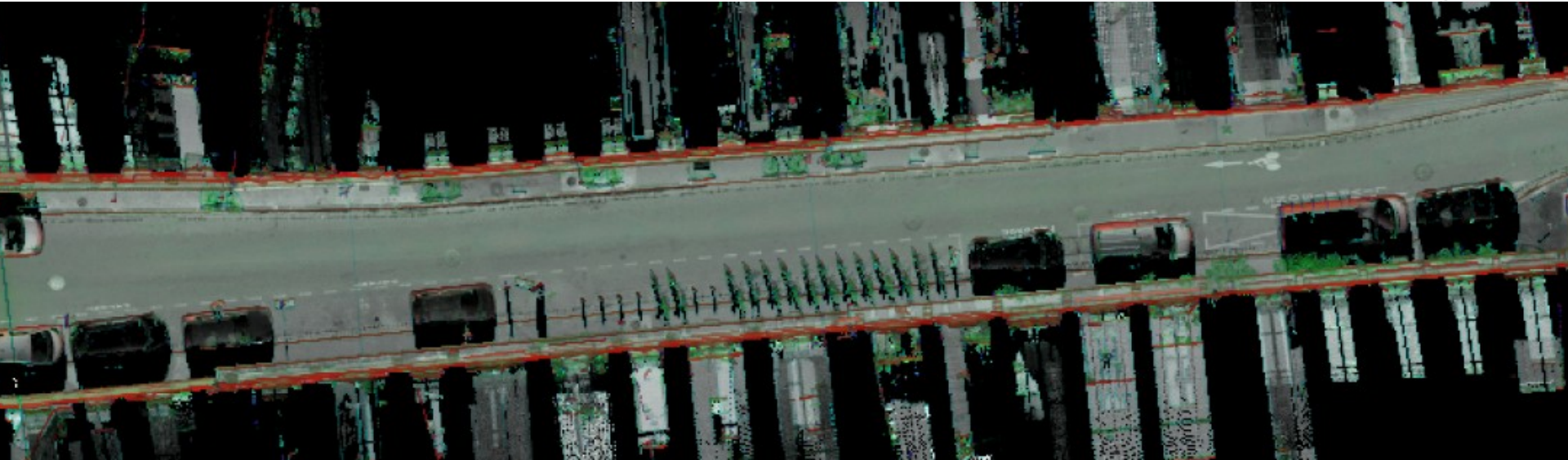


Traitement distribué de nuages massifs de points laser avec Spark.



Production (>1000 agents)

- Photographies aériennes, orthophotographies
- Création et maintenance de Bases de données 2D et 3D
- Cartographie
- Diffusion (geoportail) ...

Recherche (~100 chercheurs) : 5 équipes

- **LAREG** : Géodésie, Mathématiques appliquées ...
- **LOEMI** : Électronique, capteurs ...
- **MATIS** : Vision par ordinateur, Photogrammétrie, Laser, télédétection, Apprentissage, Informatique graphique, BigData ...
- **COGIT** : Analyse spatiale, Cartographie ...
- **LIF** : inventaire forestier

Plateformes d'acquisition de données

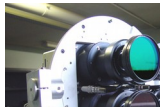
Mobile Mapping System (MMS)



trottoir



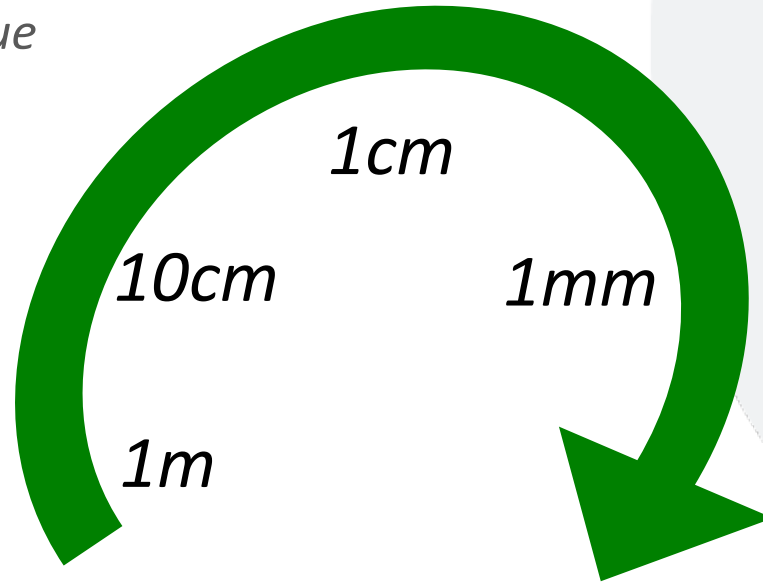
rue



ville



pays



monuments



bâtiment

Données acquises Mobile Mapping



IGN/Stereopolis

Trajectoire

- centrale inertielle + GPS + odomètre → position et orientation à 200Hz

Images

- 9 images totalisant 28MPix tous les 2m

Laser

- 300 000 points/s
- 84 octets/point
(non compressé)
- ~100Go/H
d'acquisition
- 6H par jour ...



