

INSA RENNES

DÉPARTEMENT ELECTRONIQUE ET INFORMATIQUE INDUSTRIELLE

RAPPORT DE STAGE

Extension d'un logiciel pédagogique de traitement d'images

Auteur:
Antoine PAZAT

Maître de stage:
Luce MORIN

Du 13/06/2016 au 29/07/2016 et du 16/08/2016 au 19/08/2016

Contents

1	Introduction	3
2	Analyse du logiciel existant	4
2.1	Historique du logiciel	4
2.2	Architecture du logiciel	5
2.3	Rappel de notions d'image	5
3	Travail réalisé : aspects logiciel	6
3.1	Compilation du code source	6
3.2	Debug	6
3.3	Améliorations de l'interface	6
3.4	Documentation pour de futurs collaborateurs	7
4	Travail réalisé : aspects image	8
4.1	Filtre médian	8
4.2	Modification de la transformée de Hough	9
4.3	Nouveau codage d'Huffman	9
5	Conclusion	10
6	Annexes	11

Remerciements :

Je tiens tout d'abord à remercier le département EII de l'INSA de Rennes pour m'avoir offert l'opportunité d'effectuer ce stage. Je tiens ensuite à remercier Luce Morin, mon maître de stage, enseignant-chercheur au département EII, pour ses conseils et le temps qu'elle m'a accordé tout au long du stage. Je tiens enfin à remercier Alexandre Sanchez pour son aide et ses conseils.

1 Introduction

Ce rapport décrit succinctement le travail réalisé lors de mon stage de second cycle qui s'est déroulé lors de l'été 2016. Ce stage a eu lieu au département EII de l'INSA Rennes du 13/06/2016 au 29/07/2016 et du 16/08/2016 au 19/08/2016. L'objectif de ce stage est de modifier le logiciel ImageINSA, un logiciel de traitement d'images utilisé à des fins pédagogiques en travaux pratiques. Il s'agit d'apporter des améliorations et de corriger certains dysfonctionnement relevés pendant les séances durant l'année 2015-2016. Ce stage permettra de mettre en pratique les connaissances acquises en cours d'informatique, d'interface graphique et de de traitement d'images.

Nous commencerons par détailler l'historique du logiciel et ses principales composantes. Nous décrirons ensuite les modifications qui ont été apportées au cours de ce stage, en étudiant d'abord l'aspect informatique, puis l'aspect traitement d'image de ce stage.

2 Analyse du logiciel existant

2.1 Historique du logiciel

- Avant 2012 : IMAGE_EII
Le logiciel IMAGE_EII est développé par un enseignant du département Informatique de l'INSA Rennes. Ce logiciel est développé en C sur Visual Studio. Il est ensuite amélioré par des stagiaires et des enseignants.
- 2012-2013 : detiq-t
Un projet de 4ème année du département Informatique est encadré par Marie Babel. Il s'agit d'offrir les services concernant la modélisation et le traitement des images dans une bibliothèque : la librairie ImageIn, et de fournir des outils concernant l'interface graphique dans une autre bibliothèque : la librairie GenericInterface. Le logiciel est codé en C++ et l'interface graphique est gérée à l'aide de Qt. Le logiciel est conçu de la manière la plus générique possible et dans le respect des bonnes pratiques. Le projet s'appelle detiqt.
- Été 2013 : eiimage
Stage d'été de Sachat Percot à pour objectif le refactoring d'IMAGE_EII en utilisant le projet 4INFO comme base. Le code C déjà existant depuis 2012 pour les algorithmes est réutilisé. La fonctionnalité d'ajout de Plug-Ins est ajoutée. Logiciel EIIMAGE. Le logiciel EIIMAGE est utilisé en travaux pratiques en EII dès la rentrée 2013. Cependant, les sources correspondant à l'exécutable utilisé en TP sont perdues.
- 2014-2015 : ImageINSA
Des améliorations sont réalisées par Antoine Lorence, Ingénieur de recherche en CDD. Les sources sont rétablies et correspondent à l'exécutable utilisé en TP. Plusieurs bugs sont corrigés. La compilation se fait désormais sur CMake. Un dépôt sous GitHub est créé. Le logiciel change de nom et s'appelle désormais ImageINSA. Ce changement de nom et la création du dépôt GitHub envisage l'utilisation de ce logiciel en dehors de l'INSA.

