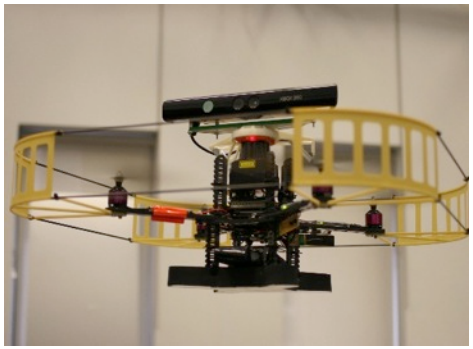
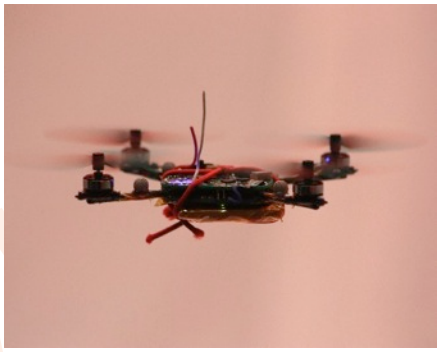
IDETC/CIE T3

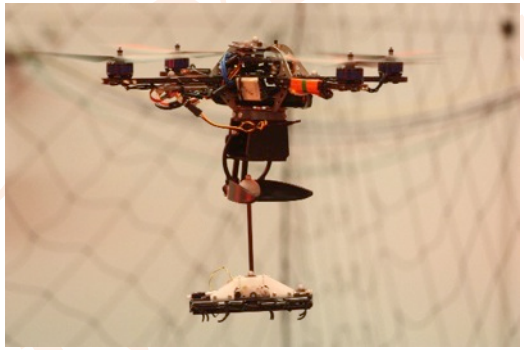
Kinematics, Dynamics, and Control of Quadrotors

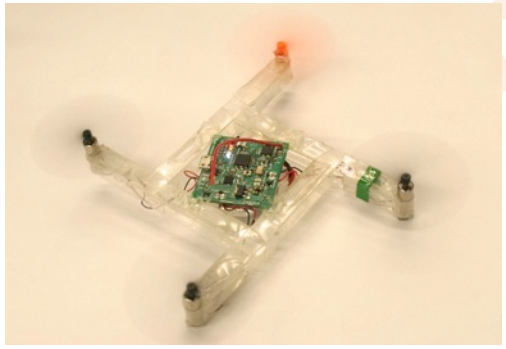
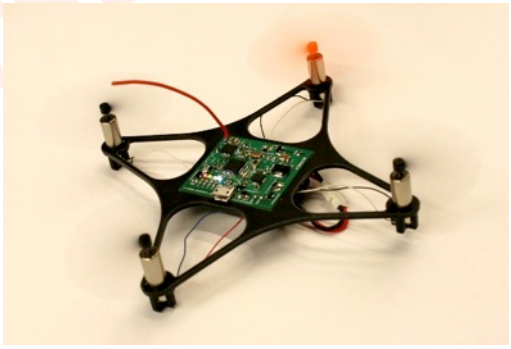
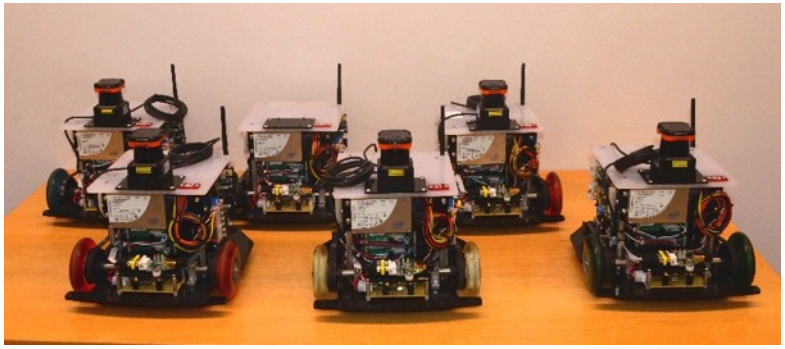
Vijay Kumar, Justin Thomas,
Mickey Whitzer, Yash Mulgaonkar

Designs @ MRSL





Logistics

- 3 Sections
  - Intro, Mechanics, and Control
  - Quadrotor Design & Laboratory Setup
  - Trajectory Generation & Quadrotors in the Real World
- Relaxed
  - Please stop us for questions or comments
- Website (http://tinyurl.com/idetc2014-T3)
  - Outline
  - Presenters
  - MATLAB Simulator
  - Presentation Slides

Outline – Section I

- Intro (Justin Thomas)
- Basics: Mechanics and Control (Vijay Kumar)
  - Transformations & Rotations
    - Euler Angles, Rotation Matrices, Quaternions, Transformation Matrices
  - Quadrotor Model & Dynamics
    - Newton's & Euler's Equations of Motion
  - Control
    - Linear controller, nonlinear controller, experimentation
- Break

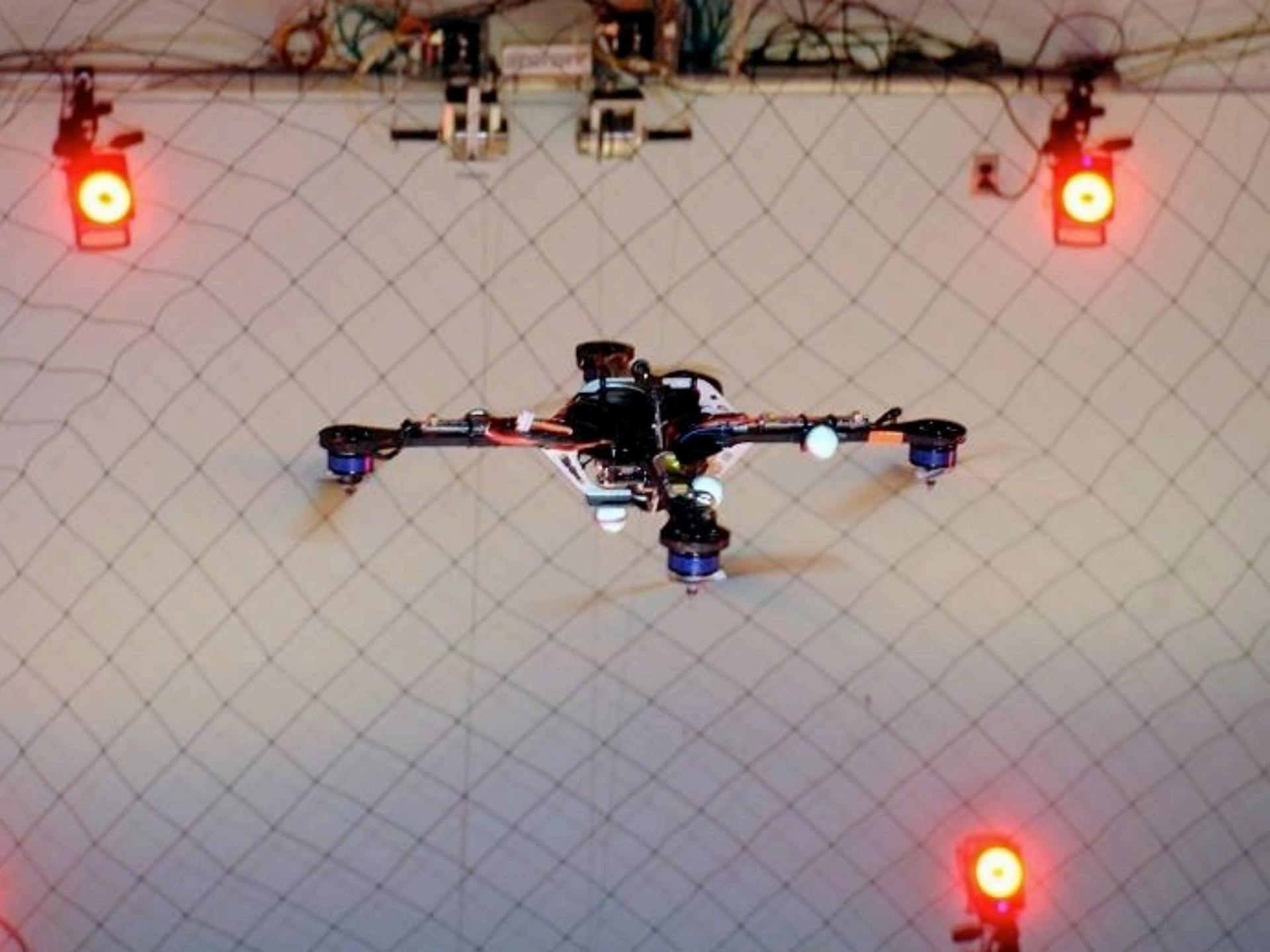
Outline – Section II

- Demonstration
- Design Tradeoffs (Yash Mulgaonkar)
  - Kinematics, power, component analysis, designing your own quadrotor
- Demonstration
- Sensors – Part I (Justin Thomas)
- Quads in the Lab (Mickey Whitzer & Yash Mulgaonkar)
  - The UAV Testbed, MATLAB Simulator, gain tuning
- Break

Outline – Section III

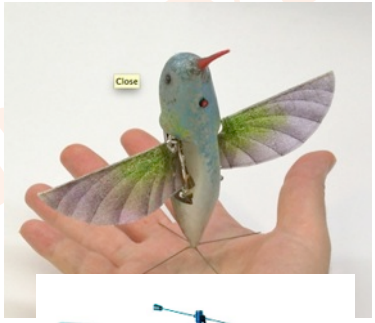
- Demonstration
- Trajectory Generation (Justin Thomas)
  - Differential Flatness and its impliciations
- Quadrotors in the Real World
  - Sensors Part II (Justin Thomas)
  - Planning Algorithms (Vijay Kumar)
    - Dijkstra, A*, RRTs
  - Estimation (Vijay Kumar)
    - Filtering
  - Vision-based localization and control
- Discussion & Close

Transformations & Rotations



Types of Micro Air Vehicles

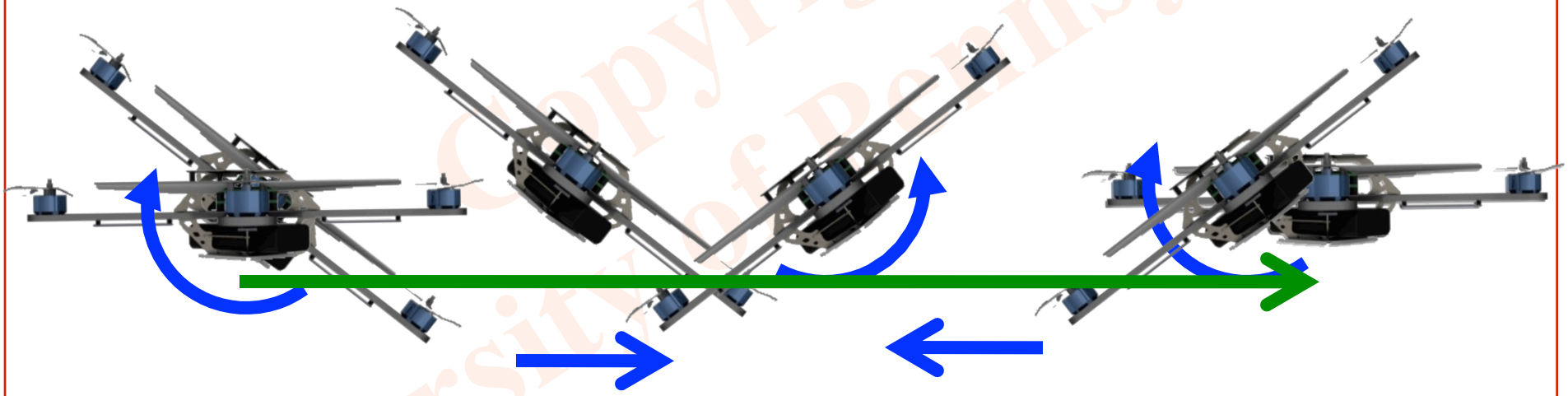
- Fixed wing
- Flapping wing
  - Avian flight
  - Insect flight



- Rotor crafts
  - Helicopter
  - Ducted fan
  - Co-axial
  - Quad rotor
  - Hexa rotor

Roll and Pitch

Translation



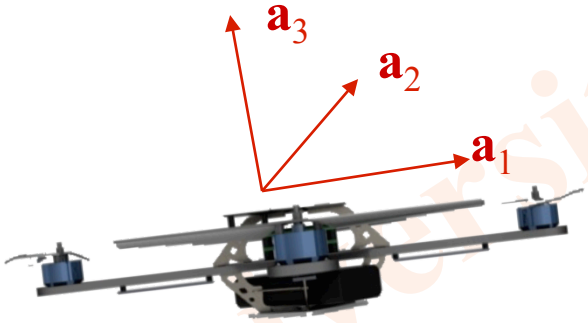
Reference Frames

We associate with any position and orientation a *reference frame*

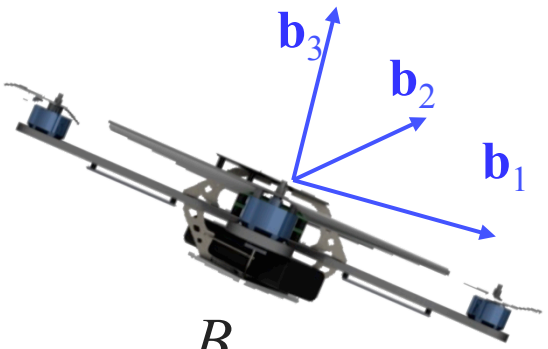
In reference frame $\{A\}$, we can find three linearly independent vectors $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$ that are basis vectors.

We can write any vector as a linear combination of the basis vectors in either frame.

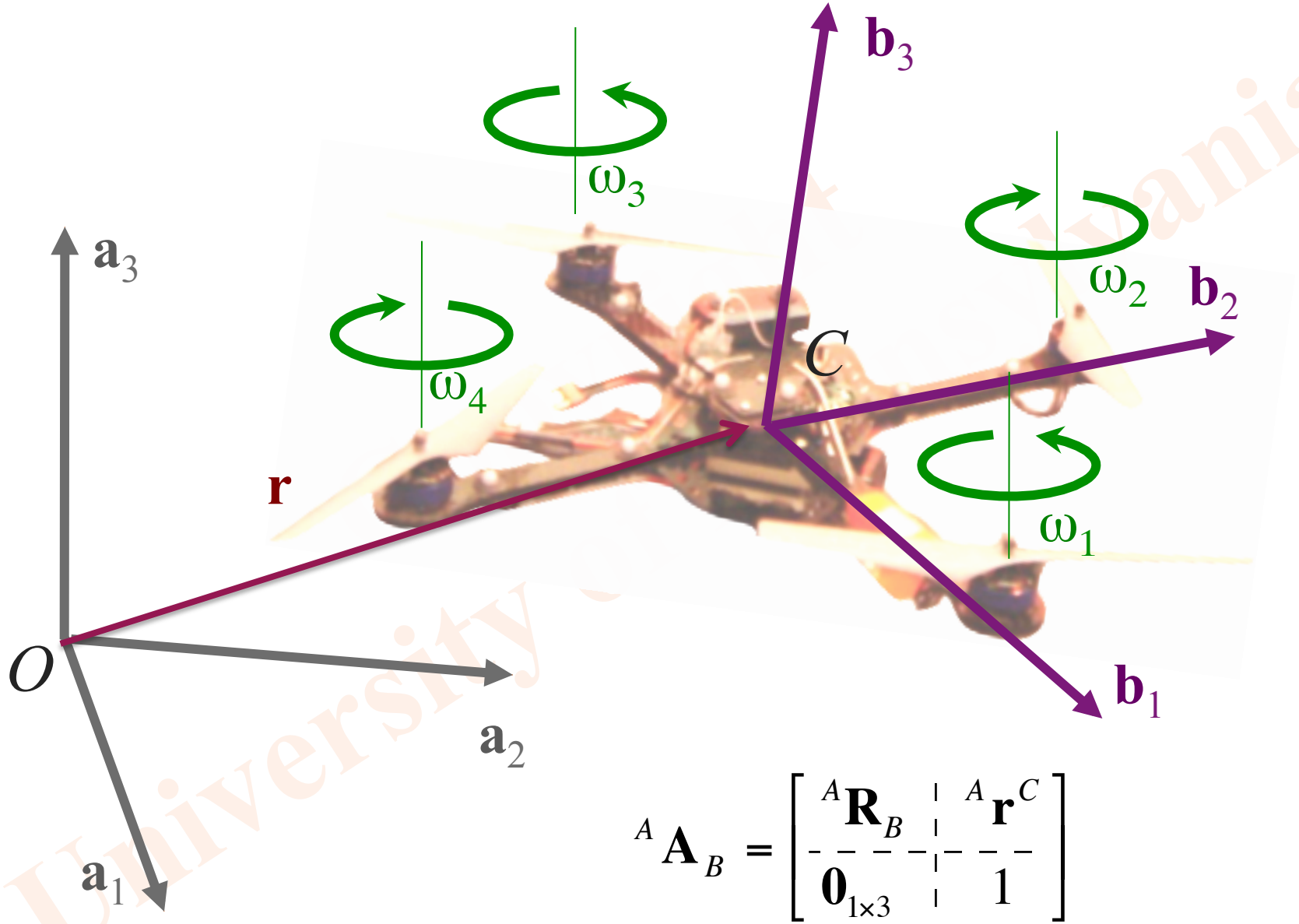
$$\mathbf{v} = v_1\,\mathbf{a}_1 + v_2\,\mathbf{a}_2 + v_3\mathbf{a}_3$$



A



B



$$ {}^{A}\mathbf{A}_{B} = \left[\begin{array}{c|c} {}^{A}\mathbf{R}_{B} & {}^{A}\mathbf{r}^{C} \\ \hline \mathbf{0}_{1\times 3} & 1 \end{array}\right] $$

$$ g_{AB} $$

The group of rotations, $SO(3)$

$$SO(3) = \left\{ R \in \mathbb{R}^{3\times 3} \mid R^T R = R R^T = I,\ \det R = 1 \right\}$$

$SO(3)$ satisfies the four axioms that must be satisfied by the elements of an *algebraic group*:

- Closure under the binary operation
- Associativity
- $SO(3)$ includes the identity element
- $SO(3)$ includes the inverse of every element

$SO(3)$ is a *continuous group*.

the binary operation above is a continuous operation

the inverse of any element is a continuous function of that element.

The Lie group $SE(3)$

$$SE(3) = \left\{ \mathbf{A} \;\middle|\; \mathbf{A} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{r} \\ \hline \mathbf{0}_{1\times 3} & 1 \end{array}\right],\ \mathbf{R} \in R^{3\times 3},\ \mathbf{r} \in R^{3},\ \mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}, |\mathbf{R}| = 1 \right\}$$

the set of all rigid body displacements (transformations)

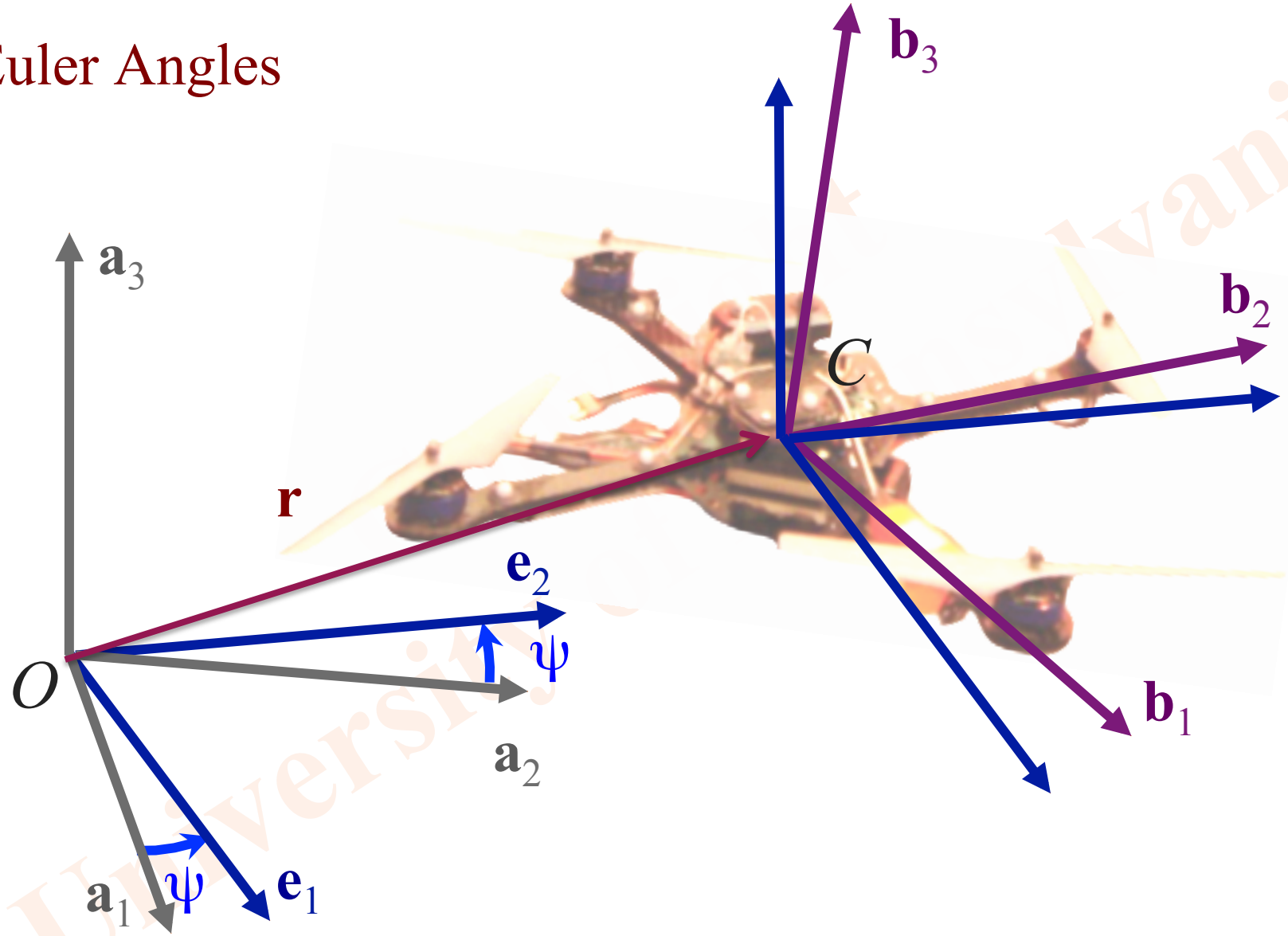
Properties of a Rotation Matrix

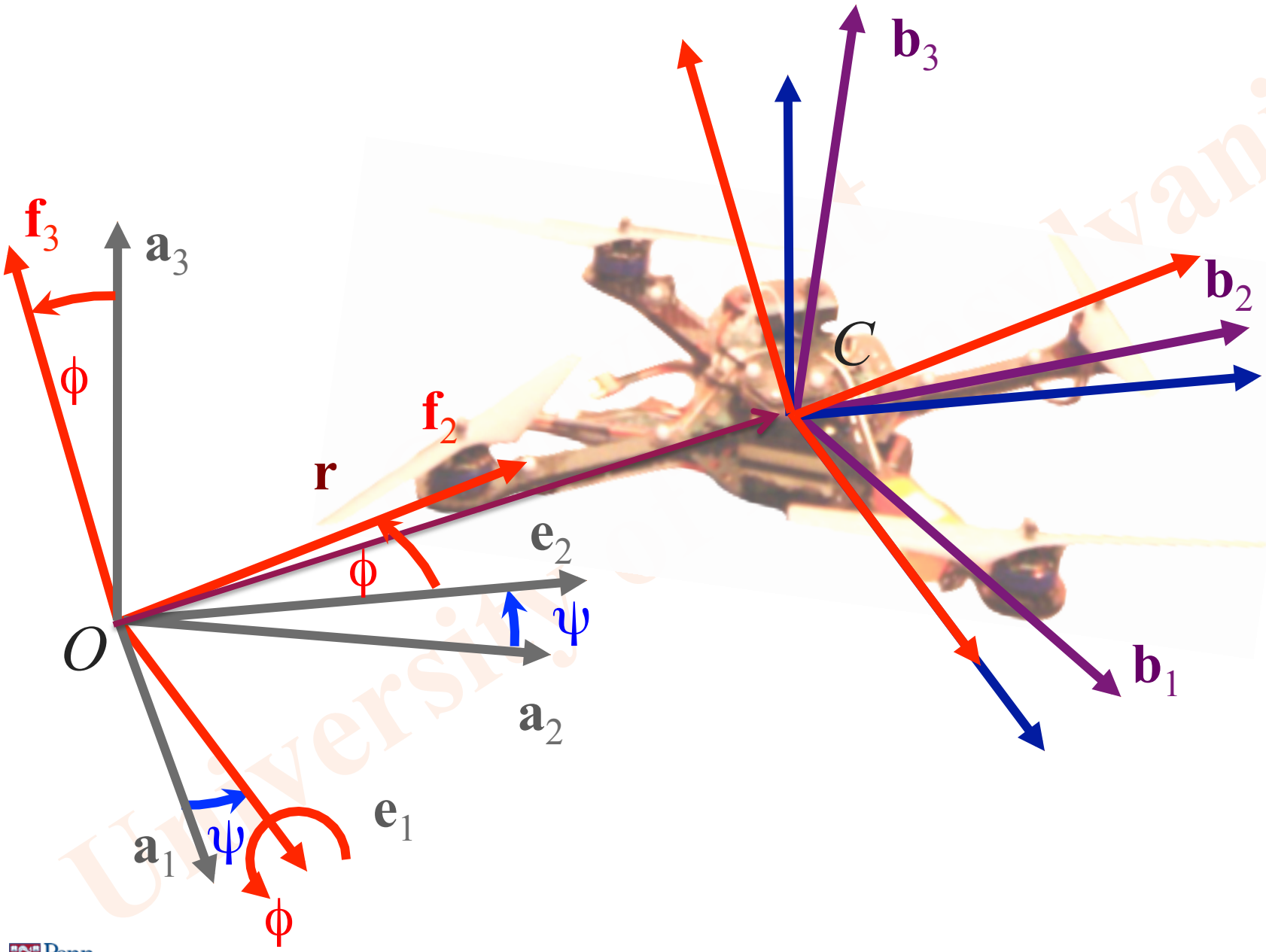
- Orthogonal
  - Matrix times its transpose equals the identity
- Special orthogonal
  - Determinant is +1
- Closed under multiplication
  - The product of any two rotation matrices is another rotation matrix
- The inverse of a rotation matrix is also a rotation matrix
- The set of all rotations is a group

