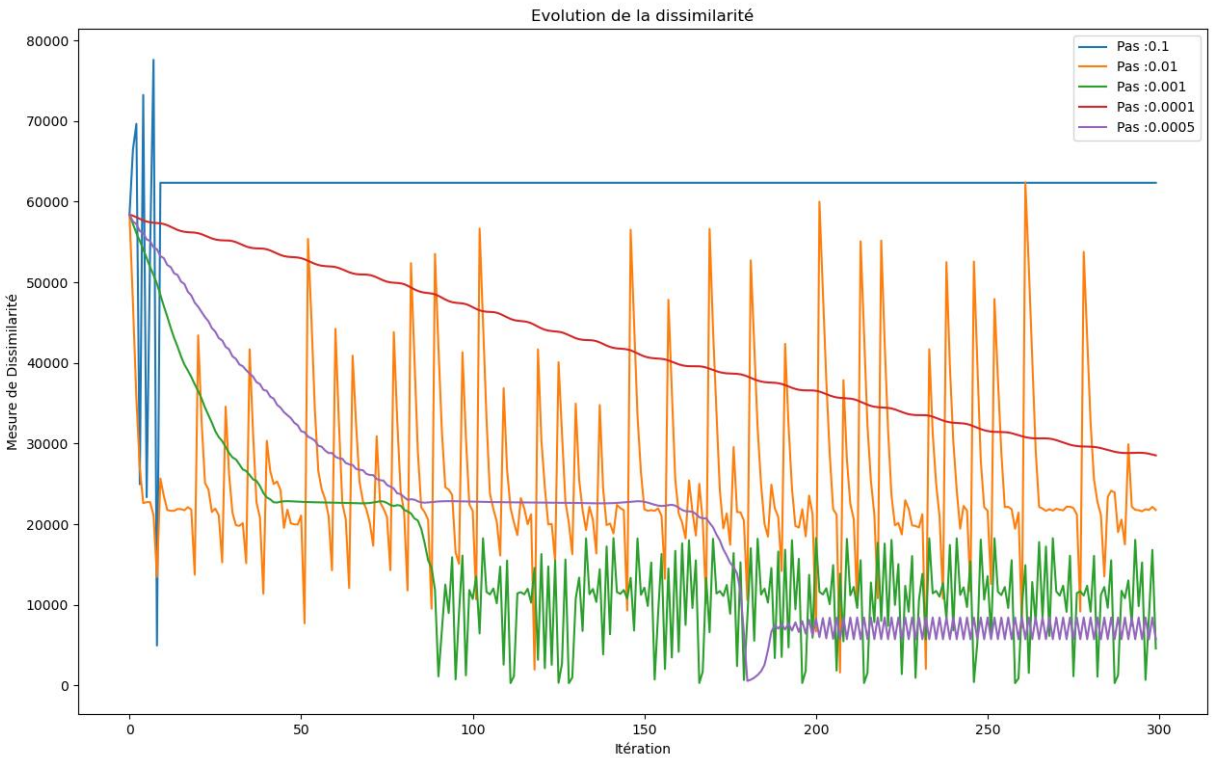


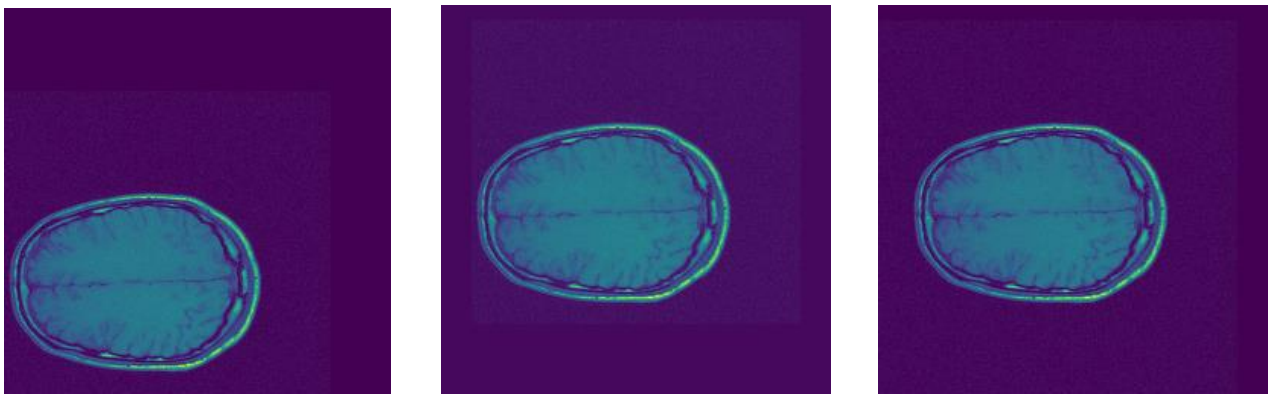
Figure 3 – Evolution des dissimilarités en fonction du nombre d'itérations, avec différentes valeurs de pas.

Le nombre de pas a été réglé à 300. C'est un nombre de pas qui permet largement de recaler les deux images proposées dans l'énoncé, mais ces dernières ne sont pas trop éloignées l'une de l'autre. Ainsi, à supposer que l'on ait besoin d'un plus grand décalage, certes les gradients seraient plus forts ce qui accélèrerait le processus, mais pas suffisamment pour permettre le recalage total de l'image avant la fin des itérations.

