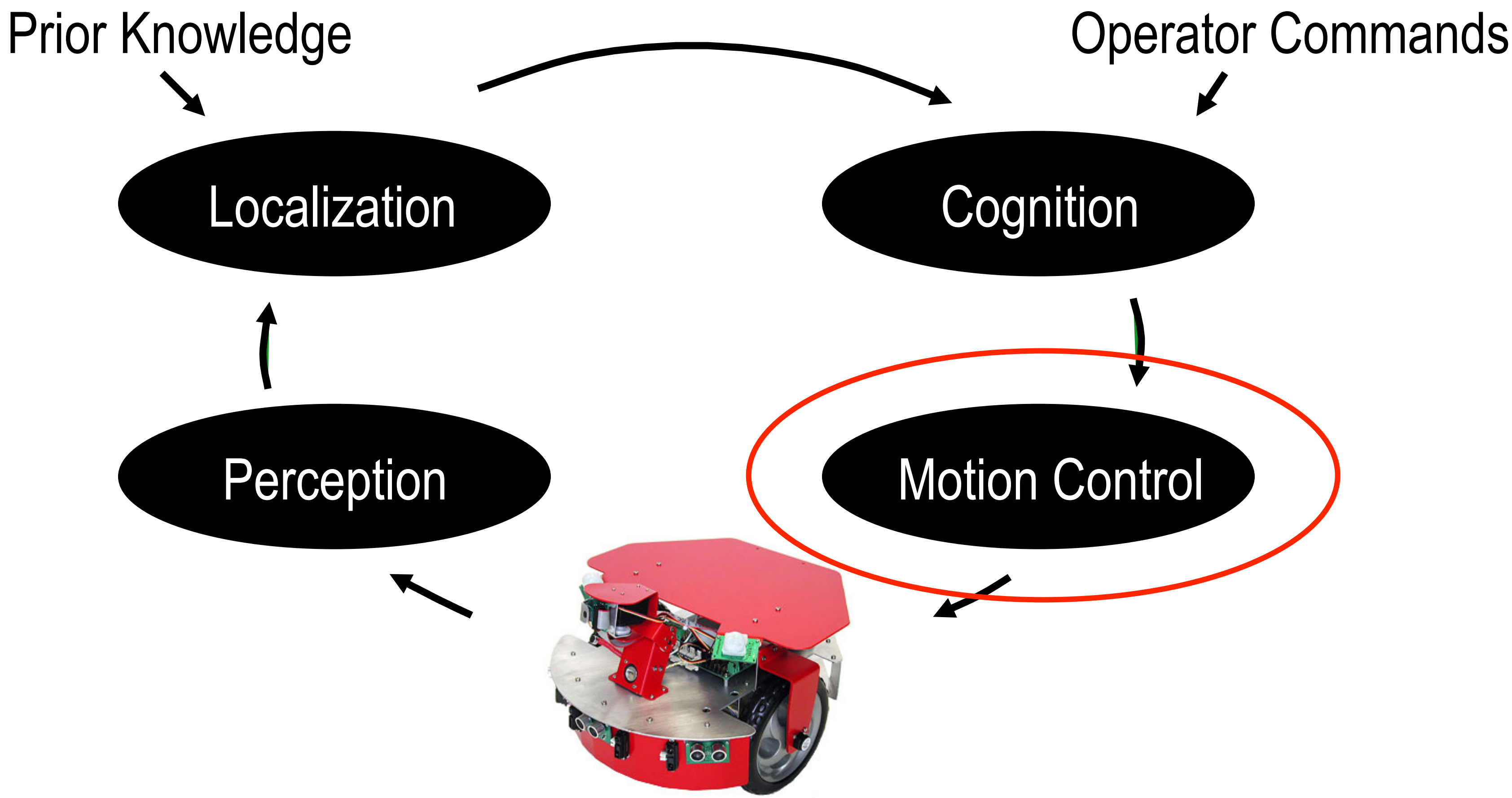
COS 495 - Lecture 6
Autonomous Robot Navigation

Instructor: Chris Clark

Semester: Fall 2011

Figures courtesy of Siegwart & Nourbakhsh

Control Structure



Point Tracking

1. Linear Systems
2. Motion Control
3. Reachable Space

Linear Systems

- Recall that the forward kinematics are a linear differential equation.
- We will use this equation to help develop a motion controller for point tracking
- We start by observing how the state x behaves if it obeys the following equation:

$$\dot{x} = dx/dt = ax$$

where *a* is a constant

Linear Systems

- It should be obvious that the solution to the equation

$$\dot{x} = ax$$

is

$$x(t) = x_0 exp(at)$$

where

x_0 is the initial state

Linear Systems

- To confirm this solution, substitute into the original equation:

