



Ecole Nationale des  
Sciences Géographiques

Projet Géomatique

Cycle des Ingénieurs diplômés de l'ENSG 2<sup>ème</sup> année

---

# Réalisation d'une interface pour MICMAC

---

Réalisé par

Lionel Benoit  
Hugues Lepetit  
Céline Puig

Février 2010 - Mai 2010

# RAPPORT

---

## 1. Contexte

On assiste actuellement au développement de logiciels de corrélation de plus en plus performants qui permettent de répondre à des problématiques diverses. Un des domaines d'application de tels logiciels est l'extraction de données en 3D à partir d'un ensemble de photos 2D préalablement orientées.

MICMAC est un logiciel de recherche (développé par M. Pierrot-Deseilligny, chercheur au Mathis) dédié à la corrélation qui permet de traiter de nombreuses configurations de prise de vue et de répondre à plusieurs problèmes posés dans le contexte de l'information géographique (recherche de points homologues, superposition de canaux, réalisation de modèles d'élévation...). Il peut donc être utilisé pour réaliser des cartes de profondeur ou des modèles 3D.

Cependant, certaines difficultés d'utilisation se posent pour un utilisateur non spécialisé : MICMAC nécessite des fichiers de paramètres difficiles à comprendre et donc à réaliser, les orientations des images doivent être réalisées à l'aide d'APERO qui est également un logiciel de recherche difficile à utiliser, et le lancement de MICMAC se fait en ligne de commande.

## 2. Objectifs

Le principe de ce projet est donc d'intégrer MICMAC dans une chaîne de logiciels relativement facile d'utilisation pour réaliser des cartes de profondeur dans le cadre de la photogrammétrie terrestre.

Pour calculer l'orientation des images, on dispose au début de la chaîne du logiciel Redresseur du DIAS (développé par M. Egels) qui permet de réaliser facilement des aérotriangulations (terrestres ou aériennes) même pour des utilisateurs peu spécialisés.

Le but du projet est alors de réaliser une interface graphique qui répond à deux objectifs : récupérer les données d'orientation des images contenues dans les fichiers en sortie de Redresseur, et créer un environnement graphique permettant de renseigner certains paramètres simples de corrélation et de lancer MICMAC. Cette interface crée automatiquement les fichiers de paramètres nécessaires à MICMAC grâce aux fichiers Redresseur et aux données apportées par l'utilisateur dans l'environnement graphique.

Cette interface permet donc d'utiliser MICMAC pour réaliser des cartes de profondeur en photogrammétrie terrestre en simplifiant son usage pour le mettre à disposition d'utilisateur non spécialisés. Cependant, ce faisant, on perd une partie de la richesse de MICMAC qui est justement de pouvoir traiter la plupart des problèmes de corrélation de l'information géographique.

## 3. Réalisation

### *1. Interface réalisée*

Pour atteindre ces objectifs, nous avons choisi de réaliser une interface graphique qui prend comme seuls paramètres d'entrée les fichiers issus de Redresseur qui donnent les orientations des images et un petit nombre de paramètres de corrélation qui sont renseignés par l'utilisateur.

L'environnement graphique permet donc d'entrer l'adresse des fichiers Redresseur, de préciser les paramètres de corrélation et de lancer MICMAC.

De plus, l'interface dispose d'une aide qui explicite son utilisation, décrit assez simplement les différents paramètres de corrélation que l'utilisateur est amené à faire varier et donne la signification d'erreurs pouvant survenir lors de son utilisation et du lancement de MICMAC ainsi que les solutions à apporter pour y remédier.

### *2. Etapes de réalisation*

La réalisation de cette interface a suivi les étapes suivantes :

#### **a. Etude des fichiers d'orientation des images issus de Redresseur :**

Rencontre avec M Egels pour prendre en main Redresseur et comprendre la structure des fichiers qui sortent du logiciel ainsi que les conventions utilisées pour l'expression des données photogrammétriques (repères image et terrain, matrice d'orientation...). A l'issue de cette phase, nous avons listé quels paramètres d'orientation nous allions utiliser et dans quels fichiers ils se trouvaient.

#### **b. Etude des fichiers de paramètres de MICMAC et choix des paramètres modifiés par notre interface :**

Rencontre avec M Pierrot-Deseilligny pour connaître la signification de tous les paramètres des fichiers d'entrée de MICMAC et savoir lesquels sont les plus importants dans le cas de la photogrammétrie terrestre. De plus, il nous a expliqué la structure des fichiers de paramètres. A l'issue de cette phase, nous avons choisi quels paramètres allaient figurer dans notre interface.

#### **c. Test des logiciels :**

Prise en main de MICMAC sur des jeux test, puis sur nos propres jeux de données dont l'orientation des images a été obtenue grâce à Redresseur.

#### **d. Programmation :**

Lors de cette étape nous avons tout d'abord choisis les langages que nous allions utiliser pour réaliser notre interface. Nous nous sommes tournés vers le C++ et la

bibliothèque Qt car l'interface doit être multi-plateforme. Ensuite nous avons codé les différentes classes que nous nous étions préalablement réparties et nous avons régulièrement assemblé notre code, ceci jusqu'à obtenir une première version d'interface.

#### **e. Test de l'interface et réalisation de l'aide :**

Nous avons effectué des tests sur différents chantiers qui, soit nous ont été donnés par Mme Héno (l'aérotriangulation sous Redresseur était déjà faite), soit ont été réalisés à l'aide de Redresseur. Lors de cette phase nous avons tout d'abord corrigé notre code en fonction des problèmes qui ont été identifiés et ensuite nous avons essayé d'augmenter l'ergonomie et la convivialité de l'interface.

