Practical: 3d point cloud processing with Cloudcompare (CC) for beginners*

D. Lague, CNRS

Nantes-Rennes Topo-bathymetric Lidar Platform

Géosciences Rennes

Dimitri.lague@univ-rennes1.fr

* With a focus on airborne and terrestrial lidar data in the context of environmental research

Objectives

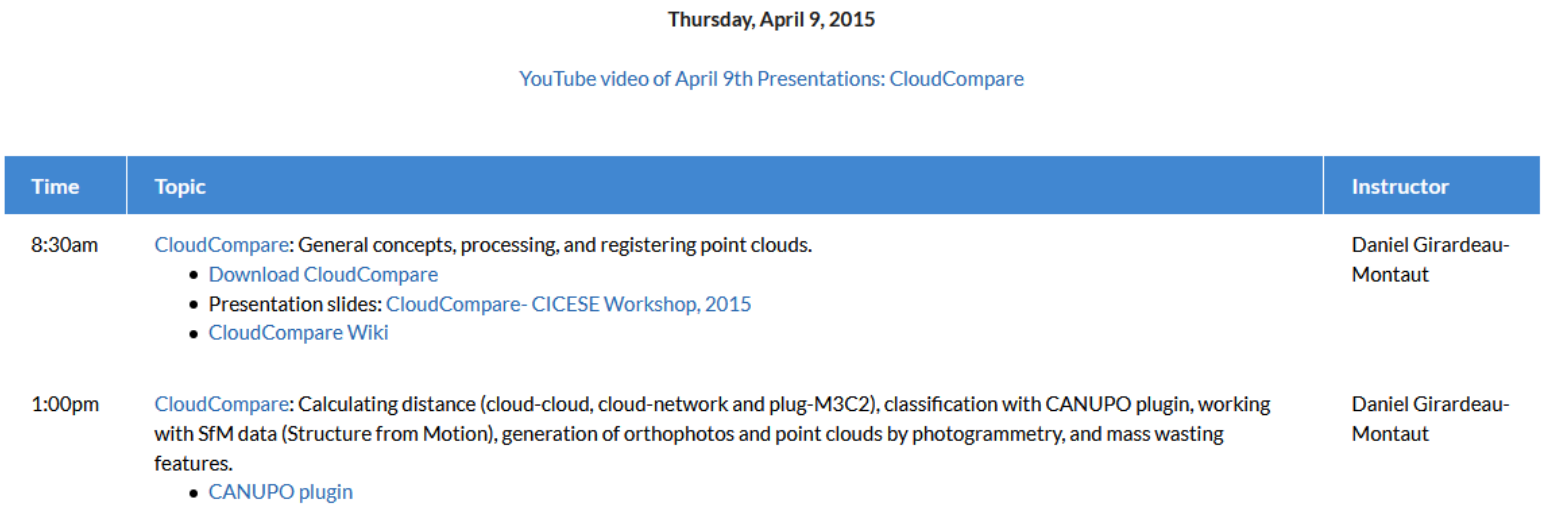
1. Basic operations and visualisation in CloudCompare
2. How to turn 3D data to 2D raster grids (rasterize tool)
3. Cloud comparison : 3D distance measurement and change detection (qM3C2)
4. Vegetation classification with qCANUPO

What you won’t find here:

- Point cloud registration
- Transformation
- Command line operation
- Feature calculation
- And many other things that CC can do
    - …. But if you follow all this practical, and if you read the wiki and experiment yourself, you should be able to

Additional ressources

- The cloudcompare website contains
  - A detailed wiki (english & french)
  - Introductory videos
  - A forum
- There's a lot of youtube videos
- This workshop contains great introductory videos



Thursday, April 9, 2015

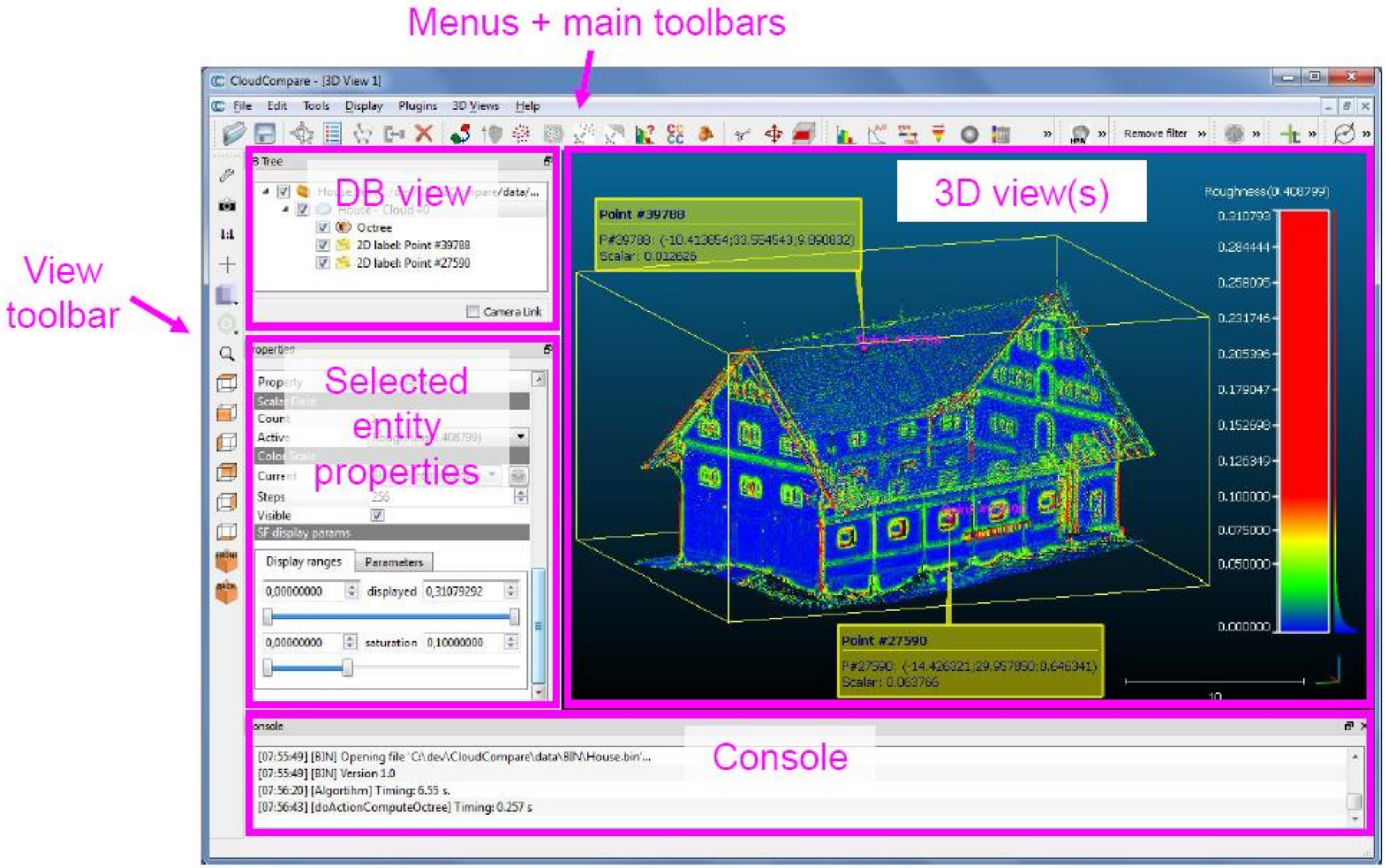
YouTube video of April 9th Presentations: CloudCompare

