

**Find3r, El Fantomas, Tipchef et Triguneur** présentent



**RAPPORT DE PROJET**



**EPITA**

**ocrx.focalstudio.me**

# ocr X

A brand new optical character  
recognition system.

# Table des matières

<b>Introduction</b>	<b>1</b>
<b>1 Préface ⚡</b>	<b>3</b>
1.1 Objectifs de cette soutenance . . . . .	3
1.2 Répartition des tâches . . . . .	4
1.3 Planning des soutenances . . . . .	4
<b>2 L'équipe 🧑</b>	<b>5</b>
2.0.1 Edward Cacioppo . . . . .	6
2.0.2 Tom-Elliott Herfray . . . . .	6
2.0.3 Paul Viallet . . . . .	7
2.0.4 Alexis Huard . . . . .	7
<b>3 Le projet 🛠</b>	<b>9</b>
3.1 Processus et traitement de l'image . . . . .	9
3.1.1 Passage de l'image en noir et blanc . . . . .	9
3.1.2 Suppression du bruits . . . . .	10
3.1.3 Détection des lignes . . . . .	10
3.1.4 Détection des caractères . . . . .	11
3.2 Le réseau de neurones . . . . .	13
3.2.1 Première soutenance . . . . .	13
3.2.2 Soutenance final . . . . .	13
3.2.3 Modifications inachevés . . . . .	14
3.2.4 Sauvegarde des poids . . . . .	14

---

3.2.5 Progression . . . . .	14
3.3 Interface GUI . . . . .	15
3.3.1 Fonctionnement . . . . .	15
3.3.2 Différents éléments . . . . .	16
<b>4 Site web &lt;/&gt;</b>	<b>21</b>
<b>5 Citations ↗</b>	<b>25</b>
<b>6 Conclusion 🏆</b>	<b>27</b>

# Introduction



**Voici OCRx.**

OCRx (pour *Optical Character Recognition eXperience*) est un système de reconnaissance optique de caractères conçu dans le cadre du projet d'INFO-SPÉ de l'EPITA<sup>1</sup>. Sa durée s'étend sur un semestre lors du Semestre 3 du cycle préparatoire. Ce rapport de projet décrit la nature du projet ainsi que les différentes parties du logiciel.

Ce projet est la création d'un système de reconnaissance optique de caractères, sous la forme d'un logiciel développé en C. Un tel système extrait le texte contenu dans une image (photographie, texte dactylographié, document scanné, etc.) et conçoit à partir du document fourni, un document texte contenant les informations textuelles du document initial. Pour ce projet, nous avons utilisé le standard C99 du C.

Le groupe est constitué de *Tom-Elliott Herfray*, de *Paul Viallet*, d'*Edward Cacioppo* ainsi que d'*Alexis Huard*. Le logiciel est d'ores et déjà disponible sur notre site web.<sup>2</sup>.

**Nous vous souhaitons une bonne lecture de notre rapport final de projet!**

---

1. École d'ingénieurs spécialisées en informatique ([epita.fr](http://epita.fr))

2. Site du projet : **OCRx.focalstudio.me**

# Préface

## 1.1 Objectifs de cette soutenance

Pour cette dernière soutenance, et comme pour la première, nous avons suivi les objectifs énoncés dans le cahier des charges, c'est-à-dire la conception du réseau de neurones (apprentissage et reconnaissance), la reconstruction du texte et la sauvegarde ainsi qu'une interface utilisateur GUI complète.

Notre avons également finalisé le site web qui comporte : le code du projet, une documentation claire et concise ainsi qu'une section présentant les membres de l'équipe. Il est disponible dès maintenant à l'adresse **OCRx.focalstudio.me**.

## 1.2 Répartition des tâches

	Cacioppo	Herfray	Viallet	Huard
Segmentation de l'image	■			
Gestion de l'UX/UI (GUI et site web)		■		
Conception du réseau de neurones			■	
Traitement de l'information				■

## 1.3 Planning des soutenances

	Intermédiaire	Finale
Chargement des images et suppression des couleurs	★★	★★
Détection et découpage en blocs	★★	★★
Logique combinatoire et fonction XOR	★★	★★
Chargement du poids du réseau de neurones	★	★★
Jeu d'images pour l'apprentissage	★	★★
Manipulation des fichiers	★	★★
Apprentissage du réseau de neurones		★★
Reconstruction du texte et sauvegarde		★★
Conception d'une interface GUI	★	★★
Site web du projet	★	★★

★	Tâche commencée
★★	Tâche finalisée (ou quasiment finalisée)

# Chapitre 2

## L'équipe

Notre équipe se compose de :

- **Edward CACIOPPO** (*edward.cacioppo@epita.fr*)
- **Tom-Elliott HERFRAY** (*tom-eliot.herfray@epita.fr*)
- **Paul VIALLET** (*paul.viallet@epita.fr*)
- **Alexis HUARD** (*alexis.huard@epita.fr*)

