



Ecole Polytechnique de l'Université 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
2018-2019

Amélioration d'un outil de word spotting

Tuteur académique
Nicolas RAGOT

Étudiant
Dorian LE THAI BINH (DI5)

1^{er} avril 2019



Liste des intervenants

Nom	Email	Qualité
Dorian LE THAI BINH	dorian.lethaibinh@etu.univ-tours.fr	Étudiant DI5
Nicolas RAGOT	nicolas.ragot@univ-tours.fr	Tuteur académique, Département Informatique



Avertissement

Ce document a été rédigé par Dorian LE THAI BINH susnommé l'auteur.

L'Ecole Polytechnique de l'Université 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 reconnaît 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 reconnaît 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

Dorian LE THAI BINH, *Amélioration d'un outil de word spotting*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2018-2019.

```
@mastersthesis{
  author={LE THAI BINH, Dorian},
  title={Amélioration d'un outil de word spotting},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université de Tours},
  address={Tours, France},
  year={2018-2019}
}
```


Table des matières

Liste des intervenants	a
Avertissement	b
Pour citer ce document	c
Table des matières	i
1 Introduction	1
1 Acteurs, enjeux et contexte	1
1.1 Acteurs.....	1
1.2 Contexte.....	1
2 Objectifs.....	2
3 Hypothèses	2
4 Bases méthodologiques.....	3
2 Description générale	4
1 Environnement du projet	4
2 Caractéristiques des utilisateurs.....	5
3 Fonctionnalités du système	6
3 État de l'art	8
1 WordSpotting	8
1.1 Prétraitements.....	8
1.1.1 Binarisation.....	9
1.1.2 Normalisation.....	9
1.1.3 Segmentation.....	9

1.2	Techniques.....	10
1.2.1	SIFT	11
1.2.2	HOG	12
1.2.3	Bag of Visual Words.....	12
2	Reranking.....	13
2.0.1	Example-based reranking.....	16
2.0.2	Interactive reranking	18
4	Analyse et conception	19
1	Analyse de l'existant.....	19
2	Stabilisation de la plateforme.....	21
2.1	Problèmes liés aux outils de recherche.....	21
3	Diva Services	22
3.1	Présentation de DivaServices	22
3.2	Utilisation de DivaServices.....	22
5	Mise en oeuvre	26
1	Stabilisation de la plateforme.....	26
1.1	Correction des chemins	26
1.2	Affichage des rectangles	26
1.3	Plantages.....	27
1.4	Problème base image Renom P1	28
2	Application DivaSegmentation	29
2.1	Modélisation.....	30
2.2	Fonctionnement	30
6	Bilan et conclusion S9	32
7	Bilan et conclusion S10	33
	Annexes	34
A	Description des interfaces externes du logiciel	35
1	Interface homme - machine	35
B	Spécifications fonctionnelles	36
1	Stabilisation de la plateforme.....	36
2	Intégration de l'outil de segmentation de Quentin Combemorel	36
3	Arrêt de la recherche	37
4	Migration de SVN à GitLab.....	37
5	Visualisation interactive des résultats (reranking)	37

6	Enregistrement et chargement des résultats.....	37
7	Confort d'usage, mise en pause, reprise de la recherche	38
8	Filtrage des résultats avec seuil.....	38
9	Mise en place d'un processus d'indexation pour accélérer la recherche	38
C	Spécifications non fonctionnelles	40
1	Contraintes de développement et conception	40
2	Capacités.....	40
3	Controllabilité.....	40
D	Gestion de projet	41
E	Guide d'installation plateforme	43
1	Prérequis	43
2	Récupération du projet	43
3	Configuration du projet	43
3.1	Outil binarization	43
3.1.1	Configuration de Visual Studio	43
3.2	Plateforme Wordspotting.....	44
3.2.1	Configuration Visual Studio.....	44
3.2.2	Configuration externe	44
F	Guide d'installation / utilisation	45
1	Prérequis	45
2	Installation	45
3	Utilisation	45
G	Comptes rendus hebdomadaires	47

1

Introduction

1 Acteurs, enjeux et contexte

1.1 Acteurs

Ce sujet est une reprise d'anciens PRD réalisés successivement par Zheng Zhan (Zheng, 2013), Dawei Shen, Loreen Lambin (Lambin, Rapport de PFE, 2015) [**RapportLoreen**], Quentin Combemorel [**RapportQuentin**], Yamin Zaidou (Zaidou, 2016), Alafate ABULIMITI et Yuanyuan LI (Mini projet, 2017), Ce projet est encadré par Monsieur Ragot (MOA), enseignant chercheur au laboratoire d'Informatique Fondamentale et Appliquée de Tours (LIFAT EA 6300).

Mon rôle dans ce projet est la MOE accompagné de Monsieur Ragot.

1.2 Contexte

Le wordspotting est la science de la reconnaissance des mots. On utilise ses méthodes afin de retrouver des informations dans des documents au format image.

Fin années 1980, des tentatives de numérisation du patrimoine ont été faites. Comme par exemple le projet de numérisation de la Bibliothèque nationale de France. Cette dernière a lancé en 1997 le site Gallica, qui permet l'accès à plusieurs dizaines de milliers de documents (livres, périodiques, images, sons) peu connus ou indisponibles.

Les récents progrès ont permis de développer les techniques de reconnaissance optique de caractères (OCR) permettant de passer du mode image au mode texte. Il y a également une amélioration dans les performances de formats comme PDF (portable document format, Acrobat d'Adobe). De nos jours, nous avons la possibilité de diffuser en réseau ces images avec des temps d'accès rapides.

L'OCR doit trouver les caractères puis valider par reconnaissance lexicale les mots qui les contiennent. Cette tâche est loin d'être simple car si l'OCR doit apprendre à distinguer la forme de chaque caractère, il doit en plus être capable de gérer polices ou les écritures différentes au sein d'un même document. Il y a donc encore beaucoup d'erreurs.

Mais le wordspotting est différent de l'OCR. En effet, le wordspotting se base principalement sur des requêtes. On peut trouver les requêtes QbE (Query by Example) et QbS (Query by String).

Pour ce qui est du Query by Example, on va sélectionner une partie de l'image contenant du texte pour la comparer à d'autres images afin de trouver des correspondances. La plateforme de wordspotting actuelle fonctionne de cette manière. Le Query by String permet d'utiliser des chaînes de caractères pour ensuite les retrouver dans des images contenant du texte.

Avant une recherche, le plus souvent, les images vont être binarisées puis segmentées. Binariser une image revient à séparer l'image en deux classes : le fond et l'objet. La binarisation intervient généralement avant la segmentation. La segmentation, elle, permet d'isoler les éléments textuels, mots et caractères. Elle se base sur des mesures de lignes, d'espaces pour faire la séparation. Cela permet donc de réduire le champ d'analyse, d'améliorer les performances et de diminuer le nombre d'erreurs. Cependant cela marche correctement dans le cas où la séparation est évidente.

Afin de trouver des informations dans des ouvrages anciens, collections numériques comme Gallica. L'objectif serait de pouvoir extraire les mots des images numérisées contenant du texte pour ensuite pouvoir faire des recherches.

Le LIFAT travaille sur le sujet du wordspotting depuis plus de 5 ans et le travail ce projet de recherche et développement dont j'ai la charge porte sur l'amélioration de la plateforme.

Aujourd'hui il existe donc une application qui permet la recherche de mots ou de caractères sur une image mais également l'affichage de résultats. On peut rechercher des mots soit par découpage d'image, ou par dictionnaire. Deux algorithmes ont été développés et sont fonctionnels (iWordSpotting et Renom). L'application a été modifiée en 2016 par Quentin Combemorel [**RapportQuentin**] et a développé un nouvel algorithme de segmentation qui n'a pas été inclut dans la plateforme.

2 Objectifs

L'objectif principal du projet, est l'amélioration et l'enrichissement de la plateforme existante. Un découpage des tâches a été fait et constitue une liste non priorisée et non exhaustive. Il faut également préciser que toutes ces tâches ne seront pas forcément réalisées au cours de ce projet de recherche et développement.

- S'assurer du bon fonctionnement de la plateforme (hors algorithmes), notamment visionnage des résultats.
- Améliorer le confort d'usage : Affichage des résultats en temps réel, pause, enregistrement, ergonomie.
- Améliorer la documentation, pour l'installation, le développement, la maintenance et l'utilisation.
- Migrer le gestionnaire de version de SVN à Git (Git Lab).
- Améliorer les algorithmes scientifiques (Amélioration de la mise en correspondance, segmentation, intégration de la méthode de Quentin [**RapportQuentin**], segmentation alternative, amélioration des caractéristiques et amélioration matching tool box.
- Travailler sur une méthode de spotting alternative à partir de deep learning.
- Visualisation interactive des résultats (reranking) recalculer les taux de pertinence en fonction des choix des utilisateurs.
- Mise en place d'un processus d'indexation pour accélérer la recherche.

3 Hypothèses

Si la partie stabilisation de la plateforme ne se déroule pas correctement, il faudra alors revoir les priorités des objectifs.

4 Bases méthodologiques

Pour la gestion de projet le serveur de version SVN est utilisé. On peut y trouver des outils de gestion de projet via Redmine. Des rapports hebdomadaires concernant l'avancement du projet sont également communiqués à M. Ragot.

Afin de gérer le projet j'ai décidé de partir sur une méthode Agile. En effet il y a la possibilité de découper le projet en plusieurs sprints basés sur les objectifs définis. De plus comme l'application existe déjà, il sera plus simple de mettre en place des prototypes que je vais pouvoir montrer au client pour obtenir un retour et modifier l'application en fonction. Enfin, un cycle en V ou en cascade semblait un peu trop risqué compte tenu de la partie critique de débogage de l'application.

J'ai mis en place un Trello pour que mon encadrant/client puisse consulter l'état d'avancement du projet.

Enfin, pour la planification, j'ai mis en place deux diagrammes de Gantt. Un détaillé pour le semestre 9, et un plus large pour le semestre 10.

2

Description générale

La plateforme wordspotting est une application permettant de réunir plusieurs outils de recherche indépendants afin d'effectuer de la recherche textuelle sur des documents numérisés.

Plus précisément, elle permet à l'aide de son interface utilisateur :

- D'ajouter/supprimer des outils des outils de recherche
- D'ajouter/supprimer des collections de documents numérisées
- De fournir des données d'entrées aux outils de recherche et de les appeler
- De visualiser les résultats retournés par les outils de recherche

1 Environnement du projet

L'environnement du projet est composé de plusieurs briques logicielles chacune étant le résultat d'un projet de recherche et développement d'anciens élèves de l'école.

Le premier projet se nomme « Binarisation », c'est le projet qui permet de binariser les images. Cette partie est décrite dans le rapport de PRD de Yamin Zaidou. Il a été développé en C++ et utilise la librairie Open CV.

La binarisation consiste à convertir une image en deux classes. Pour se faire, l'outil va prendre la couleur de chaque pixel de l'image et la transformer en noir si le niveau de gris du pixel est plus proche du noir que du blanc. Cela fonctionne dans l'autre sens pour le blanc. Dans notre cas, on obtiendra donc des lettres noires et un fond blanc.

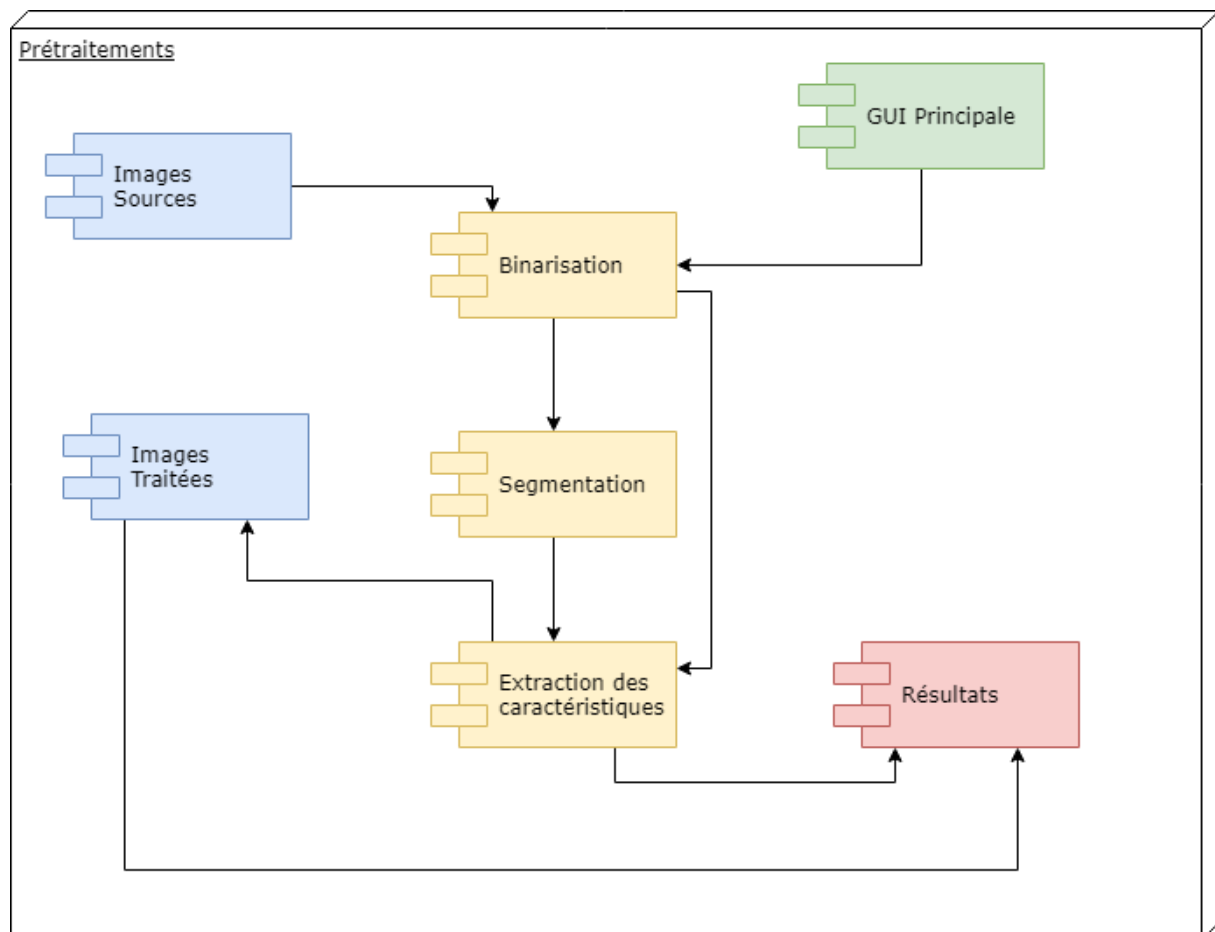
Quentin Combemorel [**RapportQuentin**] a mis en place un outil de segmentation pour remplacer celui utilisé via une API nommée DivaServices qui est une application développée par le groupe « Diva » à l'université de Fribourg. Cette application est un webService qui permet d'utiliser des algorithmes d'analyse d'images.