Ensuite, nous avons étudié les résultats obtenus lors de tests en faisant varier les paramètres pour voir leur influence sur la carte de profondeur obtenue.

Enfin, nous avons réalisé une aide qui répond aux questions qui ont été soulevées lors de la phase de test (quelles erreurs arrivent souvent et comment y remédier, quelle est l'influence des différents paramètres de corrélation, comment gérer le masque...).

#### **f. Réalisation de la documentation :**

Pour conclure ce projet nous avons réalisé deux documentations : une documentation utilisateur qui explique de façon plus détaillée que l'aide comment utiliser l'interface, et une documentation programmeur qui explicite nos choix de programmation et l'architecture du logiciel.

## **4. Difficultés rencontrées**

Au cours de ce projet, deux points ont nécessité une attention particulière :

Tout d'abord la prise en main de MICMAC et la compréhension des différents fichiers de paramètres. Cette étape a été assez difficile car MICMAC est assez peu documenté surtout en ce qui concerne son utilisation et parce que sa capacité à traiter de nombreux cas de figures le rend assez complexe.

Ensuite, l'autre difficulté rencontrée a été la gestion des conventions d'écriture des paramètres photogramétriques entre MICMAC et Redresseur. On retrouve là le problème assez courant de la non normalisation des conventions en photogrammétrie (écriture de rotations, gestion des repères image...).

## 5. Améliorations possibles et perspectives

Certaines améliorations peuvent être apportées à la version actuelle de notre interface et de la chaîne de logiciels dans laquelle elle se trouve.

Tout d'abord, il pourrait être intéressant de pouvoir créer un masque qui permette de réaliser une carte de profondeur sur une zone dépassant l'emprise d'une seule image. Ceci est possible dans MICMAC mais n'est pas pris en charge dans notre interface.

Ensuite, une fois cette modification effectuée, il serait assez simple de modifier les valeurs par défaut des paramètres de corrélation et de compléter l'aide pour permettre le traitement de chantiers aériens.

Enfin, pour rendre la chaîne plus efficace, on pourrait imaginer l'ajout d'une option dans Redresseur qui permettrait la détection automatique de points de liaison pour simplifier l'étape d'orientation des images. Cette option de grande automatisation a d'ailleurs été choisie par l'IGN qui est en train de développer une interface pour Pastis (recherche de points homologues), Apéro (orientation des images) et Micmac (corrélation) . Cette interface permettra de réaliser des cartes de profondeur dans des cas similaires à ceux traités par la présente chaîne « Redresseur + Interface » réalisée lors de ce projet mais avec une orientation des images totalement automatisée.

# DOCUMENTATION UTILISATEUR

---

## *SOMMAIRE*

<b>1. PRESENTATION DE L'INTERFACE</b>	<b>7</b>
<b>2. UTILISATION DE L'INTERFACE</b>	<b>7</b>
1. Installation de l'interface	7
2. Démarrage de l'interface	7
3. Utiliser l'aide	9
4. Remplir les adresses	9
5. Les paramètres de corrélation	10
6. Ajouter les images	10
7. Utilisation du masque	12
a. Le masque manuel	12
b. Le masque automatique	12
8. Lancement du calcul	12
9. Les erreurs	13
a. Les erreurs de l'interface	13
b. Les erreurs de Micmac	14
<b>3. INFLUENCE DES PARAMETRES DE CORRELATION</b>	<b>16</b>
1. Taille de la fenêtre de corrélation	16
2. Poids de la régularisation	16
3. Pas de quantification	16
4. Contour de recherche (altimétrique et planimétrique)	16
5. Intervalle de profondeur	17

# 1. Présentation de l'interface

L'interface est une passerelle entre deux logiciels de photogrammétrie : Redresseur (créé par M. Yves Egels) et Micmac (créé par M. Marc Pierrot-Deseilligny).

Le premier logiciel permet de réaliser une aérotriangulation de plusieurs images. Il crée un fichier de chantier .RED et un export .APX.

Le logiciel Micmac permet de faire une corrélation de ces images.

Le but de cette interface est donc, à partir d'images pointées sous Redresseur, de lancer une corrélation grâce à Micmac. Le tout se fait de manière transparente pour l'utilisateur qui doit juste rentrer les adresses des fichiers et quelques paramètres de corrélation.

## 2. Utilisation de l'interface

### *1. Installation de l'interface*

L'interface se présente sous la forme d'un dossier comprenant les librairies nécessaires au lancement de l'interface et d'un .EXE pour la version Windows ou d'un .APP pour la version Unix.

Pour lancer l'interface, il suffit de double-cliquer sur le .EXE (.APP).

### *2. Démarrage de l'interface*

Lors du lancement de l'interface, une première fenêtre apparaît, l'appui sur le bouton Valider, après avoir sélectionné le bon système d'exploitation, ouvrira la fenêtre principale.

