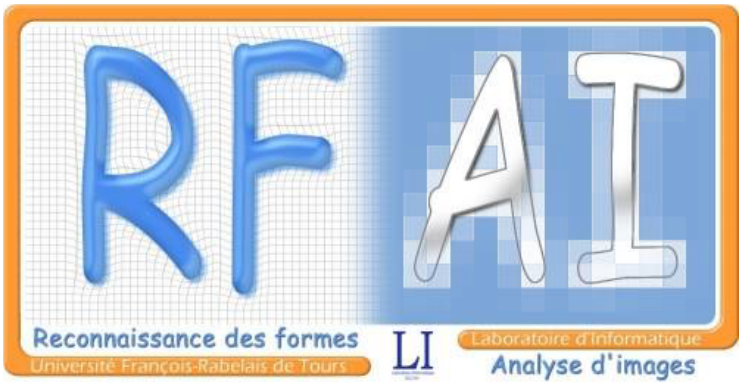
Ecole Polytechnique de l'Université François Rabelais de Tours
Département Informatique
64 avenue Jean Portalis
37200 Tours, France
Tél. +33 (0)2 47 36 14 14
polytech.univ-tours.fr

Projet Recherche & Développement

2016-2017

Plateforme pour word spotting multi-scripts

Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés



Entreprise
Laboratoire informatique

Tuteur entreprise

Étudiant
Quentin Combemorel (DI5)

Tuteur académique
Nicolas Ragot

1er avril 2017

Liste des intervenants

Entreprise

Laboratoire informatique
64 Avenue Jean Portalis, 37200 Tours
polytech.univ-tours.fr

Nom	Email	Qualité
Quentin Combemorel	quentin.combemorel@etu.univ-tours.fr	Étudiant DI5
Nicolas Ragot	nicolas.ragot@etu.univ-tours.fr	Tuteur académique, Département Informatique
		Tuteur entreprise

Avertissement

Ce document a été rédigé par Quentin Combemorel susnommé l'auteur.

L'entreprise Laboratoire informatique est représentée par susnommé le tuteur entreprise.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Nicolas Ragot susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnait assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnait qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.

Pour citer ce document

Quentin Combemorel, *Plateforme pour word spotting multi-scripts: Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2016-2017.

```
@mastersthesis{
    author={Combemorel, Quentin},
    title={Plateforme pour word spotting multi-scripts: Évaluation et amélioration d'un
       outil de recherche de mots "image" dans des carnets numérisés},
    type={Projet Recherche \& Développement},
    school={Ecole Polytechnique de l'Université François Rabelais de Tours},
    address={Tours, France},
    year={2016-2017}
}
```

Document mis en forme sous LaTeX
grâce à la classe de document `polytech.cls` V2.1.4 (2016/09/07)

Table des matières

Table des figures

5 Mise en œuvre

A Description des interfaces externe du logiciel

1 Introduction

Ce rapport concerne le Projet de Recherche et Développement « Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés ». Ce sujet est une reprise d'ancien PRD réalisés successivement par Zheng Zhan [8], Dawei Shen, Loreen Lambin [3] et Yamin Zaidou [7]. Un autre acteur est également à associer à ce projet, Bastien Meunier, qui, lors de son PFE avait développé un outil de wordspotting qui a été intégré à l'application par Laureen et Yamin. Ce projet sera encadré par Monsieur Ragot (également dans le rôle de la MOA), enseignant chercheur au laboratoire informatique de Polytech Tours, représentant des utilisateurs potentiels de l'application. Pour ma part, je prendrais le rôle de la MOE accompagné de Monsieur Ragot. Enfin, une équipe de deux étudiants, Marie Esteve et Quentin Coursimault, travailleront également avec nous sur ce projet dans le cadre de leur projet génie logiciel de quatrième années.

1 Contexte de la réalisation

De nos jours nous cherchons de plus en plus à conserver les documents qu'ils soient anciens ou non, cela passe donc par des numérisations de documents. Le problème est qu'une fois numérisé en tant qu'image, les outils de visualisation standard ne permettent pas la recherche de mots ou de caractères sur celle-ci. C'est pourquoi ce sujet de Projet de Recherche et de Développement a été ouvert il y a quelques années. Aujourd'hui une application est existante et permet la recherche de mots ou de caractères sur une image mais également l'affichage de résultats. Deux algorithmes ont été développés et sont fonctionnels (iWordSpotting et Renom). L'application a été modifiée l'année dernière par Yamin Zaidou dans le but de la rendre modulable. Dorénavant l'application permet d'ajouter de nouvelles bases d'images mais également des algorithmes de recherche que l'on souhaite utiliser. Toutefois il y a quelques régressions à noter entre le PRD de Laureen et celui de Yamin. En effet Yamin ayant repensé et refait l'entièreté du design de l'application, il n'a pas eu le temps de tout remettre en état. Voici un bref différentiel entre les deux versions :

1. L'outil renom non présent dans l'application
2. La suppression des bases d'image existante dans l'application est non fonctionnelle.
3. Un seul type de recherche est disponible (par mot) comparé à trois types de recherche dans l'application de Laureen (mot, caractère et dico)

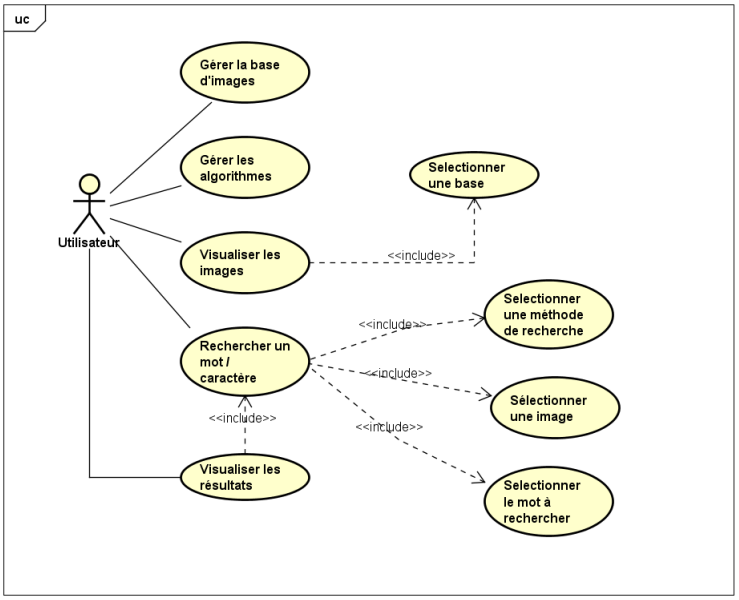


Figure 1 – *Diagramme de use case*

2 Objectifs

Le but général du projet, est l'amélioration et l'enrichissement de la plateforme existante. Plusieurs points sont envisagés afin de réaliser ces objectifs.

1. Améliorations fonctionnelles légères, cela va passer essentiellement par des corrections de bugs. En effet, quelques fonctionnalités n'ont pas eu le temps d'être mise en place l'année passée et son encore non fonctionnelles, comme l'affichage de résultats après l'exécution de l'algorithme de segmentation. Parallèlement à mon projet, des étudiants de 4ème année travailleront sur la plateforme dans le cadre de leur projet de génie logicielle. Je vais donc devoir intégrer leur changement à la plateforme existante.
2. Mise en place d'un module permettant de tester la robustesse d'une nouvelle méthode de segmentation et si possible d'en extraire des caractéristique valide. Cela implique de reprendre celles qui existent pour les généraliser aux images de lignes dont le contenu est un polygone.
3. En fonction du temps améliorer l'extraction de caractéristiques ou voir encore la méthode de spotting.

3 Hypothèses

Si au cours de la conception, je me rends compte que la difficulté est trop grande, ou que je manquerais de temps pour mettre en place l'amélioration des performances des algorithmes, je me concentrerais sur l'interface graphique. En effet, dans ce cas mon nouvel objectif sera d'améliorer l'application le plus possible en corrigeant les petits bugs que j'ai pu reporter, mais également en apportant des modifications de l'existant afin d'améliorer la compréhension du système pour n'importe quel utilisateur.

4 Base méthodologie

Pour mieux gérer ce projet, nous avons décidé avec monsieur Ragot de faire deux types de gestion de projet différentes en fonction de la période du projet. Dans la première partie de ce projet, qui est essentiellement de la recherche, la méthodologie adoptée sera des cycles itératifs qui permettront de structurer les recherches successivement les unes après les autres. Enfin, lorsque je commencerais la partie développement, la méthode de gestion de projet sera une méthode Scrum qui sera constituée de plusieurs sprints en fonction des lots qui seront définis. L'ensemble sera également synchronisé sur le Redmine de l'école, à l'aide de l'outil SVN.

2 Description générale

1 Environnement du projet

Le projet existant est composé de 5 projets distincts les uns des autres. Le but étant d'externaliser chaque tâche, c'est le fruit du travail des anciens PRD. Le premier projet se nomme « Binarization », c'est le projet qui permet de binariser les images cette partie a été développée par Yamin. Cette étape intervient en pré-traitement de la segmentation. Cette dernière est encore un autre projet également mise en place par Yamin, son rôle est d'extraire de l'image originale chaque ligne ou caractère. Ensuite intervient l'extraction, qui, comme ses deux prédécesseurs, est un autre projet à part mise en place par Yamin. Son rôle est d'extraire certaines caractéristiques d'une image suite à la segmentation. Ces caractéristiques vont permettent par la suite de comparer deux images et voir si elles sont plus ou moins proches. Le projet concernant l'outil de iwordspotting développé par Bastien Meunier est une méthode de recherche de mots sur image. Ce projet est complètement indépendant des deux autre, car c'est une autre méthode de recherche, même si il utilise le projet "Binarization" en pré-traitement. Toutefois il est important de noter que l'extraction des caractéristiques doit être dans un des formats utilisable par le projet iwordspotting. Enfin, le dernier projet est celui de la plateforme. Ce dernier existe depuis le PFE de Zhang Zheng en 2012, il a été repris ensuite par Laureen Lambin [3] et enfin par Yamin Zaidou. Tous ses projets présentés ci-dessus, sont des projets développés à l'aide de l'IDE VisualStudio et le langage utilisé est le C++ et le C#.

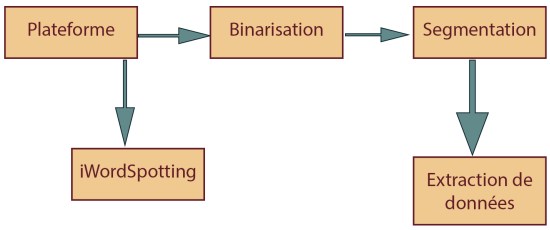


Figure 1 – *Architecture des différents projets*

2 Caractéristique des utilisateurs

Cette application est destinée à tous types d'utilisateurs, c'est dans cette optique que l'interface a été construite. En effet elle doit être suffisamment intuitive pour n'importe quelle personne. Aucun pré-requis en informatique ne doit être nécessaire pour son utilisation. Cependant on peut tout de même distinguer deux types d'utilisateurs.

1. Le simple utilisateur qui va simplement consulter la plateforme, visualiser les résultats.
2. Le manager, qui lui va plutôt gérer l'importation des bases d'images, des choix d'algorithmes à utiliser. De manière générale toute la partie un peu plus technique.

3 Nouvelles fonctionnalités du système

L'application existante permet déjà de nombreuses choses, elles sont rappelées dans le schéma ci-dessous. L'utilisateur a donc la possibilité de gérer ses bases d'images (ajout et suppression de bases). Il est également important de noter que le dossier (la base d'image) devra suivre une hiérarchie bien spécifique pour qu'il puisse être pris en compte par l'application. Il est également possible d'importer de nouveaux algorithmes de recherche ou de supprimer des existants dans l'application. Enfin la partie la plus importante est la partie recherche, où il faudra au préalable choisir une image d'une base importée dans l'application, sélectionner une zone correspondant à un mot ou un caractère que l'on souhaite rechercher. Et enfin sélectionner une méthode de recherche (importée au préalable dans l'application). Une fois la recherche effectuée, l'application affichera les résultats obtenus directement dans l'interface. L'objectif général va être de conserver l'existant et de l'améliorer sur quelques points.

1. Améliorer l'affichage des résultats du traitement de l'algorithme de wordSpotting, car actuellement les résultats affichés ne sont pas tout à fait cohérents.
2. Finaliser l'intégration de l'outil d'iwordSpotting de Bastien. En effet l'outil est disponible dans l'interface actuelle, mais n'affiche pas correctement les résultats obtenus.

De manière générale, cette première partie se rapproche à une maintenance progressive de l'application. Enfin je proposerais également de nouvelles méthodes pour améliorer les performances de la recherche en jouant sur deux aspects.

3. Amélioration de la segmentation.
4. Amélioration des caractéristiques.

En effet, l'un des objectifs majeurs de ce PRD est de pouvoir faire des recherches sur des documents manuscrits numérisés. Enfin voici les nouvelles fonctionnalités envisagées avec un niveau d'importance moins élevés.

5. Sauvegarde des recherches
6. Réorganisation des résultats de recherches
7. Réintégration de l'outil de renom
8. Mise en place de la méthode de recherche par dico / chaine

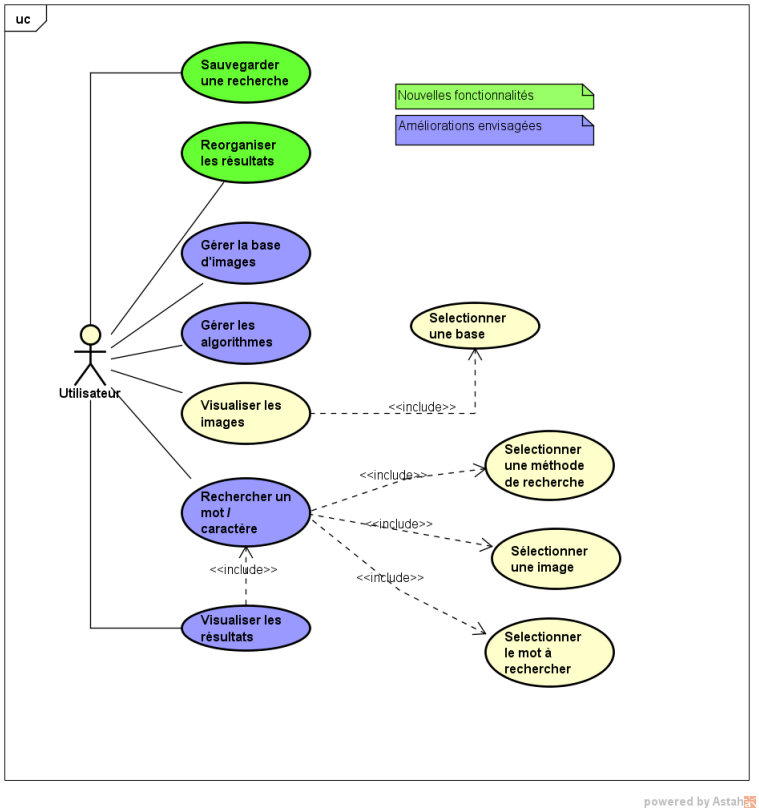


Figure 2 – *Nouveau diagramme de cas d'utilisation*

4 Structure générale du système

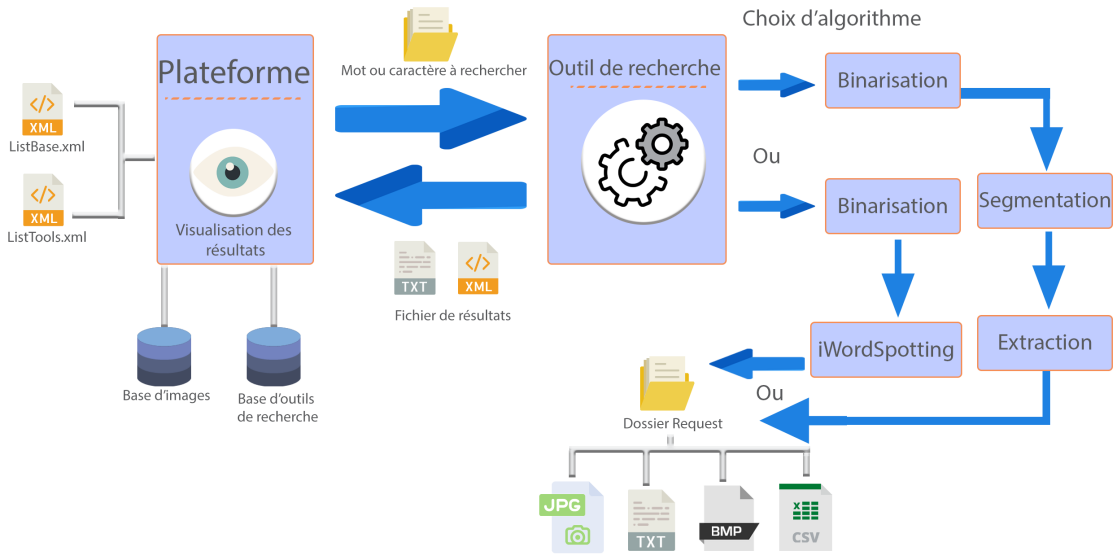


Figure 3 – *Structure générale du système*

La Figure 3illustre le fonctionnement du système général et l'interaction qu'il y a entre les différents acteurs. L'entité « outil de recherche » comprend les projets indépendants tel que l'outil développé par Bastien ou encore la segmentation et l'extraction de données développée par Yamin. Vous trouverez en Annexe H, le détail des classes que je vais modifier (en rouge), et

celle que je devrais développer (en vert) afin de répondre aux objectifs. Les classes entourées de vert sont les classes que Yamin n'a pas eu le temps d'implémenter lors de son PRD de l'année dernière. A cela je vais également devoir ajouter de nouvelles classes notamment pour :

1. Sauvegarder les recherches effectuées
2. Nouvelle méthode de segmentation
3. Nouvelle méthode d'extraction de données

3 État de l'art

Dans cette partie je vais vous présenter toutes les recherches que j'ai pu effectuer lors de la première moitié de ce PRD. Celles-ci ont été guidées par Monsieur Ragot. Le but de ces recherches est de pouvoir voir différentes techniques, essentiellement au niveau de la segmentation et de l'extraction de données, pour ensuite prendre une décision sur laquelle ou lesquels il serait pertinent de mettre en place, afin de répondre aux objectifs définis pour ce PRD.

1 Segmentation

La segmentation consiste à regrouper des pixels entre eux en fonction de critères précis. Dans notre cas, il s'agit de distinguer les mots du fond (arrière plan). Nous allons donc extraire des éléments d'une image donnée, tel que des lignes ou des mots afin de pouvoir les comparer. Celle-ci peut être faite verticalement ou horizontalement. Dans cette partie je vais donc vous présenter toutes les techniques que j'ai pu analyser lors de mes recherches. La méthode mise en place par Yamin l'année dernière était une combinaison entre la méthode de projection profile et la méthode RLSA Horizontal afin d'effectuer une segmentation de ligne [7]. Après plusieurs, tests et notamment sur des images assez lourdes, on remarque que le système est assez lent. C'est pourquoi une de mes missions cette année est de l'améliorer.

