Rapport de Projet

OCR PRO MAX

November 3, 2024

Créé par :

Alexis Chafaï

Lorys Févry

Otto Debrie

Baptiste Mandry–Delimoges

Table des matières

1 Introduction

Ce rapport à pour but de décrire les réalisations de notre groupe pour cette première soutence du projet d'OCR du S3. Pour rappel OCR en français est traductible par reconnaissance optique de caractères. L'objectif de notre projet est donc de coder un algorithme d'OCR appliqué à une grille de mots mêlés afin qu'il puisse la résoudre en retrouvant tout les mots dedans. Plusieurs étapes sont nécessaires à la construction de ce projet. Nous en avons identifié trois principales que nous nous sommmes repartis de la manière suivante : Alexis, s'occupe de la partie interface avec la fenêtre et le traitement de l'image, Otto s'occupe du réseau de créer le réseau de neurones et de son entraînement enfin Lorys et Baptiste s'occupent de la reconaissance de la grille chacun travaillant une partie différente de l'autre. Dans la suite de ce rapport, nous détaillons chacun notre travail ainsi que notre ressenti à propos de ce projet mais aussi plus généralement de notre rapport à l'informatique.

2 Lorys

2.1 Présentation et Motivation

Motivé par le domaine de l'intelligence artificielle (IA), j'ai un fort intérêt pour les systèmes de reconnaissance d'images et les algorithmes de traitement de données visuelles. Ce projet, impliquant un logiciel de type OCR (Optical Character Recognition) pour résoudre une grille de mots cachés, m'a immédiatement attiré, car il représente un défi dans la création de systèmes capables d'analyser et d'interpréter des informations visuelles, un domaine où l'IA peut apporter de nombreuses solutions innovantes.

Dans cette perspective, j'ai pris en charge la tâche de détection de la grille, étape cruciale qui vise à identifier les contours et les cases de la grille de lettres sur l'image d'entrée. Cette tâche, bien que complexe, se rapprochait de mes intérêts pour l'IA, car elle nécessitait de comprendre la structure de la grille et d'analyser ses motifs visuels. Au-delà des compétences techniques, ce travail m'a également permis de développer des compétences en recherche de méthodes et en utilisation de la bibliothèque SDL, que j'ai explorée en profondeur.

2.2 État d'avancement

La détection de la grille, dont j'étais responsable, a progressé plus lentement que ce que nous avions anticipé. L'objectif initial était d'atteindre 60 pourcents d'avancement pour cette tâche au moment de la soutenance, mais nous avons finalement atteint environ 30 pourcents. Plusieurs raisons expliquent cet écart entre les prévisions et la réalité.

D'abord, la détection de la grille s'est révélée bien plus complexe que prévu. Mon approche initiale reposait sur l'identification des motifs rectangulaires ou carrés formant la grille. En utilisant SDL pour manipuler les images, j'espérais que la bibliothèque fournirait un support suffisant pour détecter les lignes de séparation. Cependant, bien que SDL soit performante pour des manipulations graphiques de base, ses capacités pour une détection de contours précise sont limitées. L'absence de fonctions avancées pour le traitement d'image a rendu la tâche d'autant plus complexe, nécessitant un effort de recherche supplémentaire pour compenser.

Ensuite, chaque méthode de détection de la grille que j'ai explorée a nécessité des ajustements fréquents et des tests répétitifs. J'ai essayé différentes techniques, telles que le filtrage et la détection de bords, pour isoler les lignes de la grille. Malheureusement, ces méthodes, bien qu'instructives, ont pris plus de temps que prévu et n'ont souvent pas produit les résultats escomptés. Chaque nouvel essai impliquait des ajustements fins des paramètres et une période d'essais pour évaluer les améliorations potentielles. Ces ajustements constants ont ralenti l'avancement global, mais ils m'ont permis d'affiner mes compétences en traitement d'image et d'approfondir mes connaissances sur SDL.

Malgré une progression inachevée, ces tentatives m'ont permis de découvrir des techniques de traitement d'images précises et d'enrichir mes connaissances en manipulation d'images. Cette expérience, bien que difficile, m'a permis de renforcer mes compétences, ce qui sera bénéfique dans mes projets futurs.

