



**Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique**

Université de Sousse

Ecole Supérieure des Sciences et de la Technologie de Hammam Sousse

Licence Fondamentale en Sciences de l'Informatique

Rapport de Stage de Fin d'Etudes

TOPic

Reconnaissance Optique de Caractères

Réalisé par : Souhir M'RABET

Soutenu le 08/06/2015, devant le jury composé de :

Président de jury : M. Mounir HERGLI

Rapporteur : Mme. Maissa ELLEUCH

Encadrant universitaire : Mme. Lobna HLAOUA

Encadrant professionnel : Melle. Manel LAMINE

Année Universitaire : 2014-2015

*À mes parents, pour leur patience, encouragements et soutien
À toute ma famille et mes chers amis,
À tous ceux que j'aime et qui m'aiment.*

Souhir M'RABET

REMERCIEMENT

Je tiens en premier lieu à remercier Mme. Lobna HLAOUA et Melle. Manel LAMINE pour leurs conseils, leurs critiques constructives, leurs compétences et pour leur gentillesse qu'elles ont manifestées à mon égard. J'exprime aussi ma gratitude pour leurs accueils et leur confiance qui ont rendu ce travail assez intéressant. Merci enfin pour la patience et le soutien moral qu'elles m'ont apporté.

Je voulais aussi témoigner de ma profonde gratitude envers nos enseignants, pour leurs encouragements durant toutes les années d'études et formation à Ecole Supérieure des Science et de Technologie de Hammem Sousse.

Un grand remerciement également à tous les membres du jury qui m'ont fait l'honneur de présider le jury de ce projet et d'évaluer mon travail.

D'une façon générale, un grand merci à tous ceux qui m'ont aidé de près ou de loin à l'achèvement de ce projet de fin d'études et à l'élaboration du présent rapport.

RESUME

Dans le cadre de ce projet de fin d'études à l'Ecole Supérieure des Sciences et de Technologie de Hammem Sousse en vue de l'obtention du diplôme de licence fondamentale en informatique, j'ai effectué mon stage au sein de la société "Mahd" ayant pour sujet « Conception et réalisation d'une application Android permettant la traduction de texte capturé en image en intégrant l'OCR ».

Ainsi, le présent document constitue la synthèse du travail dans le cadre de ce projet qui a pour objectif de concevoir et de développer une application mobile supportée par la plateforme mobile Android et destinée aux voyageurs touristes pour traduction (affiches publicitaires, les panneaux de direction, les annonces, etc...).

J'ai étudié en premier lieu les domaines d'applications existantes ainsi que les concepts et les outils sur lesquelles se base mon travail. En second lieu, j'ai étudié les besoins des utilisateurs. Nous avons ensuite entamé la phase de développement pour concevoir l'application.

En ce qui concerne l'élaboration de l'application mobile, j'ai utilisé la plateforme Android.

A travers ce document, je décris plus en détails chaque étape de la réalisation de ce projet.

Mots clés: Application Mobile, Capture image, OCR, Traduction.

Table des matières

Introduction Générale.....	2
1 Présentation Générale.....	3
1.1 Introduction	4
1.2 Présentation de l'organisme d'accueil.....	4
1.2.1 Présentation.....	4
1.2.2 Historique	4
1.2.3 Fiche signalétique	4
1.2.4 Compétences logicielle	5
1.2.5 Références.....	5
1.3 Présentation du projet	5
1.3.1 Problématique	5
1.3.2 Solution.....	6
1.3.3 Critère de TOPic	7
1.3.4 Chronogramme prévisionnelle de la réalisation du projet.....	8
1.4 Conclusion.....	10
2 Domaine d'Application	11
2.1 Introduction	12
2.2 Historique	12
2.3 Les moyens d'acquisitions	12
2.4 Les applications	13
2.5 Description d'un OCR.....	13
2.5.1 Les différentes approches	13
2.5.2 Fonctionnement générale.....	15
2.5.3 Décomposition d'un OCR classique.....	16
2.6 Analyse de l'existant	17
2.7 Tesseract OCR.....	18

2.7.1	Historique	18
2.7.2	Architecture	18
2.7.2.1	Seuillage adaptatif.....	19
2.7.2.2	Détection de ligne et du texte	19
2.7.2.3	Reconnaissance du mot.....	20
2.7.2.4	Classification statique des caractères	21
2.7.2.5	Entrainement des données.....	21
2.8	Conclusion.....	21
3	Etude Conceptuelle	24
3.1	Introduction	23
3.2	Périmètre du projet	23
3.3	Besoin fonctionnelle.....	23
3.4	Besoin non fonctionnelle.....	24
3.4.1	Contrainte ergonomique	24
3.4.2	Contrainte technique.....	24
3.5	Conception.....	25
3.5.1	Diagramme de classes.....	26
3.5.2	Organigramme	26
3.6	Conclusion.....	27
4	Réalisation	28
4.1	Introduction	29
4.2	Besoins techniques	29
4.3	Environnement de travail	29
4.4	Interfaces	30
4.5	Tests.....	31
4.5.1	Filtres	30
4.5.2	OCR	32

4.5.3 Evaluation	33
4.5 Conclusion.....	33
Conclusion Générale	34
Bibliographie.....	36

Liste des Figure

Figure 1.2.1 : Logo « Mahd SUARL »	4
Figure 1.3-1 : Logo application TOPic	6
Figure 1.3-2 : Schéma technique	7
Figure 2.7.2 : Architecture Tesseract	18
Figure 2.7.2.1 : Exemple de seuillage adaptatif	19
Figure 2.7.2.2-1 : Détection de ligne et du texte	20
Figure 2.7.2.2-2 : Division du texte	20
Figure 2.7.2.3 : Reconnaissance du mot	20
Figure 3.2 : Diagramme de cas d'utilisation	24
Figure 3.4.1 : Organigramme	26
Figure 3.4.2 : Diagramme de classes	26
Figure 4.4-1 : Interface d'accueil	30
Figure 4.4-2 : Rogner image	30
Figure 4.4-3 : Image rogner	30
Figure 4.4-4 : Interface traduction.....	31
Figure 4.5.1-1 : Filtre Gaussien - Elimination de bruit	31
Figure 4.5.1-2 : Filtre Seuillage	31
Figure 4.5.1-3 : Filtre Gaussien et Seuillage	32
Figure 4.5.1-4 : Filtre Grayscale, Gaussien et Seuillage.....	32
Figure 4.5.2-1 : Test avec filtres	32
Figure 4.5.2-2 : Test avec filtres en augmentant le seuillage	32
Figure 4.5.2-3 : Seuillage Faible	33
Figure 4.5.2-2 : Seuillage fort	33

Liste des Tableaux

Tableau 1.3.2-1 : Diagramme de Gantt	9
Tableau 2.5 : Tableau comparative des OCR	18
Tableau 2.7.4: Les étapes de comparaison	21

Glossaire

OCR : Optical Character Recognition – Reconnaissance Optique de caractères

INTRODUCTION GENERALE

Dans la société moderne, nous comptons beaucoup sur des ordinateurs pour traiter d'énormes volumes de données. Relatif à cela et pour des raisons économiques ou des exigences d'affaires, il y'a une grande demande pour entrer rapidement toutes les informations imprimées et manuscrites dans l'ordinateur. Souvent, ces données existent sur papier et sont introduites dans l'ordinateur par des opérateurs humains.

De nos jours, l'être humain ne peut plus se décrocher de son smartphone, tablette et tout autre équipement technologique portable. Ces équipements ont été conçus pour pouvoir supporter de plus en plus d'applications dans plusieurs domaines. Les équipements technologiques est un support pour toute sorte de loisirs, d'aide et de service. La numérisation de l'image joue un rôle principal dans le monde du mobile. Dans ce cadre nous utilisons cette technologie dans le domaine du tourisme pour aider les voyageurs dans leurs découvertes des pays étranger qui écrivent et parlent des langues différentes, ainsi une nouvelle langue ne sera plus jamais un obstacle.