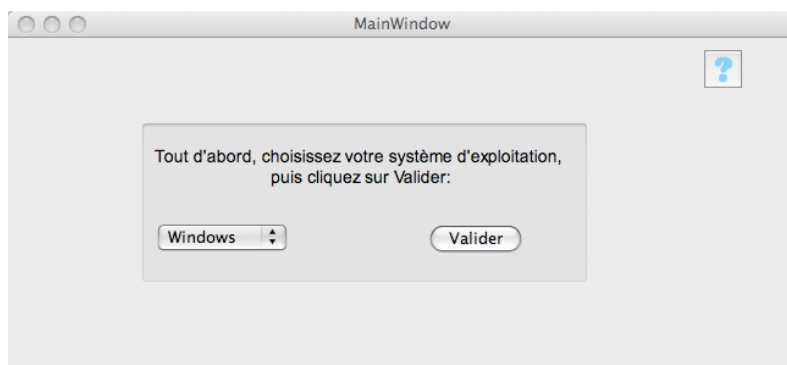


Figure 1: fenêtre de démarrage





### 3. Utiliser l'aide



Cette aide peut-être utilisée à tout moment par l'utilisateur par un simple clic sur l'icône ci-contre. Elle est divisée en 5 parties : Erreurs, Paramètres de corrélation, Images, Masque et Calcul.

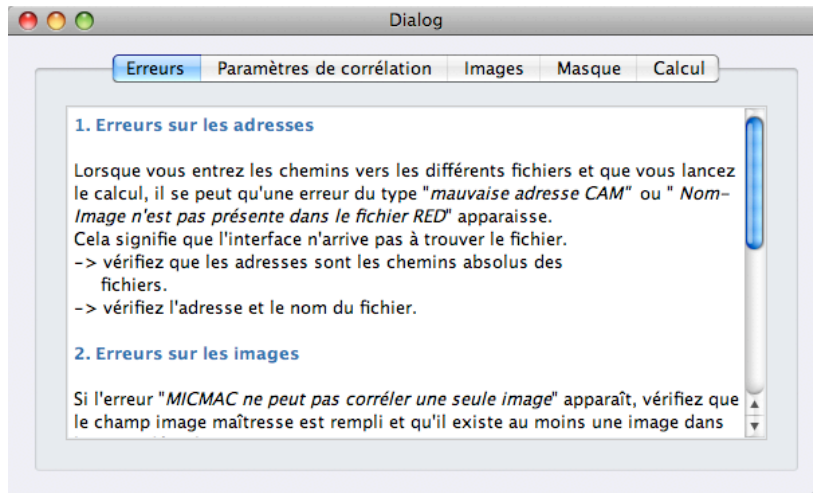


Figure 3: Onglet d'aide

Les erreurs qu'il est possible de rencontrer dans l'interface se trouvent dans l'onglet erreur de l'aide. Elles sont accompagnées des raisons de cette erreur et des solutions. On trouvera en premier lieu, les erreurs sur les adresses puis les erreurs sur les images et l'erreur « *Incohérence entre les fichiers RED et APX* ».


L'onglet « *Paramètres de corrélation* » décrit la signification et les valeurs possibles des paramètres modifiables.

La partie « *Images* » décrit le format et la compression des images à utiliser ainsi que la manière d'ajouter des images.

La partie « *Masque* » présente les deux formes de masque, manuel et automatique : la manière de les créer, de les utiliser et leurs avantages respectifs.

Enfin, la partie « *Calcul* » présente le lancement du calcul, les fichiers créés ainsi que les erreurs qu'il est possible de rencontrer avec Micmac.

### 4. Remplir les adresses

Les adresses des fichiers demandés sont les chemins absolus de ces fichiers, il est possible de les rentrer manuellement dans la ligne d'édition ou bien de parcourir les fichiers de son ordinateur à l'aide du bouton .

Sous Windows, l'adresse du dossier contenant tous les fichiers ne doit comporter ni d'espace, ni de « - ».

La fenêtre suivante s'affiche alors, il suffit de sélectionner le fichier souhaité et de cliquer sur Ouvrir.

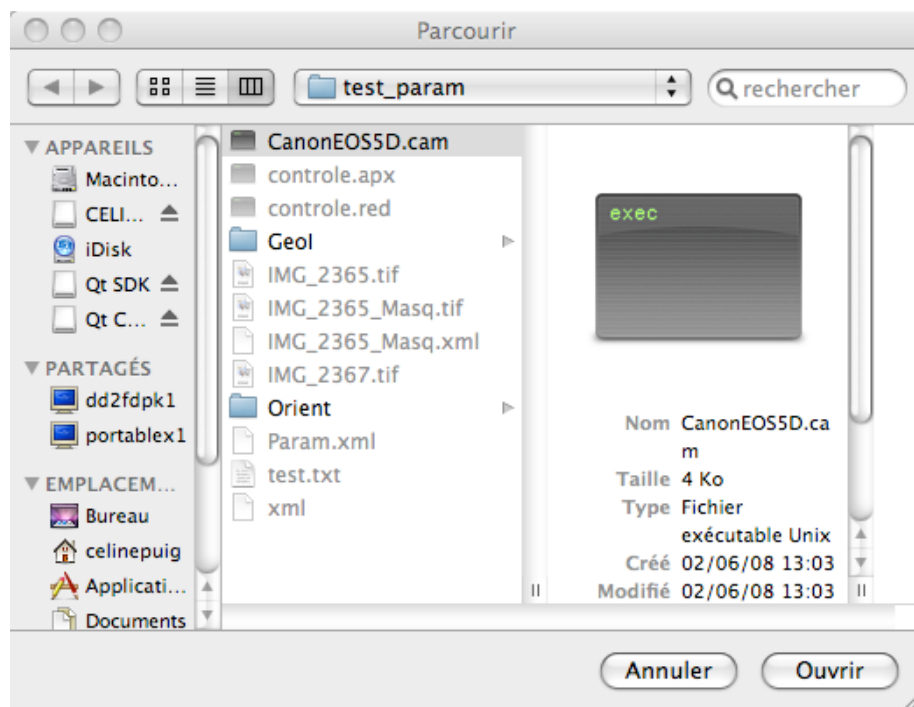


Figure 4: fenêtre Parcourir

Pour entrer l'adresse du dossier, il suffit de sélectionner l'un des fichiers contenu par ce dossier.

