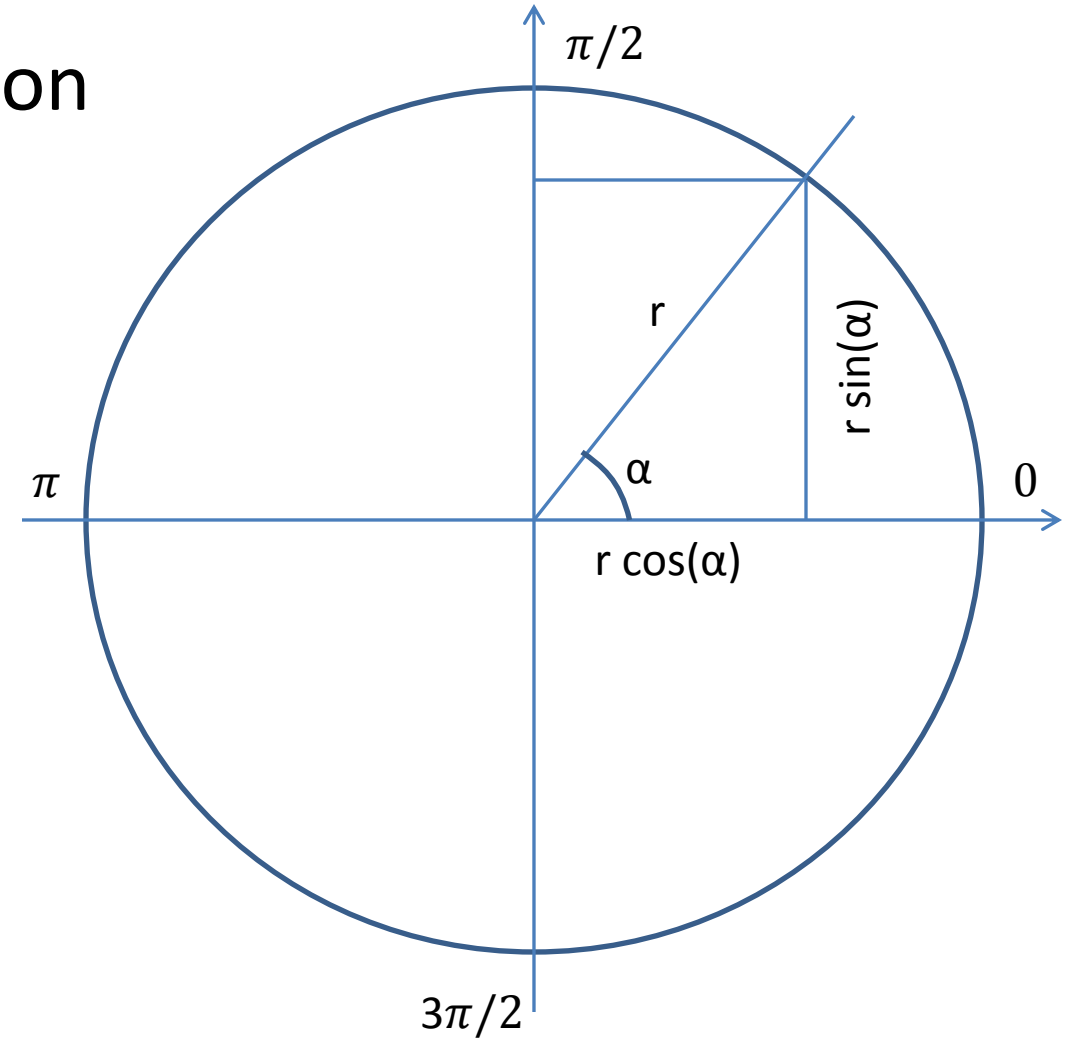
Math for AI (Robots)

