[Robotique mobile]Laser scan matching through ICP

yu-tianchi

December 2020

1 Introduction

In this project, we want to implement the 2D ICP method for one laser scan process. At first, we use the basic ICP algorithm for the u2is dataset(part of the full dataset), which gives the results as followings.

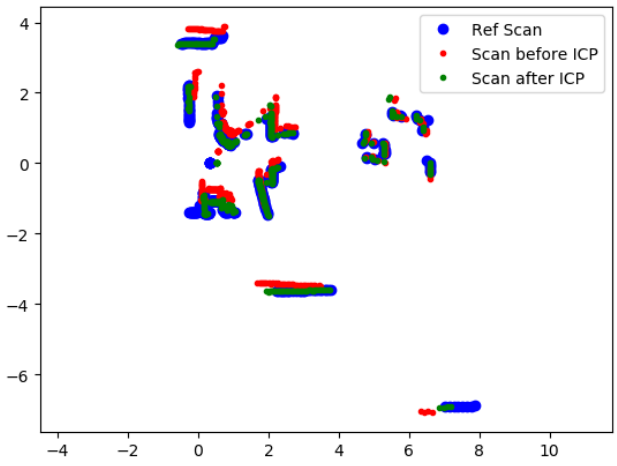


Figure 1: Basic ICP algorithm

Execution consequences

```
Reading u2is dataset
Finished reading 56 scans
Running ICP, finished with mean point corresp. error 0.057834
......
Running ICP, finished with mean point corresp. error 0.060767
Mean (var) translation error : 1.873672e-01 (1.130610e-04)
Mean (var) rotation error : 1.508370e-02 (3.413754e-05)
Mean computation time : 1.025000
```

2 Questions

2.1 Question 1

Implement points filtering (removing points too close to each other in the 'data' scan) and show the consequences (on error, variance, computation time,...) as a function of the parameter values.

Answer:

Add a function of *points_filter(dat,thres_dist)*, to filter those points which are too close to each other in the 'data' scan. *dat* is the original input,*thres_dist* is the parameter we set for the minimum distance between points for the scan. By this function, return the points filtered, then we can have an implementation as following.

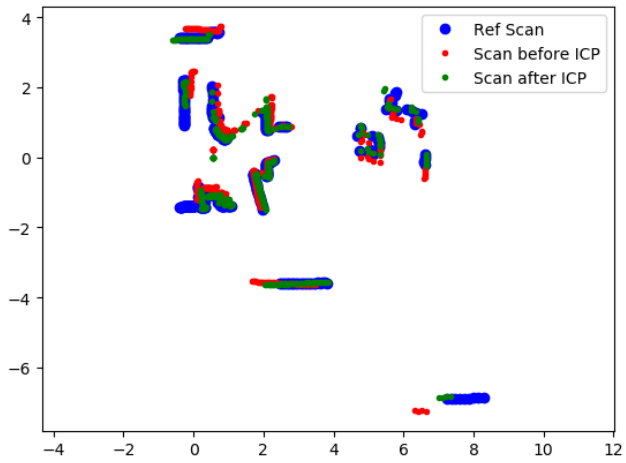


Figure 2: Points filtering ICP algorithm with parameter 0.1

Execution consequences

```
Reading u2is dataset
Finished reading 56 scans
Running ICP, Filtering points which are too close to each other...
count_append is:  124  and count_pass is:  337
finished with mean point corresp. error 0.057332
finished with mean point corresp. error 0.092322
......
finished with mean point corresp. error 0.077847
Mean (var) translation error : 2.026278e-01 (3.828591e-04)
Mean (var) rotation error : 1.713992e-02 (5.611964e-05)
Mean computation time : 0.309375
```

As we can see, the corresponding error augment a little bit, the mean (var) translation error and mean(var) rotation error increase but don't change much. So we can say that with a small value of distance limitation, this step ensure the precision as the original ICP algorithm. While the computation time is much smaller than the basic method because of the less number of points. In the above example, we set the parameter of minimum distance is 0.1, which causes 337 points are ignored. If we set a higher value, we will have a faster implementation but less points and worse precision scan result. Here is a more obvious example of parameter 0.2.