Time	Topic	Instructor
8:30am	CloudCompare: General concepts, processing, and registering point clouds. • Download CloudCompare • Presentation slides: CloudCompare- CICESE Workshop, 2015 • CloudCompare Wiki	Daniel Girardeau-Montaut
1:00pm	CloudCompare: Calculating distance (cloud-cloud, cloud-network and plug-M3C2), classification with CANUPO plugin, working with SfM data (Structure from Motion), generation of orthophotos and point clouds by photogrammetry, and mass wasting features. • CANUPO plugin	Daniel Girardeau-Montaut

Cloudcompare what for ?

- GUI : 3D point cloud vizualization and interaction for moderately size data*
  - *On a modern laptop with a dedicated GPU, you can work with point clouds up to 50-100 millions*
  - *On a dedicated workstation with a powerful graphic card, up to 500 million points*
- GUI : Test ideas, explore data, make advanced point cloud analysis, nice visualisation (all in 3D)
- Large dataset processing with command line (no GUI)
- Well suited for change detection
- Limited capabilities for classification

User interface overview



```
Point #39788
P#39788: (-10.413854;33.554543;9.890832)
Scalar: 0.012626
```

```
Point #27590
P#27590: (-14.426321;29.957850;0.646341)
Scalar: 0.063766
```

```
0,00000000    displayed 0,31079292
0,00000000    saturation 0,10000000
```

```
[07:55:49] [BIN] Opening file 'C:\dev\CloudCompare\data\BIN\House.bin'...
[07:55:49] [BIN] Version 1.0
[07:56:20] [Algorithm] Timing: 6.55 s.
[07:56:43] [doActionComputeOctree] Timing: 0.257 s
```

Input/Output

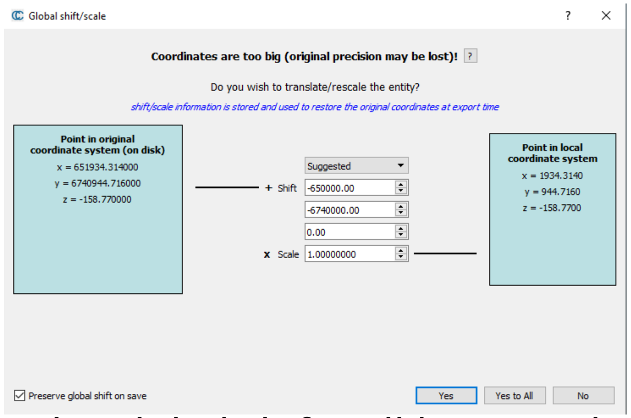
- Mainly point clouds (ASC/PTS, LAS/LAZ, E57, PTX, FLS/FWS, DP, etc.) and triangular meshes (PLY, OBJ, STL, OFF, FBX)
- Dedicated format: “BIN” (for projects)
- Other formats: calibrated photos (Bundler .OUT), CAD (Autocad DXF drawings, Aveva .PDMS scripts), GIS shapefiles

- Point cloud from Airborne Lidar are generally delivered as LAS files, or even better LAZ (a compressed file).
- They can be delivered as flight lines, or as tiles
- LAS/LAZ format has provision for several « attributes » (called scalar fields in CC), in particular: *intensity, number of returns, return number, classification, scan angle, RGB information etc…*
- Terrestrial lidar maybe delivered in LAS format, but will generally be in E57 format

IMPORTANT : **CC has no intrinsic units. They correspond to the unit in which the point cloud coordinates are stored**. *E.g., Airborne lidar data is stored in meters*

Coordinate systems and global shifts

- CC operates intrinsically with floats (to save memory)
- If you were to work directly with a Global Coordinate System (e.g. UTM), coordinates would be significantly rounded
- CC detects at loading time that are coordinates are too big, and will suggest a local translation (global shift) to preserve the precision



Global shift/scale

Coordinates are too big (original precision may be lost)!

Do you wish to translate/rescale the entity?

shift/scale information is stored and used to restore the original coordinates at export time

Point in original coordinate system (on disk)
x = 651934.314000
y = 6740944.716000
z = -158.770000

Suggested
+ Shift -650000.00
-6740000.00
0.00
x Scale 1.00000000

Point in local coordinate system
x = 1934.3140
y = 944.7160
z = -158.7700

☑ Preserve global shift on save

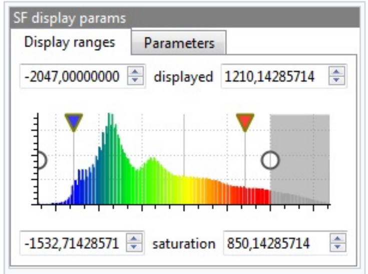
- At export/saving time the global shift will be reapplied to the data
- To ensure that global shifts are similar for all data in a given study area, make sure that you always use the same
- Tip : don't apply a translation on Z to keep working in real elevation
- Advanced user: *you can store a list of global shift in the CC.exe directory with a name for faster application of a global shift*

Scalar operations

- Visualisation and color scale

Scalar field display parameters editor

In the Properties dialog, an editor very specific to CloudCompare can be found on clouds and entities with an active scalar field:



Scalar field parameters editor

This editor lets the user set most of the display parameters of the active scalar field in a very concise way (in addition to get a quick overview of the scalar field histogram):

- use the left and right white circles to set the min and max displayed scalar values. Below and above those values, the points will either appear in grey or won't appear at all (depending on the parameter '*show NaN/out of range values in grey*' that is accessible in the 'Parameters' tab). This way the user can quickly hide points with values outside a given interval, in order to focus on the other points for instance.
- use the red and blue down arrows to set the min and max saturation values (to set the start and end of 'relative' color scales – see the Color Scales Manager)
- set additional parameters with the 'Parameters' tab (log or symmetrical scale, etc.)