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

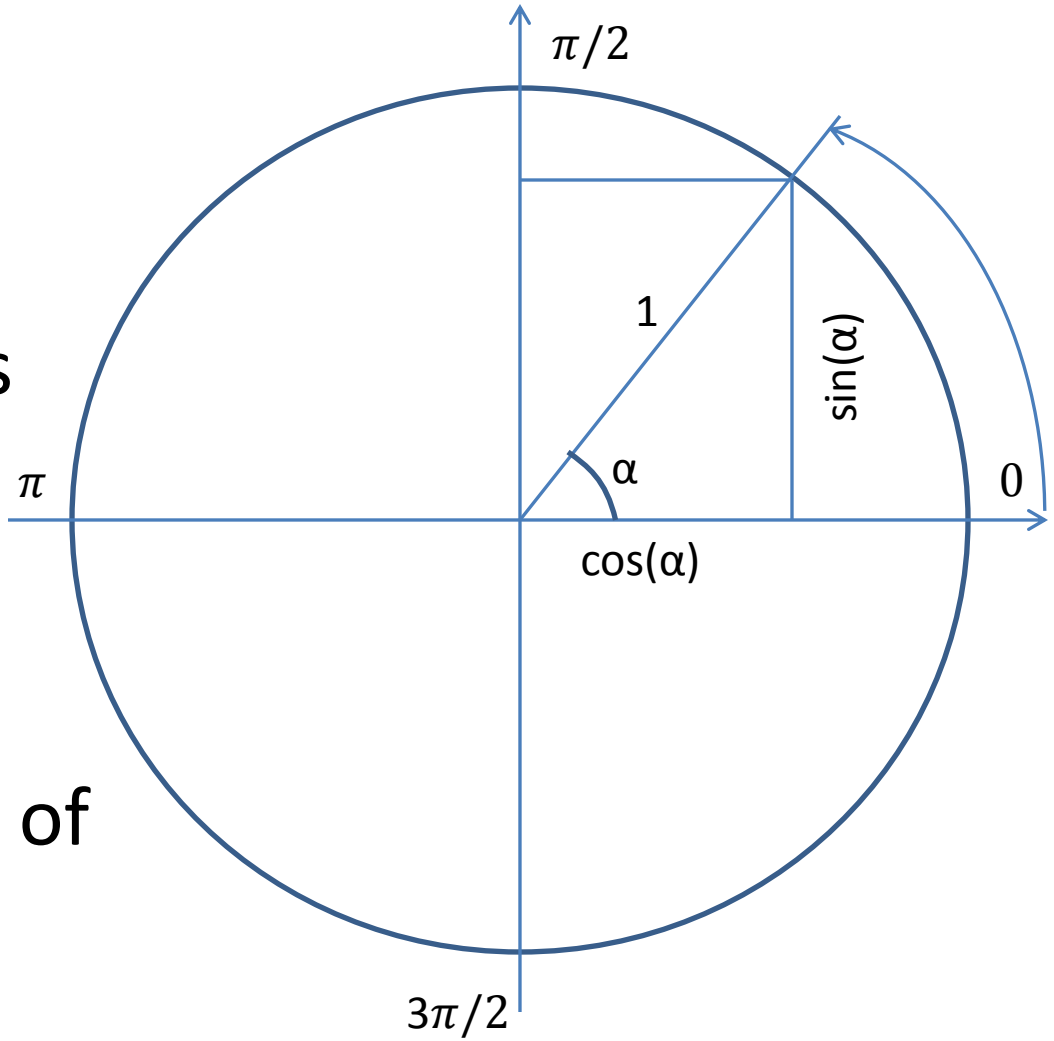
Trigonometric Functions

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



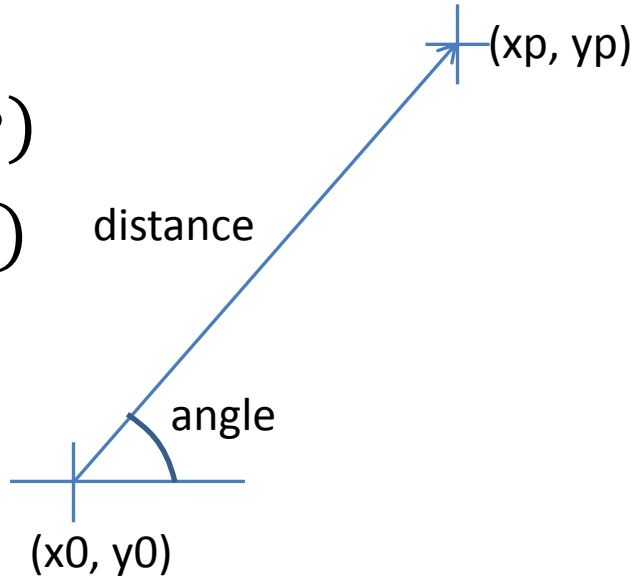
The Unit Circle

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



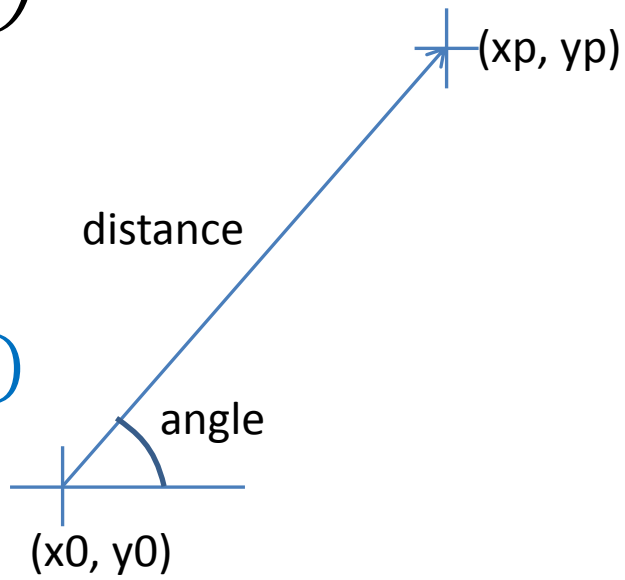
Polar Coordinates

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



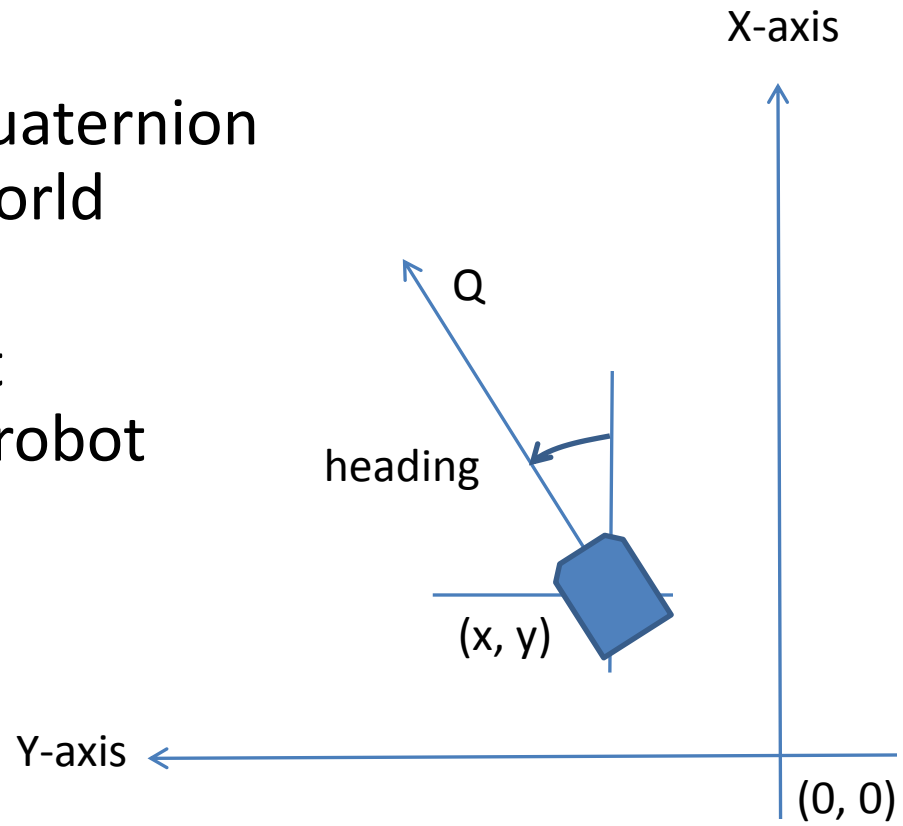
Inverse Functions

- 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

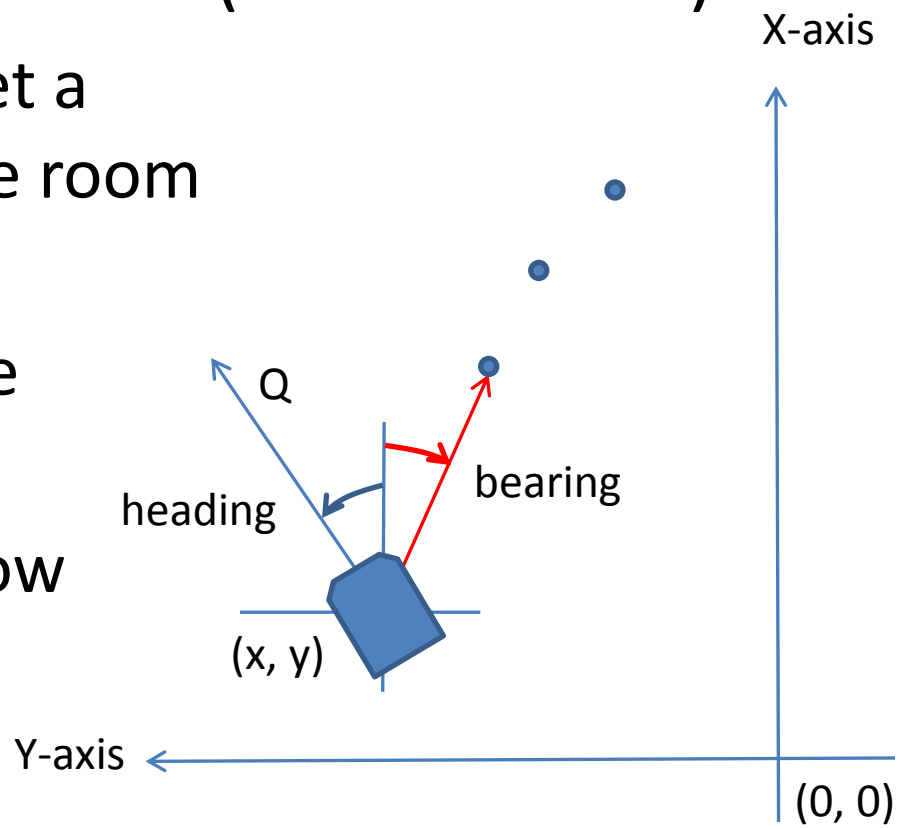
- What you get from the robot is
  - Its position, x, y, z
  - Its orientation, a Quaternion in relation to the world (the room)
  - From Q you can get the heading of the robot
  - Note the change of axes



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