Données acquises Mobile Mapping



IGN/Stereopolis

Trajectoire

- centrale inertielle + GPS + odomètre → position et orientation à 200Hz

Images

- 9 images totalisant 28MPix tous les 2m

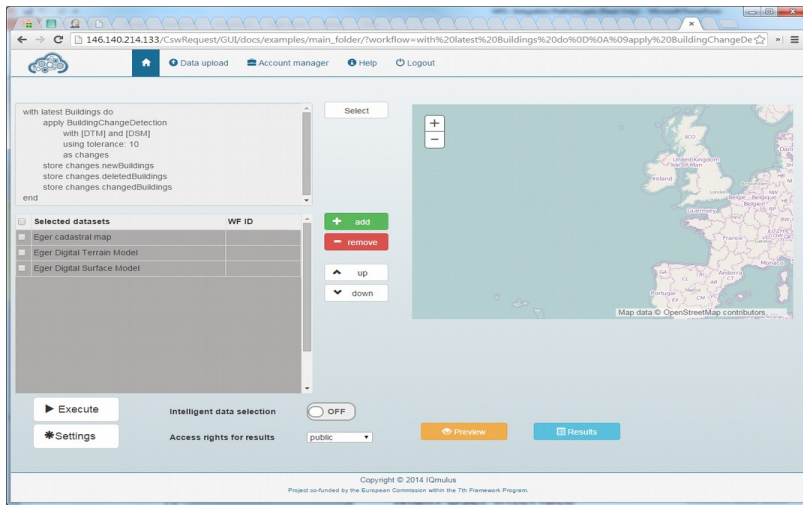
Laser

- 300 000 points/s
- 84 octets/point
(non compressé)
- ~100Go/H
d'acquisition
- 6H par jour ...

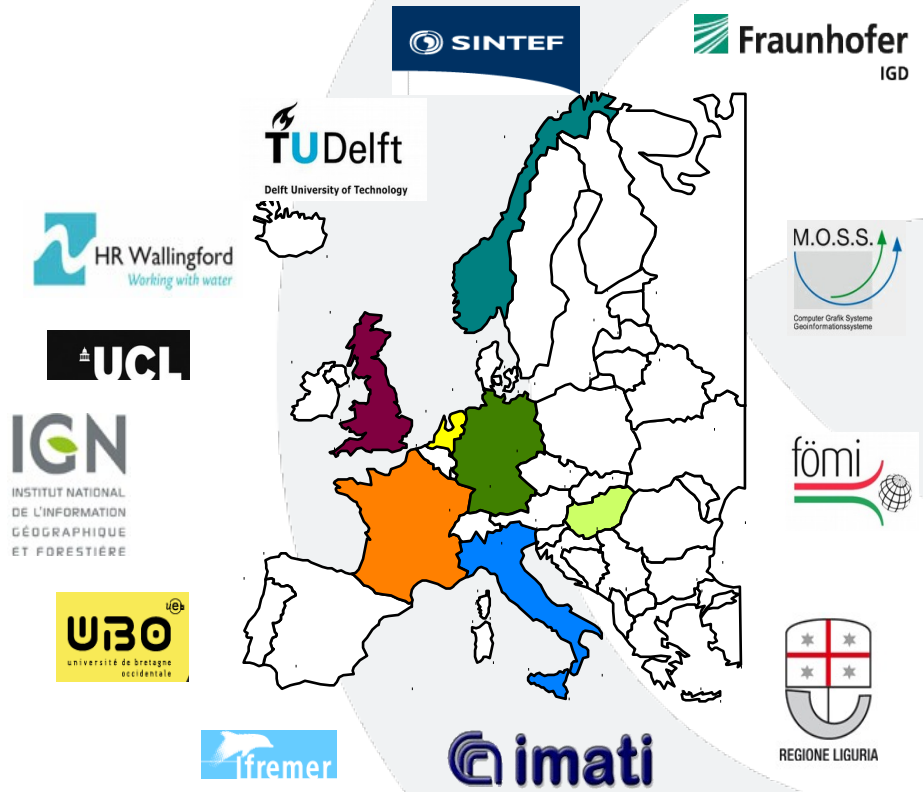
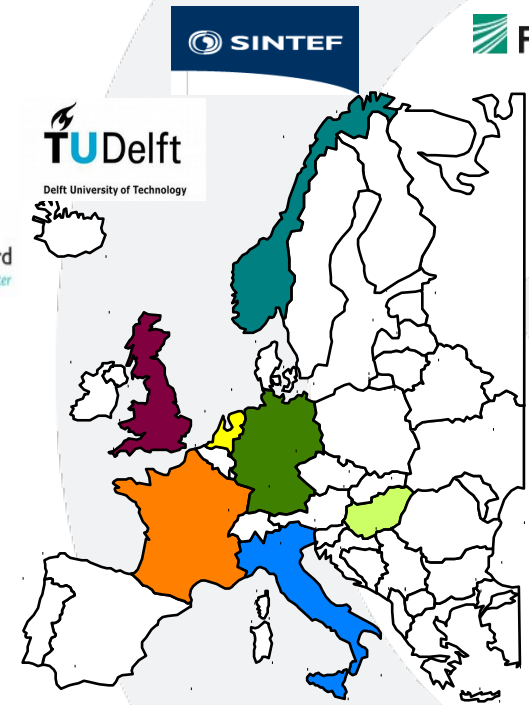