2.2 Architecture du logiciel

L'architecture du logiciel s'explique par son historique.

Le projet Detiq-T est inclus comme librairie, et contient donc ImageIn et GenericInterface. Le coeur du logiciel ImageINSA contient toute la partie relative à l'organisation de son menu. La partie app du logiciel contient le code des algorithmes et opérations.

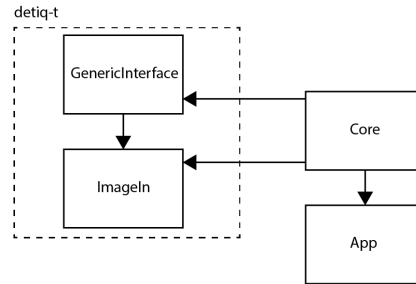


Figure 1: Architecture du logiciel ImageINSA.

2.3 Rappel de notions d'image

Il est nécessaire de rappeler comment fonctionne la représentation des images, et comment celle-ci est gérée par le logiciel ImageINSA :

Les images sont représentées de manière matricielle. Chaque valeur dans la matrice est un n-uplet dont le nombre de composantes peut varier de 1 (nuances de gris) à 4 (rouge, vert, bleu, transparence). La valeur des pixels de chaque composante peut être un entier, un entier relatif, ou un nombre réel.

Dans le cas d'images usuelles, la valeur de chaque composante d'un pixel est comprise entre 0 et 255.

Le logiciel peut traiter ces différents types d'images. L'affichage de ces images ne détériore pas les valeurs qu'elles contiennent. Cet affichage peut être enregistré comme une image normale, dans un format usuel. Cependant, l'image enregistrée sous ce type de format est uniquement la représentation affichée par le logiciel, et ne contient donc plus toutes les informations contenues par l'image.

3 Travail réalisé : aspects logiciel

3.1 Compilation du code source

A l'été 2016, au début de stage, les sources sur le github ne compilent pas, et ne permettent pas de retrouver l'exécutable utilisé en Travaux Pratiques. La première partie de ce stage consiste donc à se familiariser avec Cmake, à recompiler les sources, et à créer un environnement de travail adéquat.

Certains élèves ayant des machines linux ont aussi eu des problèmes lors de la compilation des sources pour créer un exécutable sur leur machine. Un temps a donc été pris avec un de ses élèves pour réussir à compiler le code source mis à jour sur sa machine et pour créer un exécutable linux fonctionnel.

3.2 Debug

Une fois la procédure de compilation normalisée, la seconde partie de ce stage consiste en la correction de bugs découverts et relevés au cours de l'année, pendant l'utilisation du logiciel en TP. Il s'agit donc de réussir à reproduire les bugs listés, et à déterminer les conditions d'apparitions de ces bugs avant de les corriger. Leurs causes sont très variées, et ils peuvent être divisés en deux catégories : les bugs liés à des erreurs de code, et ceux liés à des algorithmes qui ne rendent pas le résultat voulu. La deuxième catégorie relève donc plus de la compréhension des algorithmes de traitement d'Image et sera développée plus en détails dans la troisième partie de ce rapport. D'autres bugs ont été découverts lors de la reproduction des conditions, et ont été corrigés. La liste des bugs découverts peut être trouvée en Annexe.

3.3 Améliorations de l'interface

Certaines améliorations qui ne sont pas directement liées au traitement d'Image ont été apportées au logiciel. Celles-ci sont liées à l'interface graphique : réorganisation de certains menus et boîtes de dialogue, ajout d'un aperçu pour certaines transformations. Ces tâches sont principalement liées à l'utilisation de Qt, et les modifications apportées ont été faites en majorité à la librairie detiq, dans la partie GenericInterface mais certaines concernaient aussi la partie application du logiciel.

