# Performance test for CGAL and jdt triangulation

Test performed on the following machine:

AMD Athlon 64 X2 dual

2.21 Ghz, 1.00 GB RAM

With Windows XP 32

CGAL project source files:

CgalTriangulationTest.cpp – contains ‘main’

TriangulationImpl.h – class that performs test operations on a CGAL triangulation

TriangulationImpl.cpp – implementation of TriangulationImpl.h

Triangulation\_2 has been used for CGAL testing because the jdt ignores the z values anyway.

## Test1. Random points triangulation construction

It seems that CGAL has much better performance when collecting points in std::vector and passing iterators as parameters of the insert function.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of points | CGAL one-by-one | CGAL using vector iterator | Jdt | CGAL/jdt % |
| 10000 | 2265 | 78 | 360 | 21% |
| 30000 | 18672 | 297 | 1766 | 17% |
| 50000 | 42016 | 468 | 3969 | 12% |
| 100000 | 146626 | 1172 | 10266 | 11.4% |

## Test2. Reading tsin and writing smf

On this test we read tsin file and write smf file. I used the standard << operator in order to print CGAL triangulation into ofstream and therefore the file syntax is a little different from the jdt smf.

It seems that the jdt I/O worked better than the c++ one.

|  |  |  |  |
| --- | --- | --- | --- |
| Number of points | CGAL one-by-one | CGAL using vector iterator | Jdt |
| 1000 | 94 | 80 | 125 |
| 5000 | 344 | 344 | 250 |
| 13000 | 7078 | 984 | 563 |