Polygons and the convex hull

Prof. Graeme Bailey

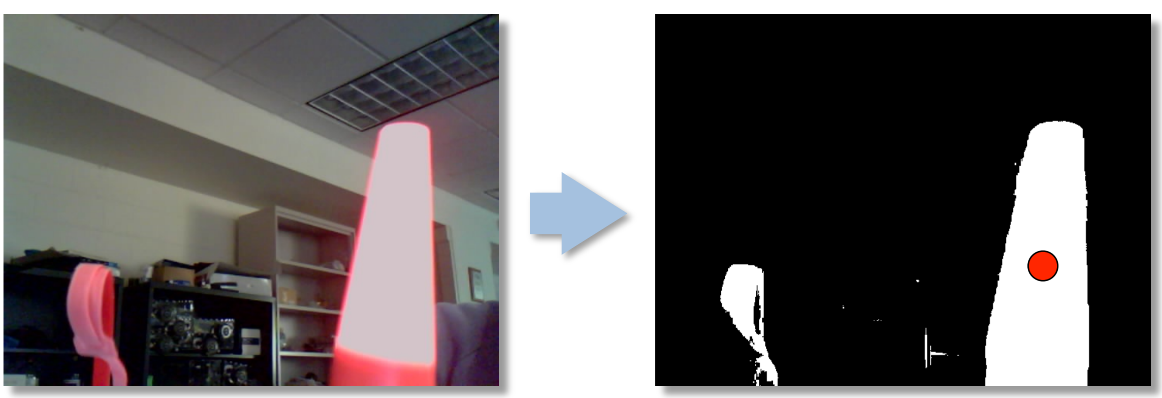
http://cs1114.cs.cornell.edu

(notes modified from Noah Snavely, Spring 2009)

Cornell University Computer Science

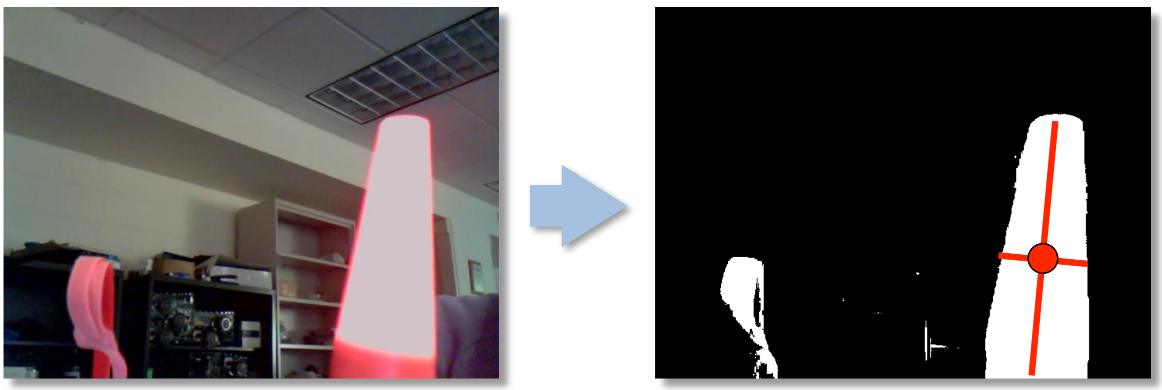
Finding the lightstick center

1. Threshold the image
2. Find blobs (connected components)
3. Find the largest blob **B**
4. Compute the median vector of **B**

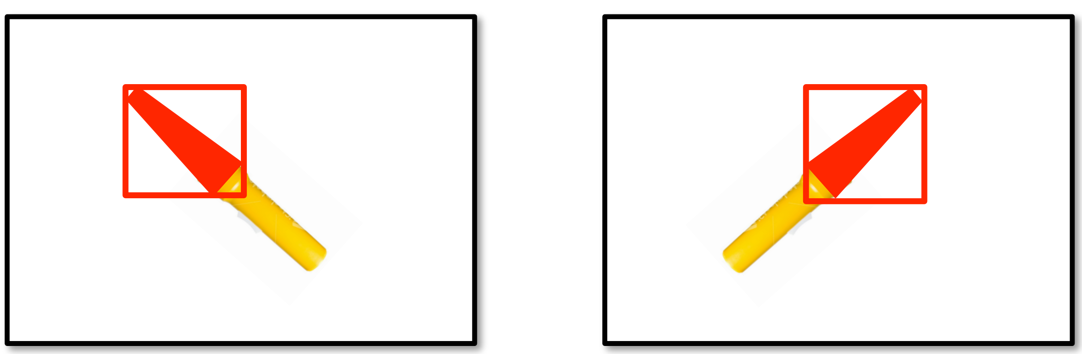


Finding the lightstick

- But we also want to control the robot based on the orientation of the lightstick
- How can we express the *shape* of the lightstick? (a box? a polygon?)