Pour rattraper ce retard, nous avons choisi de simplifier notre approche initiale. Nous avons adopté une méthode plus pragmatique, en binarisant l'image pour rendre les lignes de la grille plus visibles.

Cette approche, bien que moins avancée techniquement, permet d'isoler plus facilement les contours de la grille et de contourner les limitations de SDL. Grâce à cette solution simplifiée, nous espérons pouvoir rattraper le temps perdu tout en conservant un niveau de précision acceptable pour notre logiciel de reconnaissance de caractères.

2.3 Aspects techniques

Mon rôle principal dans ce projet a été de me concentrer sur la détection de la grille, qui constitue une étape fondamentale pour localiser les lettres avec précision et les isoler dans chaque case de la grille. Cette tâche a consisté à développer un processus permettant à notre programme de reconnaître la structure quadrillée de l'image en identifiant les lignes verticales et horizontales séparant les cases. Voici les principales étapes et méthodes que j'ai explorées pour atteindre cet objectif :

Analyse de pixels et détection de lignes : Ma première approche a consisté à parcourir l'image pour repérer les changements de pixels, signes de bordures entre les cases. L'idée était de détecter des variations de couleur ou de luminosité entre des pixels adjacents, en espérant que ces différences indiquent la présence de lignes de séparation. Cette méthode s'est cependant révélée délicate, car les variations de pixels peuvent être subtiles ou peu fiables selon la qualité de l'image. Filtrage et binarisation de l'image : Pour surmonter les limites de l'analyse pixel par pixel, j'ai ensuite essayé de transformer l'image en version binaire, où les pixels sont réduits à des valeurs de noir et de blanc. Cette binarisation simplifie l'image et met mieux en évidence les lignes en les rendant plus contrastées par rapport au fond. Ce procédé m'a permis d'isoler plus facilement les motifs rectangulaires représentant les cases de la grille. Malgré des résultats partiels, cette technique m'a permis de faire ressortir des éléments essentiels, comme les lignes principales, bien que le traitement d'image avec SDL ait ses limites.

SDL pour le traitement d'image : SDL, ou Simple DirectMedia Layer, est une bibliothèque open-source couramment utilisée pour le développement de jeux vidéo et d'applications multimédia. Elle fournit une interface pour accéder aux fonctionnalités essentielles des systèmes d'exploitation, comme le rendu graphique, la gestion des entrées utilisateur, et l'audio, tout en étant compatible avec plusieurs plateformes (Windows, macOS, Linux, etc.).

Dans notre projet OCR, SDL nous a permis de charger et manipuler les images, notamment en accédant directement aux pixels pour tenter de détecter les lignes de la grille. Bien qu'elle soit efficace pour les opérations de base, SDL n'est pas conçue pour les traitements d'images avancés, comme la détection de contours, ce qui nous a obligé à adapter notre approche pour contourner ces limites.

La bibliothèque SDL s'est avérée utile pour manipuler et afficher les images, mais elle est conçue pour des manipulations graphiques de base et n'offre pas de fonctions avancées pour le traitement d'images, comme la détection de contours et de motifs complexes. J'ai donc dû adapter et créer certaines fonctions pour simuler des opérations de base, mais j'ai rapidement constaté que cette approche nécessitait beaucoup d'efforts pour des résultats limités. Bien que cette bibliothèque ne soit pas optimale pour ce type de tâches, elle m'a permis de travailler sur une première version fonctionnelle de manipulation d'image.

Malgré une finalisation incomplète de cette tâche, chaque méthode testée m'a permis d'acquérir des connaissances approfondies en manipulation d'images et en traitement de pixels, connaissances qui seront précieuses dans mes futurs projets. Si je devais recréer ce projet, je pourrais utiliser l'expérience acquise pour démarrer directement avec des techniques plus adaptées, comme la binarisation dès le début, pour mieux structurer le programme et accélérer le processus de détection des lignes de la grille.