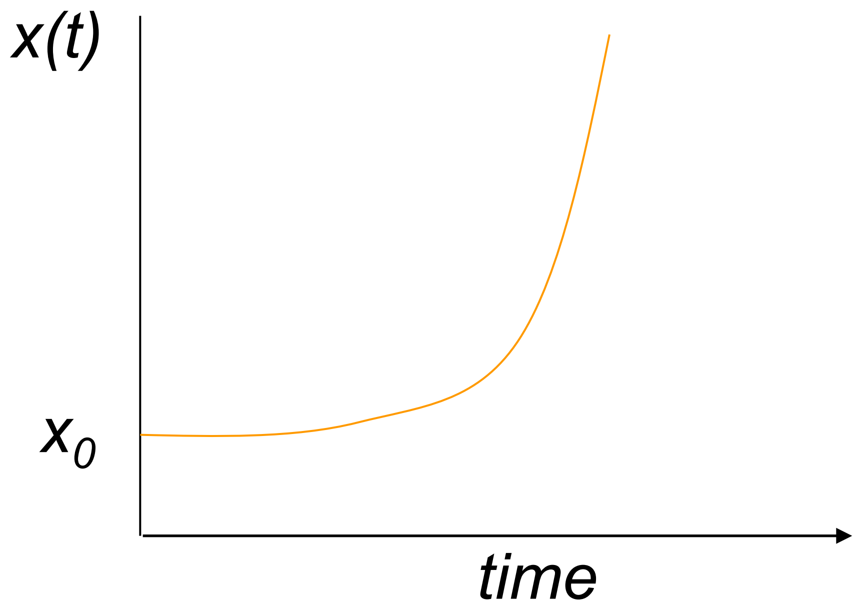
$$\dot{x} = ax$$

$$d[x_0 \exp(at)]/dt = a[x_0 \exp(at)]$$

$$ax_0 \exp(at) = ax_0 \exp(at)$$

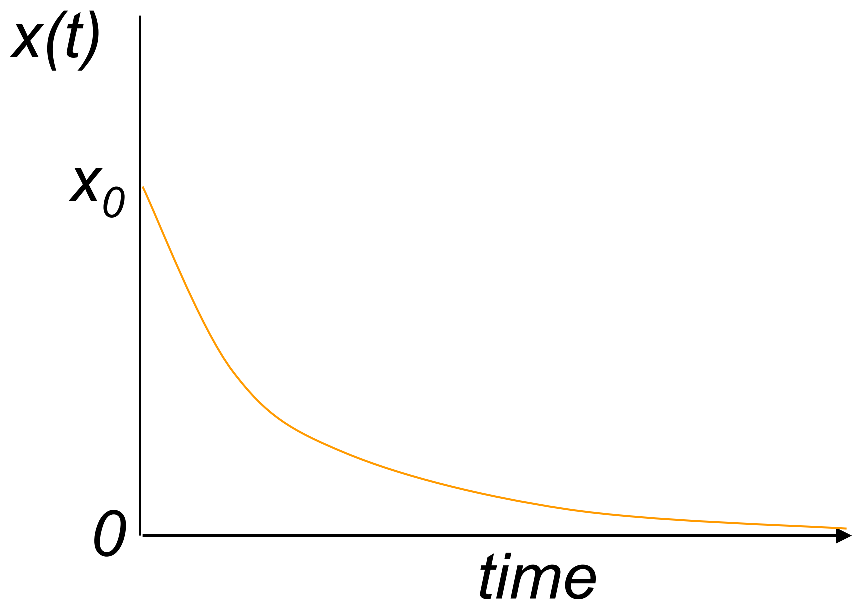
Linear Systems

- To view how the state *x* behaves over time, we can plot out $x = x_0 \exp(at)$, assuming *a* is positive:



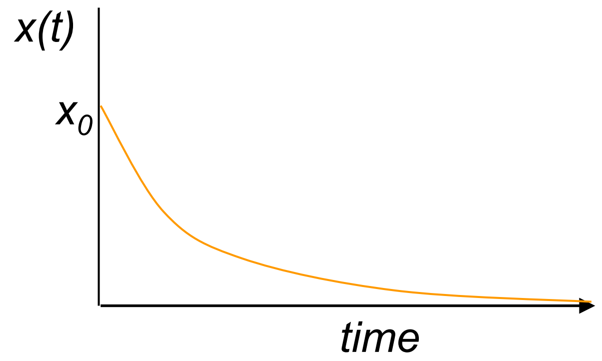
Linear Systems

- If *a* is negative and we can plot out $x = x_0\, exp(at)$, we get much different results:



Linear Systems

- This exponential decay informs us that the state *x* decays to zero over time.
  - We say this system is “STABLE”.
  - We use this property in control theory to drive states down to zero (e.g. if $e = x_{desired} - x$, drive *e* to 0).



Linear Systems

- The above example was a one dimensional linear system (i.e. single state *x*).
- Our system is a multi-dimensional system (i.e. 3 states *x*, *y*, *θ*).
- We need to describe the system with matrices:

$$\dot{x} = Ax$$

where A is a matrix such that $A \in R^{nxn}$

x is a vector such that $x \in R^{1xn}$

Linear Systems

- In this case, the system

$$\dot{x} = Ax$$

is said to be stable if the eigen-values of *A* are less than 0.

- The eigen values of *A*, represented by λ_i, are coefficients that satisfy the equation:

$$Ax_i = \lambda_i x_i$$

for particular states called x_i, called the eigen vectors.

Linear Systems

- We solve for eigen values by noting:

$$(A - \lambda I)x = 0$$

- For this to hold true,

$$\det (A - \lambda I) = 0$$

Linear Systems

- Example:

$$A = \begin{bmatrix} 3 & 6 \\ 1 & 4 \end{bmatrix}.$$

$$\begin{aligned} A - \lambda I &= \begin{bmatrix} 3-\lambda & 6 \\ 1 & 4-\lambda \end{bmatrix} \\ \det(A - \lambda I) &= (3-\lambda)(4-\lambda) - 6 \\ &= \lambda^2 - 7\lambda + 6 \\ &= (\lambda - 6)(\lambda - 1) \end{aligned}$$

Therefore $\lambda_1 = 6$, $\lambda_2 = 1$

The system is not stable!

Linear Systems

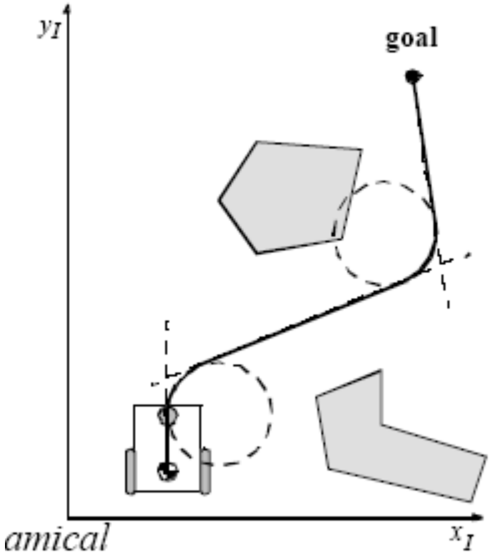
- Summary:
  - If our robot behaves like a system of the form $\dot{x}=Ax$, where the eigen values of A are negative and x represents the difference between desired and actual states, the system will move to our desired state!

Point Tracking

1. P Control
2. Linear Systems
3. Motion Control
4. Reachable Space

Motion Control

- Goal is to follow a trajectory from an initial state to some desired goal location.
- Several approaches
  - Could construct a global trajectory first, then track **points** on the trajectory locally



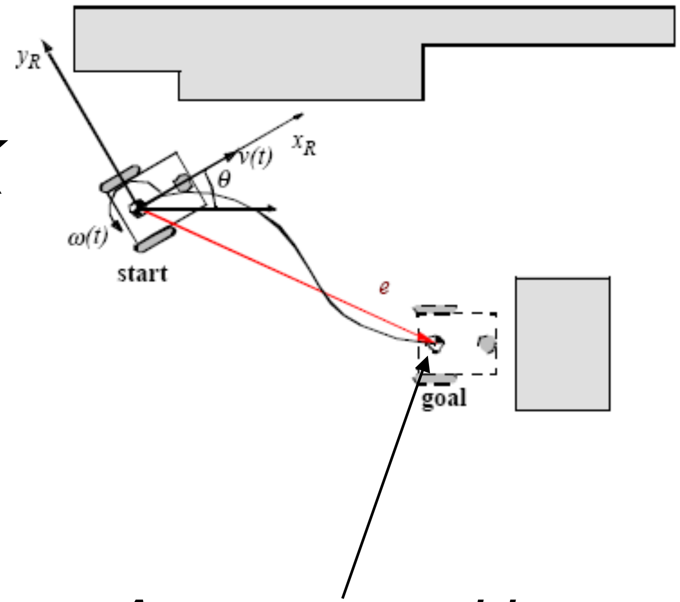
Motion Control

- If we define the error to be in the robot frame:

$$e(t) = [\, x \; y \; \theta \,]^T$$

- Goal is to find gain matrix *K* such that control of *v(t)* and *w(t)* will drive the error *e(t)* to zero.

$$\begin{bmatrix} v(t) \\ w(t) \end{bmatrix} = K e(t)$$



Assume goal is at [0 0 0]

Motion Control

- Recall our forward kinematics

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix}$$

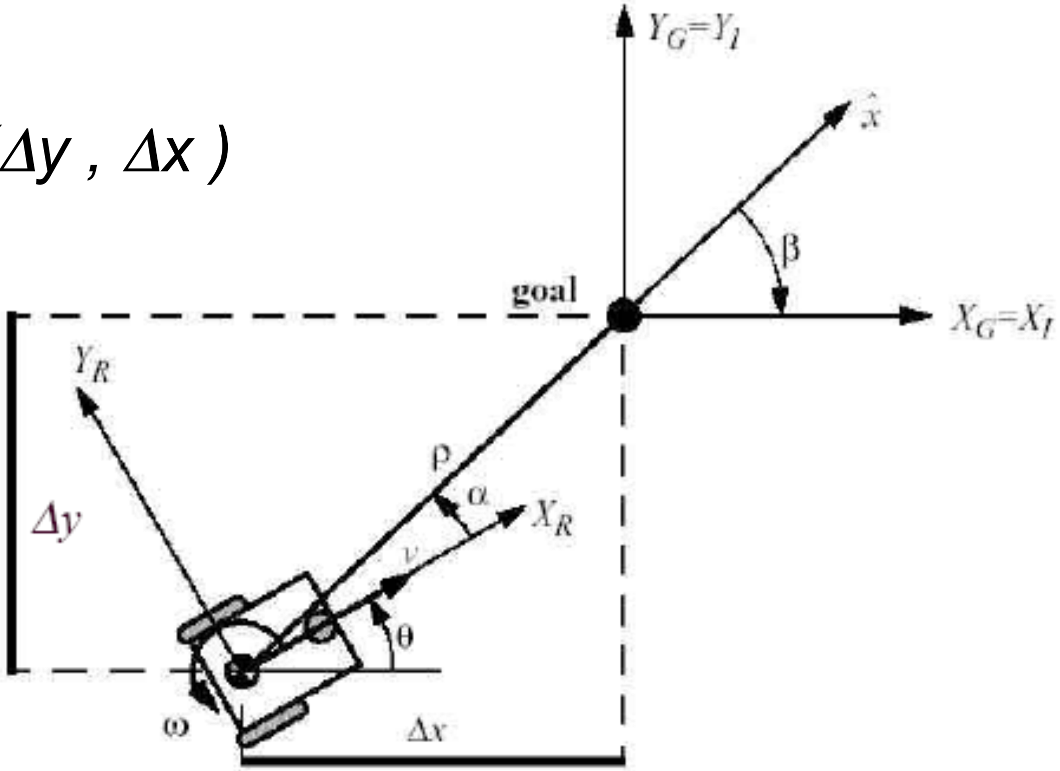
Motion Control

- We use the coordinate transformation

$$\rho = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\alpha = -\theta + atan2(\Delta y , \Delta x)$$

$$\beta = -\theta - \alpha$$



Motion Control

- Now we define the problem as driving the robot to goal

$$\begin{pmatrix} \rho \\ \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Motion Control

- We know this will happen if the dynamics of the system obey

$$\begin{pmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{pmatrix} = A \begin{pmatrix} \rho \\ \alpha \\ \beta \end{pmatrix}$$