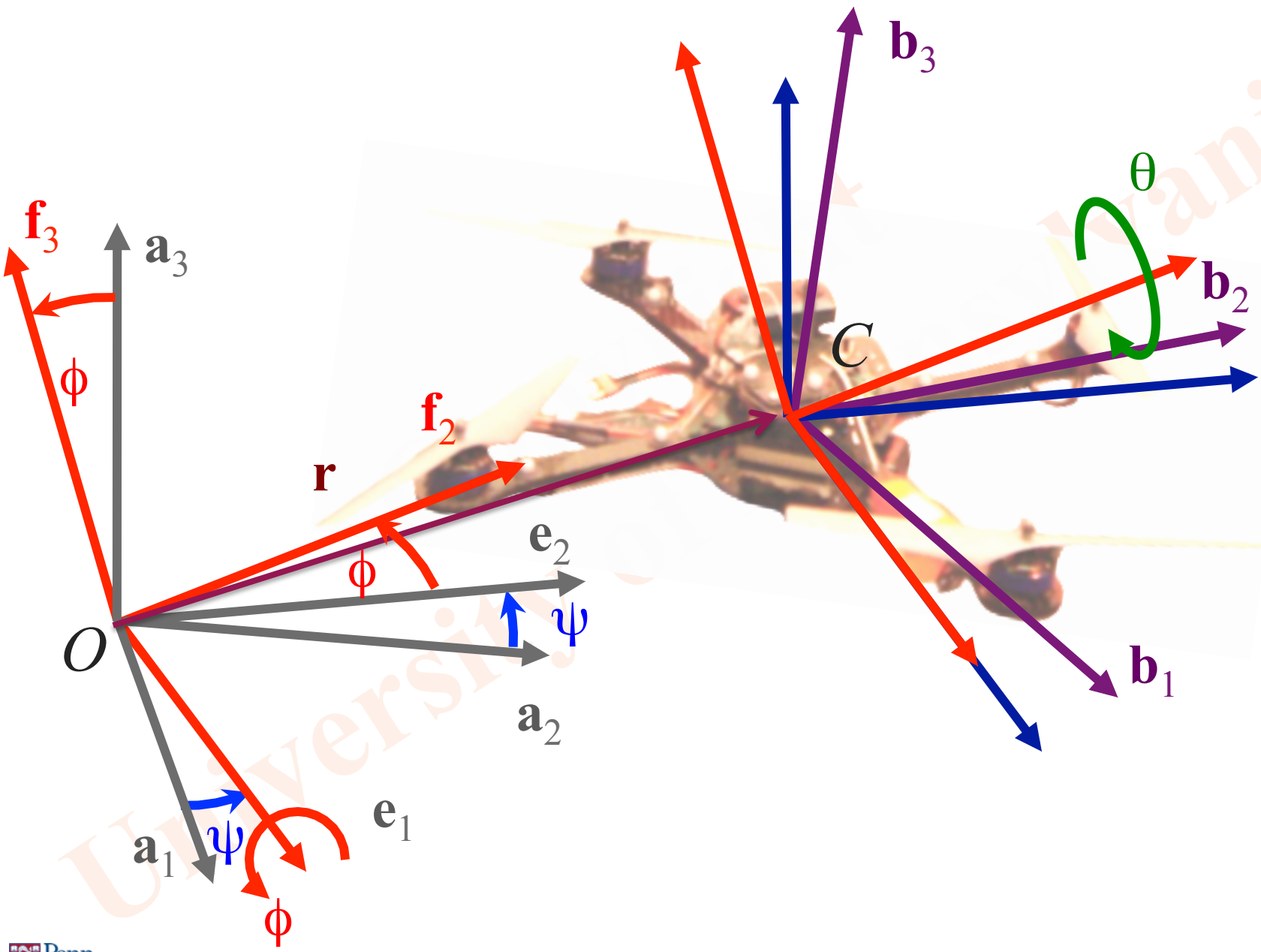
Beyond rotation matrices

- $SO(3)$ is a Lie group, properties of a differentiable manifold
- Coordinates for $SO(3)$
  1. Rotation matrices
  2. Euler angles
  3. Axis angle parameterization
  4. Exponential coordinates
  5. Quaternions

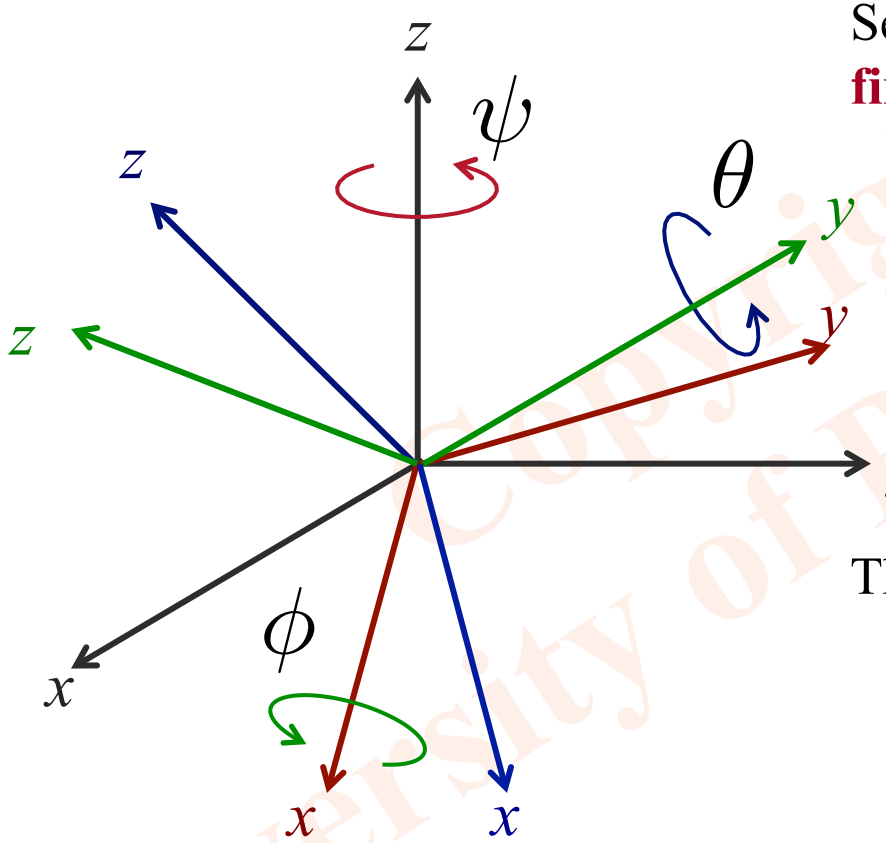
Euler Angles




$\mathbf{b}_3$
$\mathbf{f}_3$
$\mathbf{a}_3$
$\mathbf{b}_2$
C
ϕ
$\mathbf{f}_2$
$\mathbf{r}$
$\mathbf{e}_2$
ϕ
ψ
O
$\mathbf{b}_1$
$\mathbf{a}_2$
$\mathbf{e}_1$
$\mathbf{a}_1$
ψ
ϕ



Z-X-Y Euler Angles



Sequence of three rotations about **body-fixed** axes

- Rot(z, ψ)
- Rot(x, ϕ)
- Rot(y, θ)

Three Euler Angles

- ϕ, θ, and ψ
- Parameterize rotations

Note there are singularities!

$$\mathbf{R} = \text{Rot}(z, \psi) \times \text{Rot}(x, \phi) \times \text{Rot}(y, \theta)$$

Euler Angles

$$Rot(z,\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Rot(x,\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$Rot(y,\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$${}^{\mathcal{A}}[R]_{\mathcal{B}} = \begin{bmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix}$$

Angular Velocities

Angular Velocity Matrix

$$\hat{\omega} = {}^{\mathcal{A}}[R]_{\mathcal{B}}^{T} \; {}^{\mathcal{A}}[\dot{R}]_{\mathcal{B}}$$

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \longleftrightarrow \omega = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

"Absolute" angular velocity of the quadrotor but with components in the body-fixed frame

Errors in rotations

Given two rotation matrices, what is the distance between them?

or

Given a rotation matrix, *R*, what is the "size" of this rotation?

Fact: any rotation matrix subtracted from its transpose is skew!

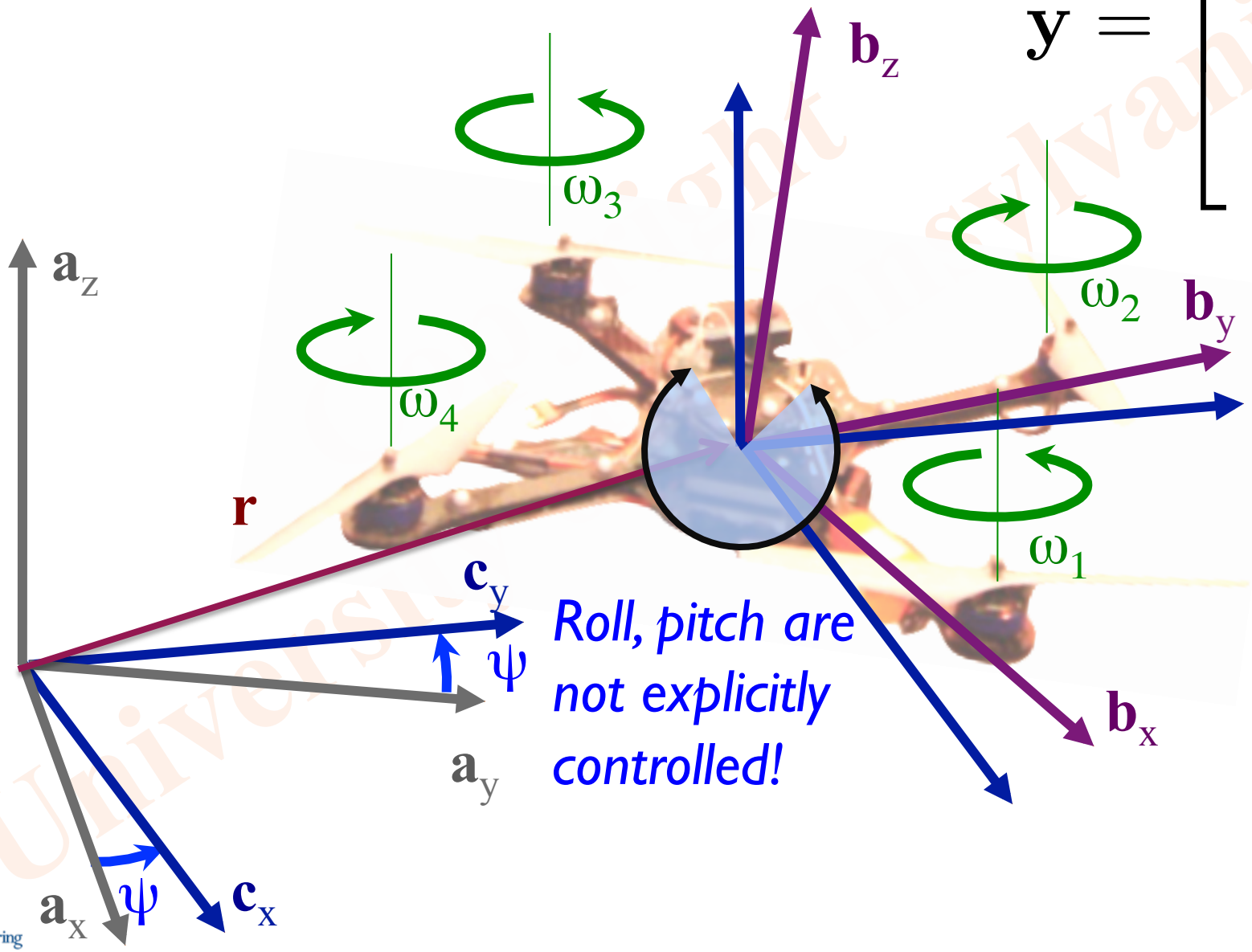
$$R - R^T = 2 \sin \rho \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}$$

Rotation angle

Rotation error $\hat{e}_R = \frac{1}{2}(R_{des}^T {}^{\mathcal{A}}R_{\mathcal{B}} - {}^{\mathcal{A}}R_{\mathcal{B}}^T R_{des})$

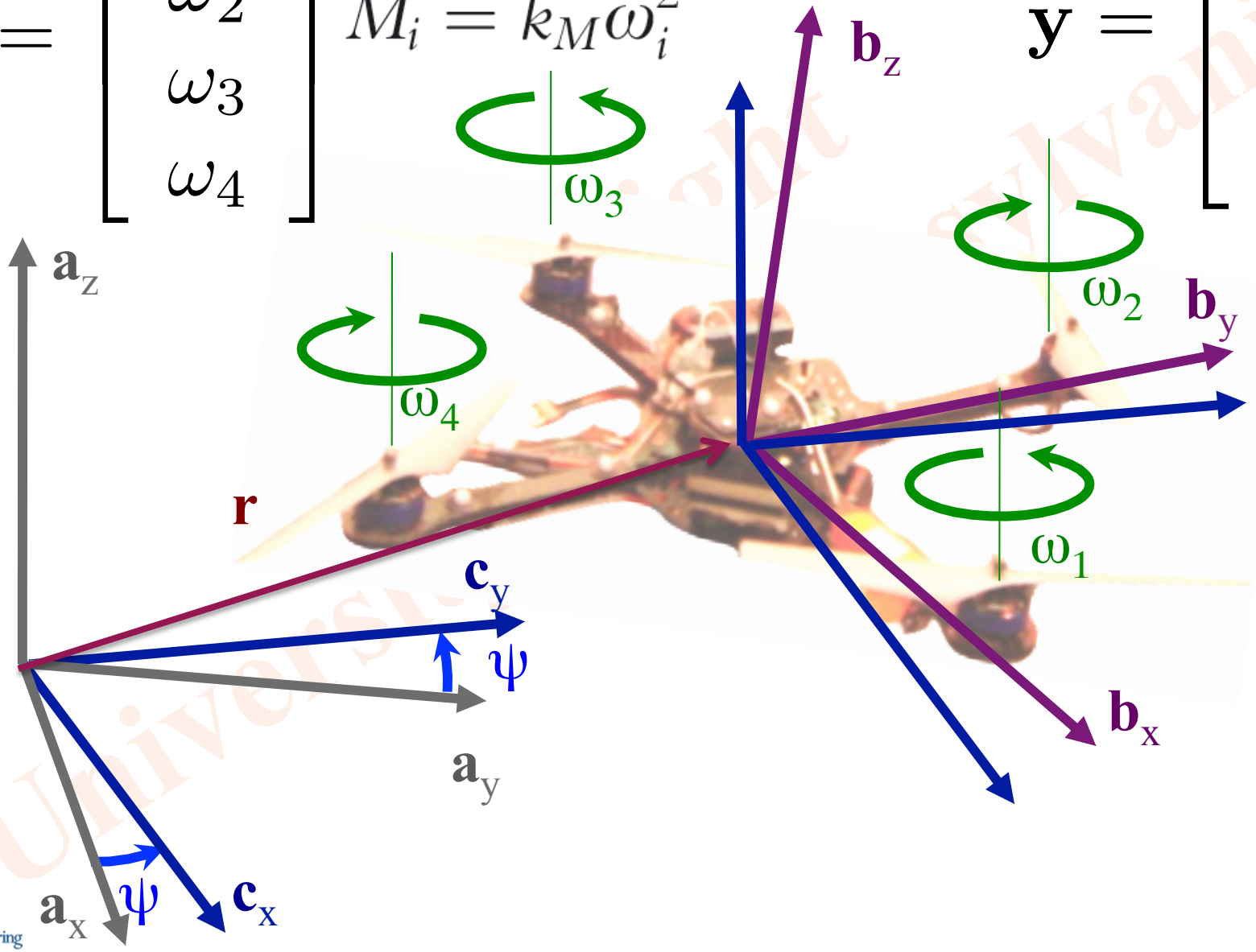
Underactuated System

$$\mathbf{y} = \begin{bmatrix} x \\ y \\ z \\ \psi \end{bmatrix}$$



Roll, pitch are not explicitly controlled!

$$\mathbf{u} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \quad F_i = k_F \omega_i^2 \quad M_i = k_M \omega_i^2 \qquad \mathbf{y} = \begin{bmatrix} x \\ y \\ z \\ \psi \end{bmatrix}$$



$$\begin{bmatrix} a_{C,x} \\ a_{C,y} \\ a_{C,z} \end{bmatrix} = \mathbf{R}^T \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \gamma \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

f

$$\mathbf{I} \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \mathbf{I} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 & \alpha & 0 & -\alpha \\ -\alpha & 0 & \alpha & 0 \\ \beta & -\beta & \beta & -\beta \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

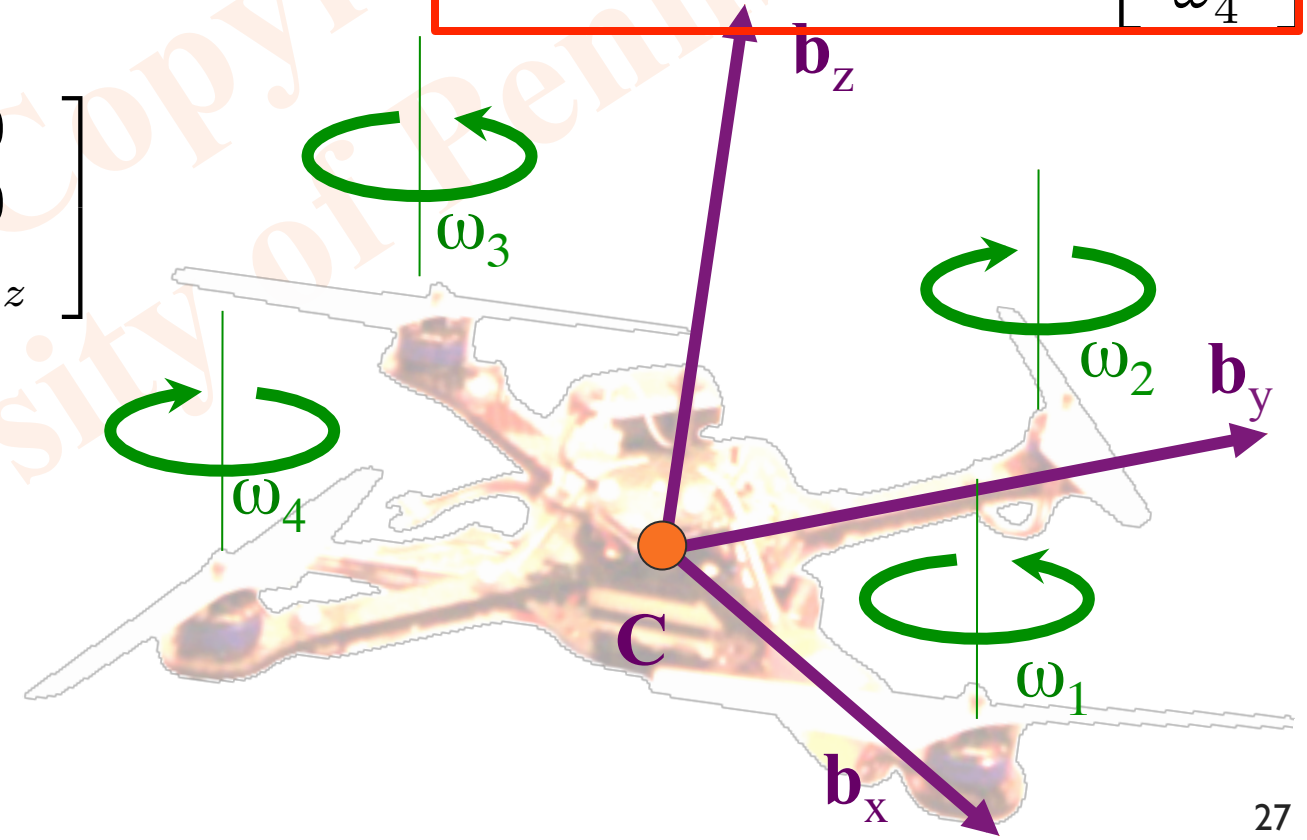
M

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

$$\alpha = k_F L$$

$$\beta = k_M$$

$$\gamma = \frac{k_F}{m}$$



Introduction to Control Theory

- Order of a system
- Linear Time Invariant systems
  - Single integrator (kinematic)
  - Double integrator
  - Feedforward, feedback control
- Linear controller for a quadrotor
- Nonlinear, affine systems

