Polygons and the convex hull

Prof. Noah Snavely
CS1114
http://www.cs.cornell.edu/courses/cs1114

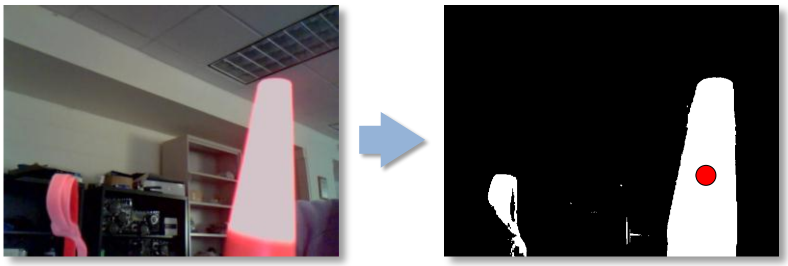
Cornell University
Computer Science

Administrivia

- Assignment 3 due this Friday by 5pm
  - Please sign up for slots on CMS

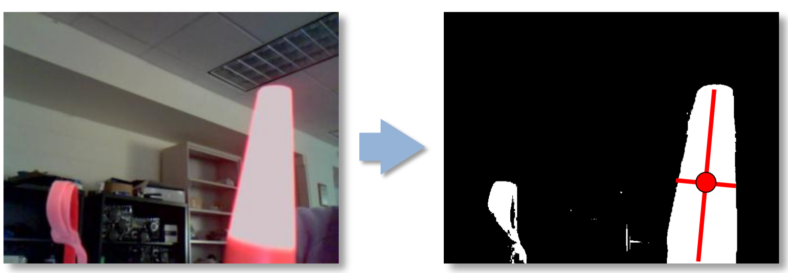
Finding the lightstick center

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



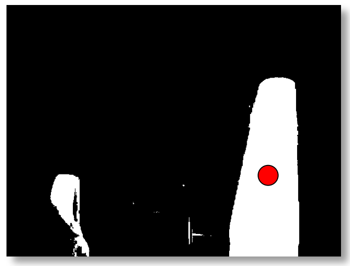
Finding the lightstick center

- But we also want to control the robot based on the orientation of the lightstick



Finding the lightstick center

- So far we've only built functions that take a set of points and return another point
  - With one exception…
- How can we express the *shape* of the lightstick?