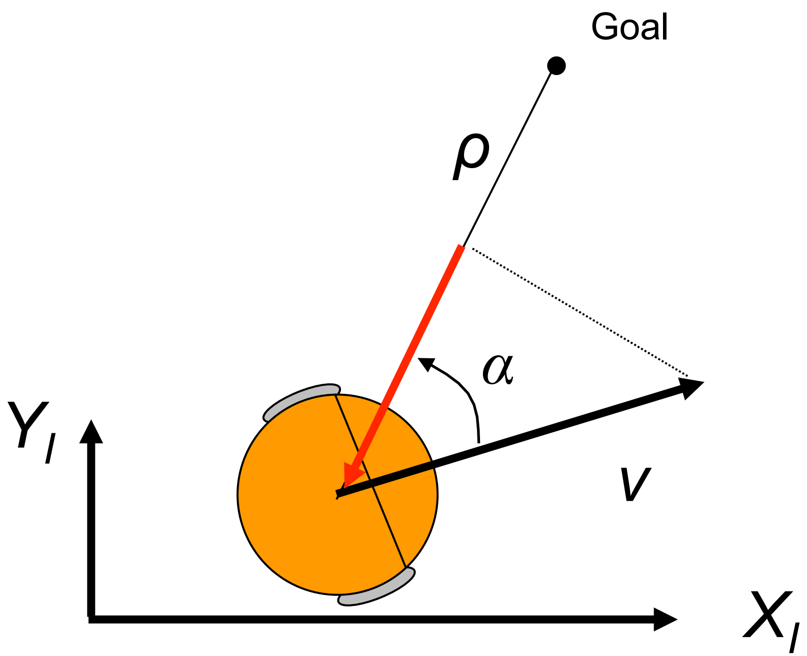
Where *A* is a 3x3 matrix with eigen values less than 0.

Motion Control

- Using the coordinate transformation, calculate the new kinematics:

$\dot{\rho}$ = *projection of v on ρ*

= $- v \cos(\alpha)$



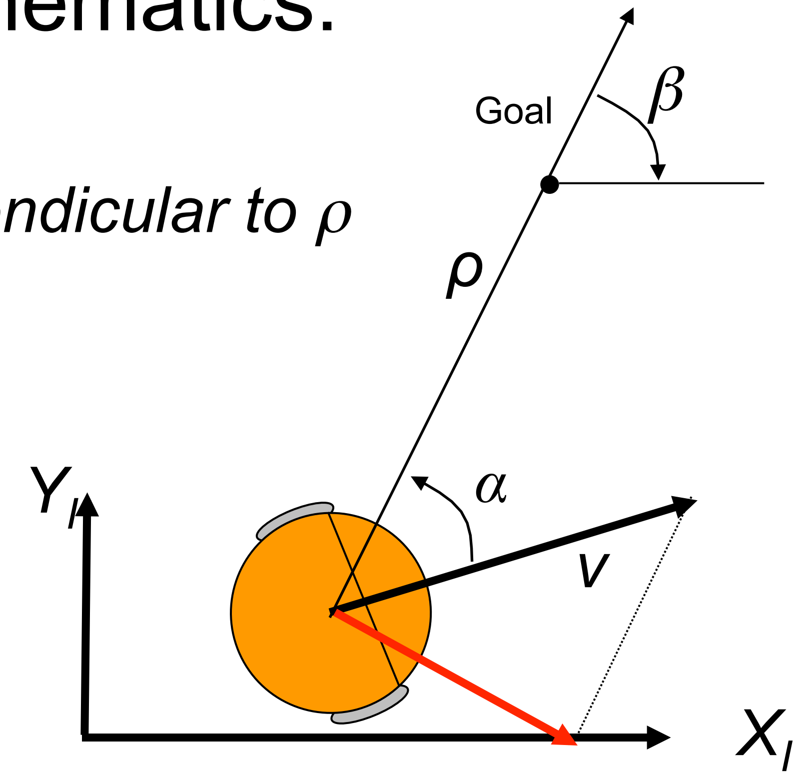
Motion Control

- Using the coordinate transformation, calculate the new kinematics:

$$\rho\dot{\beta} = \text{projection of } v \text{ perpendicular to } \rho$$

$$= -v \sin(\alpha)$$

$$\dot{\beta} = -v \sin(\alpha) / \rho$$



Motion Control

- Using the coordinate transformation, calculate the new kinematics:

$$\alpha = -\beta - \theta$$

$$\dot{\alpha} = -\dot{\beta} - \dot{\theta}$$

$$\dot{\alpha} = v \sin(\alpha)/\rho \; - \; w$$

Motion Control

- In matrix from:

$$\begin{pmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{pmatrix} = \begin{pmatrix} -\cos\alpha & 0 \\ \sin\alpha / \rho & -1 \\ -\sin\alpha / \rho & 0 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix}$$

for α within $(-\pi/2, \pi/2\,]$

Motion Control

- Let's try the control law:

$$v = k_\rho \rho \qquad w = k_\alpha \alpha + k_\beta \beta$$

- Note that this is a form of P control, and if ρ, α, β all go to zero, then *v* and *w* will go to zero.