Execution consequences

```
finished with mean point corresp. error 0.104631
Running ICP, Filtering points which are too close to each other...
count_append is:  74  and count_pass is:  387
finished with mean point corresp. error 0.084406
......
finished with mean point corresp. error 0.101717
Mean (var) translation error : 1.997821e-01 (3.046060e-04)
Mean (var) rotation error : 2.011135e-02 (1.664850e-05)
Mean computation time : 0.218750
```

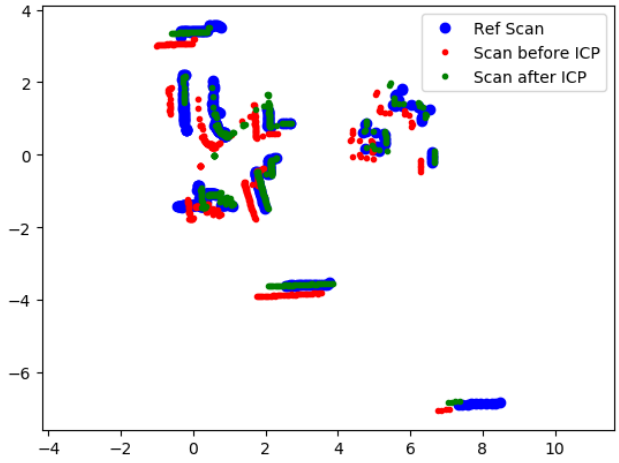


Figure 3: Points filtering ICP algorithm with parameter 0.2

2.2 Question 2

Implement match filtering by keeping the XX% best matching. Try different values for XX and show the consequences (error, variance, computation time,...).

Answer:

Add a function of *match_filter(dat_filt,distance,percent,index)*, to implement match filtering and keep only *precent(%)* best matching pairs. By this function, return the points matched and the new index of reference points, then we can have several implementations as followings with different parameters(based on the basic ICP).

Execution consequences for 30% Match filtering

```
Mean (var) translation error : 3.671927e-01 (4.467065e-02)
Mean (var) rotation error : 1.656817e-01 (8.714250e-03)
Mean computation time : 1.862500
```

Execution consequences for 50% Match filtering

```
Mean (var) translation error : 1.895264e-01 (5.388311e-03)
Mean (var) rotation error : 4.444059e-02 (3.964982e-03)
Mean computation time : 3.960938
```

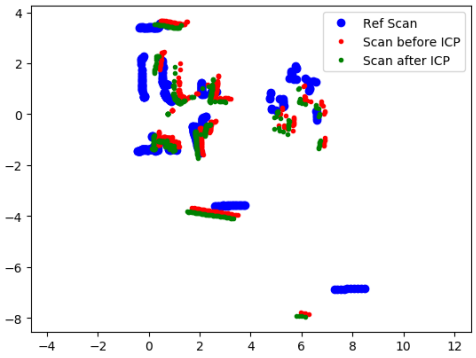
Execution consequences for 70% Match filtering

```
Mean (var) translation error : 1.547089e-01 (6.651129e-06)
Mean (var) rotation error : 4.390699e-03 (8.789329e-06)
Mean computation time : 2.251562
```

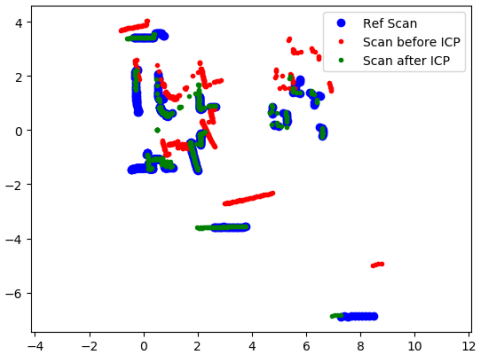
Execution consequences for 80% Match filtering

```
Mean (var) translation error : 1.568548e-01 (1.939461e-06)
Mean (var) rotation error : 4.909669e-03 (3.994120e-06)
Mean computation time : 2.075000
```

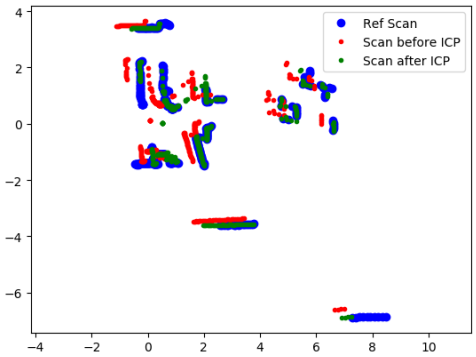
Comparing with the basic ICP implementation, for 70%, 80% matching filtering, the mean error and variance both have a little better performance. While 50% has a worse variance than basic ICP algorithm. With the decrease of the percentage, the computation time gets higher. 30% match filtering warns us that



