ALVES Vincent
HYACINTHE Clément
TESTIER Marc-Antoine

Projet d'informatique différencié: Réalisation d'un panorama d'images

2 novembre 2013

Table des matières

1 Objectif

Dans ce premier rapport, nous allons aborder la problématique du panorama d'images et son état de l'art.
Pour ce faire, nous allons parler de nos recherches sur le sujet en abordant chaque principe nécéssaire à la réalisation du panorama d'images.
Nous parlerons également de nos recherches et de notre réflexion sur les différents principes vus, afin de conclure sur quels éléments développer, lesquels abandonner, et la planification de la suite du projet.

1.1 Recherche initiale

Notre objectif durant ce projet est de concevoir un programme pouvant créer une image panoramique, le même type que celles crées par des appareils photographiques spécialisés,) à partir d'un set de photos d'un même endroit prises à des angles différents.

Combined image

Left image

Right image

Nous avons commencé nos recherches en regardant les rapports publiés sur le net de projets similaires.
La plupart n'utilisaient pas le language C, mais des programmes tels que Matlab.
Cependant, nous avons pu, après avoir lu plusieurs rapports, trouver les étapes principales à la création de panoramas d'images.

2 Fonctions de prétraitement

Comme demandé par le client, nous avons codé quelques fonctions de prétraitement d'image, afin de les utiliser dans le programme final et d'améliorer notre maîtrise du language C.
Toutes ces fonctions ont été conçues pour travailler avec le format .ppm(portable pixmap), qui est un format "raw" d'image, sans aucune compression.
De ce fait, les fichiers résultant sont très lourds, mais faciles à modifier, car il suffit de les ouvrir dans un éditeur de texte pour directement voir les codes RGB de tous les pixels de l'image.

2.1 Fonction de niveau de gris

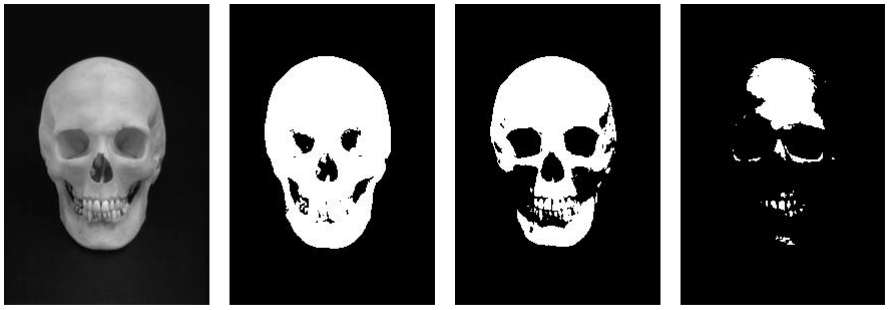
Le passage d'une image en couleur vers une image en niveau de gris est une fonction simple à réaliser, chaque pixel d'une image étant décomposé en 3 couleurs : rouge, vert bleu, il suffit d'appliquer une formule simple pour obtenir un niveau de gris :

$$Gris = 0{,}299.Rouge + 0{,}587.Vert + 0{,}114.Bleu$$

On applique alors cette formule à chaque Pixel, en remplaçant les 3 éléments du Pixel (Rouge, Vert, Bleu) par le résultat de l'équation.

2.2 Fonction de binarisation

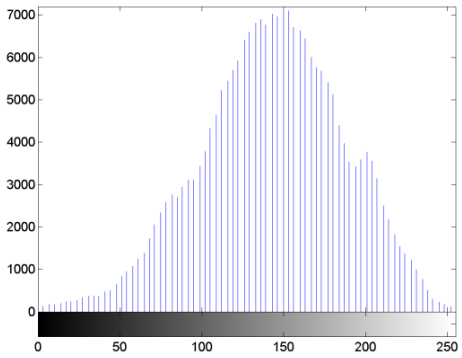
La fonction de binarisation s'applique sur une image en niveau de gris et consiste à rendre l'image uniquement en noir et blanc. L'image étant en gris, chaque « couleur » d'un pixel possède la même valeur, on regarde si cette valeur est supérieur ou inférieur à un seuil, donné au préalable par l'utilisateur, puis on transforme le pixel en noir ou en blanc.