Control of a simple first-order system

Problem

State, input

$$x,\ u\ \in \mathbb{R}$$

Kinematic plant model

$$\dot{x} = u$$

Want x to follow trajectory $x^{des}(t)$

General Approach

Define error, $e(t)=x^{des}(t)- x(t)$

Want $e(t)$ to converge exponentially to zero

Strategy

Find u such that

$$\dot{e} + K_p e = 0 \qquad K_P > 0$$

$$u(t) = \dot{x}^{des}(t) + K_P e(t)$$

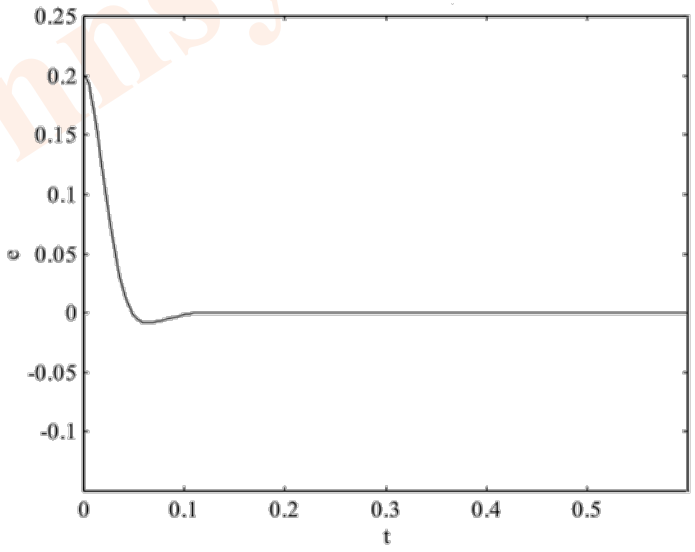
Control of a simple second-order system

Problem

State, input $x,\ u \in \mathbb{R}$

Kinematic plant model $\ddot{x} = u$

Want x to follow trajectory $x^{des}(t)$



General Approach

Define error, $e(t)=x^{des}(t)- x(t)$

Want $e(t)$ to converge exponentially to zero

Strategy

Find u such that

$$\ddot{e} + K_v \dot{e} + K_p e = 0 \qquad K_p, K_v > 0$$

$$u(t) = \ddot{x}^{des}(t) + K_V \dot{e}(t) + K_P e(t)$$

Feedforward

Derivative

Proportional

Control for trajectory tracking in a simple second-order system

PD control

$$u(t) = \ddot{x}^{des}(t) + K_V \dot{e}(t) + K_P e(t)$$

Proportional control acts like a spring (capacitance) response

Derivative control is a viscous dashpot (resistance) response

Large derivative gain makes the system overdamped and the system converges slowly

PID control

In the presence of disturbances or modeling errors, it is often advantageous to use PID control

$$u(t) = \ddot{x}^{des}(t) + K_V \dot{e}(t) + K_P e(t) + K_I \int_0^t e(\tau) d\tau$$

Integral

PID control generates a third-order closed-loop system

Integral control makes the steady-state error go to zero

Model-based control for trajectory tracking

Disadvantages of PID or PD control schemes

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = f(t)$$

- performance will depend on the model
- need to tune gains to maximize performance

Model based control law

$$f(t) = m\underbrace{\left(\ddot{x}_d(t) + k_p e(t) + k_v \dot{e}(t)\right)}_{\text{feedforward + PD feedback}} + \underbrace{b\dot{x}(t) + kx(t)}_{\text{model based}}$$

model based

Two parts of a model based scheme

- model based part
  - cancel the dynamics of the system
  - specific to the model
- servo based part
  - use PID or PD with feedforward to drive errors to zero
  - independent of the model of the system

Model-based control for trajectory tracking

Model

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = f(t)$$

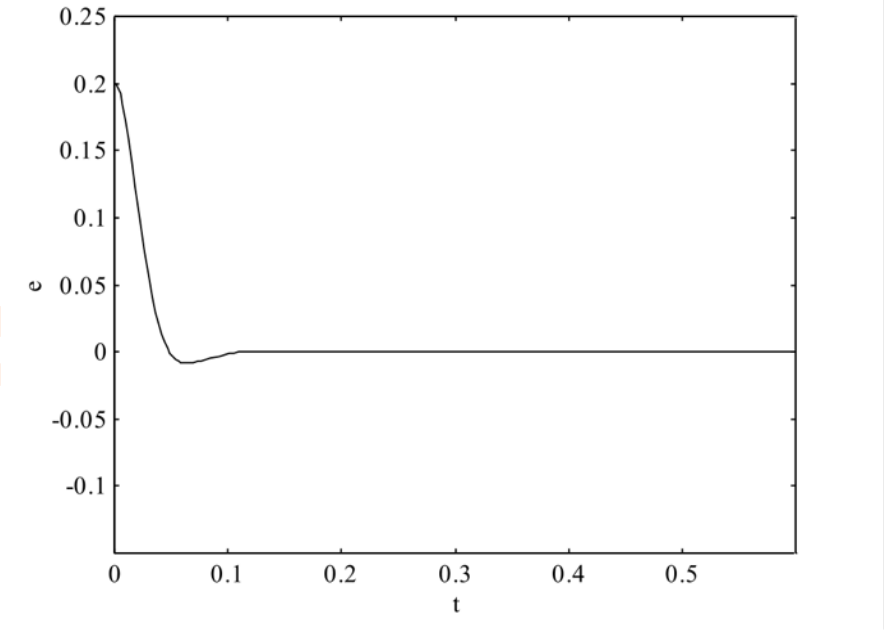
Model based control law

model based

$$f(t) = m\underbrace{\left(\ddot{x}_d(t) + k_p e(t) + k_v \dot{e}(t)\right)}_{\text{servo}} + b\dot{x}(t) + kx(t)$$

Performance

$$\ddot{e} + k_v \dot{e} + k_p e = 0$$



Model-based control for trajectory tracking

- Advantage
  - decomposes the control law into
    - model-dependent part (depends on the knowledge of the model)
    - model-independent part (servo control, gains are independent of the model)
- Disadvantage

Model based control law (based on estimates of model parameters)

$$f(t) = \hat{m}\left(\ddot{x}_d(t) + k_p e(t) + k_v \dot{e}(t)\right) + \hat{b}\dot{x}(t) + \hat{k}x(t)$$

estimates

Ideal performance

$$\ddot{e} + k_v \dot{e} + k_p e = 0$$

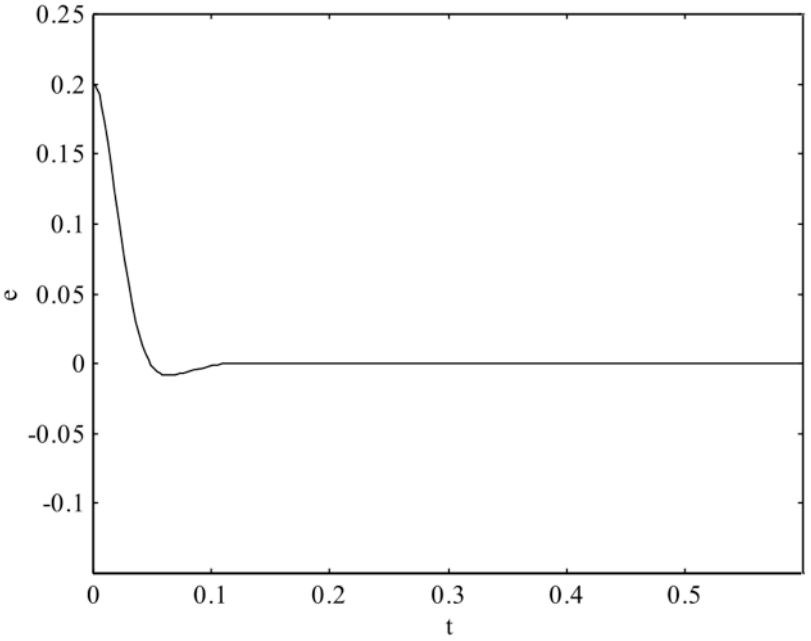
Actual performance

$$\ddot{e} + k_v \dot{e} + k_p e = \left(\frac{m}{\hat{m}} - 1\right)\ddot{x} + \frac{b-\hat{b}}{\hat{m}}\dot{x} + \frac{k-\hat{k}}{\hat{m}}x$$

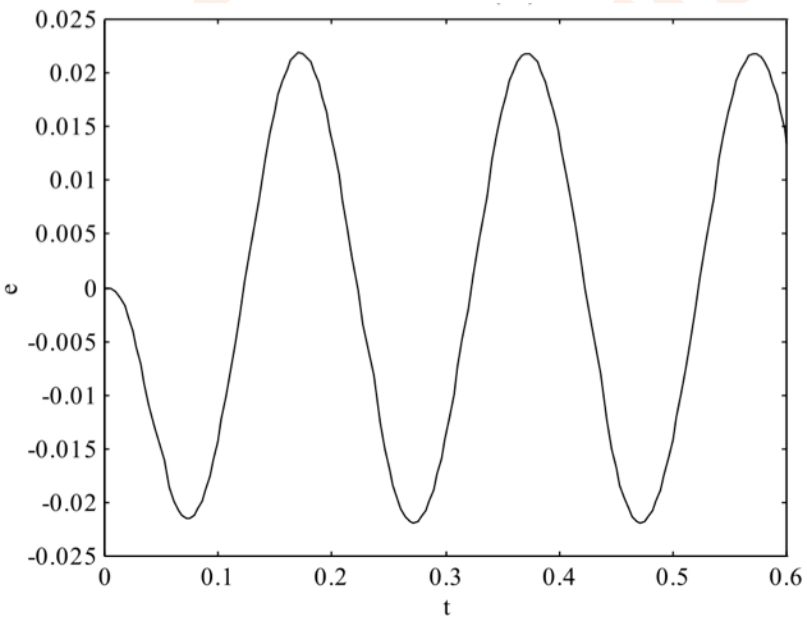
1. Error term will not go exponentially to zero
2. Right hand side is a forcing function driving the error away from zero

Model-based control for trajectory tracking

$$\ddot{e} + k_v \dot{e} + k_p e = \left(\frac{m}{\hat{m}} - 1\right)\ddot{x} + \frac{b-\hat{b}}{\hat{m}}\dot{x} + \frac{k-\hat{k}}{\hat{m}}x \qquad f_p(t)$$



Perfect model



Imperfect model, 10% errors in parameters

Not all is lost however

– Treat f_p as a perturbation or a disturbance force

– If $\max_t f_p(t) < M$ we can prove that the error $e(t)$ is also bounded

$$\begin{bmatrix} a_{C,x} \\ a_{C,y} \\ a_{C,z} \end{bmatrix} = \mathbf{R}^T \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \gamma \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

u_1 u_2

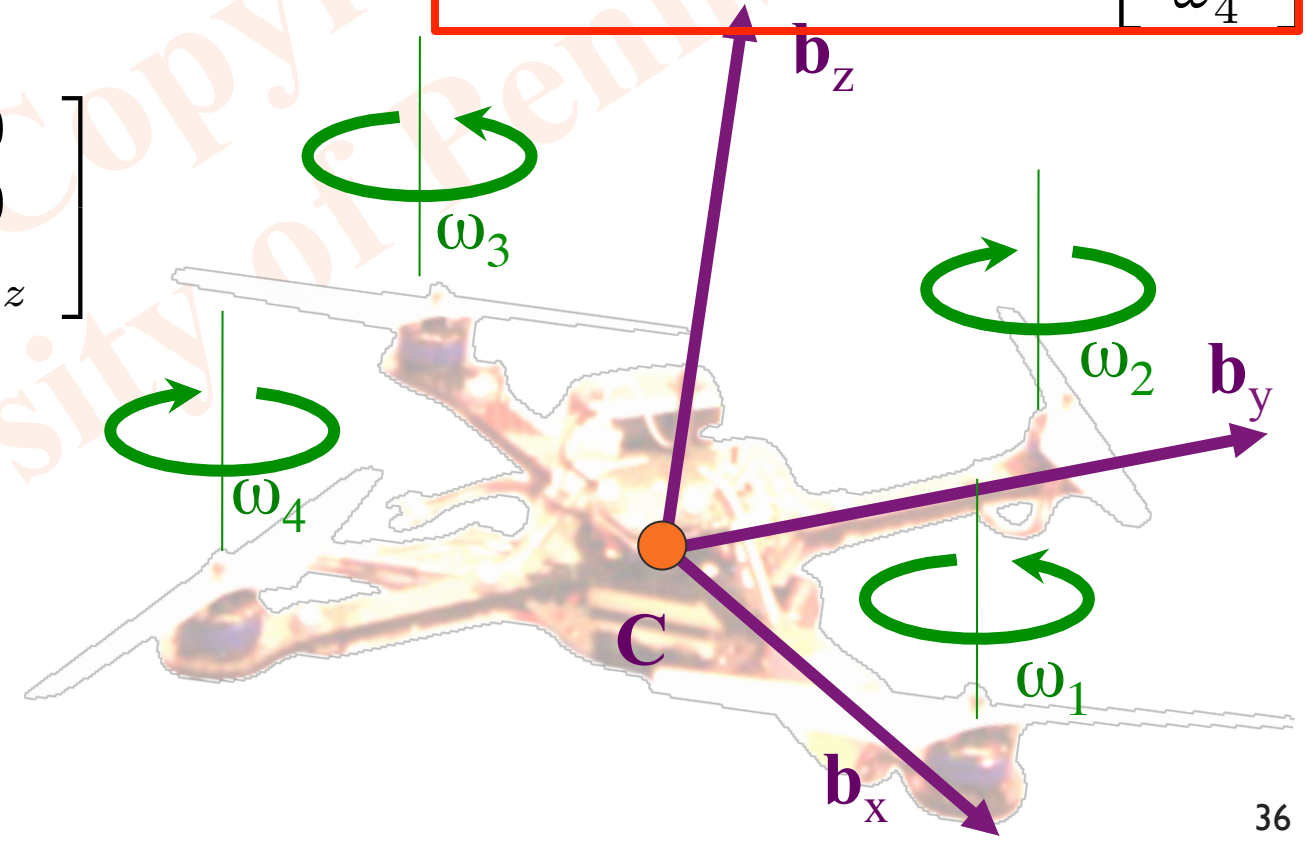
$$\mathbf{I} \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \mathbf{I} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 & \alpha & 0 & -\alpha \\ -\alpha & 0 & \alpha & 0 \\ \beta & -\beta & \beta & -\beta \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

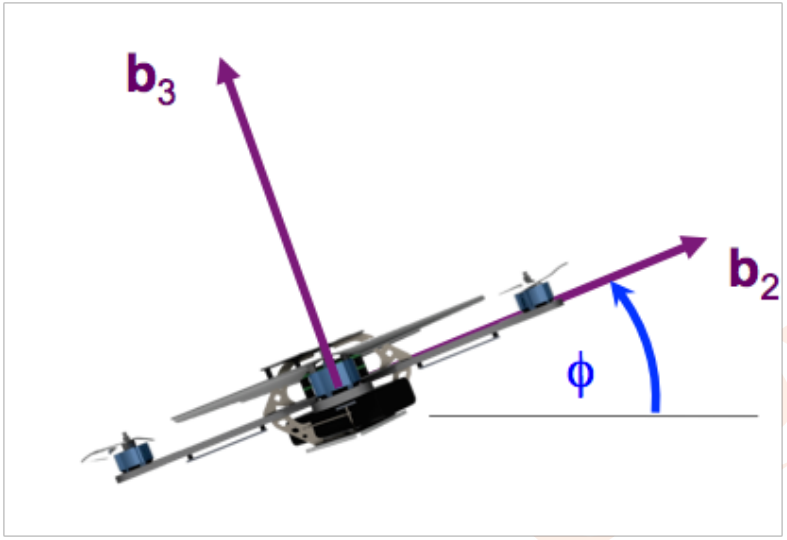
$$\alpha = k_F L$$

$$\beta = k_M$$

$$\gamma = \frac{k_F}{m}$$

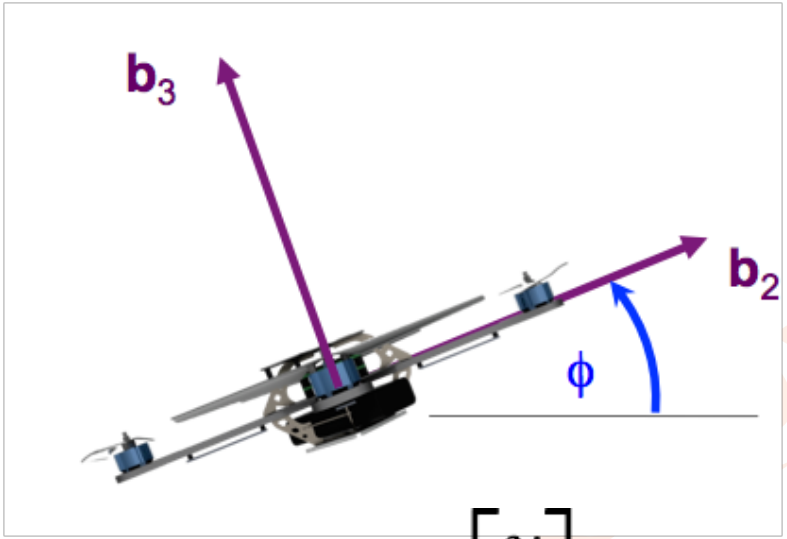


Planar Quadrotor Dynamics



$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Planar Quadrotor Dynamics



$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ z \\ \phi \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} \dot{y} \\ \dot{z} \\ \dot{\phi} \\ 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Affine Nonlinear System

State x, input u

State equations

$$\dot{x} = f(x) + g(x)u$$

Outputs

$$y = h(x)$$

39

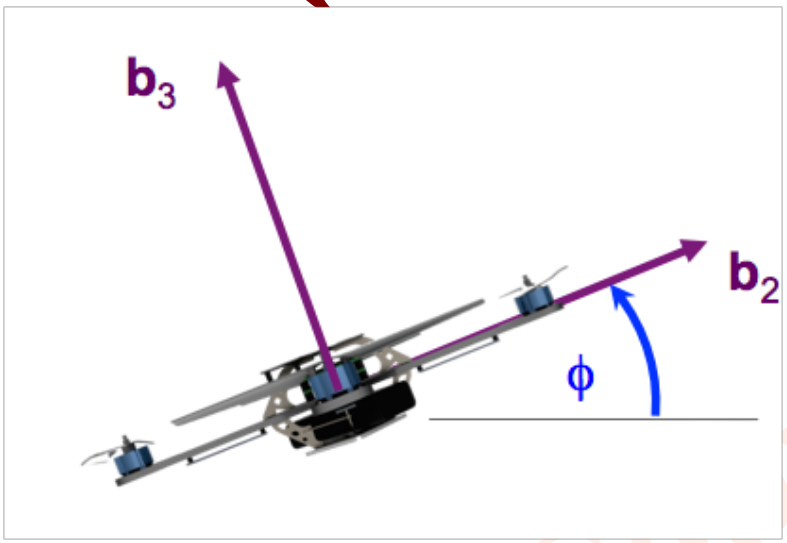
Three Approaches

1. Linear Controllers

2. Model-based nonlinear controllers

3. Exact linearization

Quadrotor (z-direction only)



$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\ddot{z} = -g + \frac{u_1}{m}\cos\phi$$

Hover configuration ($\phi \sim 0$)

$$\ddot{z} = -g + \frac{u_1}{m}$$

Equilibrium (operating point)

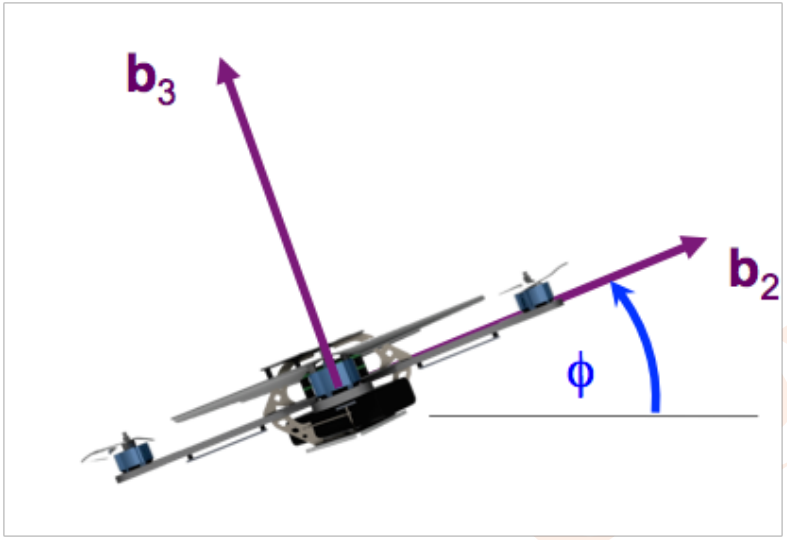
$$u_1 = mg$$

Want errors to decay to zero

$$(\ddot{z}^{des} - \ddot{z}) + K_D(\dot{z}^{des} - \dot{z}) + K_P(z^{des} - z) = 0$$

$$u_1 = m\left(g + \ddot{z}^{des} + K_D(\dot{z}^{des} - \dot{z}) + K_P(z^{des} - z)\right)$$

Quadrotor (y-direction only)



$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Hover configuration ($\phi \sim 0$)

$$\ddot{y} = -\frac{1}{m}\phi u_1$$

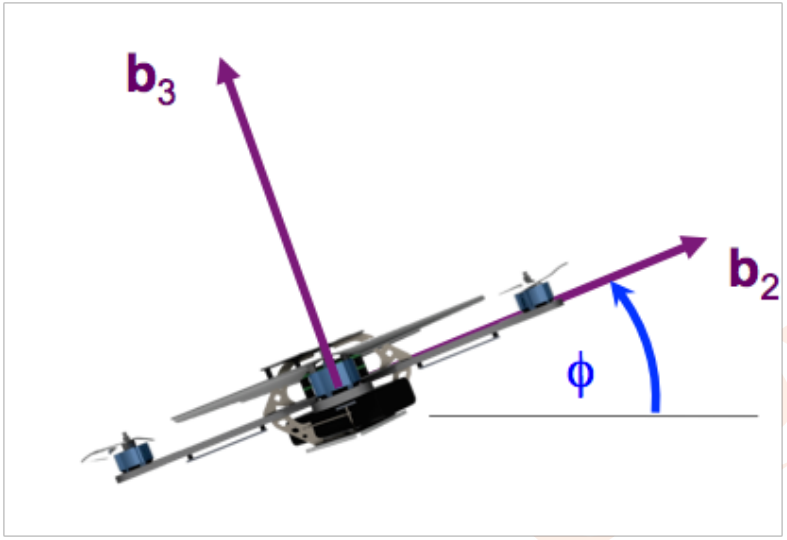
$$\ddot{y} = -\phi m g$$

Want errors to decay to zero

$$(\ddot{y}^{des} - \ddot{y}) + K_D(\dot{y}^{des} - \dot{y}) + K_P(y^{des} - y) = 0$$

$$\phi^{des} = -\frac{1}{g}\left(\ddot{y}^{des} + K_D(\dot{y}^{des} - \dot{y}) + K_P(y^{des} - y)\right)$$

Quadrotor (ϕ-direction only)

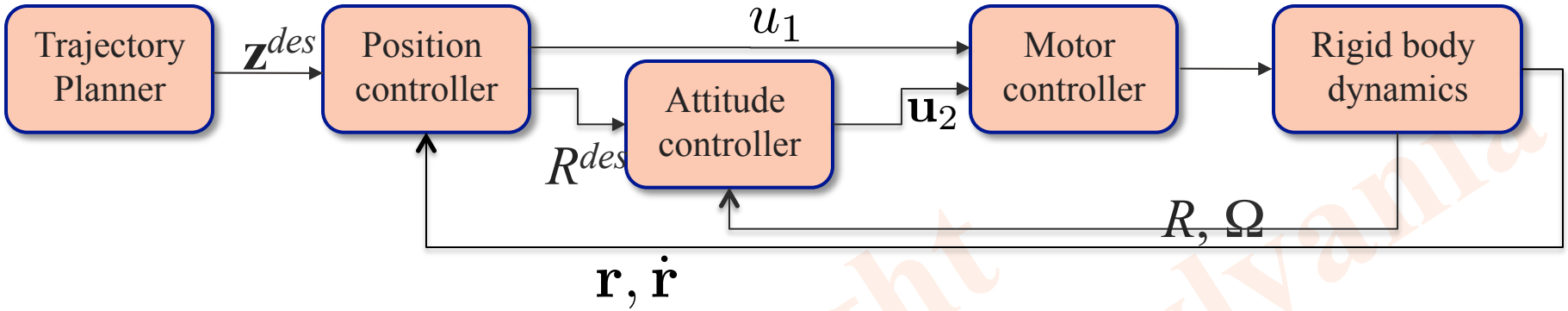


$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\ddot{\phi} = \frac{1}{I_{zz}} u_2$$

$$\phi^{des} = -\frac{1}{g}\left(\ddot{y}^{des} + K_D(\dot{y}^{des} - \dot{y}) + K_P(y^{des} - y)\right)$$

$$u_2 = I_{zz}\left(\ddot{\phi}^{des} + K_D(\dot{\phi}^{des} - \dot{\phi}) + K_P(\phi^{des} - \phi)\right)$$