## 2.0.1 Edward Cacioppo



**Edward 'Tipchef' CACIOPPO**

« Je m'appelle Edward Cacioppo, j'ai 20 ans et je suis en Info-Spé à EPITA. J'aime personnellement la science-fiction et la fantasy, j'apprécie plus particulièrement les œuvres de Tolkien, Arthur C. Clark et Martin. Ayant déjà travaillé sur un projet durant le S2, j'ai, par conséquence, de l'expérience en travail d'équipe. Mais ici, le projet se fait sans l'utilisation d'un logiciel qui réalise la majeure partie du travail. Ainsi, pour ce projet, je suis en charge de la segmentation avec Alexis, et j'espère que vous appréciez le (petit) fruit de notre travail. 🍄»

## 2.0.2 Tom-Elliott Herfray



**Tom-Elliott 'Find3r' HERFRAY**

« Geek, fan de Sci-Fi depuis mon enfance, je m'intéresse beaucoup aux nouvelles technologies (c'est rare ici, je sais). J'aime le DIY (Do It Yourself) et j'adore concevoir des projets, allant d'un casque de réalité virtuelle à un traducteur vocal intuitif, en passant par un jeu vidéo sur le thème de Star Wars. Ce dernier, projet de première année à EPITA, était une excellente expérience pour (re)découvrir le C# et Unity. Pour cette seconde année, nous entrons dans la cour des grands, et OCRx est l'occasion de passer aux choses sérieuses. Mon mantra ? **Pensez différemment!** 🎶»

## 2.0.3 Paul Viallet



**Paul 'El Fantomas' VIALLET**

« Fan de manga, notamment des œuvres de Leiji Mastumato, et d'histoire, j'ai pendant longtemps pensé m'orienter vers une filière en rapport avec l'histoire. Mais la découverte de Minecraft en 2010 en a voulu autrement : grâce à ce jeu, j'ai découvert les joies de la programmation et du monde de l'informatique, c'est ainsi que j'ai intégré EPITA. Ma nature touche à tout, du fait que je possède de nombreuses connaissances et que, une fois lancé, je suis capable d'accumuler de nombreuses informations. Grâce à OCRx, j'ai d'ores et déjà pu faire d'immenses progrès en C et notamment vis à vis de l'utilisation de gdb. Debugging is like being the detective where you are also the murderer ⚡»

## 2.0.4 Alexis Huard



**Alexis 'Triguneur' HUARD**

« Le projet de mon S2 était pour moi une première expérience de projet de groupe, me faisant ainsi découvrir le développement de programme en groupe avec certaines contraintes. Le projet de cette année est pour moi l'occasion de découvrir plus profondément le développement de programme plus complexe comme pourrait l'être l'OCR. De plus, ce projet me permet de découvrir de nouveaux algorithmes comme ceux que l'on utilise pour la segmentation ou pour le machine-learning. Un jour j'ai réussi à faire un code sans bugs, et puis on m'a dit que je m'étais trompé de sujet. 🤪»

# Le projet

## 3.1 Processus et traitement de l'image

Pour cette première soutenance, nous nous sommes tenus à un algorithme de base pour gérer la segmentation. Nous avons ainsi décidé de simplement gérer un texte en noir et blanc sans différenciation entre les paragraphes et les différentes zones de textes dans l'image.

### 3.1.1 Passage de l'image en noir et blanc

Pour que la segmentation fonctionne correctement il nous faut avoir une image dénuée de couleurs et n'avoir que du blanc et du noir.

Pour cela nous devons traité l'image grâce a un algorithme qui s'occupe de la transformer. le traitement se fait alors en plusieurs étapes, nous récupérons tout d'abord les données RGB de l'ensemble des pixels, et grâce à aux données récupérées, nous calculons le niveau de gris correspondant à chaque pixel.

Pour la première soutenance, nous avons d'cidé de choisir un seuil. Pour à partir du quel l'image passe en noir (0, 0, 0) ou en blanc (255, 255, 255). Ce seuil avait été choisis arbitrairement pour convenir à nos images de tests. Pour avoir un plus large panel d'image fonctionnant dans notre ocr de qualité nous avions besoin que l'algorithme définisse le seuil automatiquement en fonction du teint de l'image. Nous avons donc décider de nous pencher sur la méthode d'Otsu. Cette methode determine en fonction de l'histogramme de l'image le seuil. La determination du seuil se fait en fomnction de pro-

babillites sur tous les seuils et leur écarts-types et leur variances.

L'avantage de cet algorithme est qu'il est plutôt rapide avec un bon indice de décision. Grâce à cet algorithme nous avons aussi la possibilité de supprimer les pixels de gris clair entre deux lettres rapprochées. Certain de nos tests lors de la première soutenance ne fonctionnait pas avec le seuillage arbitraire. Maintenant avec cet algorithme nous avons la possibilité de segmenter sur un plus grand nombre de police de caractères.