Voici les étapes pour reconstruire ma partie : réinstaller les outils nécessaires comme Visual Studio Code avec MinGW pour le compilateur C/C++ et SDL pour le traitement d'images.Reprendre les méthodes de détection de lignes simplifiées, notamment en binarisant l'image dès le début pour simplifier la détection.

3 Baptiste

3.1 Présentation et Motivation

Depuis que je suis jeune, j'ai toujours été passionné par l'informatique et les nouvelles technologies. Cet intérêt s'est vraiment développé lorsque j'ai eu l'opportunité de faire mon stage de troisième dans une startup spécialisée en deep learning. Pendant ce stage, j'ai découvert l'univers fascinant de l'intelligence artificielle, des réseaux de neurones et de l'apprentissage automatique. J'ai pu comprendre comment les machines peuvent apprendre à reconnaître des motifs dans des données aussi variées que des images, des sons ou des textes. Cette expérience a été une révélation pour moi et m'a donné l'envie de maîtriser les techniques qui permettent de créer de telles applications. À partir de ce moment-là, j'ai su que mon avenir académique et professionnel serait dans cette voie.

Aujourd'hui, je suis étudiant à Epita, une école d'ingénierie informatique, et je continue de suivre ce rêve de devenir un expert en technologie. Mon objectif est de développer mes compétences en programmation et en développement d'applications. C'est pour cela que j'ai décidé d'apprendre le langage C, réputé pour sa puissance et sa complexité. En travaillant avec le C, je peux vraiment comprendre comment fonctionne la mémoire et comment sont structurés les systèmes informatiques. Ce sont des connaissances essentielles pour concevoir des applications robustes et performantes, et je me suis lancé dans plusieurs projets en C pour m'entraîner. Actuellement, je travaille sur un projet de reconnaissance de caractères, ou OCR (Optical Character Recognition), ce qui correspond parfaitement à mes aspirations.

L'OCR est une technologie de reconnaissance optique des caractères, qui permet à une machine de lire du texte imprimé ou manuscrit et de le transformer en données numériques exploitables. C'est un domaine complexe et passionnant qui demande des compétences en programmation, en traitement d'images et en analyse de données. Pour ce projet, j'ai formé un groupe de travail avec trois de mes camarades, avec qui j'ai des affinités. Le choix de mes coéquipiers n'a pas été fait au hasard : il fallait constituer une équipe de quatre personnes, et je savais que je partageais avec eux une même vision de l'informatique et une passion pour le développement. Travailler avec des personnes avec lesquelles je m'entends bien est un vrai plus, cela me permet non seulement de progresser, mais aussi de rester motivé et d'avancer dans une bonne ambiance. Ensemble,

nous nous sommes lancés dans le défi ambitieux de créer un programme OCR en C, et chacun de nous apporte ses compétences spécifiques au projet.

3.2 État d'avancement

Dans le cadre de notre projet d'OCR, je suis chargé de la partie "sauvegarde d'une lettre de la grille sous forme d'image". Ce travail consiste à isoler chaque lettre détectée et à la sauvegarder sous forme d'un fichier image .bmp. Cependant, je rencontre certaines difficultés pour avancer pleinement sans disposer du reste du programme. En effet, le processus de débogage devient complexe lorsque les autres parties, notamment celles permettant de localiser précisément les coordonnées des lettres dans l'image de départ, ne sont pas encore totalement opérationnelles. Sans ces informations de localisation, il est difficile de tester l'exactitude de mon code et de garantir que chaque lettre est bien isolée.

Malgré cette contrainte, j'ai réussi à développer et optimiser les algorithmes nécessaires pour réaliser cette tâche. Le code que j'ai écrit est structuré et bien pensé, de manière à être aussi performant que possible dans la gestion des pixels et la génération des images. Je suis confiant que les algorithmes sont corrects et qu'ils devraient fonctionner comme prévu une fois les coordonnées disponibles. En effet, l'essentiel du travail est déjà accompli : environ 80/100 de la tâche est finalisée et prête. J'ai déjà vérifié que les fonctions de base (création et écriture d'images, manipulation des pixels, etc.) fonctionnent indépendamment.

