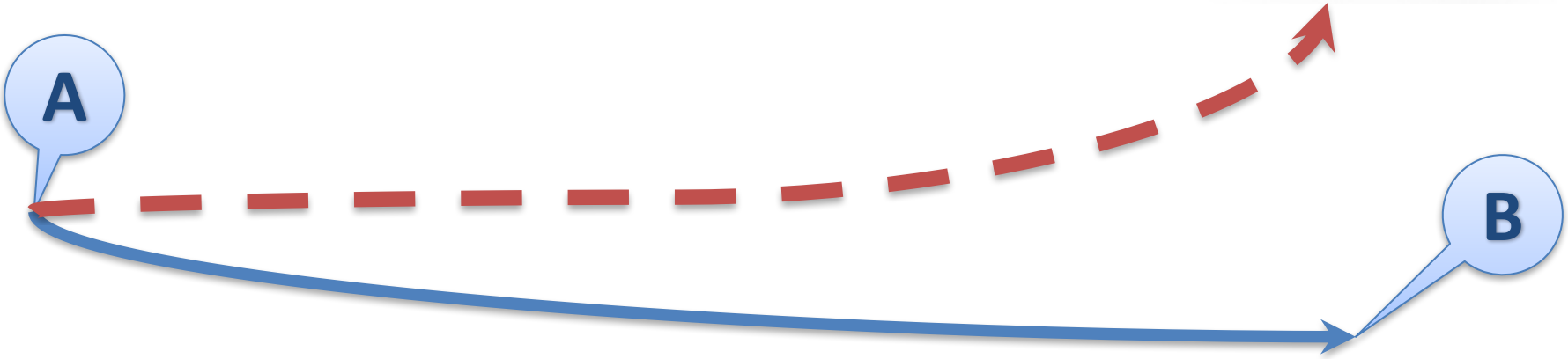
CSCI 1108

Actuators

Kinematics

Odometry



leds.prox.h(led0, led1, led2, led3, led4, led5, led6, led7) {0...32}

leds.buttons(led0, led1, led2, led3) {0...32}

leds.circle(led0, led1, led2, led3, led4, led5, led6, led7) {0...32}

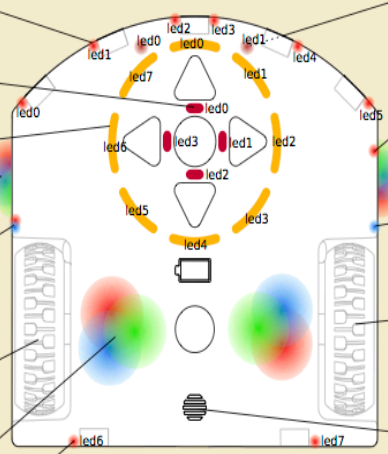
leds.bottom.left(red, green, blue) {0...32}

leds.temperature(red, blue) {0...32}

motor.left.target desired speed {-500...500}, 500 = ~20 cm/s
motor.left.speed actual speed
motor.left.pwm motor command
motor 100 Hz

leds.top(red, green, blue) {0...32}

leds.prox.h(led0, led1, led2, led3, led4, led5, led6, led7) {0...32}



leds.prox.v(led0, led1) {0...32}

leds.rc(led) {0...32}

leds.bottom.right(red, green, blue) {0...32}

leds.sound(led) {0...32}

motor.right.target desired speed {-500...500}, 500 = ~20 cm/s
motor.right.speed actual speed
motor.right.pwm motor command
motor 100 Hz