2°) Description et analyse des résultats

2.1 Résultats du programme de base

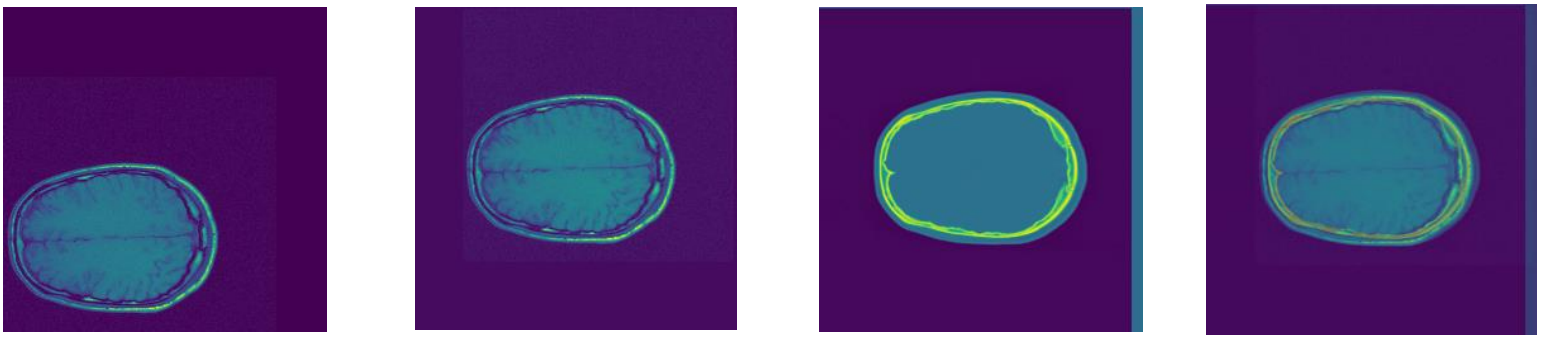
L'algorithme ainsi codé et paramétré obtient de bons résultats. Ainsi, les images proposées dans le sujet, qui sont présentées en figure 4.a et 4.c, permettent à l'issu des 300 itérations choisies, d'avoir une image plutôt bien recalée. La mesure de dissimilarité est ainsi passée de 20 500 à 3700, ce qui confirme le bon recalage de notre image.



(a) (b) (c)

Figure 4 – Images produites par l'algorithme de recalage : (a) Image mobile initiale, (b) Image mobile finale (après 300 itérations), (c) Image fixe initiale, l'objectif à atteindre

Le programme fonctionne également avec d'autres images translatées, dont l'usage est disponible au travers du dossier *fichiersEtudiants*, qui permet d'accéder à des images autres que celles proposées, et d'exécuter le programme avec ces images. Le résultat du travail de l'algorithme sera alors observable dans le dossier *Images*, où chaque image intermédiaire produite par l'algorithme, ainsi que l'image finale et l'objectif, sont observables.



(a) (b) (c) (d)

Figure 5 – Images produites par l'algorithme de recalage avec 2 modalités différentes : (a) Image mobile initiale, issue d'une IRM pondérée T1, (b) Image mobile finale (après 100 itérations), (c) Image fixe initiale, l'objectif à atteindre, obtenue avec un scanner, (d) Superposition des 2 images, à 50% de leur valeur d'intensité chacune.

L'algorithme a également été testé avec 2 modalités différentes pour le recalage, afin d'évaluer sa robustesse. La figure 5 présente le résultat de cette exécution, et comme le montre les figures 5.b et 5.c, le recalage a été correctement effectué. On peut donc en déduire que tant que les modifications à faire pour le recalage ne concernent que des translations, l'algorithme sera capable de les réaliser. Il est cependant à noter que la tâche à réaliser est ici facilitée par le fait que les 2 modalités proposent la même taille d'image, un problème qui serait sinon à prendre en compte dans la version du programme *Recalage_ModaliteDiff.py*. La figure 5.d présente l'intérêt de ce genre de recalage, en superposant les 2 modalités, et en permettant par exemple ici de mieux délimiter la boîte crânienne.

Enfin, un test fût également effectué avec des images 3D, mais l'algorithme n'étant pas prévu pour gérer ce genre d'images, il ne fût pas concluant.

2.2 Proposition d'améliorations

Afin d'améliorer le résultat obtenu par l'algorithme, il est possible d'envisager plusieurs pistes.

Tout d'abord, ajouter le processus d'échantillonnage permettrait d'avoir un programme plus robuste à des tailles d'images plus importantes que celles étudiées, et potentiellement d'augmenter les performances temporelles du programme, et lui permettre de réaliser plus d'itérations et donc de mieux fonctionner sur de long décalages, ou bien de diminuer le pas pour un meilleur résultat. Il est à noter qu'il faudrait également adapter l'interpolation en conséquence pour étudier les mêmes voxels lors de l'étude de la similarité des images.

On pourrait aussi imaginer utiliser le flou gaussien pour accélérer et améliorer le processus, avec une phase où l'algorithme travaille sur une image floutée avec un pas élevé, puis une seconde phase où l'algorithme repart avec les transformations de la première phase, mais travaille avec les images nettes. Cette seconde se ferait avec un pas plus faible pour permettre des transformations plus fines, les mouvements plus importants ayant déjà été réalisés à la première phase. Le nombre de phases pourrait augmenter avec différents niveaux de flous. Un exemple de cette solution est trouvable sous le nom *Recalage_Flou.py* dans le dossier des programmes.

