TOPICS IN VIRTUAL REALITY: MATHEMATICAL METHODS FOR VISUAL COMPUTING

EXERCISE 3 - GLOBAL OPTIMIZATION

Handout date: 22 October 2018
Submission deadline: 5 November, 17:59
Demo date: TBA

General Rules

Plagiarism note. Copying code (e.g. from other students, external sources and previous years) is strictly prohibited! We will be using automatic anti-plagiarism tools, and any violation of this rule will lead to expulsion from the class.

Late submissions will not be accepted, except in case of serious illness or emergency. In that case please notify the teaching assistants.

Software. All exercises of this course use the MATLAB programming language. The MATLAB distribution is available from `http://kftp.kaist.ac.kr/` for KAIST students. See our MATLAB tutorial slides for hints or specific functions that could be useful for your implementation.

What to hand in. Send a .zip file of your solution by email to the TA and lecturer (as a download link, **not** in attachment). The emails will be processed automatically. The *zip file* must be called "MathMethods18-Ex*-StudentID-firstname-familyname.zip" (replace * with the assignment number, e.g. MathMethods18-Ex01-20180503-John-Smith.zip). The *email subject* must be the same (except ".zip") (e.g. MathMethods18-Ex01-20180503-John-Smith). The .zip file MUST contain *a single folder* called the same as the email subject (e.g. "MathMethods18-Ex01-20180503-John-Smith") with the following data inside:

- A folder named "code" containing your MATLAB code
- A README file (in pdf format) containing a description of what you have implemented and instructions for running it, as well as explanations/comments on your results.
- Screenshots of all your results with associated descriptions in the README file.

Grading. The homeworks count for 50% of your total grade. Your homework submission will be graded according to the quality of the images/results produced by your program, the conformance of your program to the expected behaviour of the assignment, and your understanding of the underlying techniques used in the assignment. The submitted code must produce exactly the same images included in your submission.

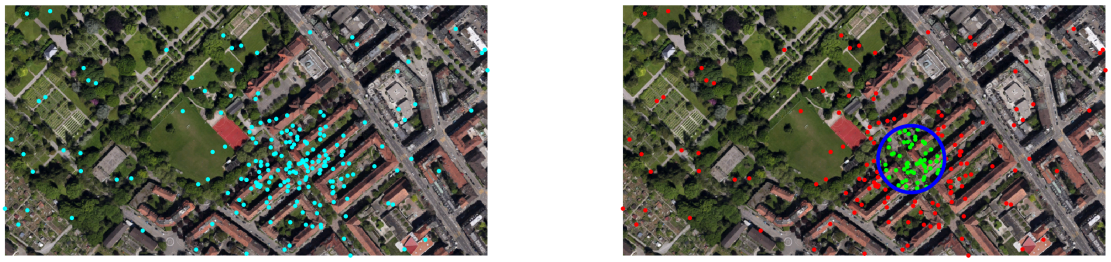
To ensure fairness of your grade, you will be asked to briefly present your work to the teaching assistants. Each student will have about 5 minutes to demo their submission and explain in some detail what has been implemented, report potential problems and how they tried to go about solving them, and point the teaching assistants to the code locations where the various key points of the assignments have been implemented. See above for the scheduled demo date for this particular assignment.

Goal of this exercise

In this exercise you will apply what you learned about global optimization, especially branch and bound (B&B), and geometric bounds. You will implement, in Matlab, branch and bound for maximizing an antenna coverage in an image.

1. EXERCISE: branch and bound for maximum coverage

Context and goal. In the context of maximum coverage in an image, you will implement coverage maximization (e.g. antenna location optimization) by branch and bound where the model to find is the (x, y) 2D position of the antenna. The antenna coverage radius is given. We provide the input image, and a set of 2D points, for example corresponding to people or home locations (i.e. that need antenna connection) (see Figure 1(a)). Given this input set of points, the goal is to find the antenna location maximizing the consensus set (i.e. maximize the number of inlier points, that is the number of covered people), identify the inlier and outlier clustering (i.e. which person is covered or not), and estimate the model (i.e. the 2D location of the antenna). The antenna coverage is modeled as a disk, i.e. the goal is to find the disk location (of a given radius) covering the highest number of points. The identified inliers and outliers, as well as the optimized antenna location, are shown in Figure 1(b).



(a) (b)

Figure 1. (a) input points, (b) the clustering of inliers (in green) and outliers (in red), as well as the antenna location and coverage (in blue), obtained by branch and bound.

As seen in the class, the consensus set maximization problem can be formulated as follows. Let the set S of the input data be partitioned into an inlier-set $S_I \subseteq S$ and an outlier-set $S_O = S \setminus S_I$. The cardinality of S_I corresponds to the number of detected inliers. Let q_i represent the i-th data point/sample/information/feature, and Θ be the unknown underlying model. The general consensus set maximization can be mathematically formulated as:

$$\max_{\Theta, S_I} \quad \text{card}(S_I) \tag{1a}$$

$$\text{s.t.} \quad f(\Theta, q_i) \leq \delta, \forall i \in S_I \subseteq S \tag{1b}$$

This formulation simply states that, given a residual tolerance δ (the inlier threshold), the goal is to find the largest consensus set (i.e. maximize the number of inliers) under a model Θ and to estimate this model.

