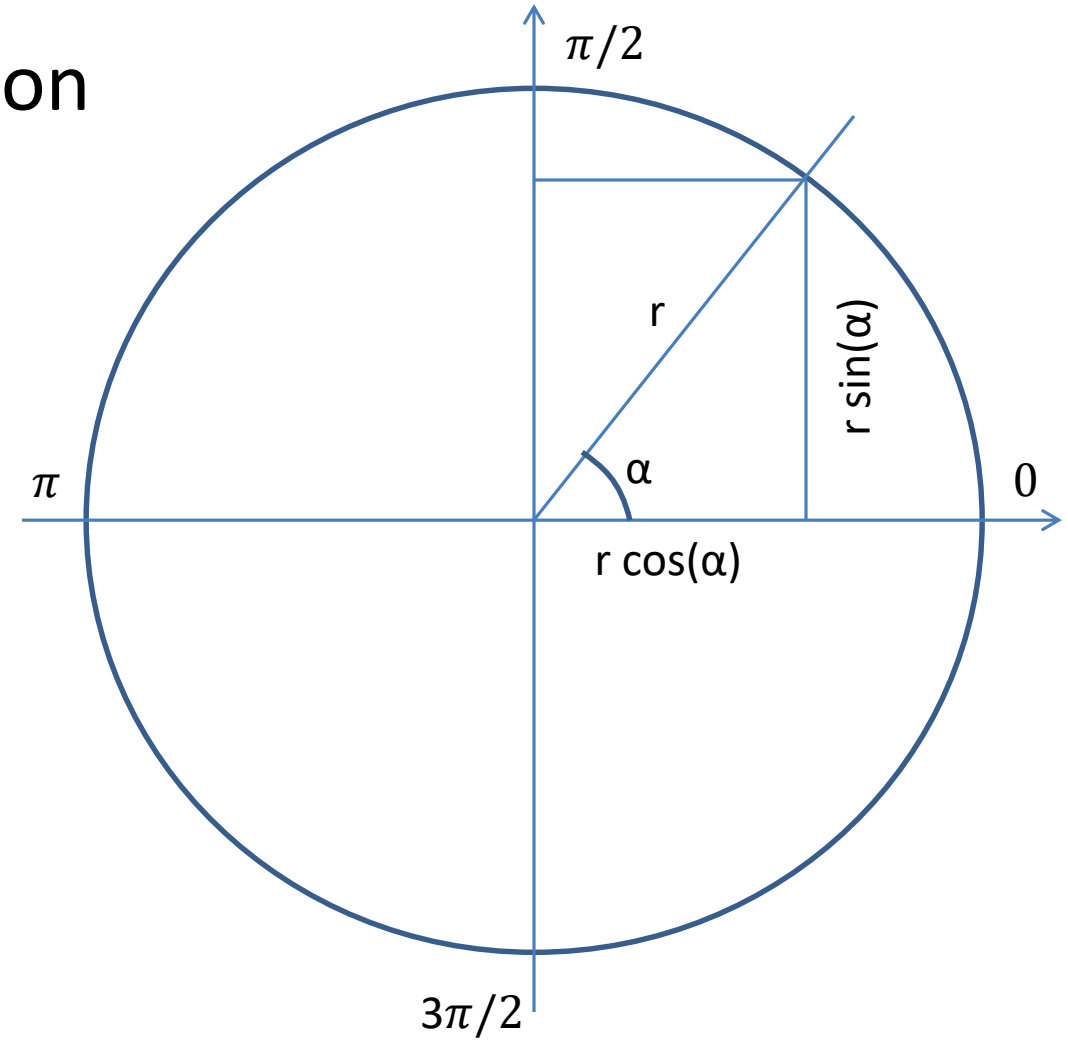
Math for AI Robots

Thomas Johansson – thomasj@cs.umu.se
Dept of Computing Science
Umeå University

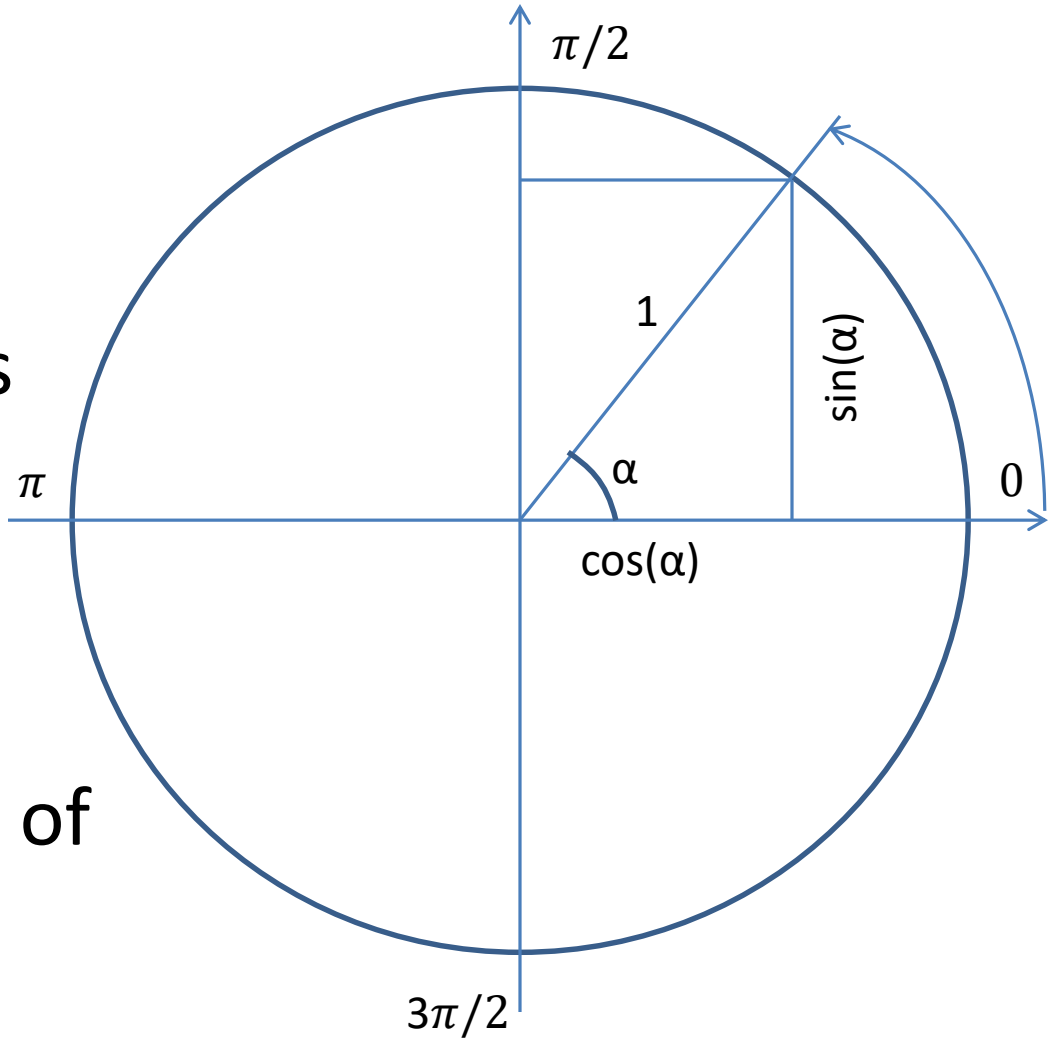
Trigonometric Functions

- The most common
  - sin(α)
  - cos(α)
  - tan(α) = $\frac{\sin(\alpha)}{\cos(\alpha)}$
  - Inverses of the above
  - acos, asin, atan