Les résultats de l'implémentation de cette méthode sont très encourageants : on obtient en effet à l'issu du premier recalage une mesure de dissimilarité de seulement 350 (à comparer à celle de 3700 de la

méthode originale), et qui est ensuite affinée par la 2ème itération du programme sans le flou gaussien pour arriver à une mesure de dissimilarité de seulement 264 entre les 2 images nettes, ce qui indique en théorie une image bien mieux recalée. On peut supposer que cette amélioration notable est due au fait que les gradients subissent moins d'altérations qui pourraient provenir des détails de l'images, et ainsi cela permet un meilleur recalage. Encore une fois, pour plus d'images des résultats, le dossier *Images* peut être consulté après avoir fait fonctionner l'algorithme de son choix.

Il serait également imaginable d'automatiser le choix de l'image finale, ce qui permettrait d'obtenir la meilleur image possible et non celle de la dernière itération, qui comme vu plus haut, n'est pas forcément la meilleure en fonction du pas. Pour cela, deux méthodes sont envisageables :

- Retenir les valeurs de dissimilarité, et choisir l'image associée à l'itération ayant la plus faible

-Mettre un grand nombre, voir infini, d'itérations à l'algorithme, et arrêter ce dernier lorsque les valeurs de dissimilarité oscillent très faiblement, ou sont contenus dans un intervalle qui n'évolue plus depuis un certain nombre d'itérations.

Cette dernière solution s'éloigne un peu de concept de base du recalage d'image pour se rapprocher du machine learning, mais elle reste une piste.

Enfin, on pourrait imaginer ajouter des rotations, ainsi qu'une prise en compte de l'aspect 3D de certaines images, pour parfaire cet algorithme et lui permettre d'effectuer des recalages plus variés.

Partie 2 : Etude de l'information mutuelle

Au cours de ce travail de développement, une seule mesure de dissimilarité a été utilisée : *la somme des différences au carré (SSD)*. Or il existe une multitude d'autres manières pour mesurer cette dissimilarité.

Dans le sujet il est proposé d'étudier plus précisément la mesure de dissimilarité de type *information mutuelle* (*MI* pour Mutual Information) [1]. Une recherche bibliographique sera donc effectuée dans le but de mettre en évidence son utilité et son fonctionnement, mais surtout de comparer son efficacité par rapport au SSD.

Par la suite, une ébauche de programmation sous python sera fournie où cette information mutuelle sera utilisée. Les résultats seront présentés à la fin de cette partie.

1°) Explication de l'information mutuelle et de son utilité

L'information mutuelle est une mesure statistique de la théorie de l'information. Elle permet de mesurer la dépendance d'un point de vue statistique entre deux variables aléatoires. En effet, l'information mutuelle reflète la quantité d'informations que possède une variable aléatoire sur une seconde variable aléatoire. [2]

De nos jours, l'information mutuelle est la méthode la plus utilisée pour le recalage d'images multimodales. C'est pour cela qu'il est important de se pencher sur son fonctionnement et les avantages qu'elle propose. [3]

Il existe plusieurs façon de représenter cette information mutuelle. Les représentations les plus courantes sont l'entropie et la distance de Kullback-Leibler.

1.1 Explication avec l'entropie

En théorie du signal, l'information mutuelle dérive de la notion d'entropie. Celle-ci se définie comme étant un nombre qui mesure la variabilité d'une variable aléatoire.

L'entropie peut être calculée par la formule illustrée à la figure 6. Il est possible de constater ainsi que plus X délivre des issues imprévisibles, plus l'entropie sera élevée.

$$H(X) = E\left[\log\left(\frac{1}{P_X}\right)\right] = -\sum_{i=1}^{n} p_i \log(p_i)$$

Figure 6 - Formule de l'entropie

Pour rappel, l'information mutuelle mesure la non-indépendance entre deux variables (si les deux variables sont indépendantes, alors l'information mutuelle est nulle). Or la formule présentée ci-dessus n'est représentative que pour le cas d'une variable aléatoire. Il est possible de l'étendre pour deux variables aléatoires.

D'un point de vue mathématique, l'information mutuelle peut être représentée par la formule présentée à la figure 7 avec cette notion d'entropie.

$$MI(R, F) = H(R) + H(F) - H(R, F)$$

Figure 7 - Formule de l'information mutuelle

Ici, H(R) et H(F) désignent respectivement l'entropie d'information de l'image de référence R et de l'image flottante F, et H(R,F) est l'entropie conjointe des deux images.

Il est également possible de réaliser un diagramme de Vann pour représenter cette notion plus visuellement. La formule correspondante est identique à celle de la figure 8, seuls les noms des variables changent. L'information mutuelle est représentée par I(X,Y) et les deux variables aléatoires sont X et Y où H(X) et H(Y) représentent respectivement leur entropie propre et X(X,Y) leur entropie conjointe. [5]

$$I(X,Y) = H(X) + H(Y) - H(X,Y)$$

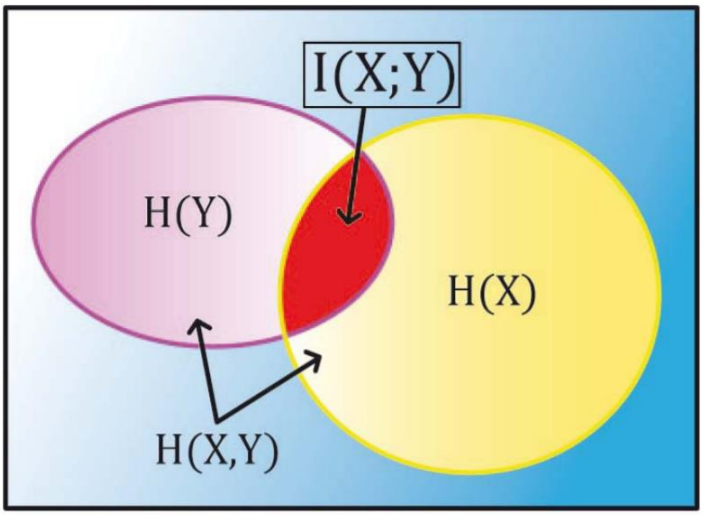


Figure 8 - Diagramme de Vann associé à l'information mutuelle

1.2 Explication avec la divergence de Kullback-Leibler

Cette méthode est beaucoup moins utilisée et documentée dans la documentation scientifique.