- **Volume !**
- **Variété : différents capteurs, lieux**
- **Vélocité : traiter aussi vite que l'acquisition**
- **Véracité : données de référence**
- **Valeur : de nombreuses applications**

« A High-volume Fusion and Analysis Platform for Geospatial Point Clouds, Coverages and Volumetric Data Sets »



→ IGN = Urban showcase



Infrastructure IQmulus

Plateforme dev actuelle du projet
→ Fraunhofer IGD

6 hypervisors (i.e. real machines)
→ Dual Octacore 2,7 GHz
→ 256 GB RAM

Total
→ 96 CPU cores, 259 GHz
→ 1536 GB RAM

Total storage
→ 19 TB



Spark cluster - standalone mode v1.5.1

- 1 master VM 6GB RAM
- 8 slaves VM 6GB RAM

Autres Services

- conteneurs Docker
- gestionnaire de jobs ad-hoc :(

Stockage des données

- HDFS
- Accessible en HTTP après authentification
- Visualisation webmapping ou WebGL directe!



Hors Projet IQmulus

- IGN: partenaire ISC-PIF depuis 01/2015
- Achat en cours de noeuds sur la grille IdF

Contexte applicatif

Orthophotographies aériennes

Résolution <10cm

Mais occultations : arbres, toits...

Motivation

- Orthophotographie de l'espace public
- ex : Cartographie des réseaux enterrés
 - Réglementation DT-DICT
 - EDF, Suez, etc doivent géoréférencer leurs plans locaux sur une donnée de référence...



Contexte applicatif

Proposition

OrthoLidar expressif à partir de laser MMS
[Brédif15]

Difficultés

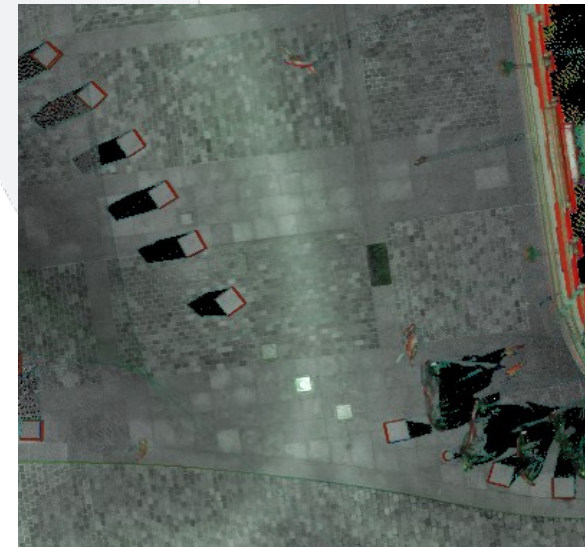
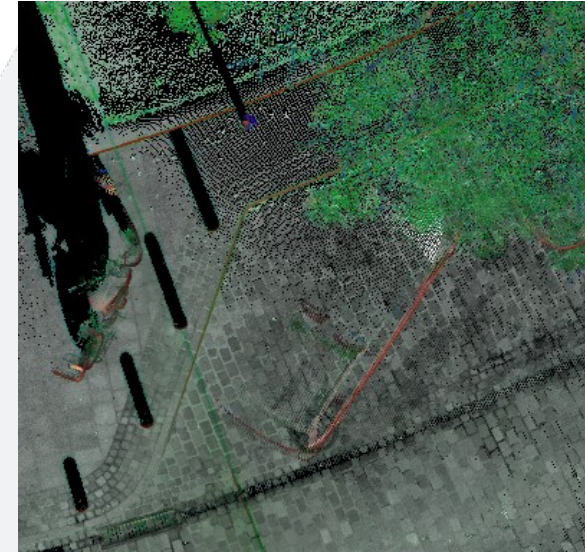
Temps de calcul