- Histogram
- Filter by value (poor's man segmentation)
- Arithmetics

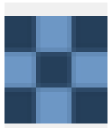
Basic operations

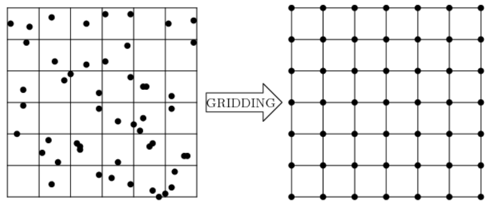
- **Clone** : data duplication -> IMPORTANT as there's no ctrl-z (undo) in CC
- **Merge** : select 2 or several clouds and merge
- **Subsample** : random, or by minimum distance *(very useful to simplify point cloud)*
- **Segment** : select one or several cloud and then cut it in the current 3D view
- **Filter by scalar field value :** central operation in CC
- **Arithmetics on scalar fields**

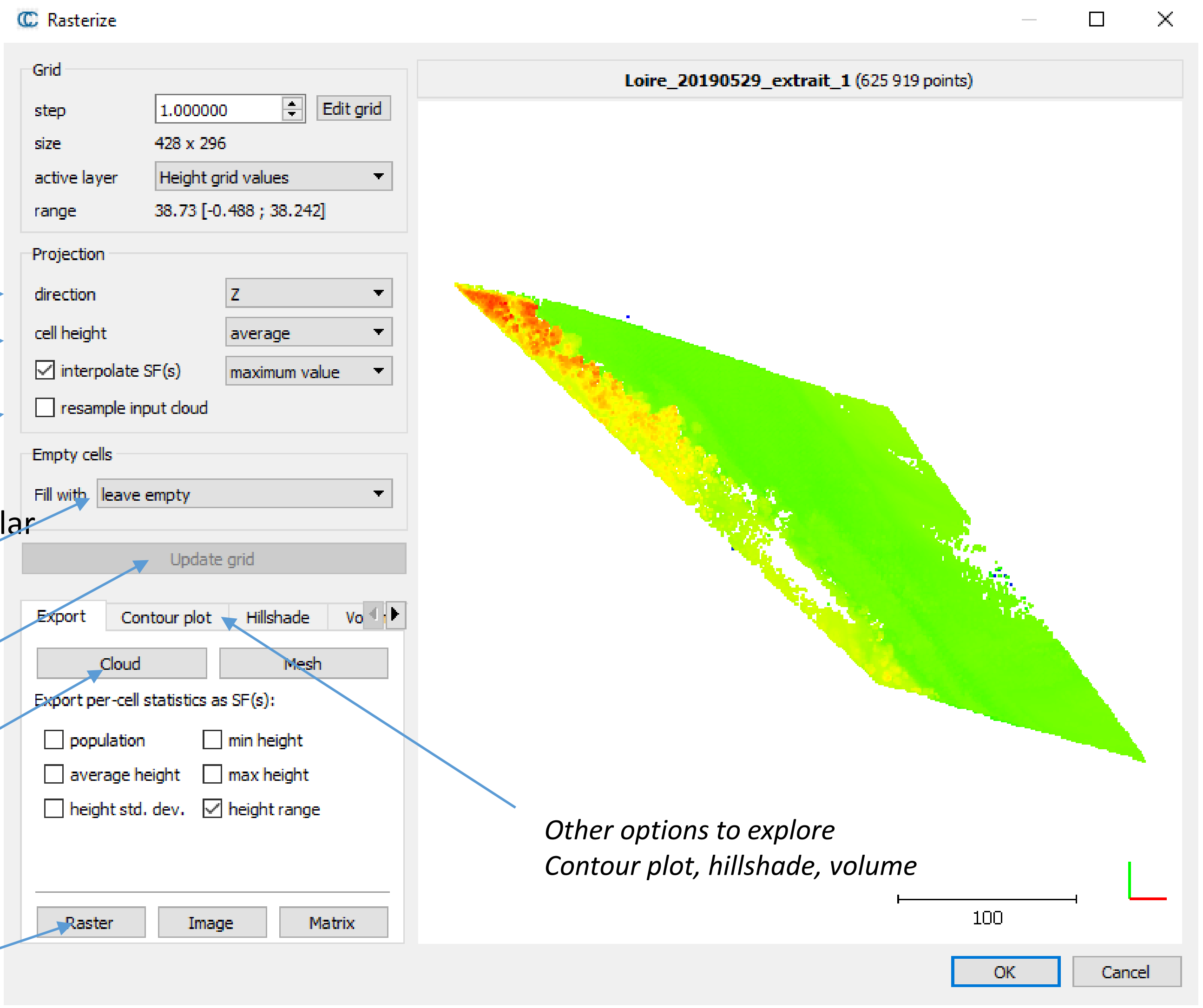
Advanced operations

- <u>Export coordinate as a scalar field:</u>
  - Menu tools -> projection -> export coordinate to scalar field-> select z
  - Perform min-max operation on z (or smoothing or anything on elevation
- Segmentation with Label Connected Components
- Normal calculation
  - On airborne lidar, choose a radius to compute the normal
  - Orient the normal with +Z (should systematically work)
  - On terrestrial lidar, a bit more complex due to overhanging parts -> advanced stuff

Rasterizing function







Rasterize

Grid
step 1.000000 Edit grid
size 428 x 296
active layer Height grid values
range 38.73 [-0.488 ; 38.242]

Projection
direction Z
cell height average
☑ interpolate SF(s) maximum value
☐ resample input cloud

Empty cells
Fill with leave empty

Update grid

Export | Contour plot | Hillshade | Vo

Cloud | Mesh

Export per-cell statistics as SF(s):
☐ population ☐ min height
☐ average height ☐ max height
☐ height std. dev. ☑ height range

Raster | Image | Matrix

Loire_20190529_extrait_1 (625 919 points)

OK | Cancel

Pixel size

On Height

Vertical projection

Min, average, Max

If ticked, only keep existing points (thinning)
-> the resulting point cloud is not regular

With or without Linear interpolation

Update button any time you change a parameter

Create a cloud (with regular x,y)

Save as GEOTIFF for 2D GIS use

Other options to explore
Contour plot, hillshade, volume

Applications of rasterize

- « Rough » DEM calculation : lowest point calculation
  - But you’re never sure that the lowest point is the ground under vegetation
  - Does not work on buildings and urban environments !
- DSM approximation: highest point calculation
- CHM as a scalar field: save « height range » for a given pixel
  - gives a crude approximation of vegetation height

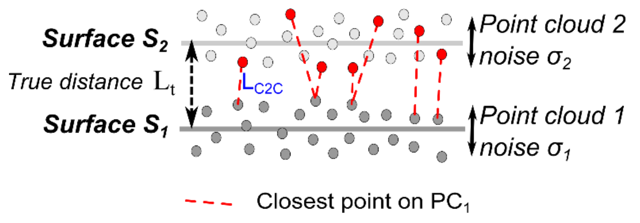
Distance measurement between 2 points clouds

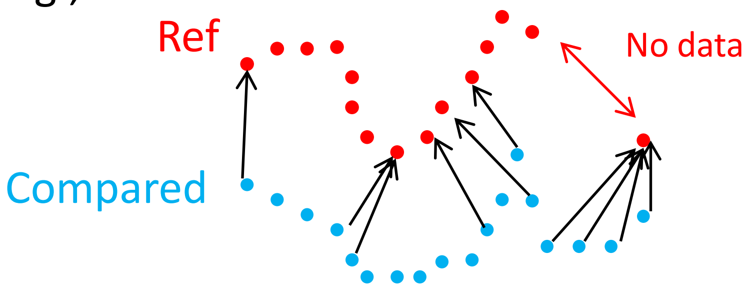
Hypothesis: *2 point clouds in the same coordinate system. How can we measure distances directly in 3D ?*