1.1 DIVAServices

C'est une application développée par le groupe « Diva » à l'université de Fribourg [3]. Cette application est un webService qui permet d'utiliser des algorithmes d'analyse d'images. 16 méthodes y sont accessibles. Pour ma part je me suis donc concentré sur la partie segmentation. Diva service propose trois méthodes de segmentation différentes :

1. Ocropy segmentation : décompose en plusieurs images les zones segmentées. Après avoir binarisé l'image, sa première action va être d'estimer l'échelle du texte en trouvant les composants connectés dans l'image binarisée (principalement des lettres). Ensuite il va essayer de trouver les différentes lignes de texte. Pour ce faire il va :
   (a) Supprimer les composants de l'image qui ne correspondent pas à l'échelle estimée (les composants trop grands ou trop petits)
   (b) Il va détecter les bords inférieurs et supérieurs

(c) Les pixels présents entre les bords supérieurs et inférieurs précédemment définies, seront donc les lignes.

Selector	Value
textline0	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline1	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline10	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline11	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline12	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline13	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline14	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline15	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline16	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline17	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline18	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline19	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline2	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline20	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf
textline21	http://divaservices.unifr.ch/static/959d25d50cf1a58b83cc80647bf

Figure 1 – *Résultats de l'algorithme Ocropy*

2. Seam carving based text line segmentation : cette méthode de segmentation est basée sur la méthode Seam Carving. Cette algorithme va dans un premier temps faire une projection profile, puis calculer les "Seams" entre deux lignes consécutives. Seam signifie un chemin connecté de pixels de basse intensité dans une image. Dans ce cas les pixels de basses intensités correspondent à une zone de fond contrairement aux pixels de hautes intensités qui eux, correspondent à une zone de texte. Plusieurs paramètres sont à renseigner pour utiliser cette méthode :
   (a) Une zone de segmentation, que l'on défini manuellement
   (b) Le paramètre "Smooth" compris entre 0 et 1. Ce paramètre sera utile pour la projection profile
   (c) Le paramètre "Slices" compris entre 2 et 8. Cela va définir en combien de partie égale verticale l'image va être découpée.
   (d) Le paramètre "Sigma" compris entre 2 et 5. Correspond à la déviation standard du filtre Gaussien appliqué à l'image.
   Cette méthode permet de découper en ligne en prenant en compte la hauteur de chaque lettre. Comme peut le montrer la Figure 2.

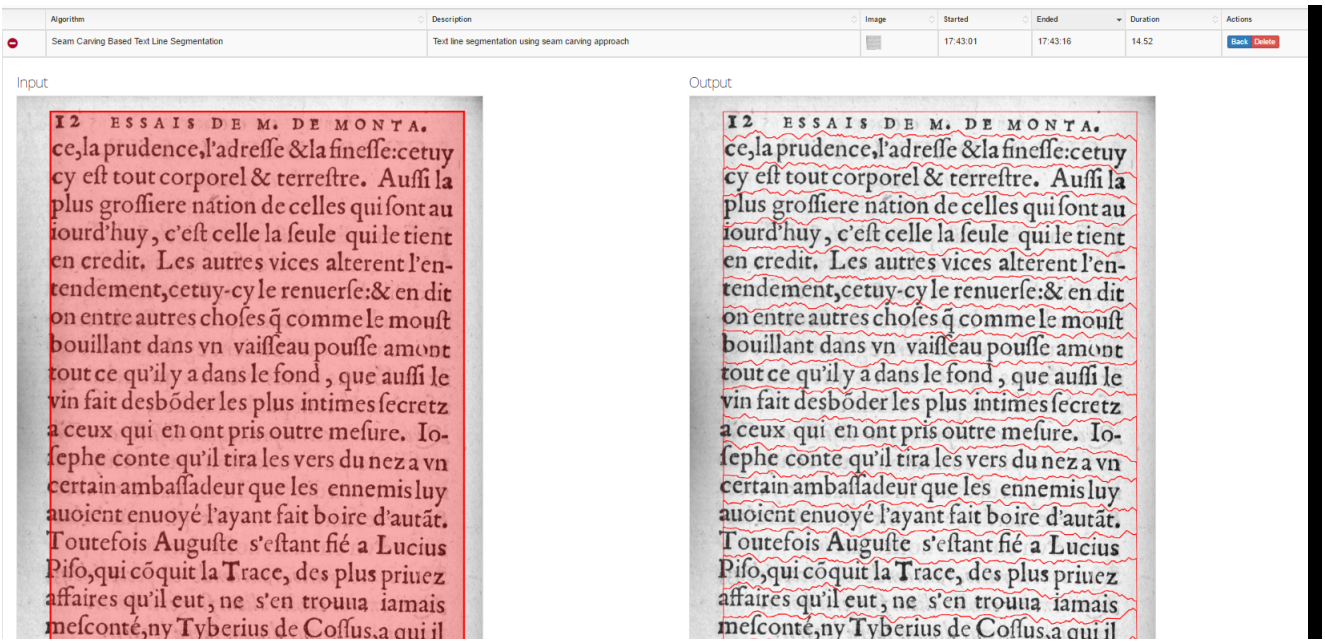


Figure 2 – *DivaService - Seam Carving Based Text Line Segmentation*

3. Histogram based text line segmentation : cette méthode extrait ligne par ligne, sans prendre en compte l'orientation des lignes, ce qui peut entraîner des pertes d'informations.

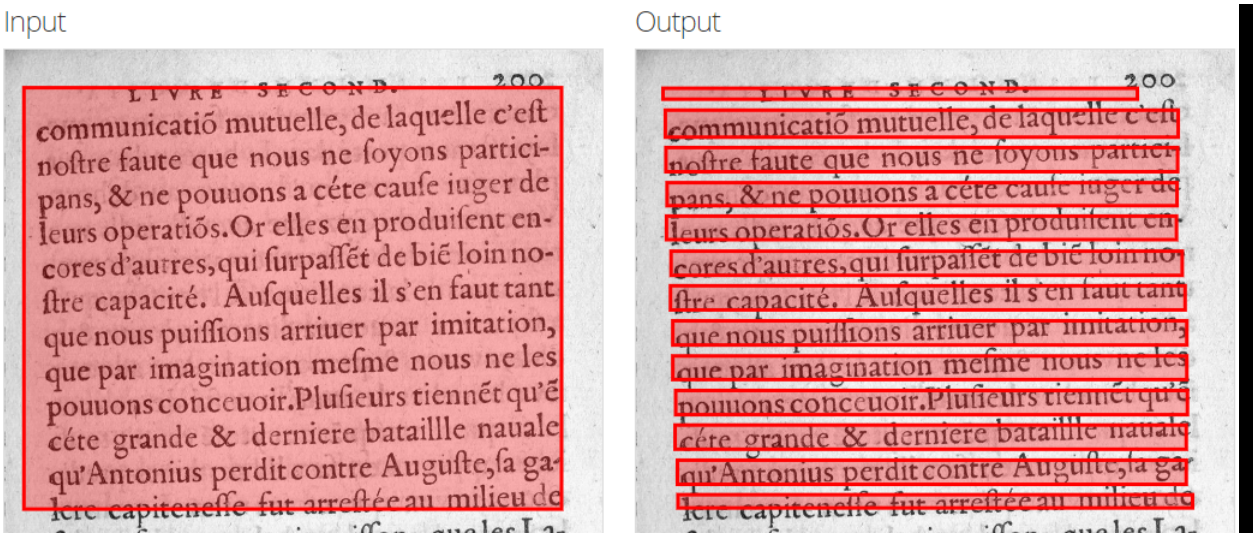


Figure 3 – *Histogram based text line extraction*

Toutes ces méthodes sont accessibles par des requêtes get et post. Les résultats retournés sont sous forme JSON. Toutefois je n'ai pas réussi à obtenir de résultats par des requêtes. Cela nous expose également à quelques contraintes fortes :

1. L'accès à internet est primordiale
2. Forte dépendance au service Diva, le jour où ils ne sont plus en ligne, notre application n'est plus capable de les utiliser et donc de fonctionner normalement.
3. Le fait d'utiliser ces services nécessite un upload d'images, ce qui peut être compromettant dans le cas où nos images (de texte manuscrite historique) sont confidentiels.

Il existe également une application web qui permet de manipuler facilement leurs méthodes, elle se nomme "DivaService Spotlight" [4]. C'est grâce à cette dernière que j'ai pu tester les différents algorithmes. En terme de performances, voici un comparatif des trois algorithmes testé ci-dessus :

Algorithm	Description	Image	Started	Ended	Duration
Histogram Based Text Line Segmentation	Simple Text Line Segmentation using Histograms		10:55:25	10:55:27	2.71
Seam Carving Based Text Line Segmentation	Text line segmentation using seam carving approach		10:24:10	10:24:22	12.10
Ocropy - Pageseg	ocropy page segmentation method		10:00:26	10:00:27	1.23

Figure 4 – *Temps d'éxécution des algorithmes de segmentation*

On peut facilement voir sur la Figure 4, que l'algorithme le plus performant (en terme de rapidité d'exécution) est "Ocropy"

1.2 Text extraction in document images : highlight on using corner points

Cette technique a beaucoup été étudiée dans la littérature des documents d'analyse d'image. C'est une approche basée sur la méthode "**corner point**". Celle-ci n'est pas très utilisée pour l'extraction de texte dans des documents image. D'un point de vue théorique, la méthode des corner points n'est pas très significative en ce qui concerne les images de texte. Cette méthode reste toutefois très performante, comme peut le montrer cette publication [6] sur laquelle je me base pour faire cette partie.

Ici, le but est simplement d'extraire des points clés détectés par la méthode FAST et analyser leur densité à l'intérieur des blocs de taille fixe sur l'image pour garder les plus denses comme régions de texte. Le principe de cette méthode peut se décomposer en 6 étapes :

1. Lissage de l'image grâce un filtre gaussien

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

   Dans cette formule x et y correspondent aux coordonnées du pixel pendant que σ représente la déviation standard de la distribution gaussienne.
2. Détection de "**Corner point**" grâce à la méthode Fast Corner Points [5] et [1]. Cette étape consiste à choisir un pixel en fonction de son intensité et de ses 16 voisins. Si les intensités d'au moins 12 pixels sur les 16 environnants sont au-dessus ou en-dessous d'un seuil spécifié, alors c'est un point clé. Le seuil choisi est 20% de l'intensité du point clé.

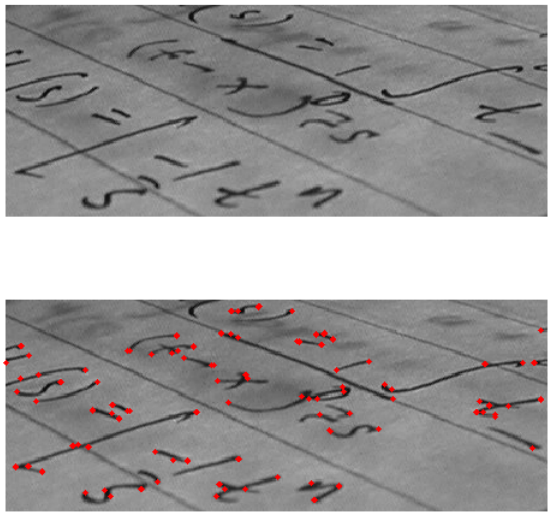


Figure 5 – *Résultats d'une corner détection. Source : [1]*

3. Diviser l'image en bloc 32x32 sans chevauchement, puis calcul du nombre de corner points dans chacun des blocs.
4. A partir du bloc qui possède le plus de corner points, on définit un seuil. Le seuil prendra la valeur de 0.2*Nmax. Ici Nmax est le nombre maximum de corner point présent dans un bloc.
5. Les blocs qui ont un nombre de "corner point" plus important que le seuil, appartiennent à des régions de texte. Et les blocs qui ont un nombre moins important appartiennent aux autres régions (image, arrière plan, bruit).
6. Après avoir détecté les blocs de texte, on regarde les liens entre les blocs pour construire les régions de textes.

L'avantage de cette méthode, est qu'elle a été conçue de façon à être moins sensible au bruit, aux variations de luminosité et à la résolution de l'image.

2 Extraction de caractéristique

L'extraction de caractéristique permet de comparer deux instances entre-elles grâce à un ensemble de caractéristiques. Ici ces instances sont des séries temporelles représentant les caractéristiques au cours du déplacement sur la ligne (de gauche à droite). Plus les caractéristiques sont proches plus les instances seront semblables. C'est pourquoi cette étape est cruciale pour un outil de wordspotting. Lors du PRD de l'année dernière, Yamin avait mis en place une méthode d'extraction de données à base de colonnes (DTW) [7]. Encore une fois nous avons estimé avec Monsieur Ragot qu'il était pertinent d'améliorer cette méthode afin de pouvoir appliquer l'extraction de données sur un document numérique manuscrit. Ici nous devons organiser les caractéristiques sous forme de série temporelle, le but de cette recherche va donc consister à organiser ces caractéristiques afin de gagner du temps dans l'exécution du DTW. Dans cette partie je vais vous présenter les différentes méthodes que j'ai pu analyser.

2.1 DTW : Dynamic Time Warping

Les deux recherches qui vont suivre sont des variantes de la méthode Dynamic Time Wraping [4]. C'est pourquoi il est important de l'expliquer. Cette méthode permet de mesurer la similarité entre deux suites dynamiques qui varient au cours du temps.
Voici un exemple, prenons deux séries temporelles : Figure 6

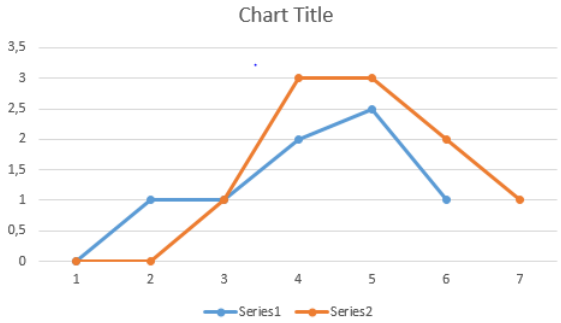


Figure 6 – *Graphique de nos deux séries temporelles*

Il faut maintenant construire la matrice des distances entres ces deux courbes, afin de pouvoir déterminer le meilleur chemin, celui qui sera le moins coûteux à l'aide de cette formule :

$$G(i,j) = D(i,j) + min \begin{cases} G(i,j-1) \\ G(i-1,j-1) \\ G(i-1,j) \end{cases} \tag{2}$$

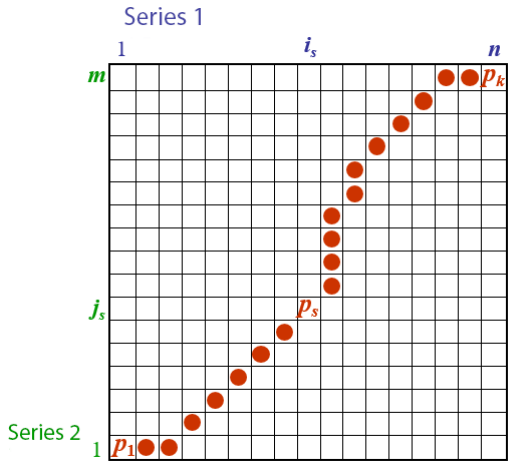


Figure 7 – *Chemin le plus optimal*

Sur la Figure 7, on distingue bien en rouge le chemin optimal obtenu suite à l'application de la formule. Le principal problème de cette technique est qu'elle engendre une forte complexité en terme de calcul $O(n)$. C'est pourquoi il existe plusieurs variantes qui ont pour but d'améliorer ce critère. En modifiant la méthode on peut prétendre à de meilleurs résultats mais également augmenter la rapidité de cet algorithme.

2.1.1 Regroupement de données par la moyenne

Une stratégie existe pour calculer rapidement la distance DTW. Cette dernière consiste à réaliser l'alignement non pas sur l'entièreté, mais plutôt sur une partie définie de nos deux séquences X et Y. Cela implique donc une réduction de la taille de nos deux séries temporelles utilisées. Cette méthode est donc basée sur la méthode Piecewise Dynamic Time Warping (PDTW).

Le principe de PDTW, est de réduire la taille originale du signal en gardant les représentations les plus significatives du signal entier grâce à la technique de sous échantillonnage. Nous allons passer d'une dimension p à une dimension réduite notée R correspondant à notre nouvelle série temporelle fraîchement transformée. $1 < R < p$ Pour pouvoir réduire notre dimension, il suffira de réduire nos données en R images de tailles égales.

$$Xi = \frac{R}{p} \sum_{j=\frac{R}{p}(i-1)+1}^{\frac{R}{p}i} xj \tag{3}$$

On calcul la valeur moyenne des données de la trame, ensuite un vecteur de ces valeurs, deviendra la représentation des données réduites (aussi appelées sous-échantillonnage).

Figure 8 – *Série temporelle*

Voici un exemple pour illustrer cette explication. Prenons une série temporelle tel que la Figure 8 Notre série temporelle sera donc représentée par ce vecteur : $X = (-1,-2,-1,0,2,1,1,0)$.

Ici notre dimension initiale est $p = 8$.

Prenons $R = 2$, notre nouvelle dimension. Notre vecteur associé sera donc :
$X = (moy(-1,-2,-1,0), moy(2,1,1,0))$. Une fois les moyennes appliquées, nous obtenons notre nouveau vecteur : $X = (-1,1)$.
Dans cet exemple nous avons transformé notre vecteur de dimension 8 en vecteur de dimension 2. Notre série temporelle a donc été séparée en deux images, enfin nous avons calculé la moyenne de chaque images. Ce nouveau vecteur représente désormais notre donnée réduite. Enfin avec cette méthode on peut calculer un taux de compression $c = \frac{P}{R}$. De manière générale on ne peut pas dégager de "meilleur" ratio, en effet chaque ratio doit être adapté aux données que l'on possède. La particularité de cette méthode, est qu'il va falloir faire un compromis entre rester fidèle à la donnée initiale et sauver de la mémoire grâce à un fort taux de compression. Plusieurs pistes sont envisageables afin de définir le taux de compression C :

1. En fonction de la résolution d'image
2. L'adapter localement en fonction de l'image. Par exemple si on estime qu'une zone est plus importante qu'une autre on va baisser le taux de compression afin d'avoir des valeurs

beaucoup plus proches de la réalité. A l'inverse il est possible de négliger certaine zone, jugées moins importantes.

2.1.2 Dérivé de données

Dans cette partie nous allons plutôt chercher un moyen d'augmenter la performance de l'algorithme DTW. Pour ce faire nous allons utiliser une méthode qui consiste à utiliser les dérivés. Cette méthode est basée sur l'algorithme DDTW : Derivative Dynamic Time Warping (DDTW).