Volume de données

Problèmes de compréhensibilité

→ fausses couleurs

→ perte de la 3eme dimension (z)

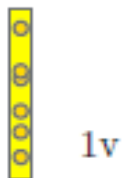
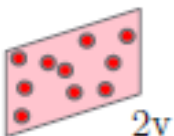


Stylisation du nuage de points

Géométrie locale : Dimensionnalité

ACP adaptative $\rightarrow D=(d_{1v}, d_{1h}, d_{2v}, d_{2h}, d_3)$ probabilité

Palette de couleurs : $C=(c_{1v}, c_{1h}, c_{2v}, c_{2h}, c_3)$



Teinte	Géométrie locale: couleur pondérée (D.C)
Saturation	Confiance en la caractérisation de la géométrie locale
Luminance	Réflectance horizontale Tone-mapping f, pondération par d_{2h}

Contraintes utilisateur

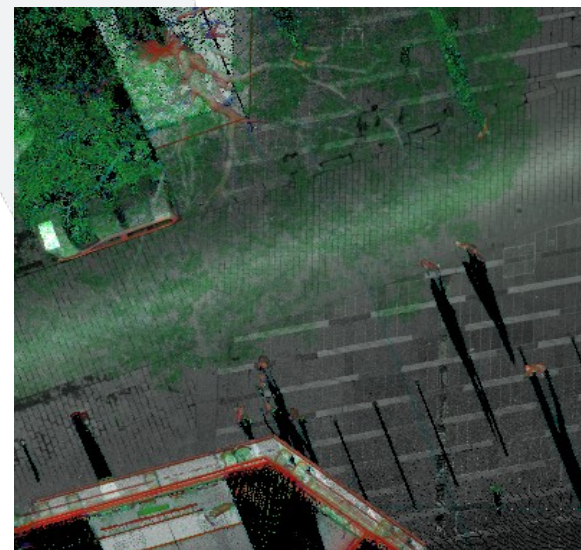
Arbres verts $\rightarrow c_3$ vert

Façades visibles $\rightarrow c_{2v}$ rouge

Réflectance au sol $\rightarrow c_{2h}$ blanc

\rightarrow pondérée par d_{2h}

(Fond blanc)

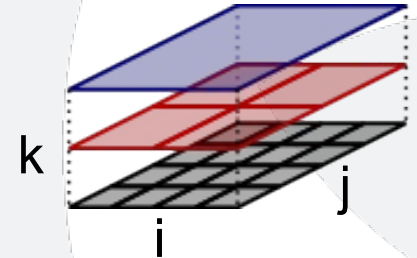


Entrée

- Ensemble de fichiers (~1 To, 3 000 000 points par fichier)
- Attributs utilisés des points : x,y,z,r (position et réflectance)

Sortie

- Pyramide d'images



$$color_{ijk} = f \left(\frac{\sum_{p \in P_{ijk}} d_{2h}(p) r(p)}{\sum_{p \in P_{ijk}} d_{2h}(p)} \right) \frac{\left(\sum_{p \in P_{ijk}} D(p) \right) \cdot C}{\sum_{p \in P_{ijk}} 1}$$

f : tone-mapping = bijection croissante vers $[0,1]$

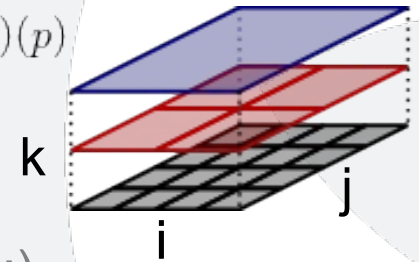
$P_{ijk} = \{\text{points laser inclus dans l'emprise du pixel } ijk\}$