1. simplest approach : nearest point distance (cloud 2 cloud)

- Pro: super fast and simple, no need to rasterize
- Con:
  - **noisy data** → underestimation of true distance
  - Non-oriented measurement
  - Non-signed measurement
  - Highly sensitive to missing data
- Great for:
  - Quick analysis of change before refined measure
  - Low precision 3D distance between clouds (e.g., vegetation points and ground)

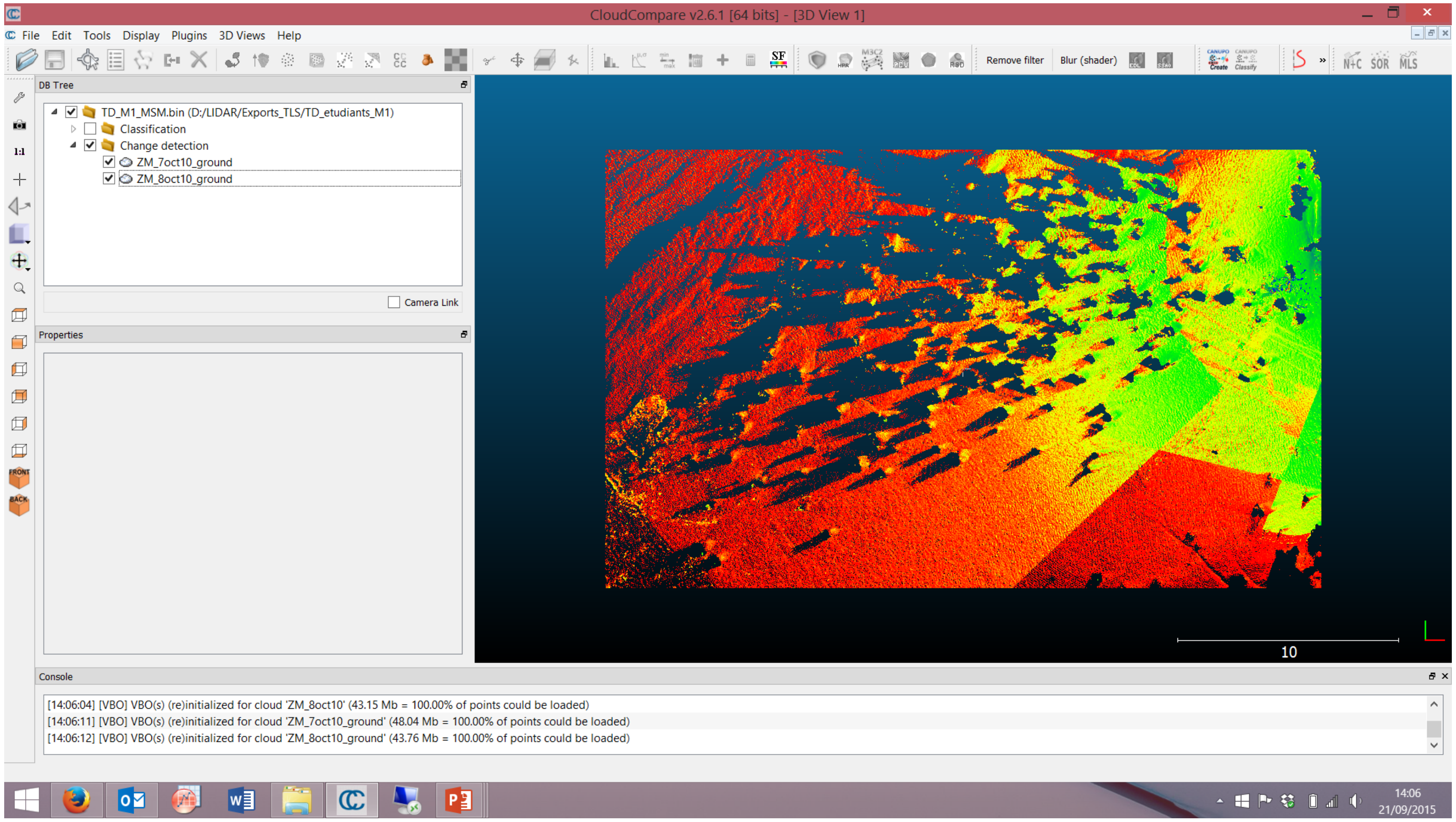
A: Closest point distance L_{C2C}





2. Accurate approach: qM3C2

Ex: load the mt st michel data from 7th and 8th october (Terrestrial lidar)

