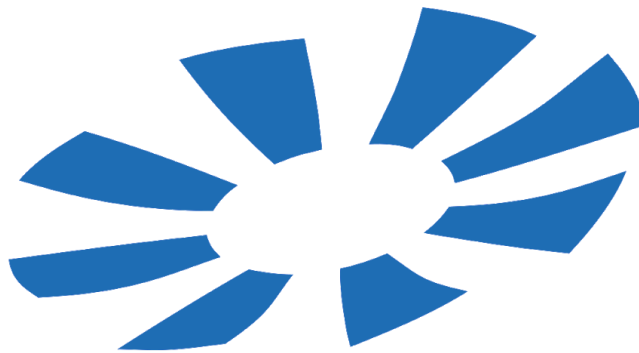


Projet LIDAR EXAMETRICS



EXAMETRICS

Remerciements	3
Introduction	4
I.1) Parties prenantes	4
Présentation de l'équipe IMERIR - CHIPS Université de Perpignan	4
Equipe pédagogique	4
Côtés client - Exametrics	4
I.2) Technologie impliquée	5
Hardware	5
Softwares utilisés	6
Méthode de travail	7
Planning	8
Mise en place des pré requis	9
Lexique/Vocabulaire	9
Mise en place de l'environnement de développement	10
Système UNIX - Kubuntu 17	10
Système Windows/ OSX MAC	11
Lien utiles	11
Documentation fichier LAS	12
Informations générales	12
Pour finir les données, elles sont formaté selon le format las utilisé. Exametrics partaient sur l'utilisation de la version 1.0, elle n'a pas d'information supplémentaire, puis finalement ils utilisent la version 1.4 pour l'ajout d'information sur chaque point (GPS, Température).	12
En python	13
Enrichissement Fichier LAS	14
Développement du plugin	15
Affichage IHM	15
Présentation de l'IHM	15
Notion mathématique liées à l'affichage	15
Création de plan	15
Notion mathématique liées à la détection des points	15
Optimisation du traitement	15
Définition	15
Au sein du plugin	16
Debug	16
Retour d'expérience	17
Conclusion	18
Bibliographie	19

1. Remerciements

Remerciements à :

Marcel BARIOU et Henry BORREIL pour leurs aides et soutiens concernant la gestion du projet et les parties techniques.

Merci également à l'équipe pédagogique encadrant les projets industriels au sein d'IMERIR : Claude LE MENAHEZE, Ahmed RHARMAHOUI.

2. Introduction

Ce projet est proposé par l'entreprise Exametrics, spécialisée dans la cartographie. Elle se caractérise par l'utilisation de drone pour cartographie, la topographie, l'art de la cartographie précise. Notamment la modélisation 3D de château ou encore d'abbaye, également la surveillance de pylône électrique basé sur un contrat avec RTE.

Ce document a comme objectif d'un point de vue pédagogique de rendre un rapport final concernant le projet de deux mois, et professionnel vis à vis de l'entreprise Exametrics.

a) Parties prenantes

Présentation de l'équipe IMERIR - CHIPS Université de Perpignan

- Clément Matysiak
- Jules Malard
- Noé Pichot
- Mahdi Smida

Equipe pédagogique

- Ahmed Rharmouli - Enseignant responsable
- Claude Le Menaheze - Enseignant responsable

Côtés client - Exametrics

- Henry Borreill - Présent Créateur et Fondateur de Exametrics
- Marcel Bariou - Dirigeant mandataire