Prétraitement

→ ACP de chaque point : $(x,y,z,r) \Rightarrow (x,y,r,d_{1v},d_{1h},d_{2v},d_{2h},d_3)$

→ Pyramide auxiliaire A à 6 canaux

$$A_{ijk} = (A_{ijk}^{2hr}, A_{ijk}^{1v}, A_{ijk}^{1h}, A_{ijk}^{2v}, A_{ijk}^{2h}, A_{ijk}^3) = \sum_{p \in P_{ijk}} (d_{2h}r, d_{1v}, d_{1h}, d_{2v}, d_{2h}, d_3)(p)$$



Itérations

→ Pyramide d'images A vers couleur : pixel à pixel ! :)

$$color_{ijk} = f \left(\frac{\sum_{p \in P_{ijk}} d_{2h}(p)r(p)}{\sum_{p \in P_{ijk}} d_{2h}(p)} \right) \frac{\left(\sum_{p \in P_{ijk}} D(p) \right) \cdot C}{\sum_{p \in P_{ijk}} 1} = f \left(\frac{A_{ijk}^{2hr}}{A_{ijk}^{2h}} \right) \frac{(A_{ijk}^{1v}, A_{ijk}^{1h}, A_{ijk}^{2v}, A_{ijk}^{2h}, A_{ijk}^3) \cdot C}{A_{ijk}^{1v} + A_{ijk}^{1h} + A_{ijk}^{2v} + A_{ijk}^{2h} + A_{ijk}^3}$$

→ sous échantillonnage additif de A :

$$A_{i,j,k+1} = A_{2i,2j,k} + A_{2i+1,2j,k} + A_{2i,2j+1,k} + A_{2i+1,2j+1,k}$$

$$\text{car } P_{i,j,k+1} = P_{2i,2j,k} \cup P_{2i+1,2j,k} \cup P_{2i,2j+1,k} \cup P_{2i+1,2j+1,k}$$

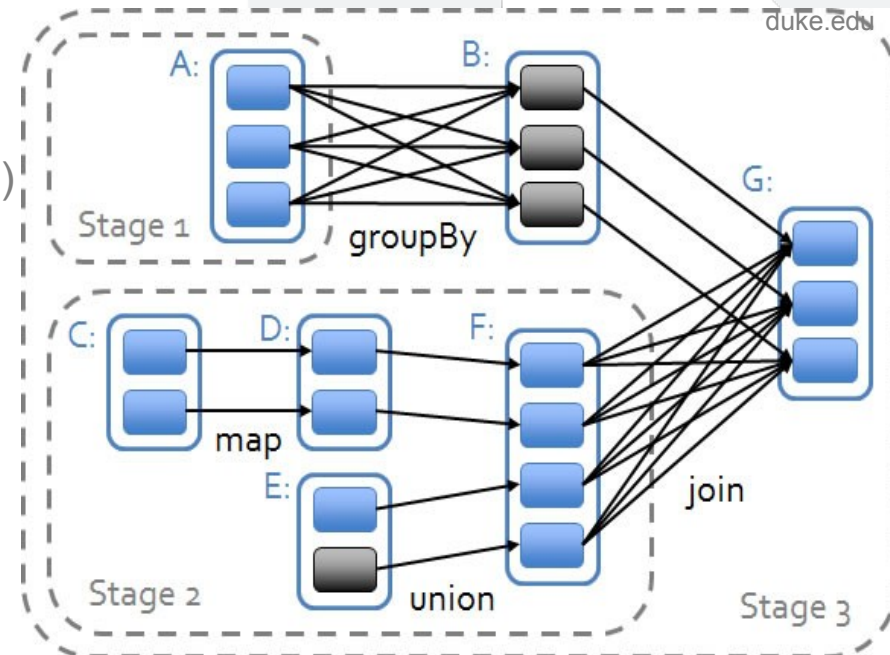
Spark Resilient Distributed Dataset (RDD)

- abstraction d'un dataset chargé dans la RAM distribuée du cluster (avec spilling si RAM trop petite)
- programmation fonctionnelle & paresseuse (scala)

Spark SQL & Dataframes

- API SQL avec optimisation du plan d'exécution

```
val A = sqlContext.read.parquet('a.parquet')  
val B = A.groupBy('id')  
val C = sqlContext.read.json('c.json')  
val D = C.map(f)  
val E = sqlContext.read.csv('e.csv')  
val F = D.union(E)  
val G = B.join(F)
```



Custom DataSource

→ lecture/écriture d'un ensemble de fichier laser (PLY ou LAS) comme une dataframe spark.