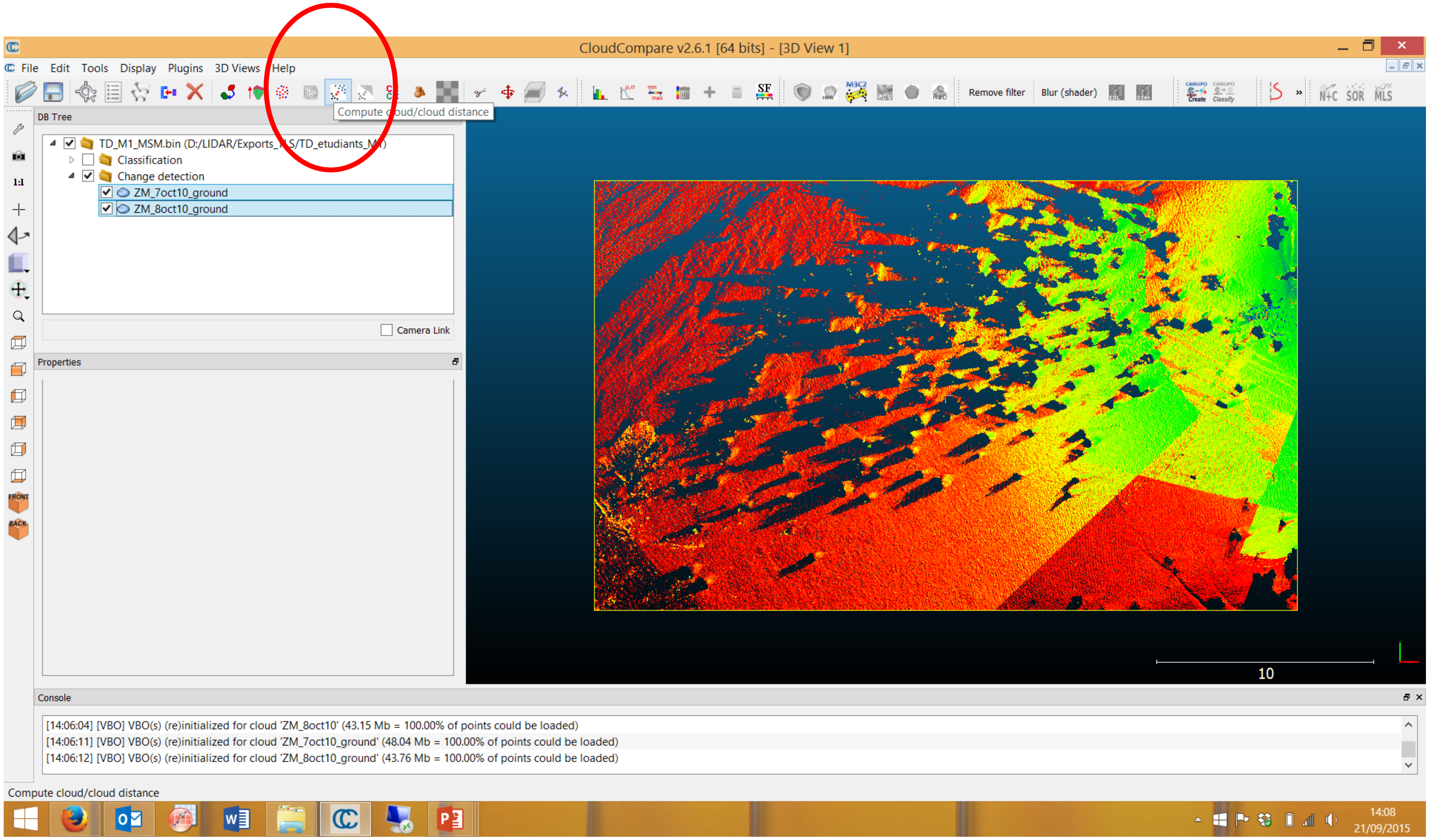

CloudCompare v2.6.1 [64 bits] - [3D View 1]
File Edit Tools Display Plugins 3D Views Help
Remove filter
Blur (shader)
DB Tree
Camera Link
Properties
Console
10
14:06
21/09/2015

TD_M1_MSM.bin (D:/LIDAR/Exports_TLS/TD_etudiants_M1)
- Classification
- Change detection
  - ZM_7oct10_ground
  - ZM_8oct10_ground

```
[14:06:04] [VBO] VBO(s) (re)initialized for cloud 'ZM_8oct10' (43.15 Mb = 100.00% of points could be loaded)
[14:06:11] [VBO] VBO(s) (re)initialized for cloud 'ZM_7oct10_ground' (48.04 Mb = 100.00% of points could be loaded)
[14:06:12] [VBO] VBO(s) (re)initialized for cloud 'ZM_8oct10_ground' (43.76 Mb = 100.00% of points could be loaded)
```

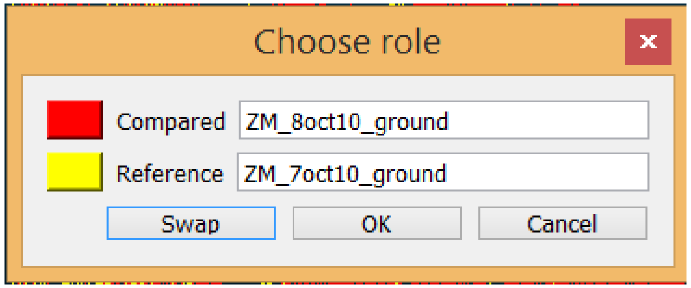
Cloud/cloud comparison in CC

- Select the 2 clouds and click on « Compute Cloud/cloud distance



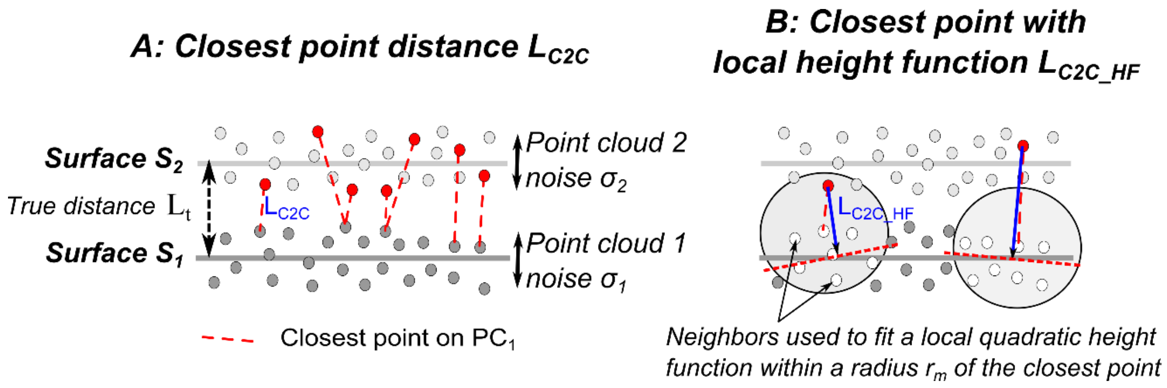
```
[14:06:04] [VBO] VBO(s) (re)initialized for cloud 'ZM_8oct10' (43.15 Mb = 100.00% of points could be loaded)
[14:06:11] [VBO] VBO(s) (re)initialized for cloud 'ZM_7oct10_ground' (48.04 Mb = 100.00% of points could be loaded)
[14:06:12] [VBO] VBO(s) (re)initialized for cloud 'ZM_8oct10_ground' (43.76 Mb = 100.00% of points could be loaded)
```

Parameters

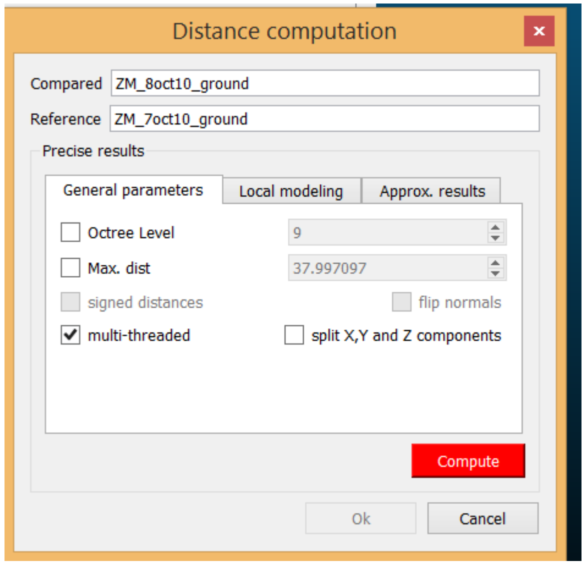

Choose role

🟥	Compared	ZM_8oct10_ground
🟨	Reference	ZM_7oct10_ground

Swap · OK · Cancel


A: Closest point distance L_{C2C}
B: Closest point with local height function L_{C2C_HF}
Surface S_2
True distance L_t
Surface S_1
L_{C2C}
Point cloud 2 noise σ_2
Point cloud 1 noise σ_1
Closest point on PC_1
L_{C2C_HF}
Neighbors used to fit a local quadratic height function within a radius r_m of the closest point

We generally choose as reference the first epoch (here 7 oct)