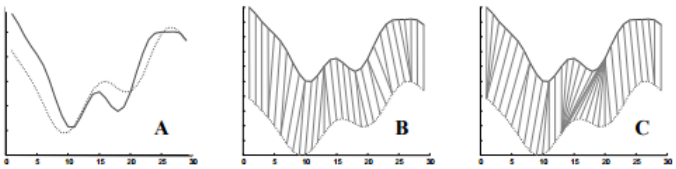


Figure 9 – *Exemple d'anomalies Sources : [2]*

De manière générale, l'utilisation de DTW classique peut conduire à quelques anomalies inattendues comme le montre le figure c de la Figure 9. Cette illustration nous montre bien la différence entre un alignement "naturel" entre les deux signaux (B) et ce que l'on peut obtenir après l'exécution du DTW (C). Pour éviter ce problème, l'algorithme DDTW ne va pas s'appuyer sur les distances euclidiennes comme peu le faire DTW, mais plutôt sur le carré de la différence des dérivés de i et j.

$$D_x(i) = \frac{(x_i - x_{i-1}) + (\frac{x_{i+1} - x_{i-1}}{2})}{2} \tag{4}$$

avec $2 \leq i \leq p-1$ et $1 \leq m \leq p-1$

$$D_x(j) = \frac{(x_j - x_{j-1}) + (\frac{x_{j+1} - x_{j-1}}{2})}{2} \tag{5}$$

avec $2 \leq j \leq q-1$ et $1 \leq n \leq q-1$

Avec les nouvelles distances obtenues, on va pouvoir remplir notre matrice des distances pour ensuite pouvoir trouver le chemin optimal. (Figure 7). Néanmoins la complexité reste toujours la même que celle de DTW à savoir O(n). Sur la Figure 10, on peut voir la différence de résultats

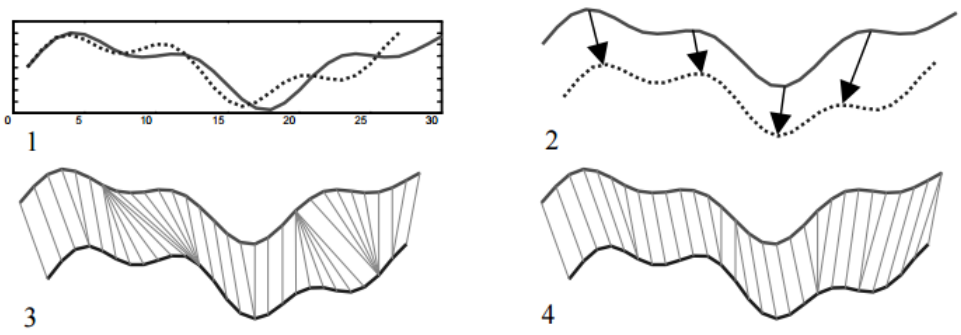


Figure 10 – *Comparatif de résultats entre DTW et DDTW Source : [2]*

sur deux signaux (1). L'image (2) nous montre l'alignement logique, en comparaison de l'image (3) qui nous montre ce que ferait l'algorithme DTW. Enfin sur l'image (4), ce que produirait l'algorithme DDTW.

2.2 HOG Feature

Le but de cette méthode, comme l'explique Yamin dans son rapport [7] basée sur [4] et [5] est de définir une fenêtre de taille fixe qui se déplacera horizontalement sur notre image afin de calculer l'histogramme du gradient orienté. Le rectangle précédemment défini ne pourra pas modifié suivant son échelle ou son orientation.

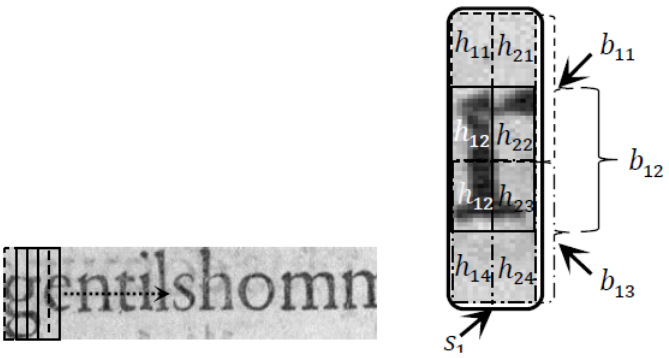


Figure 11 – *Exemple de la méthode HOG feature*

3 Mes Choix

Suite aux recherches effectuées, l'utilisation des services Diva est fortement envisageable, si toutefois ils permettent de gagner en performance comparé à l'existant. Deux options seront envisageables, à savoir impliquer une manœuvre manuelle sur leur application web afin d'avoir nos documents segmentés. Ou alors, si les sources sont disponibles, envisager de mettre en place une de leurs solutions au choix. Au vu des résultats obtenus, je pense que les deux algorithmes à prendre en compte sont :

1. Ocropy
2. Seam Carving Based Text Line Segmentation

Une autre approche intéressante serait de reprendre le travail de Yamin en y ajoutant une étape de lissage tel qu'un filtre gaussien qui permettrai d'épurer l'image en enlevant le bruit. Cette méthode seule permettrait d'améliorer les performances de l'algorithme en terme de précision notamment.

En ce qui concerne la partie extraction de données, la décision de mettre en place les techniques basées sur les méthodes PDTW et DDTW seraient presque évidente. En effet, cela permettrait dans un premier temps, de reprendre le travail de Yamin en y ajoutant une étape de pré-traitement "réorganisation de caractéristiques" afin d'avoir un algorithme bien plus performant derrière. De plus ses méthodes ont prouvé leur robustesse comme on peut le voir dans cette publication [4].

4 Analyse et conception

1 Amélioration des performances

Dans un premier temps je vais mettre en œuvre des séries de tests afin de pouvoir voir la vélocité du système actuel sur des documents manuscrits. Je vais également effectuer ces mêmes tests sur la plateforme mis en place par Diva Service [4]. Ces tests vont se focaliser sur la partie segmentation des documents. En fonction des résultats obtenus, je mettrais en place une solution :

1. Soit développer la solution de Diva Service en local si je trouve suffisamment de sources pour le faire.
2. Soit j'améliore la solution existante, en ajoutant par exemple l'application d'un filtre Gaussien qui permettrait de supprimer le plus de bruit possible sur l'image.

En ce qui concerne la partie extraction de données, mon but ici ne sera pas d'améliorer l'extraction mais plutôt modifier la structure du projet afin de la rendre plus extensible. En effet le but est de rendre facile l'ajout de nouveaux types d'extraction de caractéristiques.

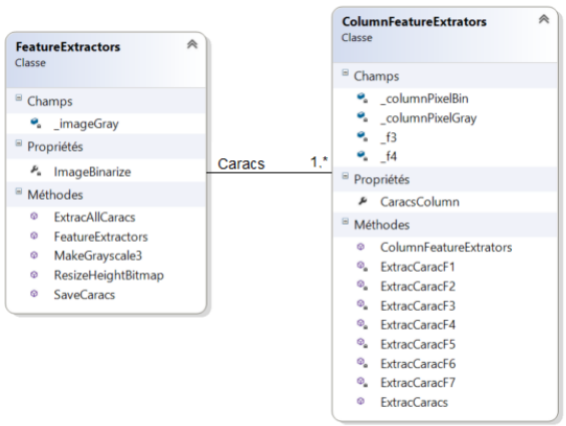


Figure 1 – *Ancien diagramme de classe du module d'extraction de caractéristiques*

2 Amélioration du fonctionnement global de l'application

Une autre partie sera dédiée à l'amélioration plateforme. La priorité de cette tâche est bien moindre que la partie précédente. Plusieurs points sont à relever :

1. Amélioration de la gestion de base d'images : permettre à l'utilisateur de pouvoir modifier la localisation d'une base d'images. En effet cette fonctionnalité est manquante. Actuellement il faut supprimer et réimporter une base d'images afin de pouvoir modifier sa localisation.
2. Afin de guider l'utilisateur dans son utilisation de l'application, il serait intéressant de griser certains boutons notamment dans les menus de gestion de base d'images et également gestion d'algorithme. En effet cela permettrai d'indiquer à l'utilisateur que l'action n'est pas disponible. Cette spécificité est déjà existante à certains endroits de l'application mais pas partout. L'idée ici est de la rendre effective dans toute l'application.
3. Amélioration des messages d'erreurs dans la plateforme. Le but ici est de guider l'utilisateur et non pas simplement d'afficher un message d'erreur.
4. Faire fonctionner l'affichage des résultats d'une recherche. Cette fonctionnalité existe, mais ne fonctionne pas correctement.

Beaucoup d'éléments cités dans cette partie seront mises en place par Quentin et Marie deux étudiants de 4ème année qui travaillent sur le projet dans le cadre de leur projet de génie logiciel. A moi d'intégrer leur production au projet final. A cette liste s'ajoutera d'autres améliorations mineures ou d'éventuels bugs que je n'ai pas encore découvert, ou encore l'intégration d'outils externes existants mais encore non intégrés dans l'application aujourd'hui.

5 Mise en œuvre

1 Campagnes de tests

Dans cette partie vous sera présenté les différents tests que j'ai pu effectuer afin de garantir la qualité et la validité des choses que j'ai mis en place.

1.1 Test des méthodes de segmentation DivaService

Cette partie s'inscrit dans la continuité de mes recherches. En effet, le but ici est de mettre en place des outils afin de pouvoir tester si les méthodes choisie parmi celles exposées dans mon état de l'art Chapitre 3, sont plus efficaces ou non que celles déjà en place. Pour se faire, j'ai donc testé les trois méthodes de segmentation suivantes :

1. Ocropy Segmentation
2. Seam Carving Segmentation
3. Histogram

Pour évaluer les tests, je me suis basé sur la qualité de la segmentation, la façon dont la méthode segmente les lignes, puis le temps d'exécution. Pour effectuer ces tests, j'ai utilisé la plateforme DivaService [4], où les différents outils sont déjà en place. Les images que j'ai utilisé sont les images d'archives que Monsieur Ragot m'a envoyé. Ce sont des documents manuscrits. Après avoir réalisé les tests, on a pu éliminer la première méthode qui ne parvenait pas à segmenter les lignes.

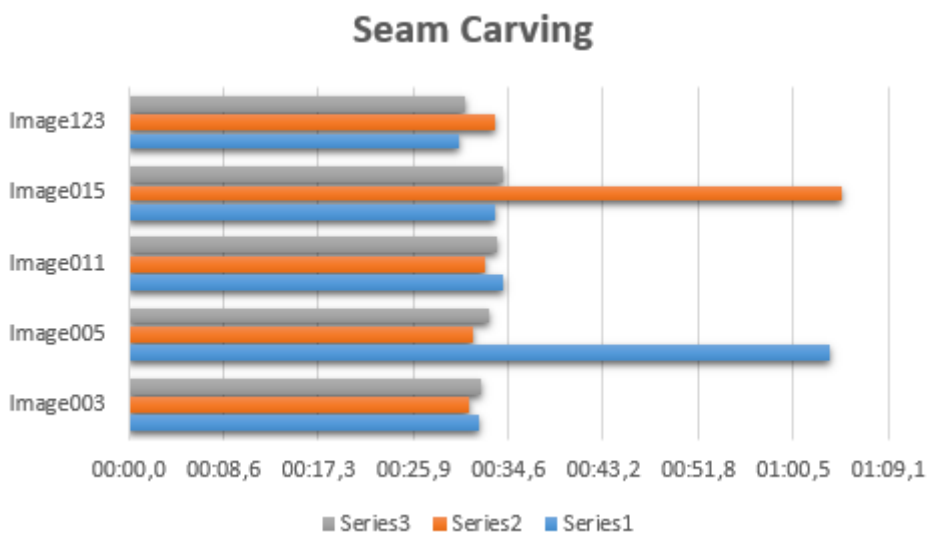


Figure 1 – *Résultats des tests sur la méthode "Seam Carving"*

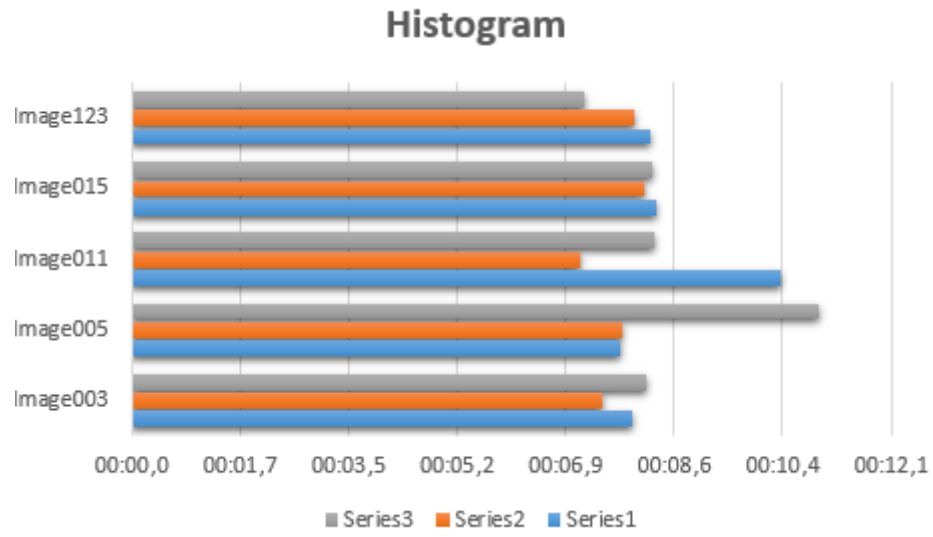


Figure 2 – *Résultats des tests de la méthode "Histogram"*

Comme vous pouvez le voir sur les graphiques Figure 1et Figure 2, la méthode par histogramme à un net avantage sur la rapidité d'exécution.

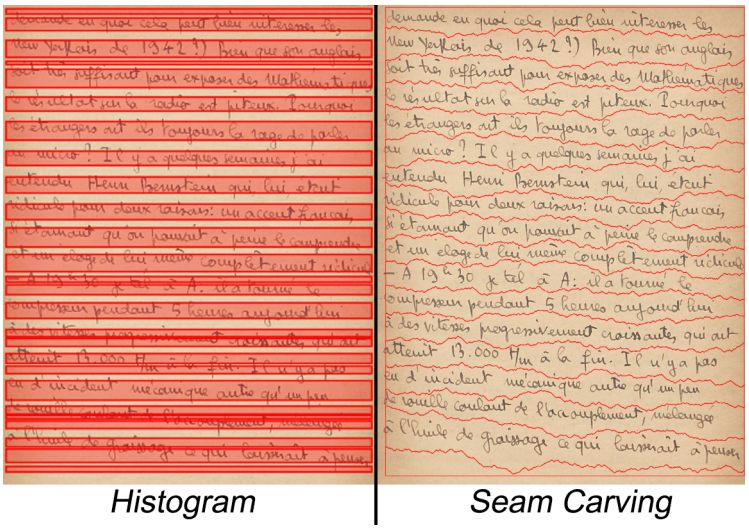


Figure 3 – *Résultats de la segmentation Histogram vs Seam Carving*

Or comme on peut le constater sur l'image Figure 3, la qualité de segmentation de la méthode seam carving est bien supérieure que la méthode "Histogram". C'est pourquoi en accord avec Monsieur Ragot, j'ai choisi d'utiliser la méthode "Seam Carving".

1.2 Test des caractéristiques

Une fois que la méthode a été choisie, et que le module de test de la méthode Seam Carving a été mis en place (décrit dans la partie suivante), il a fallu également tester si les caractéristiques qu'on extrait derrière restent justes et cohérentes. Pour faire ces tests, il a fallu utiliser directement la plateforme et le faire de façon visuelle. En effet pour effectuer ces tests, j'ai choisi certains mots et j'ai effectué des recherches via la plateforme afin de voir si globalement les résultats obtenus étaient cohérents. Pour aller plus loin, j'ai également effectué des tests entre les deux méthodes "Seam Carving" et celle de Yamin mise en place l'année dernière. Pour effectuer ces tests, j'ai encore fait 3 séries de tests, entre chaque série, je modifiais le seuil (modifiable dans le projet IwordSpotting). Plus celui-ci est petit, plus il permet d'affiner la recherche.

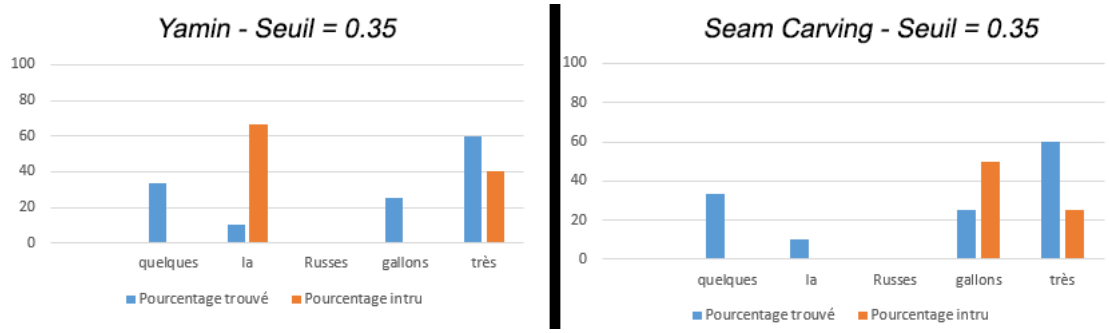


Figure 4 – *Résultats de recherche entre Yamin et la méthode SeamCarving*

Globalement, on peut dire en regardant la Figure 4que les résultats sont assez similaires. Cependant, ce qu'on peut dire, c'est qu'il faut développer les caractéristiques afin de pouvoir exploiter pleinement le fait que la méthode Seam Carving segmente par polygone et non pas par rectangle. C'est un point qui peut être mis en place dans une reprise éventuelle de ce PRD.

2 Module de test de la méthode Seam Carving

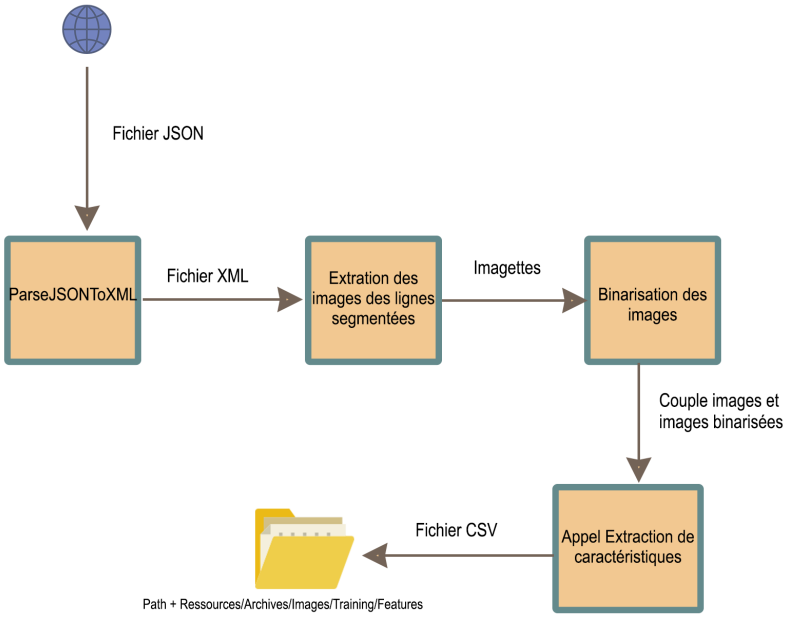


Figure 5 – *Module de test de la méthode Seam Carving*

Dans cette partie, le but est de tester si la méthode Seam Curving est pertinente à mettre en place. Pour se faire, il nous suffit de générer les fichiers CSV (une pour chaque ligne segmentée de chaque image). Chaque fichier CSV représente l'ensemble des caractéristiques extraites.