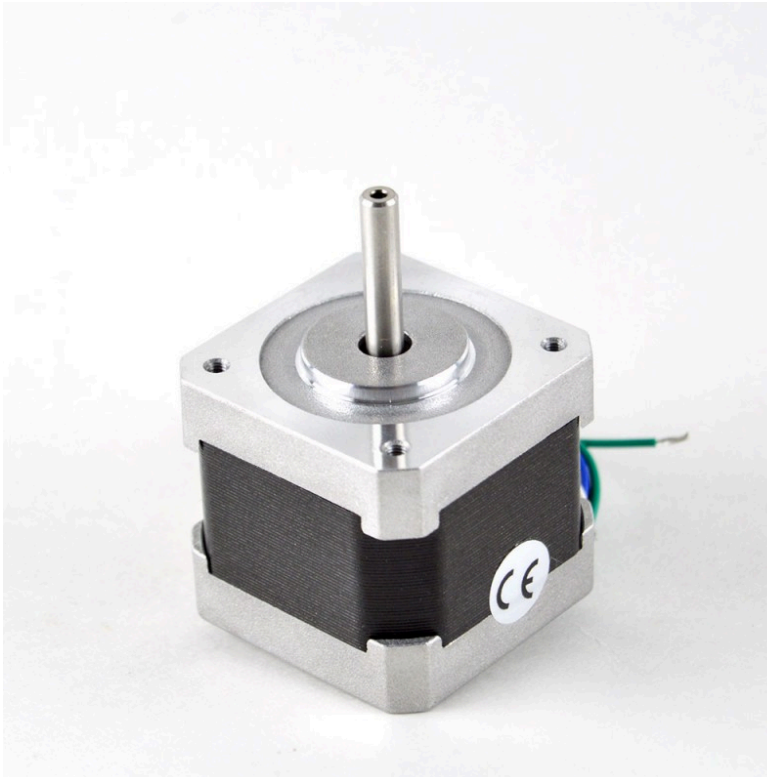
sound.finished a sound finished playing
sound.system(N) N: {0...7}, play system sound N. N=-1, stop playing
sound.freq(Hz,ds) [Hz],[1/60 s]
sound.wave(wave[142]) change primary wave, wave[i] : {-128...127}
sound.play(N) N: {0...32767}, play 'pN.wav'. N=-1, stop playing