La technologie permet d'envisager des solutions techniques qui étaient jusqu'alors bannies. Ces nouvelles technologies présentent alors une foule d'avantages mais aussi une foule de problèmes ou d'inconvénients. Voilà pourquoi il est très important de s'informer sur tout ce qui a été réalisé pour résoudre des problématiques similaires et d'étudier consciencieusement chaque point. D'autre part, la mise en place d'un projet d'une telle ampleur nécessite une bonne organisation et une méthodologie. J'ai donc décidé de suivre une démarche d'ingénierie système.

C'est dans ce cadre s'inscrit mon projet de fin d'étude qui consiste essentiellement à implémenter un logiciel de reconnaissance optique de caractères pour principale fonction la récupération des informations nécessaires sur les panneaux routiers et pouvoir les traduire. L'application prévue doit être un outil efficace qui sera intégré dans un grand projet d'outils pour le tourisme en cours de réalisation par la société Mahd.

Le présent rapport sera scindé en quatre chapitres. Le premier chapitre présente le cadre générale de travail. Le deuxième chapitre donne un aperçu général sur le sujet d'étude en décrivant les solutions existantes et la solution proposée. Le troisième chapitre est consacré à la description détaillée des besoins fonctionnels et non fonctionnels ainsi qu'à l'explication de la conception de mon projet. Dans le dernier chapitre j'ai exposé l'environnement logiciel et les interfaces réalisées. A la fin je clôture mon rapport avec une conclusion générale.

CHAPITRE 1

PRESENTATION GENERALE

Sommaire

1.1	Introduction	4
1.2	Présentation de l'organisme d'accueil	4
1.2.1	Présentation.....	4
1.2.2	Historique	4
1.2.3	Fiche signalétique	4
1.2.4	Compétences logicielle	5
1.2.5	Références.....	5
1.3	Présentation du projet	5
1.3.1	Problématique	5
1.3.2	Solution.....	6
1.3.3	Critère de TOPic	7
1.3.4	Chronogramme prévisionnelle de la réalisation du projet	8
1.4	Conclusion	10

1.1 INTRODUCTION

Le présent chapitre a pour objectif de présenter l'entreprise accueillante au sein de laquelle j'ai effectué mon stage, d'exposer la problématique générale, et détailler l'enchainement des différentes activités tout au long de la réalisation de mon projet de fin d'étude.

1.2 PRESENTATION DE L'ORGANISME D'ACCUEIL

1.2.1 PRESENTATION



Figure 1.2.1 : logo « Mahd SUARL »

Ce projet est réalisé au sein de la société tunisienne de développement web et logiciel spécialisée dans des applications de gestion, d'une période de deux mois à partir de 01/04/2015.

1.2.2 HISTORIQUE

Créée en 2013 avec un effectif de 2 employés, la société dispose actuellement d'une équipe de 5 employés spécialisée dans le développement informatique.

La société Mahd a connu dès sa première année d'existence une croissance qui l'a propulsé à l'évolution inévitable sur le marché nationale et internationale.

1.2.3 FICHE SIGNALÉTIQUE

Raison Sociale : Mahd SUARL

Numéro de registre de commerce : B 91116302013

Matricule Fiscal : 1302387WMA000

Date de création de la Société : 2013

Nom et Prénom du Directeur Général : Mohamed DARDOURI

Potentiel Technique : 5 employés qui représente tous l'effectif technique (concepteurs, développeurs, formateurs)

Siège social : 378 Rue Farhat Hached Kalaa Kebira, Sousse – 4060, Tunisie

Téléphone : 73 34 91 19

Mobile : Mohamed DARDOURI 93 62 56 49

1.2.4 COMPETENCES LOGICIELLES

Depuis 2013, et dans un souci d'être toujours parmi les meilleures, la société a opté pour une stratégie d'utilisation de nouveaux concepts de conception et développement dont je peux citer:

- Web: WordPress, Drupal, angularJs, Symfony
- Desktop:C#, VB.NET, ASP.NET, SharePoint, Visual Studio
- Application Mobile: Android

1.2.5 REFERENCES

Les logiciels développés par Mahd sont exploités dans une grande diversité de secteurs et de domaines à savoir : le secteur social, touristique, industriel, des travaux publics, de la distribution, de l'administration, des services etc.

1.3 PRESENTATION DU PROJET

1.3.1 Problématique

De nos jours, les voyages et la découverte des différentes cultures est un choix donné à tout le monde c'est-à-dire toute personne peut voyager et découvrir le monde mais le problème qui se pose toujours c'est le moyen de communication avec ceux que nous allons rencontrer.

Un pays étranger mais aussi une langue étrangère et nous devons minimiser les risques et les conséquences négatives d'une incompréhension ou une mauvaise interprétation.

Ce projet est l'un des supports de communication pour les étrangers.

Supposons que vous êtes dans un pays étranger et vous voulez savoir quel direction vous allez prendre pour atteindre une destination bien déterminée, donc vous avez besoin de traduire un panneau routier. Deux solutions s'offre à vous.

- La première consiste à réécrire le panneau sur un site de traduction en le tapant au clavier adapté, une tâche très difficile pour nous de saisir un texte au clavier, genre du chinois ou du japonais.
- La deuxième consiste à utiliser un logiciel de reconnaissance de caractère qui analyse l'image et transforme les caractères écrits en caractères numériques et les traduit par la suite. il très utile de pouvoir traduire un panneau de direction ou menu écrit en allemand en faisant une simple photo plutôt qu'en recopiant laborieusement les mots un par un.

Ces logiciels, plus communément appelé OCR, ont donc pour but de transformer un texte écrit en une représentation interprétable par une machine. Par la suite, ils doivent garantir une traduction précise et compréhensible du texte trouvé.

1.3.2 SOLUTION



Figure 1.3-1: Logo application TOPic

La société Mahd propose l'application TOPic Text Grabber pour les systèmes Android.

Cette application combine la reconnaissance de caractères et la traduction de texte. L'application est entièrement accessible et facile à utiliser. Prendre une photo d'un texte que vous souhaitez traduire, marquez du doigt la zone intéressante et vous aurez une traduction. Elle est un support efficace aux voyageurs. L'application ne coûte rien, téléchargeable gratuitement, il s'agit d'un logiciel libre « open source ».

L'application peut reconnaître le texte même lorsque l'image est inclinée ou n'est pas assez claire ce qui est vraiment pratique dans le cas où vous êtes dans un moyen de transport et vous voulez vérifier votre emplacement. Elle dispose aussi d'un accès à la galerie des images activable depuis l'application au cas où vous voulez traduire une ancienne image.

Cette application intègre également une traduction s'appuyant sur le module de Microsoft. La qualité de traduction est plutôt approximative mais elle peut rendre un service acceptable.

