**Computational Geometry Course – spring 2012, IDC**

EX3 –Triangulation and Guarding

Submitted by:

* Inna Katz (304206089)
* Daniel Shmuglin (305870636)
* Alexander Vainshtein (312717218)

## Worst case for a greedy algorithm

The worst case for greedy algorithm we could think about is the following:



* + There are 6 diamond
  + There are 2 guardians G1 and G3 which can see 3 (different) diamonds each one, i.e. together they see all the diamonds
  + There is an additional guardian G2 which can see 4 diamond – 2 diamond guarded by G1 and another two guarded by G3
  + The optimal solution is to take 2 guardians G1 and G3
  + The greedy one will select first G2 as he sees more diamonds than any other guardian. Then G1 and G2 will be selected as well as each one of them guards a diamond which is not guarded by any other guardian.
  + So the relation between the greedy solution and the optimal one is 3:2 (we can repeat the same pattern as much as we want).

## Our algorithm for finding the minimal set of guards

1. For each guardian count number of diamonds he can see
2. Select first all the guardians which “exclusively” guards any diamond, i.e. no other guardians can see those diamond
3. Remove all the selected diamonds from the diamonds group and update the number of “guarded” diamonds for the relevant guardians
4. Stop if all the diamonds group is empty
5. Find the diamond which is being guarded by the least number of the guardians
6. Greedy select the guardian from the group found in 4, i.e. take the guardian which can see the most diamonds
7. Go to step 3

Implementation details:

We implemented our solution using Java programming languages (language level complies with version 1.6).

Additional libraries used:

* Google-guava (<http://code.google.com/p/guava-libraries/>)
* JDT (Java Delaunay Triangulation)
* Apache Commons Lang (<http://commons.apache.org/lang/>)
* Java3D (<http://java3d.java.net/>)

Submission & usage:

Our submission zip archive contains:

* ex2.docx (this file)
* binaries.zip – implementation binaries - unpack into a directory and run from command shell java –jar ex2.jar for usage:  
  
* sources.zip – implementation sources (also available at GitHub <https://github.com/hugebdu/ex2>)

The full solution archive can be found here - <https://github.com/hugebdu/ex2/raw/master/ex2.zip>