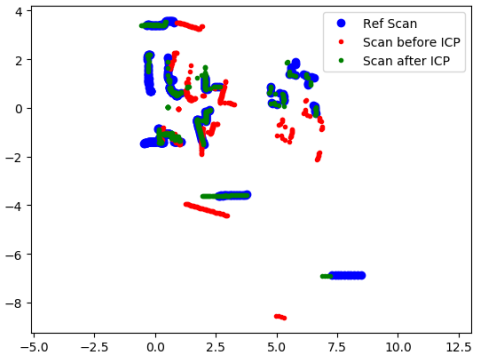
(a) pic1.30% Match filtering



(b) pic2.50% Match filtering



(c) pic3.70% Match filtering



(d) pic4.80% Match filtering

Figure 4: Match filtering ICP algorithm based on basic ICP

if we filter too much points, the performance will become much worse than normal situation, because of the few number of points.

We show the different processes of iteration with different match filtering in fig.5. With a reasonable value, we could get a better performance and a acceptable running time. Note that, a grave filtering would cause oscillations.

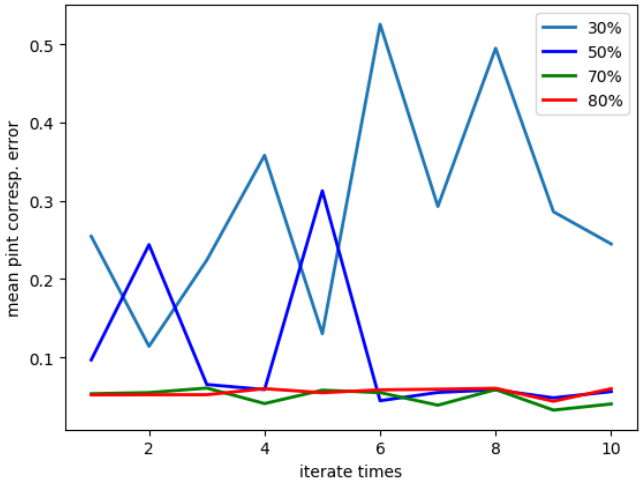


Figure 5: Mean points corresp. error with iterations

Now implement this step based on the first question(filtering points with parameter 0.1), this time we take a more small step(5%) between 50% to 70 %.

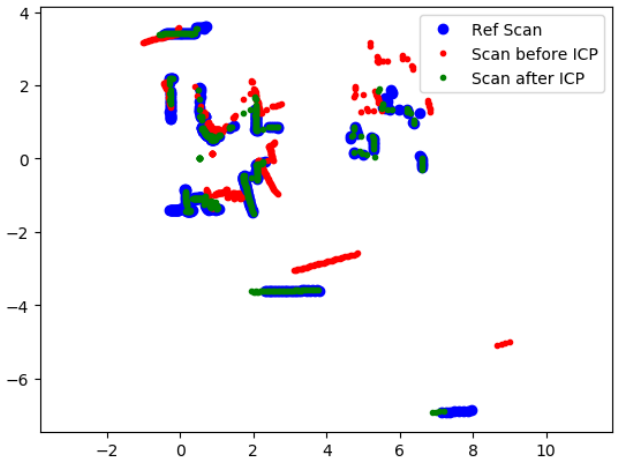


Figure 6: 50% Match filtering ICP algorithm based on filtering points with parameter 0.1