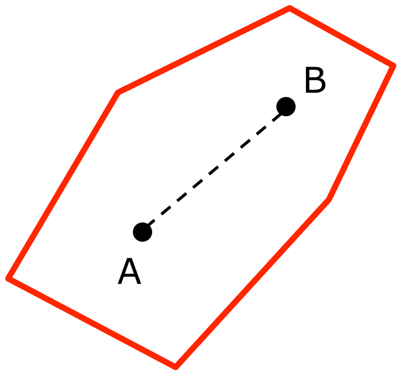
Bounding box



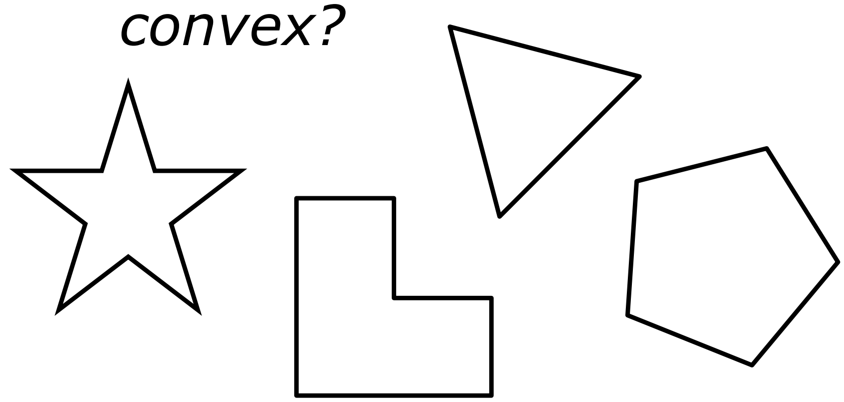
- Not as informative as we might like
- Let's come up with a polygon that fits better…

Detour: convex polygons

- A polygon *P* is **convex** if, for any two points *A*, *B* inside *P*, all points on a line connecting *A* and *B* are also inside *P*



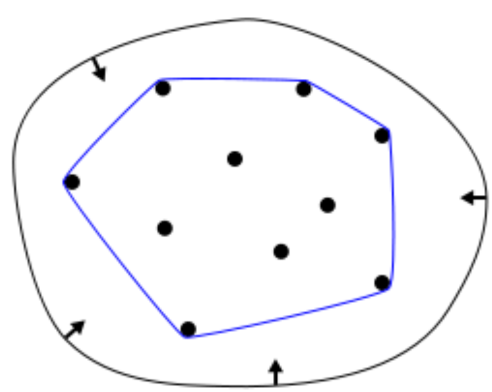
- *Which polygons are convex?*



Creating Convexity

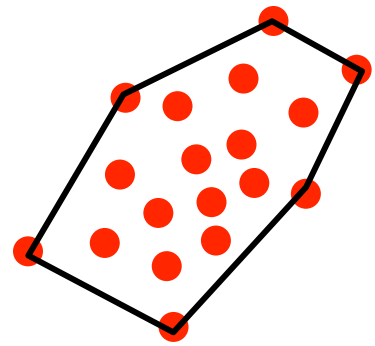
- Consider the smallest convex shape (polygon?) containing some object *P*
  - Called the **CONVEX HULL** of *P*
  - What is the convex hull of *P* if *P* is convex?

- Can also define this for sets of points in 2D: the smallest convex shape containing a set of 2D points



Convex hull of point sets

- We can use this to find a simple description of the lightstick's *shape*



http://www.cs.princeton.edu/~ah/alg_anim/version1/ConvexHull.html

- How can we compute the convex hull?

Gift-wrapping algorithm

1. Start at lowest point (this is necessarily on the convex hull)
2. Rotate the line until we hit another point (ditto)
   - All other points will lie on one side of this line
   - Look for the point that gives you the largest angle with the current line

   How do we code this part?
3. Repeat
4. You're done when you get back to the starting point

