Short Introduction to Point Cloud Processing

In this exercise, we will rely on data from the University of Potsdam (UP) campus Golm data. These are: 1. The airborne lidar data *ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz* 2. The UAV (drone) data from a Mavic Pro: *UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_cl.laz* 3. The UAV (drone) data from an inspire2: *UAV_inspire2_1031cameras_highq_dense_pc_10cm_cl.laz*

These data were distributed during the workshop on a USB memory stick.

PDAL

Use PDAL for pointcloud analysis, filtering, and classification.

Information about LAS/LAZ files

First, obtain some information about the LAS/LAZ file:

```
pdal info --summary  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz

pdal info -p 0  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz
pdal info -p 0-10  \
    ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz
```

Downsampling lidar data

We downsample the data to one point in every 0.25x0.25x0.25 m voxel. This will use the point that is closed to the center of the voxel and will rely on actual lidar points. See 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_voxel25cm.laz  \
      \
  voxelcenternearestneighbor \
  --filters.voxelcenternearestneighbor.cell=0.25
```

Ground detection (SMRF)

There are several algorithms used for ground detection - the Simple Morphological Filter (SMRF) is one of them (described in Pingel et al., 2013).

Simple SMRF filtering

```
pdal translate \
   ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz \
   -o ALS_Golm_cl.laz \
   smrf \
   --writers.las.compression=true --verbose 4
```

In Windows, you will need to replace \ by ^ to continue writing on the next line:

```
pdal translate ^
   ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
   -o ALS_Golm_nonoise_cl2.laz ^
   smrf ^
   --writers.las.compression=true --verbose 4
```

Repeat for the UAV datasets:

```
pdal translate \
   UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm.laz \
   -o UAV_mavicpro2_nadir_15deg_highq_dense_PC_10cm_cl.laz \
   smrf \
   --writers.las.compression=true --verbose 4
```

and

```
pdal translate \
   UAV_inspire2_1031cameras_highq_dense_pc_10cm.laz \
   -o UAV_inspire2_1031cameras_highq_dense_pc_10cm_cl.laz \
   smrf \
   --writers.las.compression=true --verbose 4
```

SMRF with noise filtering

```
pdal translate ^
   ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz ^
   -o ALS_Golm_nonoise_cl.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
```

For Linux/Mac OSX:

```
   pdal translate \
      ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz  \
          \
      -o ALS_Golm_nonoise_cl.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
```

You can use the file `ALS_Golm_nonoise_cl.laz` for ground interpolation.

If you already have your file classified (through `lasground` or through the ClothSimulationFilter in CloudCompare), you can create a file with ground points (class 2) only:

```
pdal translate \
   ALS_Golm_May06_2018_Milan_UTM33N_WGS84_6digit_cl_clip.laz \
   -o ALS_Golm_nonoise_cl2.laz \
   range \
   --filters.range.limits="Classification[2:2]"
```

Creating a DEM and saving a GeoTIFF

We rely on the IDW (Inverse Distance Weighted) Interpolation implemented in writers.gdal. this requires a control file (`.json`) that defines parameters through a pipeline.

Create the file `ALS_Golm_nonoise_cl2_idw.json`:

```
{
   "pipeline": [
       "ALS_Golm_nonoise_cl2.laz",
       {
           "filename":"ALS_Golm_nonoise_cl2_1m.tif",
           "gdaldriver":"GTiff",
           "output_type":"all",
           "resolution":"1.0",
           "window_size": "10",
           "type": "writers.gdal"
       }
```

```
  ]
}
```

Run the pipeline on the command line with:

```
pdal pipeline ALS_Golm_nonoise_cl2_idw.json
```

Alternatively, you can only output the interpolated DEM values with the `idw` interpolation:

```
{
    "pipeline": [
        "ALS_Golm_nonoise_cl2.laz",
        {
            "filename":"ALS_Golm_nonoise_cl2_1m.tif",
            "gdaldriver":"GTiff",
            "output_type":"idw",
            "resolution":"1.0",
            "radius": "10",
            "type": "writers.gdal"
        }
    ]
}
```

Compiled with:

```
pandoc --listings --variable papersize=a4paper \
    -H auto_linebreak_listings.tex \
    --variable urlcolor=blue \
    -V lang=en-GB \
    -s PC_pdal_for_UP_CampusGolm.md \
    -o PC_pdal_for_UP_CampusGolm.pdf
```