Execution consequences for 50%,55%,60%,65%,70%

```
[50\%]
count_append is:  124  and count_pass is:  337
finished with mean point corresp. error 0.080212
Mean (var) translation error : 1.982634e-01 (5.962255e-03)
```

```
Mean (var) rotation error : 1.585184e-02 (4.791157e-04)
Mean computation time : 0.878125
--------------------------------------------------
[55%]
finished with mean point corresp. error 0.078791
Mean (var) translation error : 1.742197e-01 (3.353524e-03)
Mean (var) rotation error : 4.188468e-02 (5.274543e-03)
Mean computation time : 0.756250
--------------------------------------------------
[60%]
finished with mean point corresp. error 0.070163
Mean (var) translation error : 1.577622e-01 (1.105591e-05)
Mean (var) rotation error : 7.733920e-03 (8.544712e-06)
Mean computation time : 0.682813
--------------------------------------------------
[65%]
finished with mean point corresp. error 0.072106
Mean (var) translation error : 1.564182e-01 (1.671099e-05)
Mean (var) rotation error : 6.508504e-03 (1.257625e-05)
Mean computation time : 0.639062
--------------------------------------------------
[70%]
finished with mean point corresp. error 0.072913
Mean (var) translation error : 1.579099e-01 (1.458793e-05)
Mean (var) rotation error : 6.683521e-03 (1.087575e-05)
Mean computation time : 0.609375
```

The conclusion is that match filtering(with a reasonable percentage, for example 65%) will perform better according to the mean error and variance error. Of course, it takes more time to finish the whole process than the method in the first question.

2.3 Question 3

Use the best setting you found with the icpLocalization function. Analyse visually the quality of the result as a function of the step parameter and of the parameters of the previous filtering. Propose a reasonable compromise between the quality of the localization and the computation time.

Answer:

According to the first and second question, we choose distance parameter as 0.1 and percentage of match filtering as 65% for this question.

Execution consequences for step 50,10,8,5,3

```
[step 50]
--- 9.943429708480835 seconds ---
--------------------------------------------------
[step 10]
--- 36.044450521469116 seconds ---
--------------------------------------------------
[step 8]
--- 36.044450521469116 seconds ---
--------------------------------------------------
[step 5]
--- 72.4792411327362 seconds ---
--------------------------------------------------
[step 3]
--- 125.20578932762146 seconds ---
```

After comparing with different results, it is obvious that the smaller step, the quality of the localization becomes better. At the same time, the computation time will be much larger. **So we propose a reasonable compromise between the quality of the localization and the computation time, which is a step between 5-8**.