Selon le fichier à sélectionner, seules certaines extensions sont acceptées. Par exemple, pour le fichier caméra, seuls les fichiers .cam peuvent être sélectionnés.

## 5. Les paramètres de corrélation

Les paramètres de corrélation possèdent des valeurs par défaut dans l'interface. Ces valeurs sont des valeurs moyennes qui permettent d'avoir des résultats corrects pour la plupart des chantiers de photogrammétrie terrestre. Les valeurs par défaut qui ont été mises tentent de respecter du mieux possible ce compromis précision/rapidité tout étant généralisables pour tous les chantiers.

Il est possible de changer ces paramètres pour affiner la corrélation ou réduire le temps de calcul. Une description et l'étude de l'influence de ces paramètres se trouvent dans la partie «Influence des paramètres de corrélation ».

## 6. Ajouter les images



Micmac demande d'avoir une image maîtresse, c'est sur cette image que se fait le masque qu'il soit manuel ou automatique. On choisit cette image à part des autres images, à partir du champ entouré en vert dans l'illustration ci-dessous.

Les images doivent être exclusivement au format Tiff (.tif), en niveaux de gris et sans compression. Si l'image est ajoutée par le bouton Parcourir, il n'aura pas de problème de format car seuls les .tif peuvent être sélectionnés. Cependant, un mauvais format ou une mauvaise compression entraînent des erreurs de Micmac (voir Erreurs).

The screenshot shows the 'MainWindow' of Micmac. The 'Images' section is highlighted. It includes a green circle around the 'Nom de l'image maîtresse' field and a yellow circle around the 'Nom des autres images' field. The 'Liste des images' dropdown is empty. The 'Lancer le calcul' button is at the bottom.

Figure 5: Ajout d'images

Pour ajouter les autres images, il faut réaliser les étapes suivantes :

- entrer le nom de la première image dans le champs « *Autres images* » (entouré en jaune sur l'illustration)
- cliquer sur le bouton  (Le nom de l'image apparaît dans le menu déroulant « *Liste des images* »)
- Si l'utilisateur souhaite enlever l'une des images de la liste des images, il faut la sélectionner dans le menu déroulant, puis cliquer sur le bouton. 

## 7. Utilisation du masque

Le masque permet de définir la zone à corrélér. Il s'agit d'une image binaire au format Tiff et sans compression. Le masque se cale sur l'image maîtresse.

Il est possible de faire son propre masque en cochant le bouton radio « oui », ou d'utiliser un masque automatique en cochant le bouton radio « non ».

### a. Le masque manuel

Dans ce cas-là, c'est l'utilisateur qui crée son propre masque.

Elle doit être en .tif et sans aucune compression, vérifier en particulier la compression LZW (voir *Erreur*).

Dans le masque, la partie en noir correspond à la zone à ne pas corrélér, tandis que la partie en blanc correspond, au contraire, à la zone à corrélér.

Le masque se cale sur l'image maîtresse, il doit donc faire la même taille.

Pour cela, à l'aide d'un logiciel de traitement d'image basique tel que Paint ou Gimp, l'utilisateur doit réaliser des polygones noir et blanc sur l'image maîtresse. Puis l'enregistrer sous la forme: *nom-image\_Masq.tif*.

Ensuite, dans l'interface, cocher « oui », puis remplir le champs « *Nom du masque* » (manuellement ou à l'aide du bouton Parcourir).

### b. Le masque automatique

Dans le cas présent, c'est Micmac qui réalise cette image binaire.

Le masque que Micmac réalise correspond à la zone de l'image maîtresse dont chaque pixel est au moins sur une autre image sélectionnée dans l'interface. Ainsi, si l'utilisateur possède une image générale de son chantier, il peut récupérer une corrélation en un seul bloc de tout son chantier en passant cette image générale en image maîtresse et avec un masque automatique.

Le choix du masque automatique permet donc de corrélér toute la zone possible d'un chantier de photogrammétrie terrestre.

## 8. Lancement du calcul

Une fois que l'utilisateur clique sur "*Lancer le calcul*", l'interface vérifie que tout ce qui a été entré est cohérent. Sinon, une erreur s'affiche (voir *Erreur*).

L'interface lance alors MICMAC dans une console. Cette console s'affiche pour que l'utilisateur puisse voir le déroulement du calcul (NB : cette console s'affiche seulement sous Windows).

Pendant le calcul, l'interface est en retrait et ne réponds pas. Dès que le calcul est terminé, elle redevient active.

Tout au long du calcul des fichiers sont créés dans le dossier qui a été donné dans l'interface dans le champs "*Adresse du dossier*".

Pour vérifier l'avancement du calcul, il faut ouvrir le dossier GeoI créé et regarder les fichiers *Correl-Geom(...).tif*. Il existe au total 6 fichiers de ce type qui se remplissent au fur et à mesure du calcul. A chaque zoom ultérieur, la corrélation est plus précise.

Les fichiers qui sont créés par l'interface sont :

- un fichier *Param.xml*: fichier des paramètres pour MICMAC.
- un dossier *Orient* comprenant un fichier *OrInit* pour chaque image qui correspond au fichier d'orientation.
- un fichier de masque .xml si l'utilisateur a coché « oui ».
- un fichier test.txt, vide, qui sert simplement de vérification à l'interface.
- un dossier *GeoI* comprenant tous les résultats donnés par MICMAC.