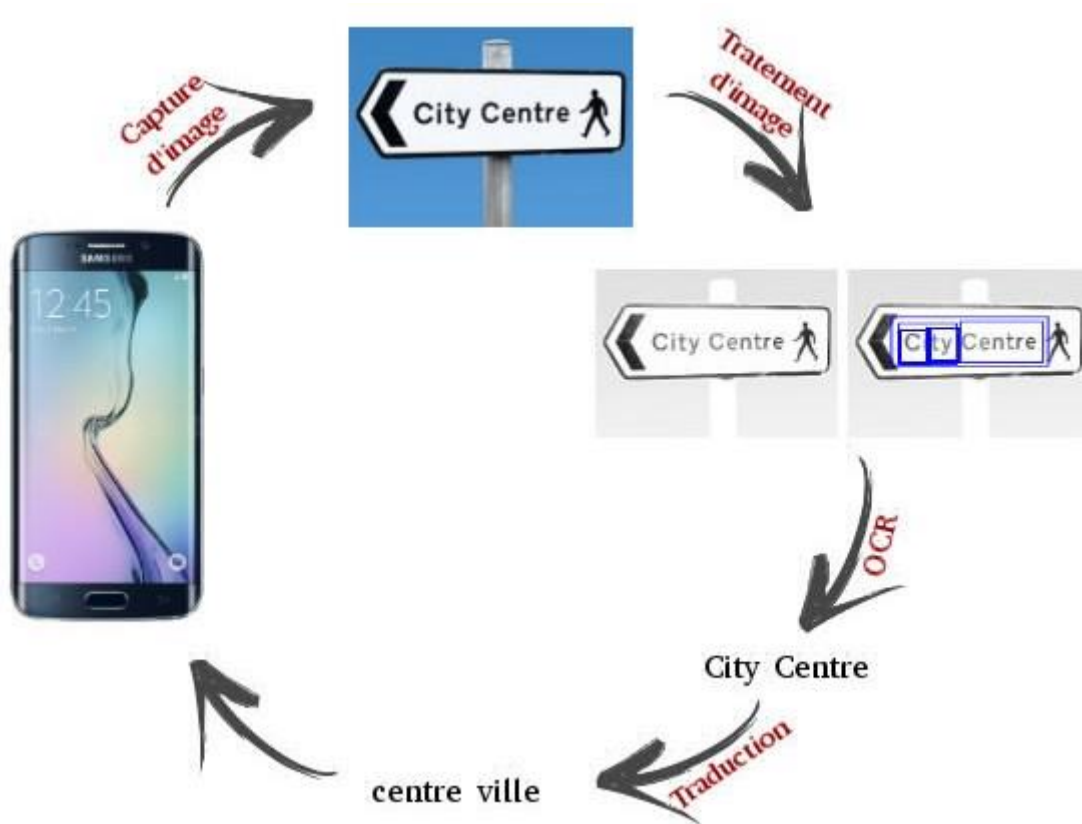


Figure 1.3-2: Schéma Technique

1.3.3 CRITERE DE TOPIC

Dans TOPic, les fichiers image que vous intégrez sont analysés, puis convertis en texte à l'aide d'algorithmes informatiques.

Pour un résultat optimal, j'ai ajouté des modules de traitement d'image, et comme résultat finale les images doivent respecter un certain nombre de critères :

- **Résolution** : les fichiers à haute résolution donnent des meilleurs résultats. En règle générale, il est préférable de capturer une image de haute résolution (>5 MegaPixel), et dans tous les cas j'ai ajouté des filtres pour une meilleure détection des caractères.
- **Orientation** : même si l'image capturée n'est pas correctement orientée vous pouvez passer faire la reconnaissance optique où mon application est capable de détecter les angles d'inclinaison.

- **Rognement** : pour assurer une détection efficace et traduction correcte un outil de rognement sera à votre disposition pour encadrer seulement la zone d'écriture.
- **Langues, polices et jeux de caractères** : le moteur de reconnaissance optique des caractères est compatible avec différents jeux de caractères. Les résultats sont de meilleure qualité si votre image contient des polices courantes telles qu'Arial ou Times New Roman.
- **Qualité de l'image** : les images nettes avec une luminosité homogène et un bon contraste donnent les meilleurs résultats. La reconnaissance des caractères est moins performante avec les images souffrant d'un flou de mouvement ou d'une mise au point médiocre.
- **Langues, polices et jeux de caractères** : le moteur de reconnaissance optique des caractères est compatible avec différents jeux de caractères. Les résultats sont de meilleure qualité si votre image contient des polices courantes telles qu'Arial ou Times New Roman.
- **Qualité de l'image** : les images nettes avec une luminosité homogène et un bon contraste donnent les meilleurs résultats. La reconnaissance des caractères est moins performante avec les images souffrant d'un flou de mouvement ou d'une mise au point médiocre.

1.3.4 CHRONOGRAMME PREVISIONNEL DE LA REALISATION DU PROJET

L'enchaînement des différentes activités dans le cadre d'un projet informatique peut se faire selon différents modèles, qui enchainent ces activités en phase plus ou moins successives. La phase de collection des informations et la création de cahier de charge ainsi que la conception UML s'est étalée sur une neuf jours au cours desquelles j'ai appris la manipulation des outils de développement de l'application. La seconde tache intitulée analyse des API d'OCR, l'intégration et le développement s'est prolongée sur vingt-quatre jours. Par la suite, j'ai consacré onze jours pour la troisième phase de recherche, intégration et développement des modules de traitement d'image. La quatrième tâche c'était la dernière pour finir l'application TOPic où on a développé le module de traduction afin d'assurer la réussite de l'application ainsi que l'adéquation des fonctionnalités de cette application aux besoins des utilisateurs. Les tests ont été effectués parallèlement avec la phase de développement vers la fin de chaque module pour assurer un fonctionnement fluide et sans erreur de l'application.

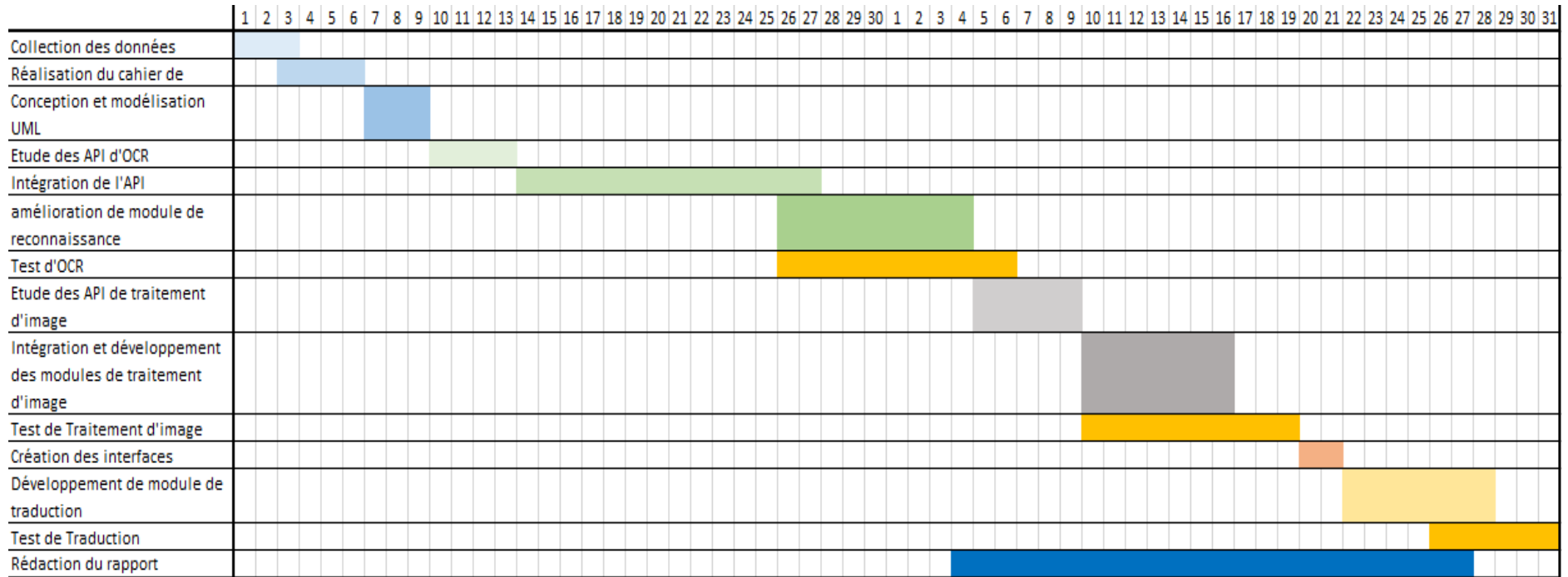


Tableau 1.3.2 : Diagramme de Gant

1.4 CONCLUSION

Après avoir présenté mon projet d'une manière générale, je vais illustrer dans ce qui suit les différentes techniques de TOPic et les supports de la reconnaissance optique de caractères qu'elles mettent en œuvre ainsi que les solutions existantes actuellement.

CHAPITRE 2

DOMAINE D'APPLICATION

Sommaire

2.1	Introduction	12
2.2	Historique	12
2.3	Les moyens d'acquisitions	12
2.4	Les applications	13
2.5	Description d'un OCR.....	13
2.5.1	Les différentes approches	13
2.5.2	Fonctionnement générale.....	15
2.5.3	Décomposition d'un OCR classique.....	16
2.6	Analyse de l'existant	17
2.7	Tesseract OCR.....	18
2.7.1	Historique	18
2.7.2	Architecture	18
2.7.2.1	Seuillage adaptatif.....	19
2.7.2.2	Détection de ligne et du texte	19
2.7.2.3	Reconnaissance du mot.....	20
2.7.2.4	Classification statique des caractères	21
2.7.2.5	Entrainement des données.....	21
2.8	Conclusion.....	21

2.1 Introduction

Pour élaborer mon projet je suis reposé sur divers concepts, technologies et outils. Dans ce chapitre je présente des notions de base que je vais exploiter dans les chapitres suivants. En plus je présente une étude de l'existant concernant la thématique de mon projet.