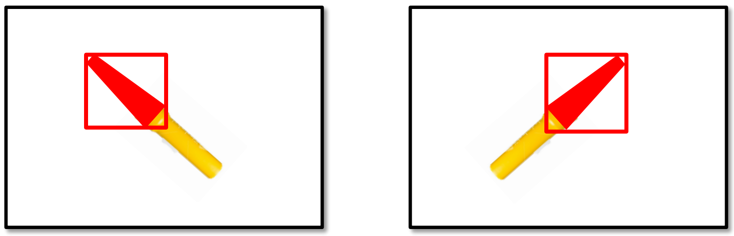


Finding the lightstick center

- We'll try to come up with a simple *polygon* to describe the lightstick

- Simplest polygon: the bounding box

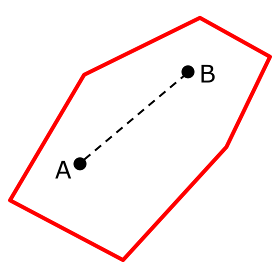
Bounding box



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

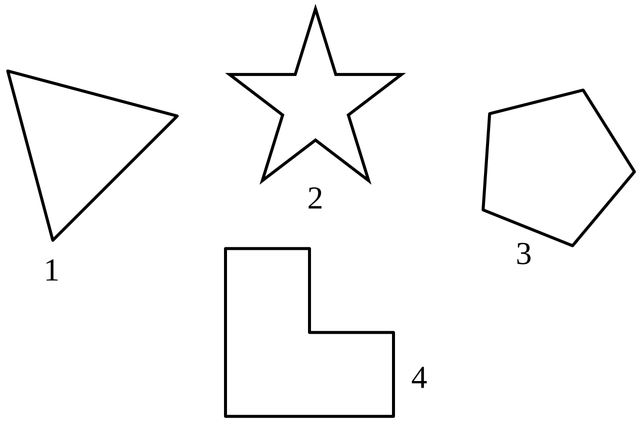
Detour: convex polygons

- 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*



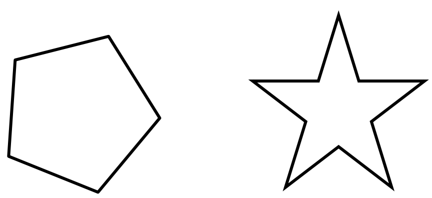
Convex polygons

- Which polygons are convex?



Testing convexity

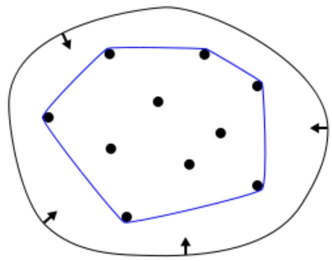
- How can we test if a polygon *P* is convex?



- Consider the smallest convex polygon containing *P*
  - Called the **CONVEX HULL**
  - What is the convex hull if *P* is convex?

Convex hull

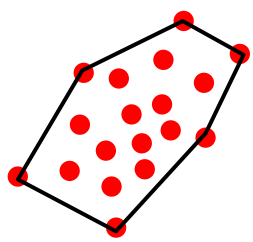
- Can also define for sets of 2D points: the smallest convex shape containing a set of 2D points



from http://en.wikipedia.org/wiki/Convex_hull

Convex hull of point sets

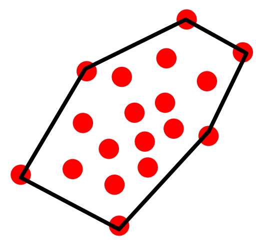
- We can use this to find a simple description of the lightstick's shape



- How can we compute the convex hull?

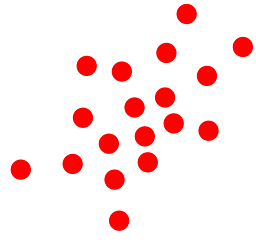
Computing convex hulls

- Idea: two points are an edge in the convex hull if ______________________



Computing convex hull

- Which two horizontal lines touch points on the convex hull?



- Which two vertical lines?
- → It is easy to identify at least four points that are part of the convex hull

Gift-wrapping algorithm

1. Start at lowest point
2. Rotate the line until we hit another point
   - 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
3. Repeat
4. You're done when you get back to the starting point

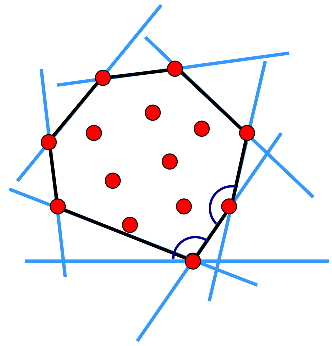


Figure credit: Craig Gotsman

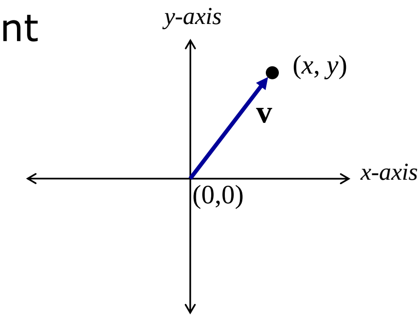
The details...

1. Start at lowest point
2. Rotate the line until we hit another point
   - 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
3. Repeat
4. You're done when you get back to the starting point

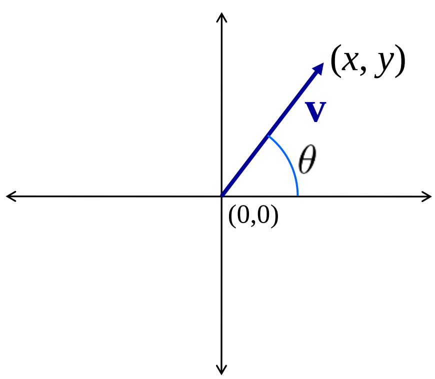
} How do we implement this part?