Pour avoir une idée de la carte de profondeur créée, il faut ouvrir les fichiers *Correl-Geom(...).tif*.

! Certaines configurations de Windows ne permettent pas de lire les fichiers .tif. . Pour visualiser les résultats, télécharger un éditeur d'images gratuit tel que GIMP.

## 9. Les erreurs

Il existe deux types d'erreurs :celles que renvoient l'interface, et celles que renvoient Micmac.

### a. Les erreurs de l'interface

- Les erreurs sur les adresses

Lorsque l'utilisateur entre les chemins vers les différents fichiers et qu'il lance le calcul, il se peut qu'une erreur du type "*mauvaise adresse CAM*" ou "*Nom-Image n'est pas présente dans le fichier RED*" apparaisse.

Cela signifie que l'interface n'arrive pas à trouver le fichier. Il faut alors vérifier que les adresses sont les chemins absolus des fichiers et/ou vérifier l'adresse et le nom du fichier.

- Les Erreurs sur les images

Si l'erreur "*MICMAC ne peut pas corrélér une seule image*" apparaît, il faut vérifier que le champ image maîtresse est rempli et qu'il existe au moins une image dans le menu déroulant.


Sinon, il faut en rajouter au moins une à l'aide du bouton  (voir Ajouter une

image).

- Incohérence entre RED et APX

Si cette erreur apparaît, cela signifie qu'un point pointé dans une image et figurant dans le fichier RED n'est pas présent dans le fichier APX.

Il faut vérifier que les fichiers RED et APX utilisés sont les bons.

Sinon, c'est un problème dans l'aérottriangulation avec Redresseur.

## b. Les erreurs de Micmac

*Attention:*

MICMAC demande d'avoir les images dans un certain format.

Un mauvais format peut entraîner des erreurs lors du calcul.

Voir 2.6 Images.

- Erreur de masque

Cette erreur correspond à un mauvais format de masque, il faut décompresser le masque. Cette erreur apparaît aussi pour les images.

```
-----  
Standard File 2d can only handle forward flux  
occured at line 380 of file src/file_image/fich_2d_gen.cpp  
please sent a bug report  
*****  
**      (YOUR)  LOCATION :  
**  
** Error probably occured in a call to ELISE_COPY  
**      at line : 505  
**      of file : applis/MICMAC/cAppliMICMAC_Result1.cpp  
*****  
-----
```

- Erreur de réécriture

Dans certains cas, si le dossier GeoI existe déjà et contient des fichiers, cela peut gêner MICMAC qui renverra alors une erreur sur l'un des fichiers déjà présent, il faut supprimer les fichiers déjà créés, puis retourner dans l'interface, enlever l'une des images (sauf l'image maîtresse) puis la rajouter.

Enfin, relancer MICMAC.

```
Documents/micmacdos/applis/monastier_MPD/GeoI/Masq_Geom_IMG_2365_DeZoom1.tif  
-----  
| the following FATAL ERROR happened (sorry)  
|  
| Photo-interpretation type non connu pour un masque  
|  
-----
```

```
|      (Elise's) LOCATION :  
|  
| Error was detected  
|       at line : 496  
|       of file : applis/MICMAC/cAppliMICMAC_Result1.cpp  
|-----
```

- Erreur sur le dossier Orient

Si MICMAC renvoie l'erreur suivante ou s'il n'est même pas lancé, vérifiez dans le dossier Orient: aucun fichier OrInit ne devrait être créé...

```
For required file /Users/celinepuig/Desktop/test_param/Orient/OrInit-IMG_2365.xml  
-----  
| the following FATAL ERROR happened (sorry)  
|  
| Cannot open  
|  
|-----  
| (Elise's) LOCATION :  
|  
| Error was detected  
| at line : 463  
| of file : src/util/stringifie.cpp  
|-----
```

Si le système d'exploitation utilisé est Windows, il peut s'agir d'un nom de dossier non conforme. En effet, il ne doit comporter ni d'espace ni de ' - '. Relancer Micmac depuis l'interface.

### 3. Influence des paramètres de corrélation

Dans cette partie nous expliquerons la signification des différents paramètres de corrélation et nous présenterons l'effet d'une variation de chaque paramètre sur la carte de profondeur obtenue en sortie de MICMAC.

#### *1. Taille de la fenêtre de corrélation*

Pour gérer la taille de la fenêtre de corrélation on se sert d'un coefficient nommé SzW dans le fichier de paramètres de MICMAC. Si SzW=n la fenêtre de corrélation aura une taille de  $(2n+1)*(2n+1)$ . Dans l'interface, le champ *Taille de la fenêtre* correspond à SzW.

La taille de la fenêtre de corrélation a un effet sur la netteté de la carte de profondeur obtenue. En effet, plus elle est grande et plus le résultat sera flou. Par contre, une fenêtre de corrélation trop petite risque d'entraîner la présence de bruit.

#### *2. Poids de la régularisation*

Ce paramètre correspond au ZRegul du fichier de paramètres de MICMAC. Il quantifie le compromis entre la régularité de la carte de profondeur souhaitée et l'attache aux données. Plus il est grand, plus on impose un résultat régulier et une faible attache aux données. Une trop grande valeur donnera une carte lissée et débarrassée des reliefs importants alors qu'une petite valeur peut donner un certain « effet de pépite ».