Niveaux de gris | Seuil = 76 | Seuil = 128 | Seuil = 176

2.3 Fonction d'histogramme

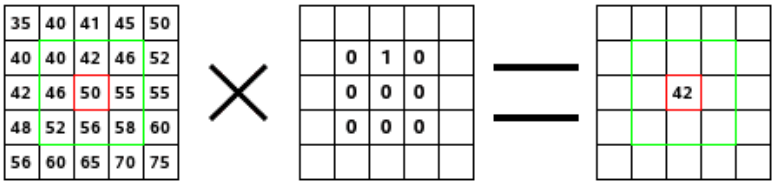
L'histogramme en niveaux de gris d'une image consiste à indiquer, pour chaque valeur entre le noir et le blanc, combien de pixels de cette valeur existent. On peut alors mettre toutes ces données dans un tableau et créer l'histogramme.



2.4 Fonction de convolution

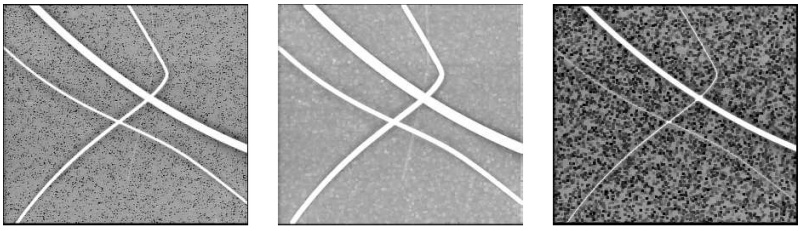
La convolution consiste à transformer un pixel d'une image en se basant sur les pixels environnants. On utilise pour cela des masques de convolution. Un masque de convolution est un tableau regroupant des coefficients permettant de calculer le nouveau pixel.

Par exemple :



2.5 Fonction d'érosion et dilatation

La dilatation consiste à éliminer les pixels noirs isolés d'une image. Pour cela, on regarde pour chaque pixel noir, les pixels environnants et suivant le nombre de pixels noirs et blancs, on transforme ou non le pixel. L'érosion est le principe inverse, il consiste à éliminer les pixels blancs isolés selon le même principe.

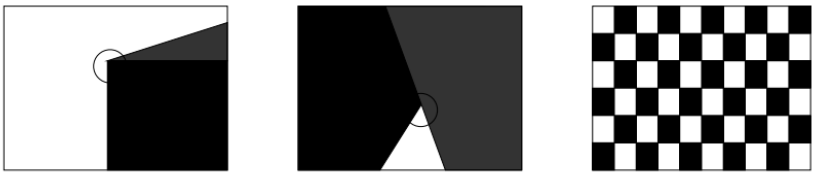


Original | Dilaté | Erodé

3 Les points d'intérêt

3.1 Comment déterminer si deux images peuvent former un panorama ?

Un des éléments les plus important lorsqu'on colle deux images est de savoir si au moins, les deux images montrent la même scène depuis des angles différents.
Pour ce faire, nous utilisons la méthode de détection de points d'intérêt, qui est,au même titre que la détection de contours, une étape préliminaire à de nombreux processus de vision par ordinateur.
Les points d'intérêts, dans une image, correspondent à des doubles discontinuités de la fonction d'intensités. Celles-ci peuvent être provoquées, comme pour les contours, par des discontinuités de la fonction de réflectance ou des discontinuités de profondeur. Ce sont par exemple : les coins, les jonctions en T ou les points de fortes variations de texture.



Différents types de points d'intérêts : coins, jonction en T et point de fortes variations de texture.