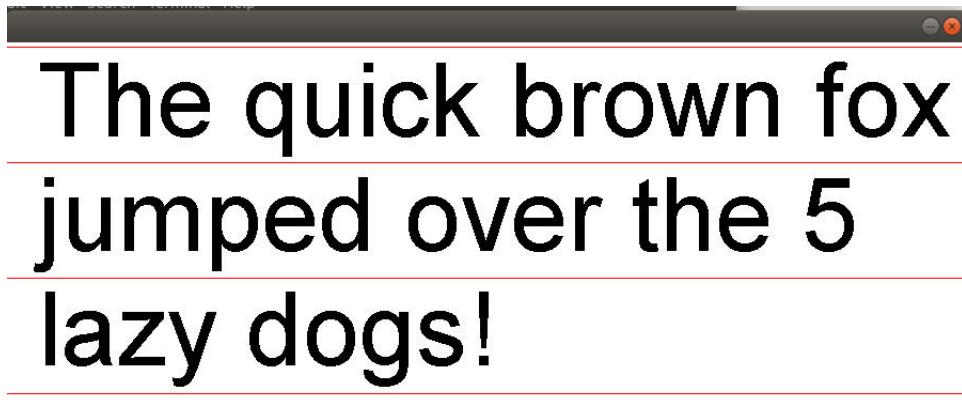
### 3.1.2 Suppression du bruits

Nous avions décidé d'implémenter la suppression du bruit pour la seconde soutenance. Mais en raison du timing nous avons décidé de nous concentrer sur la segmentation. Pour une meilleure fonctionnalité de notre OCR. A la suite de notre soutenance nous allons mettre notre code en open-source. Il y aura donc la possibilité pour une personne intelligente située quelque part sur le globe ou ailleurs, de coder cette fonctionnalité. Lors de nos recherches sur les techniques de suppression de bruits nous avons étudié la possibilité d'utiliser le filtre médian ou médian large.

### 3.1.3 Détection des lignes

La détection des lignes a peu changé par rapport à la première soutenance, le principe reste le même, on fait une vérification horizontale de la présence de pixels noirs en partant du haut de l'image, si on détecte un pixel noir alors on a découvert un caractère et donc une ligne, on continue la vérification jusqu'à qu'on ne rencontre pas un pixel noir, on a donc trouvé la fin de la ligne. L'algorithme renvoie la coordonnée verticale de la fin de la ligne qui permet de tracer la ligne.

Auparavant on utilisait la valeur renvoyer pour tracer le reste des lignes mais désormais on relance l'algorithme mais depuis la fin de la dernière ligne pour détecter la prochaine ligne. On relance donc l'algorithme en boucle jusqu'à qu'on arrive à la fin du texte. Pour aider le découpage on trace un trait rouge en haut et en bas pour permettre le découpage entre les caractères.



Détection des lignes

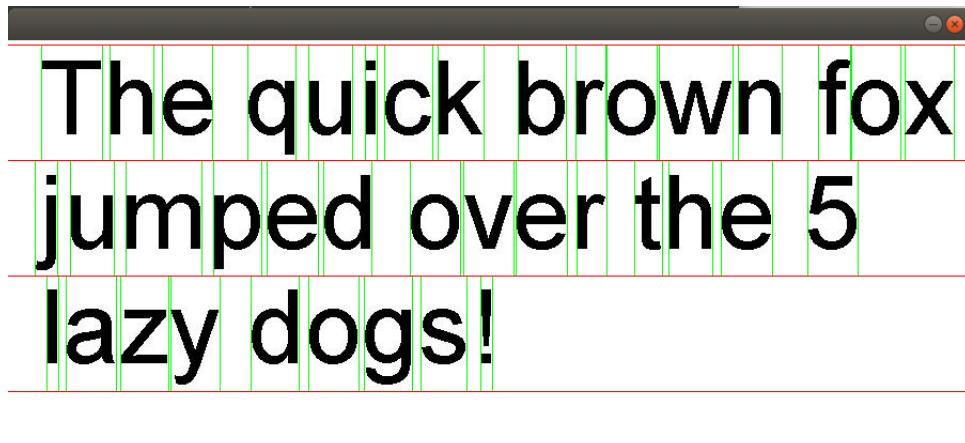
### 3.1.4 Détection des caractères

Ici le principe est sensiblement le même, ligne par ligne, on traverse horizontalement pour détecter les caractères. En effet sur la ligne on fait une vérification verticale, la ou auparavant on lorsqu'on avait un espace de blanc entre caractère on tracer, il fallait donc avoir un espace de blanc assez grands pour tracer une ligne verticale.

Maintenant lorsque on rencontre un pixel noir, on passe à un algorithme récursive qui recherche un chemin possible vers la fin de la ligne donc trouver un pixel rouge en descendant, l'algorithme limite les déplacements horizontaux de 2 pixels et si on trouve un pixel noir on retourne faux, idem lorsqu'on trouve un pixel bleue qui est la couleur utilisé lorsqu'on a trouvé un chemin. Cela permet donc de couper les caractères qui on espace entre eux minuscules.