sound.replay(N) N: {0...32767}, replay 'rN.wav'. N=-1, stop playing

Actuators

DC motor

Servo Motor

Stepper motor



Our experimental approach: motion model (Tutorial 5)

Model the effect of running the motors with different power

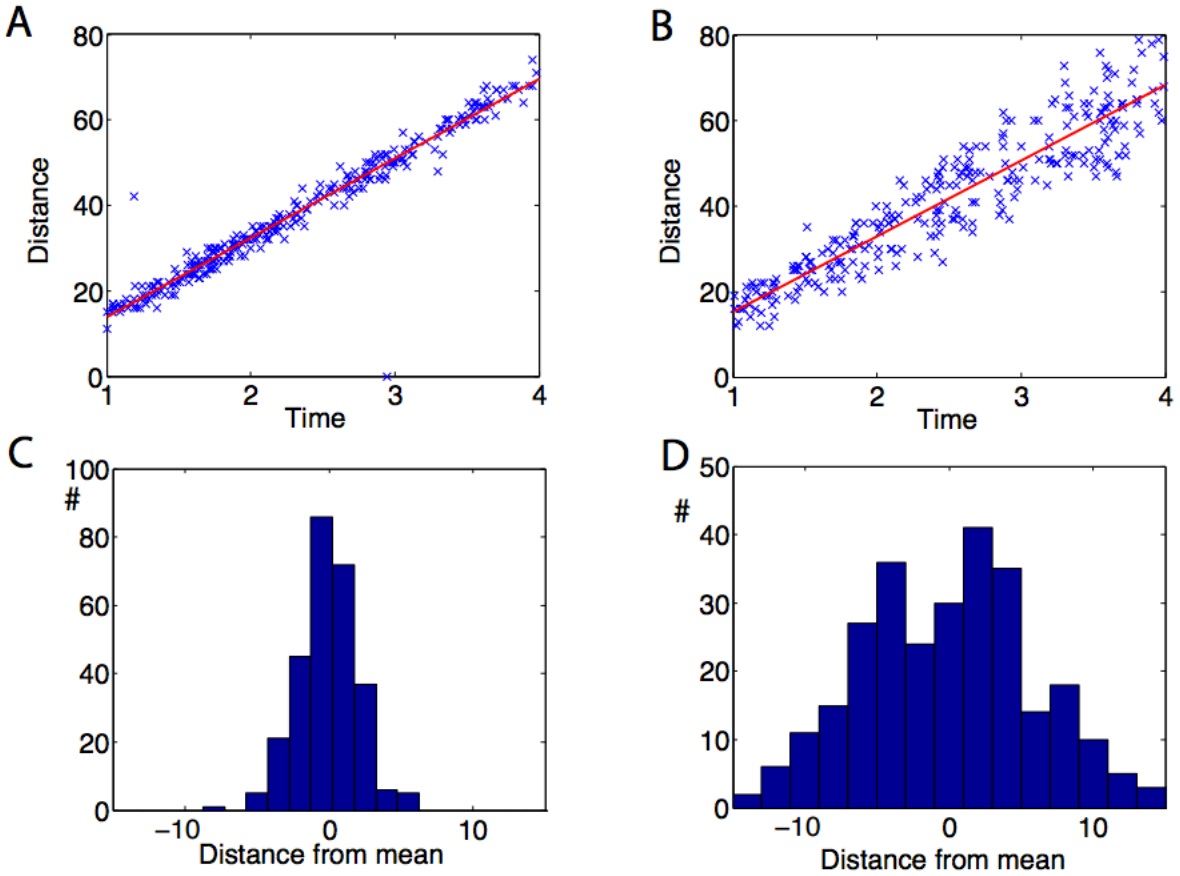
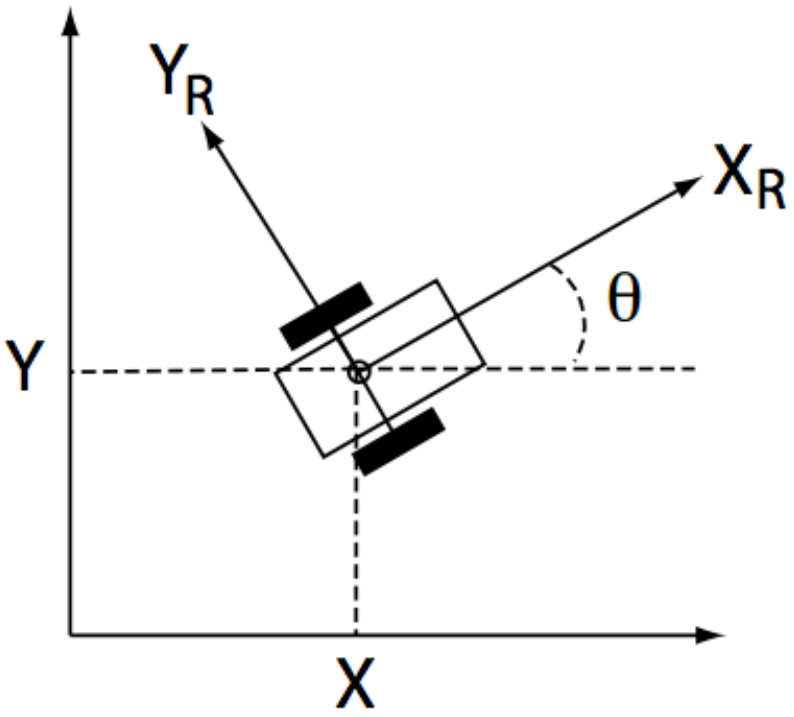