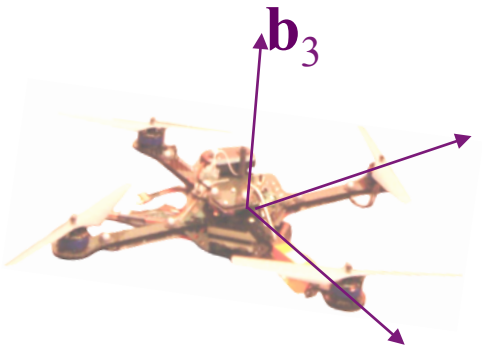
$$m\ddot{\mathbf{r}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix}$$

u_1

$$I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$\mathbf{u}_2$

Control for Hovering



$$m\ddot{\mathbf{r}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix}$$

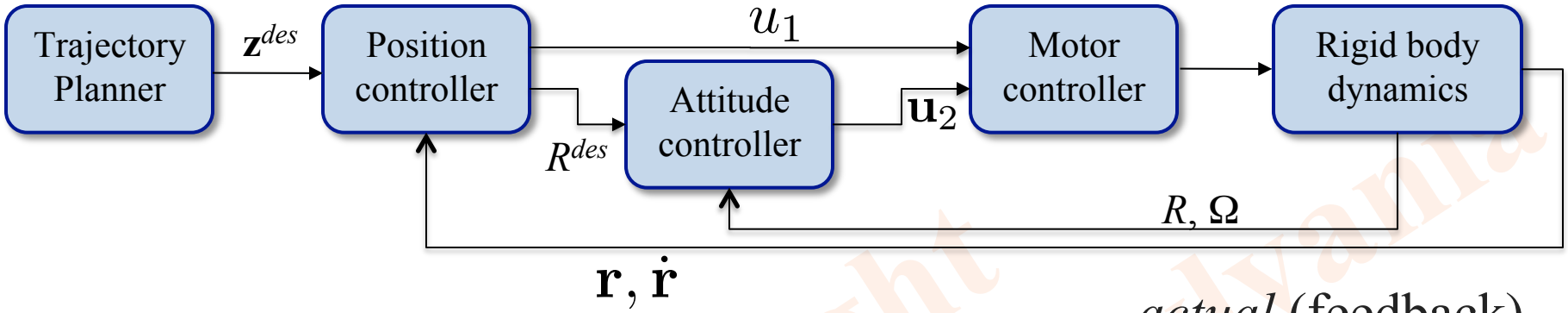
u_1

Linearization

$$(u_1 \sim mg, \theta \sim 0, \phi \sim 0, \psi \sim \psi_0)$$

$$\ddot{r}_1 = g(\Delta\theta \cos\psi_0 + \Delta\phi \sin\psi_0)$$

$$\ddot{r}_2 = g(\Delta\theta \sin\psi_0 - \Delta\phi \cos\psi_0)$$



actual (feedback)

$$(\ddot{r}_{i,T} - \ddot{r}_i^{\text{des}}) + k_{d,i}(\dot{r}_{i,T} - \dot{r}_i) + k_{p,i}(r_{i,T} - r_i) = 0$$

specified

$$u_1 = mg + m\ddot{r}_3^{\text{des}}$$

$$\phi^{\text{des}} = \frac{1}{g}(\ddot{r}_1^{\text{des}} \sin\psi_T - \ddot{r}_2^{\text{des}} \cos\psi_T)$$

$$\theta^{\text{des}} = \frac{1}{g}(\ddot{r}_1^{\text{des}} \cos\psi_T + \ddot{r}_2^{\text{des}} \sin\psi_T)$$

$$\mathbf{u}_2 = \begin{bmatrix} k_{p,\phi}(\phi^{\text{des}} - \phi) + k_{d,\phi}(p^{\text{des}} - p) \\ k_{p,\theta}(\theta^{\text{des}} - \theta) + k_{d,\theta}(q^{\text{des}} - q) \\ k_{p,\psi}(\psi^{\text{des}} - \psi) + k_{d,\psi}(r^{\text{des}} - r) \end{bmatrix}$$

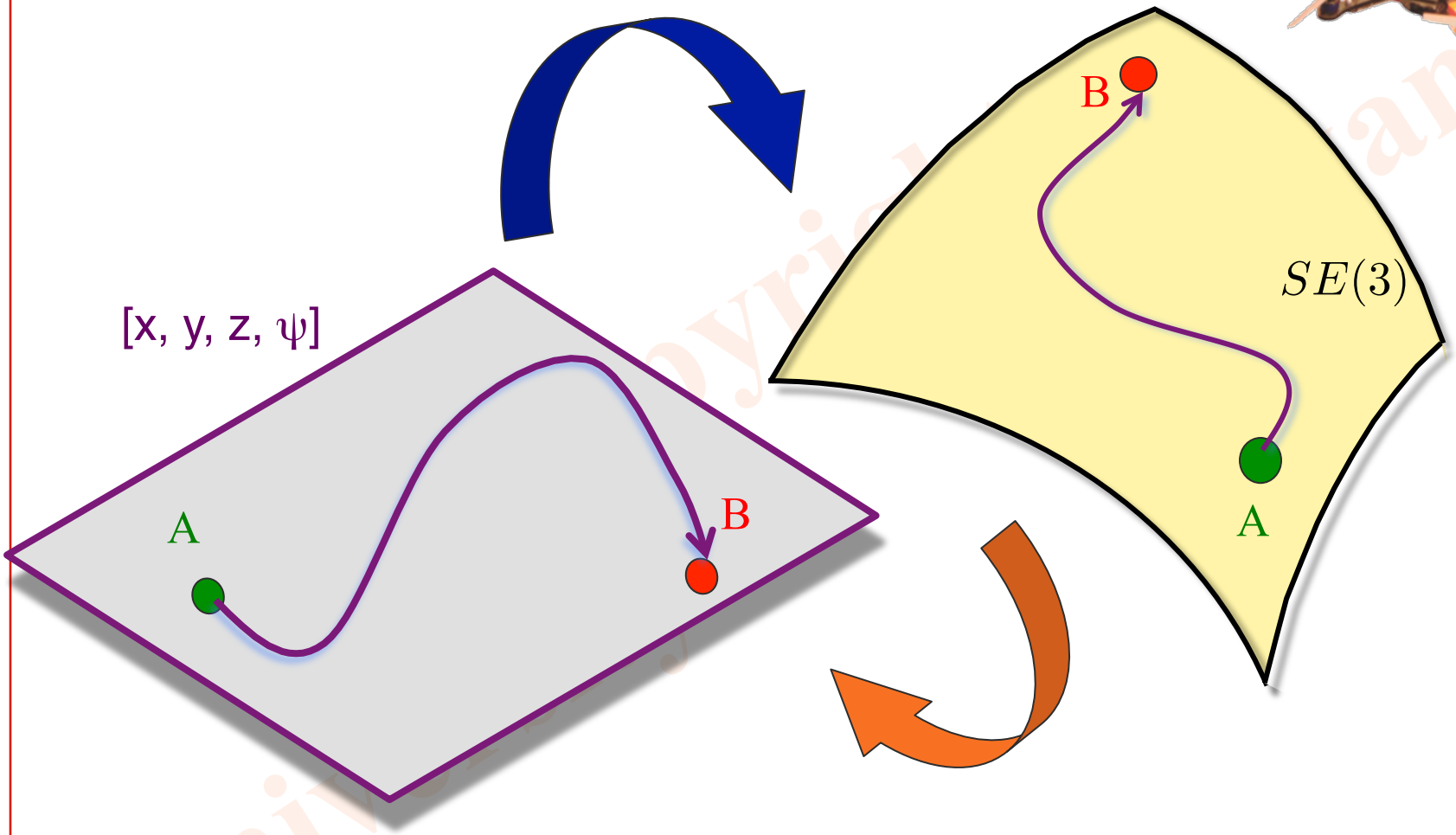
Nano+ Quadrotor (KMel Robotics)

- 25 cm diameter
- Carbon fiber frame
- 176 gm (230 gm with camera and processor)

1. Dynamic Model
2. Matlab Simulation
   (understand the simulator)
3. Controllers for the quadrotor
4. Fly the robot along planned trajectories

Nonlinear Control

Differentially Flat System



$$\sigma(t) : [0, T] \rightarrow R^3 \times SO(2)$$

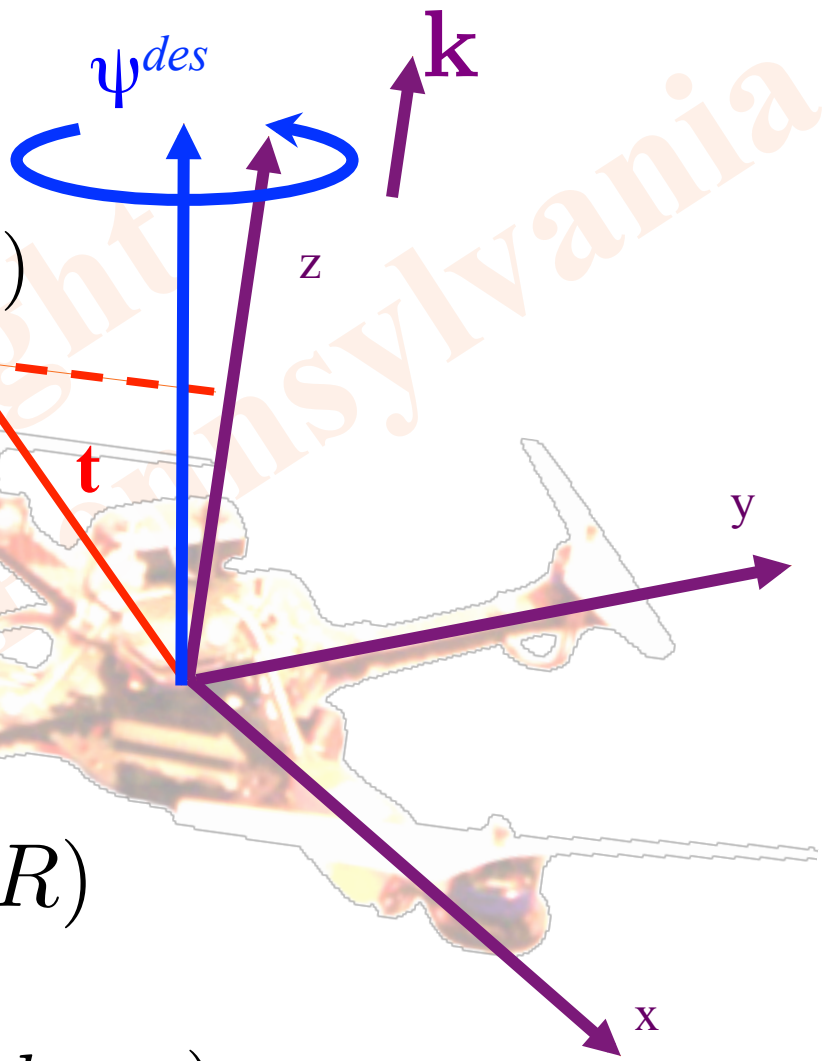
$$\begin{bmatrix} x \\ y \\ z \\ \theta \\ \phi \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix}$$

[Mellinger and Kumar, 2011]

Control on SE(3)

Specified trajectory

$$\sigma(t) : [0, T] \rightarrow R^3 \times SO(2)$$



$$f = m\left(\ddot{\mathbf{r}} + K_v \dot{\mathbf{e}}_r + K_p \mathbf{e}_r + \mathbf{g}\right) \cdot \mathbf{k}$$

$$\mathbf{R}^{des} e_3 = \frac{\mathbf{t}}{\|\mathbf{t}\|}$$
$$\psi = \psi^{des}$$

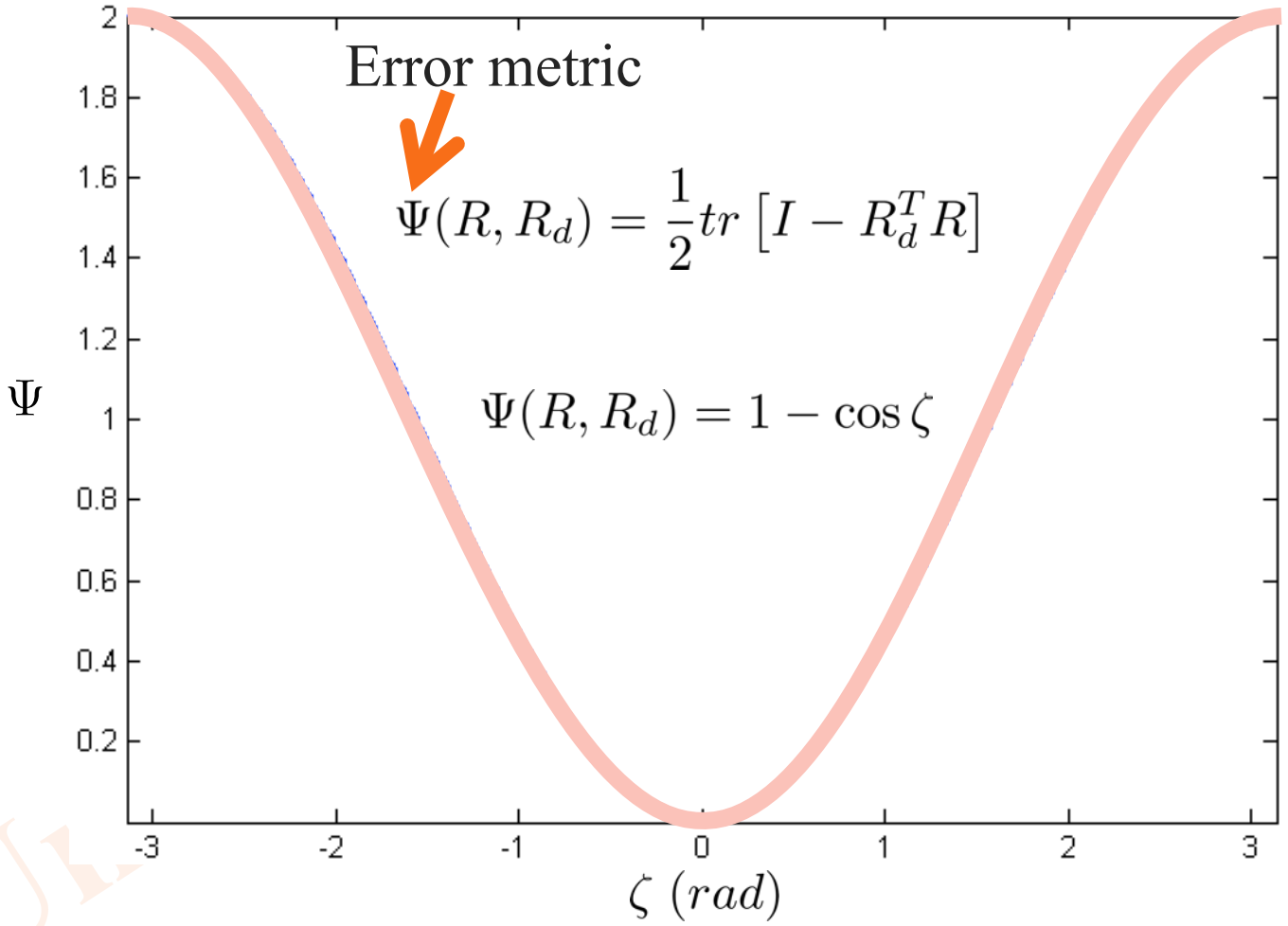
R^{des}

$e_R(R^{des}, R)$

$$M = \omega \times \mathbf{I}\omega + \mathbf{I}(-k_R e_R - k_\omega e_\omega)$$

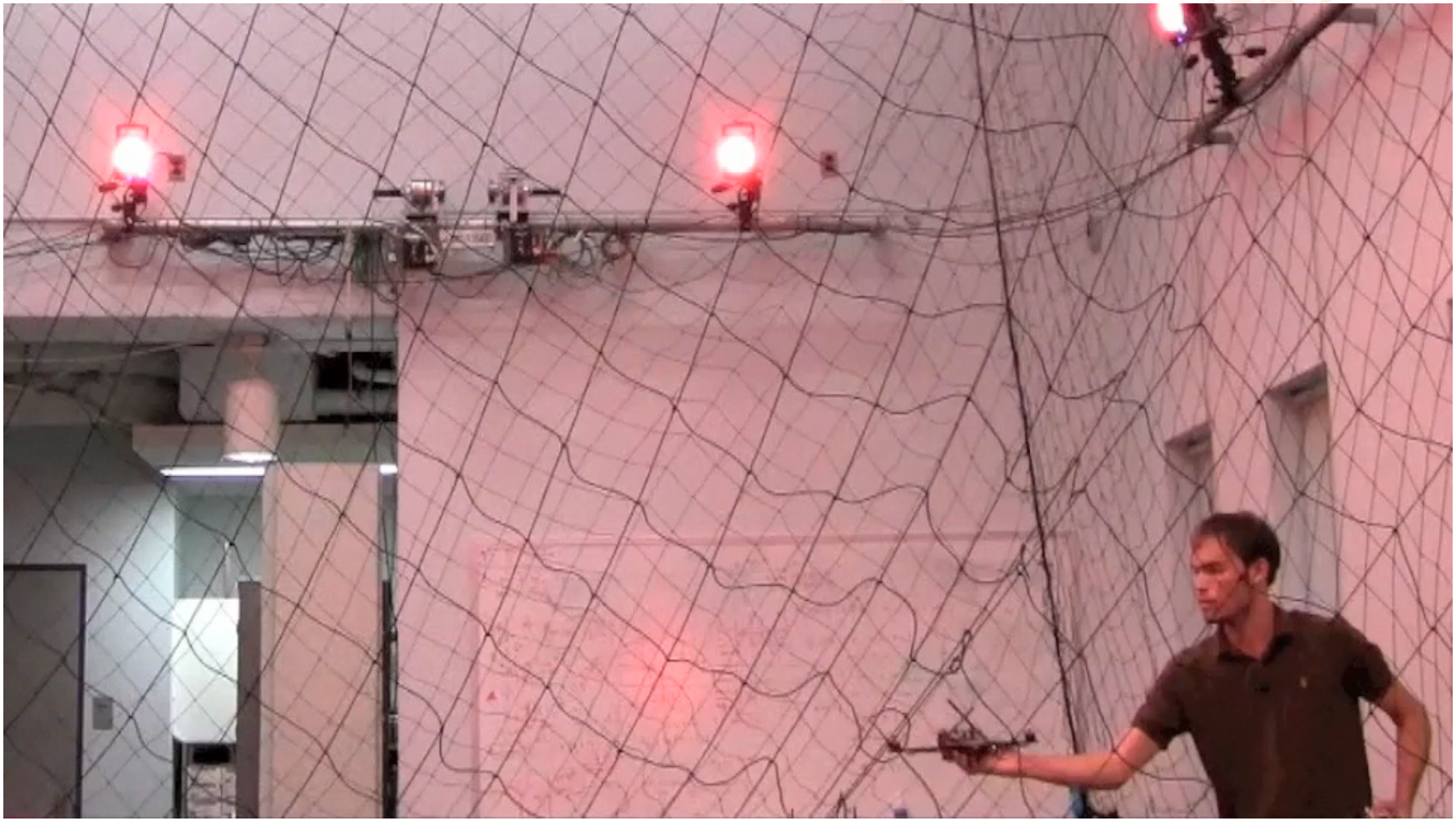
[Wen and Kreutz-Delgado, 1991; Lee *et al*, 2011; Mellinger and Kumar, 2011]

An SO(3) Controller (Attitude Control)

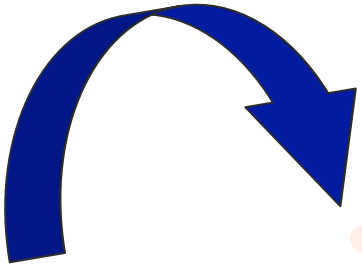


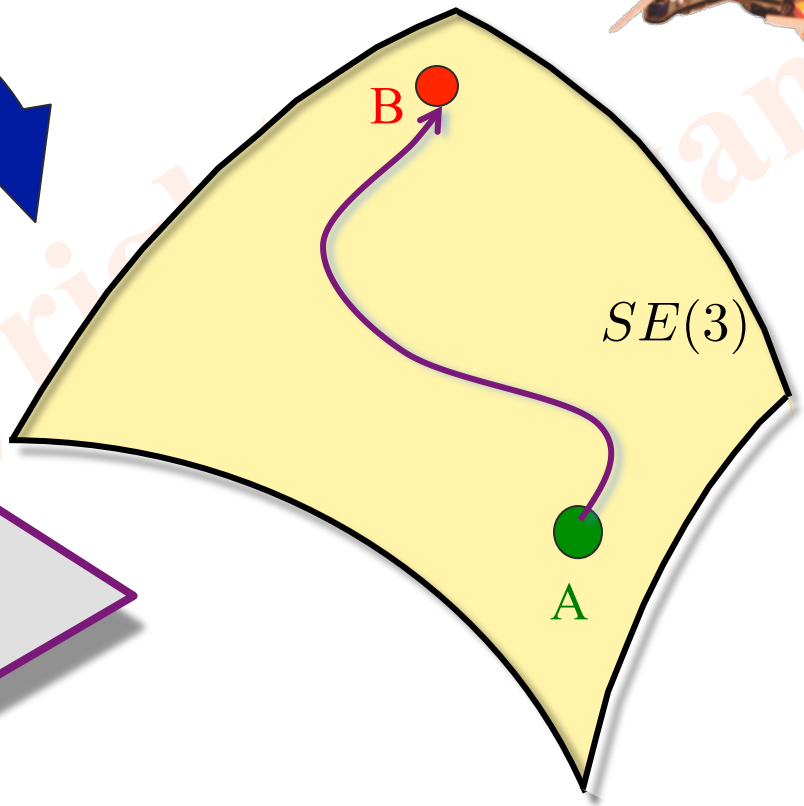
Basin of attraction

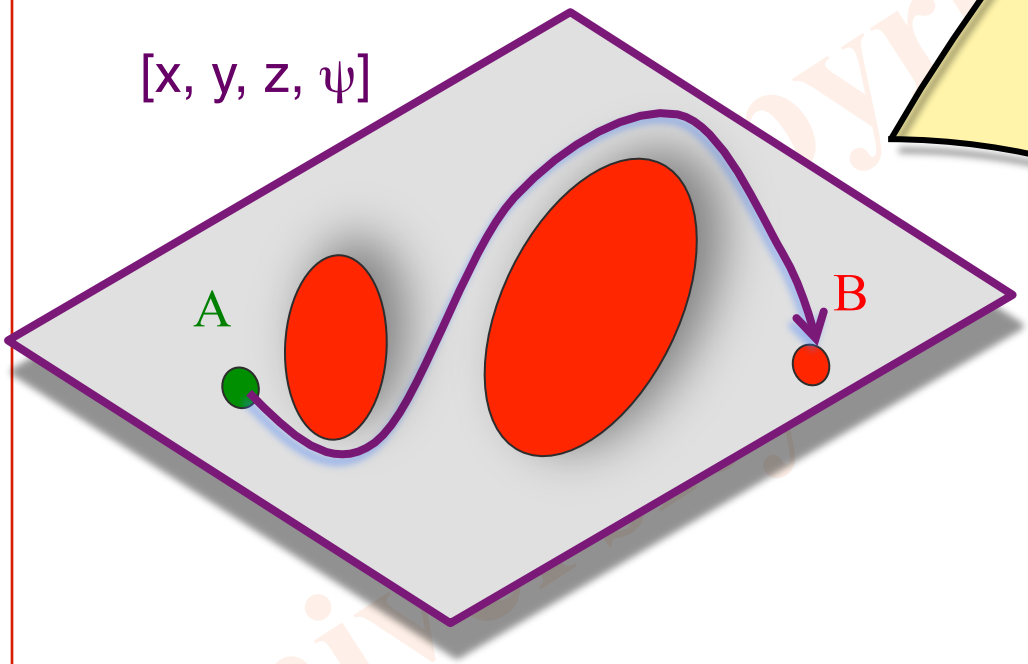
$$tr[I - (R^{des})^T R] < 2 \qquad \|e_\omega(0)\|^2 \leq \frac{2}{\lambda_{min}(I)} k_R \left(1 - \frac{1}{2} tr\left[I - (R^{des})^T R\right]\right)$$