Maintenant avec les lignes et les caractères coupé visuellement on passe au découpage réel, du fait que se qui délimite les caractère sont des courbes bleu, il faut alors créer une image qui est assez grande pour prendre le caractère est, on utilise alors un algorithme qui utilise les pointeurs pour récupérer les coordonnées pour pouvoir créer une image de taille nécessaire pour prendre toute le caractère, ensuite, lors de la création de l'image, on garde tout ce qui est entre les deux courbes bleues et tout ce qui est hors devient blanc, on utilise ensuite les valeurs des coordonnées avec l'algorithme pour calculer les coordonnées du prochain caractères. On fait sa en suite pour la prochaine ligne.

Ainsi on utilise les images reçus en les convertissant en matrices puis on les mets à une taille voulu puis on les mets dans une file qui sera utilisée par le réseau de neurones.



Détection des caractères

## 3.2 Le réseau de neurones

Partie centrale du projet, le réseau de neurones est une structure importante en informatique. La compréhension du projet nous a demandé du temps. Ainsi à la première soutenance nous avions un réseau fonctionnel et apte à apprendre la fonction XOR. Pour cette seconde soutenance nous avons essayé de l'adapter à un nombre d'entrée et de sortie plus importante, avec succès.

### 3.2.1 Première soutenance

Pour la première soutenance nous avions réalisé un réseau fonctionnel, celui-ci représenté par une struct comprenant tout les éléments requit pour son fonctionnement. Ainsi il possédait des variables (pointeurs de double) représentant les poids entre les différents neurones (entre la couche interne et la couche caché, et entre la couche caché et les sorties), représentant les biais ainsi qu'un pointeurs de size\_t représentant le nombre de neurones sur les différentes couches.

### 3.2.2 Soutenance final

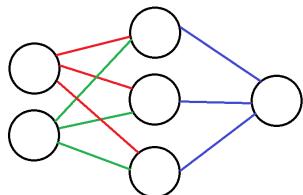
Ainsi pour cette seconde soutenance l'objectif principal était de pouvoir faire passer en entré une matrice contenant l'image binarisé. Pour cette première partie nous avons rencontré de nombreux problèmes, non pas sur le réseau de neurones en lui même, mais sur l'entrée : lors de la conversion de la l'image en double pointeur, certain pointeurs était free. Ce problème nous à fait perdre énormément de temps. Ce problème à été résolue via l'implémentation d'une struct matrice, qui a étonnamment résolue le problème. Il semblerai que ce problème vienne d'un garbage collector de la fonction main.

Les calculs ont été légèrement remaniés : nous avons ajoutés des biais pour augmenter la précision du réseau.

Il existe 52 sorties correspondant aux 52 lettres majuscules et minuscules de l'alphabet. La sortie 0 étant la lettre A et la sortie 51 étant la lettre Z. La fonction process, permettant d'utiliser le réseau de neurones sans la backpropagation, renvoie un entier qui sera ensuite rapporté a la table Ascii.

### 3.2.3 Modifications inachevés

L'un de nos objectifs a été de convertir la structure du réseau, peu modulaire, en une structure de structure de structure : une structure représentant le réseau contenant une liste de structure de layers contenant une liste de neurones.



### 3.2.4 Sauvegarde des poids

La sauvegarde des poids a été achevé et fonctionne parfaitement, les biais et les poids sont sauvegardés dans des fichiers .ocrx, ils sont chargés au lancement du réseau de neurones et sauvegardé à la fin de son exécution.

### 3.2.5 Progression

Le réseau a été achevé, mais malheureusement il nous a été impossible de l'entraîner à cause d'un manque de temps. Aussi il pourrait être intéressant de finir l'implémentation des structs dans le réseau.

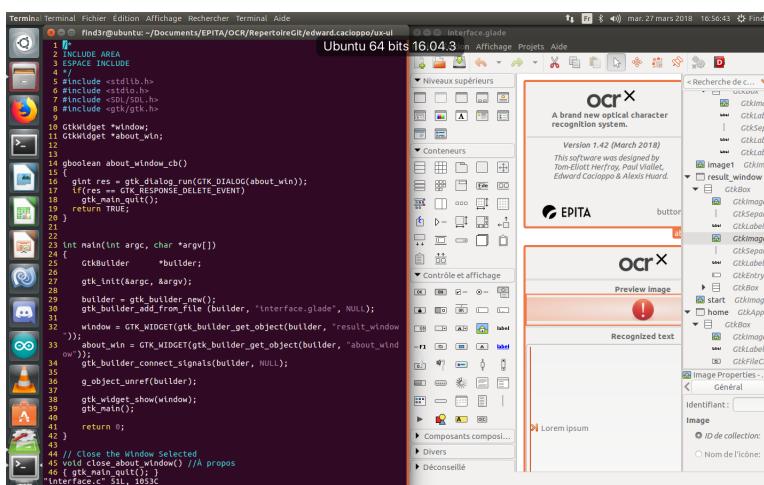
## 3.3 Interface GUI

### 3.3.1 Fonctionnement

Pour cette soutenance finale de projet, nous avons amélioré l'interface GUI qui était déjà bien avancée à la soutenance intermédiaire : nous avons connecté les différents éléments à l'interface et cette dernière est pleinement fonctionnelle. Nous avons également optimisé le programme de façon que la GUI puisse récupérer en temps réel le résultat du réseau de neurones, qui est stocké dans un fichier .txt dans notre dossier. Pourquoi un fichier texte ? Car si le réseau de neurones prend du temps à détecter les caractères, l'interface et le processus ne seront pas bloqués et ça améliorera l'UX.