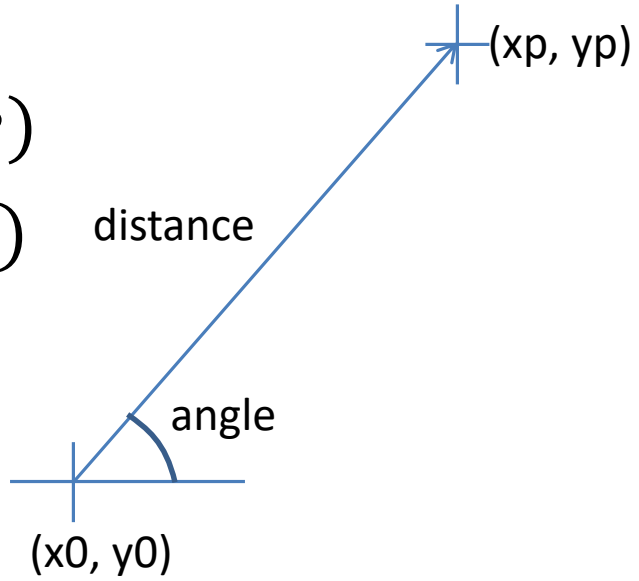
The Unit Circle

- Radius = 1
- Angles increase anticlockwise
- Angles in radians
- One radian = 180/ π degrees (about 57.3)
- Radians = length of arc



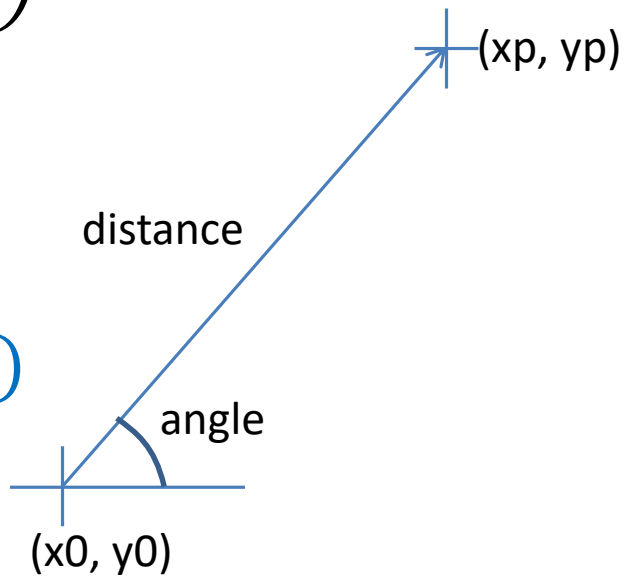
Polar Coordinates

- Angle and distance instead of x, y
  - $Let\ dx = xp - x0\ and\ dy = yp - y0$
  - $distance = \sqrt{dx^2 + dy^2}$
  - $dx = distance * \cos(angle)$
  - $dy = distance * \sin(angle)$
  - $\frac{dy}{dx} = \tan(angle)$