2.1 Récupération du fichier JSON

Dans un premier temps, pour tester la méthode Seam Curving, j'ai utilisé le site de Diva Service qui permet d'envoyer des requêtes sur des images comme expliqué dans mon état de l'art Chapitre 3. Ce site est totalement libre d'utilisation et ne requiert aucune création de compte. Pour se faire, j'ai utilisé l'outil **Insomnia**, ce dernier permet d'exécuter des requêtes GET et POST sur un serveur. Une fois nos images converties en base64, je peux exécuter une requête de type POST.

Dans celle-ci, il faut renseigner les mêmes paramètres énoncés dans l'état de l'art à savoir :

1. Les dimensions d'un rectangle qui permettent de définir la zone à segmenter. Dans notre cas, un rectangle aux dimensions de l'image : 1980 x 2580.
2. Le paramètre "Smooth" compris entre 0 et 1. Ce paramètre sera utile pour la projection profile.
3. Le paramètre "Slices" compris entre 2 et 8. Cela va définir en combien de parties égales verticales l'image va être découpée.
4. Le paramètre "Sigma" compris entre 2 et 5. Correspond à la déviation standard du filtre Gaussien appliqué à l'image.

En retour, nous obtenons un fichier JSON comprenant chaque ligne segmentée.

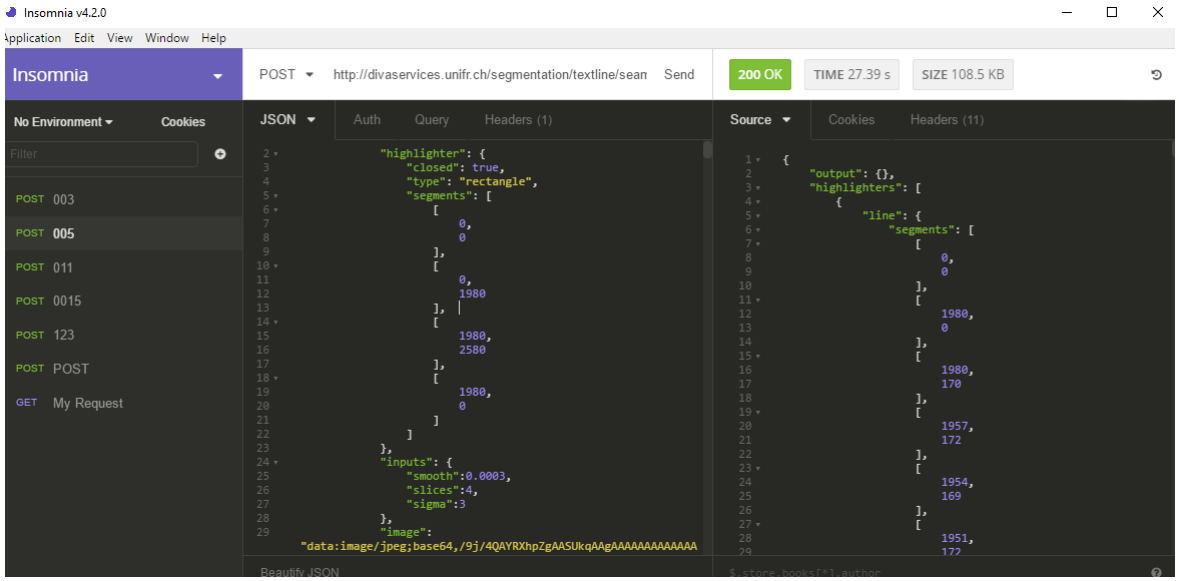


Figure 6 – *Interface d'Insomnia*

2.2 Module Test Seam Carving

Une fois les fichiers JSON récupérés, je peux appeler mon module qui me permettra de faire les actions suivantes :

1. Convertir le fichier JSON en fichier XML.
2. Lire dans le fichier XML puis extraire les polygones représentant les lignes à segmenter.
3. Appel de la binarisation afin de binariser toutes les images des lignes extraites.
4. Appel au module d'extraction de caractéristiques afin de générer nos fichiers CSV.

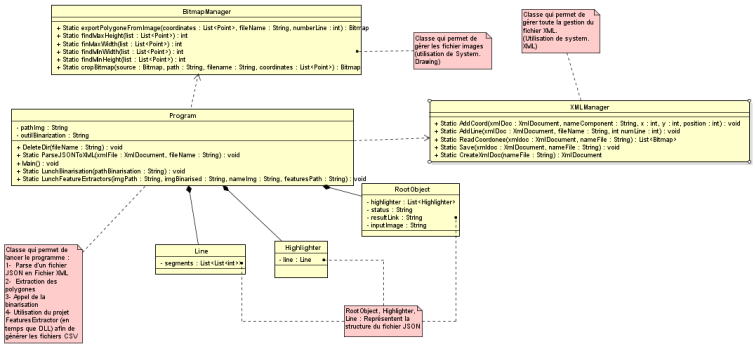


Figure 7 – *Diagramme de classe du module TestSeamCarving*

Ce projet est composé de 3 classes, comme vous pouvez le voir sur le diagramme de classe Figure 7.

1. XMLManager : c'est la classe qui s'occupe de gérer toute la partie XML (création, modification, sauvegarde, etc.)
2. BitmapManager : c'est la classe qui s'occupe de gérer toute la partie image (extraction de polygone, sauvegarde, rognage, etc.)
3. Program : c'est la classe qui lancera le projet dans son main. Cette classe possède également des méthodes qui permettent d'appeler des modules extérieurs (binarisation et extraction de caractéristiques). Enfin, elle possède également la méthode "ParseJSONToXML" qui permet de remplir un fichier XML à partir d'un fichier JSON.

Il faut voir les classes "XMLManager" et "BitmapManager" comme des librairies que j'utilise dans ma classe "Program".

2.2.1 Extraction d'un polygone sur l'image original

Une fois mon fichier XML créé, je dois extraire chaque ligne segmentée en image. La difficulté ici était d'extraire non pas un rectangle, mais un polygone de l'image original. Je me suis tourné au début vers une librairie nommée "ImageMagick" qui s'est avéré inadaptée. Finalement, le moyen le plus simple était de passer simplement par des objets "Graphics". Une fois mes images générées, le problème était la gestion de la taille de l'image, impossible d'avoir la bonne taille d'image. J'ai donc par la suite développé une méthode qui me permet de rogner une image.

2.2.2 Binarisation

Afin de binariser nos images, il m'a fallu faire une petite manipulation. En effet étant donné que j'ai des images qui sont en fait des polygones, j'ai des zones où je n'ai pas de fond, ce qui pose problème lors de la binarisation. Lors de la binarisation, tous les pixels vides deviennent noirs. C'est pourquoi j'ai ajouté une étape qui permet de colorer les pixels vides en blanc afin de ne pas impacter la binarisation. Alors certes, on perd notre polygone et on retrouve un rectangle englobant, mais nous gardons tout de même l'avantage de ne pas avoir de texte parasite dans les zones qui étaient vides au préalable.

2.2.3 Appel à l'extraction de caractéristiques

Une fois que toutes mes images sont créées, il ne me reste plus qu'à faire un appel à l'extraction de caractéristiques mise en place par Yamin. Dans la méthode mise en place par Yamin, il faut appeler son constructeur en lui passant deux images. Par souci de performance, j'ai préféré modifier cette partie en ne lui passant plutôt deux paramètres :

1. Le chemin de l'image originale
2. Le chemin de l'image Binarisée

Cela permet de gagner en rapidité.

2.3 Extraction de caractéristiques

Ici, l'objectif est de rendre le système d'extraction de caractéristiques beaucoup plus souple qu'il ne l'est aujourd'hui (Figure 1 (Chapitre 4)). En effet, nous n'avons qu'une seule méthode valide (extraction par colonne), et le but, est de pouvoir utiliser facilement plusieurs méthodes en fonction des segmentations faites. C'est pourquoi j'ai dû mettre en plus une structure permettant de facilement ajouter de nouvelles méthodes.

2.3.1 Nouvelle modélisation

Dans ce nouveau diagramme Figure 8, on peut voir qu'il n'y a que deux classes. Une classe abstraite et une autre qui hérite de celle-ci. En voici la description :

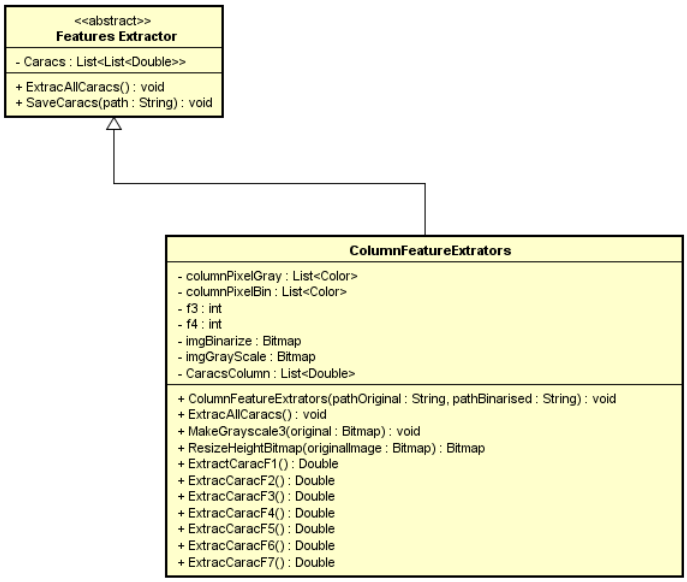


Figure 8 – *Nouveau diagramme de classe*

1. FeaturesExtractor : Classe abstraite qui permet de définir toutes les parties communes à chaque type d'extraction de caractéristiques. Elle implémentera toutes les méthodes dites génériques (Comme SaveCaracs). De plus des méthodes seront également virtuelles pures (comme la méthode ExtractAllCaracs), afin de pouvoir laisser leur implémentation en fonction des méthodes d'extraction. Le fait de mettre cette classe abstraite permet d'obliger toute les classes qui héritent de celle-ci d'avoir d'une part des méthodes toutes faites, mais également l'obligation d'implémenter des méthodes spécifiques à leur cas.
2. ColumnFeatureExtractors : classe qui hérite de la classe abstraite précédemment expliquée. Celle-ci est donc spécifique d'une extraction de caractéristiques par colonne.

2.3.2 Avantages

Les avantages de cette structure, c'est qu'elle permet facilement d'ajouter de nouvelles méthodes d'extraction de caractéristiques. En effet, il suffit simplement d'ajouter une classe qui héritera de la classe abstraite, puis d'implémenter les méthodes virtuelles pures dans la classe abstraite, puis enfin de définir ses propres méthodes. Enfin, j'ai également modifié le constructeur de la classe "ColumnFeaturesExtractor", qui prenait en paramètre deux Bitmap. Cette façon de faire n'était pas optimale dans la gestion de la mémoire. C'est pourquoi au lieu de passer en paramètre des objets, je lui passe deux chemins correspondant aux emplacements des images.

3 Modification de l'affichage des résultats

Dans cette partie, je vais brièvement vous expliquer comment j'ai résolu le problème de l'affichage des résultats dans la plateforme. L'un des objectifs de ce projet était de le faire fonctionner. En effet, comme spécifié dans le Chapitre 2, l'affichage était incorrect, il affichait des rectangles hors de la zone de l'image. Cela était dû à un problème de calcul de redimensionnement. En effet nous calculons nos résultats sur une image taille réelle (1980x2580). Or, quand nous affichons les résultats, nous les affichons sur une image qui s'adapte à la taille de l'imgBox qui la contient, nettement plus petite. Donc c'est ici qu'il faut faire un calcul de redimensionnement des rectangles afin de les adapter à la taille de l'image dans l'imgBox.

4 Cahier du développeur

Enfin pour finir j'ai mis en place un cahier du développeur (Annexe I) afin de regrouper tous les éléments techniques existants, auxquels j'ai ajouté les miens. Le but ici était de créer un document unique regroupant toutes les choses techniques qui peuvent être utiles pour les prochains étudiants travaillant sur le projet. Enfin à cela j'ajoute également toutes les informations nécessaires à la reprise du projet, toutes les informations essentielles et pratiques. C'est un document qui se verra vivre au fil du temps de projets en projets.

6 Bilan et conclusion

1 Les choses faites

Plusieurs choses ont été faites durant cette première partie. Celle-ci a été entièrement consacrée à la recherche. Pendant cette période, j'ai pu découvrir dans un premier temps l'application existante. J'ai eu le temps de me familiariser avec, afin de comprendre comment elle était faite. Cela a également impliqué une compréhension de tous les autres projets qui interagissent avec le projet principal à savoir la plateforme de wordSpotting (Binarisation, segmentation, extraction de données, IwordSpotting). Une fois toutes ces choses assimilées, je me suis lancé dans la recherche, guidé par Monsieur Ragot. J'ai donc pu découvrir plusieurs types de méthodes pour la segmentation et pour l'amélioration de l'extraction de données. Avec Monsieur Ragot nous avons établi des priorités sur les choses à mettre en place pendant le second semestre consacré à la partie développement. Enfin j'ai fait part de quelques améliorations à Monsieur Ragot en ce qui concerne la plateforme en elle-même. Le but encore une fois est de pouvoir améliorer le système dans son ensemble y compris l'expérience utilisateur. Dans cette partie est également compris les corrections de bug mineurs. C'est pourquoi j'ai également noté quelques bugs à corriger.

2 Choses encore à faire

Avant de commencer toute chose, il va falloir que je mette en place des tests qui vont déterminer la performance des algorithmes et donc des méthodes à mettre en place par la suite. En effet il est difficile de dire que telle ou telle méthode est plus efficace qu'une autre, surtout avec le peu d'expérience que j'ai dans ce domaine. Une fois les tests effectués, je mettrai en place les solutions que l'on aura décidé conjointement avec Monsieur Ragot. Dans un second temps, si le temps me le permet, je me consacrerai à l'amélioration de la plateforme de wordSpotting.

3 Bilan sur la qualité et ouverture

En terme de qualité, je pense avoir fourni quelque chose de fonctionnel. En effet grâce aux divers tests que j'ai réalisé je suis en mesure d'assurer que la qualité du projet est là. Chaque

module auquel j'ai touché a été testé plusieurs fois. Les tests peuvent paraître très simples et banals, mais il n'y avait aucune raison de compliquer ou même d'automatiser ceux-ci.

En revanche, bien que j'ai répondu à tous les objectifs principaux, il y a quelques points sur lesquels il serait, à mon avis, intéressant de travailler.

1. Mettre en place un module générique pour la segmentation comme ce que j'ai fait pour l'extraction de caractéristiques, afin de pouvoir facilement ajouter de nouvelles méthodes.
2. Mettre un système qui permettrait de facilement choisir le chemin de nos images, par exemple pour segmenter une base d'images en particulier, au lieu de devoir l'écrire en dur dans le code. Cela peut être fait par un argument à passer en console.
3. Mettre en place des tests automatiques qui permettraient par exemple, à la fin d'une segmentation ou d'une extraction de données, nous sortir des résultats statistiques.

4 Gestion de projet

4.1 Gantt réel S9

Comme vous pouvez le voir en Annexe E, la première partie de mon PRD était donc séparée en plusieurs cycles.

1. Le premier le plus important, la prise en main du sujet : celui-ci englobait la compréhension de l'existant, le report de bugs afin de les corriger plus tard, suggestions d'améliorations, et l'organisation du projet (reprise du Redmine).
2. Le cycle de recherche : celui-ci était séparé en deux sous-parties correspondant aux deux principaux axes d'amélioration du projet existant
   (a) Segmentation
   (b) Extraction de données
3. Le cycle de la rédaction de l'état de l'art. Cette partie est également importante, car elle fait état de toutes mes recherches. C'est pourquoi c'est un cycle plus grand que les autres. En effet, j'ai du à plusieurs moments revenir sur mes écrits afin de les corriger pour être le plus juste possible. Cela impliquait parfois un retour sur certains articles que je n'avais pas assez bien assimilés. A la fin de celui-ci une version finale du rapport a été livrée.
4. Enfin, le dernier cycle, la préparation à la soutenance : ce cycle est le dernier de la première partie de ce PRD. En effet, cette soutenance vient la clôturer. Dans ce dernier, je me suis concentré à mettre en place tous les éléments dont j'avais besoin afin d'être prêt pour celle-ci.

4.2 Gantt prévisionnel S10

Vous trouverez en Annexe F, le diagramme de Gantt prévisionnel. Aujourd'hui, il m'est impossible de détailler plus les tâches que je réaliserai. En effet, ils vont dépendre des résultats des tests que j'effectuerai au tout début de la partie développement. Néanmoins, je suis capable d'identifier 3 sprints :

1. Segmentation : cette partie sera consacrée à mettre en œuvre une nouvelle méthode permettant d'améliorer les performances de segmentation
2. Extraction de données : cette partie consistera à mettre en place un module de prétraitement à la partie existante afin d'améliorer les résultats obtenus, et le temps d'exécution.

3. Plateforme : dans cette dernière partie, je corrigerai les bugs que j'ai pu remonter lors de la partie recherche, ou encore d'autres que je pourrais découvrir durant la deuxième phase de ce projet. Dans cette partie, je pourrais également intégrer des modules existants mais non-disponibles dans l'application (recherche par dictionnaire, intégration de l'outil renom,..)

4.3 Gantt Réel S10

Vous trouverez en Annexe Gle diagramme de Gantt réel de la deuxième partie de ce PRD. On peut voir une nette différence entre le diagramme réel et prévisionnel. En effet, on voit nettement la différence sur les cycles "segmentation" et "extraction de caractéristiques".

4.3.1 Auto-critique

Comme évoqué précédemment, mes deux diagrammes de Gantt ont quelques différences notables. Cela est principalement dû à un manque de connaissances et de recul dans ce domaine. En effet, c'était la première fois que je faisais un projet dans ce domaine, donc il est difficile d'évaluer le temps nécessaire à une tâche que je n'ai jamais réalisé. De plus, la trame de développement n'était pas vraiment établie au moment de la création du diagramme de Gantt prévisionnel. Plus encore, certains cycles se sont même chevauchés. En effet, j'ai dû revenir quelques fois sur des éléments qui n'étaient pas ou plus fonctionnels en fonction des modifications que j'apportais. En effet, les trois cycles étaient tous plus ou moins liés. C'est pourquoi en fonction de mes modifications, il m'arrivait de revenir en arrière sur certaines tâches. Me lançant dans un domaine encore inconnu et un langage que j'ai très peu pratiqué, j'ai mis du temps à me mettre dans le projet et me sentir à l'aise, c'est pourquoi le début à été compliqué, ce qui a impacté directement les tâches suivantes.

Pour finir, bien que la planification n'a pas été au mieux sur cette deuxième partie de ce PRD, j'ai tout de même réussi à compléter les objectifs principaux de ce projet. ∎

Annexes

A Description des interfaces externe du logiciel

Interface homme - machine

L'interface est déjà existante et ne sera modifiée que sur de petites parties. Celle-ci résulte du fruit du travail de trois étudiants respectivement Zheng Zhang, Loreen lambin et Yamin Zaidou, sur trois PRD différents. Cette IHM est composée de quatre onglets :

1. Home, qui regroupe la partie principale du système. Dans celui-ci nous pouvons visualiser et naviguer dans les bases d'images, effectuer une recherche et visualiser le résultat. Dans cette partie je vais faire en sorte de modifier l'affichage des résultats afin qu'ils soient plus pertinents. Comme on peut le voir sur l'image ci-dessous, les résultats affichés ne sont parfois même pas sur des mots existants, mais sur des espaces vides.