Distance computation

Compared: ZM_8oct10_ground
Reference: ZM_7oct10_ground

Precise results

General parameters | Local modeling | Approx. results

- [ ] Octree Level: 9
- [ ] Max. dist: 37.997097
- [ ] signed distances
- [ ] flip normals
- [x] multi-threaded
- [ ] split X,Y and Z components

Compute · Ok · Cancel

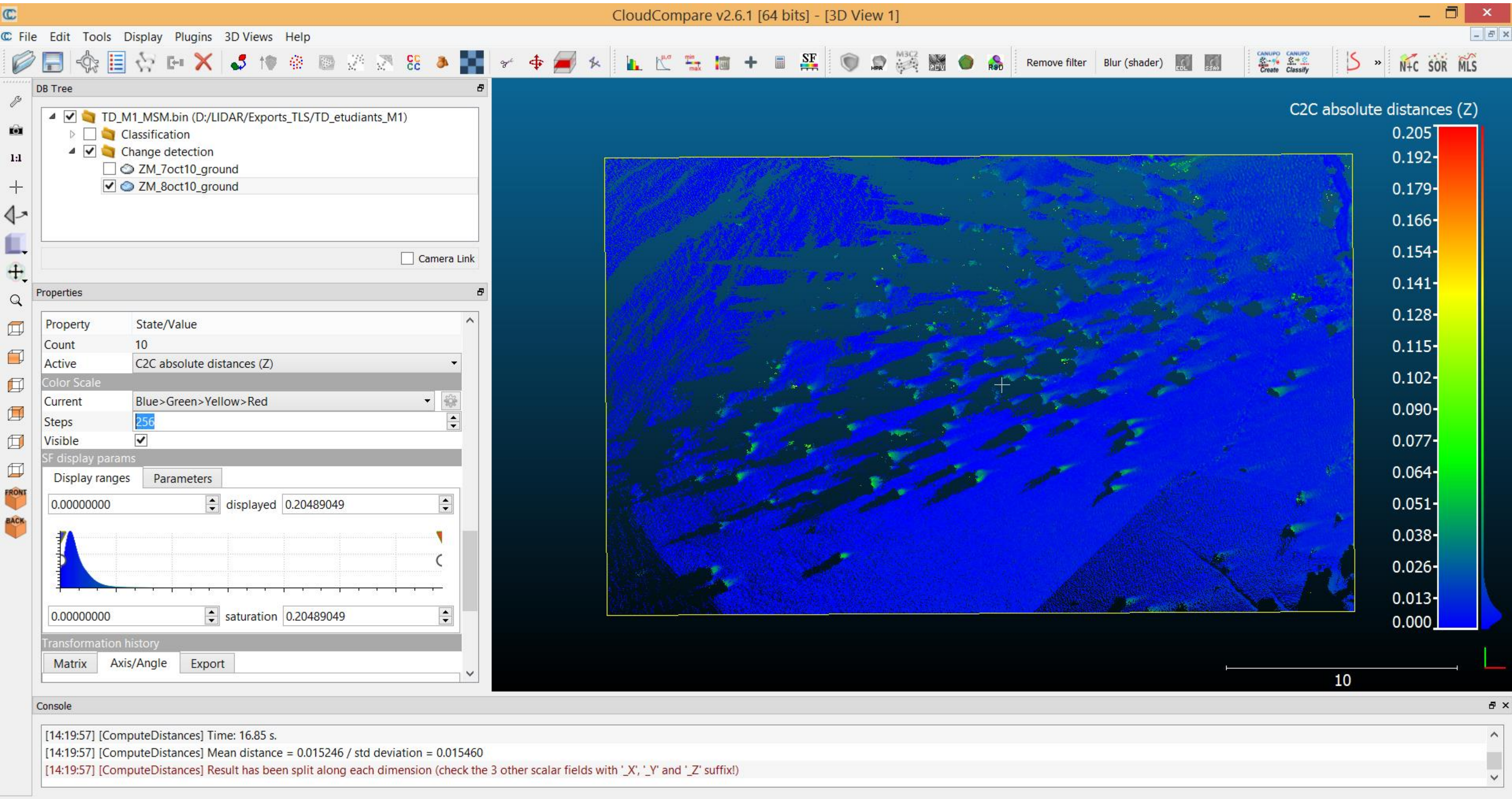
Simple computation: just click on Compute

Advanced calculation

- **split X,Y,Z** -> very useful to get a SIGNED vertical component of the 3D vector
- Local modelling -> (cf figure B) -> rarely used as qM3C2 is generally much better

After the computation

- The compared point cloud has 1 or several new scalar fields
- Choose C2C_absolute_distance (Z) (=signed vertical component)
- Display the histogram



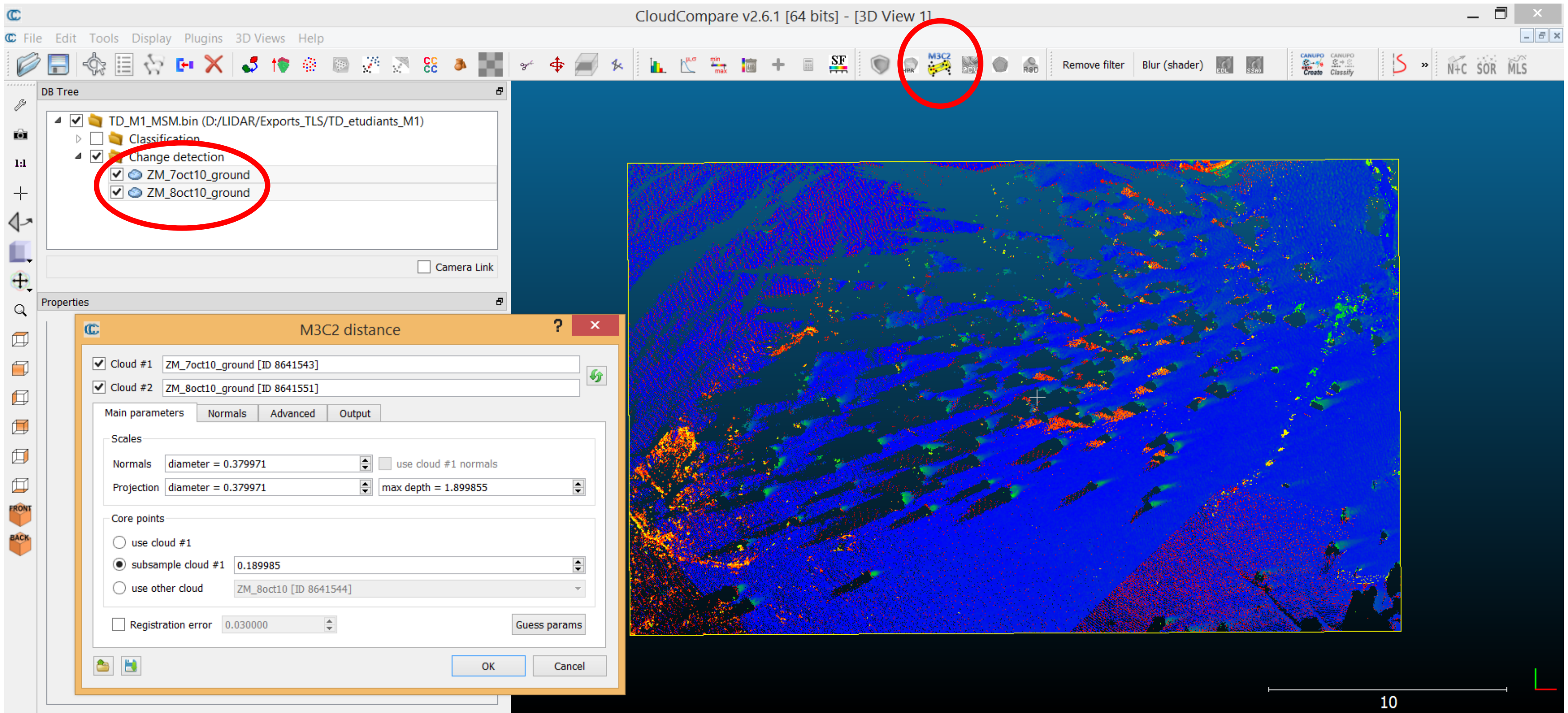
Property	State/Value
Count	10
Active	C2C absolute distances (Z)
Color Scale	
Current	Blue>Green>Yellow>Red
Steps	256
Visible	✓
SF display params	

