**מעבדה בלמידת נתונים במערכות זמן אמת – דו"ח שבועי – 06.08.2023**

המגישים:

דוד ליבר 213910144

מיכאל רודל 326184744

משימות שהשלמנו:

קראנו את המאמר שסיכומו מופיע מטה, וחקרנו את יכולותיה של הספרייה open3D בניקוי רעשים.

משימות לשבוע הבא:

לבצע Camera calibration.

להמשיך לחקור את הספרייה open3D.

לממש את האלגוריתמים במאמר שקראנו לשבוע הזה.

סיכום המאמר:

APA:

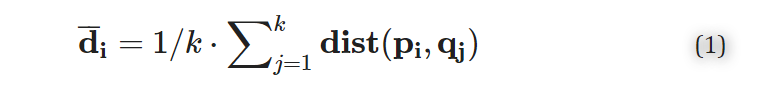
Ning, X., Li, F., Tian, G., & Wang, Y. (2018). An efficient outlier removal method for scattered point cloud data. *PloS one*, *13*(8), e0201280. <https://doi.org/10.1371/journal.pone.0201280>

המאמר הנ"ל מחלק את הoutliers ל-3 קבוצות:

1. מפוזרים.
2. מבודדים.
3. לא מבודדים.

המאמר מציע שיטה אוטומטית להוריד אותם, בהתבסס על צפיפות מקומית, ועל הטלה מקומית.

ניתן לגלות את המפוזרים בקלות, ע"י מציאת צפיפות מקומית (נמוכה). את המבודדים ניתן גם כן לזהות על סמך צפיפות מקומית נמוכה באזור שלהם. עבוד הלא מבודדים, נטיל אותם למישור ע"י ביצוע אלגוריתם PCA.



A math equation with black text

Description automatically generated with medium confidence

A black text on a white background

Description automatically generated

A white background with black text

Description automatically generated

**Algorithm 1** isolated outlier removal algorithm

1: **Input**: Three dimensional scanned PCD with various outliers

2: **Output**: sparse outlier and isolated outlier removal results

3: **for** all point **pi** in **P** do

4:  search the k nearest neighborhood of **pi**, i.e. KNN(**pi**)

5:   calculate the local density **LD**(**pi**) of **pi**

6:    compute the probability pro(**pi**)

7: **end for**

8: sort the pro(**pi**) in ascending order

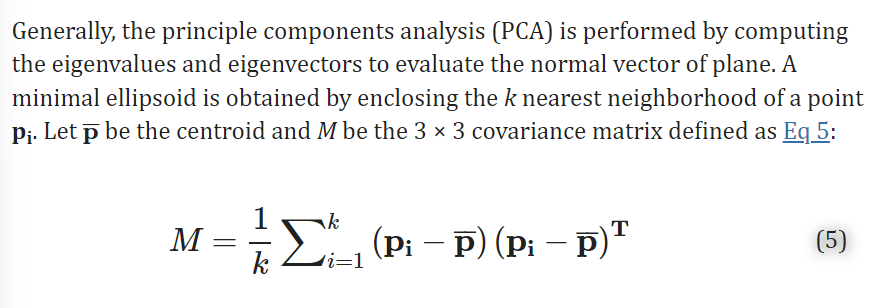
9: compare the first N number of pro(**pi**)

10: **for** i = 1: N

11:  **if** pro(**pi**) > δ

12:  **then** delete **pi** as outliers

13: **end for**

A math equations on a white background

Description automatically generated

**Algorithm 2** Non-isolated outlier removal algorithm

1: **Input**: Three dimensional scanned point cloud data with various outliers

2: **Output**: non-isolated outlier removal

3: **for** all point **pi**(**i** = **1**, **2**, …, **N**) in **P** do

4:  search the k nearest neighborhood **qj**(**j** = **1**, **2**, …, **k**)