La formule calculant l'information mutuelle en utilisant Kullback-Leibler est exprimée en figure 9.

$$I(X;Y) = KL(P(X,Y), P(X)P(Y)) = \sum P(X,Y) \log \frac{P(X,Y)}{P(X)P(Y)}$$

Figure 9 - Formule de Kullback-Leibler

Ainsi, I(X,Y) mesure une sorte de distance entre P(X,Y) et P(X)*P(Y). Or deux variables sont dites indépendantes si et seulement si ces deux distributions sont égales. Ainsi, la formule de Kullback-Leibler respecte bien cette règle, si X=Y, I(X,Y)=0.

2°) Comparaison de l'information mutuelle avec la somme des différences au carré

Après avoir vu cette mesure de dissimilarité par l'information mutuelle dans son allure générale, il serait intéressant de le comparer à la méthode qui a été utilisé dans le travail effectué dans la partie 1.

Ces deux méthodes, bien qu'elles calculent toutes les deux la dissimilarité entre deux images, ne fonctionnent pas de la même manière. Ainsi, il est possible de distinguer des avantages et des inconvénients pour chacune d'entre elles.

2.1 Avantages de l'information mutuelle par rapport au SSD

L'avantage principal de l'information mutuelle est qu'elle permet de recaller des **images multimodales**. Ainsi, à partir d'images issues de diverses méthodes d'acquisition, il est possible d'obtenir une image finale superposant ces deux images différentes (CT, IRM, TEP, …).

Cette fonctionnalité est extrêmement importante et utile pour le médecin dans le but d'établir un diagnostic plus précis et certain. Le superposition des images issues de différentes méthodes d'acquisition est donc primordial et la mesure de dissimilarité par information mutuelle répond à cette demande.

2.2 Limitations de l'information mutuelle et solutions existantes

L'information mutuelle est donc une méthode extrêmement intéressante dans le cas de l'études d'images multimodales. Toutefois, il faut également prendre en compte que cette méthode n'est pas une méthode « miracle » et possède également de nombreux défauts. Ceux-ci sont exposés dans le tableau suivant associés à leurs solutions respectives. [4]

Problèmes de l'IM	***Solutions***
Sensible au recouvrement partiel : Lorsque l'on recale des images de modalités différentes, il arrive qu'une structure, ou plus généralement qu'une partie des données présente dans une des images ne le soit pas dans l'autre. Par ailleurs, du fait des modifications apportées à l'image flottante au cours de sa transformation, la géométrie de l'image peut être impactée et le volume de recouvrement se retrouve modifié.	La méthode la plus efficace pour résoudre ce problème est le calcul de **l'information mutuelle normalisée**. $IMN(\boldsymbol{x}) = \frac{H(x^{(1)}) + H(x^{(2)})}{H(\boldsymbol{x})}$ Il est possible aussi d'utiliser le rapport de corrélation entropique défini par $RCE(\boldsymbol{x}) = \sqrt{2 - \frac{2H(\boldsymbol{x})}{H(x^{(1)}) + H(x^{(2)})}} = \frac{2IM(\boldsymbol{x})}{H(x^{(1)}) + H(x^{(2)})}$ et par l'information exclusive : $IE(\boldsymbol{x}) = H(\boldsymbol{x}) - IM(\boldsymbol{x}) = 2H(\boldsymbol{x}) - H(x^{(1)}) - H(x^{(2)})$.
Aucune interprétation de l'information spatiale : Les voxels ou pixels de l'images étant considérés comme des réalisations de variables aléatoires, la localisation spatiale n'est pas prise en compte dans le calcul de l'information mutuelle.	Des travaux de recherches essaient de prendre en compte cette information positionnelle des pixels et des voxels contenus dans l'image. Parmi ces travaux, il est possible de citer les études sur une IM : - Incorporant une information de gradient, - Ayant une dimension supérieure.
Information mutuelle et histogramme conjoint : L'utilisateur devant choisir les valeurs des paramètres de définition (paramètres de lissage, nombre de classes, largeur des noyaux, …), les performance des méthodes d'estimation sont donc extrêmement variants. Un mauvais choix dans ces paramètres et les résultats peuvent être lourdement impactés. Par ailleurs, en plus de ce problème de paramétrisation une contrainte sur la charge calculatoire peut également apparaitre.	Pour faire face à tous ces problèmes, il serait intéressant de se prémunir de la construction de l'histogramme conjoint. Plusieurs méthodes ont été développées ou sont en stade de recherche afin de résoudre ce problème. L'alternative qui rencontre le plus de succès à l'heure actuelle est l'estimation par graphe d'entropie en utilisant l'entropie de Renyi (autre méthode pour calculer l'entropie).
Information mutuelle se basant sur une autre mesure de l'entropie : Les calculs présentés plus haut résultent de l'entropie de Shannon. De nombreuses autres mesures d'entropies ont émergé dans différents contextes applicatifs depuis celle de Shannon.	L'entropie de Renyi (défini par la formule : $H_\alpha(x) = \frac{1}{1-\alpha}\log\left(\int_{\mathbb{R}} p_x^\alpha(u)\right) du,\ \alpha > 0, \alpha \neq 1$), l'entropie de Havrda-Charvat ou l'entropie généralisés sont ces nouvelles méthodes qui sont certes plus efficaces, mais également plus spécialisées