Planning







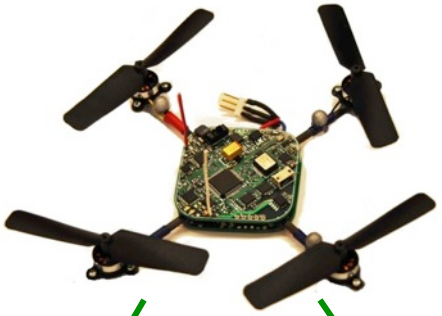
$$\begin{bmatrix} x \\ y \\ z \\ \theta \\ \phi \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix}$$

Minimum snap trajectory

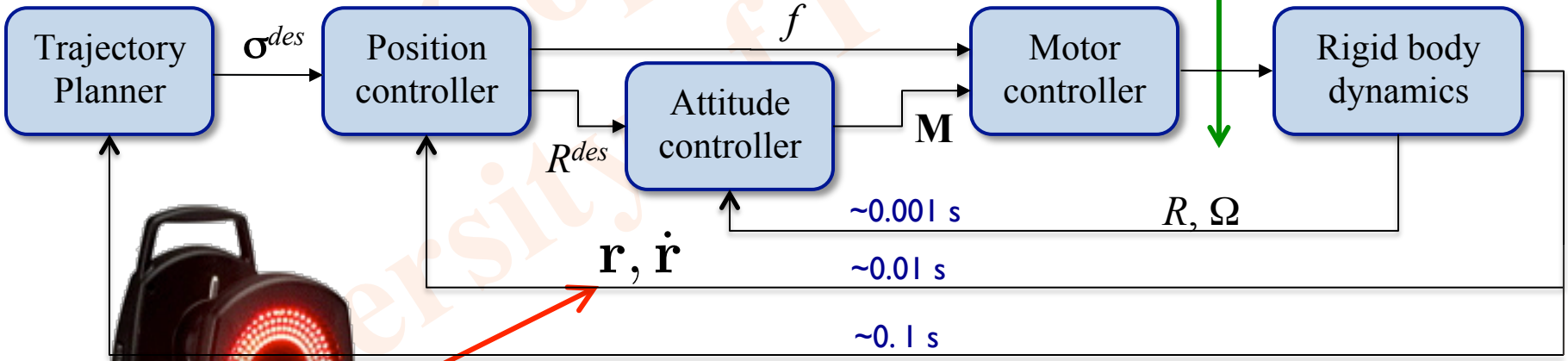
$$\min_{\sigma(t)} \int_0^T \alpha \| \dddot{\mathbf{r}}(t) \|^2 + \beta \ddot{\psi}(t)^2 dt$$

[Mellinger and Kumar, 2011]

Software Architecture



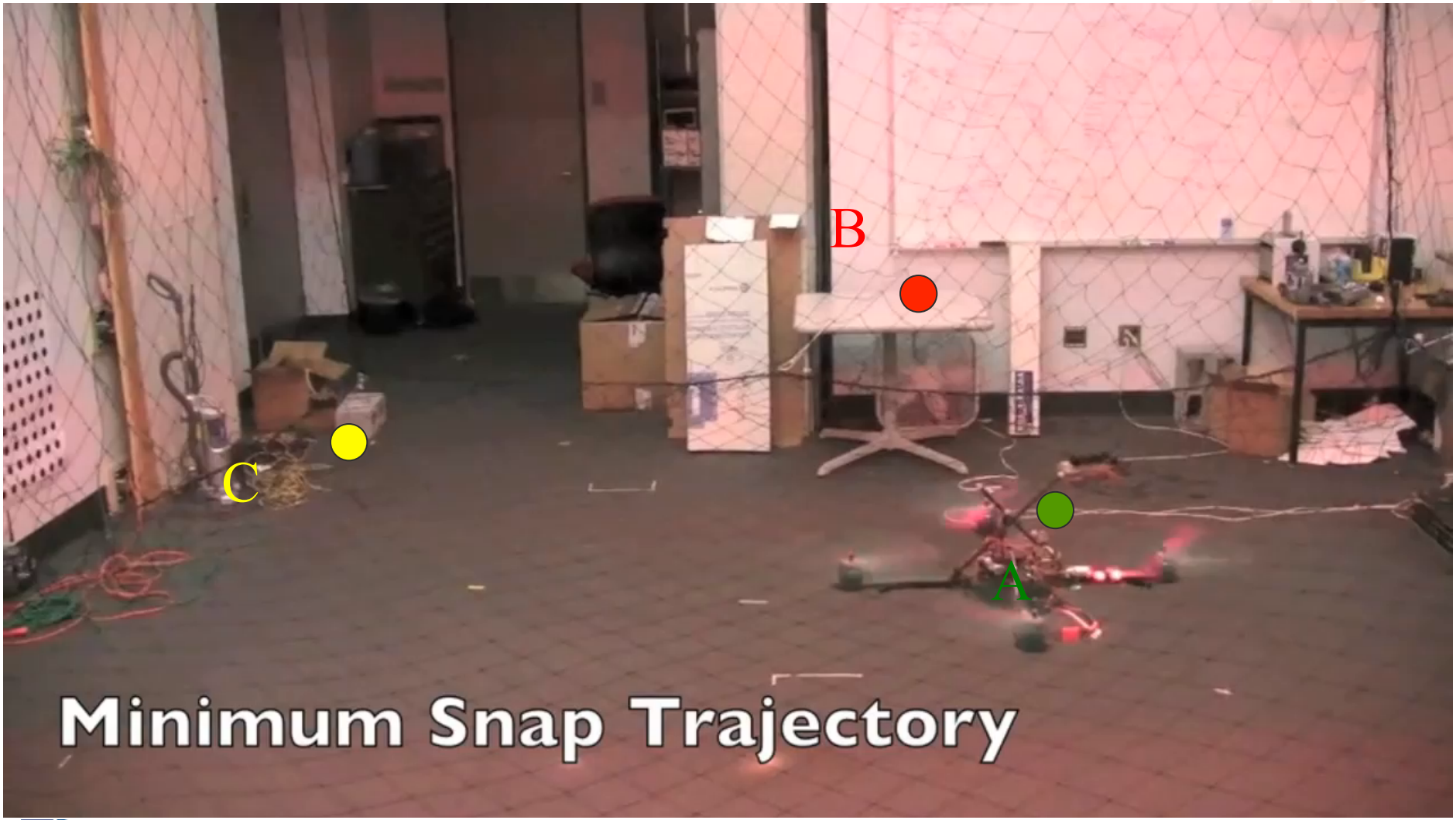
Nonlinear controller for orientation



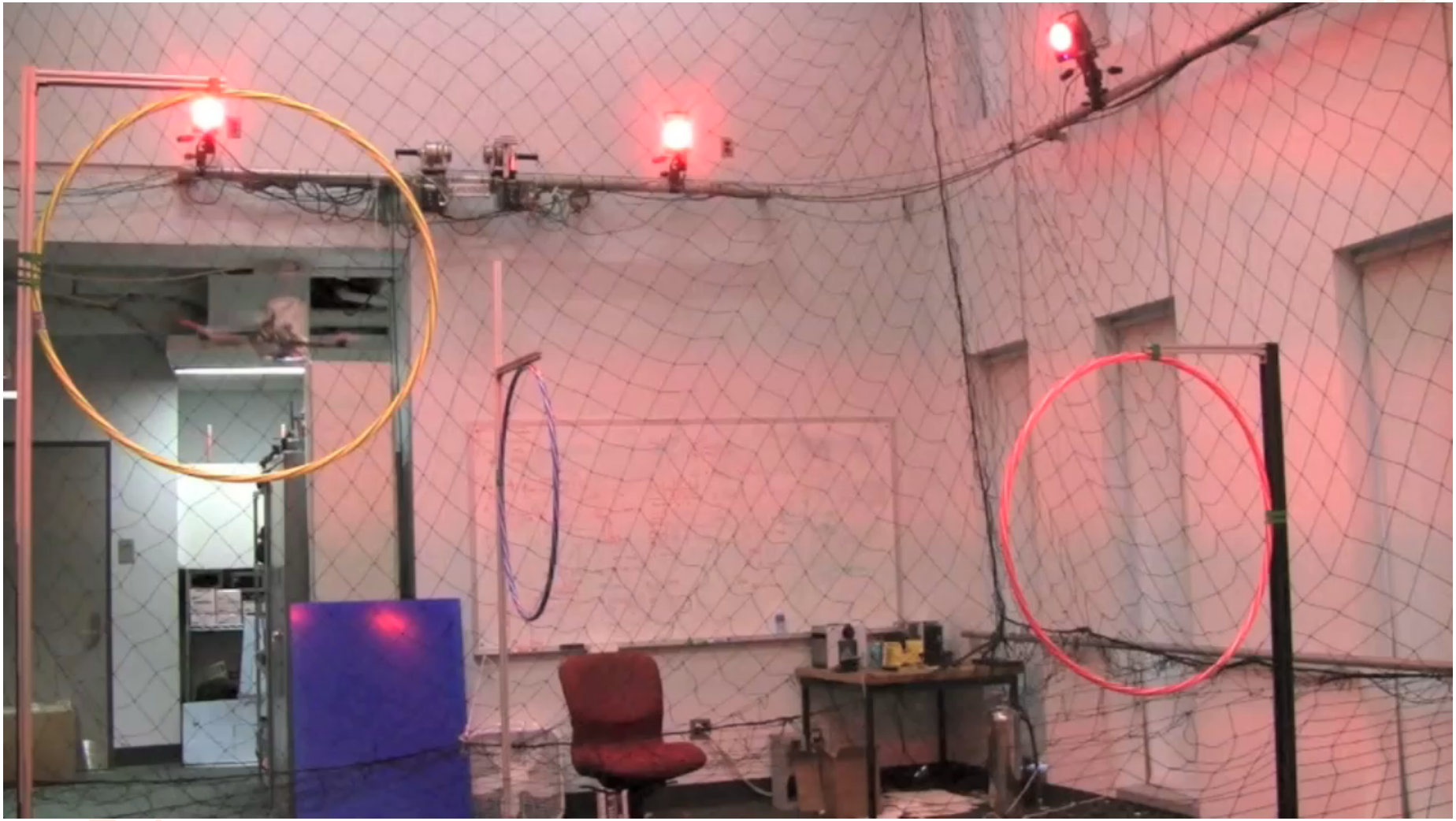
[Kraft 2003; Mellinger, Michael, and Kumar 2010; Mellinger and Kumar 2011]

Minimum Snap Trajectories

[Mellinger and Kumar, ICRA 2011]



Minimum Snap Trajectories



[Mellinger and Kumar, ICRA 2011]

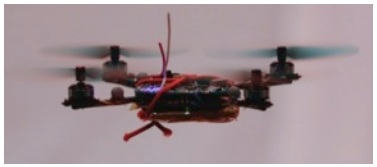
Break

Outline – Section II

- Demonstration
- Design Tradeoffs (Yash Mulgaonkar)
  - Kinematics, power, component analysis, designing your own quadrotor
- Demonstration
- Sensors – Part I (Justin Thomas)
- Quads in the Lab (Mickey Whitzer & Yash Mulgaonkar)
  - The UAV Testbed, MATLAB Simulator, gain tuning
- Break

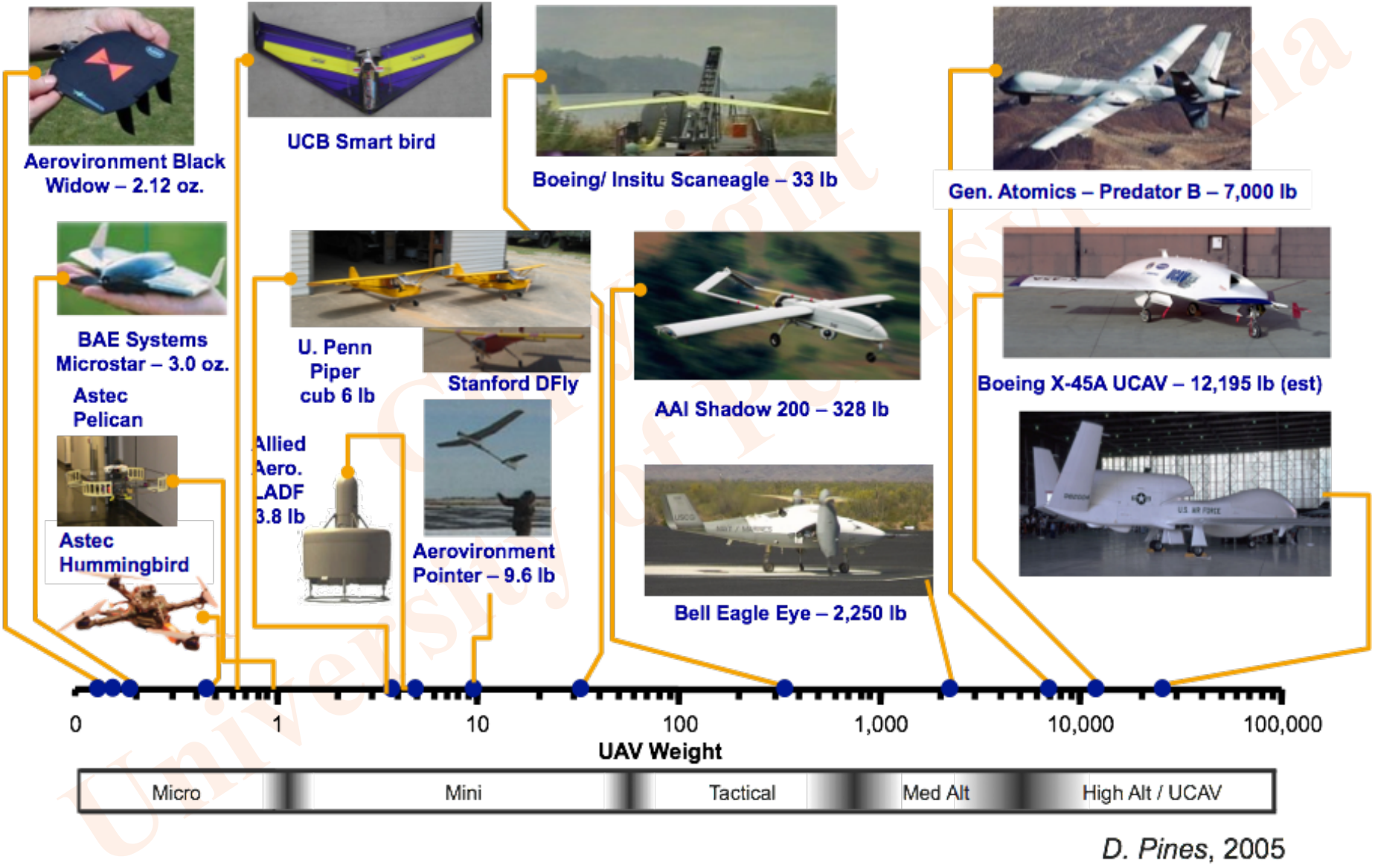
Outline

- Design Tradeoffs
  - Kinematics
  - Power
  - Component analysis
- Designing your own quadrotor
  - Open Source Projects
- Demo



*http://www.e-volo.com/information/vc1-proof-of-concept

Unmanned Aerial Vehicles

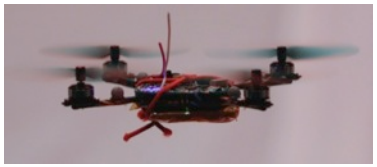


D. Pines, 2005

Outline

- Design Tradeoffs
  - Kinematics
  - Power
  - Component analysis
- Designing your own quadrotor
  - Open Source Projects
- Demo


*



*http://www.e-volo.com/information/vc1-proof-of-concept

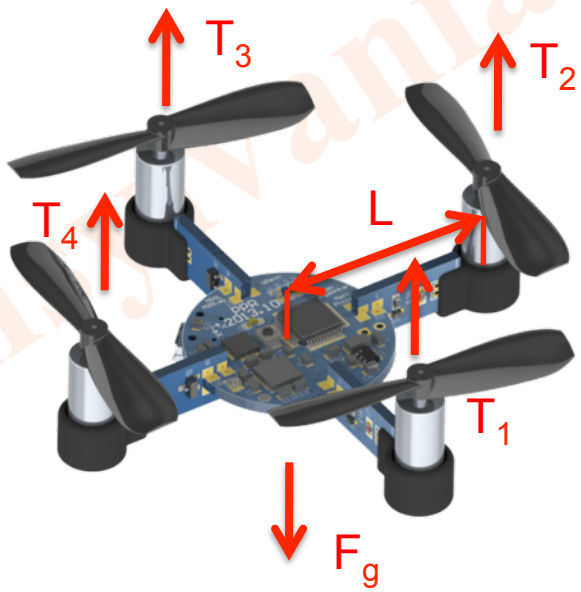
Scaling Characteristics (Kinematic)

- Mass: $m \propto V \propto L^3$
- Inertia:

$$I = \int_{body} r^2 dm = \underbrace{\int_{body} \underbrace{r^2}_{\propto L^2} \rho \, dV}_{\propto L^3}$$

$$\rho \rightarrow \text{density}$$

$$I \propto L^5$$



- Thrust and Drag:

$$T = C_T \rho A r^2 \omega_i^2 = C_T \rho \omega_i^2 \underbrace{A}_{\propto L^2} \underbrace{r^2}_{\propto L^2}$$

$$T \propto \omega_i^2 L^4 \quad , \quad D \propto \omega_i^2 L^4$$

$A \rightarrow Rotor\ Area$

$r \rightarrow Rotor\ Radius$

$\omega_i \rightarrow Angular\ Velocity$

$\rho \rightarrow Air\ Density$

- Linear and Angular Acceleration:

$$a = \frac{F}{m} \propto \frac{\omega_i^2 L^4}{L^3} \propto \omega_i^2 L, \qquad \alpha = \frac{\tau}{I} = \frac{TL}{I} \propto \frac{\omega_i^2 L^5}{L^5} \propto \omega_i^2$$

Scaling Characteristics (Kinematic)

Mach Scaling

- Compressible Flows
  - Assumes tip velocities are constant

$$\omega_i \propto \frac{1}{r_i}, \qquad r_i \rightarrow \text{Rotor Radius}$$

- Revisiting Linear and Angular Accelerations

$$a = \frac{F}{m} \propto \frac{\omega_i^2 L^4}{L^3} \propto \omega_i^2 L \propto \frac{L}{r^2} \propto \frac{1}{L} \quad (r \sim L)$$

$$\alpha = \frac{\tau}{I} = \frac{TL}{I} \propto \frac{\omega_i^2 L^5}{L^5} \propto \omega_i^2 \propto \frac{1}{L^2}$$

Froude Scaling

- Incompressible Flows
  - Assumes (for same vehicle) Froude number is constant

$$F_r = \frac{v_{tip}^2}{Lg} = \frac{\omega_i^2 r^2}{Lg} \rightarrow constant$$

$$\Rightarrow \quad \omega_i \propto \frac{1}{\sqrt{r}} \quad (\text{since} \quad L \sim r)$$

- Revisiting Linear and Angular Accelerations

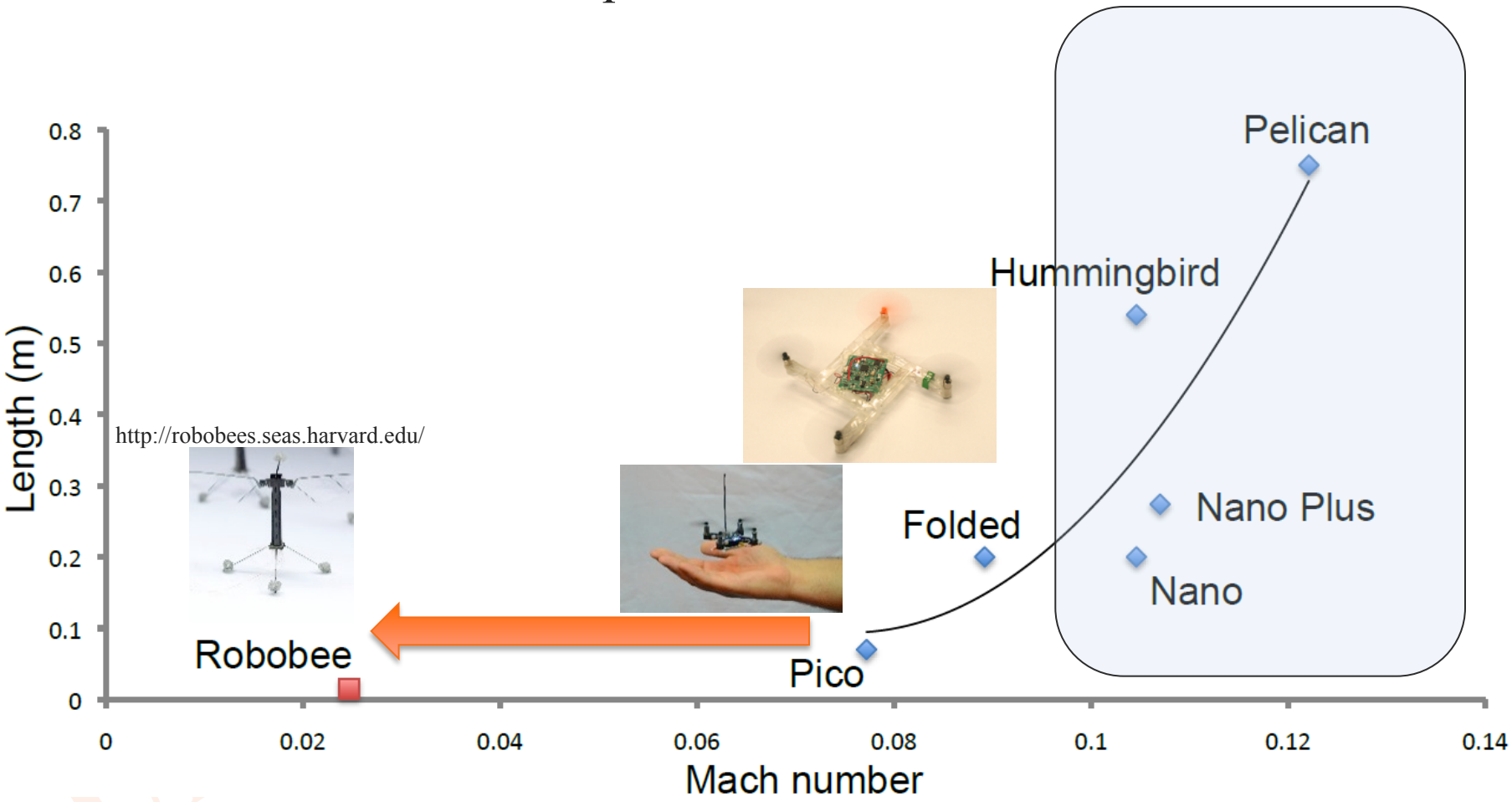
$$a = \frac{F}{m} \propto \frac{\omega_i^2 L^4}{L^3} \propto \omega_i^2 L \propto \frac{L}{r} \sim 1 \quad (r \sim L)$$

$$\alpha = \frac{\tau}{I} = \frac{TL}{I} \propto \frac{\omega_i^2 L^5}{L^5} \propto \omega_i^2 \propto \frac{1}{L}$$