Fig. 4.1 (A) Measurements of distance travelled by the tribot when running the motor for different number of milliseconds with constant motor power. (B) Same as (A) with random motor power. (C,D) Corresponding histogram of differences between data and hypothesis.

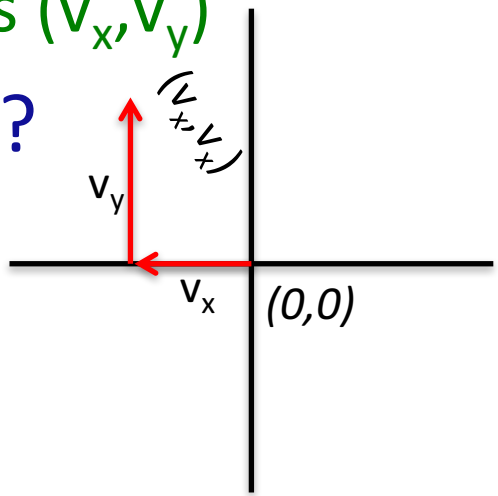
Kinematics is the branch of classical mechanics which describes the motion of points … without consideration of the masses of those objects nor the forces that may have caused the motion. (Wikipedia)

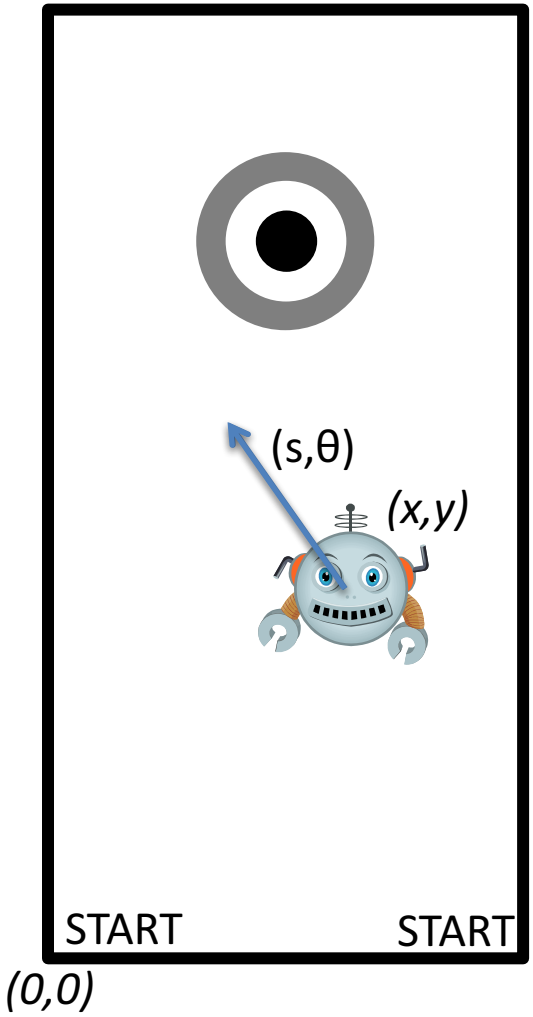


Pose of the robot

Velocity

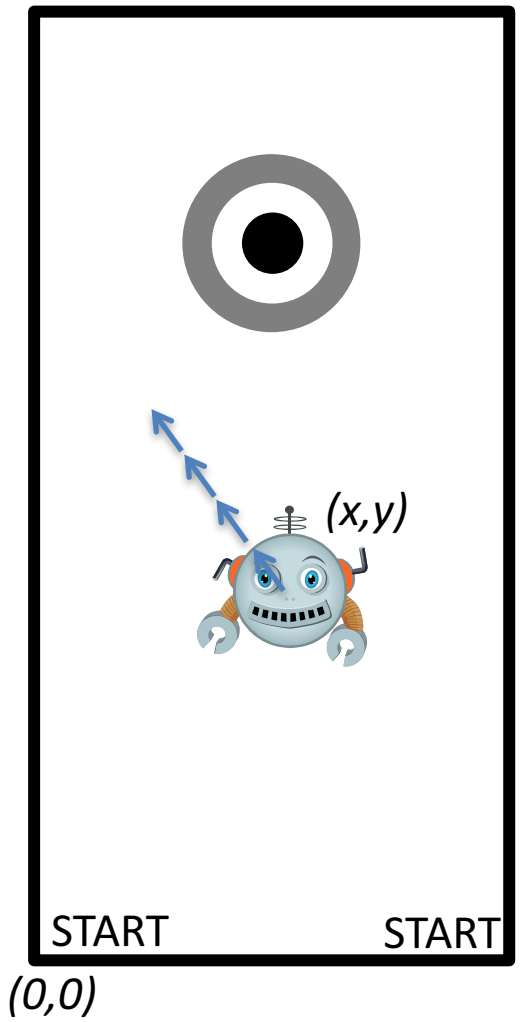
- Velocity can be represented in terms of
  - speed and direction (s,θ) or
  - horizontal and vertical speed components (v_x,v_y)
- What is (0,0)?