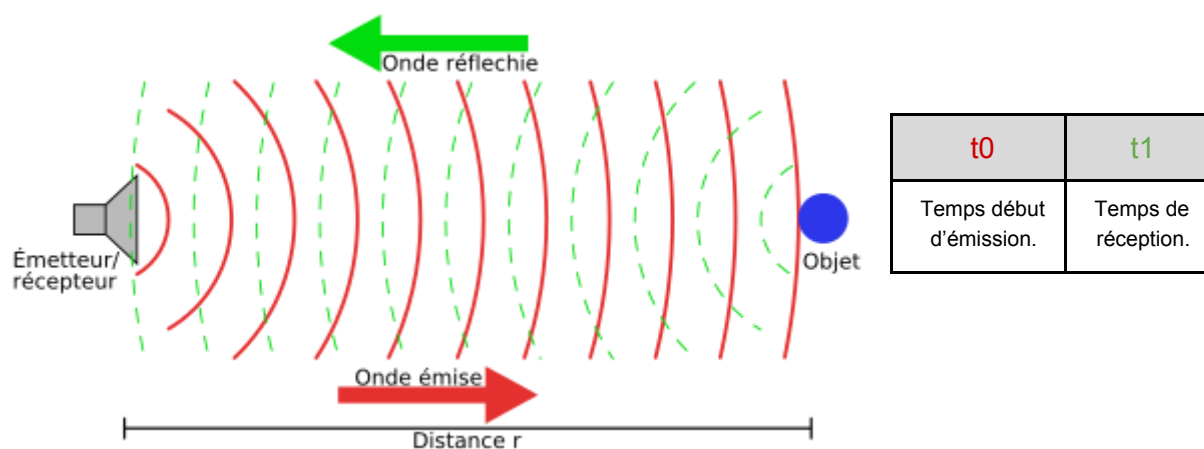
b) Technologie impliquée

Hardware

Présentation technique du LIDAR.

L'acquisition d'un LIDAR (*Laser Detection And Ranging*) mobile, donne naissance à notre projet et la problématique associée.

Un Lidar est un dispositif de télédétection, basé sur les propriétés d'un faisceau de lumière renvoyé vers son émetteur.

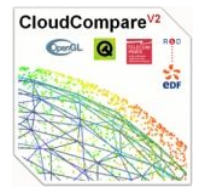
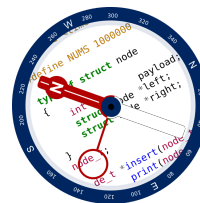

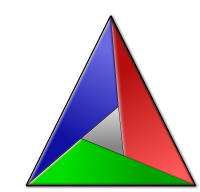



A partir de la différence entre ces deux temps, connaissant la vitesse de l'onde, il est alors possible d'en déduire la distance entre l'émetteur et l'objet.

En effectuant ce type de mesure dans plusieurs orientation, il est alors possible de "scanner" une pièce. Cette méthode a donc comme résultat un nuage de points au format *.las.

C'est sur ce type de fichier que notre projet s'appuie. Le traitement de ce nuage permet l'extraction de forme géométrique plus ou moins complexe selon le domaine d'application.

Softwares utilisés

Logo	Nom	Versio n	Description
	CloudCompare	V.2.9.1	Logiciel Open Source de visualisation/traitement de nuages de points et de maillages 3D. License GNU GPL.
	CodeCompass		Outil de compréhension de logiciel écrit en C/C++ et Java.
	Pdal Library	V.1.5.0	Point Data Abstraction Library : is a C++ library pour la traduction de données de type Nuage de Point. License BSD.
	Cmake, Cmake Gui	V.3.9	Cross platform make - utilisé pour la compilation.
	QT SDK, QT Creator	V.5.10	API orienté Objet développé en C++.

3. Méthode de travail

Concernant le travail au sein de l'équipe d'IMERIR, le planning est fait en début de semaine afin de mettre en place les objectifs, et les différents processus attribué à chaque membre de l'équipe.

Un mail bilan chaque fin de journée afin d'informer Marcel Bariou et Henry Borreill de l'avancée du projet, des éventuels problèmes. Deux briefings hebdomadaire sont mis en place en début de semaine mardi et fin de semaine vendredi avec Mr Marcel Bariou.