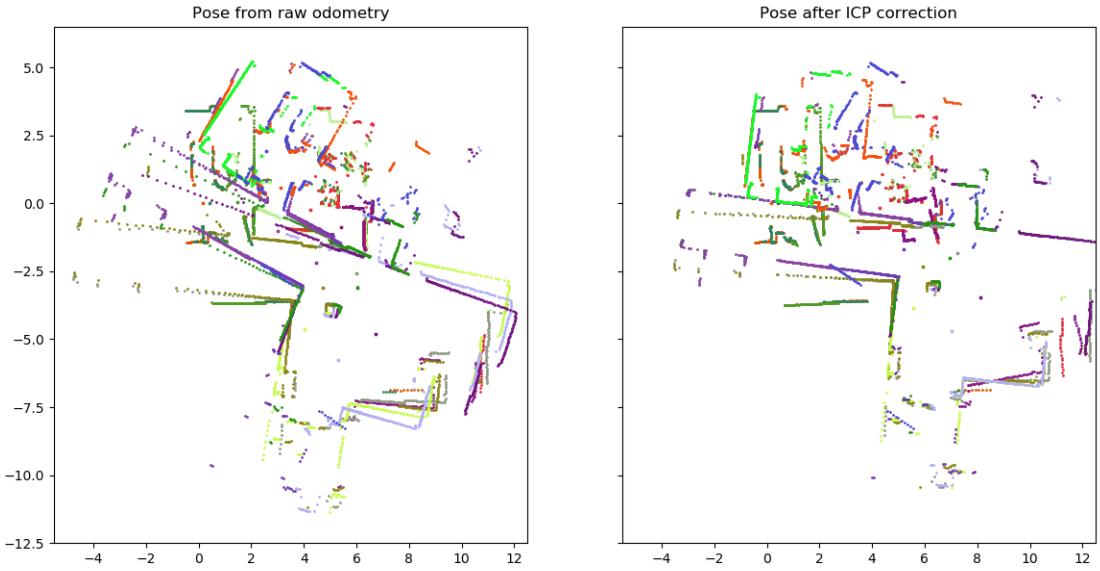


Figure 7: ICPLocalization step 50

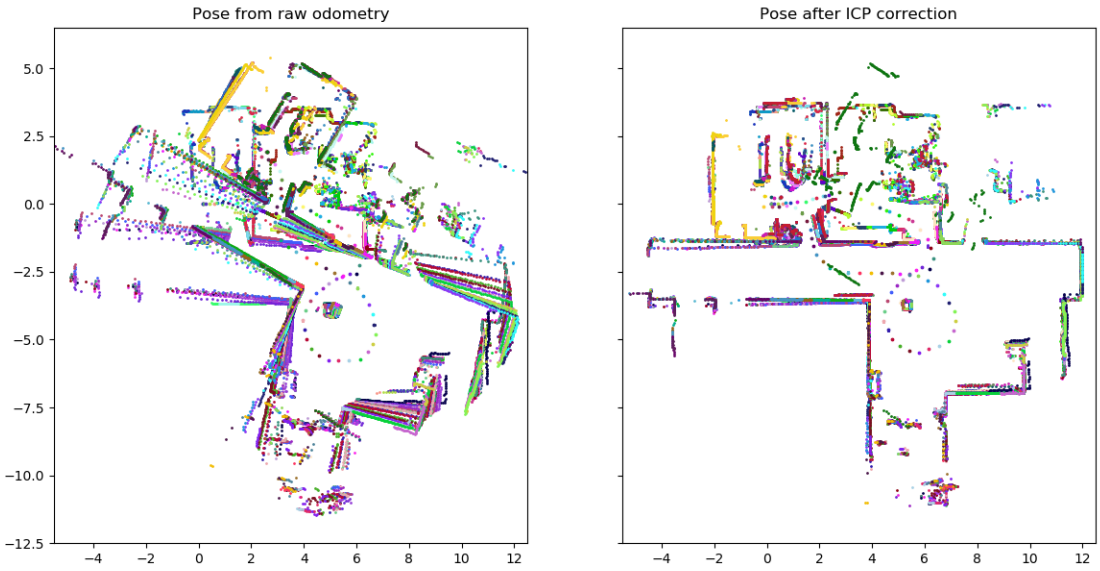


Figure 8: ICPLocalization step 10

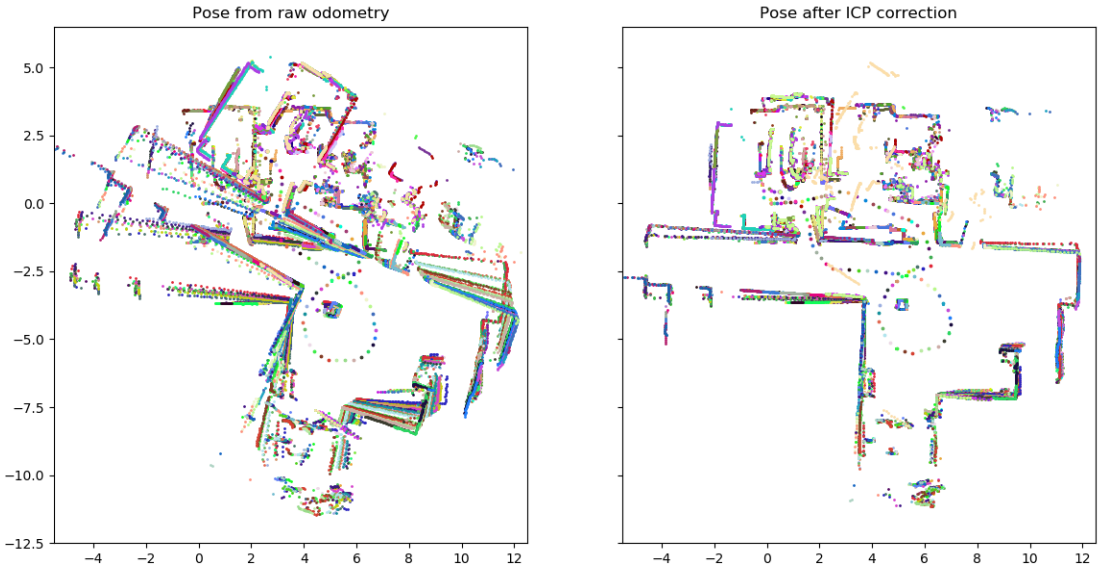


Figure 9: ICPLocalization step 8

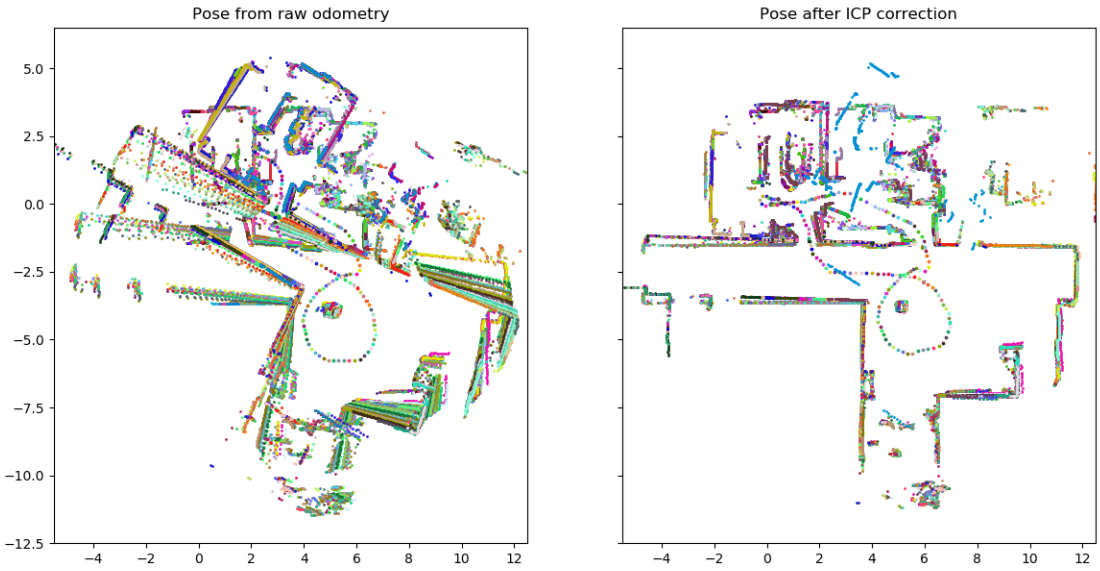


