[1] 2개의 스캔 데이터를 정합 한 경우에 대한 테스트 결과 → distance나 voxel size 이용해서는 uniform 하게 샘플링 하기 어려움.

- voxel size를 바꿔가면서 open3d를 이용하여 downsampling 한 경우

- cloudcompare에서 distance를 바꿔가면서 subsampling 한 경우

- cloudcompare/meshlab에서 점군을 보면 겹쳐진 부분이 확연히 보임 → sampling distance가 커질 수록 밀도차이가 눈으로 보기에 드러나지 않게됨(KDE colomap으로는 보임)

- 밀도함수를 계산해서 (KDE) 가시화 할 경우 distance를 작게, voxel size를 작게 할 수록 밀도 차이가 명확히 보임

- distance/ voxel size를 파라미터로 샘플링 후 밀도함수를 계산할 경우 distance/voxel size에 비례하여 밀도가 blur해지기는 하지만 전반적인 경향은 남아있음

- random sampling을 할경우 distance/voxel size를 파라미터로 할 경우보다는 밀도가 균일해지는 것 같지만 여전히 uniform 하지는 않음

→ 점군을 uniform 하게 샘플링하는 것이 필요할 경우 신경써서 개발해야함!!!

: uniform sampling이 필요한가???? 예를 들어 학습 및 인식이 더 잘된다던지..?? → 논문 주제

→ 일단 학습 데이터를 만들기 위해서 2단계를 제안함

(1) 거리를 파라미터로 해서 2048보다 약간 큰 정도로 샘플링을 함

(2) 1번의 결과에서 점의 갯수를 파라미터로 random sampling해서 갯수를 맞춤

scan01

|  |  |
| --- | --- |
| Cloud compare visualization | Point cloud density (KDE)  25290 points, 12.4 sec for computation |

scan02

|  |  |
| --- | --- |
| Cloud compare visualization | Point cloud density (KDE)  25472 points, 11.6 sec for computation |

meshlab으로 점군만 볼 경우

(1) original : scan01 + scan02

|  |  |
| --- | --- |
|  |  |

(2) subsampling distance 0.2

|  |
| --- |
|  |

(3) subsampling distance 0.5

|  |
| --- |
|  |

(4) subsampling distance 1.0

|  |
| --- |
|  |

KDE 밀도 가시화

(1)

(2) distance 0.2 – 계산 시간 약 440초, 점 갯수 173396)

|  |  |  |
| --- | --- | --- |
|  |  |  |

(3) distance 0.5 (계산시간 약 230초, 점갯수 124556)

|  |  |  |
| --- | --- | --- |
|  |  |  |

() distance 3.0 (계산시간 약 2초, 점갯수 6467)

|  |  |  |
| --- | --- | --- |
|  |  |  |

() distance 5.0 (계산시간 약 0.4초, 점갯수 2576)

|  |
| --- |
|  |

() distance 10 (계산시간 0.06초, 점갯수 729)

|  |
| --- |
|  |

(4) voxel size 0.5 (계산시간 411초, 점 갯수 162774)

|  |  |  |
| --- | --- | --- |
|  |  |  |

(5) voxel size 1.0 (계산시간 159초, 점 갯수 90833)

|  |  |  |
| --- | --- | --- |
|  |  |  |

(6) voxel size 3 (계산시간 3.7초, 점 갯수 13755)

|  |  |  |
| --- | --- | --- |
|  |  |  |

(7) voxel size 5 (계산시간 1.5초, 점 갯수 5382)

|  |
| --- |
|  |

() voxel size 3 계산시간 29초, 점 갯수 38108

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

() voxel size 2 계산시간 129초, 점 갯수 81229 → 보기에 voxel size 3일때과 큰 차이 없음

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

() voxel size 10 계산시간 0.9초, 점 갯수 3918

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

() random sampling

|  |  |  |
| --- | --- | --- |
| 2048 points | 6584 points | 17679 points |
|  |  |  |
|  |  |  |
|  |  |  |

() 밀도 계산 시간

poisson disc sampling

|  |  |
| --- | --- |
|  |  |
|  |  |

참조

<https://jakevdp.github.io/PythonDataScienceHandbook/04.05-histograms-and-binnings.html>

KDE

<https://jakevdp.github.io/PythonDataScienceHandbook/05.13-kernel-density-estimation.html>

scantter plot density

<https://stackoverflow.com/questions/20105364/how-can-i-make-a-scatter-plot-colored-by-density-in-matplotlib>

<https://stackoverflow.com/questions/27156381/python-creating-a-2d-histogram-from-a-numpy-matrix>

<https://opencv-python.readthedocs.io/en/latest/doc/20.imageHistogramEqualization/imageHistogramEqualization.html>

cmap 관련

<https://frhyme.github.io/python-lib/matplotlib_extracting_color_from_cmap/>

poisson disc

<http://www.cemyuksel.com/cyCodeBase/>

<https://codereview.stackexchange.com/questions/217231/speed-up-fast-poisson-disk-sampling-generator-in-python>

<https://scipython.com/blog/poisson-disc-sampling-in-python/>

<https://bl.ocks.org/mbostock/dbb02448b0f93e4c82c3>

<https://github.com/joosm/scipython-maths/tree/master/poisson_disc_sampled_noise>

<https://scipython.com/blog/>

3d poisson disc sampling

<https://lifeisforu.tistory.com/545> (자료 번역)

[C++]

<https://github.com/thinks/poisson-disk-sampling>

<https://www.cs.ubc.ca/~rbridson/>

[C#]

<https://gist.github.com/hiroakioishi/382a6ecbf741c5e0d463>

[python]

<https://github.com/hanbollar/Poisson-Disk-Sampler>

<https://github.com/IHautaI/poisson-disc>

[??? processing]

<https://github.com/njanakiev/poisson-disk-sampling>

[???]

<https://github.com/kchapelier/poisson-disk-sampling>

----

Original

- after\_registration.ply @ ~/Workplace/Point-Cloud-Registration/incremental\_registration\_for\_crashed\_head/

- # of points: 905,852 points

(1) min space between points: 0.5 (05)

- # of points: 397,896 points

(2) min space between points: 1.0

- # of points: 120,480 points

(3) min space between points: 1.5 (1\_5)

- # of points: 61,658 points

(4) min space between points: 2 (2)

- # of points: 37,944 points

(5) min space between points: 2.5 (2\_5)

- # of points: 24,105 points

(6) min space between points: 3.5 (2\_5)

- # of points: 13,349 points

(7) min space between points: 4 (4)

- # of points: 10,454 points

(8) min space between points: 4.5 (4\_5)

- # of points: 8,391 points

(9) min space between points: 5.5 (5\_5)

- # of points: 5,815 points

(10) min space between points: 6 (6)

- # of points: 4,926 points

(11) min space between points: 6.5 (6\_5)

- # of points: 4,226 points

(12) min space between points: 7 (7)

- # of points: 2,628 points

(13) min space between points: 10 (10)

- # of points: 1,911 points

(14) min space between points: 20 (20)

- # of points: 518 points

(15) min space between points: 30 (30)

- # of points: 208 points