Comme pour la première soutenance, nous avons conservé le modèle de 3 fenêtres distinctes : `Main_Window` (où on sélectionne le fichier image), `About_Window` (où l'on peut voir la version actuelle et l'équipe) et bien évidemment, `Result_Window`, qui affiche l'image source en haut et le texte détecté en bas.

De même que pour le fichier texte, l'image est actualisée en temps réel. Ce qui permet un gain considérable de temps ainsi qu'une optimisation non négligeable. Je vais m'attarder maintenant sur le fonctionnement des 3 fenêtres.



Environnement de travail Gnome Glade sur Ubuntu 16.04.

### 3.3.2 Différents éléments

#### Fenêtre Main

Main\_Window est la fenêtre principale du programme, et même de l'OCR. En effet, lorsque vous compilez et que vous lancez l'OCR avec ./ocrx, c'est la fenêtre Main qui s'affiche en première. Cette fenêtre se compose d'un bouton "Choose an image" qui permet de naviguer dans les fichiers de l'ordinateur et de trouver une image. Vous voyez également sur le côté une petite icône "About", cette dernière appelle la fenêtre About qui affiche le numéro de version ainsi que la liste des membres de l'équipe.

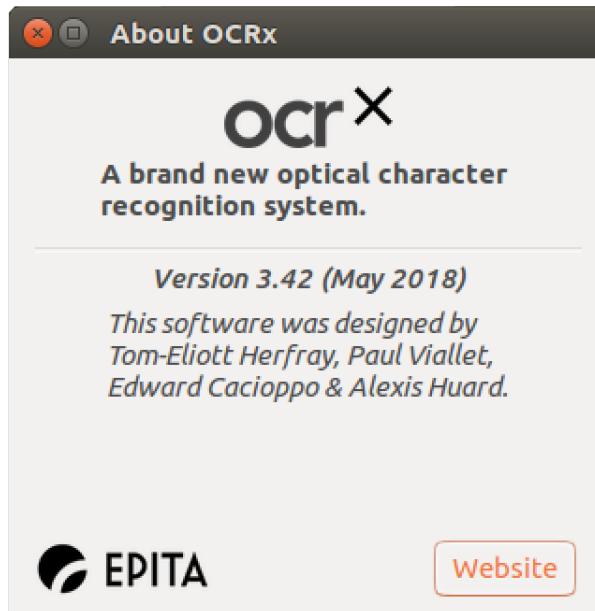
À noter par ailleurs que lorsque vous sélectionnez un fichier à l'aide du bouton "Choose an image", GUI lance directement la segmentation et le réseau de neurones en amont, et charge la fenêtre Result.



Fenêtre principale du projet.

### Fenêtre About

La fenêtre About résume en quelques mots le projet et crédite les auteurs (nous, en l'occurrence), on y voit les logos d'EPITA et notre groupe de projet, Focal Studio, ainsi qu'un lien sur le bouton de droite vers le site web du projet : <https://ocrx.focalstudio.me>. On y remarque également la version actuelle du logiciel<sup>1</sup>.



Fenêtre À propos du projet, avec les auteurs et un lien vers le site web.