In our context of antenna coverage in image, we note:

- the model $\Theta = C = (x, y)$ which corresponds to the 2D location of the antenna.
- the i-th input 2D point $p_i = (x_i, y_i)$. We have n input points, i.e. $i = 1, \ldots, n$.
- the cost function $f()$ measures the distance to the antenna (the disk center) and the inlier threshold δ (or noted R) is the given antenna coverage radius.

A point p_i is considered inlier if it is inside the antenna coverage, i.e. inside the disk of radius R. Our consensus set maximization problem can now be formulated as:

$$\max_{C, S_I} \quad \text{card}(S_I) \tag{2a}$$

$$\text{s.t.} \quad \|p_i - C\| \leq R, \forall i \in S_I \subseteq S \tag{2b}$$

What we provide. We provide you:

- the input image, see *I*.
- the list of input points (composed of inliers and outliers), see *ListInputPoints*. Its size is $n \times 2$ where n is the number of input points. The i-th row contains the coordinates of the i-th point: (x_i, y_i).
- the antenna coverage radius, see *CircleRadius*.

All these variables are contained in the .mat file *InputData.mat*. You can load them by:
load('InputData.mat','ListInputPoints','CircleRadius','I')

Required deliverables

You will provide the following deliverables:

- The Matlab code implementing branch and bound for consensus set maximization, in the context of antenna coverage optimization in an image.
- A small report (1-3 pages) containing the following results:
  - The results of the antenna location (i.e. the values of x and y), and the indices of the inliers and outliers obtained by branch and bound.
  - A figure showing the identified inlier and outlier correspondences, like Figure 1(b)
  - A figure showing the convergence of the cardinality bounds, i.e. the highest lower bound obtained so far, and the highest upper bound still in the list. See Fig. 2.

2. Hints and Notation

Here is some additional information to help you with branch and bound:

- The model Θ is the 2D location (x, y) of the antenna. Let's note the lower and upper definition ranges of Θ as $\underline{x}, \overline{x}, \underline{y}, \overline{y}$.
- Given a space of Θ (i.e. $x \in [\underline{x}, \overline{x}]$ and $y \in [\underline{y}, \overline{y}]$)

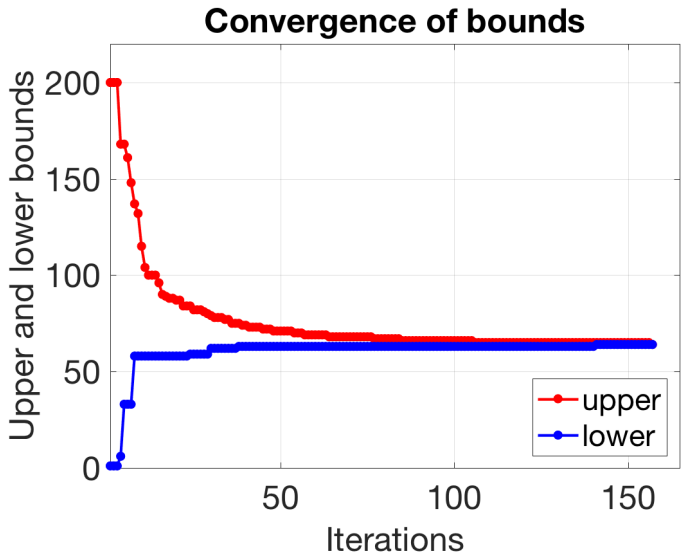


FIGURE 2. Evolution of the lower and upper bounds of the number of inliers along the branch and bound iterations.

  - Compute the higher bound of the nb of inliers by testing an "extended shape" corresponding to the union of all the antenna coverages in the current space.
  - Compute the lower bound of the nb of inliers by simply testing the model Θ at a specific Θ in the space, for example the one at the center, i.e. test $\Theta = (x, y) = \big((\underline{x} + \overline{x})/2, (\underline{y} + \overline{y})/2\big)$.
- Conduct a depth-first search: at each iteration take the best candidate in the list, split it into two children, compute their cardinality bounds, and remove all the elements in the list with a "bad" bound. When you take the best candidate, remove it from the list, and put its two children into the list
- Iterate and remove the spaces that definitely do not contain the optimal solution. For example, let m^* be the highest lower bound of the number of inliers obtained so far. If the upper cardinality bound of a space is less than m^*, it can be safely removed, because even in the best case it cannot contain a better solution.
- When branching, split the current space into two children subspaces in half along the longest dimension
- The iterations stop when the lower and upper bound are nearer than 1, because they will lead to the same integer number of inliers

We will not grade the execution speed. Typical execution takes around 2-20seconds depending on the implementation quality.