#### *3. Pas de quantification*

Ce paramètre correspond au Zpas du fichier de paramètres de MICMAC. Il est à peu près proportionnel à la résolution de l'image finale en Z. Plus il est petit plus la précision est grande. Cependant une valeur trop petite apportera du bruit et rendra le calcul extrêmement long.

#### *4. Contour de recherche (altimétrique et planimétrique)*

Ces paramètres correspondent, en altimétrie ou en planimétrie, à la zone de recherche autour du résultat issu de la corrélation au niveau de zoom inférieur (ZdilatAlti et ZdilatPlani dans le fichier de paramètres de MICMAC). Un contour de recherche trop petit ne sera pas adapté s'il y a de fortes variations (par exemple, il



ne faut pas un contour de recherche altimétrique trop petit si la scène présente une forte amplitude en Z).

### *5. Intervalle de profondeur*

Ce paramètre correspond au ZincCalc du fichier de paramètres de MICMAC. Il correspond à un intervalle d'altitude autour du Z moyen du chantier autour duquel se situe la zone à corrélérer. Un ZincCalc trop petit peut entraîner des problèmes lors de la corrélation et une grande valeur ne pose pas de problème mais ralentit le calcul.

# DOCUMENTATION PROGRAMMEUR

---

## *SOMMAIRE*

<b>1. FONCTIONNALITES DU LOGICIEL</b>	<b>19</b>
<b>2. LANGAGES DE PROGRAMMATION UTILISES</b>	<b>19</b>
<b>3. ARCHITECTURE DU LOGICIEL</b>	<b>20</b>
1. Diagramme UML	20
2. Principes de fonctionnement	20
<b>4. DESCRIPTION DETAILLEE DES DIFFERENTES CLASSES ET FONCTIONS</b>	<b>21</b>
1. La classe jeu_ori	21
a. Attributs	21
b. Méthodes	22
2. La classe jeu_xml	22
a. Attributs	22
b. Méthodes	23
3. La classe jeu_masque	23
a. Attribut	23
b. Méthodes	23
4. La fonction ecriture_orient	24
5. La fonction ecriture_param	24
6. La fonction ecriture_masque	24
7. La classe MainWindow	24

# 1. Fonctionnalités du logiciel

L'interface qui est présentée ici permet de lancer le logiciel MICMAC sans passer par une invite de commande et crée des fichiers de paramètres nécessaires au lancement de MICMAC à partir d'informations issues de l'utilisateur (qui remplit des champs de l'interface) et des fichiers d'orientation des images issus du logiciel Redresseur.

Elle réalise les actions suivantes :

- récupération de données externes apportées par l'utilisateur
- récupération de données dans des fichiers
- création de fichiers .xml contenant des informations nécessaires au fonctionnement de MICMAC
- lancement de MICMAC depuis l'interface.

# 2. Langages de programmation utilisés

Pour réaliser ce logiciel deux langages ont été utilisés : C++ et la bibliothèque Qt.

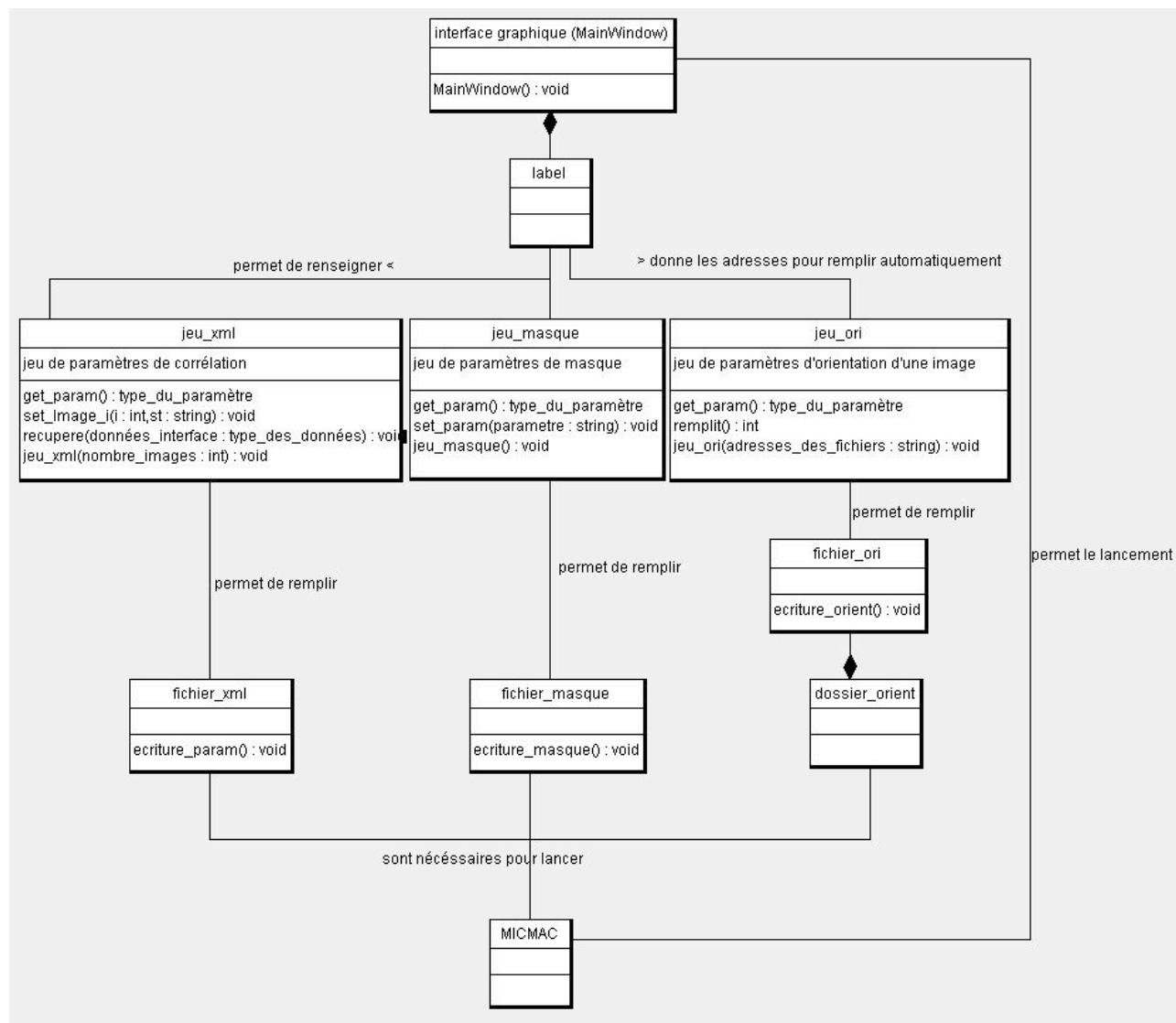
Le langage C++ a été utilisé pour réaliser les calculs ainsi que la récupération de données dans les fichiers issus de Redresseur. Il a été utilisé comme langage orienté objet avec la création de classes. Les déclarations ont été placées dans des fichiers .h alors que les implémentations des différentes méthodes de chaque classe se trouvent dans des fichiers .cpp.