3.2 Comment trouver ces points d'intérêt ?

De nombreuses méthodes ont été proposées pour détecter des points d'intérêts.
Elles peuvent être classées grossièrement suivant trois catégories :

1. Approches contour : : L'idée est de détecter les contours dans une image dans un premier temps. Les points d'intérêts sont ensuite extraits le long des contours en considérants les points de courbures maximales ainsi que les intersections de contours.
2. Approches intensité : L'idée est cette fois-ci de regarder directement la fonction d'intensité dans les images pour en extraire directement les points de discontinuités.
3. Approches à base de modèles : Les points d'intérêts sont identifiés dans l'image par mise en correspondance de la fonction d'intensité avec un modèle théorique de cette fonction des point d'intérêts considérés.

Les approches de la deuxième catégorie sont celles utilisées généralement.

3.3 Algorithmes de recherche de points d'interêt

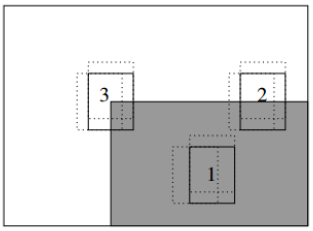
Après recherches, nous avons trouvé deux algorithmes de détection pouvant être utilisés.

3.3.1 Le détecteur de Moravec

L'idée du détecteur de Moravec est de considérer le voisinage d'un pixel (une fenêtre) et de déterminer les changements moyens de l'intensité dans le voisinage considéré lorsque la fenêtre se déplace dans diverses directions.
Plus précisément, on considère la fonction :

$$E(x,y) = \sum w(u,v)|I(x+u,y+u) - I(u,v)|^2$$

Où w spécifie la fenêtre/voisinage considérée (valeur 1 à l'intérieur de la fenêtre et 0 à l'extérieur), $I(u,v)$ est l'intensité au pixel (u,v), et $E(x,y)$ représente la moyenne du changement d'intensité lorsque la fenêtre est deplacée de (x,y).



Les différentes situations considérées par le détecteur de Moravec.

En appliquant cette fonction dans les trois situations principales suivantes (voir la figure ci-dessus), on obtient :

1. L'intensité est approximativement constante dans la zone image considérée : la fonction E prendra alors de faibles valeurs dans toutes les directions (x,y)
2. La zone image considérée contient un contour rectiligne : la fonction E prendra alors de faibles valeurs pour des deplacements (x,y) le long du contour et de fortes valeurs pour des déplacements perpendiculaires au contour.
3. La zone image considérée contient un coin ou un point isolé : la fonction E prendra de fortes valeurs dans toutes les directions.

En conséquence, le principe du détecteur de Moravec est donc de rechercher les maxima locaux de la valeur minimale de E en chaque pixel (au dessus d'un certain seuil).

3.3.2 Le détecteur de Harris

Le détecteur de Moravec fonctionne dans un contexte limité. Il souffre en effet de nombreuses limitations.
Harris et Stephen ont identifié certaines limitations et, en les corrigeant, en ont déduit un détecteur de coins très populaire : le détecteur de Harris.
Les limitations du détecteur de Moravec prises en compte sont :

1. La réponse du détecteur est anisotropique en raison du caractère discret des directions de changement que l'on peut effectuer (des pas de 45 degrés). Pour améliorer cet aspect, il sufdit de considérer le developpement de Taylor de la fonction d'intensité I au voisinage du pixel (u, v) :

   $$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y$$

   D'ou

   $$S(x,y) \approx \sum_u \sum_v w(u,v)\ (I_x(u,v)x + I_y(u,v)y)^2$$

2. La réponse du détecteur de Moravec est bruitée en raison du voisinage considéré. Le filtre w utilisé est en effet binaire (valeur 0 ou 1) et est appliqué sur un voisinage rectangulaire. Pour améliorer cela, Harris et Stephen proposent d'utiliser un filtre Gaussien :

   $$w(u,v) = exp - (u^2 + v^2)/2\theta^2$$

3. Enfin, le détecteur de Moravec repond de manière trop forte aux contours en raison du fait que seul le minimum de E est pris en compte en chaque pixel. Pour prendre en compte le comportement général de la fonction E localement, on écrit :

   $$E(x,y) = (x,y)M(x,y)^t$$

   Avec M une matrice caractérisant le comportement local de la fonction E.