, une fois que le module permettant de localiser les lettres dans la grille sera achevé, je pourrai rapidement finaliser mon travail avec un débogage efficace. Les derniers 20/100 de la tâche consisteront essentiellement à intégrer ces coordonnées pour que chaque lettre soit correctement découpée et sauvegardée. À ce stade, je pourrai tester l'ensemble du processus et m'assurer que toutes les lettres sont bien sauvegardées individuellement, ce qui devrait se faire rapidement avec quelques ajustements finaux. En somme, bien que le débogage soit difficile sans l'ensemble du programme, je suis confiant que ma partie est bien avancée et quasiment prête pour la phase finale.

3.3 Aspects techniques

Pour réaliser la partie « reconnaissance de lettres sur une image » dans notre projet d'OCR, j'ai conçu une série d'algorithmes en C, appliqués sur des fichiers au format .bmp. Ce format bitmap, ou .bmp, est un type de fichier d'image qui stocke les informations sous forme de matrices de pixels, sans compression, ce qui permet de préserver chaque détail. Un fichier .bmp contient deux parties principales : un en-tête (ou header) qui stocke les métadonnées de l'image comme la taille, la résolution, et la profondeur de couleur, puis le tableau de pixels qui décrit chaque point de l'image. Pour la reconnaissance des lettres, il est essentiel de manipuler ce tableau de pixels en détail afin d'isoler chaque caractère.

Afin de structurer mon travail, j'ai divisé le processus de reconnaissance de lettres en plusieurs fonctions spécifiques, chacune jouant un rôle clé pour arriver au résultat final. La première de ces fonctions est la détection d'un caractère dans l'image. Ici, l'objectif est de repérer chaque pixel noir, car notre image contient des lettres en noir sur un fond blanc. J'ai donc développé une fonction capable de parcourir l'image pixel par pixel pour identifier les zones noires et reconnaître où commence et se termine chaque lettre. Cette première étape, bien que fondamentale, ne suffit pas pour isoler correctement un caractère ; elle nous aide néanmoins à repérer la présence d'un caractère à un endroit donné.

La deuxième fonction que j'ai développée concerne la délimitation des lettres, qui est probablement la partie la plus complexe de cette tâche. En effet, il ne suffit pas de détecter les pixels noirs : il faut aussi déterminer précisément la zone rectangulaire qui encadre chaque lettre. Pour cela, mon algorithme parcourt l'image pour identifier les limites exactes des caractères, c'est-à-dire les coordonnées où le caractère commence et où il finit, en hauteur et en largeur. Ce processus requiert une analyse minutieuse ligne par ligne et colonne par colonne afin de ne pas manquer un pixel noir qui ferait partie du caractère. Cela permet ensuite de définir une « boîte de délimitation » autour de chaque lettre, facilitant ainsi son extraction.

Une fois la lettre isolée, j'utilise la fonction finale pour sauvegarder la lettre sous forme d'une nouvelle image .bmp. Grâce à la bibliothèque SDL (Simple Di-

rectMedia Layer), que j'utilise dans ce projet, cette étape de sauvegarde devient plus simple. SDL est une bibliothèque en C qui facilite la gestion d'images, de pixels et d'interfaces graphiques. En utilisant SDL, j'ai pu automatiser la création d'un nouveau fichier .bmp pour chaque lettre, tout en réutilisant les données d'en-tête modifiées pour chaque nouvelle image. SDL est particulièrement utile pour manipuler les images de manière rapide et simplifiée, en fournissant des fonctions prêtes à l'emploi pour accéder aux pixels ou créer des fichiers .bmp. Dans le cadre de mon projet, SDL permet de traiter efficacement chaque lettre extraite et de la sauvegarder avec précision.

Concernant la structure d'un fichier .bmp, il est essentiel de bien comprendre la composition de son en-tête pour pouvoir enregistrer correctement chaque image. L'en-tête d'un fichier bitmap contient des informations comme la taille de l'image, le nombre de bits par pixel et le type de compression (qui est en général nul pour un fichier .bmp, ce qui explique sa taille plus importante). En manipulant cet en-tête dans mon programme, je peux créer une nouvelle image en copiant seulement les pixels qui encadrent chaque caractère isolé. La gestion des métadonnées permet de générer une image contenant uniquement la lettre, sans perdre les informations de qualité des pixels, ce qui est indispensable pour que l'OCR fonctionne correctement.