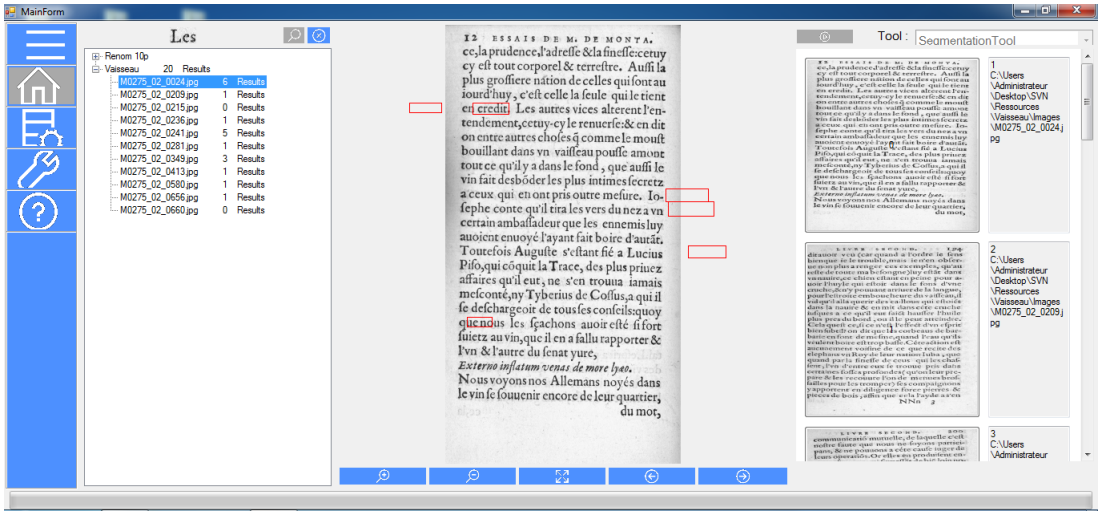


Figure 1 – *Visualisation des résultats*

2. Import / manage base, c'est la partie qui contiendra l'espace de gestion des bases d'images (ajout et suppression). Il est également possible d'affecter une ou plusieurs méthodes de recherche (Tool) à une base d'image. Dans cette partie je vais devoir modifier l'existant afin de corriger des problèmes d'actualisation
3. Tool WordSpotting, c'est l'espace qui permet de gérer les algorithmes de wordSpotting que l'on souhaite utiliser dans la base (ajouter et supprimer).

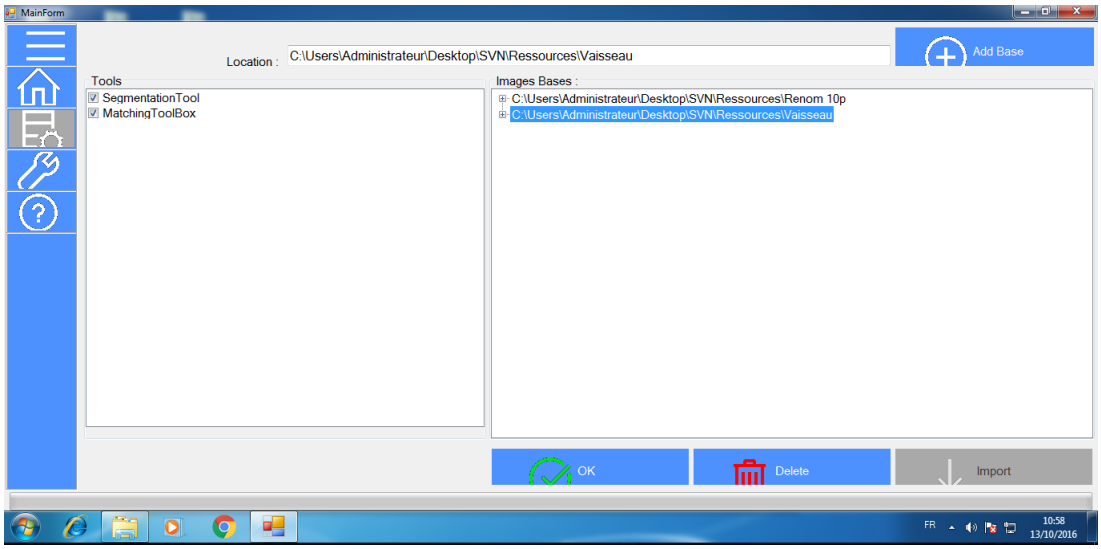


Figure 2 – *Menu importation de base*

4. About : cet espace n'est actuellement pas utilisé et est donc vide.

Interface logiciel - logiciel

L'application étant construite avec plusieurs projets, il faut que la plateforme puisse dialoguer avec chacun d'entre eux. Comme spécifié par Zheng dans son cahier de spécifications [8] et Laureen [2], plusieurs informations vont être transmises entre les différents logiciels afin de pouvoir effectuer le traitement souhaité. Ces éléments sont regroupés dans un objet appelé « request ». Voici les éléments qui constituent une request :

1. Le type de recherche : Il peut y avoir trois types
   (a) Image : correspond à la zone que l'on trace avec le curseur pour définir l'élément à rechercher
   (b) Dico : La recherche par dico permet de composer un mot à rechercher sans passer par la recherche par image
   (c) Il y a également le type Mots, mais qui n'est pas repris dans le dernière version de la plateforme.
2. Le chemin vers le fichier Jpeg : fait référence au chemin qui conduit jusqu'à l'image recherchée
3. Le chemin vers le fichier CSV : fait référence au chemin où le fichier CSV est stocké

Une fois la recherche effectuée, un fichier Resultat.xml est créé, celui-ci contiendra la liste des résultats obtenus. Enfin de manière générale, nous avons également deux fichiers de configuration

1. ListBase.xml (Figure 3) : Il contient le chemin des bases stockées dans l'application ainsi que tous les types de recherche qui y sont associés.
2. ListTools.xml (Figure 4) : Il contient les chemins vers les exécutables qui permettent de faire les recherches et qui sont stockées dans l'application. Ils sont également associés aux types de recherche (Image, word, dico).

C'est dans ces deux fichiers que je vais devoir aller écrire afin de pouvoir rajouter l'outil Renom, mais également la recherche par dico. En effet en fonction des balises présentes dans ces fichiers XML, la plateforme sera en mesure de comprendre qu'un nouvel outil existe ou encore qu'un type de recherche peut être appliqué à un type d'algorithme.

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <Base path="C:\Users\Administrateur\Desktop\SVN\Ressources\Vaisseau">
    <tool name="MatchingToolBox" />
  </Base>
  <Base path="C:\Users\Administrateur\Desktop\SVN\Ressources\Renom 10p">
    <tool name="MatchingToolBox" />
  </Base>
</root>
```

Figure 3 – *Fichier XML ListBase*

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <tool name="MatchingToolBox" path="C:\Users\Administrateur\Desktop\SVN\Trunk\Dev\iwordspotting\Bastier
    <request name="image" />
    <request name="word" />
    <request name="dictionary" />
  </tool>
</root>
```

Figure 4 – *Fichier XML ListTool*

B Spécifications fonctionnelles

Dans cette partie sera détaillé toutes les fonctionnalités que je vais devoir réaliser afin de répondre à la problématique du sujet. Pour chacune d'entre elles, une brève description ainsi qu'un niveau de priorité sera détaillé.

Amélioration de l'extraction de caractéristique

Cette fonctionnalité a été développée par Yamin dans son PRD de l'année passée en se basant sur la thèse de Tanmoy Mondal, [4]. Le but de cette méthode est de pouvoir extraire les caractéristiques d'une image, c'est l'étape qui suit la segmentation. Ces caractéristiques vont donc nous permettre de décrire une image. C'est par ces caractéristiques que nous allons pouvoir comparer deux images. La méthode que Yamin a mise en place est la méthode d'extraction à base de colonnes. Dans la version existante, la méthode prend en entrée une image originale et binarisée. Enfin en sortie la méthode va nous retourner toutes les caractéristiques dans une liste. Bien que cette fonctionnalité soit fonctionnelle, on peut bien évidemment l'améliorer. Le but de cette partie va donc de mettre en place une méthode qui permette d'améliorer le résultat obtenu ou voir même de le rendre plus rapide si le temps me le permet. En effet à l'heure actuelle, suivant la taille de l'image le temps d'exécution de l'algorithme peut être assez conséquent. Avant toute chose, cette partie va nécessiter une période de recherche afin de pouvoir faire un état de l'art des techniques existantes et de pouvoir les comparer dans le but de choisir la méthode la plus adéquat aux exigences du projet, mais également au regard de mes capacités et du temps que je dispose.

Amélioration de l'outil de segmentation

Comme pour la fonctionnalité précédente, elle avait été mise en place par Yamin lors de son PRD de l'année dernière. La segmentation intervient en prétraitement. Son but est de séparer chaque ligne de l'image originale. La méthode mise en place par Yamin utilise donc un mixte de deux méthodes, « projection profile » et « Horizontal RLSA » afin de faire une segmentation en ligne. Le programme a besoin en entrée de l'image binarisée. Une fois que le programme a fini son exécution, il sauvegarde les images segmentées suivant un certain format. En effet le résultat final de la segmentation sera stocké dans le dossier « Training » créé dans la base

d'images. D'autres informations sont également stockées tel que les coordonnées ou encore les images rejetées. Encore une fois, le but cette année va être de reprendre le travail de Yamin dans le but de l'améliorer. L'objectif va être d'avoir de meilleurs résultats, mais également faire fonctionner la segmentation sur des documents numérisés de type manuscrit. Cette partie va également nécessiter une période de recherche. Celle-ci va permettre de faire un état de l'art des autres méthodes existantes et de pouvoir les comparer sur certains critères, tel que la pertinence des résultats ou encore la vitesse d'exécution.

Amélioration de l'affichage des résultats de recherche

Actuellement le seul algorithme présent dans la plateforme est l'outil iWordSpotting intégré par Yamin. Ce dernier est fonctionnel, mais les résultats affichés ne sont parfois pas cohérents. Comme présenté dans la Annexe A. Mon but va être d'analyser le problème et de parvenir à rétablir le bon fonctionnement de l'algorithme. De plus les résultats vont devoir être bien visibles dans l'application comme dans l'existant. Voici le fonctionnement : les résultats de la recherche sont stockés dans un fichier xml stockant notamment l'image ainsi que ses coordonnées associées. Cette méthode va donc prendre en entrée le chemin vers le fichier xml. Grâce à ce dernier il va savoir à quelles coordonnées créer un rectangle de couleur rouge, afin de mettre en valeur le résultat auprès de l'utilisateur. En sortie de cette méthode nous auront une liste de rectangle qui sera affichée sur la plateforme plus tard. Cette méthode dépend donc fortement du fichier XML résultat, vu que c'est lui qui stocke et transmet tous les résultats obtenus.

Exporter les résultats

Cette fonctionnalité n'est pas existante et n'est pas d'une grande priorité. Celle-ci consistera à mettre en œuvre un moyen de pouvoir exporter les recherches effectuées. Cela peut se matérialiser de la façon suivante. Créer un dossier qui contiendra chaque fichier "resultats.xml" créé lors de chaque recherche. Ce dernier contiendra la date et l'heure à laquelle la recherche à été effectuée. Cette fonctionnalité n'implique aucune modification dans la plateforme, de plus celle-ci sera transparente à l'utilisateur standard.

Réorganiser les résultats

Cette fonctionnalité n'est également pas disponible dans la version actuelle, et de même que la fonctionnalité précédente, elle n'est pas d'une grande priorité. Cette fonctionnalité consistera à mettre en œuvre un moyen de pourvoir réorganiser les résultats de recherche proposés par l'algorithme de recherche. En effet des résultats peuvent ne pas être organisés comme le souhaiterais l'utilisateur. C'est dans cette optique qu'intervient cette tâche. Son but va être de donner à l'utilisateur la possibilité de réorganiser les résultats. Par exemple si le résultat numéro 3 est plus intéressant que le choix 2 pour l'utilisateur, il aura alors la possibilité de modifier la pertinence des résultats afin de mettre le choix numéro 3 en choix numéro 2.

Importation de base d'image

Comme expliqué dans la Annexe A, c'est une partie de l'interface qui nécessite quelques modifications. En effet, la suppression d'une base existante dans l'interface n'est pas fonctionnelle. De plus lors d'un ajout d'une base, l'écran d'accueil n'est pas mis à jour, la base n'apparait pas.

Enfin on peut améliorer le fonctionnement initial afin de pouvoir modifier un chemin d'une base existante dans le cas où l'on souhaite déplacer une base d'images.

C Spécifications non fonctionnelles

Contraintes de développement et conception

1. Langage de programmation : C++ et C# avec le framework .NET 4.5
2. Logiciel : VisualStudio 2012
3. Environnement : Machine virtuel de l'école, Windows 7 Pro 64 Bits
4. Bibliothèques de programmes imposées : OpenCV, AForge.NET

Contraintes de fonctionnement et d'exploitation

Performances

S'il me reste du temps, je mettrais en place des outils permettant de visualiser et de vérifier les performances des algorithmes. Cela passera par des protocoles d'expérimentation.

Capacités

Quelques limites sont à noter :

1. Une seule recherche à la fois, en effet le système ne pourra pas prendre en compte plusieurs recherches à la fois étant donné que pendant la période de recherche l'application est bloquée.
2. La taille des images. Il faut savoir que plus la taille des images est grande plus le traitement sera long idéalement cette taille ne doit pas dépasser 5Mo.
3. Le nombre d'images dans une base. Cette contrainte est quelque peut liée à la précédente. En effet plus le nombre d'image sera important, plus le temps de traitement sera long. De manière générale le nombre d'image va également dépendre de la taille de celles-ci. Plus les taille sont petite, plus le nombre d'images dans la base peut être grand. A l'inverse, plus la taille d'image est grande moins d'images devront être présentes dans la base.

Contrôlabilité

Tout au long de l'exécution de l'application, des indications seront visibles par l'utilisateur dans l'application lorsque ces actions déclenchent des erreurs. De plus des traces seront disponibles lors des recherches, celles-ci permettent de visualiser le détail des résultats des recherches effectuées. Mon rôle sera également d'améliorer le retour des erreurs, afin que l'utilisateur ne soit pas perdu.

Maintenance corrective

Cette partie va contenir tous les défauts visuels et fonctionnels de l'application. Le but sera d'en corriger le plus possible afin d'avoir une application la plus stable possible. La principale consistera à améliorer les messages d'erreur afin d'avertir au maximum l'utilisateur pour qu'il ne soit pas perdu et pour qu'il sache quoi faire pour éventuellement corriger cette erreur. Enfin une amélioration à apporter consistera à éviter le blocage au lancement de l'application lorsque l'application ne trouve pas l'emplacement d'une base d'image ou encore d'un algorithme. Dans la version existante si l'application ne trouve plus l'un des deux, elle affichera un message d'erreur et elle ne s'ouvrira jamais, la seule solution consistera à modifier le fichier XML à la main. C'est pourquoi il faudra mettre en place un système qui permettra d'ouvrir une boite de dialogue dans laquelle on pourra redéfinir le chemin où se trouve les bases d'images ou les algorithmes, une fois modifiée l'application pourra se lancer normalement.

Maintenance et évolution du système

Le système doit être adaptable, cela implique une bonne structure du système afin de pouvoir facilement ajouter de nouvelles parties à l'application. Le projet existe déjà depuis quelques années, et plusieurs personnes ont travaillés dessus, c'est pourquoi il va être important de conserver les versions. C'est pourquoi nous allons utiliser le Redmine de Polytech afin de faciliter le stockage et la gestion. Enfin si le temps nous le permet des tests de non régression seront mis en place sur l'interface, et des tests d'évolutions de versions d'OpenCV seront mis en place afin de pouvoir envisager ou non une migration de version.

D Gestion de Projet

Méthode de suivi de projet

En accord avec Monsieur Ragot, nous avons décidé de séparer en deux parties.

1. Première partie correspondant à la phase de recherche : Cycles itératifs
2. Deuxième partie correspondant à la phase de développement : Méthode agile Scrum

Nous avons choisi cette organisation, car c'était la méthode utilisée par Yamin lors de son PRD sur ce projet l'année passée. De plus la période de recherche ne peut pas être rythmée de la même façon que la période de développement. En effet, lors de mes recherches, beaucoup de remise en question et de re-planification peuvent avoir lieux en fonction des résultats obtenus. Durant cette première phase, de nombreux cycles seront effectués. Pour chacun d'entre eux, des recherches seront menées sur un sujet en particulier. Une fois mes recherche effectuées (correspondant à la fin du cycle), je rédigerais un bref résumé pour mon état de l'art. De plus une réunion sera fixée avec Monsieur Ragot afin de lui faire part de mes recherches pour ensuite définir sur quel axe m'orienter sur mes prochaines recherches (mes prochains cycles). Cette méthode nous permettra d'avancer pas à pas, en effet à la fin de chaque cycle un nouveau cycle sera défini en fonction des résultats du cycle précédent. De manière générale le cycle actuel ne dépendra uniquement du cycle précédent.

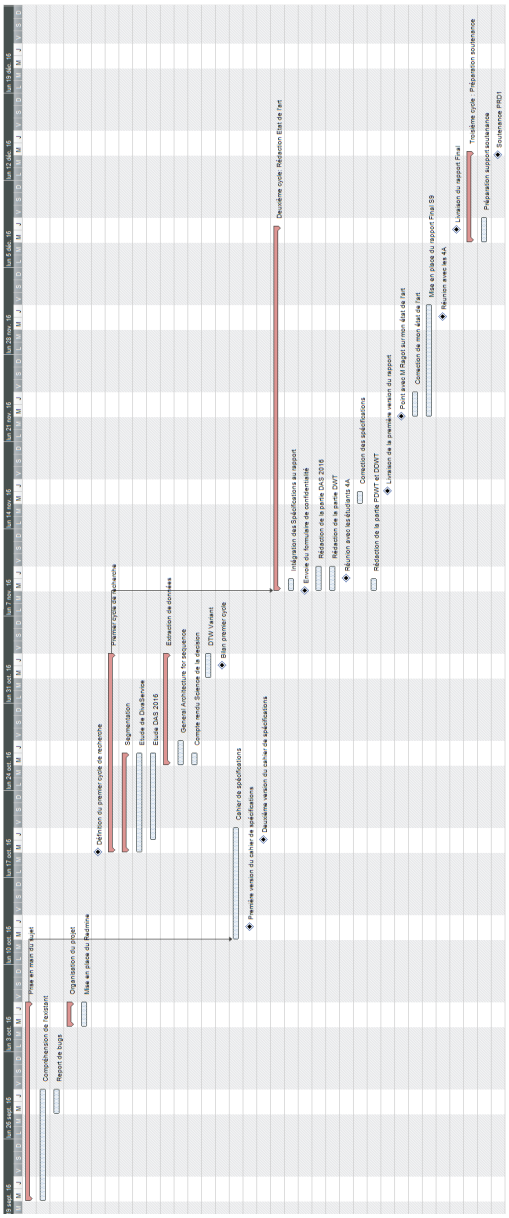
En ce qui concerne la deuxième phase, une méthode agile sera mise en place. En effet pour cette partie les choses à réaliser seront déjà définie et claires de par la précédente période de recherche. C'est pourquoi nous avons choisi d'utiliser la méthode agile Scrum. Des sprints seront mis en place, pour chacun d'eux un nombre fixé de fonctionnalité seront à mettre en place. A chaque fin de cycle une réunion avec Monsieur Ragot pourra être mise en place pour évaluer le résultat et voir si le prochain sprint doit être modifié en revenant sur certaines fonctionnalités de l'ancien sprint ou non. Cela nous permettra de nous adapter et de réévaluer la demande à chaque fin de sprint. Le but final étant d'avoir la plus de chose fonctionnelle sur la plateforme.