   $$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

   Les valeurs propres de cette matrice correspondant aux courbures principales associées à E.
   Si les deux courbures sont de faibles valeurs, alors la région considérée a une intensité approximativement constante.
   Si une des courbures est de forte valeur alors que l'autre est de faible valeur alors la région contient un contour.
   Si les deux courbures sont de fortes valeurs alors l'intensite varit fortement dans toutes les directions, ce qui caractérise un coin.
   Par voie de conséquence, Harris et Stephen propose l'opérateur suivant pour détecter les coins dans une image :

   $$R = Det(M) - kTrace(M)^2$$
   $$\text{avec } Det(M) = AB - C^2 \text{ et } Trace(M) = A + B.$$

4 L'homographie

4.1 Comment transformer nos images : La matrice homographique

Admettons que nos images soient des tableaux de points 2D (x, y) .
Cesdits points peuvent être représentés en tant que vecteurs 3D $(x1, x2, x3)$ ou $x = x1/x3$ et $y = x2/x3$.
Ceci est la représentation homogène d'un point, qui permet de placer ce point dan le plan projectif P^2.

Un plan projectif est une structure géométrique qui s'attache au concept original du plan. Dans un plan Euclidien standard, des lignes parallèles ne se croiseront jamais. Sur un plan projectif, on ajoute des points à l'infini ou ces dites lignes se croisent, afin d'imiter l'impression de perspective.

Une homographie est une transformation inversible de points et de lignes de P^2 à lui-même de telle sorte que trois points existent sur la même ligne si et seulement si leurs points transformés sont également colinéaires.
Elle est représentée par une matrice H 3x3 de telle sorte que pour tout point de P^2 représenté par un vecteur x, sa transformée est égale à Hx.
Ainsi, pour calculer l'homographie qui fait correspondre chaque x_i à son x'_i, il faut calculer cette dite matrice H.

Imaginons les images que nous voulons attacher pour former un panorama. Ces images auront forcément des parties en commun, c'est-à-dire des représentations des mêmes objets vus de différents angles. Ces angles différents correspondent à nos différents plans projectifs. Donc, si on arrive à déterminer l'homographie correspondant à la transformation entre ces deux angles, il nous est possible de transformer une des images via la matrice homographique afin d'obtenir deux images coïncidant parfaitement. En répétant cette procédure pour toutes les images, on obtient ainsi notre panorama complet.

Les matrices homographiques sont typiquement calculées en trouvant des points d'intérêt communs aux deux images.

Exemple de points d'intérêt entre deux images.

4.2 Autres usages de l'homographie

Il existe de nombreuses situations dans les traitement d'images ou l'estimation d'homographie est requise. Lors de nos recherches, nous avond pu nous rendre compte ainsi de l'importance de la méthode.

4.2.1 Calibration d'appareil photographique

Ce processus consiste à déterminer les paramètres intrinsèques d'un appareil photos, tels que la distance focale, le point principal et la distorsion de la lentille. Il faut également déterminer d'autres paramètres, tels que la position et l'orientation de l'appareil. La calibration est souvent la première étape de nombreuses applications visuelles, car elle permet de relier l'image et sa position dans le monde réel. Pour ce faire, il faut connaître la matrice de calibration de l'appareil K, qui permet d'enlever plusieurs défauts et distorsions. En prenant plusieurs images d'un même motif) partir d'angles différents et en calculant l'homographie correspondante, il est possible de retrouver cette matrice.

4.2.2 Reconstruction 3D

Ce problème arrive souvent en images de synthèse, ou il faut reconstruire des scènes et des positions de caméra à partir d'images de la scène. Un bon exemple de ce processus est l'imagerie médicale, ou l'on construit un modèle 3D d'organes à analyser à partir de plusieurs images. (Par exemple, le cerveau.) La résolution d'homographies est essentielle en reconstruction 3D, car cela permet d'obtenir les relations entre les différentes images obtenues.