2.2 Historique

Le concept d'OCR en lui-même est relativement ancien, son origine remonte aux années 1900 au cours desquelles on inventa le scanner à balayage pour la télévision et les machines à lire. Tuyrin développa alors la première application d'aide aux handicapés visuels, mais il a fallu encore attendre 1940 pour voir se réaliser la première version informatique de cette application. Depuis, une gamme importante de logiciels sont apparus avec de très bonnes performances.

Les premières applications étaient essentiellement orientées vers la reconnaissance de chiffres imprimés et de quelques lettres anglaises (latines). Ensuite, les applications ont été étendues progressivement vers la reconnaissance de caractères isolés, les caractères manuscrits isolés, les textes mono-fontes, les textes multi-fontes et enfin le manuscrit.

Plus tard, l'introduction des réseaux de neurones permit une nouvelle avancée dans le monde de la reconnaissance de caractère. Cette nouvelle technologie a permis d'améliorer encore les résultats des OCR.

De nos jours nous disposons de systèmes fiables avec un haut taux de reconnaissance même sur des textes composites (c'est-à-dire composées de textes et d'images) et il est courant de trouver des systèmes« intelligents » qui peuvent reconnaître la plupart des polices avec un haut niveau de précision.

Pour finir je peux dire à ce jour que la reconnaissance optique de caractère est un enjeu pour l'industrie et que même avec un haut taux de reconnaissance elle reste un secteur de recherche très actif surtout dans le domaine de l'analyse de l'écriture manuscrite. [1]

2.3 Les moyens d'acquisitions

Le traitement d'un texte ne peut se faire sans une acquisition préalable de l'image du texte. Cette première étape est très importante car une image de mauvaise qualité risque de créer des erreurs. Le choix du capteur est donc particulièrement important suivant l'utilisation et le taux d'erreurs désirés. Il existe un grand nombre de formes différentes de capteur dont voici les

plus courants :

- Scanner (plat ou manuel)
- Appareil photo numérique
- Caméra
- Tablette graphique

2.4 Les applications

Il existe de très nombreuses applications pour ces logiciels donc je n'en citerai qu'une petite partie :

- La numérisation et traduction des panneaux
- La numérisation de livres permet la sauvegarde de livres anciens ou la diffusion de livre au format électronique.
- Le tri postal : utilisation d'un OCR pour déterminer l'adresse.
- La reconnaissance de plaque minéralogique lors des contrôles routiers.
- L'authentification des chèques de banques : analyse de la signature pour permettre de valider l'identification du signataire
- La reconnaissance d'écriture pour un PDA: interprétation des informations inscrites à l'aide du stylet.

2.5 Description d'un OCR

Maintenant que nous avons vu quelles sont les utilisations possibles d'un OCR je vais m'intéresser plus précisément à son mode de fonctionnement.

2.5.1 Les différentes approches

Cette première partie permet de cibler plus précisément les différentes approches possible pour un OCR car il est évident que les traitements ne seront pas les mêmes s'il faut traiter une écriture manuscrite et une écriture imprimée.

Gestion des couleurs de l'image

Par défaut un texte est bicolore, mais dans le cas de documents complexes contenant des images on peut avoir besoin d'une image en niveaux de gris ou en couleur pour restituer de la façon la plus fidèle possible le document source.

Les images en noir et blanc sont les plus simples à traiter et les moins coûteuses en espace,

c'est donc la solution la plus généralement utilisé pour numériser du texte.

Les traitements sur une image en couleur ou en niveaux de gris sont plus complexes car ils impliquent la gestion d'un seuillage pour l'analyse du texte. Le seuillage est une opération qui consiste à considérer comme noir les pixels dont la valeur est en dessous d'un certain seuil et à considérer comme blanc tous les autres.

Selon le type de documents à traiter il faudra prendre en compte la gestion de la couleur dans les algorithmes de traitement et dans ceux de gestion de la mémoire. La gestion de la couleur peut être effectuée simplement par un seuillage lors du chargement en mémoire de l'image.

Imprimé ou manuscrit?

L'approche entre l'étude de l'écriture manuscrite et de l'imprimé est très différente. Dans le cas de l'imprimé, les caractères sont réguliers et correctement alignés alors que dans l'autre cas il faut pouvoir reconnaître des formes variables en fonction de la personne qui a écrit le caractère. Dans mes études et pour ne pas dépasser le cadre du sujet je me contenterai de traiter le cas des caractères imprimés (en effet il est plutôt rare de trouver des panneaux rédigés à la main). [5]

Reconnaissance mono-fonte, multi-fonte ou omnifonte

Tout d'abord, une fonte de caractères, en typographie, est un ensemble de glyphes¹, c'est-à-dire de représentations visuelles de caractères, d'une même famille, de même style, corps et graisse. Il ne faut pas confondre la fonte et la police qui regroupe tous les corps et graisses d'une même famille. Un OCR est dit mono fonte s'il n'est capable que de reconnaître qu'une seule fonte, il est dit multifonte s'il est capable de reconnaître plusieurs fontes et enfin il est omnifonte s'il peut reconnaître n'importe quel type de fonte sans apprentissage préalable. [5]

Reconnaissance en-ligne ou hors-ligne

Cette différence possède une importance particulière, il permet de choisir vers quels types de traitement s'orienter. Les traitements utilisés pour l'acquisition en ligne doivent être beaucoup plus rapide que ceux utilisés dans le hors-ligne car il faut pouvoir comprendre en temps réel les informations transmises par l'utilisateur, ce qui implique des algorithmes moins lourd et donc moins efficaces. A l'inverse un traitement hors-ligne permet une plus grande liberté quant au temps d'analyse. [5]

¹ Glyphes : c'est-à-dire de représentations visuelles de caractères

L'apprentissage

L'apprentissage est la partie qui permet à l'OCR de retenir des détails qui lui permettront par la suite d'améliorer la reconnaissance, par exemple un nouveau caractère. Ce module est assez particulier car, de nos jours, il est courant de voir des OCR se passant de cette partie grâce à l'utilisation de système intelligent qui peut reconnaître la plupart des polices avec un haut niveau de précision. De plus selon le type d'apprentissage choisi il peut ne pas avoir à intervenir dans le traitement, comme dans le cas d'une base de données de caractère par exemple où l'apprentissage se fait de manière statique, ou au contraire interroger l'utilisateur pour combler un manque de données.

2.5.2 Fonctionnement générale

La reconnaissance des caractères

Tous les documents peuvent être décomposés en entités plus petites comme des paragraphes ou des lignes, mais ici on ne s'intéresse qu'à la plus petite d'entre elles : le caractère. Ces modules ont pour objectif le traitement de chaque image de caractère en vue de son identification.

Trois techniques sont utilisées :

- La comparaison par modèles : c'est une technique qui consiste à comparer l'image du caractère avec des modèles pré-enregistrés.
- Extraction de propriétés : c'est un ensemble de techniques qui consistent à déterminer les propriétés géométriques des caractères à reconnaître, telles que les fins de segments, les angles prononcés, les croisements, etc.
- Méthodes structurelles : c'est un ensemble de méthodes dont le principe consiste à exprimer une lettre par une combinaison d'attributs généraux, comme par exemple, le rapport entre la hauteur et la largeur du caractère, etc.

Ces modules peuvent aussi s'appuyer sur les informations obtenues à partir du document pour améliorer leurs capacités de reconnaissance. En effet, un caractère n'est jamais seul il est toujours inclut dans un contexte. Une zone de texte pourra contenir tous les caractères existant alors qu'une zone de chiffre ne contiendra que les alpha-numérique. Cette action diminue le nombre de comparaisons à effectuer. Ces modules peuvent inclure des phases d'apprentissage si nécessaire.