Outil de versionning

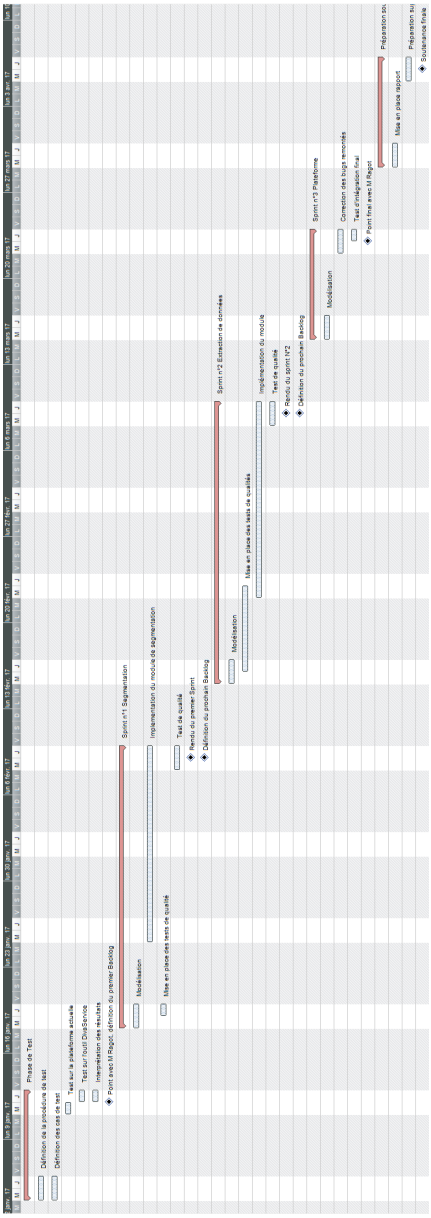
Nous avons choisi avec Monsieur Ragot d'utiliser le Redmine et tortoiseSVN afin de pouvoir intéragir facilement avec le dépôt. Nous avons choisi Redmine car polytech possède déjà un

espace sur Redmine. De plus étant donné que deux étudiants de quatrième année travaillent également sur le même projet que moi, il est plus facile de synchroniser nos tâches grâce à l'outil Redmine. En effet cet outil est pratique, en plus du simple dépôt, nous avons la possibilité de définir des demandes (des tâches) de plusieurs types différents, plusieurs niveaux de priorité différents, de les affecter à une personne. Enfin en fonction de toutes ces tâches, un diagramme de Gantt est généré automatiquement.

E Diagramme de Gantt S9 Réel

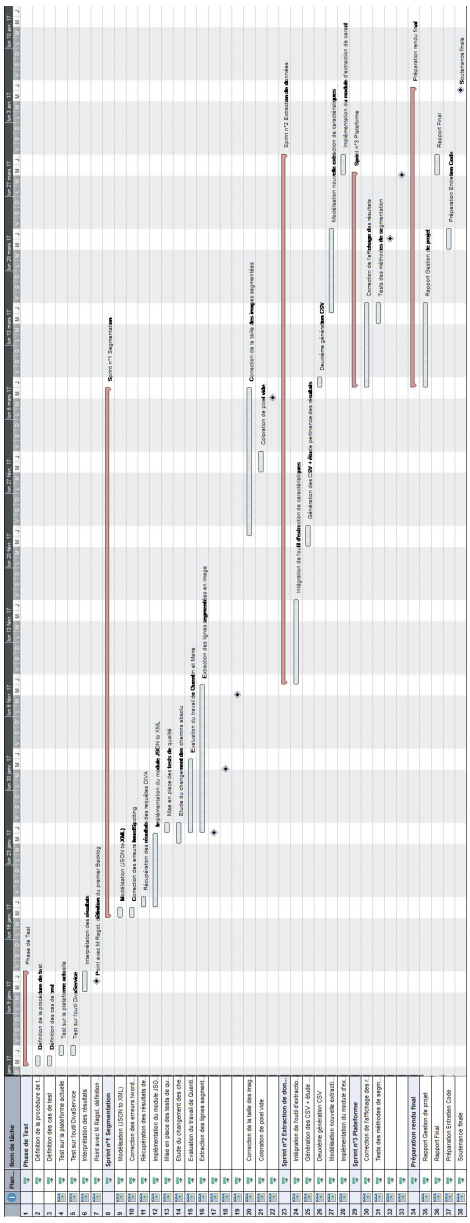


F Diagramme de Gantt S10 prévisionnel

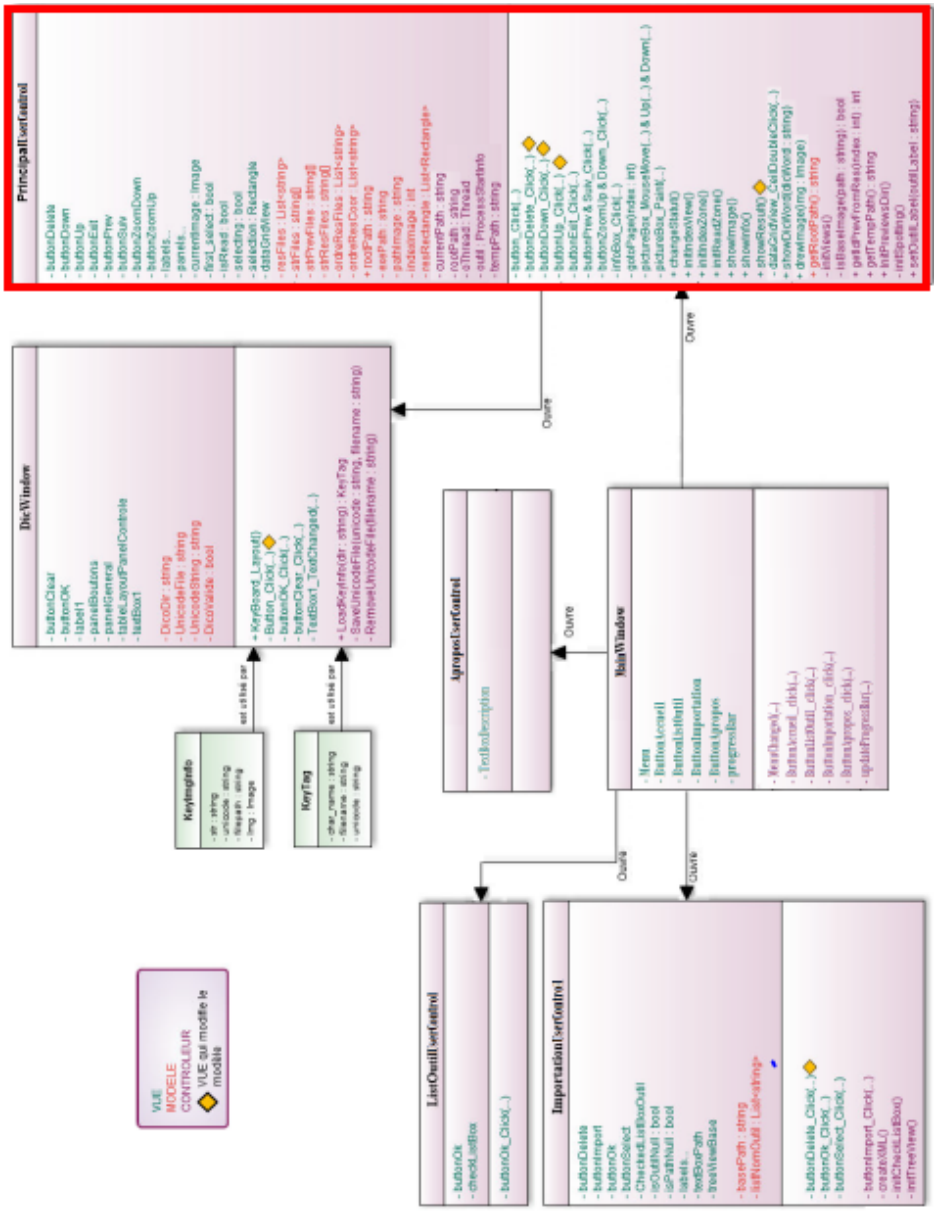


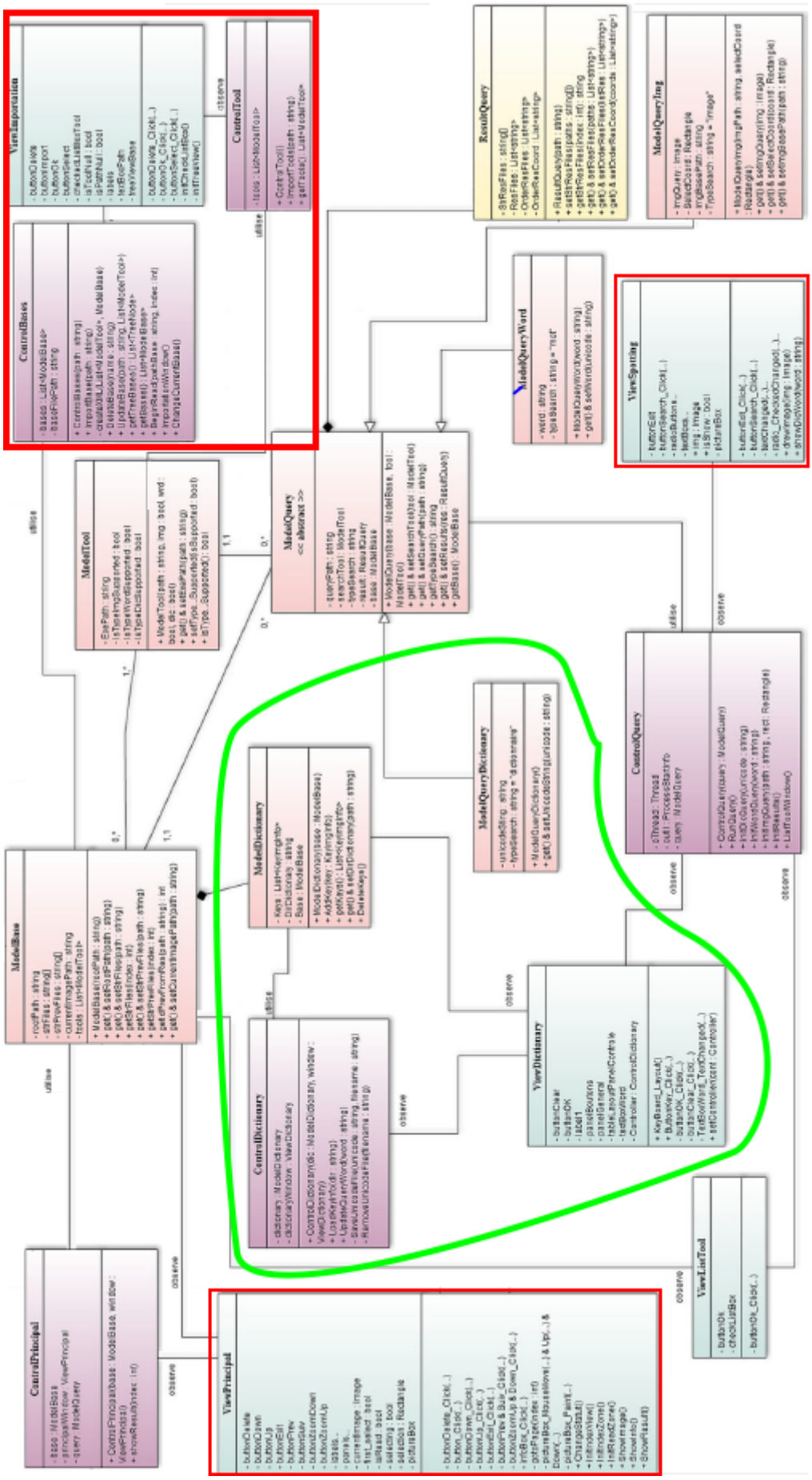
G

Diagramme de Gantt S10 réel



H Ancien diagrammes de classes





I Cahier du développeur

POLYTECH©
TOURS

CAHIER DU DEVELOPPEUR

PLATEFORME DE WORDSPOTTING

Dans ce document vous sera présenté tous les éléments techniques que vous aurez besoin afin de comprendre le projet et le reprendre facilement

TABLE DES FIGURES

Sommaire

TABLE DES FIGURES

Table des figures

Fonctionnement du projet

Le diagramme ci-dessous montre le fonctionnement global de l'application.

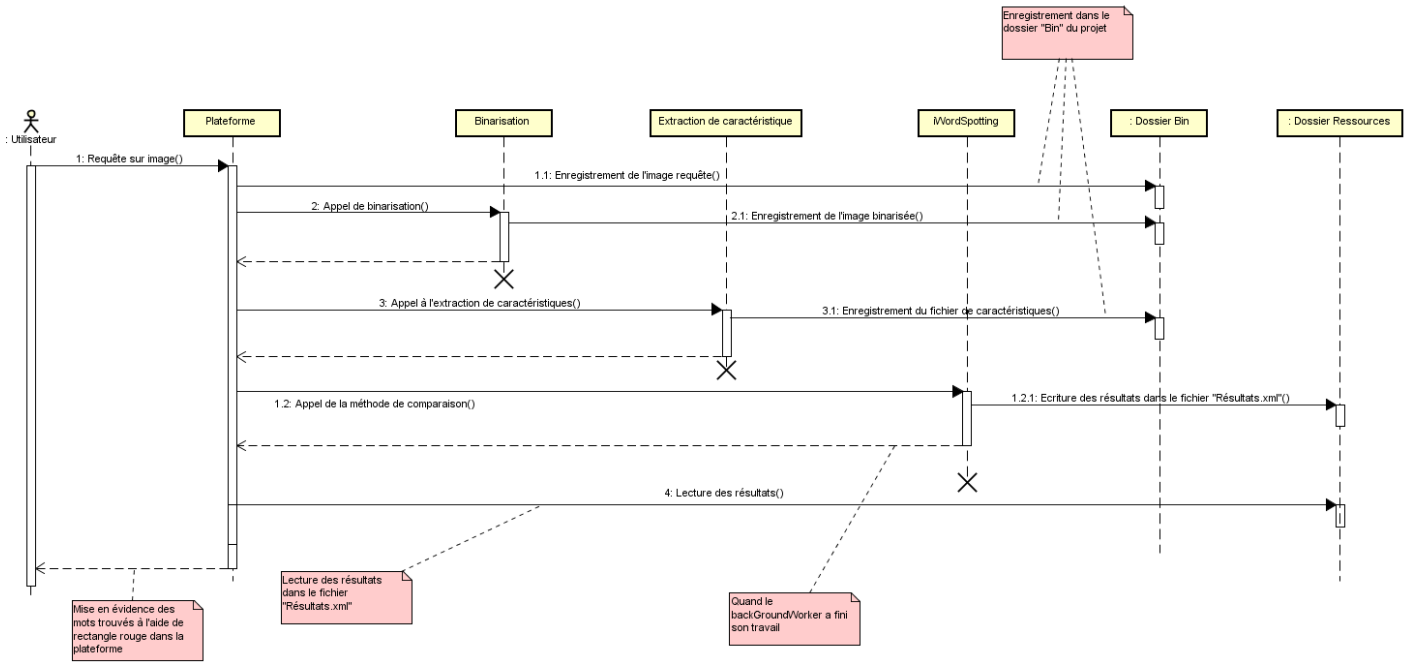


Figure 1 : Diagramme séquence du projet global

Dans le diagramme ci-dessus, on peut voir l'interaction entre les projets lors d'une recherche.

1. Une fois la requête lancée, on enregistre l'image requête dans le dossier Bin
2. Ensuite on fait appel au projet de binarisation afin de binariser notre image requête. On l'enregistre dans le dossier Bin
3. Ensuite la plateforme fait appel à l'extraction de caractéristiques afin d'avoir les caractéristiques de l'image requête. On l'enregistre encore dans le dossier Bin
4. Ensuite la plateforme fait appel au module due comparaison de caractéristiques nommé « IwordSpotting ». Ce module export ses résultats dans un fichier XML dans le dossier « Ressource » (dossier qui contient toute les bases d'images).
5. Enfin la plateforme lie ce fichier XML afin d'afficher les résultats sous forme de rectangles rouges dans la plateforme.

DESCRIPTION DES PROJETS

Description des projets

L'application de word-Spotting est composée de plusieurs projets afin d'assurer son fonctionnement, dans cette partie, je vais vous présenter chacun d'eux.

PLATEFORME

C'est le projet central de l'application, il permet de parcourir des dossiers d'images, de les afficher, et de lancer des recherches sur le dossier d'image en question. La plateforme permet également de gérer des techniques de spotting. Si plusieurs techniques sont possibles, alors il est facile de les importer dans la base afin d'avoir le choix de la méthode à utiliser lors du lancement de la recherche.

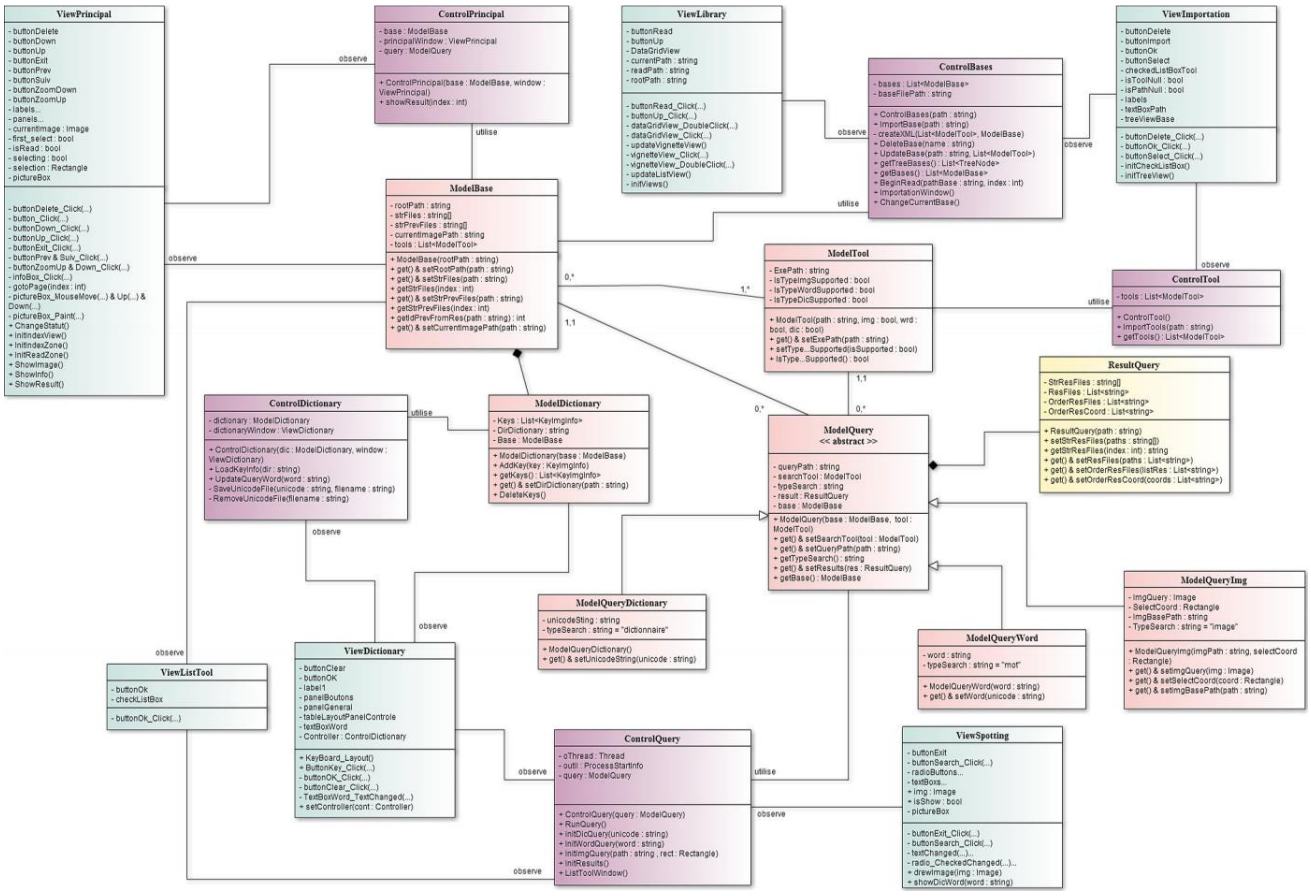


Figure 2: Diagramme de classe de la plateforme

Vous trouverez en annexe les diagrammes de séquence qui détail le fonctionnement du système.