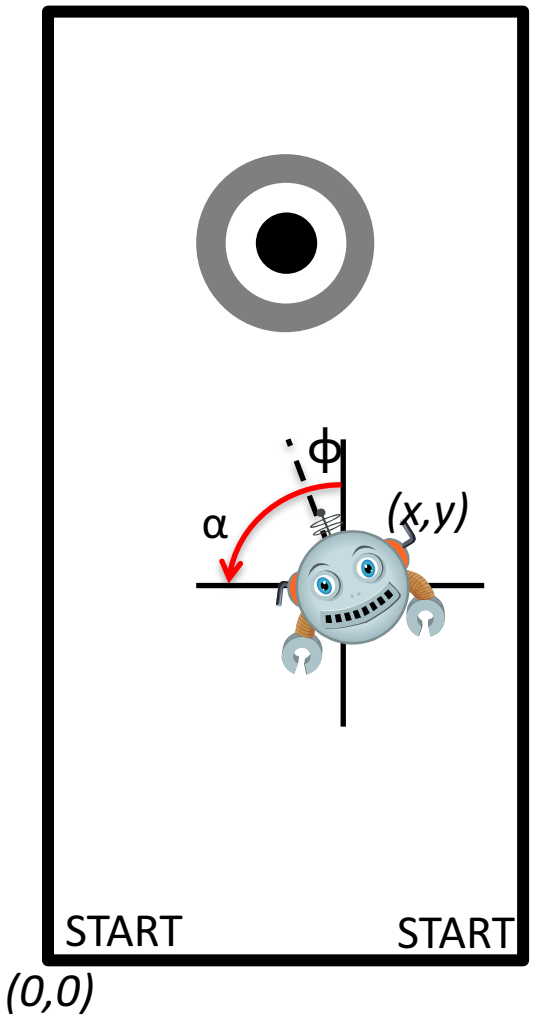
Differential (Linear) Motion

- **Observation:** The velocity vector represents distance per unit time, e.g., (cm/s)
- **Idea:** Update position by adding velocity to position proportionally to elapsed times Δt
  - new position = old position + velocity × time
- Suppose velocity is represented by (s,θ)
  - $x' = x + s \times \sin(\theta) \times \Delta t$
  - $y' = y + s \times \cos(\theta) \times \Delta t$
- Suppose velocity is represented by (v_x,v_y)
  - $x' = x + v_x \times \Delta t$
  - $y' = y + v_y \times \Delta t$



Angular Motion

- **Obs:** Robots sometimes need to turn
- **Assumption:** Robot will turn on the spot
  - Orientation **ϕ** will change
  - Position **(x,y)** does not change
  - Angular velocity **α** (deg/s) does not change
- **Idea:** Update orientation every second
  - new orient. = old orient. + angular velocity × time
  - $\phi' = \phi + (\alpha \times \Delta t)$
- How do we determine (v_x, v_y)?
- **Observations:** We know the velocity (s, θ)
  - Speed **s** is based on motor power
  - Direction **θ** is equal to the orientation **ϕ**
- Hence
  - $v_x = s \times \sin(\theta)$
  - $v_y = s \times \cos(\theta)$



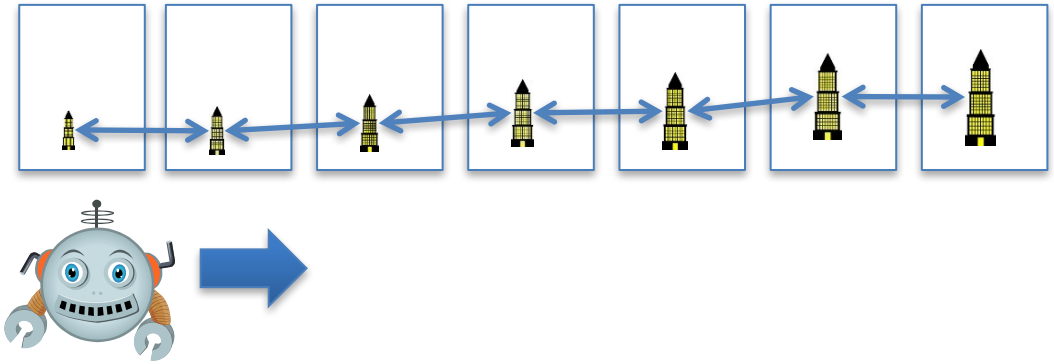
Odometry is the use of data from motion sensors to estimate change in position over time (Wikipedia)

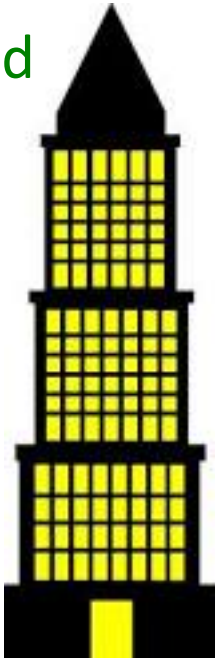
Sources of Data for Odometry

- Motor sensors
  - rotation sensors (how fast the motor is turning)
- Motion sensors
- Accelerometers and Gyroscopes
- Compass
  - Very useful for orientation
- Cameras
- Rangefinders (infrared, ultrasonic, or laser)

Optical Flow based Odometry

- Idea: Gauge the robot's velocity by comparing objects (features) in consecutive camera images
  - Extract features from image
  - Match from image to image (construct optical flow)
  - Estimate camera (robot) motion
  - Periodically update set of features being tracked
- Adjust speed of robot based on estimate





Video by James Bowman and Kurt Konolige's work.

Problems with Vision based Odometry

- Images are affected by environment conditions
  - light, fog, rain, dust, etc
- Objects can become occluded
- Feature extraction is expensive and imperfect
- Distance estimation is error-prone
- Landmarks can change
- Entire process is highly variable
- Other technologies are use specific but more accurate
  - range finders, GPS, etc
- Why do we care?
  - One of the most common tasks in robotics is to map (explore) a given environment
    - Robot must know where it is and where it was
    - This includes searching (avoid searching same place twice)

Outlook

So far we only started to described the effects of changing the position of the robot with motion. The real problem to be solved is to continiously estimate the robot position → localization