4.2.3 Métrologie visuelle

La métrologie est une branche de la physique concernant la « science des mesures et ses applications ». L'objectif de la métrologie visuelle est donc d'estimer les distances et les tailles entre plusieurs objets en se basant sur des images de ces objets. Les algorithmes de métrologie visuelle cherchent à automatiser la mesure, et pour ce faire l'estimation d'homographie est essentielle, car elle permet de mette plusieurs images d'un plan en commun pour obtenir les mesures correctes, au lieu de faire des mesures manuelles.

4.2.4 Vision stéreo

La vision stéreo est un processus de perception visuelle permettant d'obtenir un sens de profondeur à partir de plusieurs vues d'une scène, prises en utilisant différentes positions d'appareil photographique. En analysant la distance entre deux images d'un même point, on peut déterminer la profondeur. La stéreo est un problème souvent abordé en traitement d'image, et de nombreux algorithmes existent pour retrouver les propriétés stéreo d'une scène composée d'images. L'élément clé de la plupart de ces algorithmes est la recherche de correspondance de points dans les images. Pour ce faire, on peut utiliser les homographies.

Au final, on peut dire que l'homographie est utilisée par toutes les disciplines ou il est nécessaire de recréer une scène 3D à partir d'une ou plusieurs d'images 2D.

4.3 Comment calculer la matrice homographique ?

Il existe de multiples algorithmes permettant de calculer des matrices homographiques.
Le plus simple est l'algorithme DLT (Direct Linear Transformation)
Vu que nous sommes en coordonées homogènes, la relation entre les points des deux images est la suivante

$$\mathbf{c}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = H\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Avec c une constante différente de 0 et H notre matrice homographique $\begin{pmatrix} h_1\, h_2\, h_3 \\ h_4\, h_5\, h_6 \\ h_7\, h_8\, h_9 \end{pmatrix}$

On peut résoudre cette équation en utilisant la méthode du pivot de Gauss. Il nous faut au minimum 4 paires de points d'intérêt pour une matrice fiable, soit 4 équations.

La plupart des autres methodes emploient des lignes, des cones ou un mélange des deux et sont bien plus complexes que le DLT, même si elles sont plus précises.

4.4 Obtenir les paires de points d'intérêt automatiquement via les homographies

Une fois notre calcul d'homographie fonctionnel, l'étape suivante serait d'éviter l'entrée manuelle des paires de points d'inérêt.

Il existe de multiples algorithmes permettant de détecter des points d'intérêt automatiquement. Nous avons vu le détecteur de Moravec, qui semble assez simple à implémenter vu son age, ainsi que celui de Harris.
Cependant, comment faire pour déterminer les points d'intérêt communs aux deux images ? C'est là qu'intervient l'algorithme RANSAC.

Le RANSAC(RANdom SAmple Consensus) est la méthode la plus utilisée pour le calcul d'homographies. Elle prend à chaque itération 4 correspondances au hasard et calcule une homographie H. Ensuite, chaque nouvelle paire de points est classée par rapport à sa concurrence à H. Quand toutes les itérations ont eu lieu, on choisit celle avec le plus de paires compatibles. On recalcule alors H en fonction de toutes les paires considérées comme compatibles dans cette itération.

C'est un algorithme au concept assez simple mais qui requiert des paramètres assez précis (nombre d'itération, comment déterminer la compatibilité, etc).
Si nous choisissons RANSAC, il nous faudra plus de recherche sur son implémentation afin d'éviter un algorithme demandant trop de calculs.

Cependant, lors de nos recherches, nous avons vu plusieurs programmes n'implémentant pas cette recherche et demandant à l'utilisateur d'entrer les paires de points manuellement.
Nous pensons donc que cette fonction, bien qu'une amélioration évidente, n'est pas forcément une nécessité.