2.5.3 Décomposition d'un OCR classique

Classiquement on pourrait décomposer un OCR en 5 modules :

1. la numérisation
2. le pré-traitement de l'image
3. l'analyse
4. la reconnaissance
5. le post-traitement

Numérisation

Ce module est la toute première étape qui consiste à récupérer une image. On peut aussi bien utiliser un capteur optique que charger un fichier préalablement numérisé. A sa sortie on dispose d'une image en mémoire. [1]

Pré-traitement de l'image

Ce module n'est pas obligatoire mais il est très utile car, il contient tous les pré-traitements effectués sur l'image pour la préparer aux autres modules. Il permet de diminuer les erreurs de reconnaissances en améliorant la qualité de l'image, par exemple en réduisant le bruit ou en corrigeant son inclinaison. A sa sortie on dispose d'une image claire et suffisamment propre pour être correctement analysée. [1]

Analyse

L'analyse fait partie de la reconnaissance de document. C'est l'étape qui permet de reconnaître les différentes parties d'une image. Elle détecte les parties importantes de l'image comme les différentes zones de textes, les images et la mise en page. Elle s'occupe ensuite de segmenter les zones de textes en plus petites parties pour finalement isoler chaque caractère. Ce module peut également contenir un sous module permettant d'analyser les spécificités du texte comme la police ou la taille. Il fournit en sortie une liste de caractères à analyser et peut aussi fournir la structure du document. [1]

Reconnaissance

La reconnaissance fait partie de la reconnaissance de caractères. C'est la partie qui va analyser l'image d'un caractère pour pouvoir le reconnaître, il est le premier module à traiter les caractères.

Ce module peut inclure une phase d'apprentissage si l'algorithme utilisé nécessite une base de données de caractères pour la reconnaissance. Cependant, Il existe aujourd'hui certains algorithmes se passant totalement d'apprentissage. A la fin du traitement tous les caractères ont été analysés et ceux ayant été reconnus disposent d'une étiquette signalant la valeur reconnue. [1]

Post-traitement

Le post-traitement gère la mise en forme des caractères reconnus précédemment, il a pour tâche de créer le document de sortie. Il doit convertir les données récupérées au format demandé et reproduire la mise en page si nécessaire. Il peut inclure une phase de vérification des mots à l'aide d'un dictionnaire qui permet de corriger les erreurs commises lors de l'identification d'un caractère ou de compléter les mots incomplets. C'est la dernière étape du traitement global et en sortie on dispose du texte reconnu sous le format désiré. [1]

2.6 Analyse de l'existant

Quelques logiciels de reconnaissance optique de caractères

Solutions commerciales

- FineReader de Abbyy, leader mondial des logiciels d'OCR.
- Readiris de I.R.I.S., logiciel précis et rapide, OCR en Arabe, Persan, Hébreu et langues asiatiques!
- OmniPage de Nuance (ex Scansoft).
- PDFCompressor de CVISION, très précis.

Solutions à sources ouvertes

- Tesseract (Unix) est un ancien OCR fabriqué par Hewlett Packard puis abandonné. Il est entré en 2005 dans le domaine public.
- Clara OCR.
- FreePress (Windows).
- GOCR (Unix, Windows), un framework .
- Conjecture (Unix), un autre framework se basant sur GOCR.
- Ocre (Unix)

Name	Founded year	License	Online	Windows	Mac OS X	Linux	BSD	Programming language	SDK	Language
Tesseract	1985	Apache	No	Yes	Yes	Yes	Yes	C++, C	Yes	35+
ABBYY FineReader	1989	Proprietary	Yes	Yes	Yes	Yes	Yes	C/C++	Yes	198
ExTypeReader & RTK	1987	Proprietary	Yes	Yes	Yes	Yes	Yes	C/C++	Yes	21
GOCR	2000	GPL	Yes	Yes	Yes	Yes	Yes	C	?	?

Tableau 2.5 : Tableau comparatif des OCR

2.7 Tesseract OCR

2.7.1 Historique

Tesseract OCR était un projet de recherche open source au sein de laboratoire du HP entre l'année 1984 et 1994 apparait pour la première fois en 1995 au conférence UNLV (University of Nevada Las Vegas) annuel pour le test de précision d'OCR en tant qu'un projet d'une thèse qui a eu comme but le développement d'un module pour les scanners HP. En 2005 HP a réalisé Tesseract pour la communauté OpenSource mais il n'a états jamais utilisé sur ses produits. Depuis 2006, Tesseract a été largement améliorée par Google. Il est distribué sous la licence Apache 2.0. [6]

2.7.2 Architecture

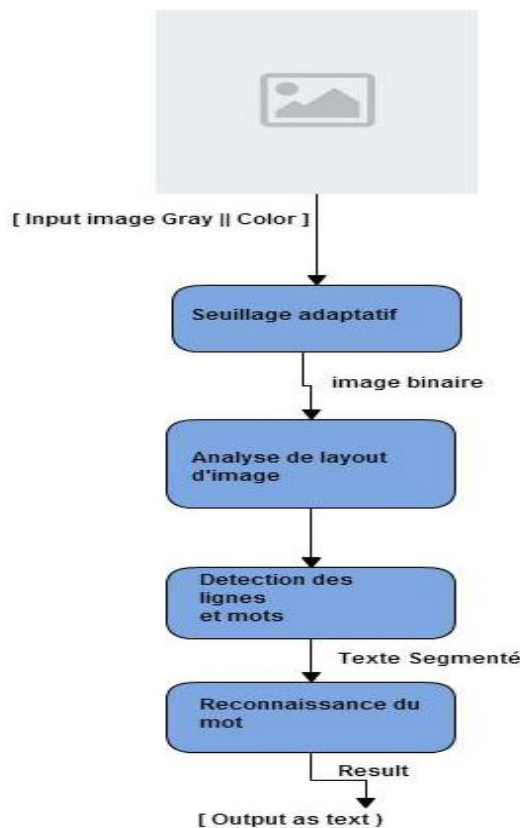


Figure 2.7.1 : Architecture Tesseract [3]

2.7.2.1 Seuillage adaptatif (Adaptive threshold)

Seuillage est le moyen le plus simple pour segmenter les objets d'arrière-plan. Si l'arrière-plan est relativement uniforme, alors on utilise une valeur globale de seuil pour binariser l'image en se basant sur l'intensité de pixel. Si il y'a une grande variation dans l'intensité de fond, un seuil adaptatif (aka seuillage local ou dynamique) peut produire des meilleurs résultats qui calcule les seuils dans les régions entourant chaque pixel. Chaque valeur de seuil est la moyenne pondérée du voisinage local moins une valeur de décalage.

Le moyen le plus utilisé pour trouver le seuil c'est de trouver le seuil locale est d'examiner statistiquement les valeurs d'intensité du quartier local de chaque pixel. La statistique la plus appropriée dépend en grande partie de l'image d'entrée. Fonctions simples et rapides comprennent la moyenne de la distribution d'intensité locale

$$T = \text{mean}$$

La valeur de médiane

$$T = \text{median}$$

Ou le moyen de minimum ou maximum des valeurs

$$T = \frac{\max + \min}{2}$$



Figure 2.7.2.1 : Exemple de seuillage adaptatif

2.7.2.2 Détection de ligne et du texte

L'algorithme est programmé de façon que même si la page est mal positionnée ou mal alignée ça ne pose aucun problème.

Après une analyse de layout d'image, on va se trouver avec des zones sauvages des textes, en appliquant un simple filtre qui va supprimer les caractères verticalement touchantes. Les blobs filtrés sont plus petits que la fraction de la hauteur médiane qui peut être une

punctuation ou un bruit. Après cette phase Tesseract traite les blobs filtrés horizontalement et l'assigne à une ligne de texte unique et puis il la rend à sa position normale. Dès que les lignes sont détectées, elles sont traitées par la méthode la plus utilisée qui analyse 4 lignes parallèles pour mieux reconnaître et organiser le texte : ligne de base, ligne descendeur, ligne moyen et ligne bloqueur. [2]

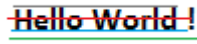


Figure 2.7.2.2-1 : détection de ligne et du texte

Après la détection de zone texte, cette API divise l'image en caractère de même hauteur après avoir trouvé un « pitch »² fixe et il le fixe pour l'étape de reconnaissance de mot. Si l'API ne trouve pas de « pitch » fixe il mesure les lacunes entre ligne de base et ligne moyen. [2]



Figure 2.7.2.2-2 : division du texte

2.7.2.3 Reconnaissance du mot

Dans l'étape précédente, Tesseract a segmenté juste les mots avec un « pitch » fixé, et pour les mots non-fixés, il essaye de les hacher pour avoir des résultats plus cohérents basant sur des points concaves des blobs mais ce processing élimine tous les points concaves opposés aux lignes segmentées qui vont être ignorées. [2]



Figure 2.7.2.3 : reconnaissance du mot

Si le résultat est encore médiocre, les blobs cassés ou déconnectés va être groupés en des caractères candidats avec l'algorithme de A*. [2]

² Pitch : terrain (ou « monospaced » espacement fixe) des polices, pitch fait référence au nombre de caractères imprimés par pouce. Pitch est une caractéristique d'une police à espacement fixe.