Motion Control

- To analyze controller, substitute control law into kinematics and linearize:
  - For small x, cosx $\approx$ 1 and sinx $\approx$ x

$$\begin{pmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{pmatrix} = \begin{pmatrix} -k_\rho & 0 & 0 \\ 0 & -(k_\alpha - k_\rho) & -k_\beta \\ 0 & -k_\rho & 0 \end{pmatrix} \begin{pmatrix} \rho \\ \alpha \\ \beta \end{pmatrix}$$

  - This is in the form…

$$\dot{x} = A x$$

Motion Control

- Check for stability:
  - Take the determinant of *A* and solving for eigen values leads to:

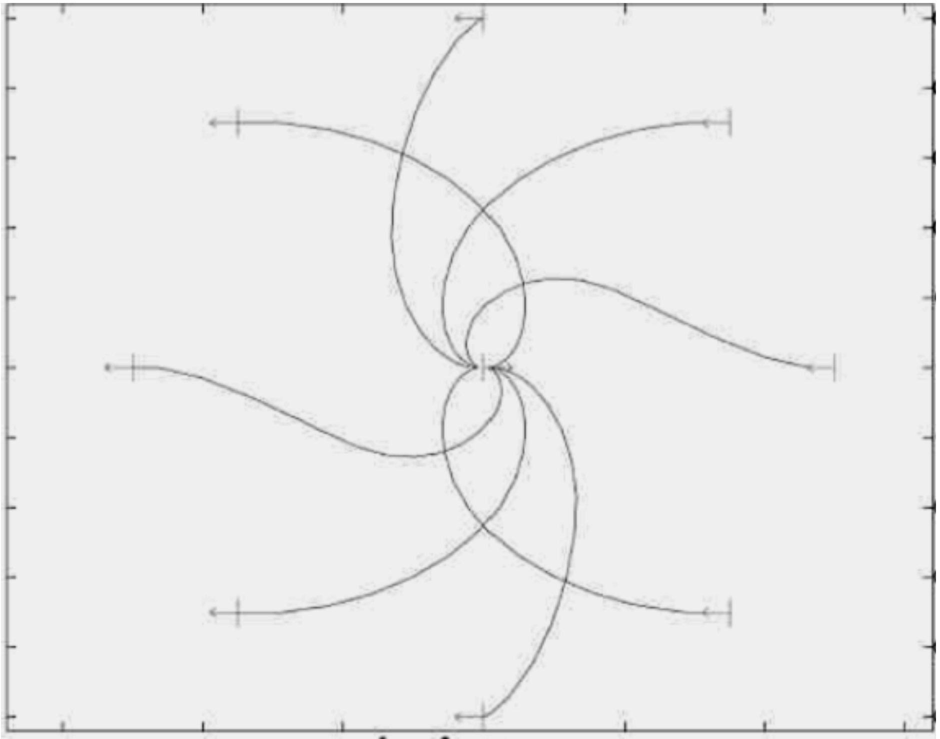
$$(\lambda + k_\rho)(\lambda^2 + \lambda(k_\alpha - k_\rho) - k_\rho k_\beta) = 0$$

  - Thus the system will be stable if:

$$k_\rho > 0 \qquad k_\beta < 0 \qquad k_\alpha - k_\rho > 0$$

Motion Control

- Testing this control law with many different start points:

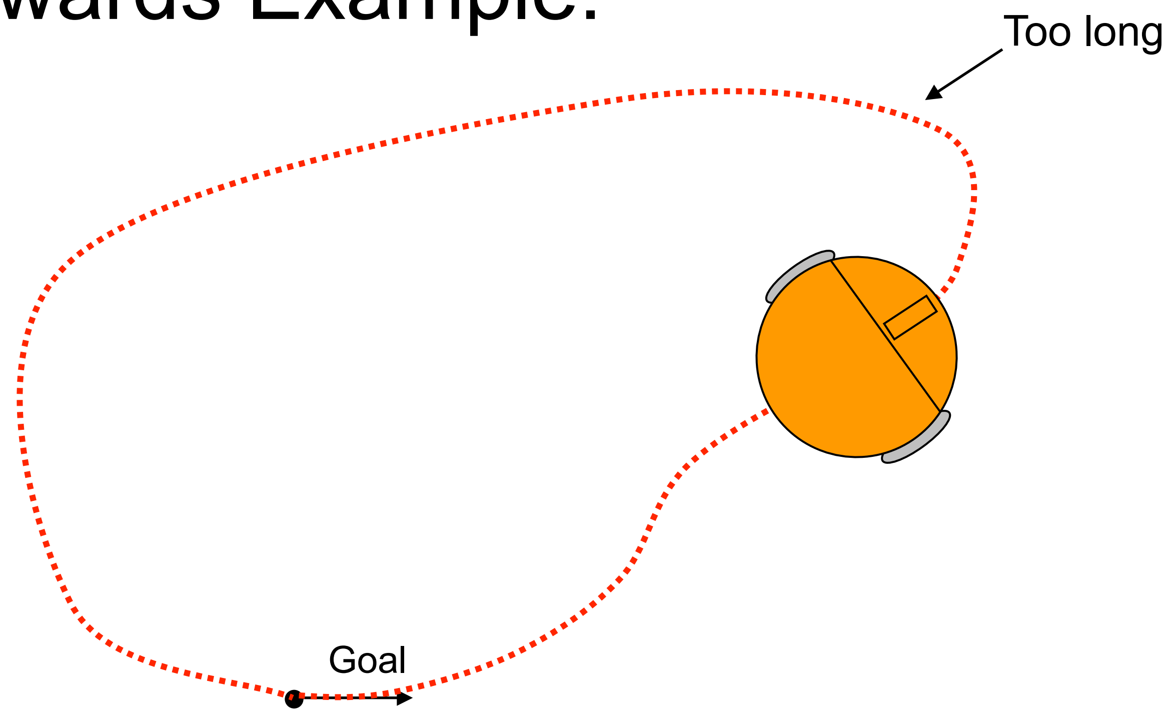


Motion Control

- The derived control law works well if $\alpha \in$ *[-π/2, π/2]*
- For other cases where *abs*(α) > *π/2*, we must modify the controller. So that the robot will move backwards to the desired position when required

Motion Control

- Backwards Example:



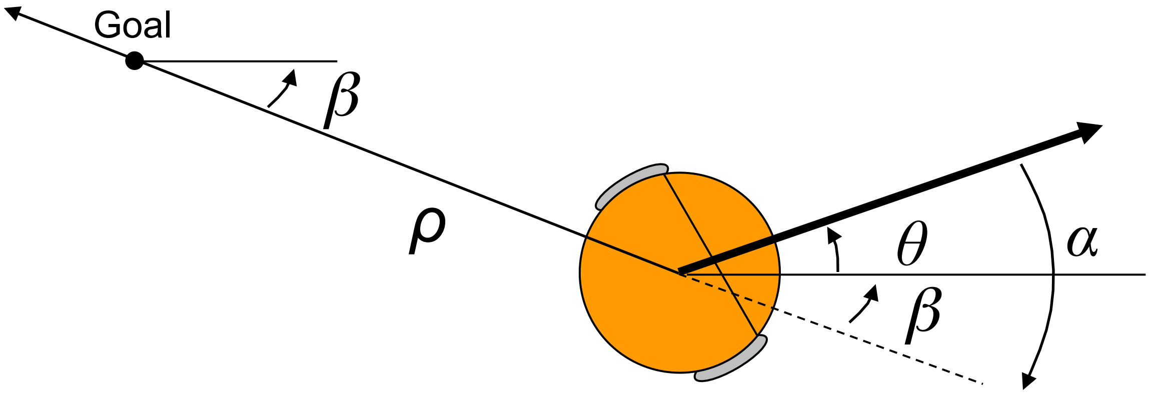
Motion Control

- Backwards Method:

$$\rho = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\alpha = -\theta + atan2(-\Delta y, -\Delta x)$$

$$\beta = -\theta - \alpha$$



Motion Control

- Backwards Method Summary:
  - If $\alpha \in [-\pi/2, \pi/2]$
    - Use regular transform to polar coordinates
    - Use control law: $v = k_\rho \rho$ $w = k_\alpha \alpha + k_\beta \beta$
  - Else
    - Redefine α as shown in backwards method
    - Use control law: $v = -k_\rho \rho$ $w = k_\alpha \alpha + k_\beta \beta$