La segmentation est ce qui suit la binarisation. Segmenter veut dire découper une image en plusieurs classes. Très souvent, on va segmenter sur une image binarisée pour rendre le découpage plus précis, et limiter le nombre de classes.

Ensuite intervient l'extraction de caractéristiques d'une image. Ces caractéristiques permettent de décrire des lignes de texte utilisées pour comparer les distances entre deux images avec du texte pour savoir si elle sont plus ou moins proches.

L'outil de iWordSpotting développé par Bastien Meunier est une méthode de mise en correspondance de séquences permettant de calculer la distance entre deux vecteurs de caractéristiques, appelés séquences, passées en paramètre. Cette distance détermine la ressemblance des deux séquences, plus elle est petite et plus les séquences sont semblables. Les deux séquences sont décrites dans un fichier au format CSV définies par Abdourahman ADEN HASSAN dans son rapport. Cet outil a été développé en C++.

Enfin, tous ces outils sont utilisés par la plateforme WordSpotting créée par Zhang Zheng en 2012, repris par Loreen [**RapportLoreen**] Lambin et enfin par Yamin Zaidou. Application développée en C.



Note : La segmentation n'est pas incluse dans l'application, elle peut être utilisée via l'API DivaServices. Elle devra être remplacée par l'outil de Quentin Combemorel [**RapportQuentin**].

2 Caractéristiques des utilisateurs

Les utilisateurs doivent pouvoir utiliser l'application sans avoir de connaissances avancées en informatique.

On peut trouver plusieurs types d'utilisateurs :

- L'utilisateur standard qui est peut-être par exemple un historien qui va uniquement utiliser l'application pour la consultation et la visualisation des résultats. Cependant, il ne pourra pas utiliser l'application si les outils et les collections numérisées n'ont pas été intégrés par le manager au préalable.

- Le manager, va s'occuper de la partie configuration comme l'importation des collections numérisées, les prétraitements qui seront exécutés et les algorithmes de recherche.

3 Fonctionnalités du système

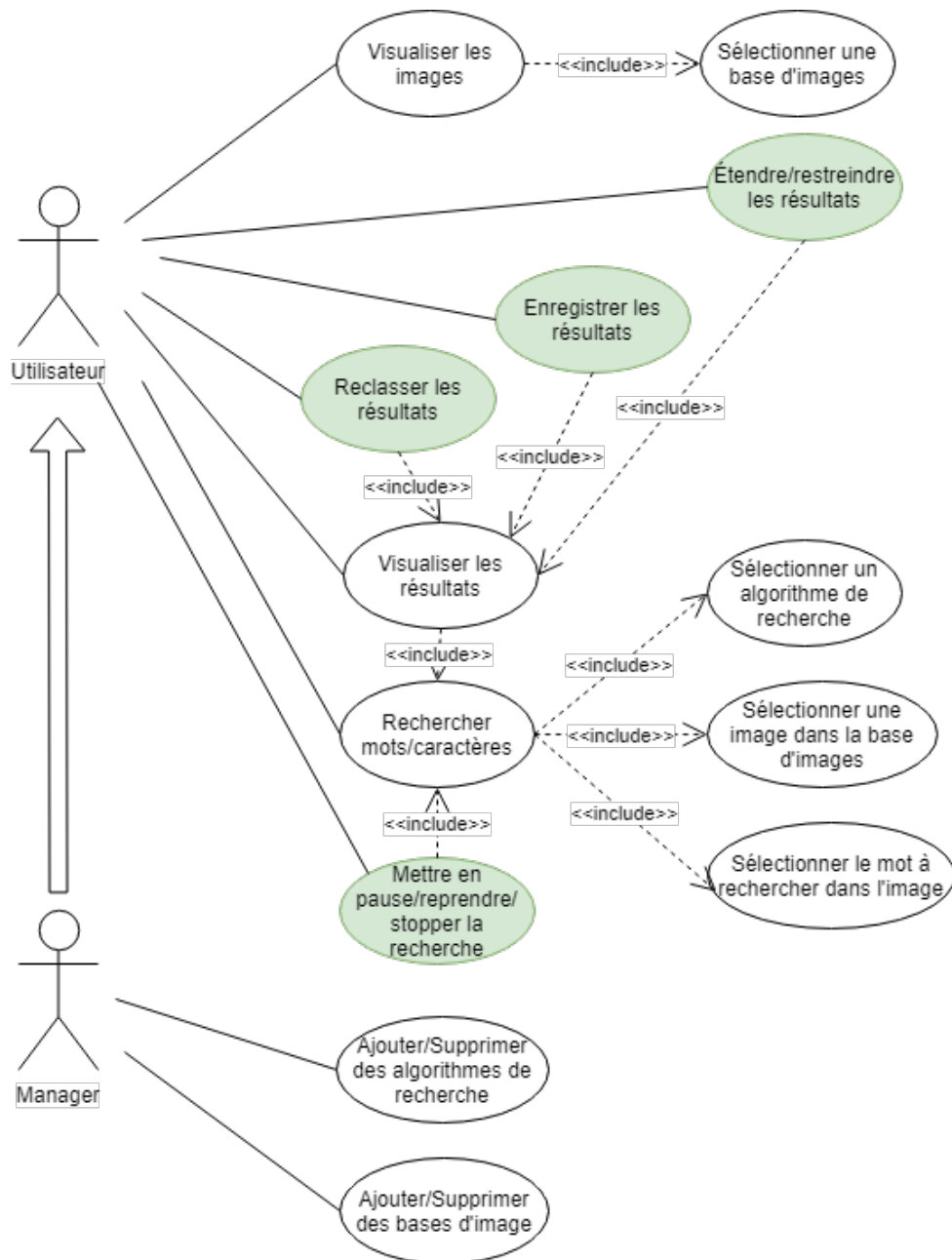
Comme précisé plus tôt, l'application est déjà existante et les outils de recherches sont également terminés. Parmi les fonctionnalités on peut trouver l'importation ou la suppression des collections numérisées ou outils de recherche.

Pour la partie recherche, il faut au préalable choisir une image d'une base importée dans l'application, sélectionner une zone correspondant à un mot que l'on souhaite rechercher.

Il peut y avoir trois types de requêtes :

- Image : correspond à la zone que l'on trace avec le curseur pour définir l'élément à rechercher.
- Dictionnaire : La recherche par dictionnaire permet de composer un mot à rechercher sans passer par la recherche par image.
- Mots : Non repris dans les dernières versions de la plateforme.

Enfin, il faut sélectionner une méthode de recherche qui a été 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.



Les nouvelles fonctionnalités sont en vert, et celles déjà présentes en blanc.

3

État de l'art

Dans cette partie je vais présenter toutes les recherches que j'ai pu effectuer sur les sujets du wordspotting, du reranking. Celles-ci ont été guidées par Monsieur Ragot. Le but de ces recherches, est de pouvoir étudier les différentes techniques de spotting, de reranking et d'indexation.

1 WordSpotting

Le but du Wordspotting est de récupérer des images de mots à partir d'une collection d'images. On peut les classer en fonction de certains critères, et les trier par leur pertinence par rapport à une certaine requête. Ces requêtes peuvent être soit une image de mot que l'utilisateur aura extrait d'une page d'un document ou en définissant une chaîne de mots qui doit être récupérée.

Les requêtes que l'on peut trouver sont :

- QbE : Les requêtes sont données sous forme d'images de mots et la récupération est basée sur une comparaison de distance. Pour effectuer une telle comparaison, un mapping doit être mis en place.
- QbS : Les requêtes sont données sous forme de chaînes de mots. Contrairement à QbE, l'utilisateur n'est pas obligé de rechercher un exemple visuel d'un mot dans une collection de documents.

Le problème principal dans la recherche de caractères est la différence d'écritures possibles et le positionnement des caractères. Le cas le plus simple est la reconnaissance de textes écrits via outil informatique car chaque caractère est bien espacé et les lignes sont bien découpées. Au contraire, lorsqu'on s'intéresse à la reconnaissance de caractères sur des images numérisées d'ouvrages manuscrits, tout se complique. Le style d'écriture n'est pas le même, les lignes ne sont pas droites, les lettres sont collées et il peut y avoir des ratures ou autre problèmes pouvant perturber la reconnaissance.

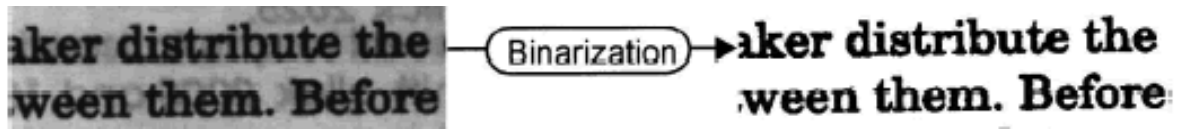
Mes recherches ont porté essentiellement sur l'article "A survey of document image word spotting techniques" [**WordSpotting**]

1.1 Prétraitements

Certaines techniques nécessite une segmentation en ligne ou en mots. Dans ces situations là, des prétraitements sont alors nécessaires.

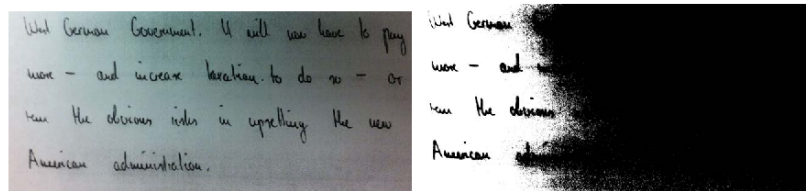
1.1.1 Binarisation

C'est la séparation de l'image en deux classes : le fond et la forme. Cela permet de rendre la segmentation plus simple pour les images contenant du texte.



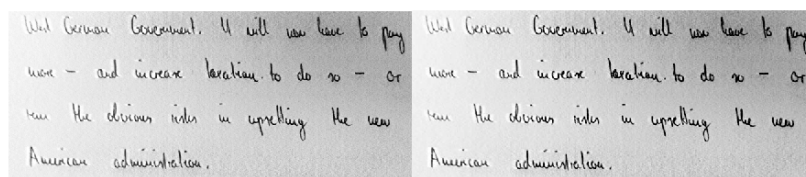
1.1.2 Normalisation

La normalisation permet de transformer une image pour qu'elle soit plus facilement exploitable. Si on utilise une image avec des ombres, la binarisation risque de transformer l'ombre en "tâche" noire et donc fausser le processus.



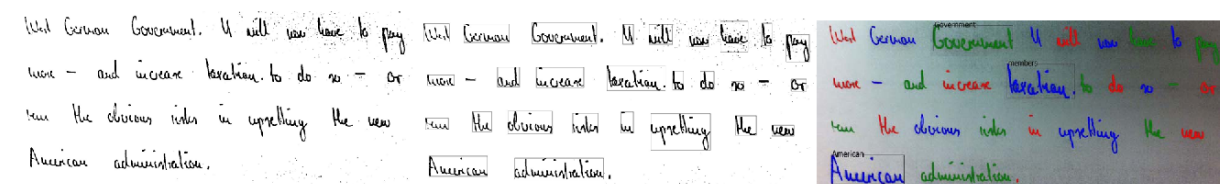
(a) Original Image

(b) Binarized Image Using Otsu



(c) Background Normalized Image

(d) Histogram Normalized Image



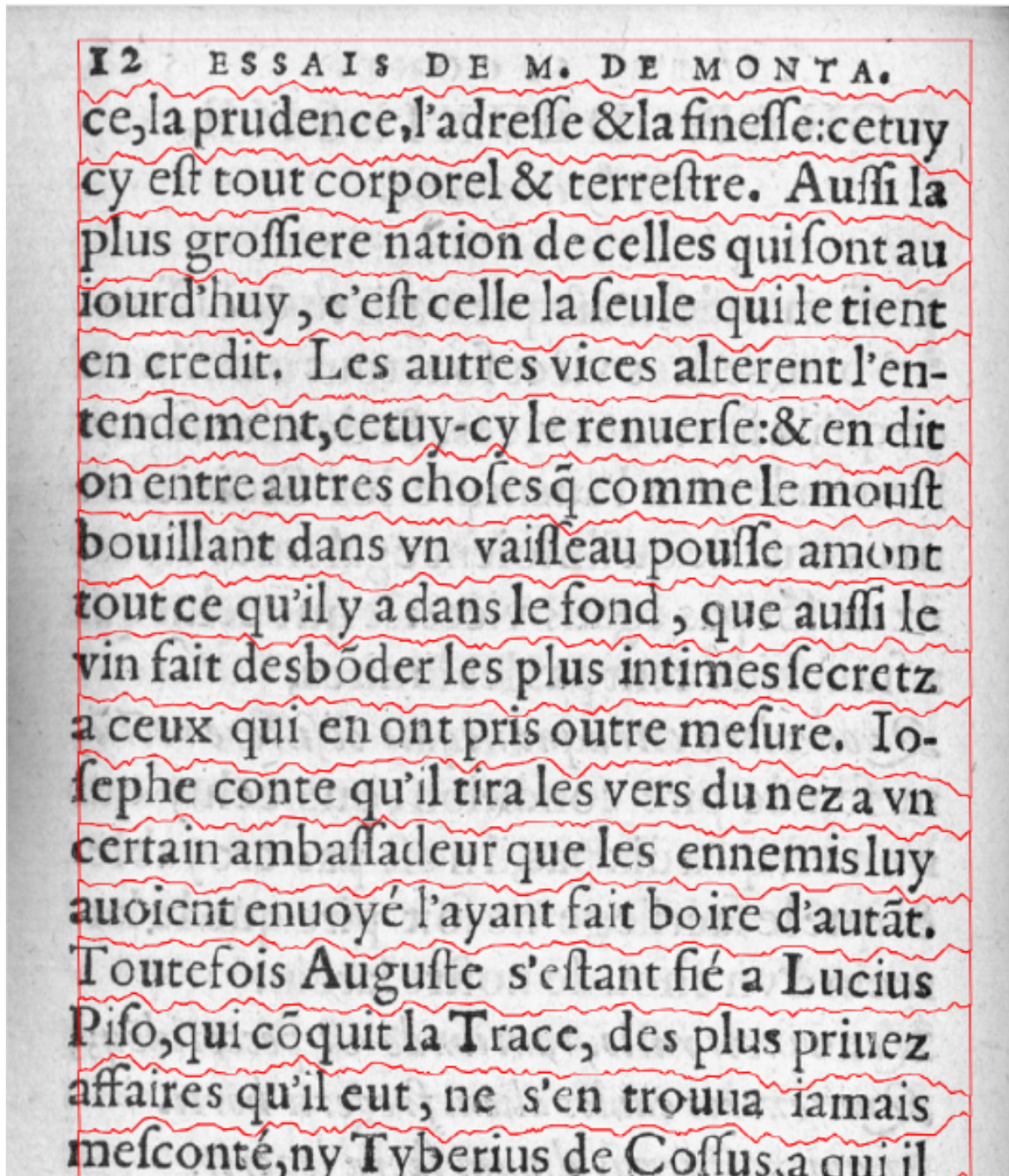
(e) Final Binarized Image after c) d) and Adaptive Binarization

(f) Word Segmentation Results

(g) Spotting

1.1.3 Segmentation

Elle se base sur des mesures de lignes, d'espaces pour faire la séparation. Cela permet donc de réduire le champ d'analyse, d'améliorer les performances et de diminuer le nombre d'erreurs. Cependant cela marche correctement dans le cas où la séparation est évidente.