2.7.2.4 Classification statique des caractères

L'utilisation des segments de la polygonal tant que caractéristique était une solution aussi pour améliorer la reconnaissance mais elle n'a pas donné de résultat bien déterminé donc on passe à une méthode de comparaison d'un à plusieurs comme cette figure montre : [3]




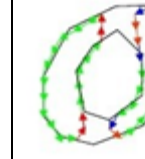
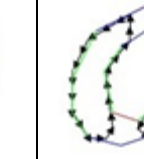
				
Prototype	Caractère à classifier	Caractéristique extraité	Comparaison de prototype à caractéristique	Comparaison de caractéristique à prototype

Tableau 2.7.2.4 : Les étapes de comparaison [3]

2.7.2.5 Entraînement des données

Pour améliorer la précision et étendre la limitation du langage, Tesseract conçu pour être entièrement recyclable. Après segmentation des caractères, une tentative en deux passes est faite pour reconnaître chaque mot à son tour. Le caractère satisfaisant est transmis au classificateur dynamique en tant que données d'entraînement. Dans la seconde passe, les mots qui ne sont pas reconnus de manière satisfaisante sont à nouveau reconnus.

Les utilisateurs peuvent trainer Tesseract pour améliorer la précision de la reconnaissance pour leurs propres cas. [2]

2.8 Conclusion

Après avoir analysé le domaine d'application et quelques applications existantes.

Dans le chapitre suivant, je vais spécifier les besoins fonctionnels et non fonctionnels de mon application on se basant sur le diagramme de cas d'utilisation.

CHAPITRE 3

ETUDE CONCEPTUELLE

Sommaire

3.1	Introduction	23
3.2	Périmètre du projet	23
3.3	Besoin fonctionnelle	23
3.4	Besoin non fonctionnelle	24
3.4.1	Contrainte ergonomique	24
3.4.2	Contrainte technique	24
3.5	Conception.....	25
3.5.1	Diagramme de classes.....	26
3.5.2	Organigramme	26
3.6	Conclusion.....	27

3.1 INTRODUCTION

L'analyse des besoins conditionne la réussite de la réalisation d'un projet dans la mesure où elle définit les besoins réels des futurs utilisateurs. La phase de communication et d'échange, est souvent le reflet du résultat final. C'est l'une des phases les plus difficiles de réalisation de projet. Il existe des besoins exprimés naturellement, des besoins implicites, des besoins inavoués et tout simplement des besoins dont les utilisateurs n'ont même pas conscience.

La première étape pour implémenter une application informatique est l'étude des besoins des clients et leurs attentes. Ainsi, ce chapitre sera dédié à l'extraction des besoins fonctionnels et non fonctionnels des utilisateurs. Cette étape me permettra, par la suite de déduire les fonctionnalités à intégrer au sien de l'application à développer.

3.2 PERIMETRE DU PROJET

Cette application destinée principalement au:

- Voyageurs

Au deuxième lieu au:

- Publique qui veut faire des traductions des phrase ou texte
- Polyglotte qui veut découvrir des nouvelle langues et fait le partage de l'information (nouveau mot/phrase à découvrir)

3.3 BESOINS FONCTIONNELS

La future application doit fournir aux utilisateurs :

- Accès à la galerie d'images ou appareil photo pour choisir ou capturer l'image à traduire
- Choix des langues du texte à traduire et de traduction
- Un texte numérisé à partir de l'image détecté
- Un texte de traduction précis
- Partage sur les réseaux sociaux

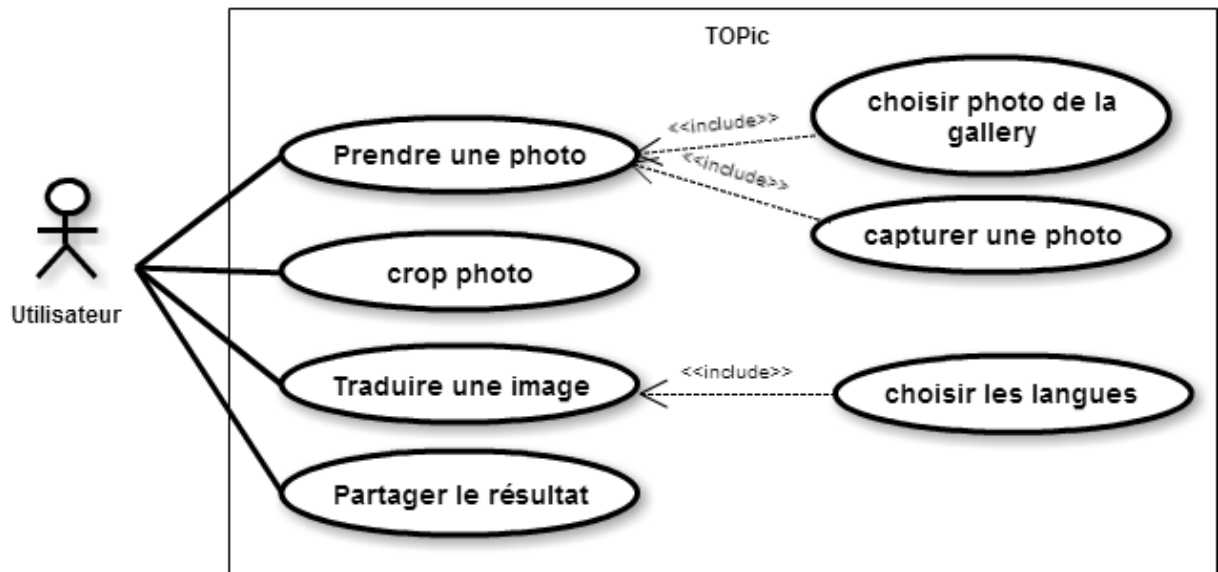


Figure 3.2: Diagramme de cas d'utilisation

La figure 3.2 : illustre les étapes que les utilisateurs suivent dans le cas d'utilisation de l'application TOPic. Lorsqu'un utilisateur lance l'application, il peut choisir de capturer une photo ou de la choisir du gallery pour enfin prendre une photo où s'écrit le texte souhaitant sa traduction, une fois l'étape mot-entrée est terminée, le mot-entrée sera traduit dans la langue attendue déjà choisie par l'utilisateur.

3.4 BESOIN NON FONCTIONNELS

3.4.1 CONTRAINTE ERGONOMIQUE

- Une interface simple et facile à utiliser
- Traduction exacte

3.4.2 CONTRAINTE TECHNIQUE

- L'application doit être toujours fonctionnelle
- Temps de réponse minimum.
- Le code doit être extensible et maintenable pour faciliter toute opération d'amélioration ou d'optimisation.

- **Traitement d'image :** Concrètement, il s'agira de transformer l'image d'origine en la faisant pivoter dans le cas où l'image n'est pas droite (pour une reconnaissance ultérieure plus facile des caractères) et de faire ressortir le tracé des caractères, c'est-à-dire : Gommer le bruit de fond et amplifier les marques des caractères. Il faudra donc envisager différentes méthodes de rotation d'images et de détection d'inclinaison du texte. Il faudra aussi trouver une méthode pour discerner le bruit de fond des caractères. Dans le cas idéal, l'image qui sortira de ce traitement sera un fichier bitmap sur lequel les caractères seront en noir sur fond blanc et mis horizontalement. Selon l'état des images fournies à traiter, il sera difficile, ou plutôt inutile, de vouloir un fond absolument blanc.
- **Analyse et extraction des caractères:** Cette étape sera réalisée à partir de l'image «parfaite» issue du prétraitement de cette dernière. Dans l'optique de réaliser cette dernière, il conviendra de créer des algorithmes capables de détecter les lignes puis les caractères découpés par l'étape précédente et de les stocker en mémoire.