Pour conclure, la mesure de dissimilarité par l'information mutuelle est très importante dans le domaine de l'imagerie médicale. Elle permet de réaliser un recalage d'images multimodales, c'est-à-dire qu'il est envisageable de faire correspondre des images issues de méthodes d'acquisitions différentes (CT, IRM, PET, …). Toutefois, il existe de nombreuses limites à cette méthode qui sont exposées dans la partie 2.2. De nombreux travaux de recherches sont en cours afin de réduire voire de résoudre ces différentes limitations dans le but d'avoir une méthode de mesure de la dissimilarité associant multimodalité et robustesse.

En ce qui concerne la partie codage, une implémentation du calcul de la mesure de l'information mutuelle a été réalisée. Avec la fonction présente dans le fichier *MI.py*, il est possible de connaitre la valeur de l'information mutuelle entre deux images en entrée. [6]

Comme le montre la figure 10, cette valeur d'information mutuelle suit la valeur de la dissimilarité SSD ce qui est logique. Plus les deux images se ressemblent, plus la valeur de la SSD diminue et plus la valeur de l'IM augmente. A l'inverse, plus les images diffèrent, plus la valeur de la SSD augmente et plus la valeur de l'IM diminue.

Ces résultats semblent cohérents : au fil des itérations, les deux images tendent à se ressembler, cela étant le but du recalage d'image.

```
Num. itération =164 , Mesure Dissimilarité SSD =21550.35 / MI =0.49 , Transformation T: x=-40.12 y=12.39
Num. itération =165 , Mesure Dissimilarité SSD =21273.12 / MI =0.49 , Transformation T: x=-40.36 y=12.58
Num. itération =166 , Mesure Dissimilarité SSD =20873.71 / MI =0.5 , Transformation T: x=-40.64 y=12.78
Num. itération =167 , Mesure Dissimilarité SSD =20717.02 / MI =0.5 , Transformation T: x=-40.95 y=12.97
Num. itération =168 , Mesure Dissimilarité SSD =20752.49 / MI =0.52 , Transformation T: x=-41.31 y=13.16
Num. itération =169 , Mesure Dissimilarité SSD =20158.65 / MI =0.51 , Transformation T: x=-41.77 y=13.36
Num. itération =170 , Mesure Dissimilarité SSD =19625.46 / MI =0.53 , Transformation T: x=-42.32 y=13.56
Num. itération =171 , Mesure Dissimilarité SSD =18722.38 / MI =0.53 , Transformation T: x=-42.96 y=13.78
Num. itération =172 , Mesure Dissimilarité SSD =17992.14 / MI =0.53 , Transformation T: x=-43.6 y=14.03
Num. itération =173 , Mesure Dissimilarité SSD =16888.14 / MI =0.55 , Transformation T: x=-44.17 y=14.34
Num. itération =174 , Mesure Dissimilarité SSD =16103.98 / MI =0.55 , Transformation T: x=-44.58 y=14.75
Num. itération =175 , Mesure Dissimilarité SSD =15197.25 / MI =0.56 , Transformation T: x=-44.78 y=15.25
```

Figure 10 - Affichage en sortie de l'algorithme

Maintenant qu'il est envisageable de calculer l'information mutuelle entre deux images en fonction des itérations, il serait intéressant d'aller au-delàs de cette étape. Actuellement, le programme fonctionne avec une métrique de dissimilarité SSD, or il serait intéressant de réaliser ce recalage avec l'information mutuelle.

Ainsi, grâce à ce changement, il serait théoriquement possible de réaliser un recalage entre deux images multimodales.

Partie 3 : Utilisation de logiciels de recalage d'images existants

A l'heure actuelle, de nombreux logiciels de recalage d'images sont utilisés dans le domaine de la médecine et de la radiologie. Ceux-ci sont important afin de pouvoir donner un diagnostic ou un pronostique à la fois rapide et fiable.

Il serait intéressant de comparer l'algorithme qui a été présenté dans la partie 1, voire l'ébauche utilisant l'information mutuelle de la partie 2, avec ces différents logiciels actuellement utilisés.

1°) Utilisation du logiciel « Elastix »

Elastix est un outil qui permet de recaller une image fixe avec une seconde image mobile. Etant un logiciel Open-source et disponible à l'adresse suivante : https://elastix.lumc.nl/index.php, n'importe qui peut l'installer et le faire fonctionner dans son environnement.

À la suite des explications fournies dans les parties 1 et 2, il serait intéressant de tester ce logiciel et relever son efficacité par rapport à l'algorithme conçue en partie 1.

1.1 Explication du logiciel et fonctionnement

Elastix est un logiciel Open-source et téléchargeable en ligne par tous. Une fois installé, il ne reste plus qu'à avoir deux images à recaller et un fichier de paramétrage. Cette partie pratique sera expliquée plus en détails dans la partie 1.2.

Bien que *Elastix* soit un logiciel utilisable immédiatement, sa prise en main n'est pas simple. Ne possédant pas d'interface homme machine, *Elastix* est utilisable uniquement en ligne de commande. Pour lancer et piloter le programme, il est donc nécessaire de passer par le terminal Windows (ou autre OS). Ainsi, il est donc nécessaire d'avoir un niveau minimum en informatique afin d'utiliser ce logiciel.