1.2 Techniques

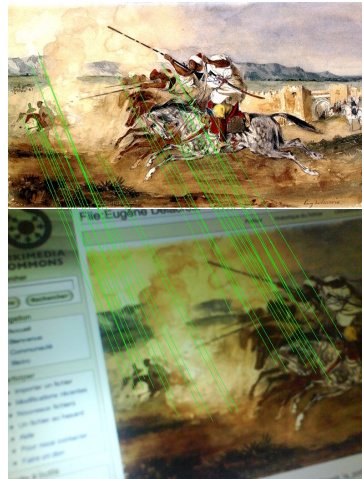
Dans cette partie je vais parler des techniques de wordspotting existantes. Une technique se caractérise par l'utilisation unique de plusieurs algorithmes existants par exemple SIFT, Bag of Word et une comparaison par Cosinus et distance euclidienne.

On peut trouver aussi des méthodes par apprentissage qui sont en général plus performantes cependant une grande base d'apprentissage est nécessaire pour un fonctionnement optimal.

1.2.1 SIFT

La scale-invariant feature transform (SIFT), que l'on peut traduire par « transformation de caractéristiques visuelles invariante à l'échelle », est un algorithme utilisé dans le domaine de la vision par ordinateur pour détecter et identifier les éléments similaires entre différentes images numériques (éléments de paysages, objets, personnes, etc.). Il a été développé en 1999 par le chercheur David Lowe.

L'étape fondamentale de la méthode proposée par Lowe consiste à calculer ce que l'on appelle les « descripteurs SIFT » des images à étudier. Il s'agit d'informations numériques dérivées de l'analyse locale d'une image et qui caractérisent le contenu visuel de cette image de la façon la plus indépendante possible de l'échelle (« zoom » et résolution du capteur), du cadrage, de l'angle d'observation et de l'exposition (luminosité). Ainsi, deux photographies d'un même objet auront toutes les chances d'avoir des descripteurs SIFT similaires, et ceci d'autant plus si les instants de prise de vue et les angles de vue sont proches. D'un autre côté, deux photographies de sujets très différents produiront selon toute vraisemblance des descripteurs SIFT très différents eux aussi (pouvoir discriminant). Cette robustesse, vérifiée dans la pratique, est une exigence fondamentale de la plupart des applications et explique en grande partie la popularité de la méthode SIFT.



Informations tirées de wikipedia [SIFT]

Les avantages de cet algorithme pour le wordspotting sont que dans des images de textes manuscrits, il va pouvoir quand même retrouver les mots même si ils sont écrits avec un angle différent ou alors que le texte sur l'image est tourné.

SIFT est utilisé dans plusieurs méthodes de wordspotting de la littérature.

1.2.2 HOG

Un autre algorithme souvent utilisé dans les techniques de wordspotting est l'HOG.

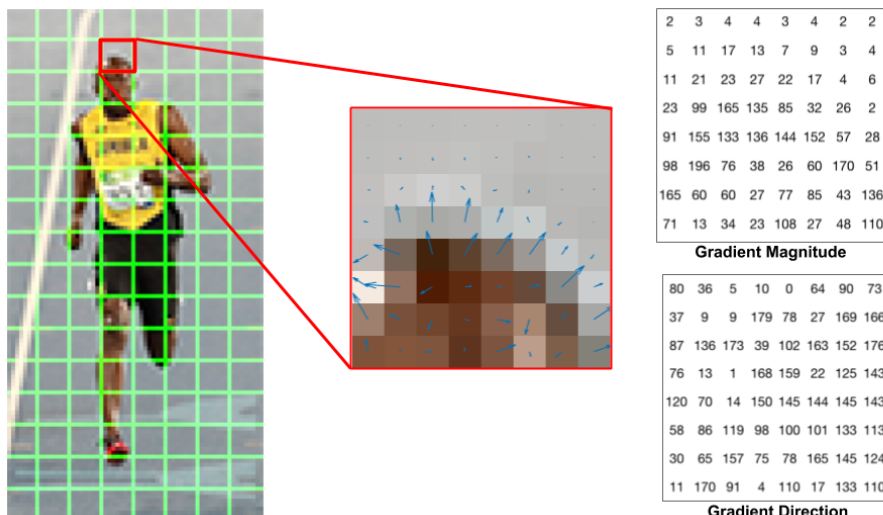
Un histogramme de gradient orienté est une caractéristique utilisée en vision par ordinateur pour la détection d'objet. La technique calcule des histogrammes locaux de l'orientation du gradient sur une grille dense, c'est-à-dire sur des zones régulièrement réparties sur l'image. Elle possède des points communs avec les SIFT, les Shape contexts et les histogrammes d'orientation de contours, mais s'en diffère notamment par l'utilisation d'une grille dense. La méthode s'est montrée particulièrement efficace pour la détection de personnes.

Les HOG ont été proposés par Navneet Dalal et Bill Triggs, chercheurs à l'INRIA de Grenoble, à la conférence CVPR de juin 2005.

L'idée importante derrière le descripteur HOG est que l'apparence et la forme locale d'un objet dans une image peuvent être décrites par la distribution de l'intensité du gradient ou la direction des contours. Ceci peut être fait en divisant l'image en des régions adjacentes de petite taille, appelées cellules, et en calculant pour chaque cellule l'histogramme des directions du gradient ou des orientations des contours pour les pixels à l'intérieur de cette cellule. La combinaison des histogrammes forme alors le descripteur HOG. Pour de meilleurs résultats, les histogrammes locaux sont normalisés en contraste, en calculant une mesure de l'intensité sur des zones plus larges que les cellules, appelées des blocs, et en utilisant cette valeur pour normaliser toutes les cellules du bloc. Cette normalisation permet une meilleure résistance aux changements d'illuminations et aux ombres.

Informations tirées [HOG]

Les avantages de cet algorithme pour le wordspotting sont que le gradient de couleurs entre les caractères et le fond est très contrasté, ce qui va rendre la matrice des directions plus précise et donc rendre la détection de contours plus simple.

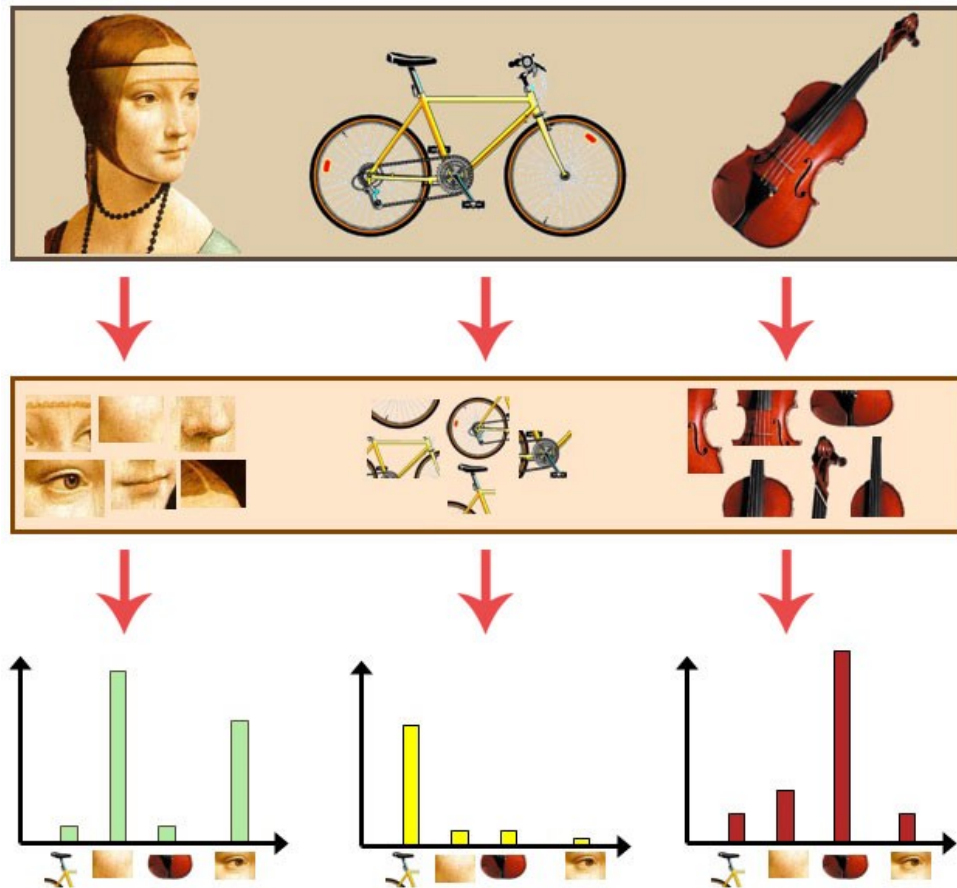


1.2.3 Bag of Visual Words

En vision par ordinateur, un sac de mots visuels est un vecteur de comptages d'occurrences d'un vocabulaire de caractéristiques d'image locales.

Fonctionnement des sacs de mots visuels :

- Extraire les points d'intérêt des images avec SIFT par exemple.
- Créer un dictionnaire visuel en partitionnant les points d'intérêt.
- Pour une image, il faut vérifier dans quelle partition est chaque point d'intérêt. Un histogramme à 1 000 bins est alors construit, où chaque bin correspond à une partition. La valeur d'un bin est égale au nombre de point d'intérêt de l'image qui sont la partition correspondante.
- Chaque image est décrite par un vecteur, la classification peut être faite par un algorithme de classification supervisé tel que les SVM.

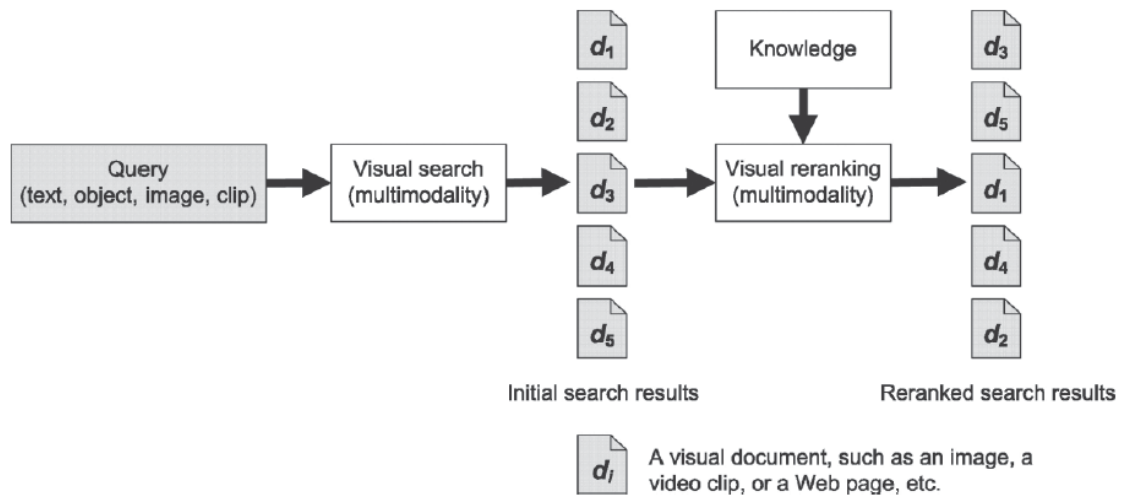


2 Reranking

L'état de l'art sur le reranking se base sur l'article [Reranking]

Pour résoudre les problèmes des approches de recherche visuelle, le reranking a attiré une attention croissante ces dernières années.

Le reranking consiste à reclasser les résultats en fonction de plusieurs paramètres pour affiner la recherche.

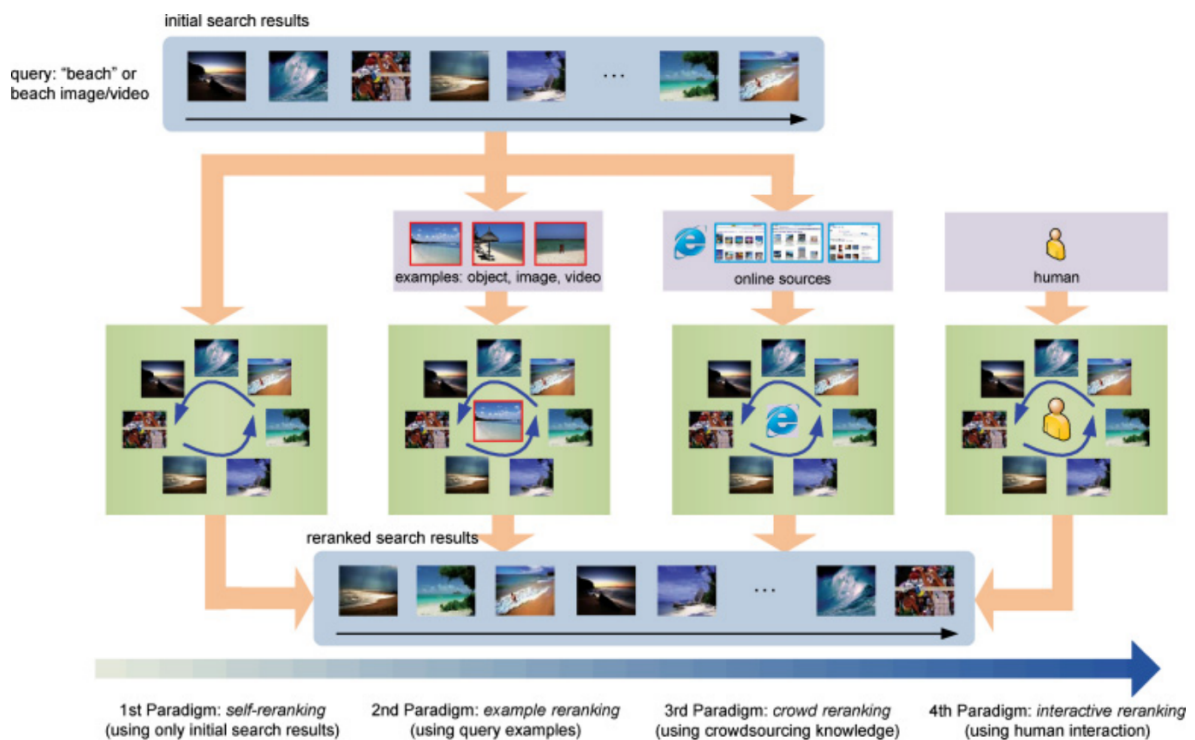
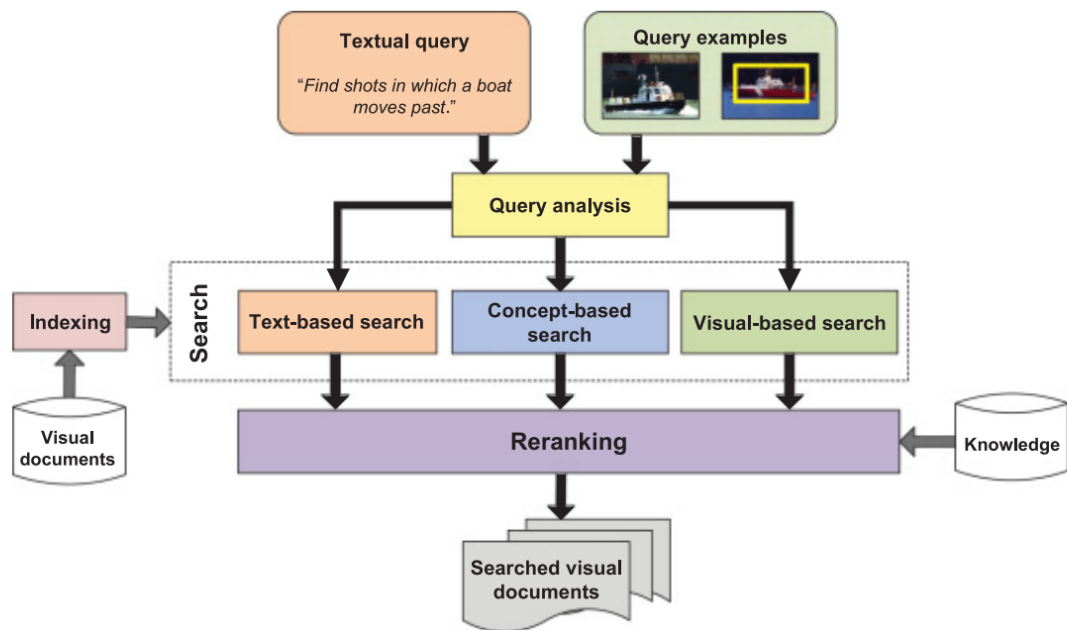