Point Tracking

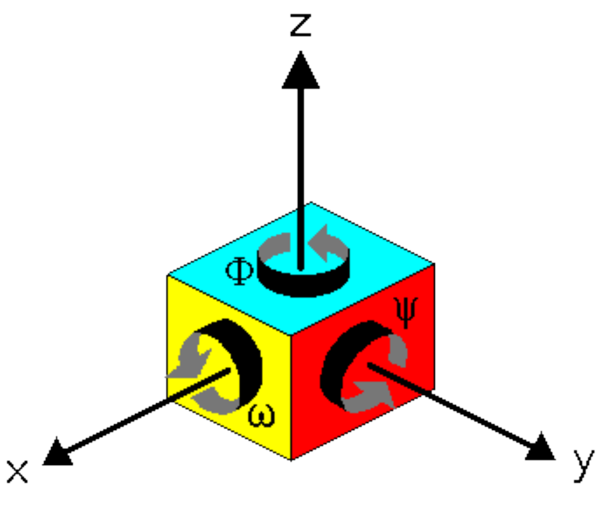
1. P Control
2. Linear Systems
3. Motion Control
4. Reachable Space

Reachable Space

- Kinematic Constraints
  - One can calculate constraints on each individual wheel, then combine for constraints on entire robot.
- Two main constraints:
  - Rolling Constraint: no slipping!
  - Sliding Constraint: no lateral movement!

Reachable Space

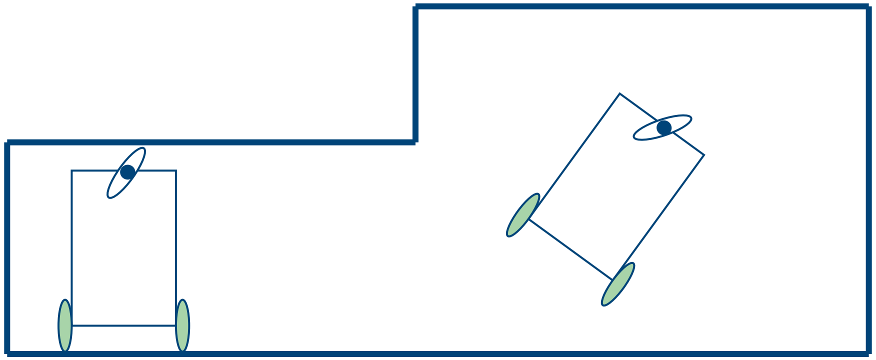
- Degrees of Freedom:
  - Def' n: The number of coordinates that it takes to uniquely specify the state of a system.
  - In 3D, there are 6 degrees of freedom associated to the movement of a rigid body: 3 for its position, and 3 for its orientation.



B J Stone, Univ. of Western Australia

Reachable Space

- Configurations in the Workspace
  - A robot’s workspace is defined by the Degrees Of Freedom of the robot state.
  - Not all robot configurations within the workspace are reachable

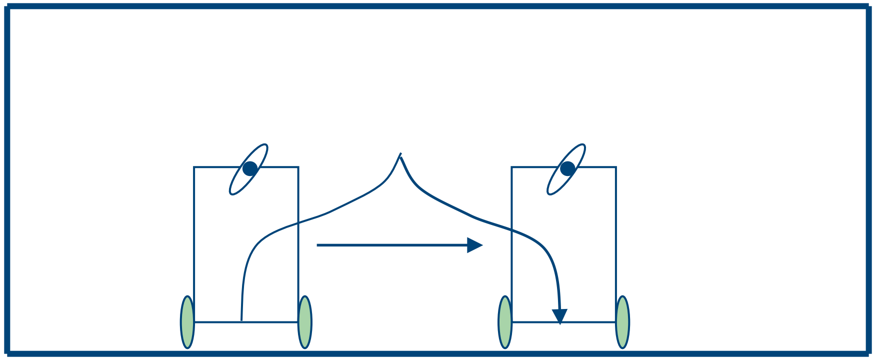


Reachable Space

- Holonomic Robots
  - A robot is holonomic if it has zero nonholonomic constraints.
  - A nonholonomic constraint is one that is not integrable.

Reachable Space

- Paths in the Workspace
  - Path's in the workspace are limited, especially if the robot is nonholonomic



Reachable Space

- Trajectories in the Workspace
  - A trajectory is a path parameterized by time.
  - Admissible paths don't always lead to admissible trajectories.