3.4 Documentation pour de futurs collaborateurs

Pour gagner du temps lors de prochains stages similaires, un temps a été accordé à la rédaction de guides de compilation et au regroupement d'informations utiles pour de futurs contributeurs. Ce rapport ayant aussi pour but de contenir des informations utiles pour ces prochains contributeurs.

4 Travail réalisé : aspects image

Cette partie concerne les algorithmes de traitement d'image modifiés ou ajoutés au cours de ce stage. Ils ne seront pas tous détaillés dans cette partie, mais des exemples qui semblent être pertinents ou représentatifs ont été sélectionnés. La liste des algorithmes ajoutés ou modifiés peut être trouvée en Annexe.

4.1 Filtre médian

Le filtre médian remplace la valeur du pixel traité par la valeur médiane calculée sur les pixels contenus dans la fenêtre du filtre. Ce type de filtre est notamment utilisé pour débruiter des images.

L'implémentation d'un filtre médian basique sous forme de plug in à fait l'objet d'une partie d'un des travaux pratiques de 4EII. Il s'agit ici d'implémenter une version plus complète, de taille et de forme paramétrable, de manière native au logiciel, pour que les élèves puissent vérifier que leur plug in donne le résultat attendu, ou puissent utiliser ce filtre au sein du logiciel pour d'autres applications. Une version complète de ce filtre sous forme de plug in a aussi été réalisé dans le cas où une correction serait nécessaire.

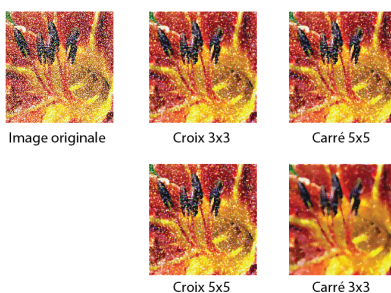


Figure 2: Filtre médian.

4.2 Modification de la transformée de Hough

La transformée de Hough est une méthode de détection de formes, qui sert le plus souvent à repérer des lignes dans une image. Cet algorithme est étudié en cours, et deux implémentations différentes de cet algorithme sont déjà intégrées dans le logiciel. Un premier travail a consisté à corriger la première implémentation de l'algorithme pour que celui-ci affiche correctement certains angles sur l'image résultat ce qui entraînait des soucis de compréhension de l'algorithme par les élèves lors de l'utilisation en travaux pratiques. Il s'est ensuite agi d'uniformiser les repères des images résultats qui étaient différents pour les deux implémentations, et aussi différent de celui présenté en cours.

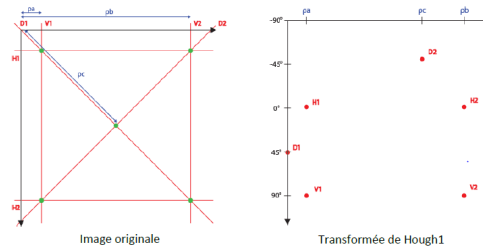


Figure 3: Exemple de transformée de Hough1.

4.3 Nouveau codage d'Huffman

Le codage d'Huffman est une méthode de compression de donnée sans perte. Il attribue le code le plus court à la valeur du pixel ayant la plus forte probabilité d'apparition dans l'image. Cet algorithme nécessite de calculer l'entropie de l'image.

Deux problèmes étaient présents dans l'implémentation de cet algorithme. Le calcul d'entropie était erroné, et ne prenait pas en compte la valeur 0 dans son calcul, ce qui aboutissait à un codage erroné. De plus, le début d'une valeur codée ne peut pas être égale à une autre valeur codée, ce qui n'était pas le cas dans l'implémentation. Le code n'était pas canonique. Ces deux problèmes ont été corrigés.