Voici les problèmes liés à la recherche actuellement :

- Performance de recherche initiale insatisfaisante. Les premiers résultats de recherche contiennent généralement une petite partie des documents pertinents.
- Manque de connaissances disponibles ou de contexte pour le reclassement. Bien que nous puissions concevoir des interfaces de recherche spécifiques pour permettre aux utilisateurs de mieux formuler leurs requêtes ou de collecter des informations démographiques (nom, intérêt, emplacement, par exemple).
- Jeu de données à grande échelle. La plupart des approches existantes ne sont pas extensibles pour une grande échelle ensemble de données en raison des scalabilités algorithmiques et du temps de réponse. En conséquence, seuls les documents retournés en haut sont généralement pris en compte dans le processus de reclassement.

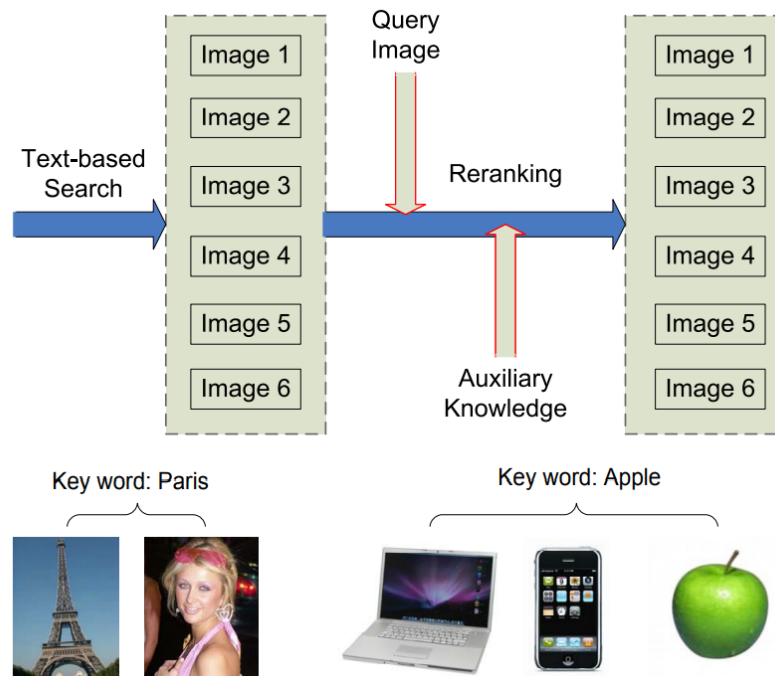
La recherche sur le reranking visuel s'est poursuivie selon quatre paradigmes pour l'exploitation des informations pertinentes :

- Selfreranking. Qui utilise uniquement les résultats de la recherche initiale.
- Example-based reranking. Qui exploite les exemples de requêtes fournis par l'utilisateur.
- Crowd reranking. Qui explore les connaissances disponibles en matière de crowdsourcing sur Internet (par exemple, les multiples moteurs ou sites de recherche d'images et de vidéos, l'encyclopédie en ligne fournie par l'utilisateur telle que Wikipedia).
- Interactive reranking. Qui implique une interaction de l'utilisateur pour guider le processus de reranking.



Les méthodes qui nous intéressent pour le projet sont l'**exemple-based reranking** et l'**interactive reranking** car l'utilisateur peut jouer un rôle.

2.0.1 Example-based reranking



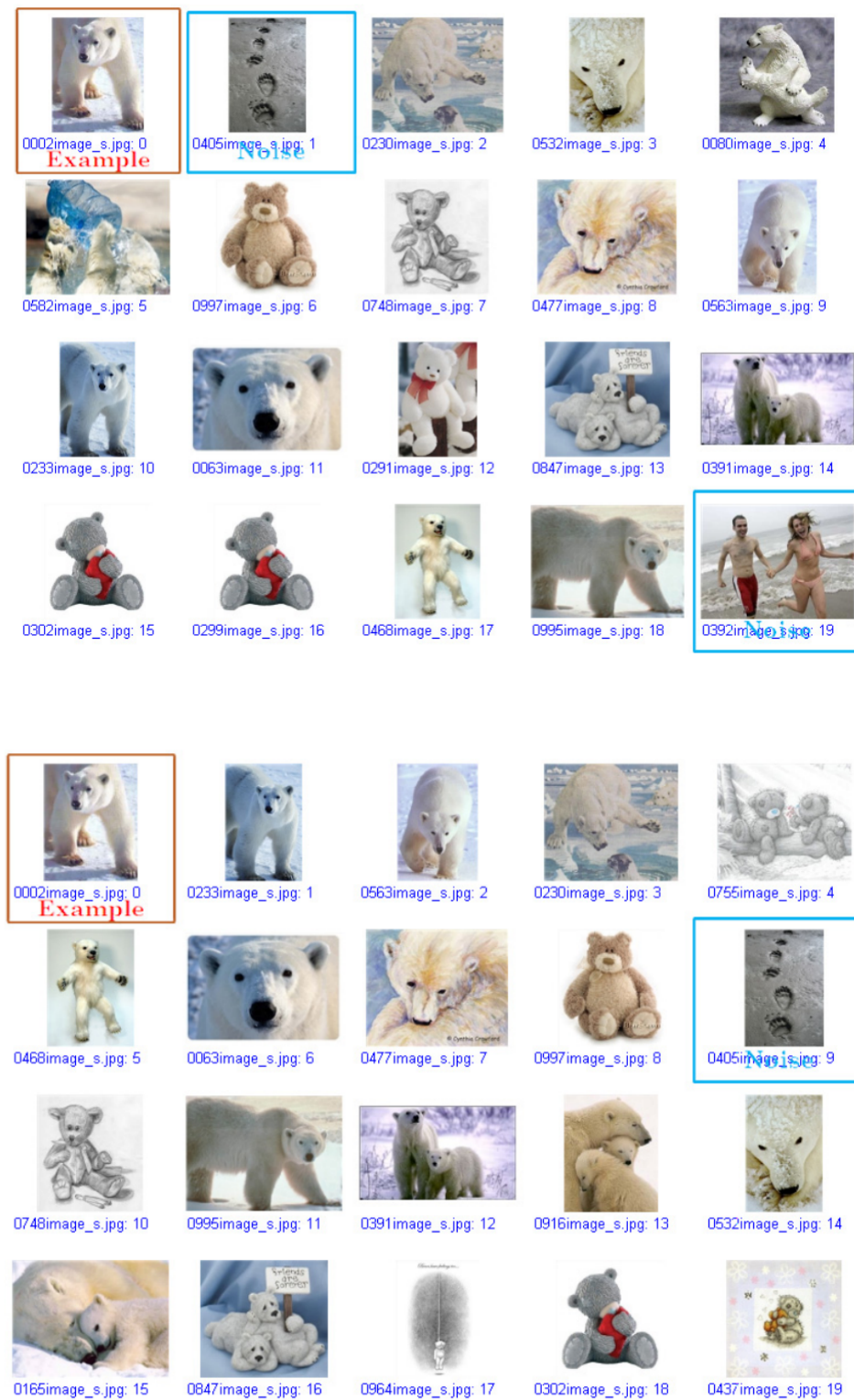
Cette méthode est simple et efficace, car elle ne nécessite aucune autre information outre la requête textuelle et les résultats de la recherche initiale.

L'objectif de cette méthode est de trier les images à l'aide de requêtes QbE. Plus les images ressembleront à celle de la requête, plus elles seront prioritaires.

Cependant, les moteurs de recherche multimédia textuels n'ont pas toujours des performances satisfaisantes, principalement à cause des textes environnants bruyants ou manquants.

Une méthode de comparaison a été utilisée par Zhang Wei, Xue Tianfan dans l'article [**ExampleBasedReranking**]

Les résultats obtenus sont assez satisfaisant car cela élimine également les "bruits".



2.0.2 Interactive reranking

L'interactive reranking ou reclassement interactif est un concept de reclassement des résultats faisant intervenir l'utilisateur. Ce dernier va indiquer quelles sont les images pertinentes ou non. Cela va permettre au système de prendre en considération les données fournies par les utilisateurs afin d'affiner les résultats pour les futures recherches.

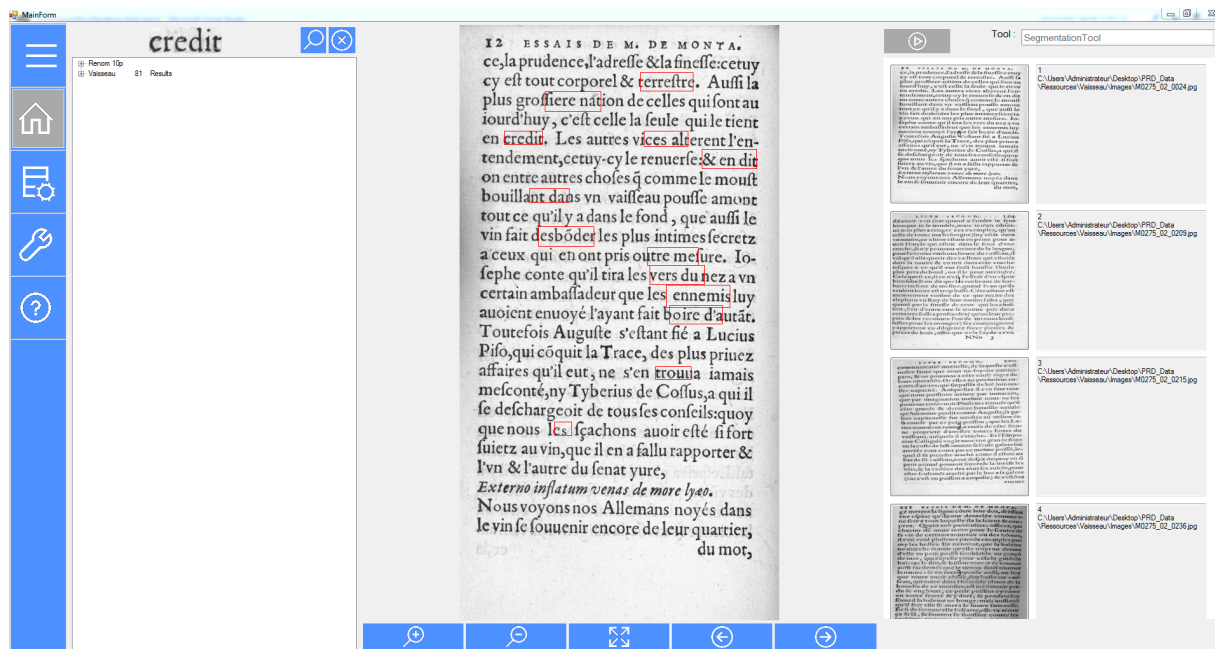
L'interaction de l'utilisateur permet de régler les problèmes suivants :