Y. Mulgaonkar, et al. " Power and weight considerations in small, agile quadrotors ", *Proc. SPIE* 9083, Micro- and Nanotechnology Sensors, Systems, and Applications VI, 90831Q (June 4, 2014); doi:10.1117/12.2051112

Scaling Characteristics (Kinematic)

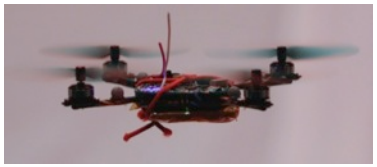
- Mach Number for small quadrotors: $\sim \sqrt{L}$



Trying to bridge this gap

Outline

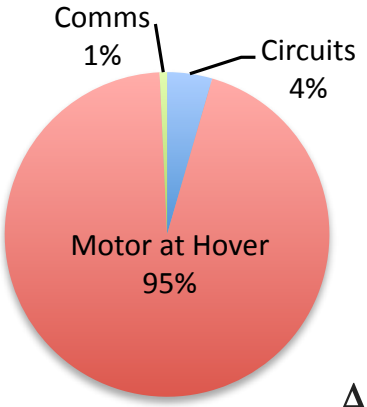
- Design Tradeoffs
  - Kinematics
  - Power
  - Component analysis
- Designing your own quadrotor
  - Open Source Projects
- Demo



*http://www.e-volo.com/information/vc1-proof-of-concept

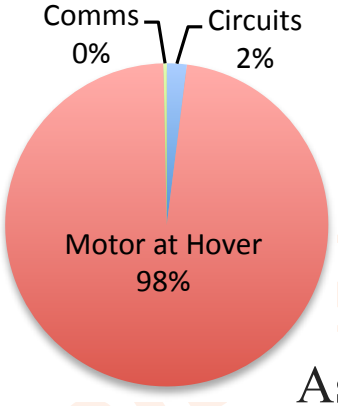
Scaling Characteristics (Power)

Power Draw at Hover



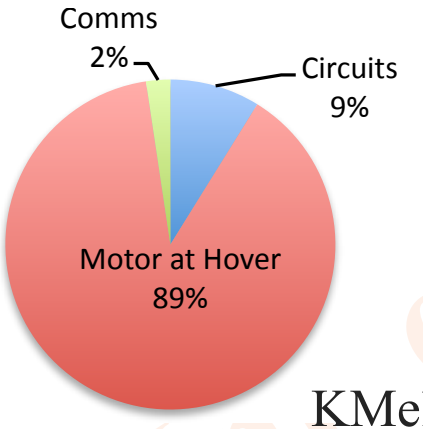
AscTec Pelican

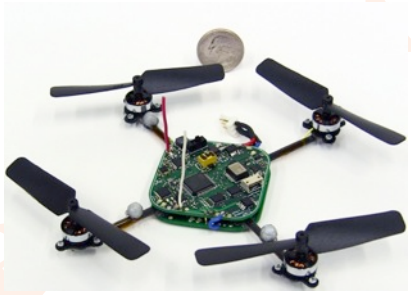
650W @ Hover



AscTec Hummingbird

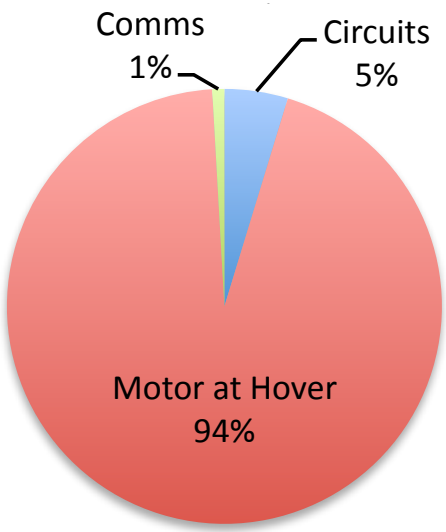
75W @ Hover





KMel Nano Quadrotor

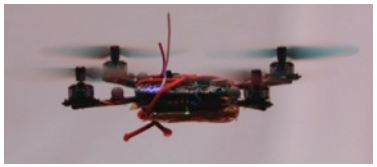
18W @ Hover



Average Power Consumption

Outline

- Design Tradeoffs
  - Kinematics
  - Power
  - Component analysis
- Designing your own quadrotor
  - Open Source Projects
- Demo



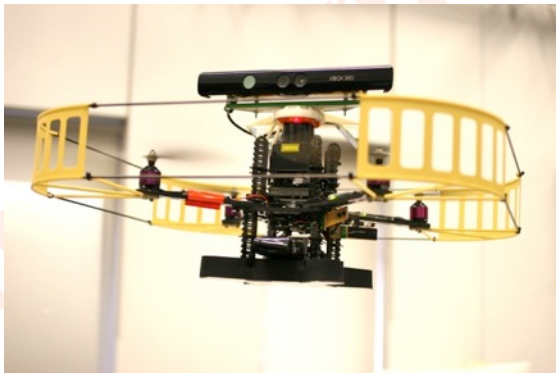
*http://www.e-volo.com/information/vc1-proof-of-concept

Scaling Characteristics

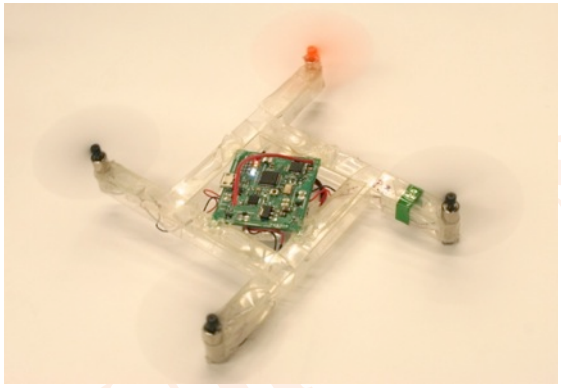
Platforms Tested

AscTec Hummingbird
(m = 486g)

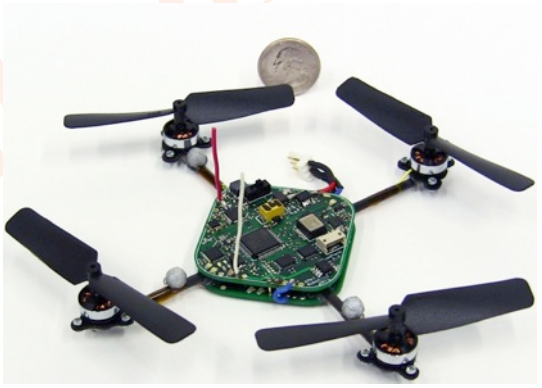
AscTec Pelican (unloaded)
(m = 964g)



AscTec Pelican (loaded)
(m = 1937g)



Folded Quadrotor
(m = 43.4g)

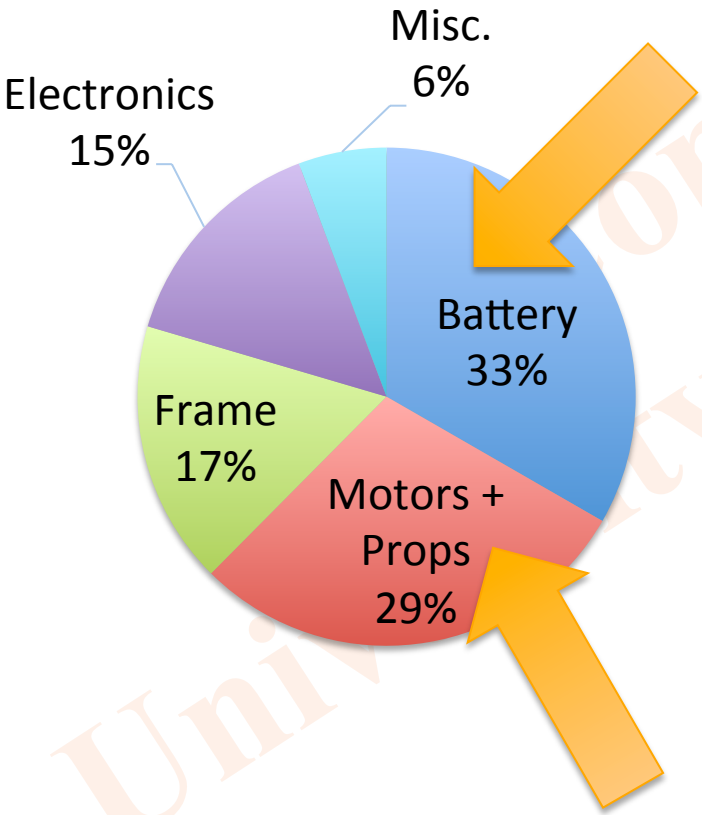


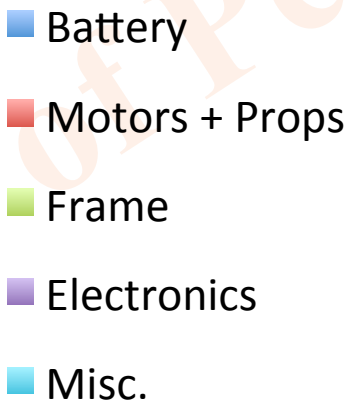
KMel Nano
(m = 83g)

KMel kQuad 500
(m = 594g)

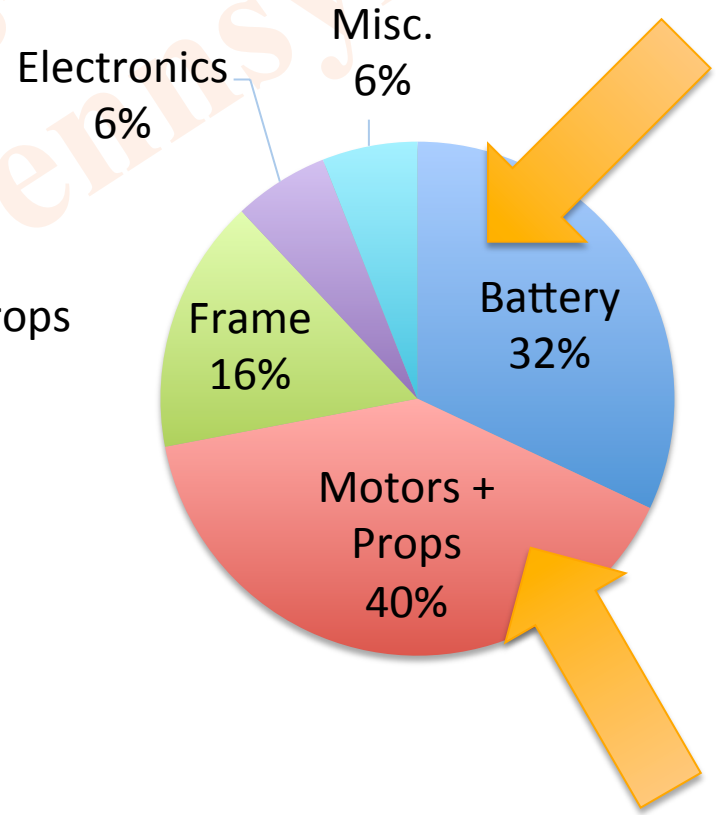
Scaling Characteristics (Weight)

Average Mass Distribution for all test platforms





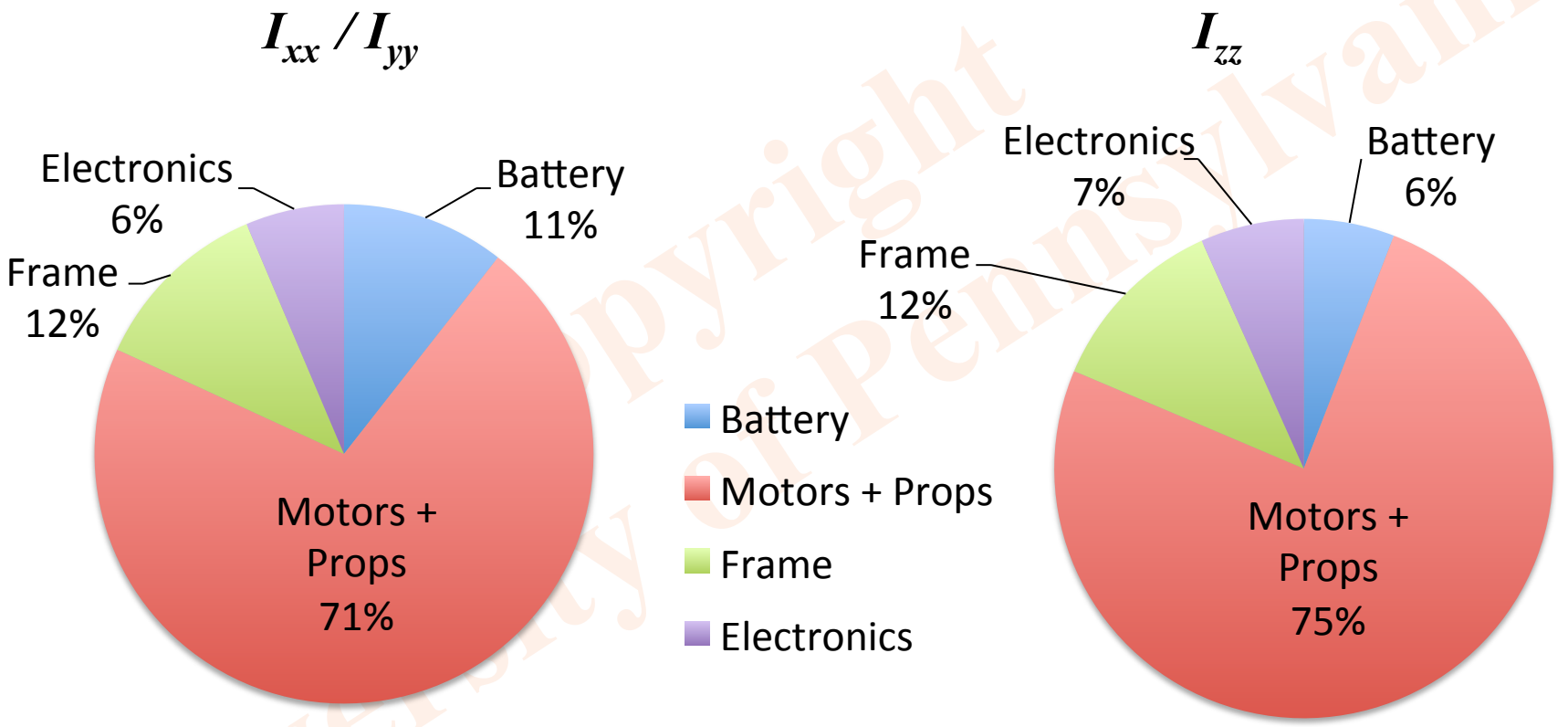
Average Mass Distribution for the Pico-Quadrotor



Y. Mulgaonkar, et al. " Power and weight considerations in small, agile quadrotors ", *Proc. SPIE* 9083, Micro- and Nanotechnology Sensors, Systems, and Applications VI, 90831Q (June 4, 2014); doi:10.1117/12.2051112

Scaling Characteristics (Weight)

Average quadrotor inertia distribution



AscTec Hummingbird

$$I_{xx} \approx I_{yy} \approx 2.6 \times 10^{-3} kgm^2$$
$$I_{zz} \approx 5.0 \times 10^{-3} kgm^2$$

Pico-Quadrotor

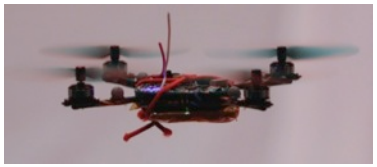
$$I_{xx} \approx I_{yy} \approx 9.19 \times 10^{-6} kgm^2$$
$$I_{zz} \approx 16.01 \times 10^{-6} kgm^2$$

Y. Mulgaonkar, et al. " Power and weight considerations in small, agile quadrotors ", *Proc. SPIE* 9083, Micro- and Nanotechnology Sensors, Systems, and Applications VI, 90831Q (June 4, 2014); doi:10.1117/12.2051112

Outline

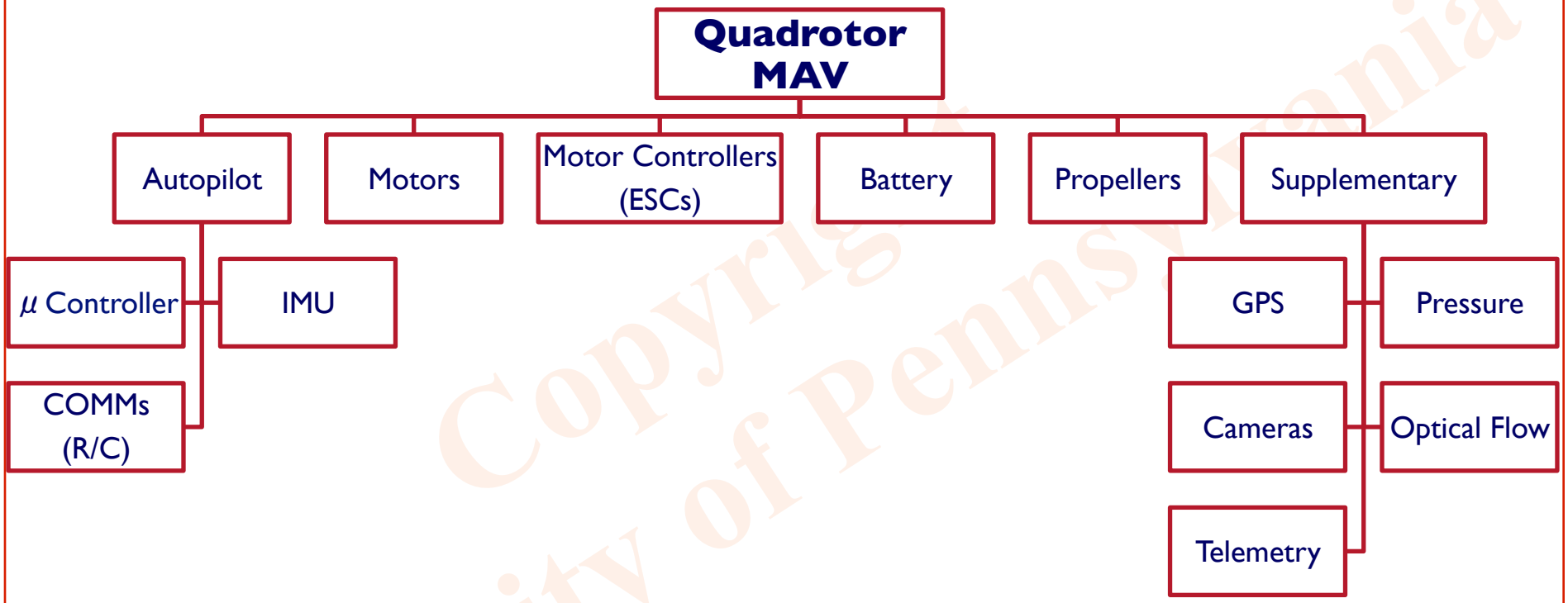
- Design Tradeoffs
  - Kinematics
  - Power
  - Component analysis
- Designing your own quadrotor
  - Open Source Projects
- Demo


*



*http://www.e-volo.com/information/vc1-proof-of-concept

Build Your Own Quadrotor!

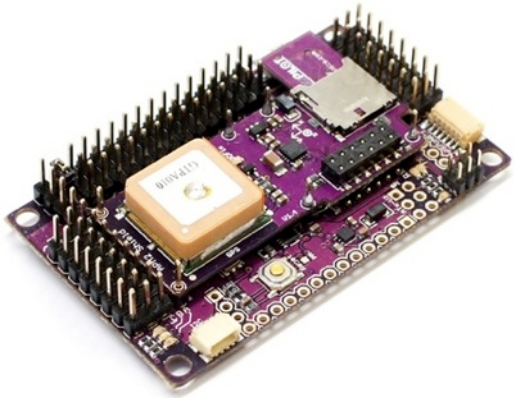


Autopilot

- PX4 Pixhawk:
  - MCU, IMU, PWM, PMU
  - External Transceiver
  - 81mm x 50mm x 15mm
  - Open Source
  - 38g

PX4 – Pixhawk
https://pixhawk.org/modules/pixhawk

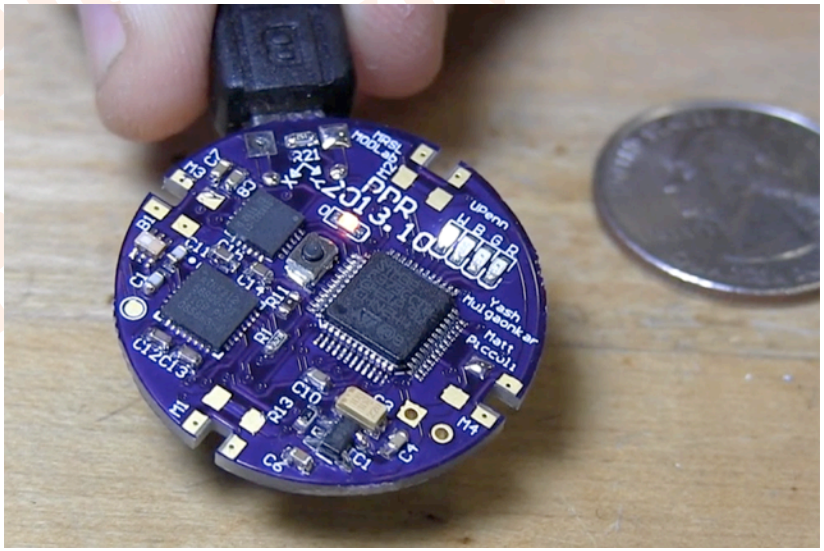
- 3DR APM 2.6:
  - MCU, IMU, PWM, PMU
  - External Transceiver
  - 66mm x 40mm x 11mm
  - Open Source
  - 33g



3DR – APM2.6
https://store.3drobotics.com/products/apm-2-6-kit-1

Autopilot

- No external components needed
  - Except motors + motor mounts
- Modular Approach
  - One autopilot, any size vehicle
- Hardware/Software Database
  - Database of all individual sub-modules
  - Drivers made with macros for hardware interfacing pins
- Key Features
  - 72 Components
  - 30mm x 30mm
  - Smallest component is 1mm x 0.5mm
  - 5g ≈ US 25¢ coin