Les fichiers C++ ont été réalisés sous CodeBlocks puis injectés dans l'interface graphique.

La bibliothèque Qt a été utilisée pour la création et la gestion de l'interface graphique avec la création de fichiers .h et .cpp pour les déclarations et l'implémentation des méthodes des classes de formulaires et de fichier .ui pour gérer le côté graphique des formulaires. L'interface graphique (gestion des formulaires) a été réalisée à l'aide de QtCreator.

### 3. Architecture du logiciel

#### 1. Diagramme UML



#### 2. Principes de fonctionnement

Pour fonctionner, MICMAC a besoin de trois types de paramètres :

- Paramètres de corrélation : ils sont directement issus de l'interface graphique (c'est l'utilisateur qui les saisi)
- Paramètres de masque : ils sont directement issus de l'interface graphique (c'est l'utilisateur qui les saisi)
- Paramètres d'orientation des images : ils sont récupérés par l'interface dans les fichiers d'orientation des images issus de Redresseur.

Les paramètres sont donc soit récupérés dans des fichiers, soit issus directement de l'interface. Une fois récupérés, ces paramètres vont être utilisés pour instancier des attributs d'un élément d'une classe de paramètres.

Ensuite une fois que les attributs d'une classe ont été remplis, on se sert de ces derniers pour écrire les fichiers de paramètres en utilisant des getters qui renvoient les valeurs des différents attributs.

Enfin, une fois que les différents fichiers de paramètres ont été écrits, on peut lancer MICMAC à partir de l'interface.

## 4. Description détaillée des différentes classes et fonctions

### 1. La classe *jeu\_ori*

Cette classe gère les paramètres d'orientation d'une image. Ses attributs sont un jeu de paramètres d'orientation et ses méthodes permettent de remplir le jeu de paramètres d'orientation et de le renvoyer.

#### a. Attributs

- *string* nom\_image : nom de l'image sur laquelle porte le jeu de paramètre.
- *float* PPx : coordonnée x du point principal d'auto-collimation de la caméra ayant été utilisée pour la prise de vue.
- *float* PPy : coordonnée y du point principal d'auto-collimation de la caméra ayant été utilisée pour la prise de vue.
- *float* F : focale de la caméra ayant été utilisée pour la prise de vue.
- *int* SzImx : taille en x de l'image (en pixels).
- *int* SzImy : taille en y de l'image (en pixels).
- *float* CDistx : coordonnée x du point principal de symétrie de la caméra ayant été utilisée pour la prise de vue.
- *float* CDisty : coordonnée y du point principal de symétrie de la caméra ayant été utilisée pour la prise de vue.
- *float* CoeffDist3 : coefficient de distorsion d'ordre 3 de la caméra.
- *float* CoeffDist5 : coefficient de distorsion d'ordre 5 de la caméra.
- *float* CoeffDist7 : coefficient de distorsion d'ordre 7 de la caméra.
- *float* AltSol : altitude moyenne de la scène.
- *float* Profondeur : profondeur moyenne de la scène.
- *float* Centrex : coordonnée terrain X du sommet de prise de vue.
- *float* Centrey : coordonnée terrain Y du sommet de prise de vue.
- *float* Centrez : coordonnée terrain Z du sommet de prise de vue.
- *float* Lij : coefficient ligne i colonne j de la matrice d'orientation de la caméra en convention micmac sous la forme L(numéro ligne)(numéro colonne).

Les attributs suivants sont relatifs à des « points de vérification ». Ces points servent à MICMAC pour vérifier si les paramètres d'orientation d'une image sont donnés dans la bonne convention.

- *int* Im1x : coordonnée image x du point de vérification 1.
- *int* Im1y : coordonnée image y du point de vérification 1.