Le principe en lui-même est assez simple puisque, suite au traitement de l'image, les caractères seront de couleur foncée et le fond de couleur claire. Il saura donc d'isoler les pixels foncés pour délimiter la zone de texte à traiter. Le travail se fera alors en plusieurs étapes. Tout d'abord, la délimitation des lignes que comporte le texte en les délimitant en hauteur. Ensuite viens le repérage des caractères dans chaque ligne en les délimitant en largeur dans un premier temps puis en hauteur dans un second temps. Toutes les informations ainsi récoltées seront transmises au système de reconnaissance.

- **Apprentissage :** L'apprentissage est une phase clef de la reconnaissance mais peut être aussi utile pour la segmentation. Un apprentissage continu : c'est l'ajout de caractère dans la base de connaissances s'il n'a pas été reconnu pendant la reconnaissance.

3.5 CONCEPTION

Dans cette section, je vais présenter la conception de l'application. En premier lieu, le diagramme de classe, puis le chronogramme en décrivant la démarche de l'application.

3.5.1 DIGRAMME DE CLASSES

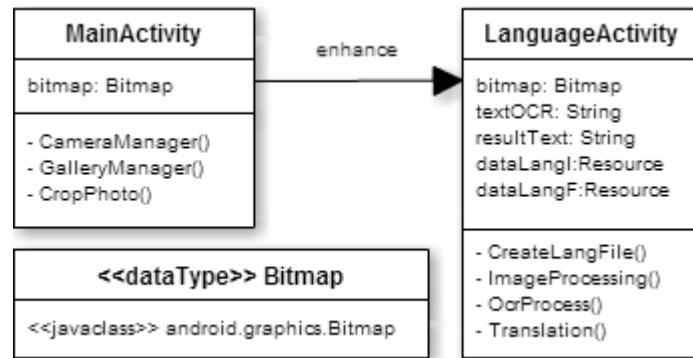


Figure 3.4.2: Diagramme de classes

La figure 3.4.2 : Lorsque l'utilisateur accède à MainActivity deux fragments s'ouvrent avec la camera et le crop par la suite sera le renvoie automatique vers LangueActivity qui ouvre trois fragments de traitement d'image, l'OCR et la traduction.

3.5.2 ORGANIGRAMME

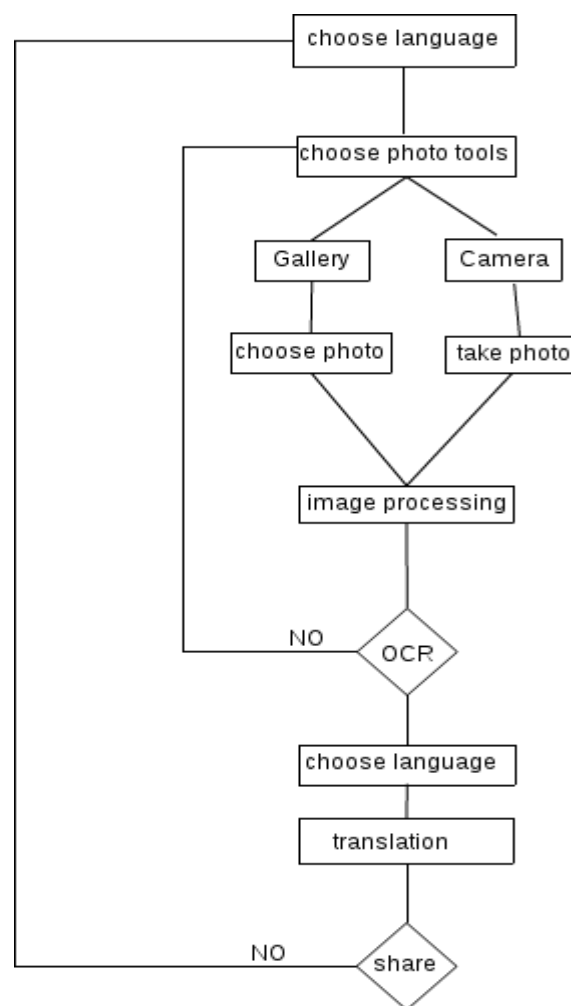


Figure 3.4.1: Organigramme

La figure 3.4.1 : décrit le traitement basique avec un organigramme. Quand le choix de photo est fini un traitement d'image se lance. Par la suite, on passe à la reconnaissance optique de caractères et on finira par le choix des langues et la traduction ainsi que la possibilité de publier le résultat.

3.6 CONCLUSION

A la fin de ce chapitre, j'ai abouti à l'expression des différents besoins fonctionnels et non fonctionnels de cette application ainsi les cas d'utilisations pour mieux comprendre la situation de l'acteur. Ensuite, j'ai décrit le processus de conception de mon application, je me suis focalisée sur le diagramme de classes pour fournir une vue global de mon système en présentant les classes et les relations entre elles. Dans la suite je présente la partie réalisation où se trouve l'évaluation ainsi que quelques captures d'écran de mon application.

CHAPITRE 4

REALISATION

Sommaire

4.1	Introduction	29
4.2	Besoins techniques	29
4.3	Environnement de travail	29
4.4	Interfaces	30
4.5	Tests.....	31
4.5.1	Filtres	30
4.5.2	OCR	32
4.5.2	Evaluation	33
4.5	Conclusion.....	33

4.1 INTRODUCTION

Après avoir achevé l'étape de la conception, j'entame dans ce chapitre la partie réalisation. Je commence par décrire les environnements de développement logiciel. Ensuite, je détaille le processus de développement de l'application. Enfin, je présente quelques captures d'écrans de mon application ainsi développée.

4.2 BESOINS TECHNIQUES

- API OCR: reconnaissance optique de caractères.
- Android NDK (Native Development Kit): Un outil qui permet le développement en C/C++ sur les appareils Android.
- Android SDK: Le kit de développement est un ensemble complet d'outils de développement.
- API Catalona pour le traitement d'image: C'est une discipline de l'informatique et des mathématiques appliquées qui étudie les images numériques et leurs transformations, dans le but d'améliorer leur qualité ou d'en extraire de l'information.