4.5 Implémentations déjà existantes du calcul d'homographies

4.5.1 Via MATLAB

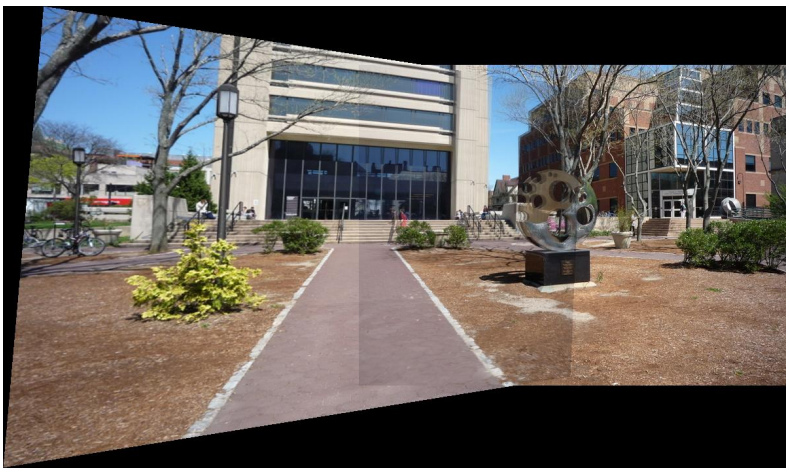
De nombreux projets de création de panorama que nous avons étudiés ne se concentraient pas sur une création de programme complet en utilisant le C, mais sur un algorithme se basant sur MATLAB pour effectuer tous les calculs. De ce fait, il est devenu plus difficile de trouver des ressources sur le calcul des homographies et la manière dont on transforme les images(cela est d'ailleurs l'un de nos problèmes).

4.5.2 Via OpenCV

OpenCV (pour Open Computer Vision) est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. Cette librairie est disponible en C/C++, et contient plusieurs fonctions automatisant le calcul et le traitement des homographies.

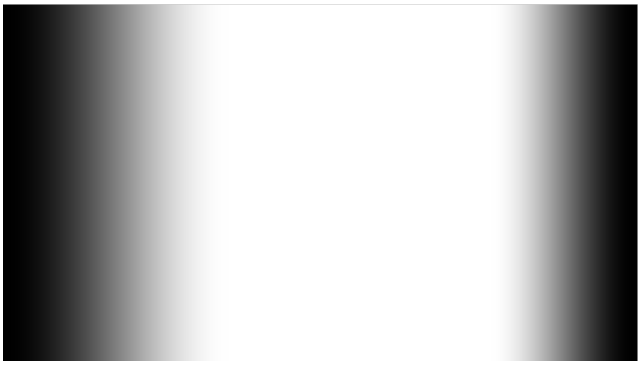
4.6 Problème 1 : Différences entre les images fusionnées

Un problème de la création de panorama est la différence entre les images utilisées.
Lorsqu'on colle les images ensemble, pour chaque pixel existant dans les deux images, la méthode la plus simple est de prendre la moyenne des couleurs des deux pixels. Cependant, cela peut amener des défauts dans l'image finale.
En effet, avec deux images montrant une scène sous un angle différent, il peut y avoir des différences de lumière ou de couleurs, comme montré ci-dessous, ou les différences viennent d'un léger déplacement de l'appareil photographique.



4.6.1 Résolution via masque alpha

Nous pouvons résoudre le problème assez simplement en appliquant un masque aux images avant de les transformer pour atténuer l'intensité des défauts en agissant sur le paramètre alpha(la transparence) des pixels aux bouts des images.



Un masque alpha que nous pourrions utiliser.

Cette méthode est simple à implémenter, mais peut laisser quelques défauts visibles.

4.6.2 Résolution via "Graph Cut"

La méthode du "Graph Cut" consiste à faire en sorte que lorsqu'on colle deux images ensemble, lorsqu'un pixel existe dans les deux images, on n'en prend qu'un, au lieu de faire une moyenne. Le résultat est plus propre que celui du masque alpha, car il ne laisse aucun défaut.

4.7 Problème 2 : Comment appliquer la matrice homographique à nos images pour les transformer ?

Un autre problème est que malgré les recherches que nous avons faites, nous n'avons pas réussi à comprendre comment transformer les images une fois la matrice homographique calculée. Nous avions pensé pendant un moment que nous pourrions utiliser la fonction de convolution calculée précedemment, mais les matrices de convolution ne modifiant que les couleurs d'une image, cela n'est pas possible.
Des recherches sur le net nous ont amené à trouver plusieurs formules mathématiques, qui semblent cependant encore plus obscures que celles utilisées pour calculer les matrices homographiques.