0.00000000 displayed 0.20489049

0.00000000 saturation 0.20489049

```
[14:19:57] [ComputeDistances] Time: 16.85 s.
[14:19:57] [ComputeDistances] Mean distance = 0.015246 / std deviation = 0.015460
[14:19:57] [ComputeDistances] Result has been split along each dimension (check the 3 other scalar fields with '_X', '_Y' and '_Z' suffix!)
```

More accurate vertical distance with qM3C2


CloudCompare v2.6.1 [64 bits] - [3D View 1]
File Edit Tools Display Plugins 3D Views Help
Remove filter
Blur (shader)
DB Tree
Camera Link
Properties
10

TD_M1_MSM.bin (D:/LIDAR/Exports_TLS/TD_etudiants_M1)
- Classification
- Change detection
  - ZM_7oct10_ground
  - ZM_8oct10_ground

M3C2 distance

Cloud #1 ZM_7oct10_ground [ID 8641543]
Cloud #2 ZM_8oct10_ground [ID 8641551]

Main parameters | Normals | Advanced | Output

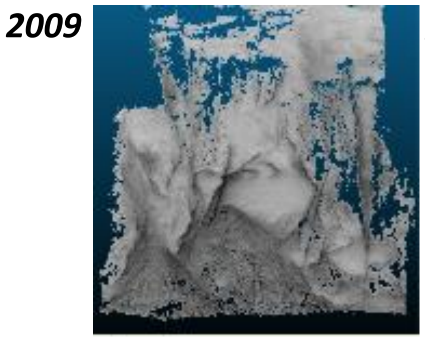
Scales
Normals diameter = 0.379971 — use cloud #1 normals
Projection diameter = 0.379971 — max depth = 1.899855

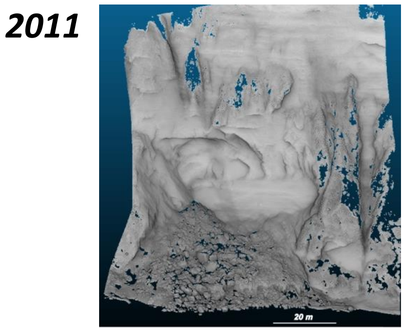
Core points
- use cloud #1
- [✓ subsample cloud #1] 0.189985
- use other cloud ZM_8oct10 [ID 8641544]

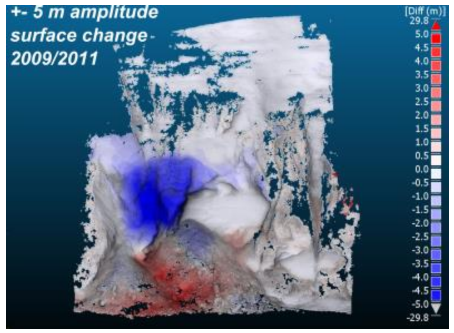
Registration error 0.030000 — Guess params