4.3 ENVIRONNEMENT DE TRAVAIL

Système d'exploitation : Ubuntu 14.04 / 64 bits

Processeur : Intel Core i5-3230M / 2.60 GHz

Mémoire installé RAM : 6 Go

Disque dur 750 Go

4.4 INTERFACES

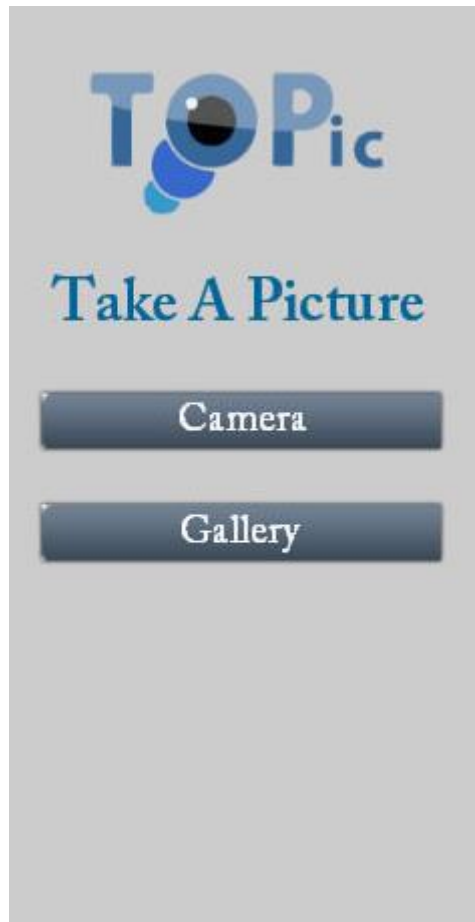


Figure 4.4-1: Interface d'accueil



Figure 4.4-2: Rogner l'image



Figure 3.4-3: Image rogner



Figure 3.4-4: Interface de traduction

4.5 TESTS

4.5.1 FILTRES



Figure 3.5.1-1: Filtre Gaussien – Elimination de bruit



Figure 3.5.1-2: Filtre de Seuillage



Figure 3.5.1-3: Filtre Gaussien et Seuillage

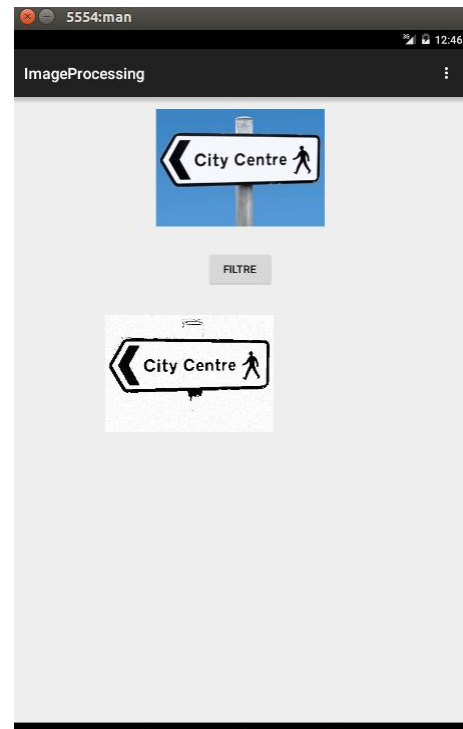


Figure 3.5.1-4: Filtre Grayscale, Gaussien, Seuillage

4.5.1 OCR



Figure 3.5.1-4: Test avec filtres



Figure 3.5.1-4: Test avec filtres en diminution de seuillage

Threshold (int threshold)

Initialisation d'une nouvelle instance de la classe Threshold (Seuillage)

Int thershold : est l'entier de taux de seuillage

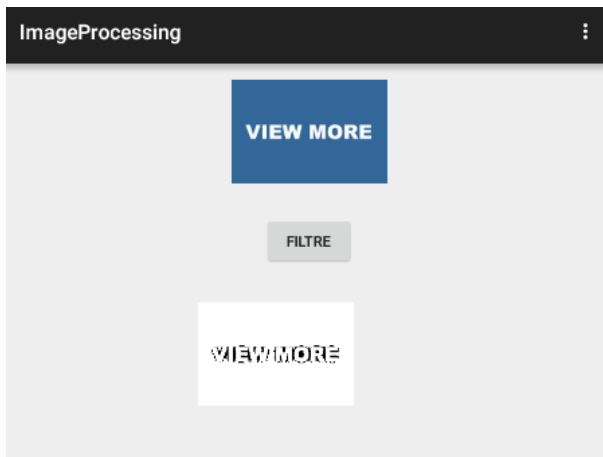
Différence de seuillage :

Figure 3.5.1-5: Seuillage faible

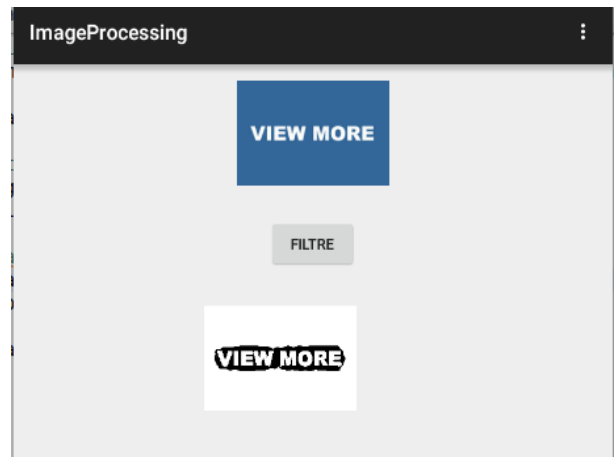


Figure 3.5.1-6: Seuillage fort

4.5.3 EVALUATION

Tesseract utilise des filtres de Leptonica pour le traitement d'image mais ce dernier ne génère pas de bons résultats. Pour cela nous avons utilisé d'autres filtres pour améliorer la qualité des images.

Pour un meilleur résultat d'OCR, un seuillage moyen ainsi qu'une image de haute résolution sera la meilleure solution pour avoir un résultat efficace.

Les tests actuels sont effectués sur un émulateur Android, cependant avec l'émulateur il n'est pas possible de réaliser ces tests avec des images de haute résolution. Ainsi les résultats obtenus sur émulateur sont pénalisés par la qualité moyenne des images utilisées. Les tests de l'application à réaliser directement sur un appareil mobile avec des images de haute résolution auront les résultats attendus et de très bonne qualité comme prévu.

4.6 CONCLUSION

Dans ce chapitre, j'ai commencé par citer les besoins technique, détailler l'environnement de travail. Ensuite j'ai donné une description des logiciels et des interfaces graphiques de l'application.

CONCLUSION GENERALE

Le présent rapport est le résultat de mon stage que j'ai effectué dans le cadre de mon projet de fin d'études de la Licence Fondamentale en Informatique au sein de l'entreprise Mahd.

Le but de ce travail a consisté principalement à réaliser une application Android pour la traduction des panneaux routiers dédiée aux touristes. Lors de ce stage de deux mois, j'ai pu mettre en pratique mes connaissances théoriques acquises durant ma formation. De plus, je suis arrivée à réaliser les objectifs que j'ai fixés au début de cette période. Bien que j'ai fait face à certains obstacles, j'ai pu trouver, avec de la recherche et de la persévérance les moyens de les résoudre et de réaliser l'application visée par le projet.

Ce présent rapport résume les étapes de mon travail. Dans le premier chapitre j'ai présenté le cadre générale de travail. Ensuite j'ai montré une analyse de domaine d'application en expliquant les architectures OCR et Tesseract. J'ai donné un aperçu général concernant le sujet en décrivant les solutions existantes et la solution proposée au niveau du deuxième chapitre. Le troisième chapitre est consacré à la description détaillée des besoins fonctionnels et non fonctionnels ainsi qu'à l'explication de la conception de mon projet. Enfin dans le dernier chapitre j'ai exposé les interfaces réalisées et l'évaluation de mes résultats.

Au cours de ce stage réalisé dans "Mahd", j'ai découvert l'ambiance du travail professionnel en équipe, les contraintes, les avantages. Ce stage m'a permis de percevoir les responsabilités que pouvait avoir le développeur, le chef de projets, l'analyste programmeur, l'ingénieur ou encore le directeur technique. Ce cadre m'a aidé à améliorer mes compétences en développement Andorid, l'étude et la réalisation d'un projet technique. Ces travaux m'ont réellement donné envie de conduire mes propres projets, de développer des solutions utiles au grand public. Ce stage m'a donné envie d'approfondir mes connaissances en matière de développement logiciels à forte valeur ajoutée.

BIBLIOGRAPHIE

- [1] Wikipedia Reconnaissance optique de caractères URL : http://fr.wikipedia.org/wiki/Reconnaissance_optique_de_caract%C3%A8res
- [2] Smith R. An Overview of the Tesseract OCR Engine URL: <http://static.googleusercontent.com/media/research.google.com/zh-CN/pubs/archive/33418.pdf> Accessed 10 April 2014
- [3] Smith R. Tesseract OCR Engine, 2007.
URL: <http://tesseract-ocr.googlecode.com/files/TesseractOSCON.pdf>. Accessible 10 Avril 2014
- [4] Verma R, Ali J. A-Survey of Feature Extraction and Classification Techniques in OCR Systems, 2014. URL: <http://www.ijcait.com/IJCAIT/index.php/www-ijcs/article/download/140/86> Accessible 10 Avril 2014
- [5] Abdel Belaïd et Yolande Belaïd. Reconnaissance des formes, chapitre 6. InterEditions, Paris, 1992.
- [6] Tesseract logiciel URL : [http://fr.wikipedia.org/wiki/Tesseract_\(logiciel\)](http://fr.wikipedia.org/wiki/Tesseract_(logiciel))

.