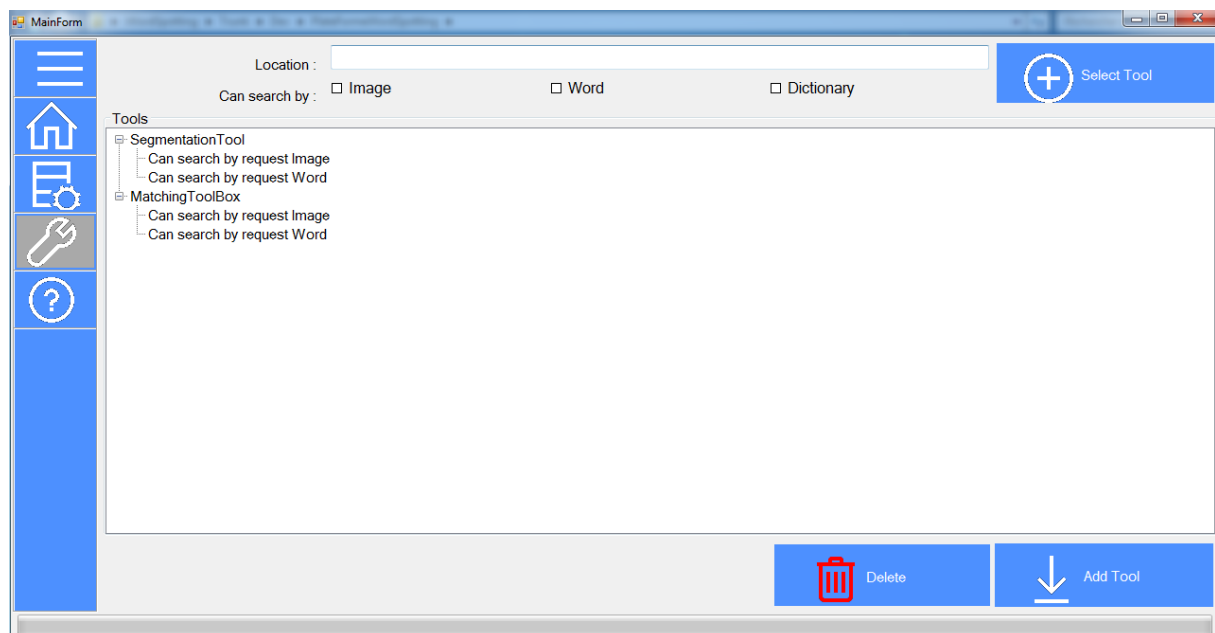
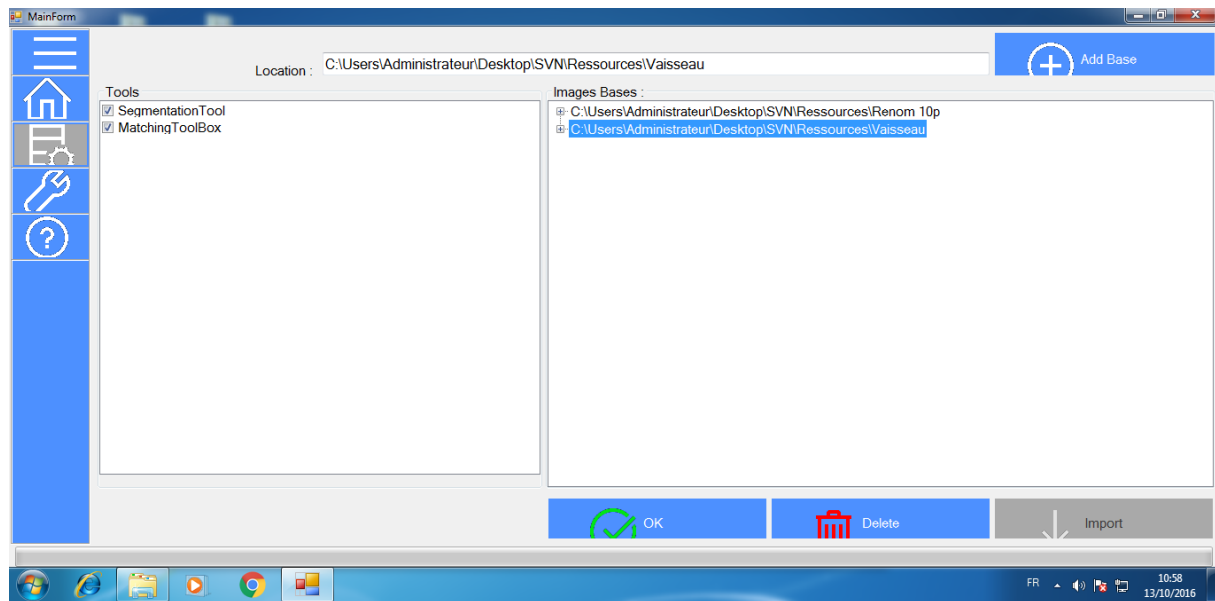
- L'indexation automatique des documents visuels n'est pas encore au point.
- Il est difficile de fournir une bonne requête visuelle au système qui va faire que le reclassement se fasse bien.
- L'ordinateur interprète plus facilement du texte que des images comparé à un humain.

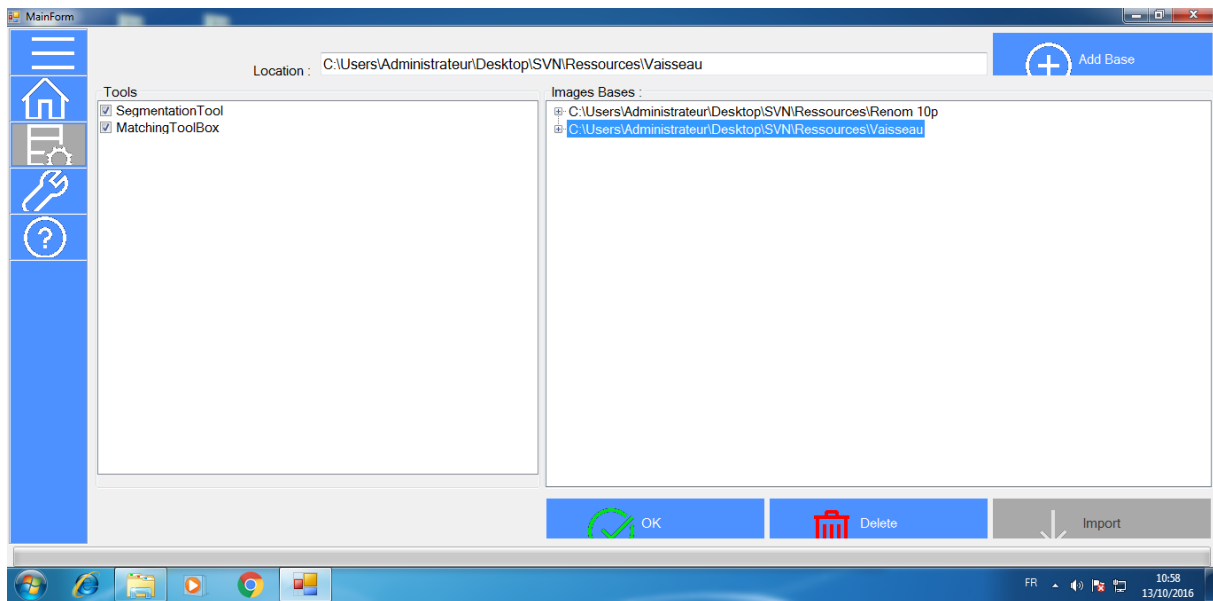
4

Analyse et conception

1 Analyse de l'existant







Actuellement, je n'ai pas pu trouver la recherche par dictionnaire comme indiqué dans le rapport de Loreen [RapportLoreen].

2 Stabilisation de la plateforme

Il existe dans l'application plusieurs problèmes qui rendent la plateforme instable. L'application de wordspotting fonctionne dans certains cas précis et demande de modifier des chemins codés en dur dans l'application.

```
//Attention, on a du mettre le chemin en dur pour que cela fonctionne
string outilBinarization = "C:/Users/Administrateur/Desktop/iwordspotting/Trunk/Dev/Binarization/Debug/Binarization.exe";
//string outilBinarization = ConfigurationManager.AppSettings["outilBinarization"];
Console.WriteLine(outilBinarization);
```

```
private void backgroundWorker1_RunWorkerCompleted(object sender, System.ComponentModel.RunWorkerCompletedEventArgs e)
{
    HandleThreadDone();
    if (treeViewBases.SelectedNode.Parent != null)
    {
        zoomPanImageBox1.RectanglesResults = ResultQuery.LoadResultForImage(_currentBase.RootPath, treeViewBases.SelectedNode);
    }
    else
    {
        zoomPanImageBox1.RectanglesResults = ResultQuery.LoadResultForImage(_currentBase.RootPath, treeViewBases.SelectedNode);
    }
    _resultData = ResultQuery.LoadResult(_currentBase.RootPath, treeViewBases.SelectedNode);
    RefreshListBases();
    treeViewBases.Focus();
}
```

L'exception `NullReferenceException` n'a pas été gérée par le code utilisateur. La référence d'objet n'est pas définie à une instance d'un objet.

Conseils de débogage :

- Déterminez si l'objet est null avant d'appeler la méthode.
- Utilisez le mot clé "new" pour créer une instance d'objet.

Pour régler ces problèmes je compte mettre en place des tests unitaires et lancer l'application en mode debug et tester le plus de cas possible.

2.1 Problèmes liés aux outils de recherche

L'ancien outil de segmentation utilisé dans la plateforme ne fonctionne pas. L'outil Matching tool box ne fonctionne également pas si utilisé sur la base d'image "renom p10". Il est donc prévu

d'intégrer l'outil développé par Quentin Combemorel.

3 Diva Services

3.1 Présentation de DivaServices

Lors de son PRD Quentin Combemorel a étudié l'API Diva Services qui est une API permettant d'utiliser des algorithmes utiles au Wordspotting. Cet outil a été développé à l'université de Fribourg en Suisse par un étudiant en PhD Marcel Würsch, le Prof. Dr. habil. Marcus Liwicki et le Prof. Dr. Rolf Ingold.

L'API est accessible sans authentification et sans limitation de requêtes. Les seules contraintes sont que les images et les résultats ne sont hébergés que 48h et qu'il est nécessaire d'avoir une connexion internet.

La documentation de l'API se trouve ici : <https://lunactic.github.io/DIVAServicesweb/articles/api/> L'algorithme que l'on souhaite utiliser est la méthode Seam Carving Text Line Extraction.

L'url pour utiliser cet algorithme est le suivant : <http://divaservices.unifr.ch/api/v2/segmentation/seamcarvingtextlineextraction/1>

On peut obtenir un résultat de ce type :

pas longtemps à cette influence pernicieuse. Elle s'approprie les poisons que distillent les mauvaises lectures, et qui passent, pour ainsi dire, dans la circulation intellectuelle. En voulez-vous un exemple? Nous avons longtemps cherché d'où venait l'incohérence d'idées et de sentiments qu'on remarque chez M. Victor Hugo. *Le Témoin de sa vie* vient de nous expliquer ce problème. La mère du poète, devenue coupable à force d'être insensée, l'enfermait dans l'entresol d'un cabinet de lecture rempli de livres contre la religion et les mœurs, et le chargeait d'*essayer* les ouvrages avant qu'elle les lût elle-même. Depuis ce moment l'ordre n'est pas rétabli

pas longtemps à cette influence pernicieuse. Elle s'approprie les poisons que distillent les mauvaises lectures, et qui passent, pour ainsi dire, dans la circulation intellectuelle. En voulez-vous un exemple? Nous avons longtemps cherché d'où venait l'incohérence d'idées et de sentiments qu'on remarque chez M. Victor Hugo. *Le Témoin de sa vie* vient de nous expliquer ce problème. La mère du poète, devenue coupable à force d'être insensée, l'enfermait dans l'entresol d'un cabinet de lecture rempli de livres contre la religion et les mœurs, et le chargeait d'*essayer* les ouvrages avant qu'elle les lût elle-même. Depuis ce moment l'ordre n'est pas rétabli

3.2 Utilisation de DivaServices

Pour utiliser l'api il faut commencer par upload des images. Pour se faire, il faut exécuter des requêtes POST avec un corps JSON sur l'url suivante : <http://divaservices.unifr.ch/api/v2/collections>

Voici le corps JSON pour upload une image via une URL.

```
{
  "name": "collectionName",
  "files": [
    {
      "type": "url",
      "value": "imageUrl",
      "name": "imageName"
    }
  ]
}
```

On peut également upload une image en utilisant le type file et en passant en valeur une image en base 64. C'est ce que nous utiliserons pour l'application.

Pour utiliser l'algorithme Seam Carving on doit effectuer une requête POST sur l'url suivante : <http://divaservices.unifr.ch/api/v2/segmentation/seamcarvingtextlineextraction/1>

Le corps JSON peut ressembler à ça :

```
{
  "parameters": {
    "smooth": 0.0003,
    "slices": 4,
    "sigma": 3
  },
  "data": [
    {
      "inputImage": "PRDWordspotting/example1.png"
    }
  ]
}
```

Une requête sur l'API Diva prend toujours des paramètres et des données.

Dans notre cas, les paramètres sont des options pour la méthode Seam Carving Text Line Extraction.

En valeur inputImage nous avons PRDWordspotting/example1.png

PRDWordspotting correspond au nom de la collection et example1.png correspond au nom de l'image de la collection.

Après exécution, nous obtenons un résultat sous forme JSON :

```
{
  "results": [
    {
      "resultLink": "http://divaservices.unifr.ch/api/v2/results/chubbyconsciousseahog/data_0/data_0.json"
    }
  ],
  "resultCollection": "PRDWordspotting_seamcarvingtextlineextraction_2019_0_30_10_3_48",
  "resultCollectionLink": "http://divaservices.unifr.ch/api/v2/collections/PRDWordspotting_seamcarvingtextlineextraction_2019_0_30_10_3_48",
  "resultLink": "http://divaservices.unifr.ch/api/v2/results/stableconsiderateozarkbigearredbat.json",
  "message": "This url is available for 24 hours",
  "status": "done",
  "statusCode": 202
}
```

Ce qui nous intéresse ici c'est l'URL du resultLink. Lorsqu'on se rend dessus, on obtient le JSON suivant :

```
{
  "output": [
    {
      "file": {
        "name": "textlines",
        "type": "json",
        "description": "extracted text lines",
        "options": {
          "visualization": false,
          "filename": "textlines.json"
        },
        "mime-type": "application/json",
        "url": "http://divaservices.unifr.ch/api/v2/results/chubbyconsciousseahog/data_0/textlines.json"
      }
    },
    {
      "file": {
        "name": "visualization",
        "type": "image",
        "description": "Text line visualization",
        "options": {
          "visualization": true,
          "filename": "visualization.png"
        },
        "mime-type": "image/png",
        "url": "http://divaservices.unifr.ch/api/v2/results/chubbyconsciousseahog/data_0/visualization.png"
      }
    },
    {
      "file": {
        "mime-type": "text/plain",
        "type": "log",
        "url": "http://divaservices.unifr.ch/api/v2/results/chubbyconsciousseahog/data_0/logFile.txt",
        "name": "logFile.txt",
        "options": {
          "visualization": false
        }
      }
    }
  ],
  "status": "done",
  "resultLink": "http://divaservices.unifr.ch/api/v2/results/chubbyconsciousseahog/data_0/data_0.json",
  "collectionName": "chubbyconsciousseahog"
}
```

3 fichiers sont générés après plusieurs secondes d'attente :

- `textlines` : C'est le fichier qui nous intéresse pour l'application, il contient les coordonnées des lignes à extraire.
- `visualization` : Fichier intéressant également car il permet de visualiser comment la méthode a découpé l'image.
- `logFile.txt` : Permet de savoir si il y a eu un problème avec la méthode.

5

Mise en oeuvre

1 Stabilisation de la plateforme

Un des objectifs de ce projet était de rendre l'application de wordspotting plus stable. Voici ce que j'ai réalisé pour améliorer cela.

1.1 Correction des chemins

La première chose faite a été de corriger le problème des chemins.

Après avoir retrouvé dans l'application le code correspondant au chemin de l'application de binarization j'ai remarqué que celui-ci était entré en dur. J'ai remplacé alors ce dernier par le code suivant :

```
string outilBinarization = ConfigurationManager.AppSettings["outilBinarization"];
```

Cela permet de récupérer le chemin indiqué dans le fichier App.config de l'application.