---

1. Version actuelle : Golden Master 3.42

## Fenêtre Result

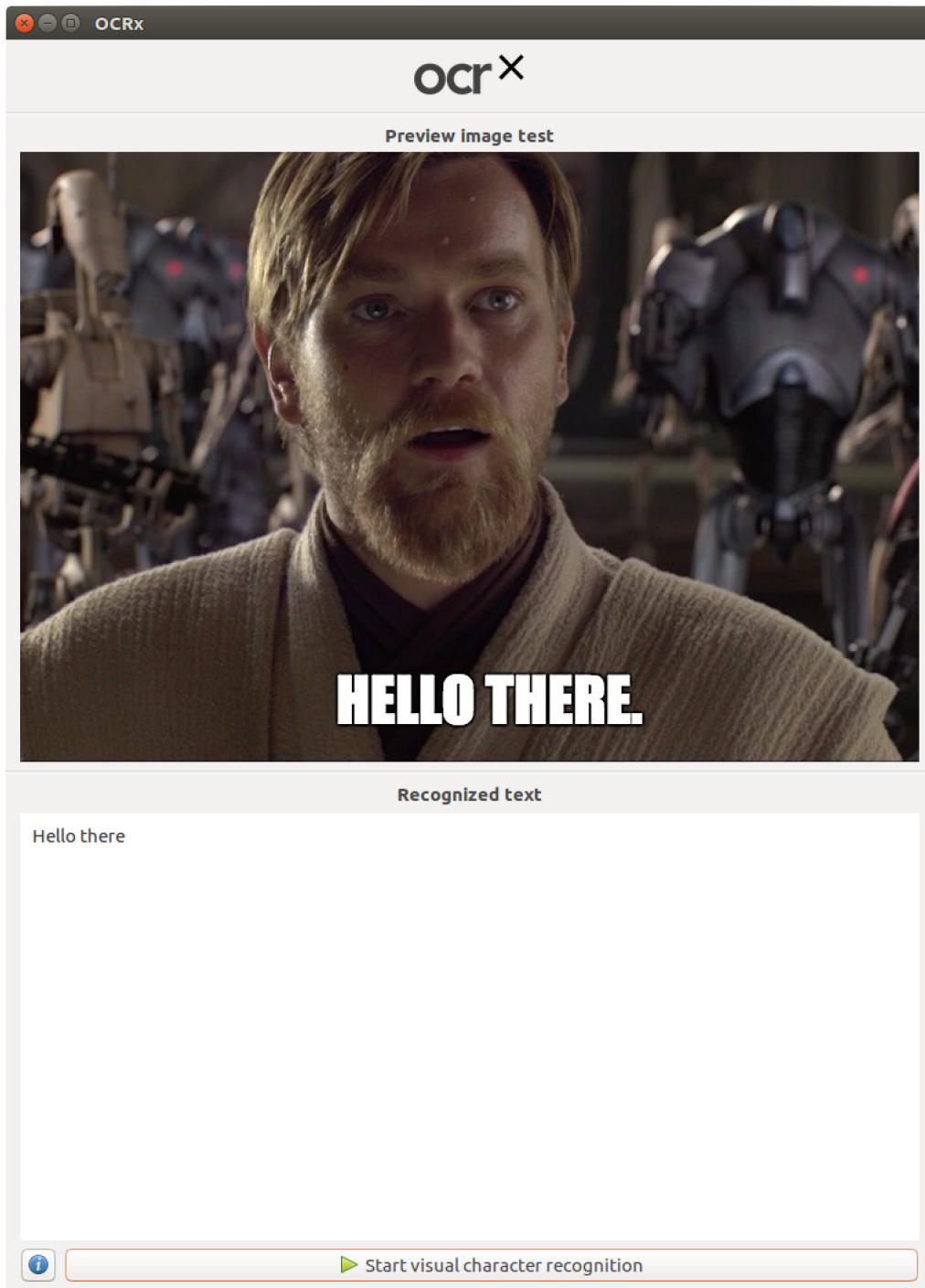
Parmi les 3 fenêtres du logiciel, c'est bien `Result_Window` qui a subit le plus de transformations, puisque dorénavant elle n'utilise plus une `GtkEntry` mais une `GtkTextView`, ce qui permet de rafraîchir les résultats à n'importe quel moment (un pointeur pointe directement sur le contenu du fichier). L'icône rouge qui trônait encore à la soutenance intermédiaire a disparu... et l'image s'affiche à la place. Et en temps réel en plus! L'interface gère par ailleurs une immense majorité des formats image : nous n'avons pas constaté de problèmes avec les centaines d'images pour les tests.

Autre nouveauté, le texte est sauvegardé automatique grâce à l'implémentation d'une partie de la GUI dans le réseau de neurones qui retourne une chaîne de caractères (`char` pour les intimes).

Nous avons laissé l'icône `About` sur la gauche mais le bouton de droite devient "`Start visual character recognition`" qui rafraîchit les résultats du réseau de neurones si l'on constate un problème ou une faute d'orthographe par exemple.

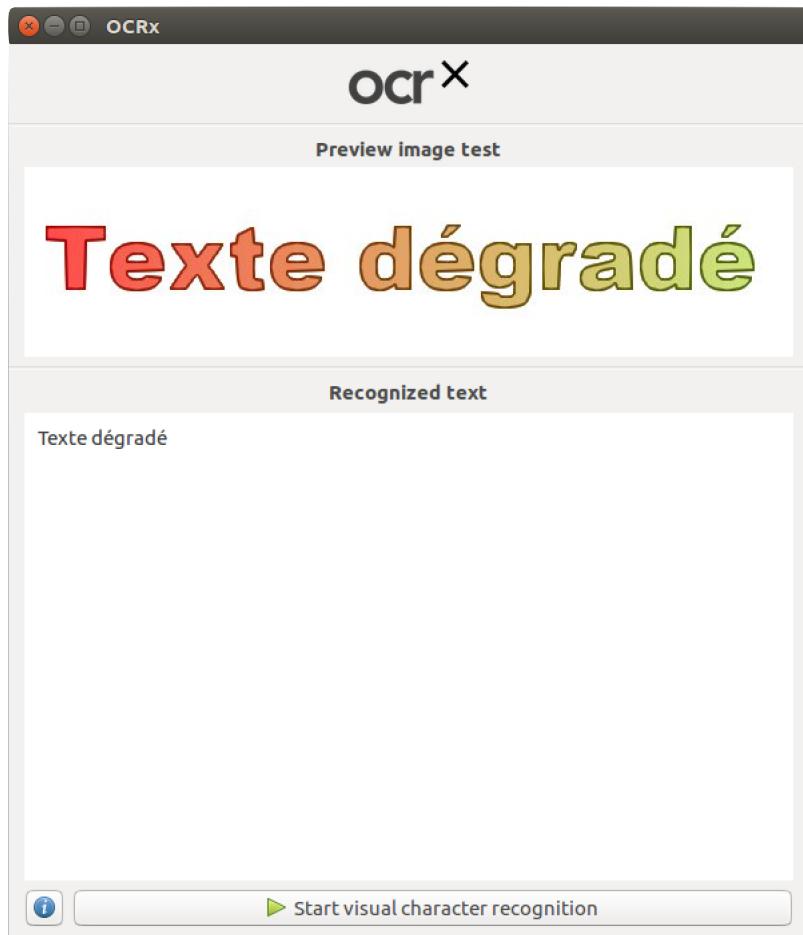
De même, si l'image est trop grande par rapport à la taille imposée pour la fenêtre s'adaptera automatique, nous avons également donné la possibilité d'un mode plein écran. 