Ainsi, en structurant le code en différentes fonctions, j'ai réussi à clarifier le processus de reconnaissance de lettres et de sauvegarde en images. Chacune de ces fonctions est conçue pour être indépendante, ce qui facilite le débogage et rend le code plus modulaire. La détection des pixels noirs, la délimitation des lettres et leur sauvegarde en tant qu'images distinctes forment ainsi un ensemble logique, chaque étape s'appuyant sur la précédente. Grâce à SDL et à une gestion attentive des fichiers .bmp, mon code est optimisé et structuré, prêt pour les prochaines étapes du projet.

En somme, l'algorithme est conçu de manière à ce que, une fois les coordonnées des lettres correctement identifiées, le programme puisse isoler chaque caractère sans difficulté et les sauvegarder sous forme d'images exploitables pour la reconnaissance de caractères.

4 Otto

4.1 Présentation et Motivation

Depuis toujours, l'intelligence artificielle me fascine, en particulier sa capacité d'apprendre et d'évoluer en analysant des données. Ce projet revêt donc un intérêt particulier pour moi, car il met en œuvre des techniques de reconnaissance de lettres qui reposent précisément sur des principes d'apprentissage automatique. J'ai eu l'opportunité d'explorer plus en profondeur les mécanismes qui permettent à une machine de distinguer et de comprendre les lettres, un défi passionnant qui allie logique et créativité.

Dans ce contexte, j'ai pris en charge la conception et la mise en place du réseau de neurones, en lien direct avec ce sujet complexe. Cette étape a nécessité une réflexion approfondie sur l'architecture du réseau, les types d'algorithmes d'apprentissage, et l'optimisation des paramètres, afin d'obtenir des résultats précis et fiables. Mon implication dans ce projet m'a permis de développer mes compétences dans le domaine de l'intelligence artificielle.

4.2 État d'avancement

L'objectif initial pour l'avancement du réseau de neurones consistait à concevoir un modèle de base capable de reconnaître la formule logique "A.B + not(A).not(B)", où A et B sont deux variables booléennes. Cette formule, qui représente une fonction booléenne simple, devait servir de point de départ pour construire et tester une structure de réseau de neurones efficace avant d'aborder des tâches plus complexes, comme la reconnaissance de caractères. Ce premier jalon représenterait environ 30 pourcents du travail global sur le réseau de neurones. Cependant, le projet est actuellement moins avancé que prévu, atteignant environ 20 pourcents des objectifs, en raison de difficultés techniques rencontrées dans la phase d'implémentation.

En effet, bien que le réseau de neurones renvoie parfois des résultats corrects, les valeurs sont fréquemment erronées, et ce malgré les multiples ajustements de différents paramètres tels que la durée d'entraînement, la taille du réseau, ou encore la taille des mini-lots. Ces erreurs persistent et limitent l'efficacité du réseau, ce qui laisse penser qu'une erreur pourrait se situer dans l'algorithme de rétropropagation. Il est probable que les calculs réalisés au sein de cet algorithme soient incorrects, ou bien que les matrices de poids et de biais ne soient pas correctement initialisées. Cela pourrait conduire à des couches du réseau dont les poids et/ou les biais restent à zéro, ce qui bloque l'apprentissage efficace du modèle.

Cette situation a mis en lumière l'importance cruciale de la précision dans chaque étape de l'algorithme, particulièrement dans le calcul des gradients et dans la mise à jour des poids, des biais, et des autres paramètres essentiels pour un entraînement optimal. Malheureusement, faute de temps, ces problèmes n'ont pas encore pu être résolus. Le débogage de l'algorithme de rétropropagation et la vérification minutieuse des calculs matriciels prennent plus de temps que prévu, nécessitant des vérifications point par point pour identifier et corriger les erreurs. La correction de cet algorithme reste donc une priorité, car elle constitue un passage obligé pour que le réseau de neurones puisse pleinement remplir ses objectifs et évoluer vers des tâches de reconnaissance de caractères plus avancées.

