Geometry for Programming Competitions

Eugene Fink (www.cs.cmu.edu/~eugene)

Available online:
- Word version: www.cs.cmu.edu/~eugene/research/talks/compete-geom.doc
- PDF version: www.cs.cmu.edu/~eugene/research/talks/compete-geom.pdf

Let no one who is ignorant of geometry enter here.
— Incription on the entrance to Plato’s academy

We humans are good at visual reasoning, but computers are not; intuitive visual operations are often hard to program.

Overview

- Basic geometry: points and lines, triangle, circle, polygon area
- Convex hull: gift wrapping, Graham scan

Readings:
- Steven S. Skiena and Miguel A. Revilla. Programming Challenges
- Joseph O’Rourke. Computational geometry in C
- Wikipedia (wikipedia.org)
- Wolfram MathWorld (mathworld.wolfram.com)

Points and lines

Representation in Cartesian coordinates:
- Point: \((x_1, y_1)\)
- Line:
  - \(y = m \cdot x + b\) (more intuitive but does not include vertical lines)
  - \(a \cdot x + b \cdot y + c = 0\) (more general but non-unique and less intuitive)
- Line segment: two endpoints

Yes/No tests:
- Point \((x_1, y_1)\) is on line \((m, b)\) iff \(y_1 = m \cdot x_1 + b\) (available in Java)
- Lines \((m_1, b_1)\) and \((m_2, b_2)\) are...
  - identical iff \(m_1 = m_2\) and \(b_1 = b_2\)
  - parallel iff \(m_1 = m_2\)
  - orthogonal iff \(m_1 = -1 / m_2\)
- Counterclockwise predicate \(ccw\) (available in Java):
  Consider points \((x_1, y_1), (x_2, y_2),\) and \((x_3, y_3),\)
  and let \(A = x_1 \cdot y_2 + x_2 \cdot y_3 + x_3 \cdot y_1 - y_1 \cdot x_2 - y_2 \cdot x_3 - y_3 \cdot x_1.\)
  - If \(A > 0\), the points are in a counterclockwise order
  - If \(A < 0\), the points are in a clockwise order
  - If \(A = 0\), the points are collinear
Basic computations:

- Distance between \((x_1, y_1)\) and \((x_2, y_2)\):
  \[
  \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}
  \]
- Line through points \((x_1, y_1)\) and \((x_2, y_2)\):
  \[
  m = \frac{(y_1 - y_2)}{(x_1 - x_2)}
  \]
  \[
  b = y_1 - m \cdot x_1
  \]
- Intersection of \((m_1, b_1)\) and \((m_2, b_2)\), where \(m_1 \neq m_2\):
  \[
  x_1 = \frac{(b_2 - b_1)}{(m_1 - m_2)}
  \]
  \[
  y_1 = m_1 \cdot x_1 + b_1
  \]
- Angle between \((m_1, b_1)\) and \((m_2, b_2)\):
  \[
  \arctan \left(\frac{(m_2 - m_1)}{(m_1 \cdot m_2 + 1)}\right)
  \]

Other useful operations:

- Intersection of two segments (available in Java)
- Distance from a point to a line or a segment (available in Java)
- Point on a line closest to a given point

Triangle

A triangle specification may include three sides and three angles. Given three of these six values, we can find the other three.

- \(\alpha + \beta + \gamma = \pi\) (or 180°)
- Law of sines:
  \[
  \frac{a}{\sin \alpha} = \frac{b}{\sin \beta} = \frac{c}{\sin \gamma}
  \]
- Law of cosines (generalized Pythagorean theorem):
  \[
  c^2 = a^2 + b^2 - 2 \cdot a \cdot b \cdot \cos \gamma
  \]
- Law of tangents (less useful):
  \[
  \frac{(a - b)}{(a + b)} = \frac{\tan \left(\frac{(\alpha - \beta)}{2}\right)}{\tan \left(\frac{(\alpha + \beta)}{2}\right)}
  \]

Area:

- \(\text{Area} = a \cdot b \cdot \sin \gamma / 2\)
- \(\text{Area} = | x_1 \cdot y_2 + x_2 \cdot y_3 + x_3 \cdot y_1 - y_1 \cdot x_2 - y_2 \cdot x_3 - y_3 \cdot x_1 | / 2\)
- Heron’s formula:
  Let \(s = (a + b + c) / 2\); then \(\text{Area} = \sqrt{s \cdot (s - a) \cdot (s - b) \cdot (s - c)}\)
- Less useful:
  \(\text{Area} = r \cdot s = a \cdot b \cdot c / (4 \cdot R)\), where \(r\) is inradius and \(R\) is circumradius

Other useful operations:

- Centers of inscribed and circumscribed circles
Circle

- A circle is the set of points located at a given distance from a given center.
- A disk is the set of points located no further than a given distance from a given center.

Representation:
Let \( r \) be the radius and \((x_c, y_c)\) by the center.
- Circle: \((x - x_c)^2 + (y - y_c)^2 = r^2\)
- Disk: \((x - x_c)^2 + (y - y_c)^2 \leq r^2\)

Basic computations:
- Circumference: \(2 \cdot \pi \cdot r\)
- Area: \(\pi \cdot r^2\)
- Point \((x_1, y_1)\) is inside the circle iff \((x_1 - x_c)^2 + (y_1 - y_c)^2 \leq r^2\)

Other useful operations:
- Intersection of a circle and a line
- Intersection of two circles
- Tangent to a circle through a given point

Polygon area

- Based on vertex coordinates:
  \[|x_1 \cdot y_2 + x_2 \cdot y_3 + \ldots + x_n \cdot y_1 - y_1 \cdot x_2 - y_2 \cdot x_3 - \ldots - y_n \cdot x_1| / 2\]
- Pick’s formula for a polygon on a lattice:
  If a polygon has \(i\) lattice points inside and \(b\) lattice points on the boundary, its area is \(i + b/2 - 1\)

Convex hull

The convex hull of a geometric object is the minimal convex set containing the object. Intuitively, it is a rubber band pulled taut around the object.
- The hull of a finite point set is a convex polygon.
- The hull of a set of polygons is identical to the hull of their vertices.
- A polygon is convex (i.e. identical to its hull) iff all its angles are at most \(\pi\) (180°).
Gift wrapping (a.k.a. Jarvis march)

Intuitively, wrap a string around nails; simple but slow. Time complexity is $O(n \cdot h)$, where $n$ is the number of points and $h$ is the number of hull vertices.

- Select the smallest-$y$ point as the first hull vertex; if several, choose the largest-$x$ point among them
- At each step, select the next hull vertex, which is the “rightmost” as seen from the previously selected vertex


Graham scan

Efficient version of the gift wrapping. Time complexity is $O(n \cdot \lg n)$.

- Select the smallest-$y$ point as the first hull vertex; if several, choose the largest-$x$ point among them
- Sort the other points right-to-left, as seen from the selected vertex
- Walk through the points in the sorted order; when making the “right turn,” prune the respective point
- The remaining points are the hull vertices