

Besprechung 0509

Donnerstag, 9. Mai 2019 07:57

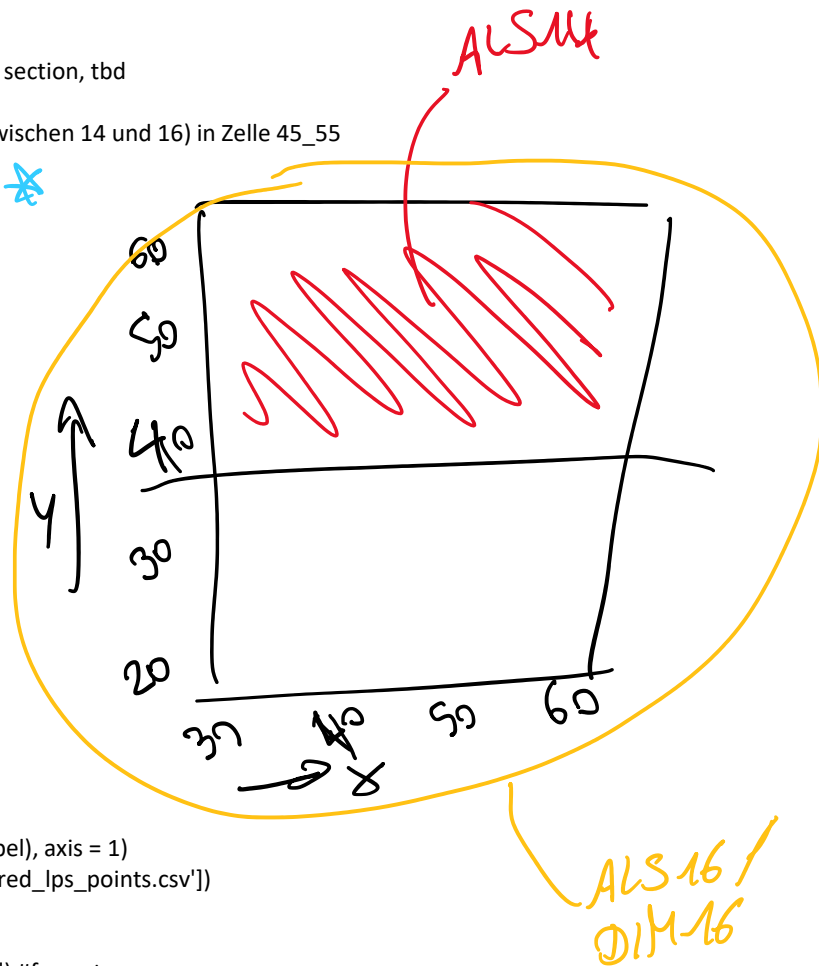
- Normalvektor-Funktionen generell programmiert
- Auswahl der ausschnitte über cloud compare cross section, tbd

Sicherer Unterschied (Gebäude und Bäume abgerissen zwischen 14 und 16) in Zelle 45_55

ALS16: 1km	3331359920	x: 33313000	y: 5992000
DIM16: 500m	33313059920	x: 33313000	y: 5992000

Anfragen von Punktnachbarschaften über einen kDtree (mit XYZ Achsen) und Kugel-Anfragen

ALS-Scanstreifen über die Zeit trennen, da
Ab bestimmter Überlappung einfach von beiden Streifen
Das Ende in die 20er gewandert ist und nicht von einer
alles



Als ASCII-Datei abspeichern:

```
output = np.concatenate((self.points, self.lps, self.label), axis = 1)
filename = "".join([str(seeds), '_', str(sigma), '_clustered_lps_points.csv'])
np.savetxt(self.dir_out + filename, #pfad + name
           output, # numpy array
           fmt = "%.2f %.2f %.2f %.8f %.8f %.8f %.0f %.0f") #format
```



Suchen von Filenamen

```
def get_matching_filenames(filename):
    """
    finds matching filenames from one point cloud to another.
```

Matches ALS16 to DIM16

Inputs:

filename: string; filename of the original file to split

Outputs:

[s1, s2, s3, s4]: list of string;

s1: xmin and ymin

s2: xmin and ymean

s3: xmean and ymin

s4: xmean and ymean

"""

```
# get filenames
```

```
s = filename.split('_')[0]
```

```
# get minimal and mean values
```

```
xmin = int(s[0:5] + '0')
```

```
ymin = int(s[5:])
```

```
xmean = xmin + 5
```

```
ymean = ymin + 5
```

```
# build file strings
```

```

prep = 'DSM_Cloud_'
ending = '.las'
s1 = "".join([prep, str(xmin), '_', str(ymin), ending])
s2 = "".join([prep, str(xmin), '_', str(ymin), ending])
s3 = "".join([prep, str(xmin), '_', str(ymin), ending])
s4 = "".join([prep, str(xmin), '_', str(ymin), ending])
return [s1, s2, s3, s4]

```



kdTree-Beispiel

```

from scipy.spatial import cKDTree
Tree = cKDTree(points, leafsize) #hier sichergehen, dass x, y, z drin ist
_, location = tree.query_ball_point(query_point, radius)
Neighbours = points[location] #nachbarnpunkte innerhalb von radius, XYZ

```