4.3 Aspects techniques

Pour implémenter le réseau de neurones, nous avons développé une structure appelée "network" qui représente le réseau dans son intégralité. Cette structure est conçue pour être flexible et facilement extensible, permettant ainsi d'ajuster le nombre de couches et de neurones selon les besoins spécifiques des applications. Au cœur de cette structure se trouve un tableau d'entiers, qui spécifie le nombre de neurones dans chaque couche. Cette représentation simplifie la gestion de l'architecture du réseau et facilite l'initialisation des paramètres.

En parallèle, nous avons inclus un tableau de vecteurs pour gérer les biais associés à chaque couche. Les biais sont des valeurs critiques qui aident à ajuster les sorties des neurones et à améliorer la capacité d'apprentissage du réseau. Chaque élément du tableau de biais correspond à une couche spécifique, garantissant que chaque neurone peut recevoir un ajustement individuel en fonction de son rôle dans le réseau.

De plus, nous avons mis en place un tableau de matrices pour représenter les poids reliant les neurones entre les différentes couches. Dans ce tableau, l'élément à la ligne i et colonne j de la matrice d'indice n correspond au poids entre le i-ème neurone de la couche n et le j-ème neurone de la couche suivante. Cette organisation matricielle est essentielle pour effectuer les calculs nécessaires lors de la propagation des données à travers le réseau. Elle permet de multiplier efficacement les entrées par les poids pour obtenir les activations des neurones..

Pour ce réseau, nous avons opté pour l'utilisation de la fonction sigmoïde en tant que fonction d'activation. La fonction sigmoïde est une fonction d'activation couramment utilisée dans les réseaux de neurones, en particulier dans les couches cachées et de sortie. Elle transforme les valeurs d'entrée en une sortie comprise entre 0 et 1, plutot qu'en une simple valeur binaire. L'une des principales caractéristiques de la fonction sigmoïde est sa capacité à introduire une non-linéarité dans le modèle. Cela permet au réseau d'apprendre des relations complexes dans les données. Sans cette non-linéarité, un réseau composé uniquement de fonctions linéaires ne pourrait pas modéliser des phénomènes complexes.

Nous avons décidé d'entraîner le réseau de neurones via l' algorithme de descente de gradient stochastique (SGD). Contrairement à la descente de gradient classique, qui calcule le gradient de la fonction de perte sur l'ensemble du jeu de données, la descente de gradient stochastique met à jour les poids du modèle en utilisant un sous-ensemble aléatoire des données, appelé mini-batch. Cette approche permet d'accélérer le processus d'apprentissage. En mettant à jour les poids plus fréquemment, le réseau peut converger plus rapidement vers une solution optimale. Cela réduit également la charge de calcul, car il n'est pas nécessaire d'évaluer l'ensemble du jeu de données à chaque itération.

5 Alexis

5.1 Présentation et Motivation

J'ai commencé réellement à m'intéresser à l'informatique un peu avant ma seconde à travers la programmation. J'ai tout de suite accroché l'aspect challengeant poussant toujours à réfléchir un problème et sa résolution par soi même avec une possibilité et une variation énorme des réponses. J'ai surtout été ébahi devant toutes ces lignes de mots qui donnaient un résultat qui n'avait rien à voir avec du texte, ces lignes qui peuvent intéragir directement sur ce qu'affichait l'ordinateur et sur quoi et comment je manipule ce qui est à l'écran. Après cette découverte, à mon entrée en seconde, j'étais sur d'une chose : j'allais prendre la spécialité Numérique et Sciences de l'Informatique jusqu'en terminale et j'allais en faire mes études post-bac. Cette envie s'est renforcée lorsqu'en seconde j'ai assisté au cours de SNT qui ne sont ni plus ni moins une initiation à l'informatique et à l'immensité des domaines auxquels elle a de l'impact. En voyant tout c'est champs d'action dans autant de domaine me faisait dire que si j'aimais l'informatique sans pour autant savoir quoi précisément j'avais le temps car il y a tellement à découvrir. D'autant plus que ce domaine est en constante évolution depuis maintenant plusieurs années ne serait-ce que sur l'intelligence artificielle. Voilà ce qui m'a attiré en informatique, à la fois la variété des domaines d'application mais aussi les possibilités et les résultats impréssionnants de complexité avec des outils paraissant facile à manipuler.