a. Planning

Référence	Description	Responsable	Durée Initiale	Temps passé	Reste à faire
1	Rédaction DOP	Mahdi	10	7	3
2	Rendez vous - Contact Client	Jules	10	4	6
3	Visite entreprise	Clément	1	0	1
4	Etat de l'art - LIDAR	Jules	2	2	0
5	Etat de l'art - Fichier *.LAS	Clément	2	2	0
6	Etat de l'art - Création d'un Plugin CloudCompare	Jules	3	3	0
7	Etat de l'art - Ajout Plugin, compilation de CloudCompare avec CMAKE	Mahdi	5	5	0
8	Etat de l'art - Library de CloudCompare	Mahdi	5	5	0
9	Etat de l'art - Normes de Codage	Jules	1	1	0
10	Etat de l'art - Principe Mathématique	Jules	3	3	0
11	Etat de l'art - Arbres d'Octree, parcours de Morton, Voxel	Jules	5	1	4
12	Documentation - LIDAR	Mahdi	1	1	0
13	Documentation - Fichier *.LAS	Jules	1	1	0
14	Documentation - Enrichissement Fichier *.LAS	Jules	2	2	0
15	Documentation - Création d'un Plugin CloudCompare	Clément	2	2	0
16	Documentation - Ajout Plugin, compilation de CloudCompare avec CMAKE	Jules	4	1	3
17	Documentation - Library de CloudCompare	Noé	2	0	2
18	Documentation - Normes de Codage	Clément	2	2	0
19	Documentation - Principe Mathématique	Mahdi	2	2	0
20	Documentation Environnement de travail	Mahdi	2	2	0
21	Documentation - Arbres d'Octree, parcours de Morton, Voxel	Jules	4	0	4
22	Mise en place de l'environnement de travail.	Noé	5	5	0
23	Création de l'IHM du plugin dans CloudCompare (Qt Creator). - (Code)	Jules	3	3	0
24	Débugage de l'IHM du plugin dans CloudCompare (Qt Creator). - (Code)	Jules	3	3	0
25	Enrichissement d'un fichier LAS Température °C.	Clément	1	1	0
26	Création de l'affichage Plan + Vecteur dans CloudCompare. - (Code)	Clément	20	12	8
27	Débugage de l'affichage Plan + Vecteur dans CloudCompare. - (Code)	Noé	5	0	5
28	Création du code Plugin traitement du nuage de points en intersection avec le plan.	Clément	5	0	5
29	Mise en place du Système sur Matériel Exametrics	Noé	5	0	5
30	Mise en place du Pluggin sur Matériel Exametrics	Jules	5	0	5
31	Participation à la rédaction du Cahier Des Charges.	Jules	1	0	1
32	Planification du projet	Noé	3	1	2
33	Rédaction Rapport Final	Noé	10	1	9
34	Débugage	Noé	7	0	7
35	Test du Plugin	Mahdi	5	0	5
36	Préparation à la soutenance.	Mahdi	4	0	4
37	Soutenance.	Noé	1	0	1

4. Mise en place des pré requis

a. Lexique/Vocabulaire

Lidar

La télédétection par laser ou lidar, acronyme de l'expression en langue anglaise « *light detection and ranging* » ou « *laser detection and ranging* » (soit en français « détection et estimation de la distance par la lumière » ou « par laser »), est une technique de mesure à distance fondée sur l'analyse des propriétés d'un faisceau de lumière renvoyé vers son émetteur.

Nuage de points

Ensemble de points données dans un système de coordonnées (généralement X,Y,Z).

.las format

Format standard pour les données LIDAR, regroupe différentes informations, voir [ce lien](#).

.laz format

Format compressé du format “.las”.

Raster

Représentation matriciel de données géographiques: les Raster sont des images (plans scannés, photographies aériennes, images satellitaires) repérées dans l'espace.

Matrice de transformation

Dans le plan cartésien, une matrice de transformation est une matrice qui permet, à partir des coordonnées d'un point initial représentées par une matrice colonne, de trouver celles de son image par une transformation géométrique donnée (translation, rotation). Les coordonnées de l'image sont alors obtenues en effectuant la multiplication de la matrice colonne (les coordonnées d'un point) par la matrice correspondant à la transformation géométrique concernée.

Squelettisation

La squelettisation est une classe d'algorithmes utilisée en analyse de formes. Elle consiste à réduire une forme en un ensemble de courbes, appelées squelette, centrées dans la forme d'origine. La squelettisation est un outil d'analyse de forme non scalaire, qui conserve les propriétés topologiques de la forme d'origine ainsi que les propriétés géométriques, selon la méthode employée.