R / C

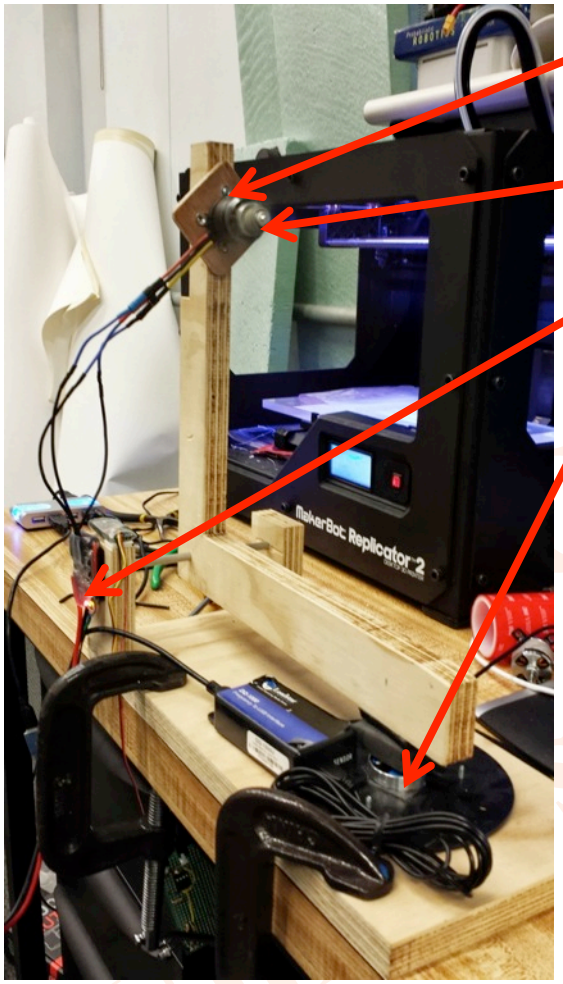
Futaba®

SPEKTRUM®
Leaders in Spread Spectrum Technology

Futaba 6EX
http://www.futaba-rc.com/systems/futk6900.html

Spektrum DX6i
https://www.spektrumrc.com

Motors + ESCs + Propellers



Propeller Testing Rig

T-Motor MT2208

T-Motor T12A ESC

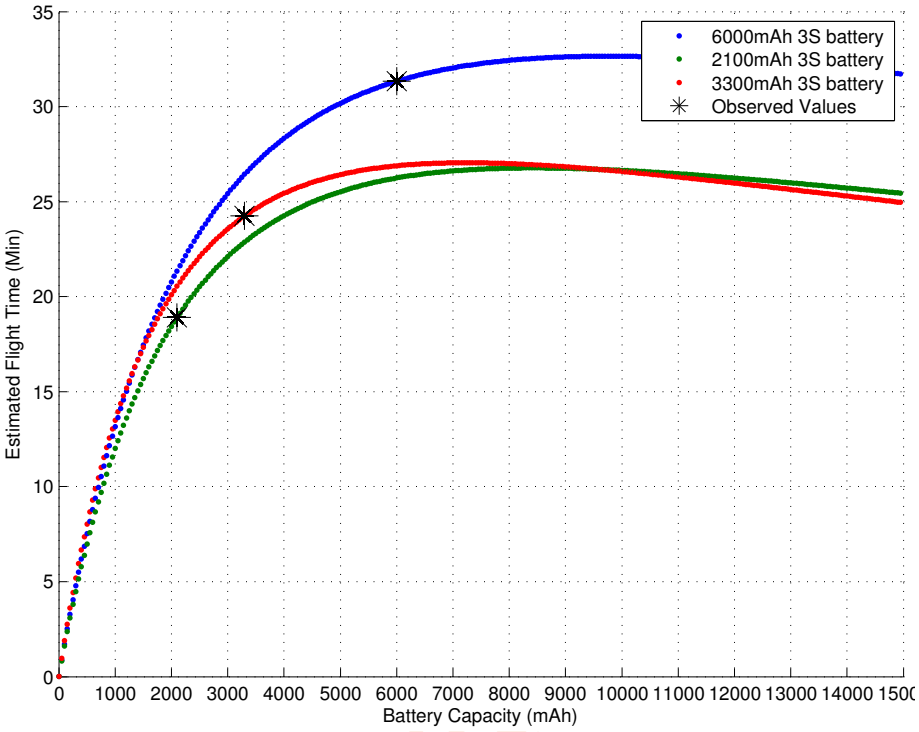
T-Motor 8x2.7 CF Propeller

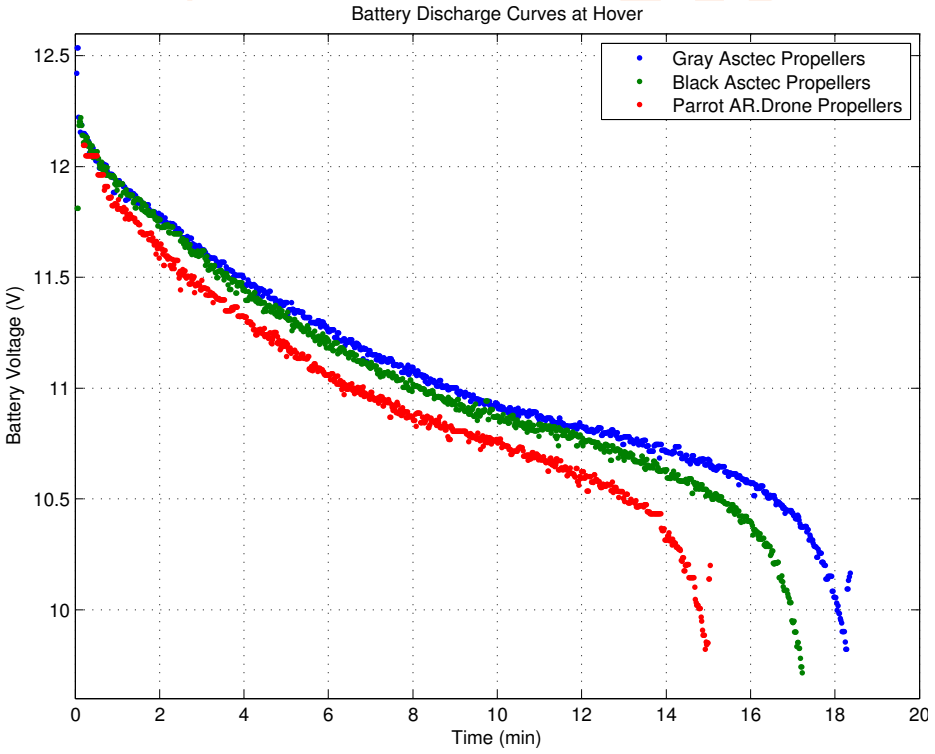
Batteries

Turnigy nano-tech 3300mah 3S 45~90C Lipo

ARL (Kroninger,Morgan)

Battery + Propeller Selection

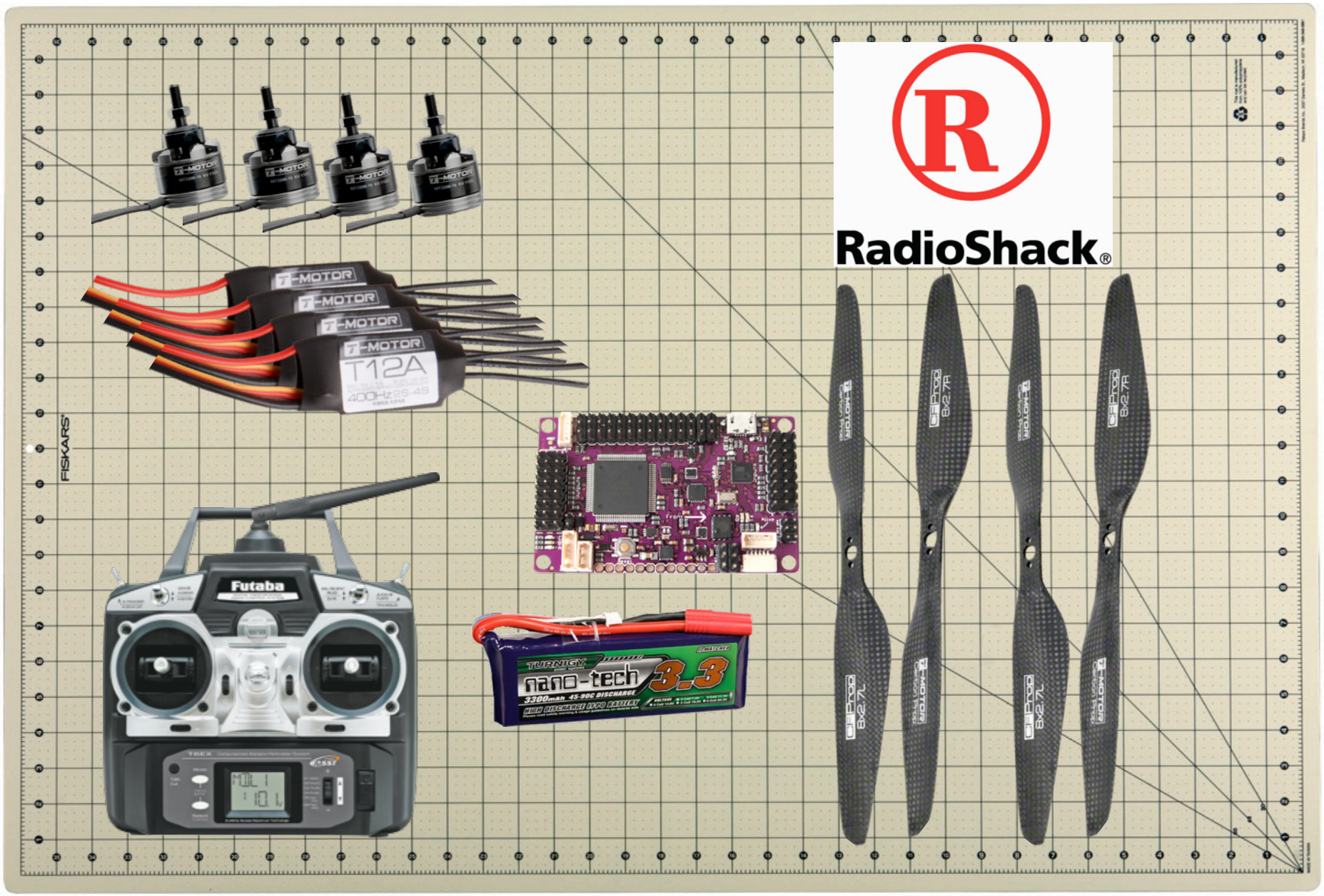




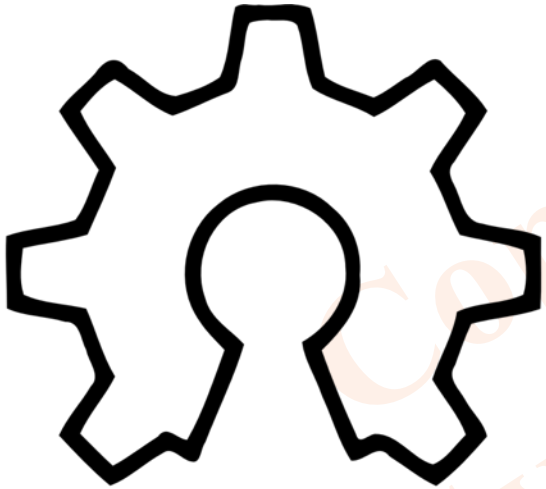
Yash Mulgaonkar, Vijay Kumar, ***"Autonomous charging to enable long-endurance missions for small aerial robots"***, Proceedings of SPIE Vol. 9083, 90831S (2014)

Sensors

S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, ***"Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV,"***
In Proc. of the IEEE Intl. Conf. on Robot. and Autom., Hong Kong, China, May 2014


RadioShack®
T-MOTOR
T12A
400Hz 2S-4S
Futaba
TURNIGY
nano-tech 3.3
3300mah 45-90C DISCHARGE
HIGH DISCHARGE LIPO BATTERY
8x2.7R
FISKARS

Open Source!!!



open hardware

open source initiative

Demo!

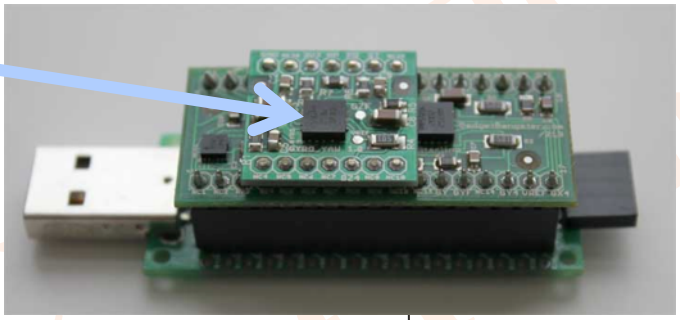
Sensors – Part I

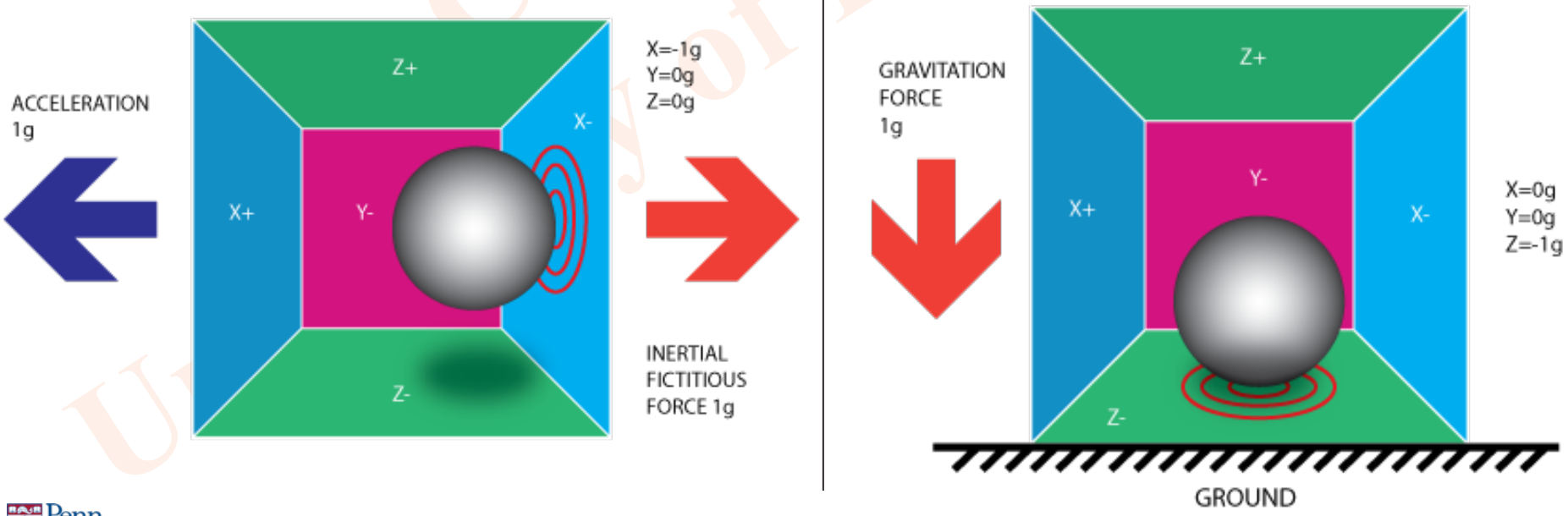
Many types of sensors

- Inertial Measurement Unit (IMU)
- Motion Capture System
- Ultrasound
- Monocular Vision
- Stereo vision
- Laser Scanner
- RGBD
- GPS
- Other

IMU: Accelerometer

- MEMS device that measures acceleration
  - If stationary, will measure the gravity vector





Images on this slide from: http://www.starlino.com/imu_guide.html

IMU: Gyroscope

- MEMS device that measures angular rates
  - In the body frame (i.e. p, q, r)

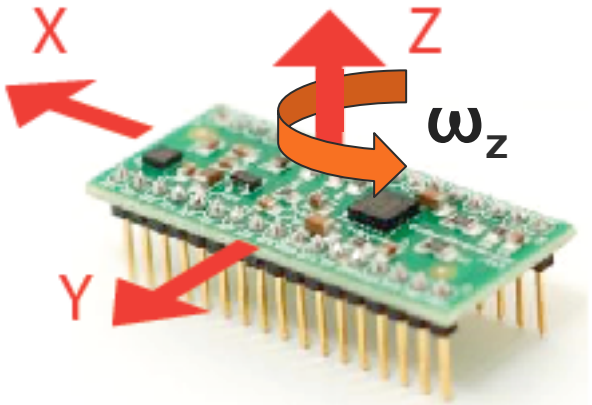
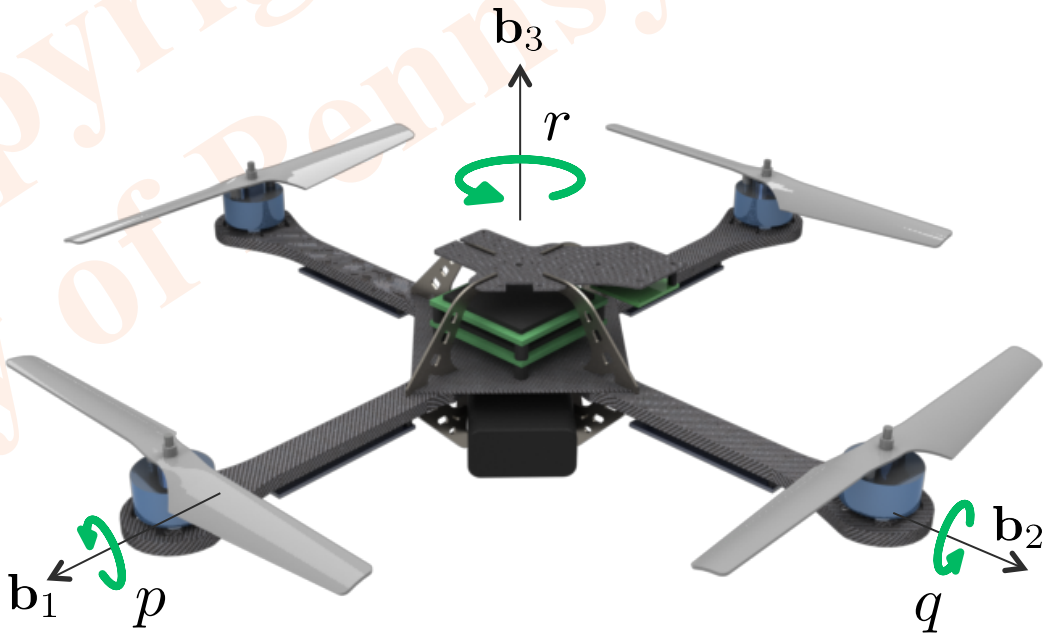
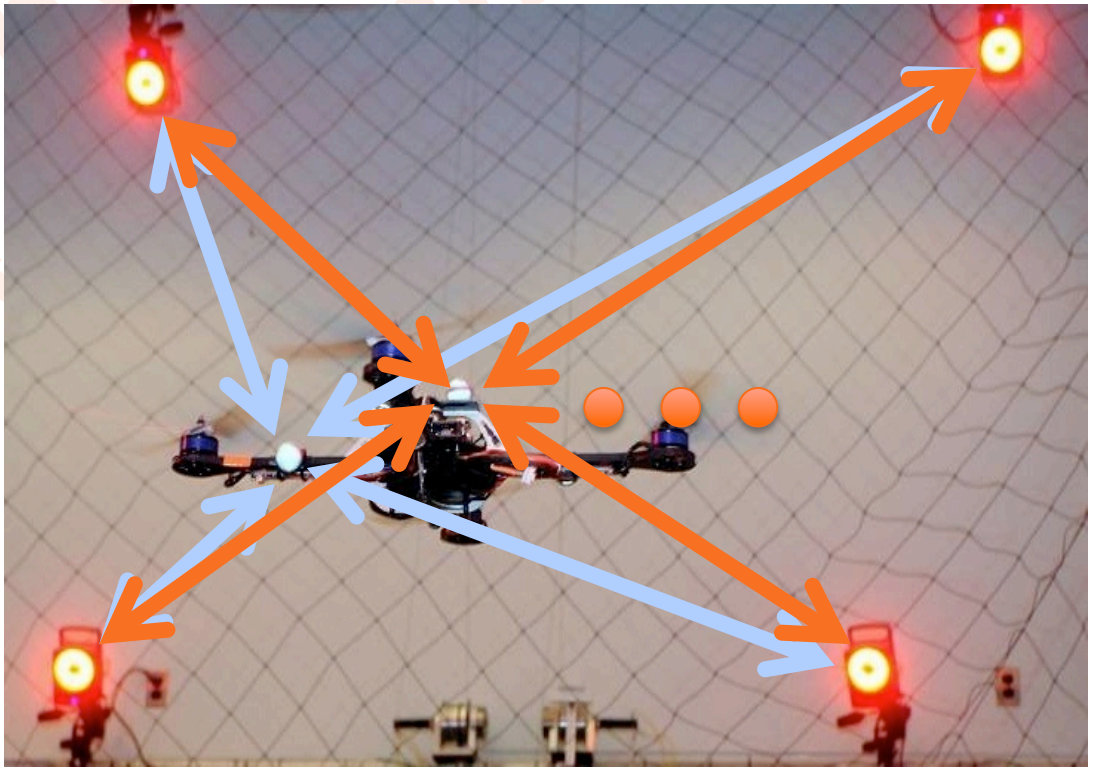


Image from: http://www.starlino.com/imu_guide.html

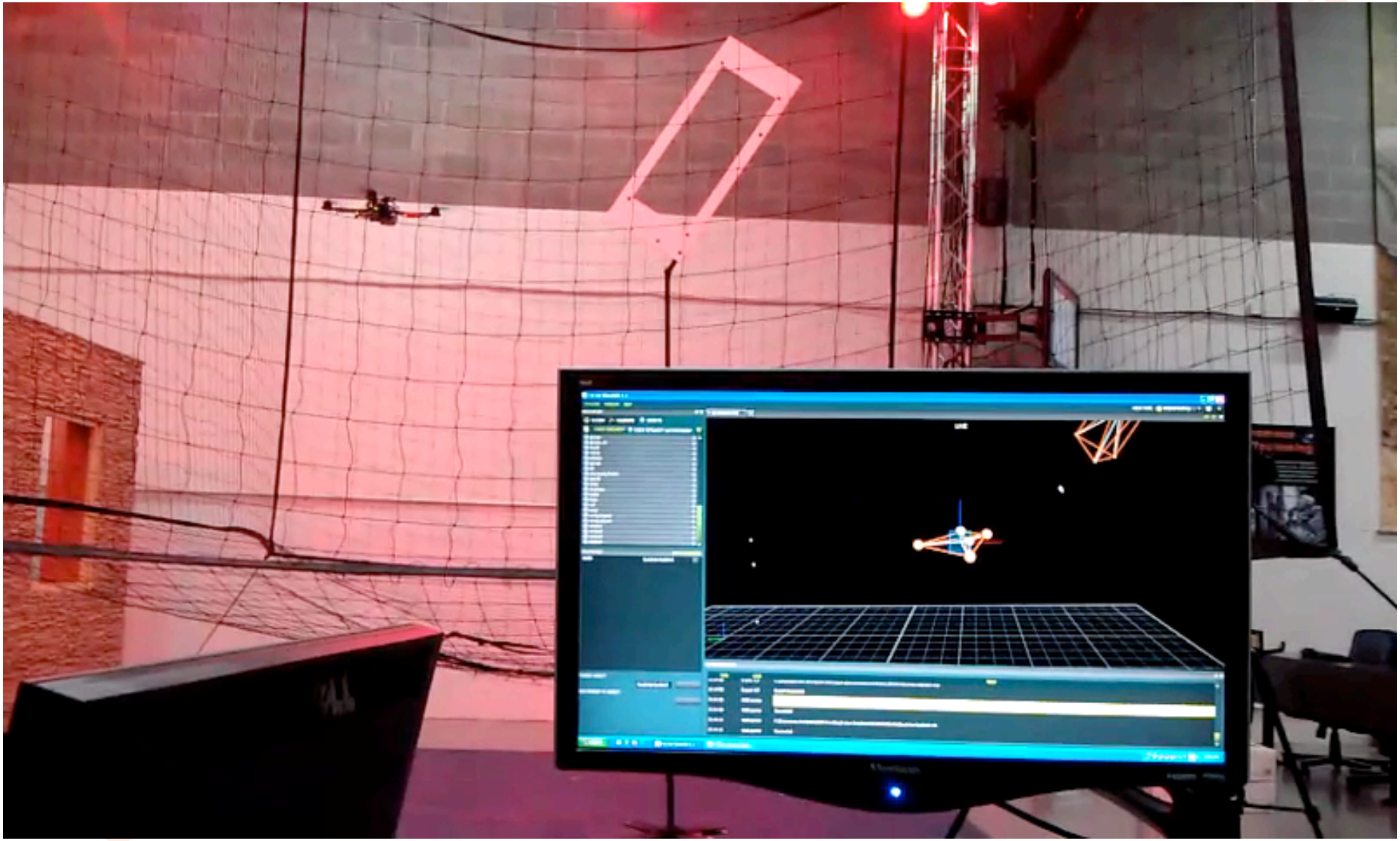


Motion Capture System

- LEDs & cameras
- Retro-Reflective markers on robots
  - Location determined using approach similar to triangulation