Aujourd'hui, je suis toujours dans une mentalité de découverte et de curiosité afin d'être sur de ne pas me fermer des portes. C'est pourquoi je n'ai pas de domaine que je repousse automatiquement. J'essaye de toucher un peu à tout pour me faire un avis sur tout. Dans ce projet je m'attaque à quelquechose que je n'avais jamais fait avant ça me permet de continuer dans ma dynamique de tout faire avant de choisir pour ne passer à côté de mon potentiel futur domaine préféré. La partie technique n'est pas le seul aspect du projet sur lequel j'ai besoin de travailler mais aussi le travail en groupe. Ce travail permet de développer chez moi des qualités essetielles que ce soit pour un ingénieur ou pour momi en tant qu'humain. Par exemple, Otto, Baptiste et Lorys sont des personnes avec qui j'avais assez peu ou pas de relations avant ce projet et me mettre avec eux m'a permis de mieux les connaître, en bien évidemment, ce qui a augmenté ma confiance en moi et en mes relations avec les autres. Cet exemple n'est qu'un parmi une multitude d'améliorations humaines et techniques qui font de moi, je pense, un meilleur futur ingénieur en informatique, un meilleur collègue de travail de groupe et une meilleure personne globalement.

5.2 État d'avancement

De mon côté, je m'occuppe de la partie graphique du projet, c'est à dire tout ce qui touche à l'interface et donc aux interactions que l'utilisateur aura avec notre application. Pour cette première soutenance, je devais donc produire le code pour avoir une fenêtre de laquelle l'utilisateur peut charger son image depuis le dossier duquel l'utilisateur travail. Au chargement de cette dernière, un filtre noir et blanc sera appliqué pour facilité le travail de reconnaissance de notre alogrithme de résolution car cela augmentera le contraste de l'image ce qui permettra de mieux identifier les lettres ou la grille dans une image initialment de mauvaise qualité. Enfin l'utilisateur doit pouvoir faire tourner son image afin que cette dernière soit dans le sens adéquat. Cette dernière étape s'inscrit toujours dans cette dynamique de simplifier à la fois le travail de notre algorithme et l'algorithme en lui-même puisqu'il n'a besoin de reconnaître les lettres que dans un seul sens, le bon.

Tout d'abord en voyant les tâches qui m'ont été attribuées je pensais que cela allait être rapide car nous avons un tp sur comment manipuler des images et une fenêtre et que je m'en étais pas mal sorti. Cependant, à cause de cela je m'y suis pris un peu tard et j'ai pu faire à peine la moitié de ce que j'avais à faire en terme de tâche mais beacoup moins en terme de temps de travail. En effet les premières étapes de création de fenêtre ainsi que de chargement d'image ont été les plus simples. J'ai modulé l'interface afin de rendre l'expérience utilisateur la plus agréable possible sans pour autant y passer trop de temps dessus en ajoutant un titre en rendant la taille de l'image adaptative aux changements de taille de fenêtre. Cependant les premières difficultés sont arrivées lors de l'implémentation du filtre noir et blanc. En effet, même si j'avais l'algorithme et le code parcourant chaque pixel de l'mage afin d'assombrir ou clarifier le pixel afin d'augmenter le contraste. En effet au moment du chargement de limage, au bout d'un petit moment, le programme s'arrêtait sur une "Segmentation Error". Cette erreur est en général produite lorqu'on manipule un pointeur dans un espace mémoire qui ne lui est pas alloué. J'ai passé au moins un jour entier sur cette erreur passant en revue chaque ligne en faisant à chaque fois des petites modifications, sans trouver pour autant l'origine de cette erreur. J'ai donc pris l'intiative de laisser ma fonction commentée et de rester à la version où l'utilisateur peut simplement charger une image.

