Welcome to...

Convex Hell

...er, that’s

Convex HULL

• Convexivity
• Package-Wrap Algorithm
• Graham Scan
• Dynamic Convex Hull
What is the Convex Hull?

Let $S$ be a set of points in the plane.

**Intuition:** Imagine the points of $S$ as being pegs; the *convex hull* of $S$ is the shape of a rubber-band stretched around the pegs.

**Formal definition:** the *convex hull* of $S$ is the smallest convex polygon that contains all the points of $S$. 
Convexity

You know what *convex* means, right?

A polygon $P$ is said to be *convex* if:
1. $P$ is non-intersecting; and
2. for any two points $p$ and $q$ on the boundary of $P$, segment $pq$ lies entirely inside $P$
Why Convex Hulls?

Who cares about convex hulls?

I don’t … … but robots do!

shortest path avoiding the obstacle
The Package Wrapping Algorithm
Package Wrap

• given the current point, how do we compute the next point?
• set up an orientation tournament using the current point as the anchor-point...
• the next point is selected as the point that beats all other points at CCW orientation, i.e., for any other point, we have

\[ \text{orientation}(c, p, q) = \text{CCW} \]
Time Complexity of Package Wrap

• For every point on the hull we examine all the other points to determine the next point
• Notation:
  • $N$: number of points
  • $M$: number of hull points ($M \leq N$)
• Time complexity:
  • $\Theta(MN)$
• Worst case: $\Theta(N^2)$
  • all the points are on the hull ($M=N$)
• Average case: $\Theta(N \log N) — \Theta(N^{4/3})$
  • for points randomly distributed inside a square, $M = \Theta(\log N)$ on average
  • for points randomly distributed inside a circle, $M = \Theta(N^{1/3})$ on average
Package Wrap has worst-case time complexity $O(N^2)$

Which is bad...

But in 1972, Nabisco needed a better cookie - so they hired R. L. Graham, who came up with...
The Graham Scan Algorithm

Rave Reviews:

• “Almost linear!”
  - Sedgewick
• “It’s just a sort!”
  - Atul
• “Two thumbs up!”
  - Siskel and Ebert
• Nabisco says...

  “A better crunch!”

and history was made.
Graham Scan

• Form a simple polygon (connect the dots as before)

• Remove points at concave angles
Graham Scan
How Does it Work?

Start with the lowest point (anchor point)
Graham Scan: Phase 1

Now, form a closed simple path traversing the points by increasing angle with respect to the anchor point.
Graham Scan: Phase 2

The anchor point and the next point on the path must be on the hull (why?)
Graham Scan: Phase 2

- keep the path and the hull points in two sequences
- elements are removed from the beginning of the path sequence and are inserted and deleted from the end of the hull sequence
- orientation is used to decide whether to accept or reject the next point
(p,c,n) is a right turn!

Discard c
(p,c,n) is a right turn!

(p,c,n) is a right turn!

(p,c,n) is a right turn!
Time Complexity of Graham Scan

- Phase 1 takes time $O(N \log N)$
  - points are sorted by angle around the anchor
- Phase 2 takes time $O(N)$
  - each point is inserted into the sequence exactly once, and
  - each point is removed from the sequence at most once
- Total time complexity $O(N \log N)$
How to Increase Speed

- Wipe out a lot of the points you know won’t be on the hull! This is *interior elimination*
- Here’s a good way to do interior elimination if the points are randomly distributed in a square with horizontal and vertical sides:
  - Find the farthest points in the SW, NW, NE, and SE directions
  - Eliminate the points inside the quadrilateral (SW, NW, NE, SE)
  - Do Package Wrap or Graham Scan on the remaining points (only $O(\sqrt{N})$ points are left on average!)
Dynamic Convex Hull
a.k.a. the Wrath of Khan

• The basic convex hull algorithms were fairly interesting, but you may have noticed that you can’t draw the hull until after all of the points have been specified.

• Is there an interactive way to add points to the hull and redraw it while maintaining an optimal time complexity?
Two Halves = One Hull

• For this algorithm, we consider a convex hull as being two parts:

- An **upper** hull:

- and a **lower** hull:
Adding points: Case 1

• Case 1: the new point is within the horizontal span of the hull
  - Upper Hull 1a:
    If the new point is above the upper hull, then it should be added to the upper hull and some upper-hull points may be removed.
  
  - Upper Hull 1b:
    If the new point is below the upper hull, no changes need to be made.

![Diagram showing convex hull with added point and logic for Case 1](image-url)
Case 1 (cont.)

- The cases for the lower hull are similar.

- **Lower Hull 1a:**
  If the new point is below the lower hull, then it is added to the lower hull and some lower-hull points may be removed.

- **Lower Hull 1b:**
  If the added point is above the existing point, it is inside the existing lower hull, and no changes need be made.
Adding Points: Case 2

- Case 2: the new point is outside the horizontal span of the hull
  - We must modify both the upper and lower hulls accordingly.
Hull Modification

- In Case 1, we determine the vertices $l$ and $r$ of the upper or lower hulls immediately preceding/following the new point $p$ in the $x$-order.

- If $p$ has been added to the upper hull, examine the upper hull rightward starting at $r$. If $p$ makes a CCW turn with $r$ and its right neighbor, remove $r$. Continue until there are no more CCW turns. Repeat for point $l$ examining the upper hull leftward. The computation for the bottom hull is similar.