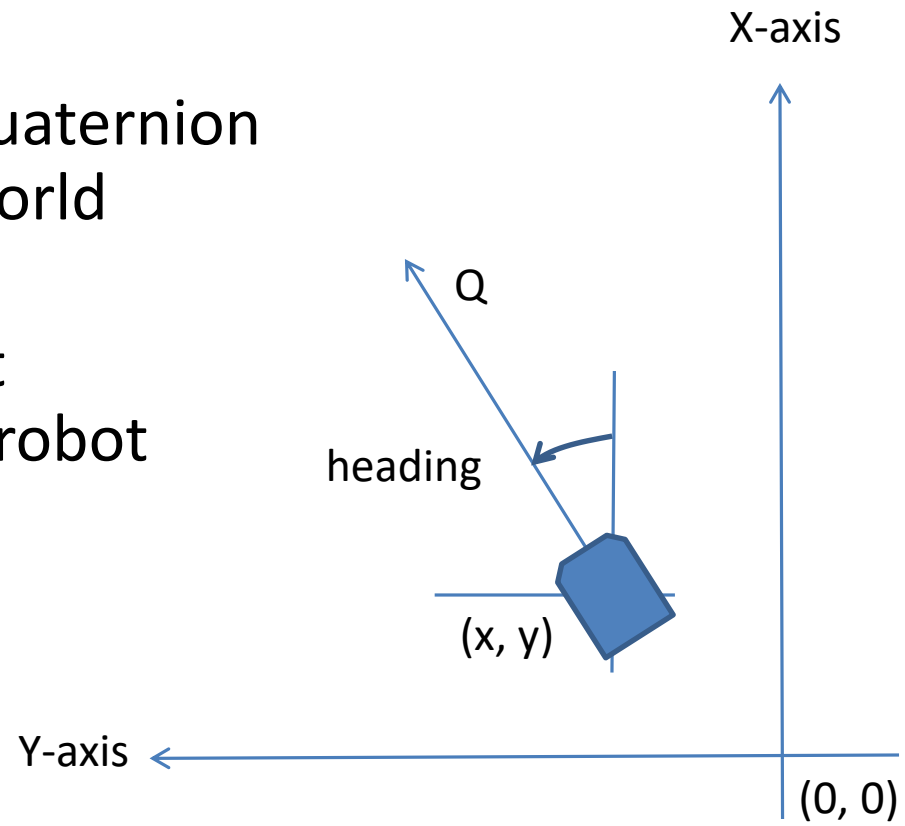
Inverse Functions

- Angles from x, y
  - $angle = arcsin(dy/distance)$
  - $angle = arccos(dx/distance)$
  - $angle = \arctan(\frac{dy}{dx})$
  - But dx may be zero
  - Use $angle = atan2(dy,\ dx)$



The Robot

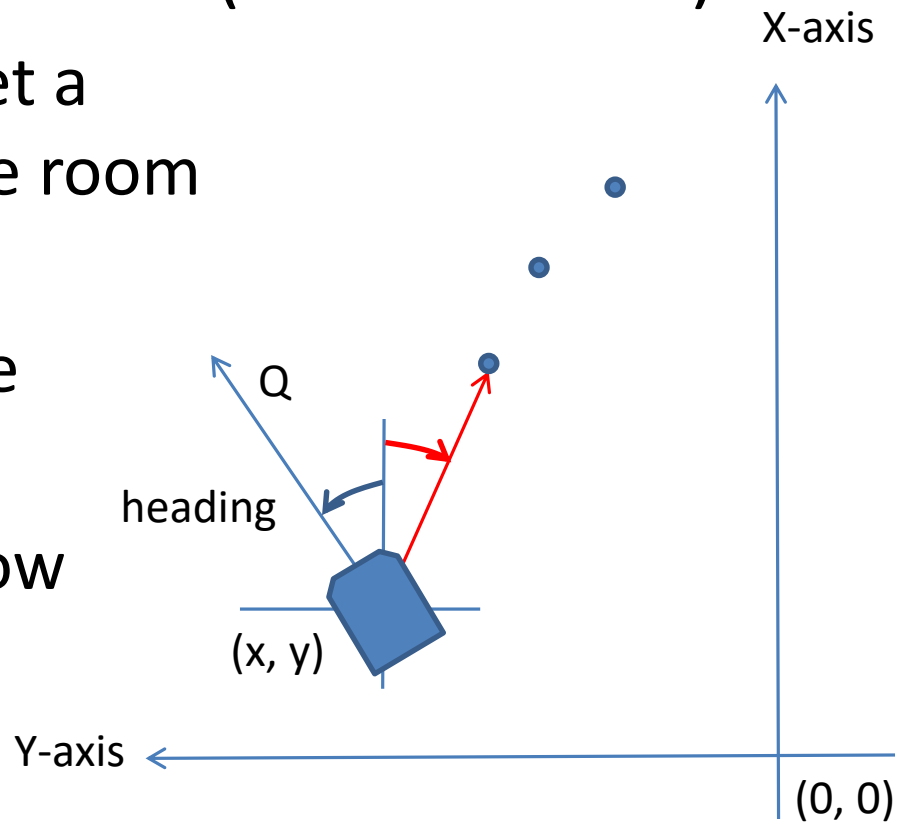
- What you get from the robot is
  - Its position, x, y, z
  - Its orientation, a Quaternion in relation to the world (the room)
  - From Q you can get the heading of the robot
  - Note the change of axes