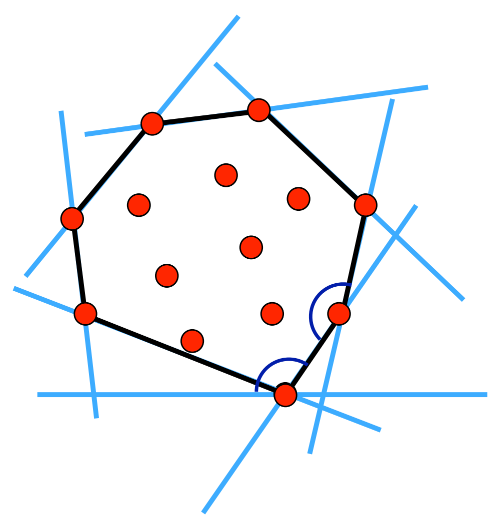
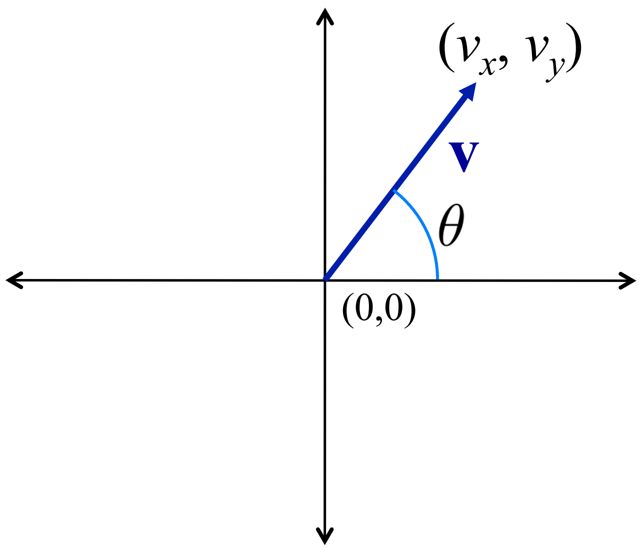


Figure credit: Craig Gotsman

Vectors

- To construct algorithms to compute convex hulls it will help to remind ourselves about vectors.



length of **v** :

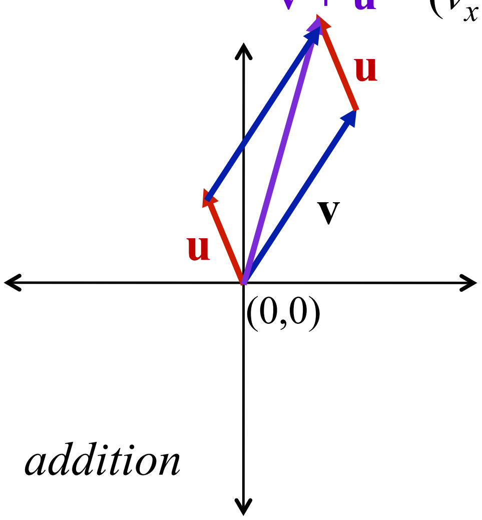
$$\|\mathbf{v}\| = \sqrt{v_x^{\,2} + v_y^{\,2}}$$

direction of **v** :

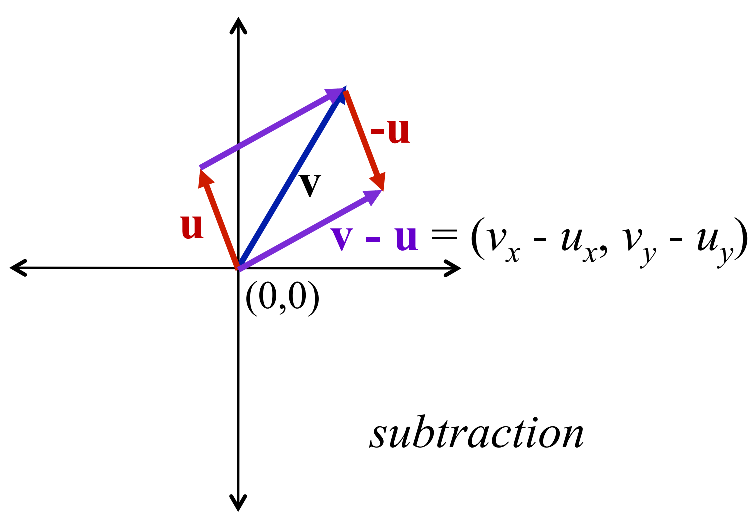
$$\theta = atan\left(\frac{y}{x}\right)$$