- *float* Ter1x : coordonnée terrain X du point de vérification 1.
- *float* Ter1y : coordonnée terrain Y du point de vérification 1.
- *float* Ter1z : coordonnée terrain Z du point de vérification 1.
- *int* Im2x : coordonnée image x du point de vérification 2.
- *int* Im2y : coordonnée image y du point de vérification 2.
- *float* Ter2x : coordonnée terrain X du point de vérification 2.
- *float* Ter2y : coordonnée terrain Y du point de vérification 2.
- *float* Ter2z : coordonnée terrain Z du point de vérification 2.
- *string* AdresseAPX : adresse du fichier .apx issu de redresseur où l'on va extraire des informations permettant de remplir le jeu de paramètres d'orientation.
- *string* AdresseRED : adresse du fichier .red issu de redresseur où l'on va extraire des informations permettant de remplir le jeu de paramètres d'orientation.
- *string* AdresseCAM : adresse du fichier de caméra où l'on va extraire des informations.

## **b. Méthodes**

- *jeu\_ori(string nom\_image\_in, string AdresseAPX\_in, string AdresseRED\_in, string AdresseCAM\_in)* : constructeur de la classe. Il prend en paramètres le nom de l'image à laquelle on se rapporte ainsi que les adresses des fichiers où se trouvent les informations nécessaires à l'affectation des paramètres. Ce constructeur affecte les valeurs entrées en paramètres aux attributs concernés et donne une valeur par défaut (0) aux autres attributs .
- *~jeu\_ori()* : destructeur de la classe.
- *int remplit()* : méthode permettant d'attribuer une valeur à chaque attribut en allant chercher des informations dans les fichiers .apx, .red et .cam. Elle renvoie un entier pour préciser comment s'est déroulé la récupération d'informations (cf commentaires dans le code).
- *type get\_param\_nomattribut()* : renvoie la valeur de l'attribut concerné (il y a un getter par attribut et donc par paramètre). Le type de retour est celui de l'attribut.

## **2. La classe *jeu\_xml***

Cette classe gère les paramètres de corrélation qui seront utilisés pour lancer MICMAC. Ses attributs sont un jeu de paramètres de corrélation et ses méthodes permettent de remplir le jeu de paramètres de corrélation et de le renvoyer.

### **a. Attributs**

- *float* ZincCalc : intervalle de recherche en Z autour du Zmoyen de la scène.
- *string* MT\_Image : adresse de l'image masque.
- *string* MT\_Xml : adresse du fichier de masque.

La liste des images est gérée par un tableau dynamique nommé tab.

- *string* \*tab : tableau dynamique contenant le nom des images à corrélér
- *const int* nbElement : nombre d'images à corrélér.
- *int* SzW : taille de la fenêtre de corrélation.
- *float* ZRegul : poids de régulation.
- *float* ZPas : pas de quantification.
- *int* ZDilatAlti : contour de recherche en Z autour de la solution issue de l'échelle précédente.
- *int* ZDilatPlani : idem mais en planimétrie.

## **b. Méthodes**

- *jeu\_xml(int nb)* : constructeur de la classe. Il nécessite en paramètre le nombre d'images à corrélér (pour dimensionner le tableau dynamique).
- *~jeu\_xml()* : destructeur de la classe.
- *type get\_param\_nomattribut()* : renvoie la valeur de l'attribut concerné (il y a un getter par attribut et donc par paramètre). Le type de retour est celui de l'attribut. Attention : il y a un getter particulier pour les images dont la structure est la suivante:
- *string get\_Image\_i(int i)* : renvoie le nom de l'image contenue dans la case i du tableau dynamique.
- *void recupere(float in\_ZIncCalc,string in\_MT\_Image,string in\_MT\_Xml,int in\_SzW,float in\_ZRegul,float in\_ZPas,int in\_ZDilatAlti, int in\_ZDilatPlani)* : méthode permettant d'attribuer une valeur à chaque attribut en utilisant les valeurs passées en paramètre.
- *void set\_Image\_i(int i,string st)* : méthode permettant d'affecter la valeur st dans la case i du tableau de gestion des images à corrélér.

## **3. La classe *jeu\_masque***

Cette classe gère les paramètres de masque.

### **a. Attribut**

*string* param : adresse de l'image de masque.

### **b. Méthodes**

- *jeu\_masque(string parametres)* : constructeur de la classe
- *~jeu\_masque()* : destructeur de la classe
- *void setParam(string parametres)* : affecte à l'attribut de la classe la valeur passée en paramètre.
- *string getparam()* : renvoie la valeur de l'attribut de la classe.

#### *4. La fonction `ecriture_orient`*

*int* `ecriture_orient`(*QString* dossier, *jeu\_ori* jeuO) : cette fonction crée un fichier d'orientation dans le dossier entré en paramètre et écrit dans ce fichier à l'aide des informations sur l'orientation de l'image contenues dans le jeu de paramètres d'orientation jeu\_0 de type jeu\_ori entré en paramètre.

#### *5. La fonction `ecriture_param`*

*int* `ecriture_param`(*QString* dossier, *jeu\_xml* jeu1) : cette fonction crée un fichier de paramètres de corrélation dans le dossier entré en paramètre et écrit dans ce fichier à l'aide des informations contenues dans le jeu de paramètres de corrélation jeu\_1 de type jeu\_xml entré en paramètre.

#### *6. La fonction `ecriture_masque`*

*int* `ecriture_masque`(*QString* dossier, *jeu\_masque* jeu2, *jeu\_ori* jeu3) : cette fonction crée un fichier de paramètre de masque dans le dossier passé en paramètres.

#### *7. La classe `MainWindow`*

Cette classe gère la fenêtre de l'interface graphique. Elle instancie des objets des classes définies précédemment et synchronise l'appel des méthodes des différentes classes pour créer les fichiers nécessaires au lancement de MICMAC et à lancer ce dernier logiciel.