Du fait de cette interface peu intuitive, il parait peu probable qu'un médecin puisse utiliser facilement cet outil au quotidien.

1.2 Exemple concret sur des images médicales

Maintenant que l'outil a été brièvement décrit dans la partie précédente, il serait intéressant de réaliser un essai concret en utilisant des images médicales.

La première étape est de télécharger le logiciel *Elastix* à l'adresse suivante : https://github.com/SuperElastix/elastix/releases/tag/5.0.1, et de personnaliser son emplacement dans votre architecture logicielle.

Ensuite, veillez à créer un répertoire où se situent les images que vous souhaitez recaler : les images en entrées. Une fois le chemin vers ce répertoire connu, il sera plus simple de « naviguer » vers ces images.

Également, pour que les images soient recalées correctement, il est indispensable de préciser certains paramètres dans un fichier propre. Différents paramétrages sont possibles, pour choisir le plus adapté à la situation, il est possible de faire son choix parmi les différents modèles proposés à la page suivante : https://elastix.lumc.nl/modelzoo/.

La Figure 5 illustre comment doit se présenter l'architecture du dossier en entrée du logiciel.

```
├── Data
│   ├── image1.nii
│   ├── image2.nii
│   ├── image3.nii        → contient les images d'entrées
│   ├── image4.nii
│   │
└── └── paramFile.txt     → contient tous les paramètres que l'on veut mettre en place
```

Figure 11 – Architecture requise pour faire fonctionner correctement Elastix

Une fois toutes les étapes ci-dessus réalisées, il est maintenant possible de recaler les images. Pour ce faire, il faut ouvrir un terminal de commande et se placer dans le répertoire où est téléchargé *Elastix*.

Par exemple : *D:\Programs\elastix-5.0.1-win64\elastix-5.0.1-win64*

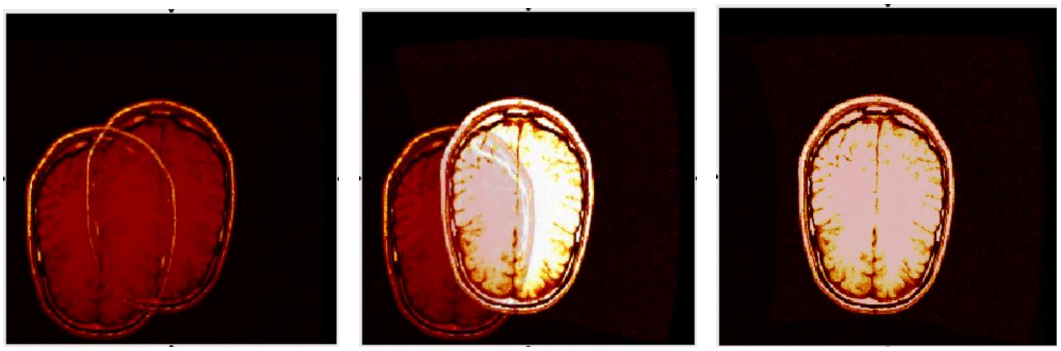
Ensuite, pour lancer le processus de recalage, il faut exécuter la commande suivante :

```
elastix -f <path/of/fixe/image> -m <path/of/mobile/image> -out <path/of/result/image> -p
<path/of/parameters/file>
```

Par exemple : *elastix -f C:\Users\Utilisateur\Documents\Donnees\image1.nii -m C:\Users\Utilisateur\Documents\Donnees\image2.nii -out C:\Users\Utilisateur\Documents\Donnees -p C:\Users\Utilisateur\Documents\Donnees\paramFile.txt*

Un processus va se lancer et durera quelques secondes. A la suite de cela, de nombreux fichiers seront créés dans le chemin renseigné en -out. Pour visualiser le résultat, plusieurs méthodes sont possibles :

- Avec le logiciel de visualisation d'image : *AMIDE*. Il est possible d'ouvrir l'image recalée et de constater l'efficacité de la méthode visuellement. La figure 12 permet de voir cette transformation et le déplacement de l'image mobile.



(a) (b) (c)

Figure 12 – Visualisation du recalage entre 2 images médicales : (a) Images initiales : Fixe au milieu et mobile à gauche, (b) Transformation de l'image mobile : position initiale de la mobile à gauche en rouge et position finale de la mobile en jaune à droite, (c) Superposition des deux images : fixe en rouge et mobile en jaune

- En analysant les fichiers créés. Il est possible de voir la métrique, le temps et le gradient pour chacune des itérations réalisées, comme le montre la figure 13.

