Documentation Utilisateur de Grounded

Guide d’installation :

Installation des dépendances python

```
>> pip install -r requirements.txt
```

Spécificité metashape

- Se rendre sur la page de téléchargement de Agisoft Metashape
- Télécharger les fichiers le module python3 correspondant à votre distribution
- Effectuer la commande suivante sur le fichier qui vient d'être téléchargé : « pip install Metashape-.whl` »

Dépendances logicielles

Linux

MicMac

- Installation des dépendances :

```
>> sudo apt-get update && sudo apt-get install make imagemagick libimage-exiftool-perl exiv2 proj-bin
```

```
>> sudo apt update && sudo apt install qtbase5-dev qt5-qmake
```

- Copie du dépôt github :

```
>> git clone https://github.com/micmacIGN/micmac.git
```

- Se déplacer à l’intérieur du projet :

```
>> cd micmac
```

- Créer un dossier de build :

```
>> mkdir build && cd build
```

- Générer les fichiers makefiles en utilisant cmake :

```
>> cmake ../
```

- Effectuer la compilation en remplaçant {cores number} par le nombre de cœurs du processeur de la machine :

```
>> make install -j{cores number}
```

Recommandé:

- Sortir du projet MicMac

```
>> cd ../..
```

- Déplacer MicMac dans le dossier /opt :

```
>> sudo mv micmac/ /opt/
```

CCTag

- Installation des dépendances :

```
>> sudo apt-get update && sudo apt-get install g++ git-all libpng-dev libjpeg-dev libeigen3-dev libboost-all-dev libtbb-dev
```

```
>> sudo apt update && sudo apt install libopencv-dev libboost-all-dev
```

- Copie du dépôt github :

```
>> git clone https://github.com/alicevision/CCTag.git
```

- Se déplacer à l'intérieur du projet :

```
>> cd CCTag
```

- Créer un dossier de build :

```
>> mkdir build && cd build
```

- Générer les fichiers makefiles en utilisant cmake :
  - Si vous possédez une carte graphique Nvidia :

    ```
    >> cmake ../
    ```
  - Sinon :

    ```
    >> cmake -DCCTAG_WITH_CUDA:BOOL=OFF ../
    ```

- Effectuer la compilation en remplaçant {cores number} par le nombre de cœurs du processeur de la machine :

```
>> make install -j{cores number}
```

<u>Recommandé :</u>

- Renommer le fichier nouvellement généré

```
>> mv Linux-* CCTag
```

- Déplacer le fichier CCTag dans le dossier /opt

```
>> sudo mv CCTag/ /opt/
```

CloudCompare

```
>> sudo snap install cloudcompare
```

Comment utiliser l’application en ligne de commande ?

Configuration :

Avant toute utilisation de l’application grounded, mettre à jour le fichier de configuration selon vos préférences et les spécificités de votre système.

Parmi les informations à mettre à jour se trouve :

- Les chemins menant aux exécutables
- Les préférences de l’utilisateur concernant les modules utilisés par défaut (optionnel)
- La configuration de réglets habituellement utilisée si différente de la configuration par défaut (optionnel)
- Les préférences de l’utilisateur concernant la configuration par défaut des modules (optionnel)

Format des fichiers de reglets:

Les fichiers de réglets sont au format csv. Ces fichiers csv sont composé de 3 colonnes et de plusieurs lignes qui elles représenteront les réglets. La première ligne du fichier csv n’est pas lu lors de l’analyse car elle est considéré comme contenant les titres des différentes colonnes.

Voici le fichier par défaut utilisé par l’application.

Start	End	Length
0	1	0..22
2	3	0.22
4	5	0.22
6	7	0.22

Les deux premières colonnes représentent les identifiant des mire situées aux extrémités des réglets tandis que la troisième correspond à la distance en mètres séparant les mires

Attention le séparateur du fichier csv doit être une virgule « , »

Utilisation basique

Après avoir effectué l'installation de Grounded, il est possible de l'utiliser en ligne de commande de de cette façon :

```
>> grounded path/to/photo_before_excavation path/to/photo_after_excavation
```

Conseil : Si vous ajoutez le fichier grounded.py au path il sera plus facile d'accéder au fichier grounded.py lors de vos futurs utilisations.