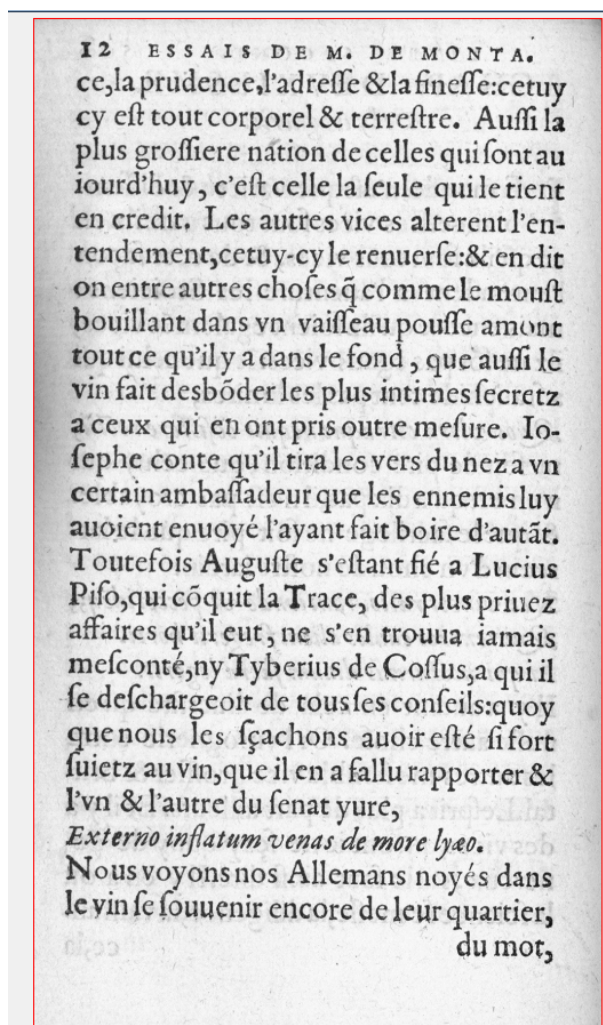
1.2 Affichage des rectangles

Après avoir étudié le code correspondant à l'affichage des rectangles. Il a été assez difficile de comprendre si le problème de positionnement des rectangles de résultat venait de la méthode d'affichage.

On peut alors se demander si la position des rectangles de recherche ne sont pas mal placés à cause des faux positifs de l'algorithme MatchingToolBox.

La méthode responsable de l'affichage est la suivante : `LoadResultForImage(string rootPath, string fileImage)`

Pour vérifier les hypothèses j'ai créé un fichier de résultats pour les tests affichant un rectangle faisant la taille de l'image.



On peut constater que ce dernier se place correctement.

Le problème de positionnement, apparaît donc avant l'écriture des coordonnées dans le fichier Resultats.xml. Conclusion, cela vient donc bien de l'application MatchingToolBox et de non de la plateforme.

1.3 Plantages

Les plantages arrivent quand un outil externe est utilisé et que ce dernier ne s'exécute pas correctement. Cela rend l'application instable et incohérente.

Dans la classe Tool.cs j'ai trouvé la méthode qui appelle les outils externes : SearchByImg(string pathBase, string pathRequest).

Afin d'utiliser les outils externes, la classe Process est utilisée. J'ai lu la documentation pour comprendre comment cela fonctionnait : <https://docs.microsoft.com/fr-fr/dotnet/api/system.diagnostics.process?view=netframework-4.7.2>

C'est la méthode backgroundWorker1_DoWork (object sender, System.ComponentModel.DoWorkEventArgs e) de la classe Home qui utilise SearchByImg.

Cette méthode possède une boucle infinie qui s'arrête quand le fichier de résultats est créé.

```
if (File.Exists(_currentBase.RootPath+"\\Resultats.xml")) {
    _modeResult = true;
    _ResultForBase = _currentBase.RootPath;
    e.Cancel = true;
    return;
}
```

Cependant, si on arrête l'exécution de l'outil ou que ce dernier plante, le fichier n'est jamais créé et donc on ne sort jamais de la boucle. Pour régler le problème j'ai ajouté des paramètres à l'objet Process.

```
ProcessInfo.EnableRaisingEvents = true;
ProcessInfo.Exited += new EventHandler(myProcess_Exit);
```

Cela permet d'appeler une méthode (ici myProcess_Exit) quand l'outil est fermé. L'appel est donc fait quoi qu'il arrive : fermeture normale ou plantage.

```
private void myProcess_Exit(object sender, System.EventArgs e)
{
    PlateFormeWordSpotting.Views.Home.ProcessExited = true;
}
```

Cette méthode met un flag à vrai dans la classe Home. C'est dans la classe Home que les interactions n'allaient pas.

```
if (processExited)
{
    e.Cancel = true;
    return;
}
```

Si le flag est à vrai, on sort de la boucle infinie.

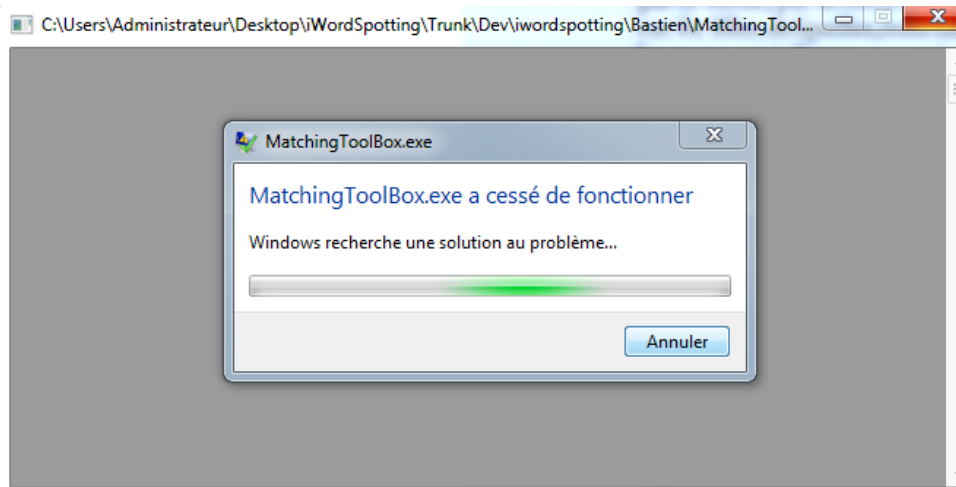
Enfin, pour éviter un plantage de la méthode backgroundWorker1_RunWorkerCompleted qui est appelée dès que la méthode backgroundWorker1_DoWork est terminée, j'ai ajouté une condition pour savoir si le fichier de résultat est présent. Si il n'existe pas, on ne va pas aller chercher les résultats et en plus, j'utilise l'action d'annulation (code existant) pour remettre toute l'interface utilisateur en phase fonctionnelle.

```
if (File.Exists(_currentBase.RootPath + "\\Resultats.xml"))
{
    if (treeViewBases.SelectedNode.Parent != null)
        zoomPanImageBox1.RectanglesResults = ResultQuery.LoadResultForImage(_currentBase.RootPath, treeViewBases.SelectedNode.ToolTipText);
    else
        zoomPanImageBox1.RectanglesResults = ResultQuery.LoadResultForImage(_currentBase.RootPath, treeViewBases.SelectedNode.FirstNode.ToolTipText);
    _resultData = ResultQuery.LoadResult(_currentBase.RootPath);
}
else
{
    Cancel();
}
```

1.4 Problème base image Renom P1

L'outil MatchingToolBox ne fonctionne pas avec la base renom. Une première hypothèse serait que les fichiers soient beaucoup trop gros.

Il est également fortement possible que le problème vienne de l'outil MatchingToolBox et non de la plateforme.

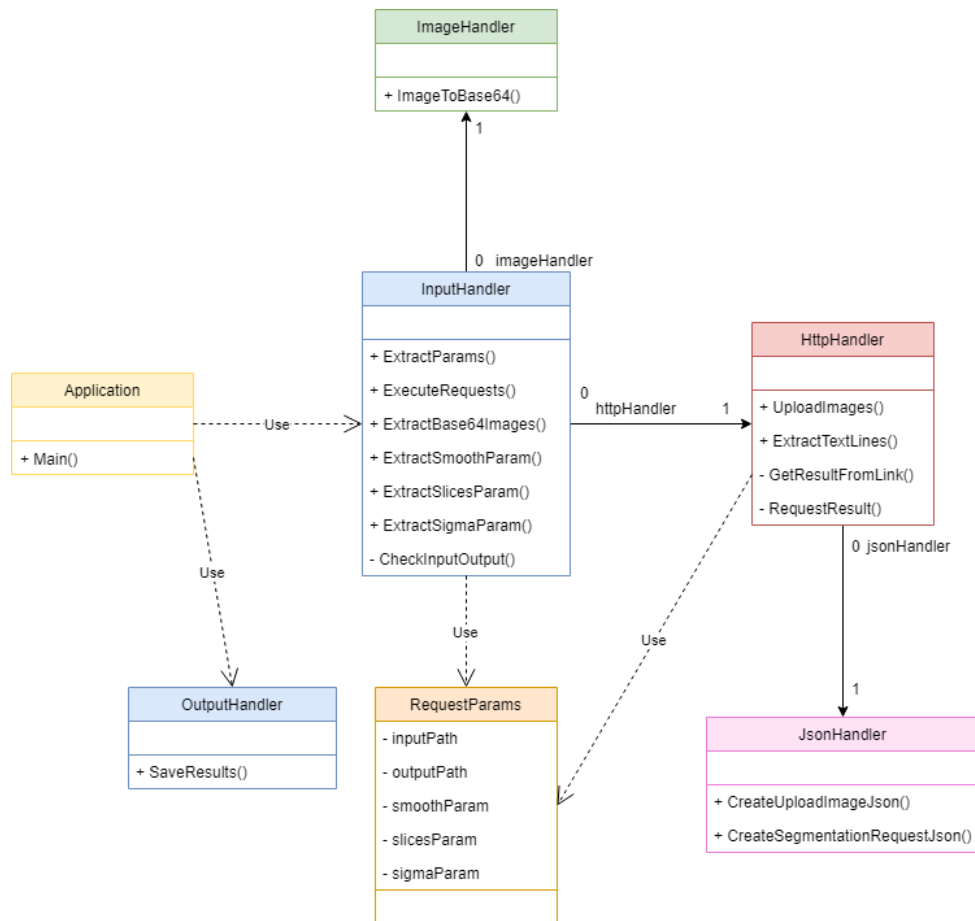


La console n'affiche pas d'erreur. Pour régler le problème il faudrait voir directement le code de MatchingToolBox. Cependant cela risquerait de prendre un certain temps et cela sort du cadre du projet.

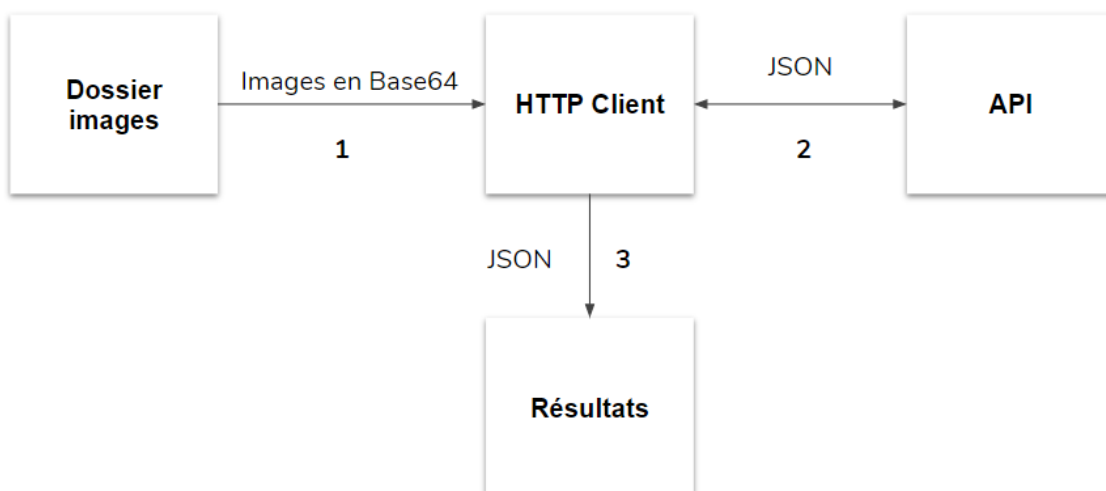
2 Application DivaSegmentation

DivaSegmentation est l'application que j'ai développée lors de ce semestre 10. Cette application permet d'automatiser les appels à l'API DivaServices.

2.1 Modélisation



2.2 Fonctionnement



L'application prend en entrée un dossier contenant des images de texte et un dossier de sortie. Des paramètres liés à l'analyse d'image peuvent être précisés.

Les images sont converties en base 64 puis envoyées une par une par le client HTTP (format JSON) au serveur d'API. Les réponses (format JSON) sont récupérées une fois les traitements réalisés par l'API. Enfin, les résultats sont sauvegardés dans le dossier de sortie.

```
[
  {
    "array": {
      "name": "textline",
      "dimensions": {
        "n": 2,
        "shape": [
          1011,
          2
        ]
      },
      "type": "numeric",
      "values": [
        [
          1,
          1
        ],
        [
          1008,
          1
        ]
      ]
    }
  }
]
```

6

Bilan et conclusion S9

La difficulté principe du projet est la reprise d'une application contenant de nombreuses dépendances et utilisant plusieurs algorithmes différents. Pour me lancer dans la partie recherche et spécifications, j'ai dû prendre connaissance des projets des anciens étudiants ce qui n'a pas été une chose simple. J'ai également dû installer l'application. Cependant elle ne fonctionnait pas correctement et c'est après plusieurs tentatives de débogage et la récupération d'une autre version de l'outil de binarisation que j'ai pu enfin lancer l'application pour la première fois pour la tester.

Après cette phase de mise en place, j'ai eu plusieurs rendez-vous avec mon encadrant M. Ragot afin de définir des objectifs précis à réaliser sur la plateforme. J'ai pu m'apercevoir qu'il y avait vraiment beaucoup de choses possibles à réaliser mais assez peu de temps pour le faire en réalité.

J'ai ensuite passé une grande partie de mon temps à rédiger les spécifications. Il y a eu au total 4 versions. L'erreur que j'ai pu faire est que je n'ai pas bien su répartir mon temps entre rédaction des spécifications et état de l'art ce qui a fait que j'ai perdu en qualité de recherche et en pertinence des choix de solution.

La gestion de projet est quelque chose que je n'ai pas l'habitude de faire. Le PRD a été l'occasion d'appliquer les cours de gestion de projet enseignés en 4ème année. Malgré cela, planifier est une tâche difficile. Même en ayant les objectifs il a été difficile pour moi de donner des dates et durées pour la réalisation des tâches.

Pour la partie recherche et analyse, c'est la partie qui m'a posé le plus de problèmes. Le fait d'avoir à installer et étudier un existant m'a pris du temps et comme les objectifs n'étaient pas encore définis je ne savais pas sur quoi axer mon état de l'art. Grâce à l'aide de M. Ragot j'ai pu obtenir une orientation.

Pendant le projet, j'ai pu prendre conscience qu'il fallait bien se concentrer sur les objectifs et ne pas hésiter à contacter son encadrant le plus souvent possible pour ne pas rester bloqué.

Enfin, j'ai tout de même pu apprendre de nouvelles choses sur le sujet du word spotting, apprendre à lire des articles scientifiques en anglais, rédiger des spécifications, gérer un projet, échanger avec mon encadrant etc... Et contrairement aux autres projets réalisés à Polytech, celui-ci se fait seul.

7

Bilan et conclusion S10

La deuxième partie du projet a été assez complexe pour l'organisation des tâches. En effet, il y avait beaucoup d'objectifs pour peu de temps à la réalisation.