1:ItNr	2:Metric	3a:Time	3b:StepSize	4:||Gradient||	Time[ms]
0	-1.039013	0.000000	553.224398	0.030307	386.1
1	-0.841156	0.000000	553.224398	0.022268	4.4
2	-0.803011	1.000000	528.077834	0.018023	7.1
3	-0.844375	2.000000	505.117929	0.023436	5.8
4	-0.826630	3.000000	484.071348	0.021708	7.0
5	-0.820714	4.000000	464.708494	0.020417	6.0
6	-0.853592	5.000000	446.835091	0.023165	6.0
7	-0.857496	6.000000	430.285643	0.017964	6.0
8	-0.845499	5.612893	436.544517	0.019295	5.6
9	-0.869696	6.612893	420.735072	0.025273	7.2
10	-0.848628	7.612893	406.030684	0.019289	5.6
11	-0.816216	8.612893	392.319403	0.020513	7.1
12	-0.856073	9.612893	379.503907	0.017018	5.6
13	-0.910562	10.612893	367.499188	0.021337	6.5
14	-0.913758	11.612893	356.230663	0.023910	5.9
15	-0.903436	12.612893	345.632627	0.018583	5.3
16	-0.894671	13.612893	335.646965	0.023564	7.3
17	-0.909885	14.612893	326.222893	0.025382	5.5
18	-0.894060	15.612893	317.312060	0.019667	7.1
19	-0.911837	16.612893	308.875802	0.029522	5.5
20	-0.907848	17.612893	300.876510	0.024452	5.9
21	-0.910088	18.612893	293.281091	0.020139	5.5
22	-0.950071	19.612893	286.059712	0.026302	5.5
23	-0.901764	19.225838	288.812192	0.025702	5.7
24	-0.875943	20.225838	281.806581	0.030394	5.4
25	-0.903354	21.225838	275.132786	0.023673	7.1
26	-0.899875	22.225838	268.767778	0.026430	5.5
27	-0.919815	23.225838	262.690612	0.024687	7.0
28	-0.930622	24.225838	256.882193	0.020415	5.6

Figure 13 - Résultats quantitatifs issus du recalage par Elastix

Ainsi, en analysant ces deux aspects (visuel et quantitatif), il est possible de tirer des conclusions sur l'efficacité du logiciel *Elastix* sur le recalage des images médicales. Avec l'exemple qui a été présenté ci-dessus, les résultats visuels semblent assez précis et la métrique s'approche de zéro, ce qui traduit une dissimilarité faible.

Pour conclure, *Elastix* est un logiciel assez puissant et précis dans le recalage des images. Toutefois, son utilisation est assez complexe et absolument pas intuitive. Un utilisateur possédant peu voire pas de connaissance en informatique aura beaucoup de mal à évoluer dans l'interface en ligne de commande.

2°) Utilisation des logiciels « SimpleElastix » et « Nifty Reg »

Il n'a pas été possible de télécharger d'autres logiciels tels que *SimpleElastix* ou *Nifty Reg.* En effet, leur téléchargement n'a malheureusement pas fonctionné et par manque de temps, il n'a pas été possible de tester leur fonctionnement.

Conclusion

Ainsi, l'algorithme de recalage codé dans la partie 1 permet de recaler différentes images qui ne nécessitent que des translations, et également de recaler des images ayant des modalités différentes. Ce dernier respecte l'algorithme proposé dans le sujet, et permet de se rendre compte du potentiel de ce genre d'algorithme et de leur utilité pour le personnel de santé, notamment en superposant plusieurs vecteurs d'informations.

Dans l'optique d'améliorer le programme qui a été élaboré dans la partie 1, il serait intéressant de se pencher sur la mesure d'une autre métrique de dissimilarité : l'information mutuelle. Avec celle-ci, il sera envisageable de recaler deux images multimodales, c'est-à-dire issues d'outils d'acquisitions différents. Par ailleurs, une ébauche d'amélioration a été fournie dans le code de l'application. En effet, la mesure de l'information mutuelle est effectuée et une valeur numérique est donnée en sortie à l'utilisateur. A l'avenir, il serait envisageable d'effectuer le recalage des images avec cette mesure spécifique.

Enfin, un test sur un logiciel déjà existant et utilisé de nos jours a été effectué : le logiciel *Elastix*. Celui-ci se révèle extrêmement efficace mais manque cruellement d'une interface homme-machine, ce qui rend son utilisation complexe voire impossible pour un médecin en situation routinière.

Bibliographie

[1] *'Information Mutuelle'*, Wikipédia [en ligne]. Disponible : https://fr.wikipedia.org/wiki/Information_mutuelle

[2] Witold Kosiński, Paweł Michalak et Piotr Gut (Mai 2012), *'Robust Image Registration Based on Mutual Information Measure'* [en ligne]. Disponible: https://www.scirp.org/pdf/JSIP20120200005_70217955.pdf

[3] Frederik Maes, André Collignon, Dirk Vandermeulen, Guy Marchal et Paul Suetens (Avril 1997), *'Multimodality Image Registration by Maximization of Mutual Information'* [en ligne]. Disponible : https://cs.uwaterloo.ca/~jhoey/teaching/cs793/papers/MaesTMI97.pdf

[4] Mathieu Rubeaux. *'Approximation de l'Information Mutuelle basée sur le développement d'Edgeworth : application au recalage d'images médicales.'* Traitement du signal et de l'image [eess.SP]. Université Rennes 1, 2011. Français. fftel-00632128f [en ligne]. Disponible : https://tel.archives-ouvertes.fr/tel-00632128/document

[5] Jérémy Cohen, Julien Huillery, 'Estimation de l'information mutuelle' PAR n132, Ecole Centrale Lyon, [en ligne]. Disponible : http://www.gipsa-lab.fr/~jeremy.cohen/docs/Rapport_PAR.pdf

[6] *'How to calculate Mutual information of 2d images in python'*, Stack Overflow [en ligne]. Disponible : https://stackoverflow.com/questions/60738217/how-to-calculate-mutual-information-of-2d-images-in-python