Octree

Un octree est une structure de données de type arbre dans laquelle chaque nœud peut compter jusqu'à huit enfants. Les octrees sont le plus souvent utilisés pour partitionner un espace tridimensionnel en le subdivisant récursivement en huit octants.

b. Mise en place de l'environnement de développement

Système UNIX - Kubuntu 17

Dans l'invite de commande veuillez entrer les commandes suivantes :

Step	Descriptif	Commande
1	Création d'un répertoire de travail	<ul style="list-style-type: none">• <code>mkdir Exametrics</code>• <code>cd Exametrics</code>
2	Installation des dépendances	<ul style="list-style-type: none">• <code>sudo apt-get install cmake qt-sdk qt5-default cmake-qt-gui libgeotiff-dev libgeos-dev libjsoncpp-dev libxml2-dev</code>
3	Installation de GDAL	<ul style="list-style-type: none">• <code>mkdir gdal</code>• <code>cd gdal</code>• <code>wget download.osgeo.org/gdal/CURRENT/gdal-2.2.3.tar.gz</code>• <code>tar -zxvf gdal-gdal-2.2.3.tar.gz</code>• <code>./configure</code>• <code>make</code>• <code>sudo make install</code>
4	Installation de PDAL	<ul style="list-style-type: none">• <code>git clone https://github.com/PDAL/PDAL.git pdal</code>• <code>cd pdal && mkdir makefiles && cd makefiles</code>• <code>cmake -G "Unix Makefiles" ../</code>• <code>sudo make install</code>
5	Configurer la compilation de CloudCompare	<ul style="list-style-type: none">• <code>mkdir cloudcompare</code>• <code>git clone --recursive https://github.com/cloudcompare/trunk.git</code>• <code>cd trunk/plugins</code>• <code>git clone https://github.com/CleiK/Exametrics.git</code>• <code>cd ../../ && mkdir build</code>• <code>cd build && make all</code>• <code>sudo make install</code>• <code>sudo ldconfig</code>
6	Lancer CloudCompare	<ul style="list-style-type: none">• <code>/usr/local/bin/CloudCompare</code>

Système Windows/ OSX MAC

Concernant la distribution windows, tout les plugins natif sont directement intégrés à CloudCompare.

Il est cependant nécessaire de compiler celui ci en utilisant Cmake-gui afin d'ajouter un nouveau plugin.

Lien Utiles : <http://www.gadom.ski/2017/04/21/pdal-on-windows.html>

Lien utiles

Compilation CloudCompare	https://github.com/CloudCompare/CloudCompare/blob/master/BUILD.md
GitHub Exametrics	https://github.com/CleiK/Exametrics
User manual CloudCompare	http://www.cloudcompare.org/doc/qCC/CloudCompare%20v2.6.1%20-%20User%20manual.pdf
Documentation CloudCompare	http://www.cloudcompare.org/doc/

5. Documentation fichier LAS

a. Informations générales

Un fichier las est constitué en 3 parties : Header, VLR et les données

Le header contient les informations a propos du fichier (voir document format las) tel que le nom, le format, etc...

le VLR et un genre de gestionnaire de version, a chaque modification d'un fichier las ou ajoute une information au VLR.

Pour finir les données, elles sont formaté selon le format las utilisé. Exametrics partaient sur l'utilisation de la version 1.0, elle n'a pas d'information supplémentaire, puis finalement ils utilisent la version 1.4 pour l'ajout d'information sur chaque point (GPS, Température).

b. En python

Certaines fonctions utiles pour la modification et l'utilisation de fichier las en python.
Le package Laspy est utilisé, ainsi que Numpy.

```
pip install laspy  
pip install numpy
```

Lecture et enregistrement et fermeture de fichier :

```
inFile = laspy.file.File("monFichierLas.las", mode= "r")
```

Le mode Write ("w") n'est pas nécessaire si les modifications sont faites sur une copie du fichier. (C'est le cas pour nous).

```
outFile = laspy.file.File("monNouveauFichierLas.las", mode= "w", vlrs = vlrsA,  
header = inFile.header)  
outFile.close()
```

Ici, nous enregistrons un nouveau fichier avec de nouvelles données dans le VLR, et un header non modifié par rapport au fichier original.