La partie débogage de l'application a pris plus de temps que prévu car les bugs étaient nombreux et difficile à corriger. De plus, j'ai tenté d'extraire l'application de la machine virtuelle sans succès.

Pour le développement de l'application de segmentation, j'ai sous estimé le temps de la réalisation. Je n'avais pas pris en compte le temps de mise en place de la qualité logicielle : tests unitaires, commentaires et rédaction des guides.

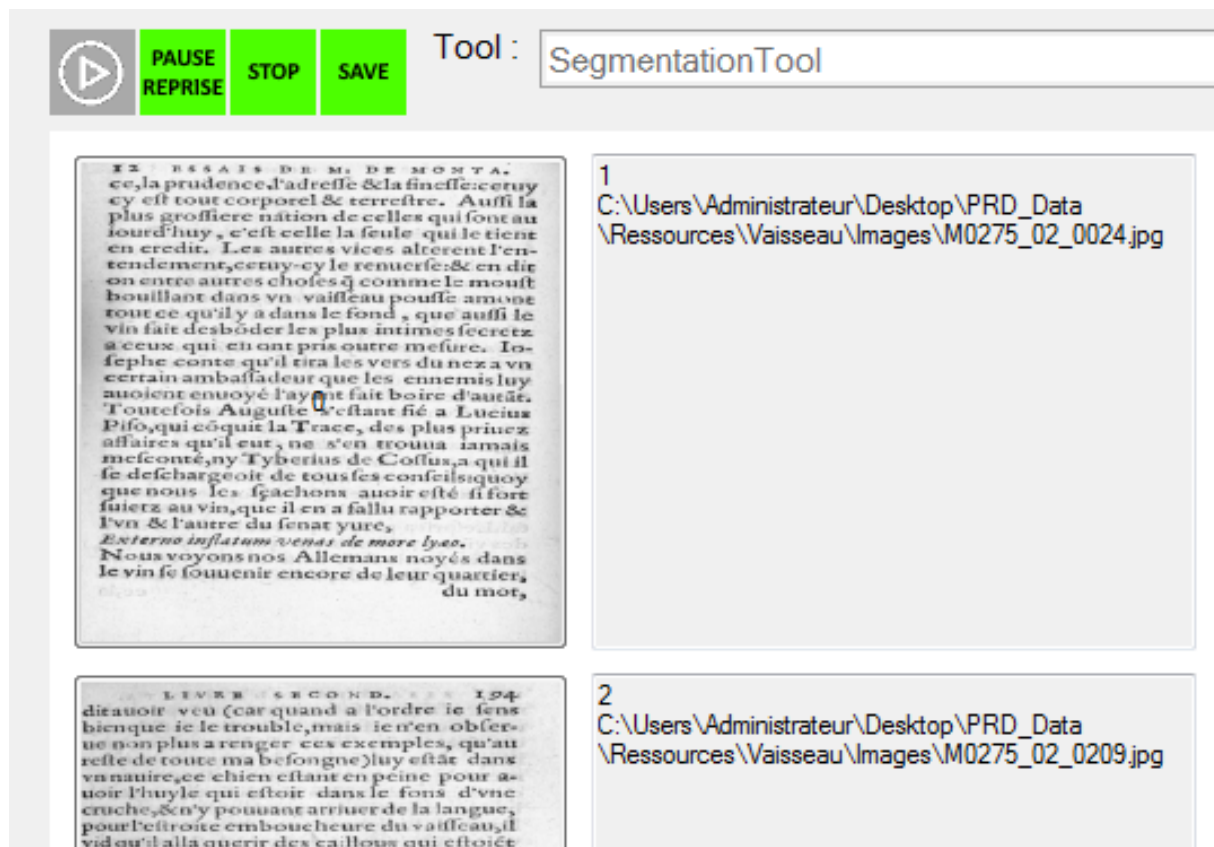
Pendant ce PRD 2 j'ai progressé sur la partie qualité logicielle et j'ai bien assimilé le fait qu'il est important que les applications soient simples à reprendre.

Annexes

A

Description des interfaces externes du logiciel

1 Interface homme - machine



En vert se trouve l'emplacement des futurs boutons de contrôles pour la mise en pause/reprise, l'arrêt et l'enregistrement des résultats.

B

Spécifications fonctionnelles

Dans cette partie sont détaillées toutes les fonctionnalités que je vais devoir réaliser afin de répondre à la problématique du sujet. Les tâches sont dans l'ordre de réalisation.

1 Stabilisation de la plateforme

Il existe dans l'application plusieurs problèmes qui rendent la plateforme instable. Pour commencer, la visualisation des résultats n'est pas toujours correcte. En effet, les rectangles rouges placés sur les mots trouvés sont parfois décalés. On peut également trouver des chemins de configuration codés en dur. Enfin, plusieurs bugs mineurs ou majeurs sont présents dans la plateforme et il serait bon de les corriger.

Cet objectif sera validé si les bugs indiqués dans le rapport de bug (annexe) ne sont plus présents et que l'emplacement des rectangles pour les résultats soient correctement placés.

Afin d'arriver au bout de cet objectif, je vais avoir besoin de refaire une analyse du code avec des tests et débogages.

La durée de cette tâche est estimée à 1 semaine et devra être réalisée fin Décembre.

2 Intégration de l'outil de segmentation de Quentin Combemorel

L'application ne possède pas d'outil de segmentation intégré. Afin de segmenter une image, il faut passer par l'API Diva Services. Lors de son PRD de 2016-2017, Quentin Combemorel [**RapportQuentin**] a développé un outil de segmentation, cependant il n'a pas eu la possibilité de l'intégrer à la plateforme. L'objectif est donc l'intégration de l'outil développé par Quentin Combemorel [**RapportQuentin**].

L'objectif sera validé si l'outil de segmentation est bien intégré et fonctionnel.

Afin de réaliser cette tâche, je vais devoir comprendre en détails le fonctionnement de cet outil et ensuite mettre en place des tests pour savoir si tout se passe bien.

La durée estimée pour la réalisation de cette tâche est de 2 semaines début Janvier.

3 Arrêt de la recherche

Actuellement pour obtenir les résultats d'une recherche, il faut attendre la toute fin. Cependant, cela peut prendre du temps et l'utilisateur peut vouloir uniquement des résultats contenu dans les premières pages d'un livre par exemple.

L'objectif sera validé si l'utilisateur peut stopper la recherche et quand même consulter les résultats.

Le temps estimé pour la réalisation est de 1 semaine pendant la deuxième moitié du mois de Janvier.

4 Migration de SVN à GitLab

Le projet est actuellement versionné sous SVN. L'objectif est de passer à l'outil GitLab qui propose plus de fonctionnalités. Les fonctionnalités intéressantes de GitLab comme :

- Gestion des rôles des utilisateurs
- Création, assignation, classification d'issues (tâches)
- Versionning avec notifications

Cet objectif sera réalisé si le projet est bien présent sur GitLab et que le versionning fonctionne encore après.

Pour migrer de SVN à GitLab il va falloir s'assurer qu'aucun fichier n'est manquant.

Le temps de réalisation de cette tâche est estimé à 1 semaine.

5 Visualisation interactive des résultats (reranking)

Cette fonctionnalité consistera à mettre en oeuvre un moyen pour pouvoir réorganiser les résultats de recherche proposés par l'algorithme de recherche. En effet des résultats peuvent ne pas être organisés comme le souhaiterait l'utilisateur.

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. Il pourra également supprimer les résultats non pertinents et en rajouter à la main.

Je pourrai intégrer soit la méthode Example Based Reranking ou Interactive Reranking.

Cette fonctionnalité sera validée si il y a la possibilité de reclasser les résultats de façon satisfaisante.

Le temps estimé est de 3 semaines.

6 Enregistrement et chargement des résultats

Les résultats sont pour le moment directement affichés et il n'y a aucun système de conservation de ces derniers. L'ajout d'une fonctionnalité d'enregistrement permettrait de consulter les résultats plus tard sans avoir à relancer une nouvelle recherche.

Cet objectif sera validé si il y a présence d'un bouton d'enregistrement et de chargement. Il faut également que le chargement du fichier affiche bien les résultats dans l'application.

Pour réaliser cette tâche, je vais devoir trouver un format pour les fichiers en m'inspirant de la façon dont l'application passe de la recherche à l'affichage des résultats. Puis je vais également devoir mettre en place des tests pour m'assurer du bon fonctionnement de l'enregistrement et du chargement.

J'estime le temps de mise en place de cet fonctionnalité à 2 semaines entre Janvier et Février.

7 Confort d'usage, mise en pause, reprise de la recherche

Actuellement lors d'une recherche, un invité de commande apparaît et l'emplacement des rectangles est répertorié dans des CSV. L'opération peut prendre du temps et l'utilisateur doit attendre la fin pour consulter les résultats. Et si une nouvelle recherche est faite, les résultats sont perdus.

Une solution proposée, serait alors de donner la possibilité à l'utilisateur de pouvoir consulter les résultats au fur et à mesure, de mettre en pause la recherche si elle est longue.

Cet objectif sera validé si les boutons pause, reprise, arrêt sont bien ajoutés et qu'ils fonctionnent tous et que les résultats sont affichés au fur et à mesure que la recherche se fait.

Afin de valider cet objectif, je vais devoir retoucher l'IHM et faire en sorte que l'interaction avec l'invité de commande affichant les résultats ne pose pas de problème tel que un blocage de l'application.

Le temps estimé pour la réalisation est de 2 semaines pendant la deuxième moitié du mois de Janvier.

8 Filtrage des résultats avec seuil

Les résultats affichés ne sont pas toujours concluant. Le ou les mots recherchés sont souvent trouvés cependant de nombreux autres mots qui n'ont rien à voir sont également pris en compte. L'ajout d'une barre de défilement pour réduire ou augmenter la précision de résultats affichés pourrait permettre d'afficher uniquement les bons mots et donc de rendre la navigation plus agréable pour l'utilisateur.

Cette tâche sera validée si il y a présence d'une barre de défilement et que lorsqu'on déplace le curseur vers la gauche, on restreint les résultats et si on la déplacement vers la droite, on élargit les résultats.

Pour mettre en place cette fonctionnalité il va falloir modifier la façon dont les résultats sont affichés. Il est possible que la réalisation soit plus compliquée que prévu en fonction de la façon dont est développée l'affichage actuel.

Le temps de réalisation de cette tâche devrait être de 2 semaine en parallèle avec l'enregistrement et chargement des résultats.

9 Mise en place d'un processus d'indexation pour accélérer la recherche

Cette fonctionnalité doit permettre de gagner du temps sur la recherche. Lors des recherches il y a création de fichiers de caractéristiques. L'idée serait d'indexer les caractéristiques afin de pouvoir les réutiliser directement sans avoir à tout recréer. Par exemple, en utilisant des tables de hachage, la plateforme pourra utiliser les indexes pour accéder plus rapidement aux informations. Ce qui aura pour conséquence de réduire drastiquement le temps du processus de recherche.

Cette tâche sera validée si l'indexation permet bien de gagner du temps en faisant plusieurs recherches.

Cette fonctionnalité demande de modifier beaucoup de choses dans le code en plus de développer le système d'indexation.

Cette tâche est moins prioritaire que les autres, et si elle n'est pas réalisée, elle pourra constituer une fonctionnalité à implémenter pour les prochaines personnes travaillant sur la plateforme.

C

Spécifications non fonctionnelles

1 Contraintes de développement et conception

- Langage de programmation : C++ et C avec le framework .NET 4.5.
- Logiciel : VisualStudio 2012.
- Environnement : Machine virtuel de l'école, Windows 7 Pro 64 Bits DEVELOPPEMENT.
- Bibliothèques de programmes imposées : OpenCV, AForge.NET.
- Modélisation UML

2 Capacités

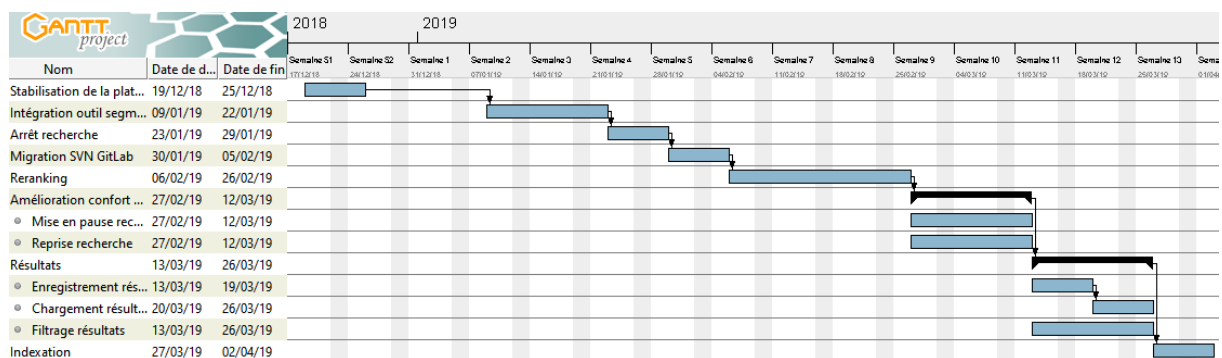
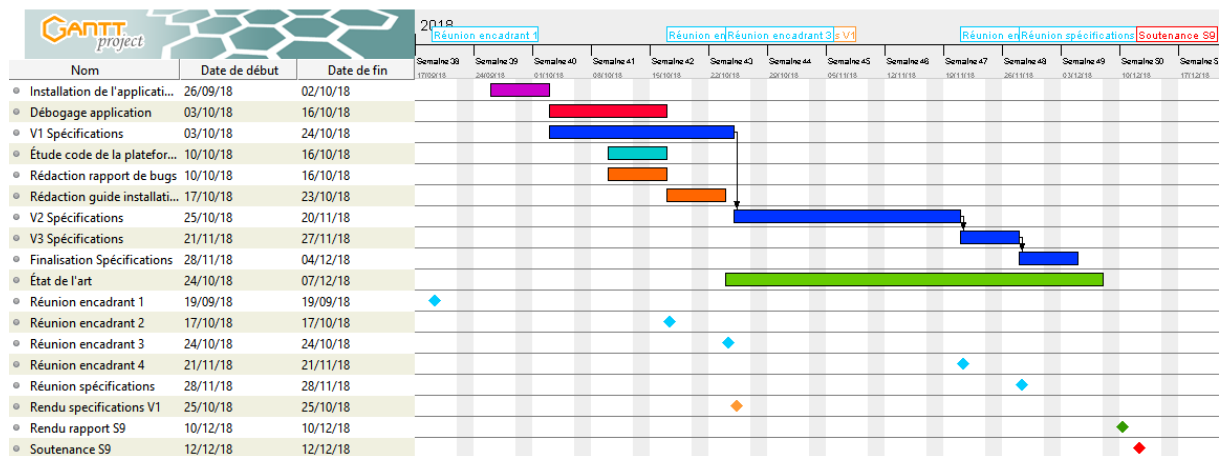
- Une seule recherche à la fois, en effet le système ne pourra pas prendre en compte plusieurs recherches à la fois étant donné que pendant la période de recherche l'application est ne pourra exécuter d'autres tâches.
- La taille des images. Il faut savoir que plus la taille des images est grande plus le traitement sera long. L'idéal serait d'utiliser des images ayant une taille réduite.
- Le nombre d'images dans une base. Cette contrainte est liée à la précédente.