DESCRIPTION DES PROJETS

Ajout d'une base ou d'un dossier d'image

Comme évoqué plus tôt, il est possible d'ajouter via la plateforme une nouvelle base d'image ou encore un nouvel outil de spotting. Pour que le système le garde en mémoire, il va l'enregistrer dans le fichier xml (listBases.xml pour les bases d'images, ou listTools.xml pour les outils de spotting). Si vous avez des problèmes de chemin ou autre et que la plateforme n'affiche pas vos images ou encore vous ne pouvez pas choisir votre nouvelle méthode de spotting, c'est ici que vous allez devoir regarder si les chemins sont corrects. Ses fichiers sont présents dans le dossier Bin du projet.

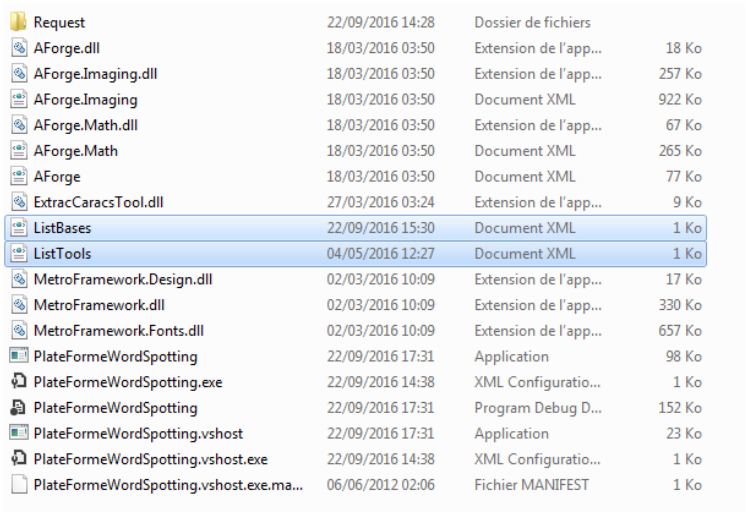


Figure 3: Emplacement fichier listTools et listBases

BINARISATION

Le projet de binarisation est très simple, il permet simplement de binariser une image.

IWORDSPOTTING

C'est le projet qui représente notre méthode de spotting. Ce projet va donc permettre de comparer deux images (images requête et notre image original) afin de repérer les endroits éventuels ou notre image requête serait présente. Pour comparer les images, il va comparer les fichiers de caractéristiques (.csv). Ce projet est développé en C++.

EXTRACTION DE CARACTERISTIQUES

C'est le projet qui s'occupe d'extraire les caractéristiques d'une image. Le but ici, c'est d'extraire 7 caractéristiques et de les stocker dans un fichier CSV. Voici sa modélisation :

DESCRIPTION DES PROJETS

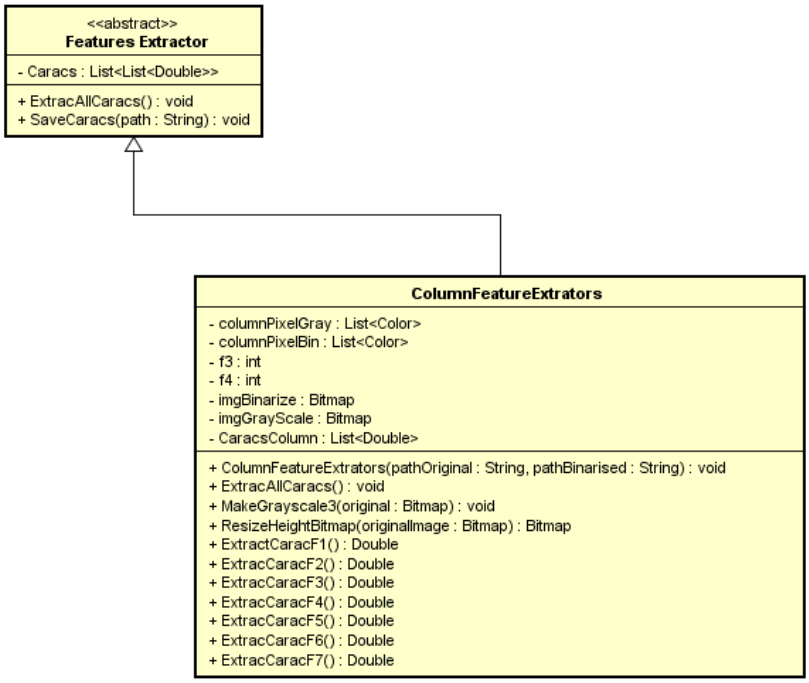


Figure 4: Diagramme de classe Extraction de caractéristiques

Ici, le principe est de rendre le système facilement extensible. Chaque nouvelle méthode devra hériter de la classe abstraite « FeaturesExtractor » afin d'utiliser les méthodes dites « générique » et d'implémenter les autres méthodes spécifiques à son cas.

SEGMENTATION

Ce module a été mis en place par Yamin (PRD 2015-2016). Celui-ci permet d'extraire des lignes rectangulaires d'une image. Voici sont diagramme de classe :

DESCRIPTION DES PROJETS

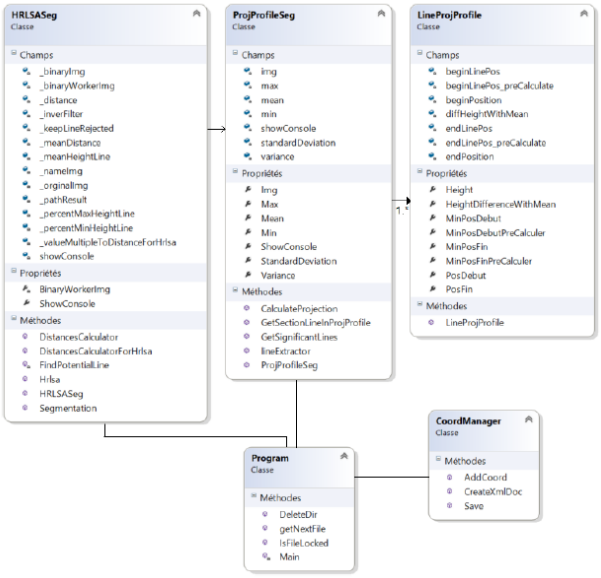


Figure 5 : Diagramme de classe de l'outil segmentation de Yamin

TESTSEAMCARVING

Ce projet, permet de tester une méthode de segmentation utilisant des polygones au lieu de rectangle comme la méthode précédente. Le but de ce module est donc de tester la robustesse de cette méthode, la méthode « Seam Carving ». Ce projet n'implémente donc pas cette méthode. Au lancement du programme, je possède déjà les résultats de la segmentation sous forme JSON.

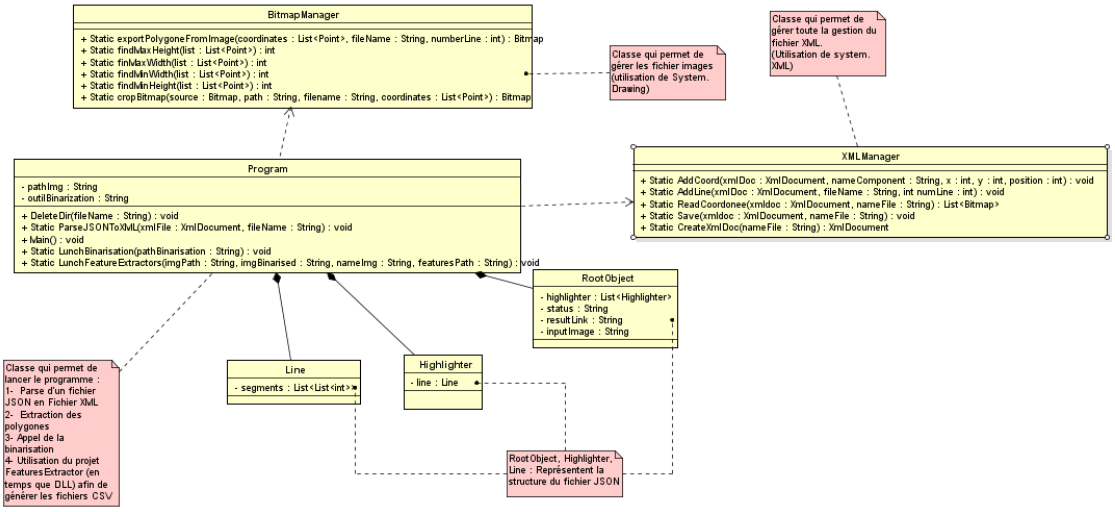


Figure 6 : Diagramme de classe du module TestSeamCarving

Reprise du projet

SVN

Le projet possède un dépôt SVN sur le Redmine de Polytech. L'adresse pour y accéder est la suivante : « http://redmine.polytech.univ-tours.fr/svn/iwordspotting »

Voici son organisation :

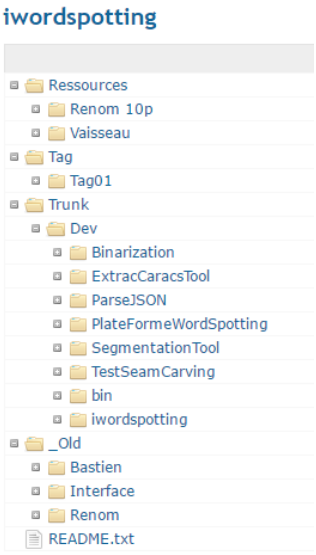


Figure 7 : Arborescence du dépôt SVN

Le dossier « Ressources » contiendra tous les dossiers d'images que nous utiliserons en tant que base d'image dans la plateforme.

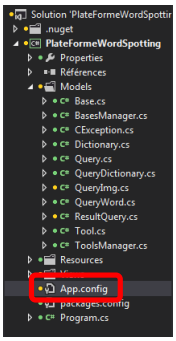
L'onglet « Tag » contiendra toute les versions dites « stable » du projet général.

L'onglet « Trunk – Dev » correspond à la version en cours de développement.

Le dossier « _Old » contient tous les éléments du dépôt avant que je ne le remette en forme.

REPRISE DU PROJET

MODIFICATION DU PATH DANS VISUAL



Lorsque vous allez reprendre le projet, il y a un élément que vous allez devoir modifier, c'est la path. Ce dernier correspond à l'endroit précis ou vous avez placé votre dépôt SVN dans votre environnement local.

Voici la procédure :

1. Ouvrez le projet de la plateforme
2. Allez dans le fichier App.config
3. Modifier la variable Path avec le chemin de votre environnement en local.

```
<appSettings>
  <add key="Path" value="C:/Users/Administrateur/Desktop/SVN"/>
  <add key="outilBinarization" value="C:/Users/Yamin Zaidou/Documents/Visual Studio 2012/Projects/Binarization/Debug/Binarization.exe" />
  <add key="ClientSettingsProvider.ServiceUri" value="" />
</appSettings>
```

Vous noterez également qu'il en va de même pour la variable « outilBinarization ».

LANCEMENT DU PROJET

Le projet compile sous Visual Studio 2012 de la machine virtuelle de l'école « Windows 7 pro 64 bit ». Le projet est généré en debug avec comme plateforme cible « 32 » bits.

OpenCV doit être installé dans la machine virtuelle avec les variables d'environnements (la version utilisé est la 2.4.9). Pour l'installer voici le tutoriel à suivre : http://opencv-srf.blogspot.fr/2013/05/installing-configuring-opencv-with-vs.html

Ensuite il faudra recharger les packages de Nuget, puis configurer les fichiers de config par rapport à la machine.

Remarque : Pour restaurer les packages de Nuget, clic droit sur la solution puis « enable Nuget package restore »

Quelques points à vérifier :

1. Allez dans "Trunk/Dev/PlateFormeWordSpotting/PlateFormeWordSpotting/bin/Debug", puis ouvrez les fichiers listBase et listTools
2. Ensuite dans chacun d'un il va falloir modifier les chemins d'accès pour que l'application puisse retrouver les sources. "C:\Users\Administrateur\Desktop\SVN\Ressources\Renom 10p" est le chemin par défaut (pour la base d'image Renom 10p). Soit vous construisez l'arborescence de cette façon, soit vous modifier le chemin. Exemple : "VotreCheminPerso\Ressources\Renom 10p"
3. Faire de même pour le fichier listTools

Enfin pour lancer l'application, il faut aller dans le dossier debug de l'application "Trunk/Dev/PlateFormeWordSpotting/PlateFormeWordSpotting/bin/Debug", ensuite chercher "PlateFormeWordSpotting.exe" puis lancer le.

Annexe

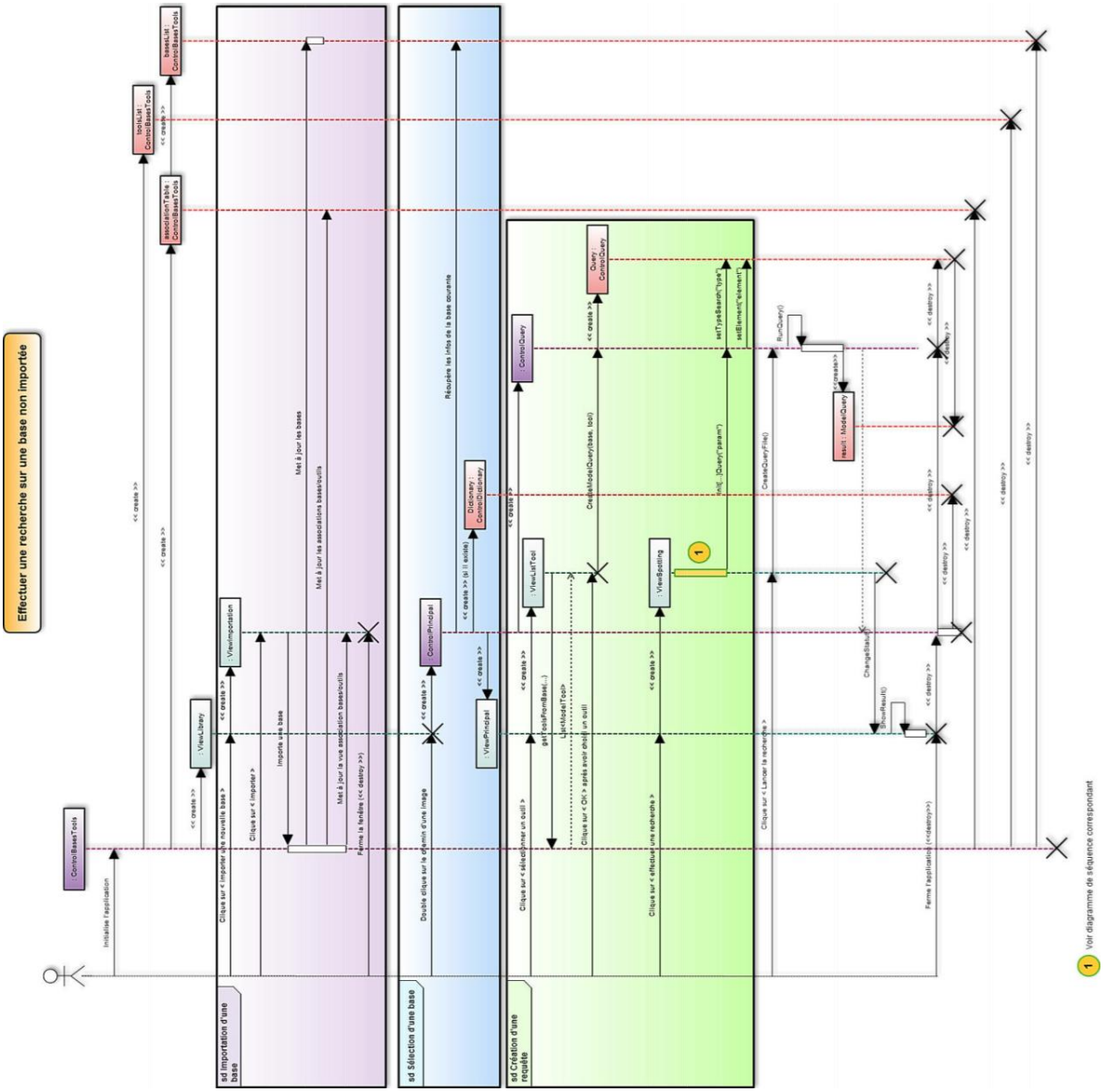


Figure 8 diagramme de séquence global de la plateforme

ANNEXE

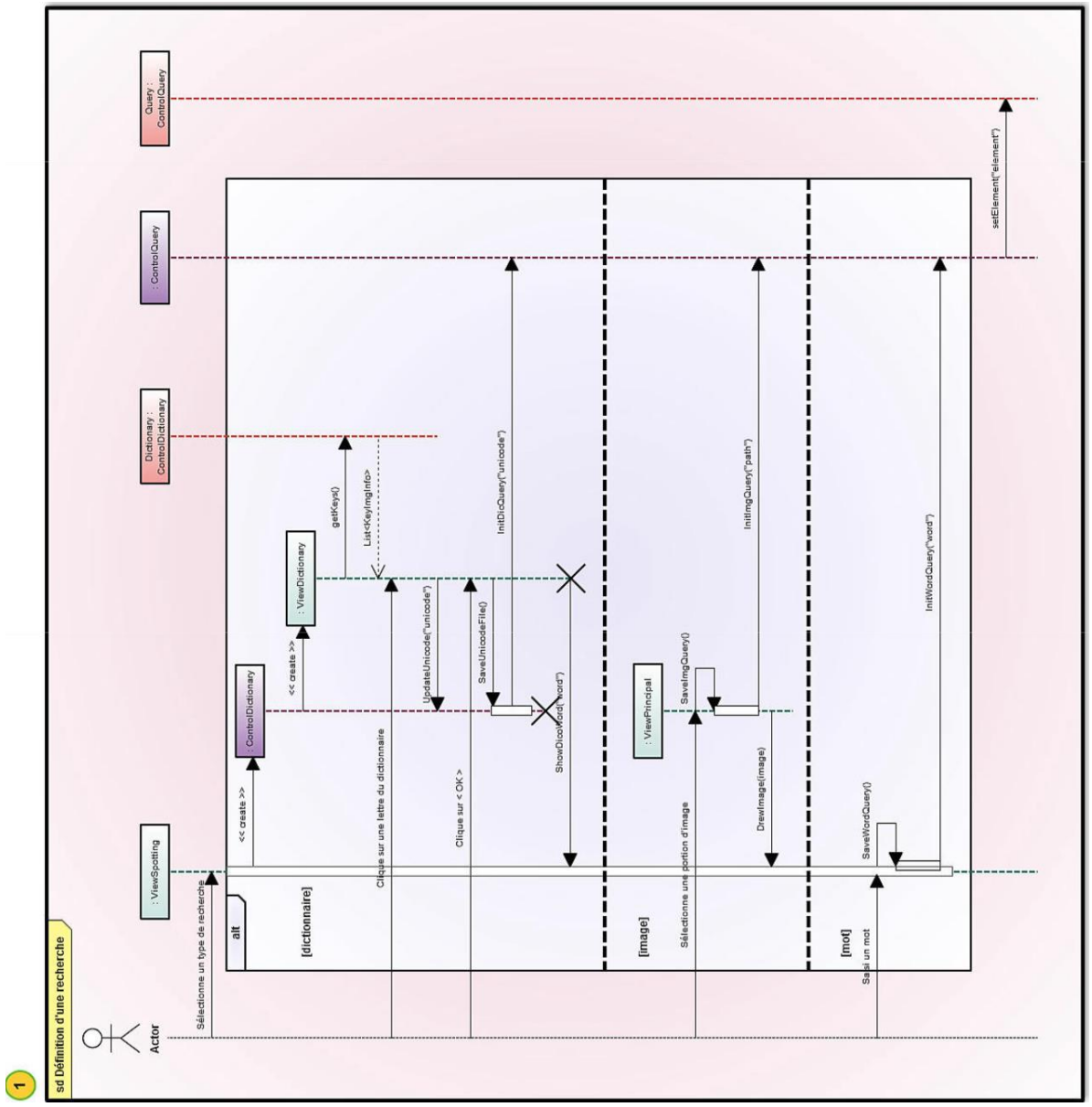


Figure 9 Diagramme de séquence spécifique sur la partie définition d'une recherche