**Nota Bene :** *L'ensemble des screenshots de l'interface GUI proviennent d'Ubuntu 16.04 mais le module Glade/GTK+ est déjà implémenté sur les machines à EPITA, donc pas de problème de compatibilité.*



Fenêtre du résultat retourné par le réseau de neurones d'OCRX.<sup>2</sup>

2. Les fans de la saga *Star Wars* remarqueront la sublime référence de cette image. Il s'agit de la fameuse citation du Jedi Obi-Wan Kenobi lorsque ce dernier s'apprête à combattre le sinistre Général Grievous dans l'*Épisode III : La Revanche des Sith*.



Un autre exemple, moins geek, mais qui montre que l'on peut gérer également les textes de différentes couleurs.

# Chapitre 4

## Site web </>

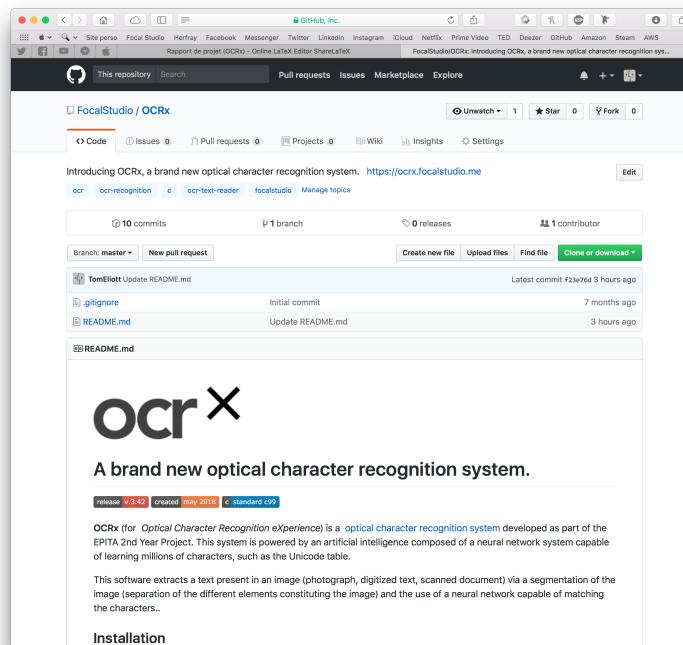
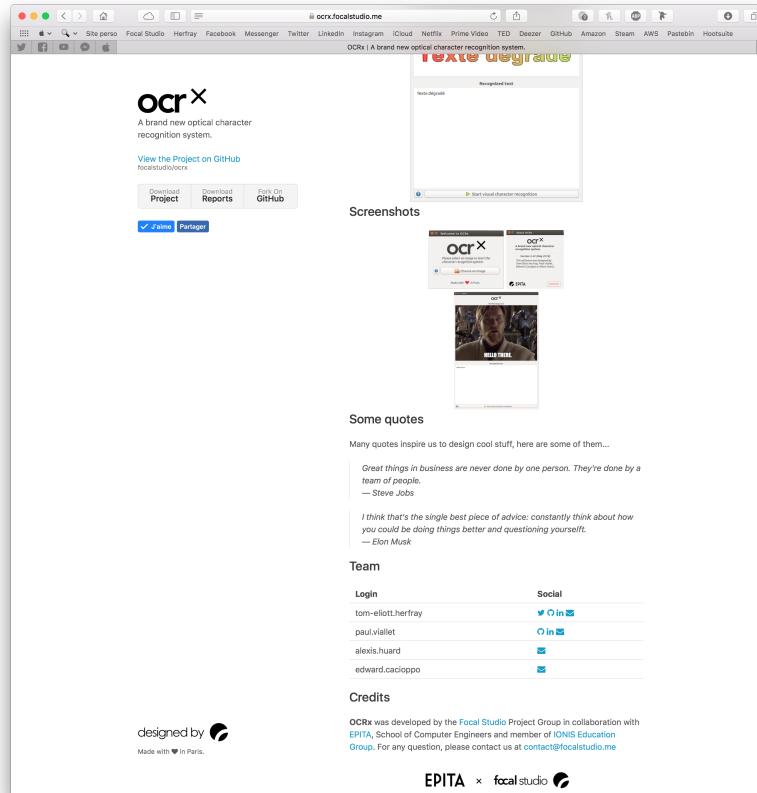
Comme annoncé lors de la soutenance intermédiaire, nous avons conçu un site web regroupant les différents éléments du projet, à savoir principalement un lien vers le GitHub de notre projet (le code source sera publié après le clonage des dépôts Git), une documentation sur le fonctionnement de l'OCR, les différents rapports (soutenances intermédiaire et finale) ainsi que d'autres éléments. Nous vous invitons à consulter notre site à l'adresse **OCRX.FOCALSTUDIO.ME**.