5 Maintenant, que faire ?

5.1 Nos conclusions : Quels éléments développer et lesquels abandonner

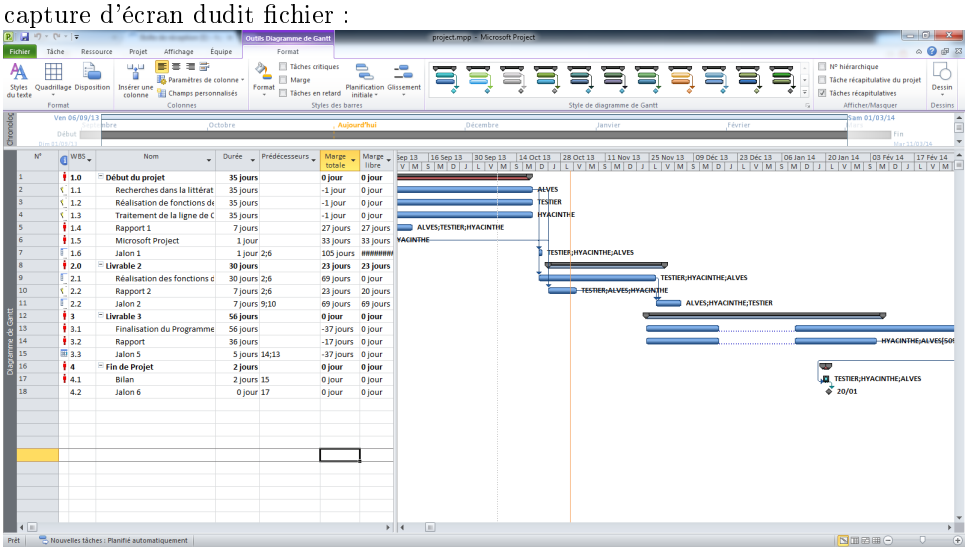
Pour la suite de ce projet, il nous est d'abord nécessaire de parfaitement comprendre le calcul des matrices homographiques(l'algorithme DLT semble être le meilleur à implémenter), puis de trouver comment les appliquer à nos images afin de les transformer.
Le collage des images est en soi assez simple à faire si nous utilisons le masque alpha. Le graph cut est plus complexe à utiliser, car il demande une bonne maîtrise de la théorie des graphes.

Quand à la détection des points d'intérêt, il serait mieux de réussir à implémenter RANSAC, mais vu la difficulté de l'homographie, il est fort probable que nous restions sur une détection manuelle des paires de points d'intérêt.

5.2 Planning Prévisionnel

Nous avons décidé d'utiliser le logiciel Microsoft Project pour concevoir un planning pour le reste de la conception du projet.

Nous avons joint le fichier correspondant à ce rapport. Voici également une capture d'écran dudit fichier :



6 Bibliographie

Automatic Panoramic Image Stitching using Invariant Features - Matthew Brown et David G. Lowe
http://www.cs.ubc.ca/~lowe/papers/07brown.pdf

Homography Estimation - Elan Dubrofsky
https://www.cs.ubc.ca/grads/resources/thesis/May09/Dubrofsky_Elan.pdf

CS 195-G : Automated Panorama Stitching - Evan Wallace
http://cs.brown.edu/courses/csci1950-g/results/proj6/edwallac/

Project 4 : Auto Stitching Photo Mosaics - William Wedler
http://www.cs.cmu.edu/afs/andrew/scs/cs/15-463/f07/proj4/www/wwedler/

Détection de points d'interêt
http://devernay.free.fr/cours/vision/pdf/c4.pdf

ProjectiveHomography Class Reference
http://www.clemson.edu/ces/crb/students/vilas/projects/cvutils/docs/Homography/htm/classProjectiveHomography.htm