Modification header et VLR :

```
outFile.header.offset = ...  
outFile.header.file_source_id = ...
```

Tous les éléments modifiables sont dans la documentation format las.

```
vlrA = copy.copy(inFile.header.vlrs)  
vlrNouveau = laspy.header.VLR("Exametrics,1,"x00*0, description = "las  
modified")  
vlrA.append(vlrNouveau)
```

Deux méthodes de modification, la première directement sur le fichier de sortie, la deuxième en copiant les données du fichier de base et en ajoutant les modifications. (Voir plus haut, enregistrement fichier). (Argument fonction VLR : Voir document format las).

Modification données :

```
len(inFile.points)  
inFile.x[50]  
inFile.gps_time[50]  
inFile.intensity[50]
```

De haut en bas, récupère le nombre de points, récupère la valeur x, le gps time et l'intensité du point 50.

6. Enrichissement Fichier LAS

JULES

7. Développement du plugin

a. Affichage IHM

Présentation de l'IHM

Code couleur	Définition
	Onglet - Permet de configurer le premier et deuxième plan.
	Inputs - Coordonnées 3D du points A (vecteur AB).
	Inputs - Coordonnées 3D du points B (vecteur AB).
	Slide - Positionnement du plan le long du vecteur.
	Input - Réglage de la tolérance du plan.
	CheckBox - Passage du mode 1 plan à 2 plans (Version future).
	List - Sélection Algorithme : Python ou Octree.
	Button - Compute, lance l'affichage et le traitement.
	Button - Fermeture de la fenêtre IHM

La définition des points A et B détermine le vecteur normal au plan désiré. Le slider juste dessous permet de déplacer un point du vecteur entre A et B pour complètement définir notre plan.

Enfin la tolérance est utilisé pour préciser la hauteur de la “boîte” permettant le calcul d’intersection.

Nous pouvons sélectionner 2 algorithmes, le premier est linéaire est exécuté en python pour les premiers tests. Le deuxième utilise les octree et permet une optimisation en temps du calcul. A l’appuis du bouton “Compute” l’algorithme sélectionné est exécuté dans un Thread séparé pour ne pas bloqué l’IHM.

b. Notion mathématique liées à l’affichage

Création du plan

La génération du plan se fait à partir de 4 points définis grâce à:

$$N = \text{VectorPoint} - \text{VectorPointA}$$

La médiatrice M.

Ainsi les points obtiennent les valeurs suivantes:

- 0 -N.z N.y
- -N.z 0 N.x
- N.y -N.x 0
- M.x Mx.y M.z

Les 3 premiers points sont perpendiculaires au vecteur.

Une méthode de la classe ccPlane permet de créer un plan adapté à ces 4 points.

Nous opérons ensuite une translation sur les 3 axes paramétrés avec VectorPoint - PlanCenter pour déplacer la plan sur le point du vecteur.

Création de la boîte

Nous définissons les dimensions de la boîte comme ceci: largeur et hauteur du nuage de point d’entrée et tolérance.

Nous récupérons la transformation qui a été appliqué au plan et on génère la boîte avec ces paramètres.

c. Notion mathématique liées à la détection des points

Afin de réaliser une première intersection de manière linéaire, nous utilisons une méthode trouvé après recherches sur le net ([lien](#)).

Pour cela il nous faut transmettre quelques paramètres au script python:

- Nom du fichier las à traiter
- Coordonnées de 4 points (P1 P2 P4 P5 selon le lien au dessus)

Après une première tentative de récupération des points de la boîte, nous obtenons 24 points dont certains sont confondus. Nous appliquons donc un petit algorithme pour éliminer les points dupliqués et ainsi récupérer que 8 points définissant notre boîte correctement.

Nous transmettons alors le nom du fichier et les 4 points au script python qui nous fourniras un fichier de sortie las comprenant seulement l’intersection du nuage avec notre boîte.

d. Optimisation du traitement

Le traitement de fichier de nuage de point (extension LAS) volumineux peut être très énergivore. Pour minimiser le temps de traitement dans ce cas la, il est nécessaire d'organiser les données afin d'optimiser son traitement.

Définition

Un Octree désigne une structure de donnée de type arbre. Chaque noeud peut compter jusqu'à huit enfants. Notre cas d'utilisation réside dans le partitionnement de l'espace tridimensionnel récursivement en huit octants associés.