Le site et le logo du projet ont été réalisés par nos soins.

**ocrx.focalstudio.me**

The screenshot shows the main page of the OCRX website. At the top, there's a navigation bar with links to Focal Studio, Herfray, Facebook, Messenger, Twitter, LinkedIn, Instagram, iCloud, Netflix, Prime Video, TED, Deezer, GitHub, Amazon, Steam, AWS, Pastebin, and Hootsuite. Below the navigation bar, the title "ocrX" is displayed with a subtitle "A brand new optical character recognition system." A "View the Project on GitHub" link and a "focalstudio/ocrx" URL are shown. There are three download buttons: "Download Project", "Download Reports", and "Fork On GitHub". Below these buttons are "J'aime" and "Partager" social sharing buttons. The main content area starts with a heading "Introducing a all-new optical character recognition system." followed by a detailed description of what OCRX is and how it works. It includes sections for "Installation", "Requirements", and "How it works?". A note at the bottom states: "The division of the image into blocks then into characters (passing through a".

Page principale du site web.



Screenshot du GitHub ([github.com/focalstudio/ocrx](https://github.com/focalstudio/ocrx))

# Citations “

Durant les 6 derniers, nous avons travaillé quotidiennement sur Git et avons décidé de faire une compilation des meilleurs commentaires (messages des *commit*) :

*"praise our lord and savior, gcc"*

- Tipchef

*"Do you know the movie inception, here it's the same thing but with struct, NOW WE HAVE SO MANY STRUCT, struct for network, struct for layer, struct for neuron, event struct for unsigned int, IT'S MADNESS. I love struct. Struct are love, struct are my life. I hope you like struct."*

- El Fantomas

*"I decided to abandon SDL near a road, so I fully adopted GTK+ <3"*

- Find3r

*"Finished char cut, solved the north-korean problem and fought an alien invasion."*

- Tipchef

*"Introducing OCRX Version 1.42.8.5.45.C.12, it's (almost) amazing. Looks like a version number of Google Chrome"*

- Find3r

*"HELLO THERE. GENERAL NETWORK. So basicly : improve, save, thing, beginning of the final v. other thing"*

- El Fantomas

"may the force be with me"

- Triguneur

"Wait, that's no moon, that's... trash files"

- Find3r

"Just remain to implement the weight update & save :D. Yup there is a variable call deltajesaisplusquo. I jsut didn't remember what it was after implementing it but i know we need it so i give it this name. maybe it will remain until the end. it's funny. this monday i will finish it test it implement the save (ez pz). Maybe i will try to use picture as matrice. it could be great. let's the fun begin."

- El Fantomas

"good sample for more success"

- Triguneur

"The time has come (to compile), execute ./Order66"

- Fin3dr

"Fix this third iteration of nn, now it's work. fabulous. finally. please help"

- Tipchef

"plz make it work"

- El Fantomas

# Conclusion

Cette seconde année au sein d'EPITA a été très riche et la réalisation d'un système de reconnaissance des caractères dès la seconde année de notre cycle a été une formidable expérience pour l'ensemble du groupe, nous donnant un aperçu du futur de nos études et nous permettant de maîtriser de nombreuses nouvelles notions.

Malgré d'importants problèmes rencontrés, nous avons toujours gardé la tête haute. Et même après la fin "officielle" du projet, nous comptons travailler sur OCRx afin de rendre une version totalement fonctionnelle et dédiée au monde de l'open-source.

C'est la raison pour laquelle nous avons décidé de publier, à l'issue de la soutenance finale, notre projet sur la plateforme collaborative *GitHub*, afin de donner la possibilité à des millions d'utilisateurs de contribuer à notre projet. De plus, nous avons également finalisé le site web (disponible à l'adresse **ocrx.focalstudio.me**) qui contient les sources du projet, les rapports de soutenance et de projet ainsi qu'une documentation sur le fonctionnement du programme. Même si le réseau de neurones n'est pas parfait à 100%, nous comptons grandement l'améliorer, en sachant par ailleurs que nous avons déjà passé une centaine d'heures dessus.

En vous remerciant encore, **paix et prospérité** 

– Tom-Elliott, Edward, Alexis et Paul.

*Great things in business are never done by one person. They're done by a team of people.*  
— Steve Jobs

---

*I think that's the single best piece of advice: constantly think about how you could be doing things better and questioning yourself.*  
— Elon Musk



Proudly powered by  $\text{\LaTeX}$

**ocrx.focalstudio.me**