Utilisation avancée

Lors de l'appel en ligne de commande de l'application grounded il est possible de configurer les paramètres d'exécution. Voici un tableau détaillé montrant les différents arguments et options possibles :

Options possibles :

Argument	**Description**	**Valeurs possibles**
-sfm	Module sfm utilisé lors de l'exécution	« micmac »
		« metashape »
-sfm_arg	Paramètres du module sfm	(voir détail plus bas)
-detector	Module de détection de mire utilisé lors de l'exécution	« detection_cctag »
		« detection_metashape »
-detector_arg	Paramètres du module de détection de mire	(voir détail plus bas)
-cloudprocessor	Module opérateur de nuages de points utilisé lors de l'exécution	« cloudcompare »
-cloudprocessor_arg	Paramètres du module opérateur de nuages de points	(voir détail plus bas)
-h	Affichage de l'aide	
-v	Paramètre de verbosité	« 0 » : ne conserve que les résultats
		« 1 » : conserve les résultats et les fichier intermédiaires
		« 2 » : conserve les résultats, les fichier intermédiaires et enregistre les sorties des programmes tierces
-o	Paramètre de sortie	Un nom de dossier de sortie
-scalebar	Paramètre de réglets personnalisés.	Un fichier csv au format souhaité
-display_padding	Option d'affichage de la zone détection sur le fichier de résultat au format pdf	

Détail de l'option -sfm_arg :

Module SFM	Variable	Description	Valeurs possibles
micmac	path_mm3d	Chemin vers l'exécutable mm3d	
	working_directory	Nom du répertoire de travail	
	output_dir	Nom du fichier de sortie dans lequel sera placé le répertoire de travail	
	distorsion_model	Modèle de distorsion utilisé par micmac lors de la commande Tapas	« RadialExtended »
			« RadialBasic »
			« Fraser »
			« FraserBasic »
			« FishEyeEqui »
			« HemiEqui »
			« AutoCal »
			« Figee »
	zoom_final	Résolution du nuage de points généré par micmac lors de la commande C3DC	« QuickMac »
			« MicMac »
			« BigMac »
	tapioca_mode	Mode de détection des points homologues lors de la commande Tapioca	« All »
			« MulScale »
	tapioca_resolution	Résolution de la commande Tapioca permettant la recherche de points homologues	
	tapioca_second_resolution	Seconde résolution de la commande tapioca permettant la recherche de points homologues. Cette dernière n'est utilisé que lorsque tapioca_mode est MulScale	
metashape	working_directory	Nom du répertoire de travail	
	output_dir	Nom du fichier de sortie dans lequel sera placé le répertoire de travail	
	downscale	Résolution du nuage de points généré par metashape. Plus celui-ci est bas, plus la résolution sera élevée.	Un nombre entier.

Détail de l’option -detector_arg :

Détecteur de mire	Variable	Description	Valeurs possibles
detection_cctag	path_cctag_directory	Chemin vers le dossier contenant les exécutables de CCTag	
	working_directory	Nom du répertoire de travail	
	output_dir	Nom du fichier de sortie dans lequel sera placé le répertoire de travail	
detection_metashape	working_directory	Nom du répertoire de travail	
	output_dir	Nom du fichier de sortie dans lequel sera placé le répertoire de travail	

Détail de l’option -cloudprocessor_arg :

Opérateur de nuages de points	Variable	Description	Valeurs possibles
cloudcompare	path_cloud_compare	Chemin vers l’exécutable CloudCompare	
	working_directory	Nom du répertoire de travail	
	output_dir	Nom du fichier de sortie dans lequel sera placé le répertoire de travail	
	version	Version installée de CloudCompare	Exemple : « 2.11.1 »

Exemples d’utilisation de la ligne de commande pour comprendre comment la mettre en place:

```
>> grounded -sfm micmac -cloudprocessor cloudcompare -cloudprocessor_arg version=2.12.3 path/to/directory_before_excavation path/to/directory_after_excavation
```

```
>> grounded path/to/directory_before_excavation path/to/directory_after_excavation  -v 0 -o -sfm metashape -display_padding
```

```
>> grounded  path/to/directory_before_excavation path/to/directory_after_excavation  -v 2 -sfm_arg tapioca_mode=All -detector detection_metashape -sfm_arg tapioca_mode=MulScale -sfm_arg tapioca_second_resolution=300
```