EXERCISE 3: GLOBAL OPTIMIZATION

GCT722 MATHEMATICAL METHODS FOR VISUAL COMPUTING

20183151 Chaelin Kim

PART 1: BRANCH AND BOUND FOR MAXIMUM COVERAGE

Description of implementation

There are 1 script file & 2 function files for the exercise branch and bound.

- **Script file**
  - **main.m**

    : This is the main script that execute branch and bound, find inlier and outlier points, and draw images and convergence plot.

    Branch and bound is done in while loop. At each iteration, take the best candidate in the list, split the space into two children, put them into the list and remove the parent space from the list. Then find the highest lower bound and the lowest upper bound in the list, and update the best bound values with them. After that, remove all the elements in the list that upper bound is lower than current best lower bound. The iterations stop when the current lower bound and upper bound are nearer than 1.

    If the iteration is over, compute inlier and outlier points in the result antenna location and draw the plot on the image.

- **Function files**
  - **makeChilds.m**

    ```
    function [ firstChild, secondChild ] = makeChilds( currentSpace, circleRad, points )
    ```

    : This function is used to split the current space into two children along the longest dimension. It returns first and second child node that lower and upper bound is calculated in space.
  - **calBounds.m**

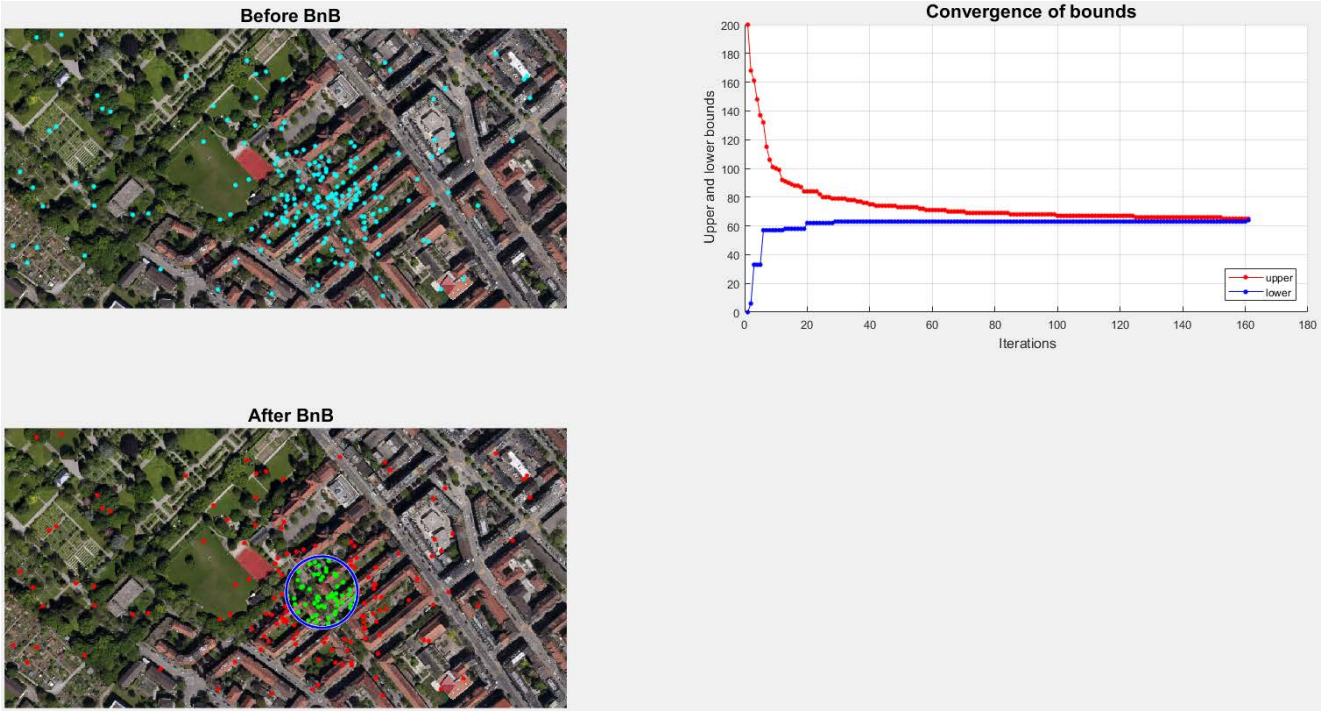
    ```
    function [ lowerBound, upperBound, lowerInliers, upperInliers ] = calBounds( listValue, circleRad, points )
    ```

    : This function is used to calculate the cardinality bounds. For calculating the lower bound, test the model at the center in the current space. For calculating the upper bound, test the extended shape corresponding to the union of all the antenna coverages in the current space. The return values *lowerBound* and *upperBound* are the number of inliers in each, and *lowerInliers* and *upperInliers* are the indices list of inliers in each.

Instructions for running

1. Open the file “main.m” in Matlab.
2. Execute that file.
3. The window that shows the result images and graph is opened.
    a. The left part shows result images of applying BnB before and after.
    b. The right part shows the result plot of convergence of bounds

Screenshots



The results of the antenna location

- (x, y) = (623.5918, 322.0938)

Indices of the inliers and outliers

Inliers			
2	58	100	152
4	62	102	153
8	76	105	156
10	80	106	158
13	87	110	161
14	89	111	164
26	90	114	165
30	92	116	167
33	93	120	168
39	94	121	170
42		122	171
44		125	179
45		129	182
46		132	184
		133	188
		137	189
		139	192
		142	193
		143	194
			198
			200

Outliers			
1	50	101	150
3	51	103	151
5	52	104	154
6	53	107	155
7	54	108	157
9	55	109	159
11	56	112	160
12	57	113	162
15	59	115	163
16	60	117	166
17	61	118	169
18	63	119	172
19	64	123	173
20	65	124	174
21	66	126	175
22	67	127	176
23	68	128	177
24	69	130	178
25	70	131	180
27	71	134	181
28	72	135	183
29	73	136	185
31	74	138	186
32	75	140	187
34	77	141	190
35	78	144	191
36	79	145	195
37	81	146	196
38	82	147	197
40	83	148	199
41	84	149	
43	85		
47	86		
48	88		
49	91		
	95		
	96		
	97		
	98		
	99		

Discuss the results

I have implemented coverage maximization by branch and bound where the model to find is the (x, y) 2D position of the antenna. The upper and lower bounds are converged at **iteration 161**. The result of the antenna location is **(623.5918, 322.0938)** and the number of inliers is **64** (the iterations stop when the lower and upper bound are nearer than 1). The execution time is about 2 secs, so it finds the optimal solution quite fast.

The branch and bound is methods for global optimization problems, so normally it is slower than methods for local optimization problems. In addition, if the points are comparatively scattered, it will search most spaces. So the execution time depends on the number of data points and how data points are scattered. But the branch and bound method is faster than exhaustive search, because in this method we cannot search the space estimated there is no optimal solution. Also, it guarantees the optimal solution because it considers global objective values.