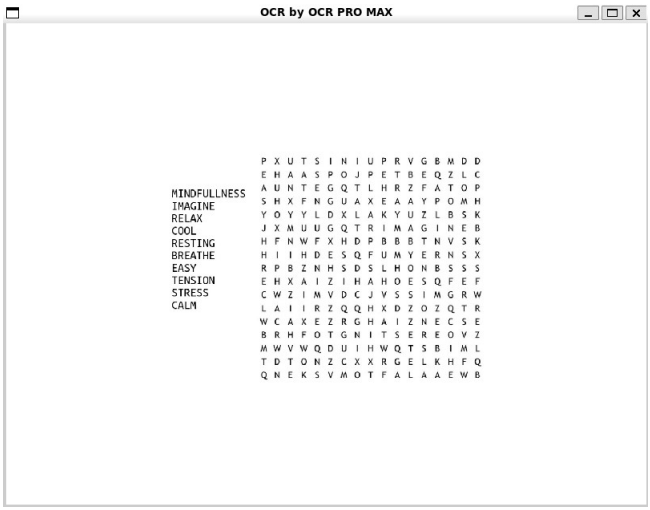
5.3 Aspects techniques

Dance cette partie je vais parler de choix que j'ai fait dans mon travail, dans le dossier Interface, afin d'avoir une idée à la fois précise et concise de mon cheminement de pensée ainsi que de son fonctionnement. Le dossier est composé d'un fichier main où tout les autres fichiers seront appelés pour exécuter entièrment le code, cela permet si nous avons besoin plus tard de créer de nouveaux fichiers dans ce dossier d'avoir un fichier qui résume l'exécution pour y voir plus clair. Il y a aussi un dossier Roboto contenant la police uilisée fournie gratuitement par Google j'ai choisi cette police car dans mes recherches c'est celle qui me semblait être la plus neutre. Enfin le plus intéressant est le fichier interface.c car c'est lui qui contient toute mon avancée. Je détaille son contenu plus bas.

Tout d'abord, j'importe dfférentes bibliothèques, certaines sont quasi indispensables pour coder en C telles que stdio.h et string.h, cependant d'autres sont moins courantes et pourtant indispenables dans la conception du projet, ce sont les différentes bibliothèques SDL. La première, SDL.h permet d'accéder au fonction de base de manipulation de fenêtre avec SDL elle permet par exemple d'ouvrir et de fermer une fenêtre. La deuxième, SDL image.h comme son nom l'indique donne accès à des fonctions de traitement d'image. C'est grâce à cette bibliothèque que l'utilisateur peut charger son image et faire varier sa taille. Enfin SDL ttf.h est liée aux options d'écriture au sein même de la fenêtre. Plus précisément c'est avec ses fonctions qu'on va pouvoir manipuler des polices d'écritures afin d'écrire dans la fenêtre.

- Fenêtre au lancement avec entrée du nom de l'image

Après ça le fichier est divisé en trois parties. La première composée essentiellement d'initialisation de pointeurs de texture, de fenêtre ou de police. La deuxième, la plus importante contient le code exécutable tant que la fenêtre est ouverte. C'est à dire qu'il y a la partie qui gère l'écriture du chemin d'accès à l'image, celle qui affiche l'image dans la fenêtre tout en conservant les proportions des éléments de la fenêtre en fonction de la taille de cette dernière. La dernière est finalement la conclusion du code puisqu'elle supprime tous les éléments qui étaient présent lors de la fermeture de la fenêtre.



- Grille chargée

6 Conclusion

Durant cette première partie du projet, nous avons fait face a de nombreux défis. Nous avons bien avancé plusieurs points, nous avons exploré des domaines que nous ne connaissions pas. Certains points sont encore entrain d'être étudié afin de pouvoir avancer dans le projet et obtenir un résultat qui correspondra au cahier des charges. Si la situation est moins avancée que prévue à ce jour c'est notamment à cause d'un manque d'oragnisation à la fois personnelle et collective du à par exemple une sous-estimation de la difficulté du projet. Mais nous avons dores et déjà établi un plan afin d'être mieux organisés et mieux préparés pour la prochaine soutenance. Finalement nous avons appris beaucoup de choses en programmation et en gestion de groupe,