3 Contrôlabilité

Si jamais des erreurs sont lancées, une boîte de dialogue sera visible par l'utilisateur afin qu'il puisse faire des retours. De plus des traces seront disponibles lors des recherches, celles-ci permettent de visualiser le détail des résultats des recherches effectuées.

La plupart des erreurs sont déjà gérées, mais quand un plantage survient, rien ne s'affiche. Il serait peut être judiciable d'afficher les codes d'erreurs à l'utilisateur.

Gestion de projet



PRD Wordspotting ☆ | Personnel | Privé |  NR 2 

To Do	Doing	Done
Stabilisation de la plateforme	Modification Gantt	Debug application binarization
Modifier les chemins pour l'installation de l'application	Ajouter diagramme de classe (ancien)	Rédaction du cahier des specs
Migration du SVN à GIT	Intégrer le guide d'installation au rapport	Tester le bon fonctionnement de la plateforme
Intégration outil segmentation Quentin	Création diapo soutenance	Rédaction état de l'art
Reranking	+ Ajouter une autre carte	Rédaction bilan et conclusion
Arrêt recherche		+ Ajouter une autre carte
Mise en pause, reprise recherche		
Enregistrement résultats		
Chargement résultats		
Indexation		
Filtrage des résultats		
+ Ajouter une autre carte		

SPRINT 1 - Démarche qualité, identification correction bugs	SPRINT 1 - Done	SPRINT 2 - Application de segmentation	SPRINT 2 - Done
+ Ajouter une carte	Chemins en dur	+ Ajouter une carte	Rédaction guide utilisation
	Vérifier affichage des rectangles		Rédaction rapport, cahier dev
	Trouver le pb de la base Renom (voir taille etc...)		Développement de la partie HTTP
	Voir les plantages (gérer les plantages pour que l'application fonctionne encore après)		Développement de la partie Parsing de résultats
	Vérifier que DIVA fonctionne encore		Diagramme de classe
	+ Ajouter une autre carte		Développement de la partie input / output
			Documentation
			Tests unitaires
			Développement de la partie Json
			Tentative de migration de l'application hors VM (echec)
			Étude de l'API Diva Services
			+ Ajouter une autre carte

E

Guide d'installation plateforme

1 Prérequis

Une VM développement de l'école sous Windows.

Open CV 2.4.9 installé avec les variables d'environnement pointant sur la version x86. Tuto : <http://opencv-srf.blogspot.fr/2013/05/installing-configuring-opencv-with-vs.html>.

Le nom de la variable d'environnement à utiliser est : `OPENCV_DIR`.

2 Récupération du projet

Le projet est disponible via GitLab. Afin de le récupérer sur l'ordinateur, depuis un dossier nommé iWordSpotting et placé sur le bureau*, lancer un clone avec l'adresse suivante : <https://gitlab.com/nicolas.ragot/word-spotting-framework.git>. Puis récupérer la branche "all" en local pour travailler.

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

L'onglet Tag contiendra toutes les versions dites "stable" du projet général.

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

Note : Penser à vérifier que tous les dossiers / fichiers ont été téléchargés.

* Les chemins sont actuellement en dur dans l'application, et nécessite donc de s'adapter à la hiérarchie.

3 Configuration du projet

3.1 Outil binarization

3.1.1 Configuration de Visual Studio

Ouvrir le projet Visual Studio de l'outil binarization.

Si non présent : Ajouter le tag `_CRT_SECURE_NO_WARNINGS` dans Propriété de Binarization > Propriétés de configuration > C/C++ > Préprocesseur > Définitions de préprocesseur.

Uniquement pour lancer l'outil binarization depuis visual studio : Modifier les arguments de lancement de l'application. Penser à indiquer un dossier et non une image.

Régénérer la solution et compiler.

3.2 Plateforme Wordspotting

3.2.1 Configuration Visual Studio

Recharger les packages de Nuget, puis configurer les fichiers de config par rapport à la machine. Pour se faire : clic droit sur la solution puis "Enable Nuget package restore".

3.2.2 Configuration externe

Se rendre dans "Trunk/Dev/PlateFormeWordSpotting/PlateFormeWordSpotting/bin/Debug", puis ouvrir les fichiers `listBase.xml` et `listTools.xml`. Ensuite dans chaque dossier 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 modifiez le chemin.

Faire de même pour le fichier `listTools.xml`

F

Guide d'installation / utilisation

1 Prérequis

Prérequis pour faire fonctionner l'application :

- Une connexion internet
- Windows 7, 8 ou 10 avec Net Framework 4.5 ou plus récent
- Vous pouvez télécharger le Net Framework ici : <https://dotnet.microsoft.com/download>

2 Installation

L'application est mise à disposition par l'encadrant du projet M. Nicolas Ragot qui possède les accès Git des sources et de l'exécutable.

Pour exécuter l'application seul le dossier contenant l'exécutable et les librairies sont nécessaires.

3 Utilisation

L'application se veut indépendante et donc peut être utilisée directement via l'invite de commande windows.

L'application prend en paramètre : `<input> <output> [<smooth> <slices> <sigma>]` input : Un répertoire d'entrée contenant des images au format png, jpg ou jpeg.

Attention, les images doivent contenir des lignes de texte. De plus l'application prend les images dans l'ordre de leur nommage.

- output : Un répertoire de sortie où les résultats seront enregistrés.
- smooth : Valeur de lissage (entre 0 et 1)
- slices : Nombre de découpes de l'image (entre 2 et 8)
- sigma : Valeur sigma pour le filtre gaussien (entre 2 et 5)

Exemples valides d'arguments :

- `"C :/Users/onutr/Desktop/DivaTests/Input C :/Users/onutr/Desktop/DivaTests/Output"`
- `"C :/Users/onutr/Desktop/DivaTests/Input C :/Users/onutr/Desktop/DivaTests/Output 0.5 3 4"`

Note : Les paramètres smooth, slices et sigma sont facultatifs. Cependant si un des 3 est renseigné, tous doivent l'être. Les valeurs par défaut sont respectivement : 0.0003, 4 et 3.



Comptes rendus hebdomadaires

Compte rendu n°1 du 19/09/2018

- Premier rendez vous avec M. Ragot.

Compte rendu n°2 du 26/09/2018

- Lecture des nouveaux documents : Rapports et spécifications
- Installation de l'application

Compte rendu n°3 du 03/10/2018

- Création et rédaction du rapport et cahier des specs.
- Mise en place d'un Trello pour la gestion de projet.
- Premières tentatives de débogage de l'application.

Compte rendu n°4 du 10/10/2018

- Relecture du rapport de Quentin pour voir ce qui a été fait et non terminé.
- Étude du code de la plateforme.
- Récupération et débogage de l'outil de binarization (cf rapport de débogage)
- Tests des fonctionnalités et écriture d'un rapport de l'état de la plateforme (cf rapport de test de fonctionnement)
- Début de création d'un cas d'utilisation pour l'application.

Compte rendu n°5 du 17/10/2018

- Avancement cahier des specs
- Rédaction guide d'installation
- Réunion encadrant

Compte rendu n°6 du 24/10/2018

- Finalisation de la V1 du cahier des specs
- Réunion encadrant

Compte rendu n°7 du 7/11/2018

- Lecture de l'article sur le Reranking

- Début de rédaction partie Reranking
- Début rédaction de la V2 du cahier des spécifications

Compte rendu n°8 du 14/11/2018

- Rédaction de la V2 du cahier des spécifications

Compte rendu n°9 du 21/11/2018

- Réunion encadrant
- Rédaction de la V3 du cahier des spécifications
- Lecture articles wordspotting et OCR
- Début rédaction état de l'art sur le wordspotting

Compte rendu n°10 du 28/11/2018

- Réunion avec Monsieur Ramel
- Finalisation du cahier des spécifications
- Rédaction État de l'art et analyse (non terminé)
- Création du Gantt S10

Compte rendu n°11 du 5/12/2018

- Rédaction état de l'art WordSpotting et Reranking

Compte rendu n°12 du 9/01/2019

- Repérage des bugs dans l'application

Compte rendu n°13 du 16/01/2019

- Correction de bugs
- Premières tentatives de déplacement de l'appli hors VM

Compte rendu n°14 du 23/01/2019

- Dernières tentatives de déplacement de l'appli hors VM

Compte rendu n°15 du 30/01/2019

- Analyse API DivaServices

Compte rendu n°16 du 5/02/2019

- Début développement application DivaSegmentation

Compte rendu n°17 du 12/02/2019

- Suite des développements de l'application DivaSegmentation (partie JSON)

Compte rendu n°18 du 19/02/2019

- Suite des développements de l'application DivaSegmentation (partie HTTP)

Compte rendu n°19 du 5/03/2019

- Suite des développements de l'application DivaSegmentation (partie INPUT/OUTPUT)

Compte rendu n°20 du 12/03/2019
--

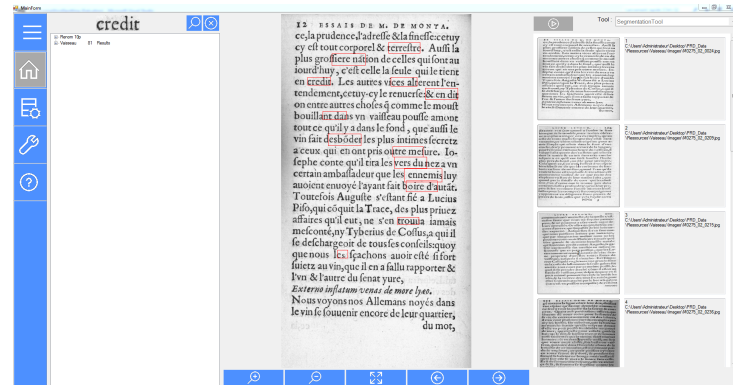
— Finalisation de l'application DivaSegmentation en appuyant sur la partie qualité
--

Compte rendu n°21 du 19/03/2019
--

— Mise en place du GitLab avec les sources des projets
--

Plateforme WordSpotting

La plateforme de wordspotting permet à des historiens ou lecteurs de rechercher un mot sur un document contenant du texte au format image. De nombreux outils sont utilisés par la plateforme pour effectuer des pré-traitements ou exécuter des algorithmes de recherche sur les images.



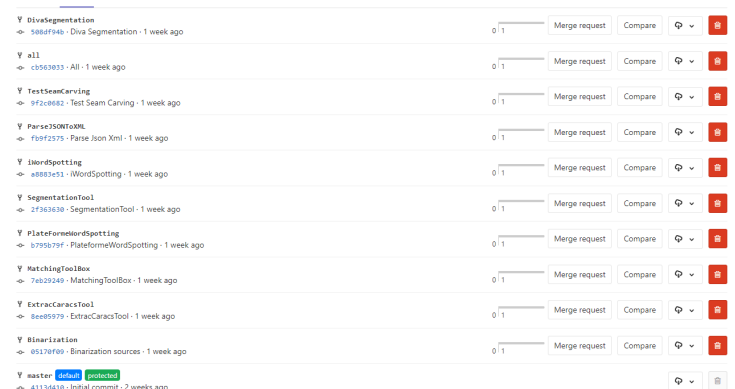
Objectifs

- Reprise de l'application et stabilisation
- Développement d'un outil de segmentation
- Restructuration du versionning avec GitLab



Conclusion

- Il est important de prendre du recul sur l'estimation des durées des tâches
- Il faut privilégier la qualité à la quantité
- Il est important de faciliter la reprise des projets



Amélioration d'un outil de word spotting

Dorian LE THAI BINH

Encadrement : Nicolas RAGOT

Plattformen Wordspotting

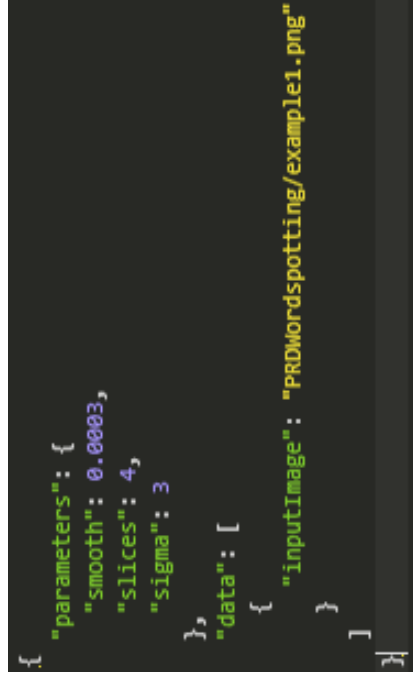
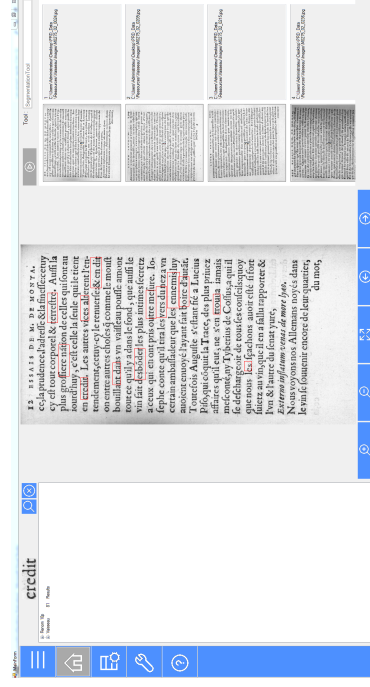
La plateforme de wordspotting permet à des historiens ou lecteurs de rechercher un mot sur un document contenant du texte au format image. De nombreux outils sont utilisés par la plateforme pour effectuer des pré-traitements ou exécuter des algorithmes de recherche sur les images.

Objectifs

- Reprise de l'application et stabilisation
- Développement d'un outil de segmentation
- Restructuration du versionning avec GitLab

Conclusion

- Il est important de prendre du recul sur l'estimation des durées des tâches
- Il faut privilégier la qualité à la quantité
- Il est important de faciliter la reprise des projets

[illegible]

Amélioration d'un outil de word spotting

Résumé

Le wordspotting est la science de la reconnaissance des mots. On utilise ses méthodes afin de retrouver des informations dans des documents au format image. Aujourd'hui il existe une application qui permet la recherche de mots ou de caractères sur une image mais également l'affichage de résultats. Les objectifs du projet sont la stabilisation de cette dernière et l'ajout de fonctionnalités visant à l'améliorer.

Mots-clés

Word Spotting, Binarisation, Segmentation, Normalisation, Reclassement, Indexation, L^AT_EX,

Abstract

The wordspotting is the science of word recognition. Its methods are used to retrieve information in image format documents. Today there is an application that allows the search for words or characters on an image but also the display of results. The objectives of the project are the stabilisation of the latter and the addition of features to improve it.

Keywords

Word Spotting, Binarization, Segmentation, Normalization, Reranking, Indexation