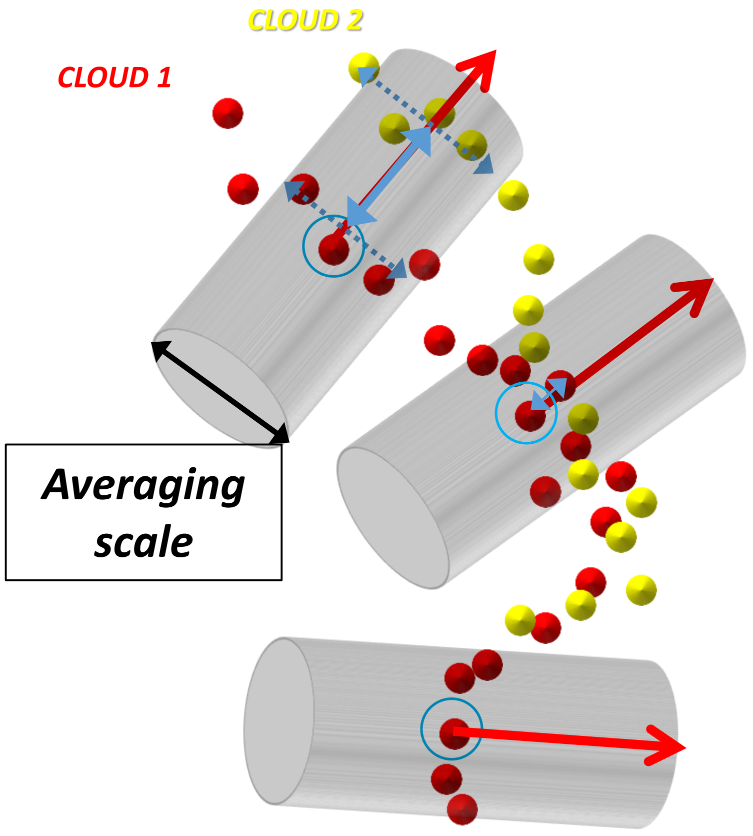
OK Cancel

M3C2: Acccurate 3D point cloud differencing *(Lague et al., 2013)*









1: Normal direction calculation on cloud 1

→ **Oriented difference**

*2: **Average distance** between the two PC*

→ **Noise and roughness averaging**

3: Local confidence interval calculation

→ **Local roughness**

→ **Local point density**

→ **Global registration**

4: Distance smaller than confidence interval

→ **statistically not significant**

5: No intercept with other cloud

→ **no calculation**

→ **no need to trim the data**

M3C2 options

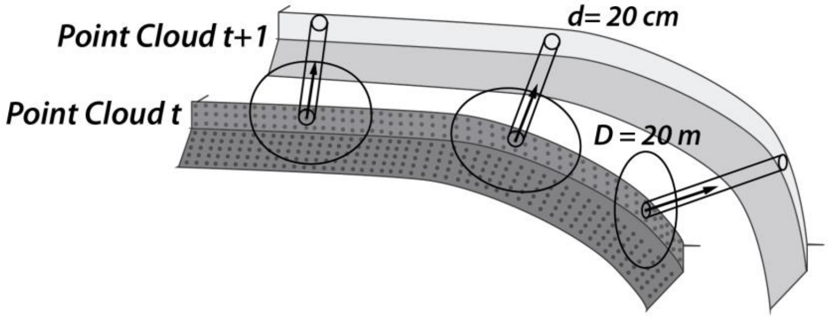
Option 1: **Vertical mode**

→ **no normal calculation →** Faster

Horizontal mode

⟶ Bank or cliff retreat

⟶ No need to rotate the data



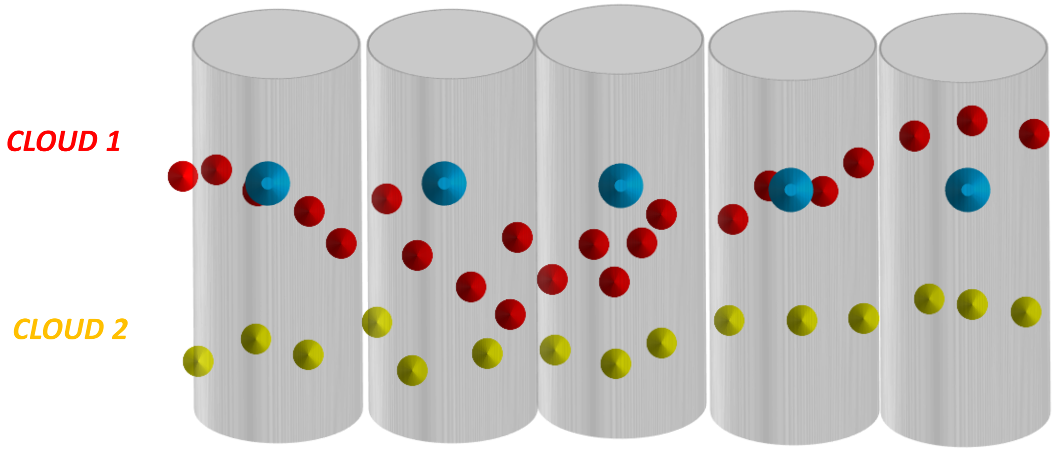
Option 2: **CORE POINTS**

- **Subset of points of arbitrary geometry**
  - **→ Grid of core points solve the sampling irregularity issue**
- **Calculation using the RAW DATA**
  - **→ Faster**

Core point grid + vertical mode

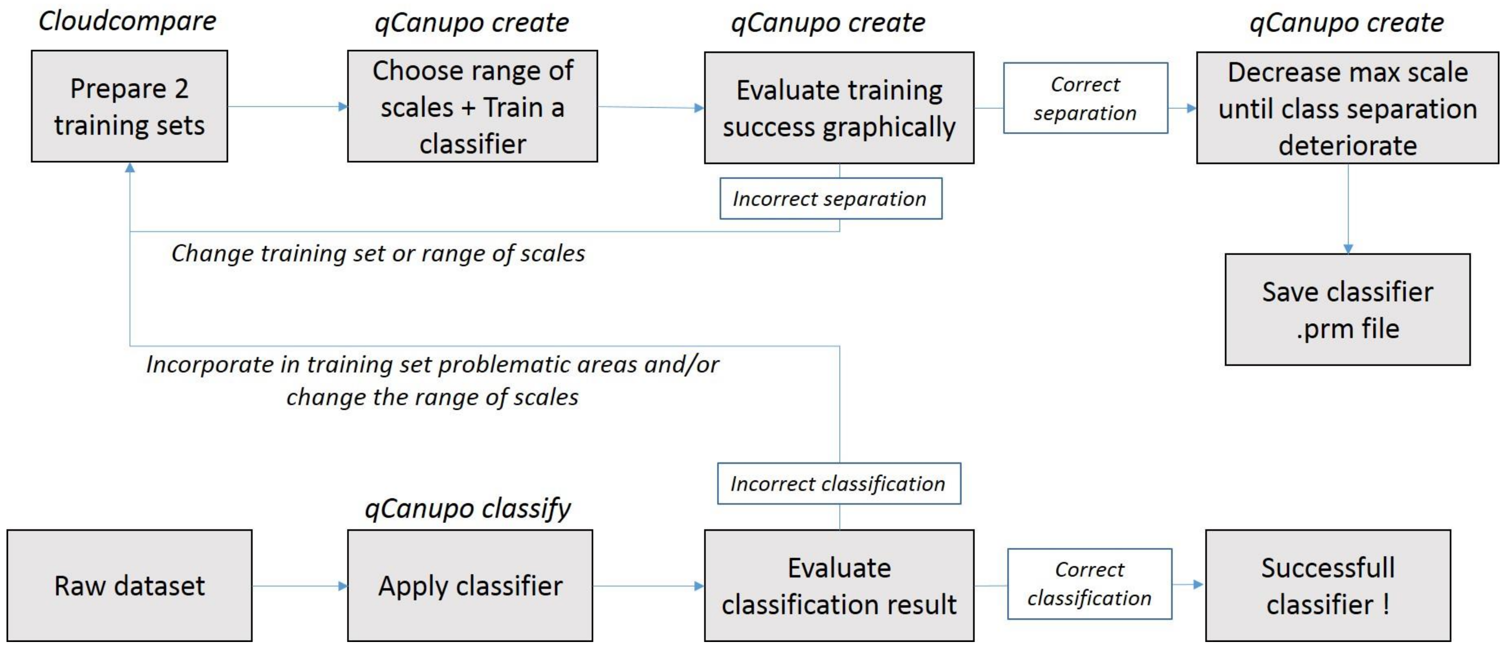
=

Equivalent to difference of DEM without gridding issues



Qcanupo: binary semantic classification using multiscale dimensionality

- Generic workflow:


Cloudcompare
Prepare 2 training sets
qCanupo create
Choose range of scales + Train a classifier
qCanupo create
Evaluate training success graphically
Correct separation
qCanupo create
Decrease max scale until class separation deteriorate
Incorrect separation
Change training set or range of scales
Save classifier .prm file
Incorporate in training set problematic areas and/or change the range of scales
Incorrect classification
qCanupo classify
Raw dataset
Apply classifier
Evaluate classification result
Correct classification
Successfull classifier !

- Application:
  - Try on the Mt St Michel Terrestrial lidar dataset : first merge the vegetation and ground points into a single cloud (to put yourself in a realistic configuration ! No cheating ;-).
  - Make a copy of the merge cloud
  - Manually select (with the scissor tool) samples of vegetation (and vegetation only) and samples of ground
  - Use qCanupo create to train the classifier : choose the scales meaningfully (about 10 scales is enough).
  - Evaluate the accuracy on the training data
  - Apply the classifier to the complete dataset and visually evaluate its quality