2D Convex Hull in CGAL

Pierre Alliez

http://www.cgal.org
Outline

- Definitions
- Convex hull in CGAL
  - Interface
  - Extreme points
  - Subsequences of hull points
  - Convexity Checking
- Exercise
Definitions

• A subset $S \subseteq \mathbb{R}^2$ is **convex** if for any two points $p$ and $q$ in the set the line segment with endpoints $p$ and $q$ is contained in $S$.

• The **convex hull** of a set $S$ is the smallest convex set containing $S$.

• The **convex hull of a set of points $P$** is a convex polygon with vertices in $P$. 

http://www.cgal.org
Definitions

- A point in $P$ is an **extreme** point (with respect to $P$) if it is a vertex of the convex hull of $P$.
- A point set is said to be **strongly convex** if it consists of only extreme points.
Convex Hull in CGAL

5 algorithms to compute the counterclockwise sequence of extreme points for a 2D point set.
Convex Hull in CGAL

$n$ input points with $h$ extreme points

Algorithms:

- [Bykat 78] $O(nh)$ output-sensitive
- [Akl & Toussaint] $O(n \log n)$ worst case
- [Graham-Andrew] $O(n \log n)$
- [Jarvis 73] $O(nh)$
- [Eddy] $O(nh)$
Interface

template <class InputIterator, class OutputIterator>
OutputIterator convex_hull_2(InputIterator first,
    InputIterator beyond,
    OutputIterator result,
    Traits ch_traits = Default_traits)

generates the counterclockwise sequence of extreme points of
the points in the range \([first,beyond]\). The resulting sequence
is placed starting at position result, and the past-the-end
iterator for the resulting sequence is returned (it is not
specified at which point the cyclic sequence of extreme points
is cut into a linear sequence).
template <class InputIterator, class OutputIterator> 
OutputIterator convex_hull_2(InputIterator first,
    InputIterator beyond,
    OutputIterator result,
    Traits ch_traits =
    Default_traits)

InputIterator::*value_type 
OutputIterator::*value_type 

Traits contains types and functions:
    Traits::*Point_2, Traits::*Less_xy_2, Traits::*Less_yx_2,
    Traits::*Leftturn_2.
```cpp
#include <CGAL/Cartesian.h>
#include <CGAL/convex_hull_2.h>
#include <list>

typedef CGAL::Cartesian<double> Kernel;
typedef Kernel::Point_2 Point;

std::list<Point> points;
std::list<Point> hull;
points.push_back(Point(0,0));
points.push_back(Point(0,1));
// ...
CGAL::convex_hull_2(points.begin(),
    points.end(),
    std::inserter(hull,hull.begin()));
```
#include <CGAL/Cartesian.h>
#include <CGAL/graham_andrew.h>
#include <list>

typedef CGAL::Cartesian<double> Kernel;
typedef Kernel::Point_2 Point;

std::list<Point> points;
std::list<Point> hull;
points.push_back(Point(0,0));
points.push_back(Point(0,1));
// ...  
CGAL::ch_graham_andrew(points.begin(),
    points.end(),
    std::inserter(hull,hull.begin()));
Extreme Points

- North, South, East, West and combinations.
SubSequences of Hull Points

- Lower Hull
- Upper Hull
#include <CGAL/convexity_check_2.h>

template <class ForwardIterator, class Traits>
bool is_ccw_strongly_convex_2 (  
    ForwardIterator first,  
    ForwardIterator beyond,  
    Traits ch_traits = Default_traits)

returns true iff the point elements in [first,beyond) form a  
counterclockwise-oriented strongly convex polygon.
Exercise on Computer