5 Conclusion

Les corrections les plus importantes et les ajouts principaux ont pu être réalisés. Cependant, l'implémentation de certains algorithmes plus complexes n'ont pas pu aboutir par manque de temps. La compilation des sources, la prise en main de certains outils et la compréhension du code existant ont représenté une part importante de la première partie du stage, que j'avais probablement sous estimée.

Ce stage a été ma première vraie expérience sur un logiciel de taille conséquente comparé aux projets que j'avais jusque là effectués. Cela m'a permis non seulement d'approfondir et de mettre en pratique mes compétences en programmation mais aussi de découvrir de nouvelles problématiques. J'ai pu ainsi prendre conscience de l'importance d'une bonne organisation et de méthodes de travail efficaces, ainsi que l'intérêt des outils comme git que j'avais jusqu'ici utilisé pour des applications beaucoup plus sommaires. J'ai pu aussi me rendre compte de que l'estimation du temps nécessaire pour mettre à bien certaines tâches n'est pas toujours triviale. Cela m'a ainsi appris à mieux repérer les éléments chronophages, notamment en ce qui concerne des bugs qui paraissent bénins à première vue, et à mieux établir la priorité entre différentes améliorations. J'ai aussi eu la chance de côtoyer d'autres élèves ayant déjà utilisé ce logiciel, et de pouvoir prendre en compte leur avis d'utilisateur au cours de mon stage, ce qui n'est la plupart du temps pas aussi aisé dans d'autres domaines.

Le logiciel ImageINSA a encore de grandes perspectives d'évolution. Deux principaux axes se dégagent. D'un côté, continuer la maintenance du logiciel et ajouter du contenu en termes d'algorithmes de traitement d'image. De l'autre, améliorer l'ergonomie et l'interface graphique du logiciel. Ces deux axes pourront être traités parallèlement par différents intervenants. La structure du logiciel rend ces améliorations faciles à implémenter. Il faudra penser à centrer les préoccupations sur l'utilisation et les besoins en Travaux Pratiques, et faire évoluer le logiciel en accord avec le cadre pédagogique qui a donné naissance à ce logiciel.

6 Annexes

Compilation		Commentaires
Windows (MinGW32)	■	
Windows (MSVC)	■	<i>Jugé inutile / non prioritaire</i>
Linux (gcc)	■	
Exécutable salle 101 fonctionnel	■	<i>Vérifier l'inclusion automatique de libwinpthread-1.dll</i>
Guide Compilation	■	
Debug		
Huffman	■	
Multiplication RVB	■	
Edition Filtre	■	
Hough1 (dernière ligne)	■	
Croissance ($\approx 360 \times 360$)	■	
DMM elt non symétrique	■	
filtre QMF	■	<i>Bug non reproductible</i>
Rétablir fenêtres fermées	■	
Arrangement fenêtres	■	<i>Bug non reproductible</i>
Grille de pixel bas	■	
Ajouts		
Entropie valeurs < 0	■	
Quantificateur Lloyd-Max	■	
jpeg2000	■	<i>Jugé inutile / non prioritaire</i>
Raccourcis zoom	■	
Décomposition HSV	■	
Affichage logarithmique	■	
PlugIn Filtre médian	■	
Intégration médian dans le menu filtre	■	
Aperçu seuillage	■	
Repères Hough1 & Hough2	■	
Légende		
	■	Implémenté - Validé
	■	Implémentation en cours - Pas encore validé
	■	Non implémenté
	■	Implémentation non nécessaire

Figure 4: Liste des modifications et bugs découverts.

Abstract

Ce stage de quatrième année a été réalisé au département EII de l'INSA Rennes du 13 Juin 2016 au 29 Juillet 2016 et du 16 au 19 Août 2016. Le sujet de ce stage était la maintenance et la modification d'un logiciel de traitement d'image, utilisé a des fins pédagogiques dans le département EII.

This fourth year internship took place at the Electronics and Computer Science department of the INSA Rennes, from June 13 to July 29 and from August 16 to August 19 of the year 2016. The internship's topic was about maintaining and modifying an image processing software used for pedagogic purpose at Electronics and Computer Science department.