- Splittable InputFormat
- Project : seul les attributs nécessaires sont lus
- Filter : pruning des fichiers en se basant sur les statistiques en métadonnées ou dans un premier scan

```
val pointcloud1 : DataFrame = sqlContext.read.ply(filenamees)  
val pointcloud2 : DataFrame = sqlContext.read.las(filenamees)  
pointcloud1.write.las(outdir1)  
pointcloud2.write.ply(outdir2)
```

→ spark package spark-iqmulus en cours de dépôt open source
<https://github.com/IGNF/spark-iqmulus>

Voxelisation et ACP

// lecture des fichiers

```
val input = sqlContext.read.ply(filenamees)
```

// projection et renommage des attributs

```
val projected = input.select('x','y','z','reflectance as 'i')
```

// filtrage, test des boites englobantes des fichiers/splits

```
val filtered = projected where ('x > xmin) // ...
```

// création de la clé de tuile pour chaque point avec duplication des points au bord

```
val keyed = filtered flatMap addKey(resolution,N,W,_)
```

// agrégation des tuiles, en regroupant les voxels de la tuile et un bord de W voxels

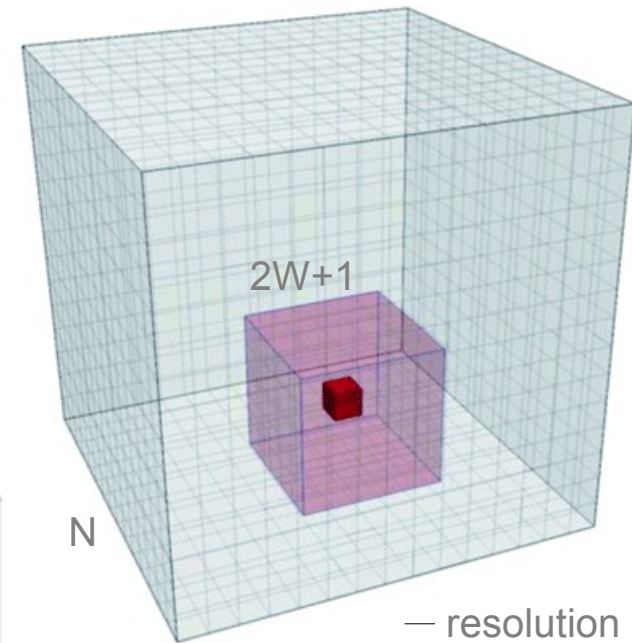
```
val tiled = keyed.aggregateByKey(Tile(N,W))(_ :+ _, _ ++ _)
```

// calcul pour chaque voxel d'une ACP, définie sur un voisinage $(2W+1)^3$

```
val mapped = tiled.mapValues(_.windowedmap(W,ACP))
```

// écriture des tuiles

```
mapped foreach { case ((tx,ty,tz),tile) => tile.saveAsPly(res,tx,ty,tz,s"$out/$tx/$ty/$tz.ply")}
```





Dataset


10km, 10^9 points, 81.5 Go (ACP précalculée)

8 Spark slaves

Pyramide d'images

2cm de résolution : 17min, 511Mo

1cm de résolution : 41min, 2Go

- 
- **Ecriture de la pyramide de PNG sur le HDFS**
 - **Génération d'une page web avec un js de webmapping (leaflet) sur le HDFS**
 - **Acces HTTP sécurisé au HDFS**

Alternative : Webservice HTTP
→ **1 seule dalle**
1024x1024 : de 30s à 5min



Perspectives

- **Interpolation de l'image finale : boucher les trous**
- **Optimisations (mémoire, plan d'exécution)**
- **Autres workflows**
- **Opérations plus complexes avec différentes localités**
- **Domain Specific Language (DSL) permettant à l'utilisateur/développeur de se focaliser sur ses fonctions métier et de lui cacher la complexité sous-jacente**

A person wearing a dark, textured jacket and red pants is walking on a cobblestone path. The path is adjacent to a green lawn on the left. The person is walking away from the camera, and their shadow is cast on the path. The text "Merci !" is overlaid on the image in white, bold font.

Merci !

Questions ?