  - $q = \cos(a/2) + i (x * \sin(a/2)) + j (y * \sin(a/2)) + k (z * \sin(a/2))$

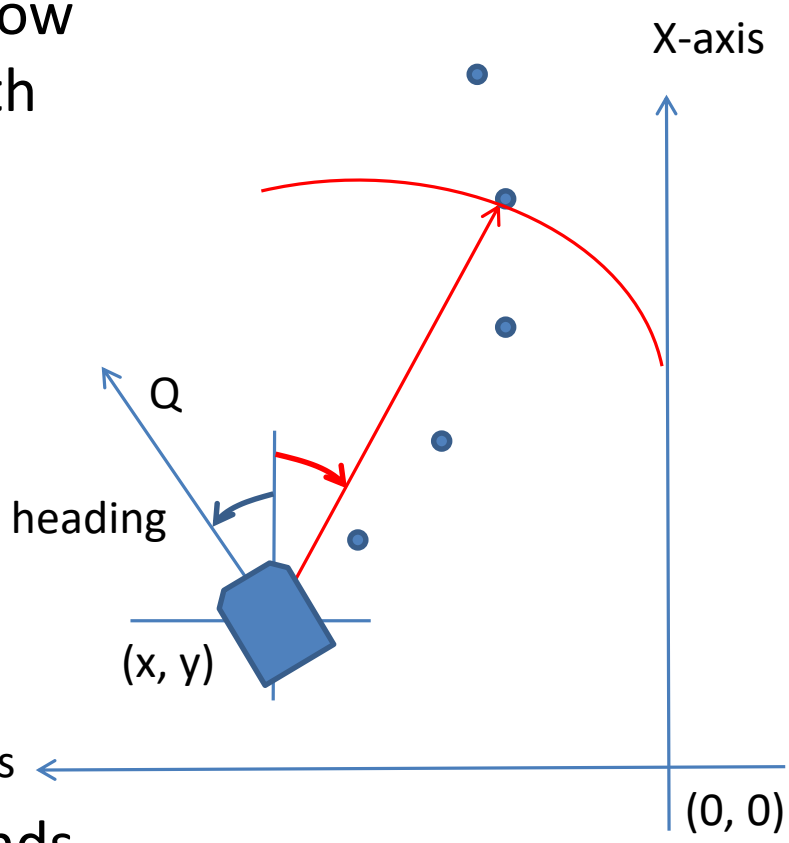
The Path

- A sequence of positions (+ orientation)
  - From it you can get a set of points in the room
  - You must find the distance and angle to all these points
  - Then you know how to steer the robot



The Algorithm(s)

- Choose *Follow the Carrot* or *Pure Pursuit*
  - You will have to choose a 'lookahead distance'
  - The size of this determines how accurately you follow the path
  - Long distance – faster but riskier
  - You should not visit every point exactly
  - Do not stop-turn-start, instead alter the speed and turn as the robot moves
  - Note: put a small delay (10-50 ms) between commands



Localization

- A JSON object (string)
  - Returned from the robot
  - Stored in the path file in a JSON array
    [{...}, {...}, ...]
  - In Python – map to dictionary or similar
  - In Java – map to Map<String, Object> or Collection<Map< ...>>


```
{
  "Pose":
  {
    "Orientation":
    {
       "W":-0.707528083159215 67,
       "X":0,
       "Y":0,
       "Z":0.706685228047852 94
    },
    "Position":
    {
       "X":0.062024351309485075,
       "Y":-4.878735976436840 5,
       "Z":0
    }
  },
  "Status":0,
  "Timestamp":75949220
}
```