Figure 10: ICPLocalization step 5

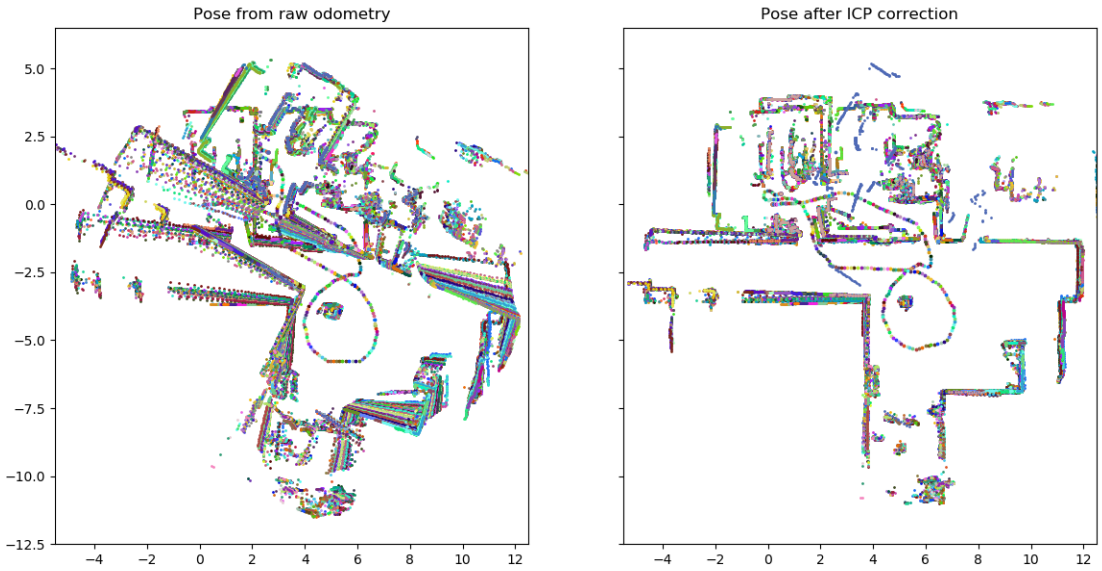


Figure 11: ICPLocalization step 3

2.4 Question 4(Optional)

Implement matching using 'normal shooting' instead of nearest neighbor matching and compare with the previous approach.

Answer: Not solved.