Au sein du plugin

e. Debug

Compilation de cloudcompare

La compilation de CloudCompare peut poser certains problèmes liés aux dépendances des bibliothèques. Si la compilation pose problème en utilisant CMAKE, la solution est de lancer CMAKE-GUI, et d'utiliser l'interface graphique afin de fournir manuellement le path de dossiers posant problème.

Segmentation des fichiers LAS

JULES

Installation de l'environnement, library PDAL

Documentation classes CloudCompare

La documentation du code de CloudCompare est minimal, il faut étudier le code en place et les plugins développés par d'autres personnes.

Déformation de la box

non affichage du plan

Octree qui prend pas le nuage de point en considération

Problème de gestion de mémoire (SegFault)

Problème de gestion de la mémoire lors de l’affichage d’objets graphiques et de leur mise à jour au fur et mesure de l'exécution du programme.

8. Retour d'expérience

Au début de ce projet, la tâche semblait difficile étant donné que nous ne présentions aucune connaissance en rapport à la technologie LIDAR, ou même aux nuages de point. En effet aucun des membres de l'équipe n'avait de spécialisation dans le domaine. Cependant nous avons rapidement cerner l'intitulé de la problématique malgré sa complexité avec l'aide de Marcel Bariou.

L'objectif est de tirer des enseignements avant tout profitables aux prochains acteurs de ce projet. Le but est donc de garder en mémoire des événements marquants, d'identifier des pistes de progrès.

Il est primordial dans un premiers temps de prendre connaissance de l'état de l'art des technologies impliquées. Que ce soit pendant la phase de mise en place de l'environnement de développement que pendant le développement lui-même.

Une fois que le contexte du projet est assimilé, l'étape suivante consiste à trouver les plateformes et les ressources utiles (cf bibliographie) à la compréhension et l'utilisation des librairies, méthodes et fonctions qui seront exploitées.

Il est également à noter que les différents acteurs du projet sont invités à travailler sur le même environnement de DEV. Cela afin d'éviter tout problème de compatibilité, de versions et de dépendances. Nous invitons donc aux prochains acteurs de mettre en place un environnement identique stable. Cela concerne l'OS, compilateur, librairies...

La particularité de ce projet réside également dans la cohabitation du domaine mathématique (géométrie dans l'espace) et le développement informatique (CloudCompare C++).

A partir de ce constat et de notre expérience au cours de ces 2 mois, la réflexion commune concernant un problème mathématique à transposer dans le code est de mise. En effet si toute l'équipe réfléchit en même temps sur le problème, cela permet de confronter les idées, de trouver de nouvelles pistes de réflexion, et donc par la suite des solutions.

Les mathématiques sont certes une science exacte, néanmoins les outils sont variés, et offrent une certaine liberté quant aux méthodes utilisées.

Concernant le planning, la phase de test et de debug ne sont pas sous-estimées (temps, ressources), tout comme la documentation associée.

9. Conclusion

Ce projet nous a permis de découvrir un thématique nouvelle.

L'union entre le domaine mathématique tridimensionnel pur et le développement est un exercice de choix, nous a permis de monter en compétences concernant le développement et la prise de recul.

En effet au cours du projet nous avons été amenés à jongler entre notions mathématique et développement informatique. Ce qui nécessite une parfaite compréhension entre les deux domaines technologiques, et une vision générale de la jonction entre ces derniers.

Il est regrettable que nous ayons pas eu le temps de faire une acquisition de points munis du LIDAR mobile afin de tester expérimentalement notre solution.

10. Bibliographie

- [1] <https://www.brasnah.fr/>
- [2] <https://www.brasnah.fr/index.php/documents/projets-de-recherche-memoire-stage>
- [3] https://www.pdal.io/tutorial/liblas_to_pdal.html
- [4] <https://www.liblas.org/doxygen/index.html>
- [5] <https://sites.google.com/site/lidaretforet/ressources/synoptique-des-outils>
- [6]
- [7]
- [8]
- [9]
- [10]
- [11]
- [12]
- [13]
- [14]
- [15]
- [16]
- [17]
- [18]
- [19]
- [20]
- [21]
- [22]
- [23]
- [24]
- [25]
- [26]
- [27]
- [28]
- [29]
- [30]