Annexes

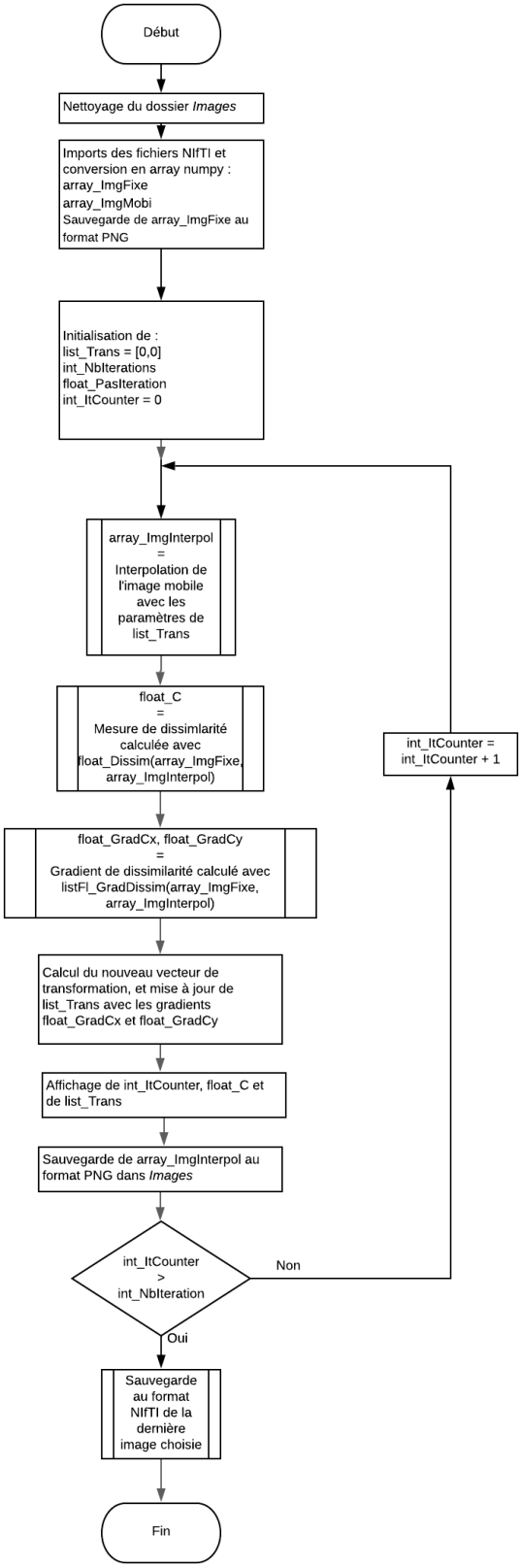
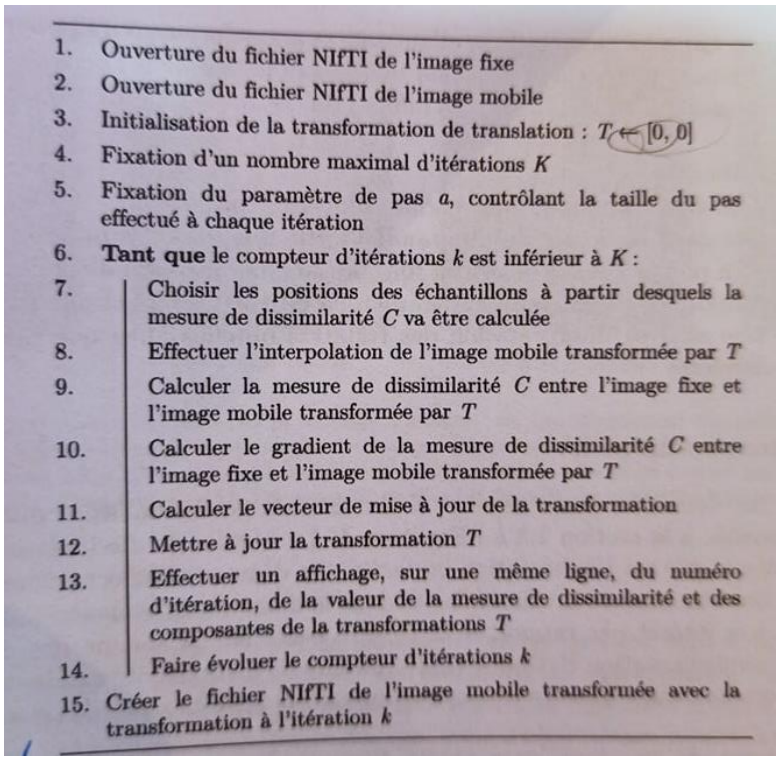


Figure 14 - Diagramme fonctionnel du programme Recalage.py



1. Ouverture du fichier NIfTI de l'image fixe
2. Ouverture du fichier NIfTI de l'image mobile
3. Initialisation de la transformation de translation : $T \leftarrow [0, 0]$
4. Fixation d'un nombre maximal d'itérations K
5. Fixation du paramètre de pas a, contrôlant la taille du pas effectué à chaque itération
6. **Tant que** le compteur d'itérations k est inférieur à K :
7. Choisir les positions des échantillons à partir desquels la mesure de dissimilarité C va être calculée
8. Effectuer l'interpolation de l'image mobile transformée par T
9. Calculer la mesure de dissimilarité C entre l'image fixe et l'image mobile transformée par T
10. Calculer le gradient de la mesure de dissimilarité C entre l'image fixe et l'image mobile transformée par T
11. Calculer le vecteur de mise à jour de la transformation
12. Mettre à jour la transformation T
13. Effectuer un affichage, sur une même ligne, du numéro d'itération, de la valeur de la mesure de dissimilarité et des composantes de la transformations T
14. Faire évoluer le compteur d'itérations k
15. Créer le fichier NIfTI de l'image mobile transformée avec la transformation à l'itération k

Figure 15 - Algorithme proposé pour le recalage d'images