Vectors

- To do this, let's talk about *2D vectors*
- A vector $\mathbf{v} = (x, y)$ is an "arrow" with a direction and length
- Similar to a 2D point



Vectors



length of $\mathbf{v}$: $\|\mathbf{v}\|$

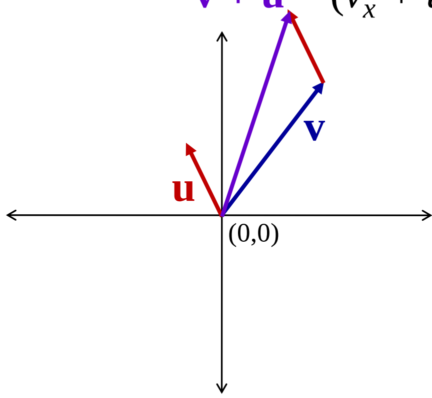
$$\|\mathbf{v}\| = \sqrt{x^2 + y^2}$$

direction of $\mathbf{v}$:

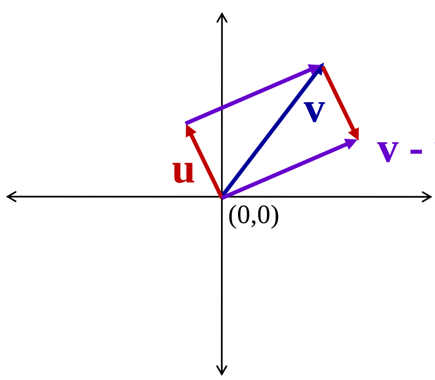
$$\theta = atan\left(\frac{y}{x}\right)$$

Vectors

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



Vectors



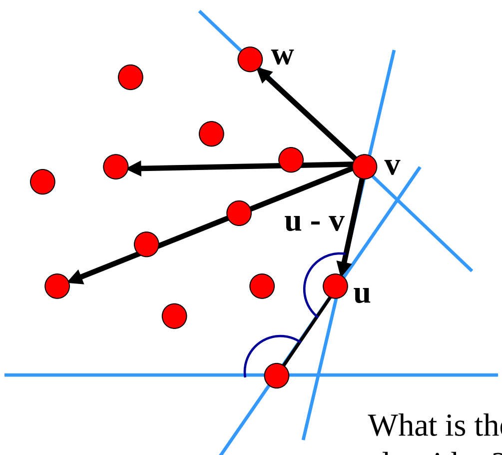
$\mathbf{v} - \mathbf{u} = (v_x - u_x, v_y - u_y)$

Vectors

- Can also "multiply" two vectors:
  - Dot product: $\mathbf{v} \cdot \mathbf{u} = v_x u_x + v_y u_y$
  - Useful fact: $\mathbf{v} \cdot \mathbf{u} = ||\mathbf{v}|| \, ||\mathbf{u}|| \cos\theta$

$$\cos\theta = \frac{\mathbf{v} \cdot \mathbf{u}}{||\mathbf{v}|| \, ||\mathbf{u}||}$$

Back to the convex hull



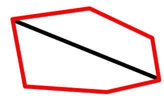
Which point is next?

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

What is the running time of this algorithm?

Lightstick orientation

- We have a convex shape
  - Now what?

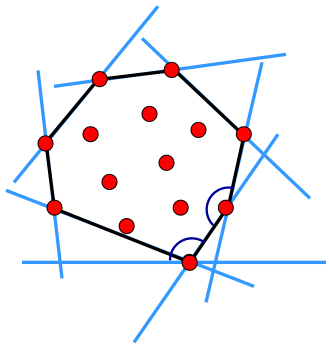


- Want to find which way it's pointed
- For now, we'll find the two points that are furthest away from each other, and call that the "major axis"

Questions?

Computing the convex hull

- Gift wrapping algorithm ("Jarvis march")



- Quickhull (divide and conquer)