Vector arithmetic

$\mathbf{v} + \mathbf{u} = (v_x + u_x, v_y + u_y)$



addition



subtraction

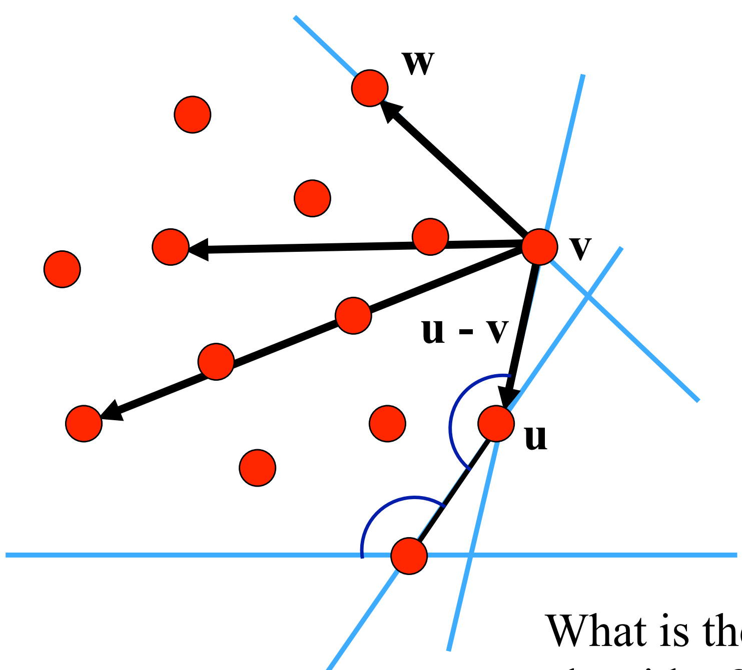
Vector lengths and angles

- Can define a scalar (inner) product of two vectors. Technically, this is anything that satisfies:
  1. **v.v** $\geq$ 0, and **v.v** = 0 if and only if **v** = **0**
  2. **v.**(a**u**) = a(**v.u**), for a any (real) number
  3. **v.u** = **u.v**
  4. **v.**(**u**+**w**) = **v.u** + **v.w**

- And then use this to *define* length and angle via
  - Length (aka norm) $||\mathbf{v}||^2 = \mathbf{v}.\mathbf{v}$
  - Angle θ by $\mathbf{v}.\mathbf{u} = ||\mathbf{v}||\ ||\mathbf{u}|| \cos\theta$

- In 2D we usually define **v.u** = $v_x u_x + v_y u_y$

Gift-wrapping revisited

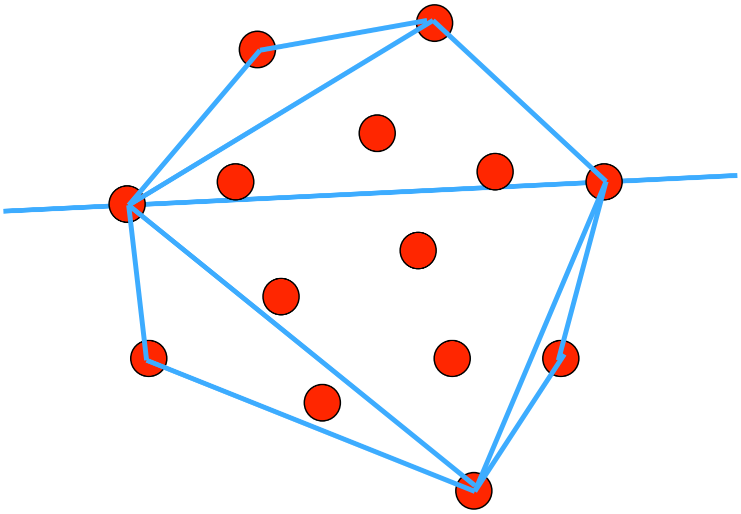


Which point is next?

Answer: the point **w** that maximizes the angle between **u** – **v** and **w** – **v**

What is the running time of this algorithm?

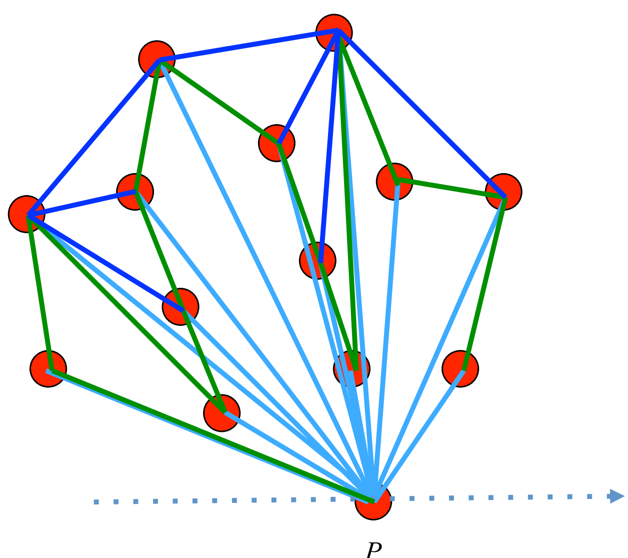
Other convex hull algorithms



1. Connect the leftmost and rightmost points (since both must be on the convex hull).
2. Recursively … find the furthest point to the left (right) of this line and form a triangle.

What is the running time of this algorithm?

Other convex hull algorithms

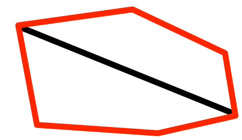


1. Start with the lowest point, and sort the points by decreasing angle to the horizontal
2. Create a polygon by joining the points in that order
3. Trace this polygon, deleting edges requiring a clockwise angle

What is the running time of this algorithm?

Lightstick orientation

- We have a convex shape
  - Now what?



- Want to find which way it's pointed

- For now, we'll find the two points that are furthest away from each other, and call that the "major axis"