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



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
}
```

Read the path file - Java

```
File pathFile = new File("Path-around-table.json");

BufferedReader in = new BufferedReader(new InputStreamReader(
new FileInputStream(pathFile)));mapper = new ObjectMapper();

// read the path from the file
Collection <Map<String, Object>> data =
   (Collection<Map<String, Object>>) mapper.readValue(in, Collection.class);

nPoints = data.size();
path = new Position[nPoints];

int index = 0;
for (Map<String, Object> point : data)
{
   Map<String, Object> pose = (Map<String, Object>)point.get("Pose");
   Map<String, Object> aPosition = (Map<String, Object>)pose.get("Position");

   double x = (Double)aPosition.get("X");
   double y = (Double)aPosition.get("Y");
   path[index] = new Position(x, y);
   index++;
}
```

Talk to the robot - Java

```
RobotCommunication robotcomm = new RobotCommunication(host, port);
LocalizationResponse lr = new LocalizationResponse(); // response
DifferentialDriveRequest dr = new DifferentialDriveRequest(); // request

dr.setAngularSpeed(Math.PI * 0.25); // set up the request to move in a circle
dr.setLinearSpeed(1.0);
int rc = robotcomm.putRequest(dr);  // move

for (int i = 0; i < 16; i++)
{
   // wait a second

   robotcomm.getResponse(lr); // ask the robot about its position and angle

   double angle = lr.getHeadingAngle();
   System.out.println("heading = " + angle);

   double [] position = getPosition(lr);
   System.out.println("position = " + position[0] + ", " + position[1]);
}

// set up request to stop the robot
dr.setLinearSpeed(0);
dr.setAngularSpeed(0);
rc = robotcomm.putRequest(dr);
```