Comptes rendus hebdomadaires

Compte rendu n°1 du 19/09/2016

1. J'ai pu prendre connaissance des documents fournis (spécifications, puis les rapports de projet).
2. J'ai testé l'application fournie par Monsieur Ragot
3. J'ai fait également quelques brèves recherches sur les outils à utiliser.
4. J'ai installé visual SVN afin de me faciliter l'utilisation de git dans visual.

Compte rendu n°2 du 26/09/2016

1. Prise en main et compréhension de l'application
2. Prise de notes sur les améliorations que j'imagine être pertinentes
3. Report de bugs de l'application existante

Compte rendu n°3 du 03/10/2016

1. Réorganisation du redmine du projet
2. Rédaction du cahier des spécifications

Compte rendu n°4 du 10/10/2016

1. Réunion avec Monsieur Ragot et les deux étudiants DI4 qui sont sur le projet
2. Création des différentes tâches sous redmine
3. Mise en place de la deuxième version du cahier des spécifications
4. Prise de connaissance des documents de recherches

Compte rendu n°5 du 17/10/2016

1. Étude de la solution divaService et mise en place des tests
2. Prise de connaissance du document DAS 2016

Compte rendu n°6 du 24/10/2016

1. Lecture d'un extrait de thèse pour la partie extraction de caractéristiques
2. Lecture du dossier de Vincent Bathellier et Etienne Gourinchas
3. Écriture du rapport pour la partie Gestion de projet. Création du diagramme de Gantt prévisionnel

Compte rendu n°7 du 31/10/2016

1. Utilisation des services diva. J'ai contacté une personne en charge du projet afin de répondre à mes questions. En parallèle j'ai également trouvé leur application sous forme d'application web.
2. Lecture des fichiers :
   — DAS2016
   — General architecture for sequence matching based word
   — DTW Variant
3. Rédaction de la partie gestion de projet

Compte rendu n°8 du 07/11/2016

1. Modification des spécifications et intégration au rapport final
2. Rédaction de la partie DAS2016 pour l'état de l'art
3. Rédaction de la partie DTW pour l'état de l'art
4. Réunion avec Quentin et Marie afin de les aider à comprendre le fonctionnement de l'application, la structure, corriger les problèmes qu'ils rencontraient et enfin les guider sur leur travaux sur l'application.

Compte rendu n°9 du 14/11/2016

1. Rédaction de la partie PDTW pour l'état de l'art
2. Rédaction de la partie DDTW pour l'état de l'art
3. Correction des spécifications suite à la réponse de Monsieur Ragot
4. Mise en place de la première version de mon rapport final

Compte rendu n°10 du 21/11/2016

1. Correction de l'état de l'art suite aux remarques de Monsieur Ragot
2. Rédaction rapport final S9

Compte rendu n°11 du 28/11/2016

1. Finalisation du rapport final S9
2. Réunion avec Quentin et Marie afin de corriger quelques erreurs bloquantes pour leur avancé sur le projet.

Compte rendu n°12 du 19/01/2017

1. Corriger les erreurs lié à l'appel de l'exécutable du projet Iwordspotting de Bastien
2. J'ai réussi à récupérer les résultats sous des fichier JSON
3. J'ai fait un petit projet qui permet de générer un fichier XML avec le fichier JSON en entrée
4. Je suis en train d'analyser le travail de Quentin et Marie afin de voir si il est pertinent de la garder ou non.

Compte rendu n°13 du 26/01/2017

1. Prendre en compte la totalité de l'image et ajout des balises lignes dans le XML
2. Évaluation du travail de Quentin et Marie
3. Étude des modifications de chemin
4. Mettre en place un moyen de découper les imagettes segmentées

Compte rendu n°14 du 02/02/2017

1. Lire le fichier XML afin de pouvoir créer le polygone à extraire (ligne segmenté) de l'image originale
2. Extraction de la ligne segmenté de l'image original

Compte rendu n°15 du 09/02/2017

1. Intégration de l'outil d'extraction de données
2. Finition de l'extraction de la ligne segmenté de l'image original

Compte rendu n°16 du 16/02/2017

1. Génération des fichiers CSV (3h)
2. Etude de la pertinence des caractéristique

Compte rendu n°17 du 02/03/2017

1. Commenter le code + documentation XML
2. Coloration en blanc des pixels vide
3. Génération des nouveau CSV de l'image 3
4. Correction de la taille des images des lignes segmentées

Compte rendu n°18 du 09/03/2017

1. Crop de l'imagette segmenté
2. Rédaction du rapport de gestion de projet
3. Correction de l'affichage des rectangle dans la plateforme
4. J'ai généré les caractéristiques pour chaque ligne segmentée avec la méthode Seam curving afin de pouvoir procéder au test avec la méthode de Yamin

Compte rendu n°19 du 16/03/2017

1. Correction de l'affichage des résultats : Le problème était lors du calcul de redimensionnement des rectangles à afficher
2. Finition du rapport de gestion de projet
3. Nettoyage du code et de sa structure
4. Test comparatif des méthodes de segmentions
5. Réflexion sur la méthode d'extraction de caractéristiques

Compte rendu n°20 du 23/03/2017

1. Re-modélisation du module de segmentation Seam Carving
2. J'ai également revu toute la documentation du projet et généré celle-ci grâce à Doxygen (présente sur le SVN)
3. Nettoyage du code et de sa structure
4. Proposition d'une méthode pour l'extraction de caractéristiques, plus prise en compte de vos commentaires

Compte rendu n°21 du 30/03/2017

1. Implémentation de la modélisation du module d'extraction des caractéristiques
2. Rendez vous avec monsieur Ramel pour l'évaluation de code
3. Nettoyage du code et de sa structure
4. Finition du rapport

Webographie

[1] *Corner Detection*. URL : https://en.wikipedia.org/wiki/Corner-detection.

[2] *Derivative Dynamic Time Warping*. URL : https://www.cs.rutgers.edu/~pazzani/Publications/sdm01.pdf.

[3] *DIVAServices Github*. URL : https://github.com/lunactic/DIVAServices.

[4] *DIVAServices-Spotlight*. URL : http://wuersch.pillo-srv.ch/.

[5] *Segmentation-based Historical Handwritten Word Spotting using Document-Specific Local Features*. URL : https://lib-repos.fun.ac.jp/dspace/bitstream/10445/3001/4/kterasaw_2009_01_icdar116.pdf.

Bibliographie

[1] Reid Porter Edward Rosten et Tom Drummond. « Faster and better : a machine learning approach to corner detection ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (2008), p. 105–119.

[2] Laureen Lambin. « PFE2014-RAPPORT191-LAMBIN-cahier-de-specifications ». Mém.de mast. Polytech, 2014-2015.

[3] Laureen Lambin. « PFE2014-RAPPORT191-LAMBIN-rapport-final ». Mém.de mast. Polytech, 2014-2015.

[4] Tanmoy Mondal, Jean-yves Ramel Nicolas Ragot et Umapada Pal. « Comparative Study of Conventional Time Series Matching Techniques for Word Spotting ». In : (2016).

[5] Edward Rosten et Tom Drummond. « Machine learning for high-speed corner detection ». In : *Computer Vision* 3951 (2006). Lecture Notes in Computer Science, p. 430–443.

[6] Nicolas Ragot Vikas Yadav. « Text extraction in document images : highlight on using corner points ». In : (2016).

[7] Yamin Zaidou. « RapportPRDDevZaidouYamin ». Mém.de mast. Polytech, 2015-2016.

[8] Zhang Zheng. « Cahier-de-spécification-v0.6-ZHANG-Zheng ». Mém.de mast. Polytech, 2013-2014.

Projet Recherche & Développement

Plateforme pour word spotting multi-scripts : Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés

Quentin Combemorel

Encadrement : Nicolas Ragot

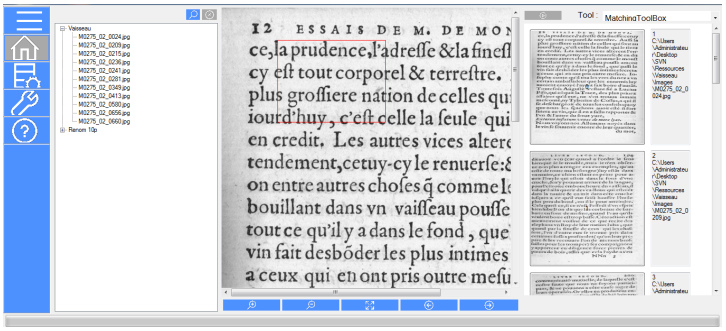
POLYTECH TOURS

En collaboration avec

Laboratoire informatique

Plateforme de WordSpotting

La plateforme de wordspotting va permettre à son utilisateur de rechercher un mot sur un document numérisé. Une fois la recherche effectué, les résultats sont affichés et classés en fonction de leur pertinence. L'utilisateur peut choisir ses bases d'images ainsi que ses algorithmes de recherche.

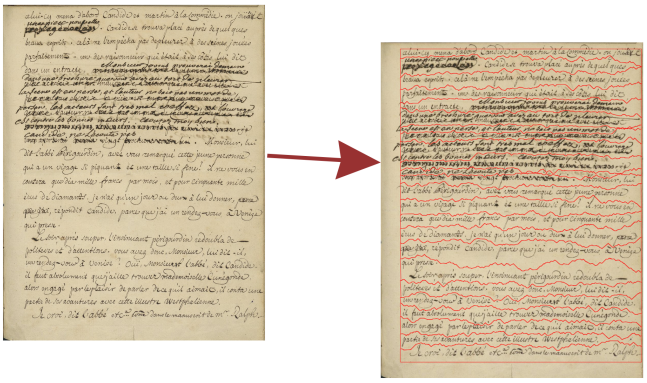


plateforme de wordSpotting

Objectifs

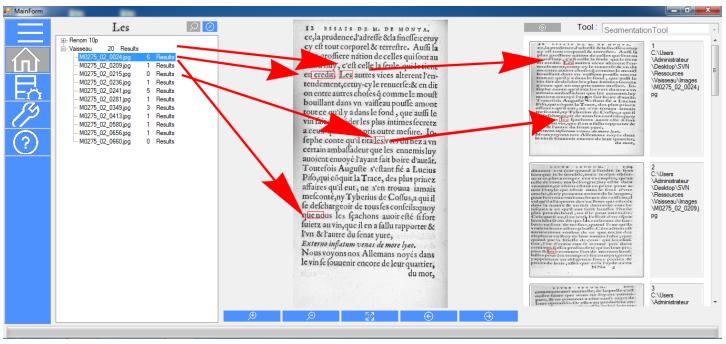
Améliorer la plateforme existante sous plusieurs angles:

1. Améliorer de la segmentation
2. Améliorer l'extraction de données
3. Visuellement, afin d'améliorer l'expérience utilisateur



Résultats attendus

Une plateforme beaucoup plus stable que l'existante. Cette dernière devra également plus performante en terme de pertinence des résultats et du temps d'exécution. L'amélioration générale de la plateforme permettra la recherche sur documents manuscrits numérisés



Ecole Polytechnique de l'Université François Rabelais de Tours

Département Informatique

64 avenue Jean Portalis, 37200 Tours, France

polytech.univ-tours.fr

POLYTECH TOURS

Informatique

Projet Recherche & Développement

Plateforme pour word spotting multi-scripts : Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés

Quentin Combemorel

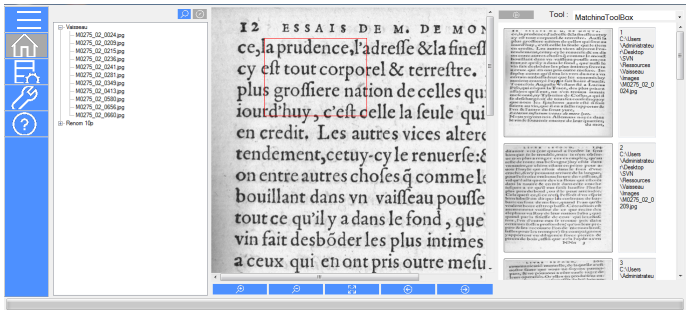
Encadrement : Nicolas Ragot

POLYTECH TOURS

En collaboration avec Laboratoire informatique

Plateforme de WordSpotting

La plateforme de wordspotting va permettre à son utilisateur de rechercher un mot sur un document numérisé. Une fois la recherche effectué, les résultats sont affichés et classés en fonction de leur pertinence. L'utilisateur peut choisir ses bases d'images ainsi que ses algorithmes de recherche.

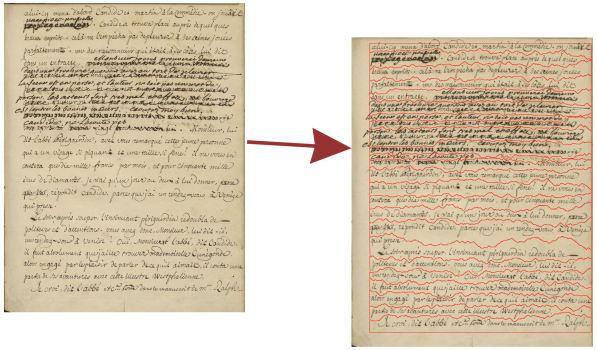


plateforme de wordSpotting

Objectifs

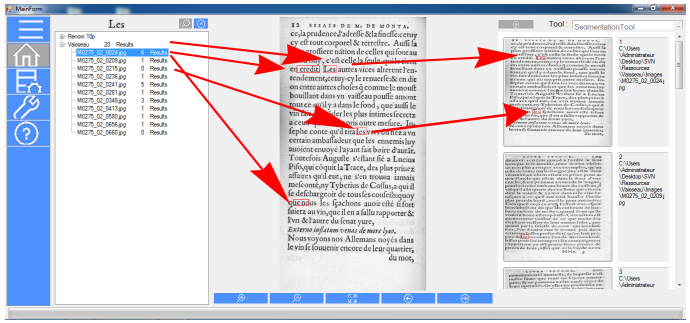
Améliorer la plateforme existante sous plusieurs angles:

1. Améliorer de la segmentation
2. Améliorer l'extraction de données
3. Visuellement, afin d'améliorer l'expérience utilisateur



Résultats attendus

Une plateforme beaucoup plus stable que l'existante. Cette dernière devra également plus performante en terme de pertinence des résultats et du temps d'exécution. L'amélioration générale de la plateforme permettra la recherche sur documents manuscrits numérisés



Ecole Polytechnique de l'Université François Rabelais de Tours
Département Informatique
64 avenue Jean Portalis, 37200 Tours, France
polytech.univ-tours.fr

POLYTECH TOURS
Informatique

Plateforme pour word spotting multi-scripts

Évaluation et amélioration d'un outil de recherche de mots "image" dans des carnets numérisés

Résumé

Le but du projet est d'évaluer et améliorer un outil permettant de faire des recherches dans des documents numérisés correspondant à des livres imprimés ou manuscrits pour lesquels on ne dispose pas de la transcription textuelle.

L'outil fonctionne en repérant les lignes de texte puis en essayant de trouver à l'intérieur de celles-ci des portions pouvant correspondre à une requête correspondant à l'image d'un mot recherché.

La première partie du PRD consistera à prendre en main l'outil existant et à le tester sur des images provenant d'un carnet de notes personnelles afin d'identifier ces limites et les choses à améliorer. Ensuite, un état de l'art sera effectué afin d'envisager des solutions d'amélioration. Ces dernières seront mises en place lors de la deuxième partie du PRD. Les développements se feront en C# et C++. Pendant tout le travail, un attention particulière sera portée sur la qualité du code et sa maintenabilité car plusieurs projets pourront se dérouler en parallèle sur le même logiciel.

Mots-clés

Recherche de mots, carnets numérisés, Segmentation, Extraction de caractéristiques

Abstract

The aim of this project is to test and to improve a software, which allow to make research in a scanned document (books, handwritten notes). The software works by locating the text lines, then trying to find within them sections that can correspond to a request corresponding to the image of the word researched. The first part of this project will consist taking control the existing software and test it with images from a personal notebook to indentify its limits and some points to improve. Afterwards a state of the art will be done to envisage some improvement solutions. These latter will be implemented in the second part of the project. These developpement will be done in C# and C++. During the whole work, a particular attention will be given to the quality of the code and his maintainability, because several projects can run in parallel on the same software.

Keywords

word-spotting, book, scanned, Segmentation, Features extraction

Entreprise
Laboratoire informatique

POLYTECH TOURS

Tuteur entreprise

Étudiant
Quentin Combemorel (DI5)

Tuteur académique
Nicolas Ragot