5:  fitting a local plane **Li** for **pi** and **qj**

6:  compute the deviation  of **pi**, **qj** from plane **Li**

7:  project **qj** onto the corresponding fitted plane **Li**

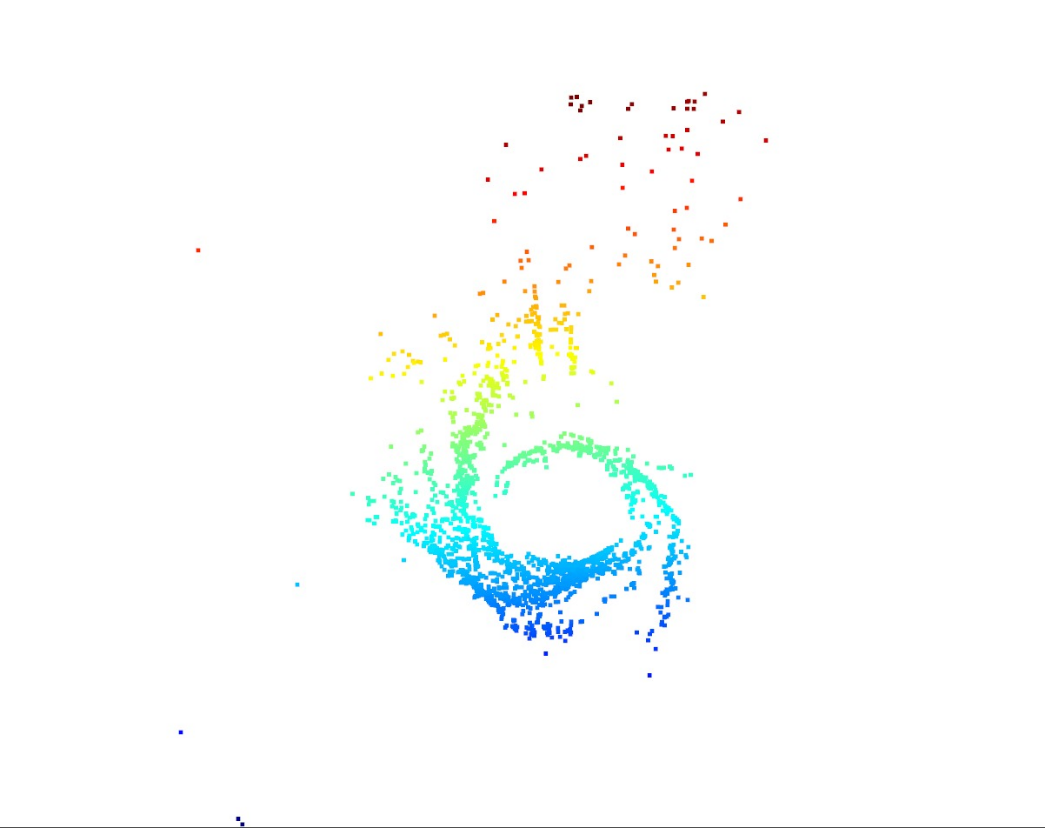
8:  the projected point **qj**′ is the new coordinate of **qj**

9: **end for**

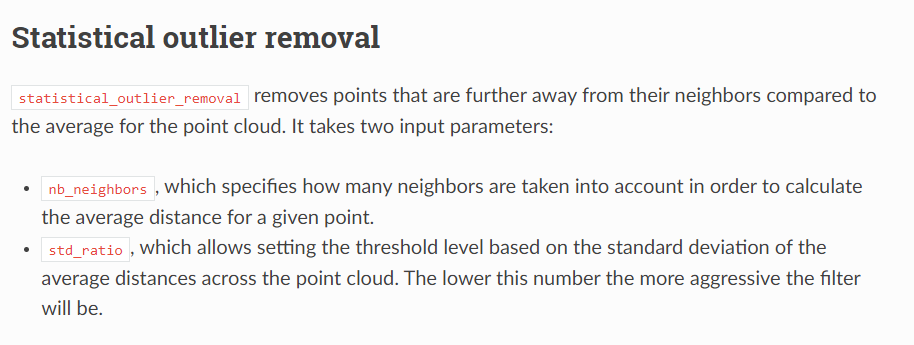
Limitation:  
This method rely on the points distribution and need to calculate the local density of points, our method might fail for very dense outliers exist. When there are denser outliers, it may lead to deformation of object after removing the outliers. To eliminate this problem, we plan to adopt learning-based strategy that could estimate and distinguish the outlier from original points.

כמו כן, ניסינו לנקות את הרעשים מענן הנקודות בעזרת הספרייה open3D. להלן התוצאות:

לפני:



ע"י:



אחרי:

