PC filtering and classification with PDAL and a voxel-based classification using Python

B. Bookhagen and A. Rheinwalt

01-10-2019

PC Filtering and Classification Steps

Setup your environment with:

1. Start Anaconda Prompt
2. `conda activate lidar`
3. Navigate to the directory with your LAS/LAZ data: `cd \ D:\PC_Workshop_Oct2019 d:`
4. We will work with `ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz`

First Steps: Obtaining information about LAS/LAZ files.

```
pdal info --boundary  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz
lasinfo ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz
```

PC filtering

See decimation, voxelcenternearestneighbor, voxelcentroidnearestneighbor, and Poisson Sampling for more details.

Apply voxelcenternearestneighbor:

```
pdal translate  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
-o  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_50cm.laz  \
    ^
voxelcenternearestneighbor ^
--filters.voxelcenternearestneighbor.cell=0.5
```

Apply voxelcentroidnearestneighbor:

```
pdal translate  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
-o  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_centroid50cm.laz  \
    ^
voxelcentroidnearestneighbor ^
--filters.voxelcentroidnearestneighbor.cell=0.5
```

For the UAV data, you may use:

```
pdal translate  \
    UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz ^
  -o  \
      UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_centroid50cm.laz  \
      ^
  voxelcentroidnearestneighbor ^
  --filters.voxelcentroidnearestneighbor.cell=0.5
```

Alternative, you can use a PDAL pipeline to perform the filtering and put all parameters and steps into the pipeline:

```
{
  "pipeline":[
    "mavicpro2_nadir_15deg_highq_dense_PC.laz",
    {
      "type":"filters.voxelcenternearestneighbor",
      "cell":0.1
    },
    "mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz"
  ]
}
```

PC classification

Using SMRF:

```
pdal translate  \
    UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz ^
  -o  \
      UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_SMRF_cl2only.laz  \
      ^
  smrf range ^
  --filters.range.limits="Classification[2:2]" ^
  -v 4
```

Classification with a preceding filtering step:

```
pdal translate  \
    UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz ^
  -o  \
      UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_filt_cl2only.laz  \
      ^
  outlier smrf range  ^
  --filters.outlier.method="statistical" ^
  --filters.outlier.mean_k=8 ^
  --filters.outlier.multiplier=3.0 ^
  --filters.smrf.ignore="Classification[7:7]"  ^
  --filters.range.limits="Classification[2:2]" ^
  --writers.las.compression=true ^
  --verbose 4
```

Applying SMRF classificatiton for ALS:

```
pdal translate  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
  -o  \
      ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_SRMF_cl2only.laz  \
      ^
  smrf range ^
  --filters.range.limits="Classification[2:2]" ^
  -v 4
```

Using LASTools lasground for ground classification:

```
#First, set path on Command prompt (not Anaconda, using  \
    standard command prompt)
set PATH=%PATH%;C:\Software\LASTools\LAStools\bin

#Second, navigate to your LAS/LAZ data directory:
d:
cd D:\PC_Workshop_Oct2019

#Third, apply lasground classification (with standard setting)  \
    on 50-cm ALS PC:
lasground -i  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_50cm.laz  \
    ^
-city -olaz -o  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_50cm_g.laz
```

NOTE: we are using the option `-city` *to ensure that step size for ground-identification is large enough, because of some large building on Campus Golm.*

Apply lasground classification (with standard setting) on full-resolution ALS PC:

```
lasground -i  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
-city -olaz -keep_class 2 -o  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip_g.laz
```

Apply lasground classification (with standard setting) on full-resolution UAV PC:

```
lasground -i UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz ^
-city -olaz -o  \
    UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_g.laz
```

Voxel Classification (Python)

Here, we develop a voxel-based application to classify a lidar or UAV PC. There are two approaches: One optimized for lidar data using last return information (Python Code voxel-classification-lidar.py) and an approach more suitable for SfM/dense point clouds relying on the standard deviation in a voxel (Python Code voxel-classification-zstd). See comments inside the code.