Motion Capture System



Motion Capture System

- The unique model of robot is matched using Singular Value Decomposition

Quadrotors in the Laboratory

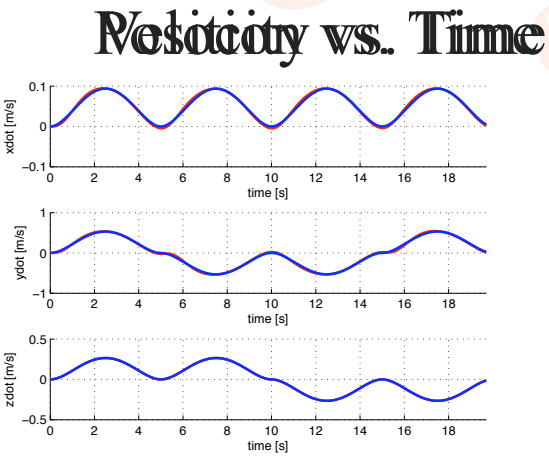
Quadrotors in the Laboratory

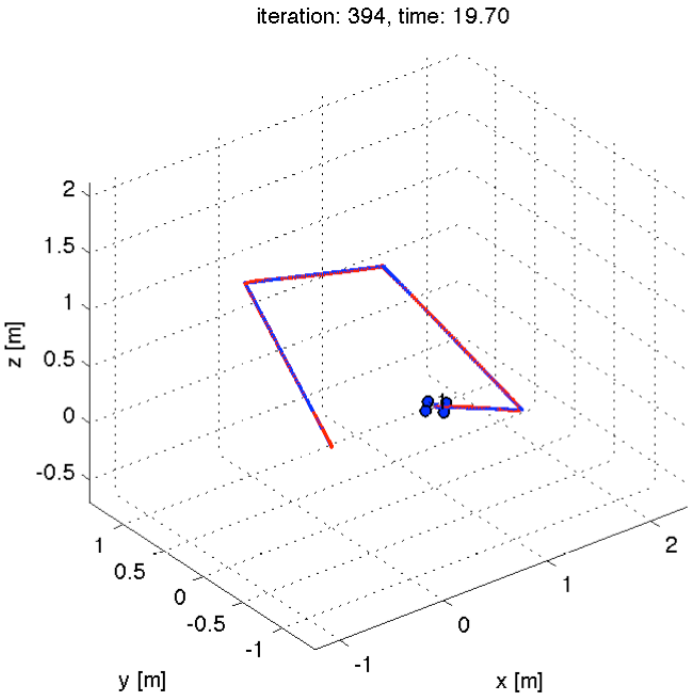
- Section Outline
  - Quadrotor Simulator
    - Position Control
  - Vicon

    Motion Capture System
  - Demos
  - MEAM 620
    - Entry-level course for graduate students (online)

Quadrotors in the Laboratory

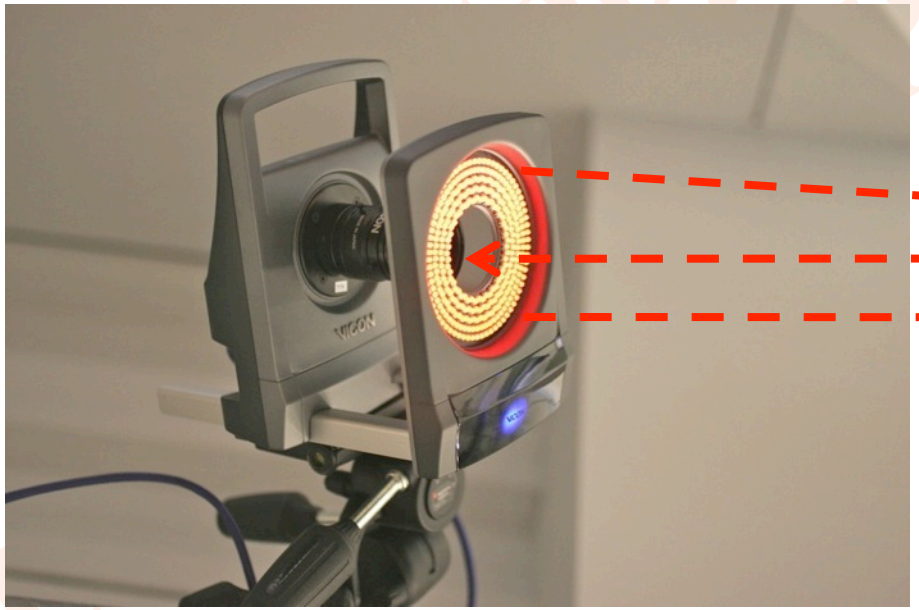
- Quadrotor Simulator
  - Incorporates dynamic model
    - True physics simulator
  - Position Control



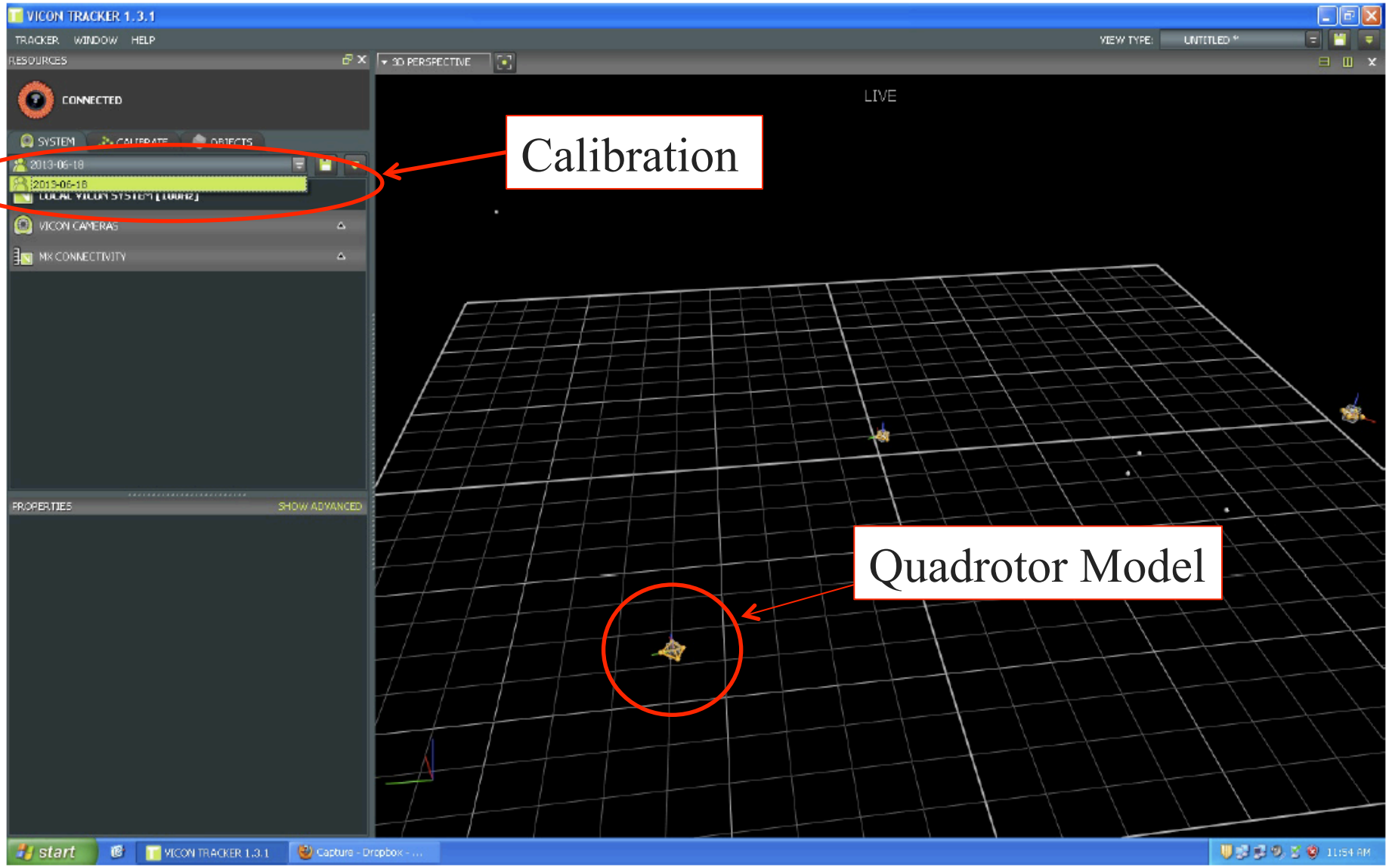


Quadrotors in the Laboratory

- Quadrotor Simulator
  - Position Control
- Vicon
  - Motion Capture System



Quadrotors in the Laboratory



Quadrotors in the Laboratory

- Quadrotor Simulator
  - Position Control
- Vicon
  - Motion Capture System

Position / Pose

[1]www.anandtech.com

Quadrotors in the Laboratory

- MEAM 620: Advanced Robotics
  - Project 1, Phase 1: Dijkstra/A*
  - Project 1, Phase 2: Control and Simulation
  - Project 1, Phase 3: Integrate Phase 1 and 2
  - Project 1, Phase 4: Fly Quadrotor!

Quadrotors in the Laboratory

Quadrotors in the Laboratory

- Quadrotor Simulator
  - Position Control
- Vicon
  - Motion Capture System
- Demos
  - Outreach
    - Middle and high school students

Quadrotors in the Laboratory

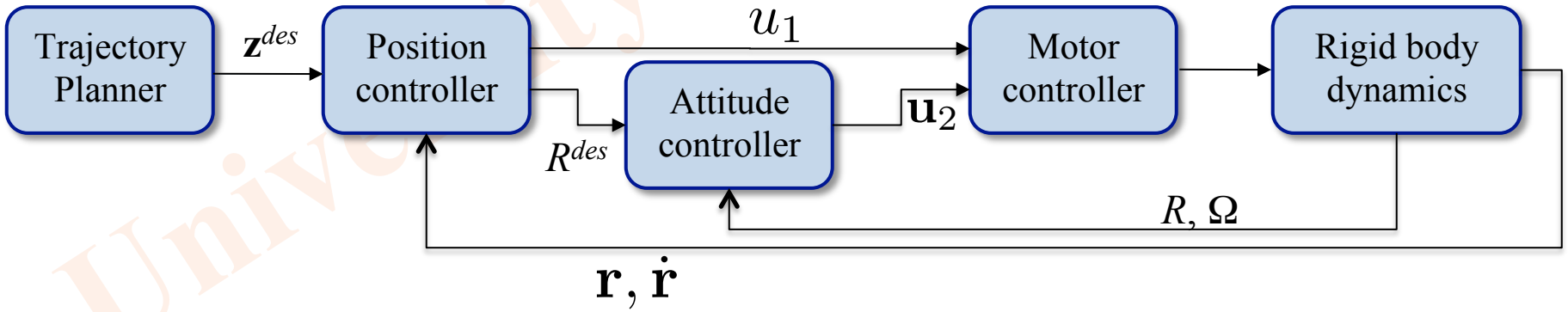
- Gain Tuning
  - Utilize quadrotor simulation
  - Perturb quadrotor while hovering
  - Provide step input and observer response

$$(\ddot{r}_{i,T} - \ddot{r}_i^{\text{des}}) + k_{d,i}(\dot{r}_{i,T} - \dot{r}_i) + k_{p,i}(r_{i,T} - r_i) = 0$$

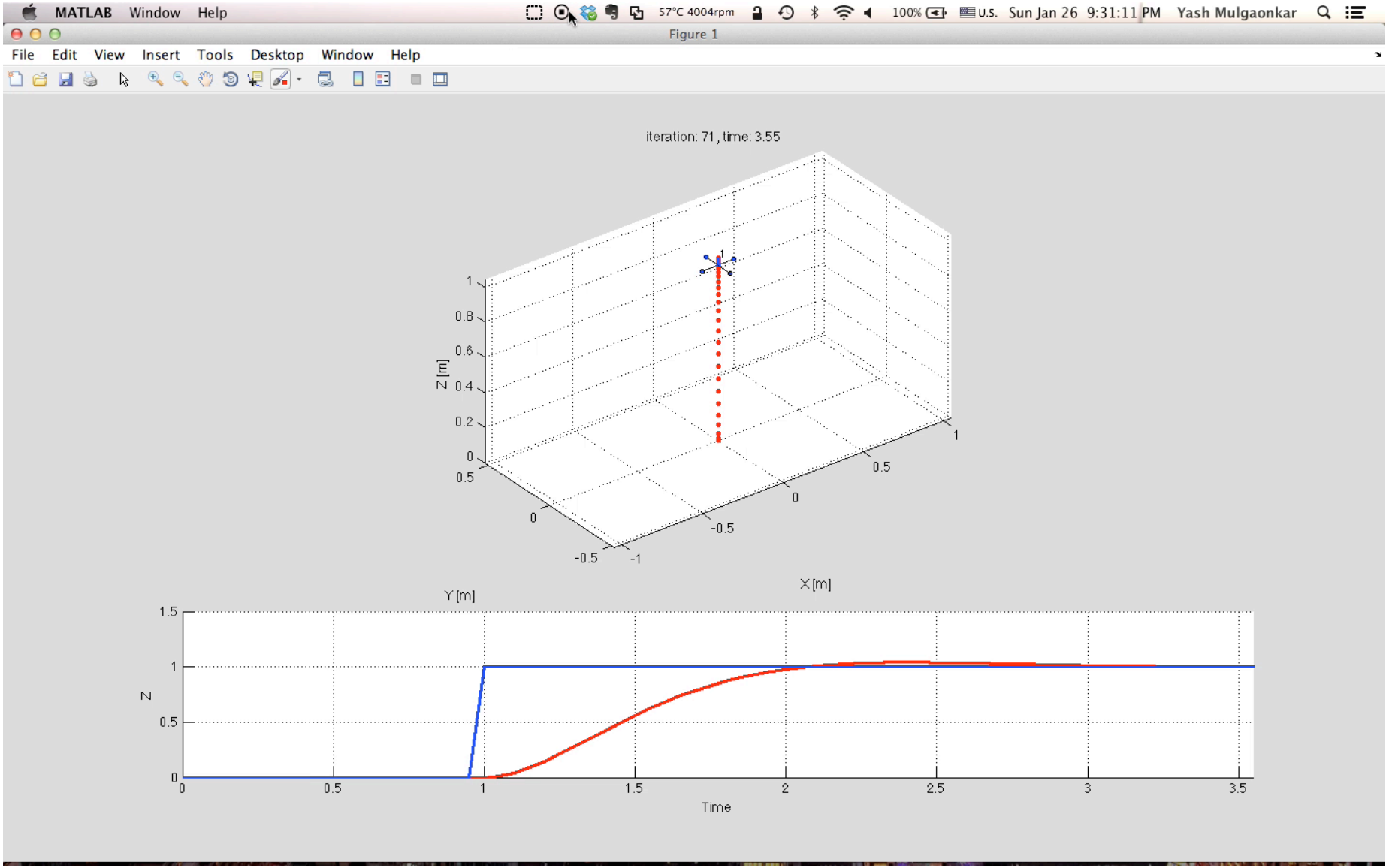
actual (feedback)

specified

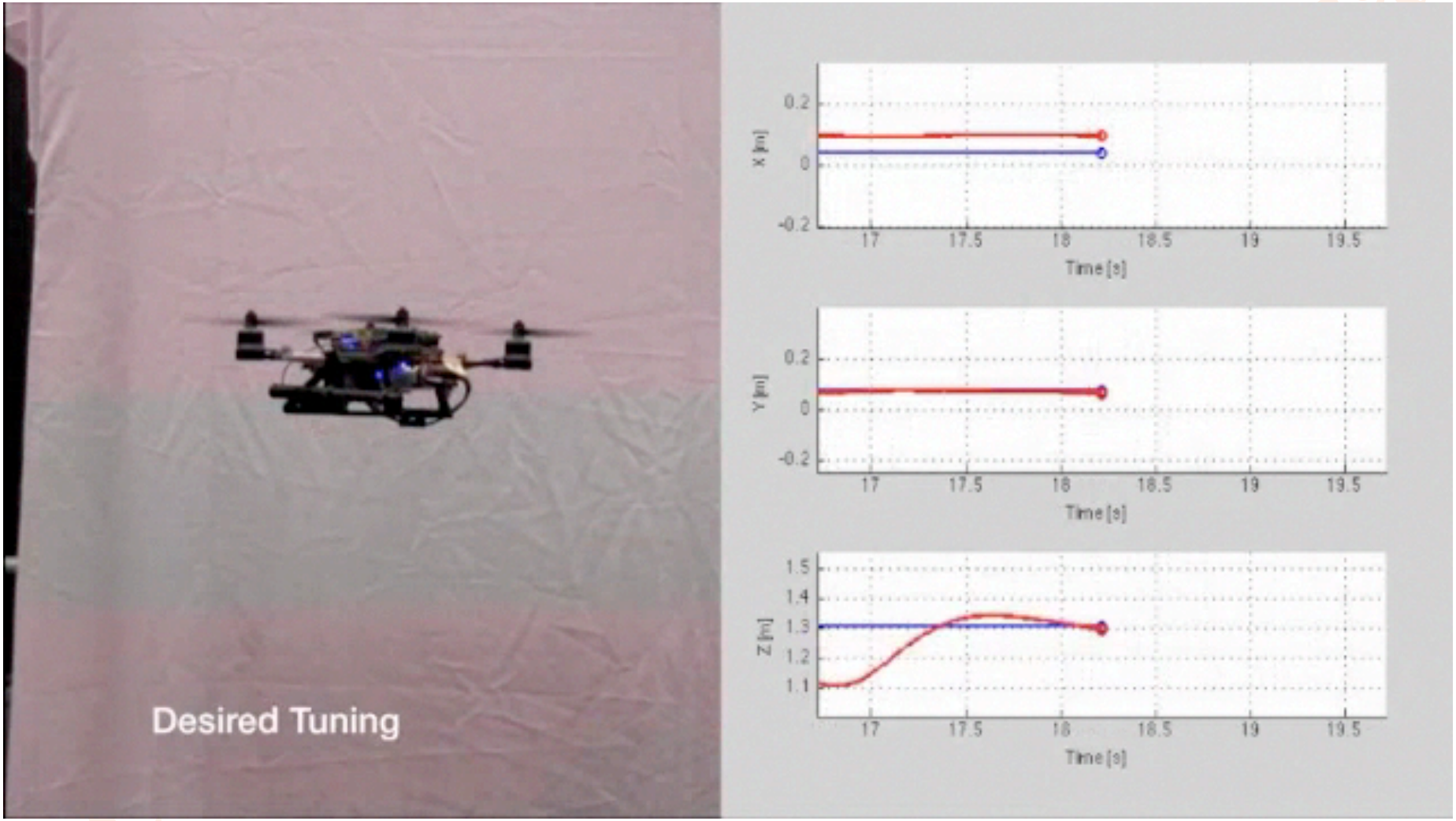
$$u_1 = mg + m\ddot{r}_3^{\text{des}}$$



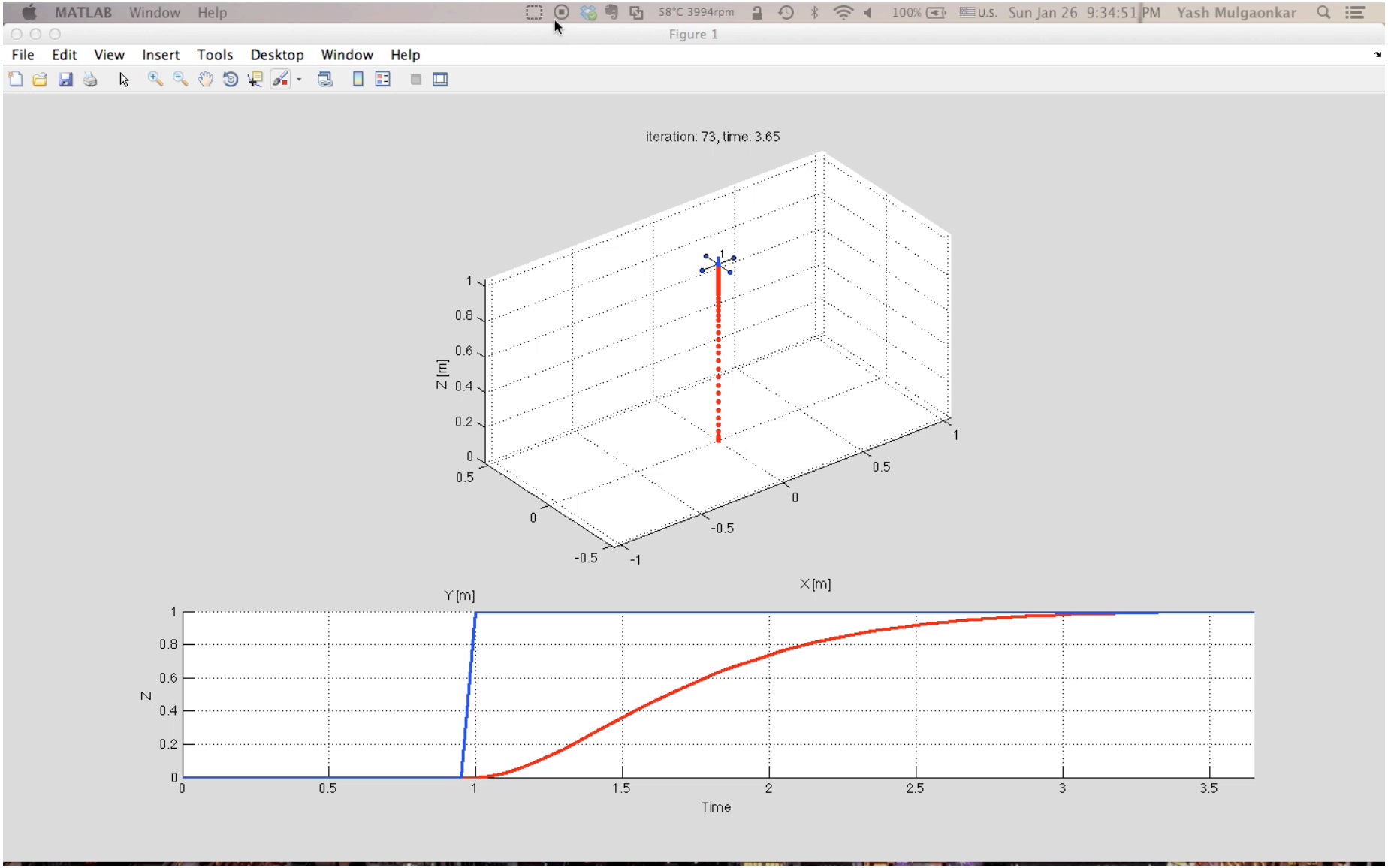
Quadrotors in the Laboratory



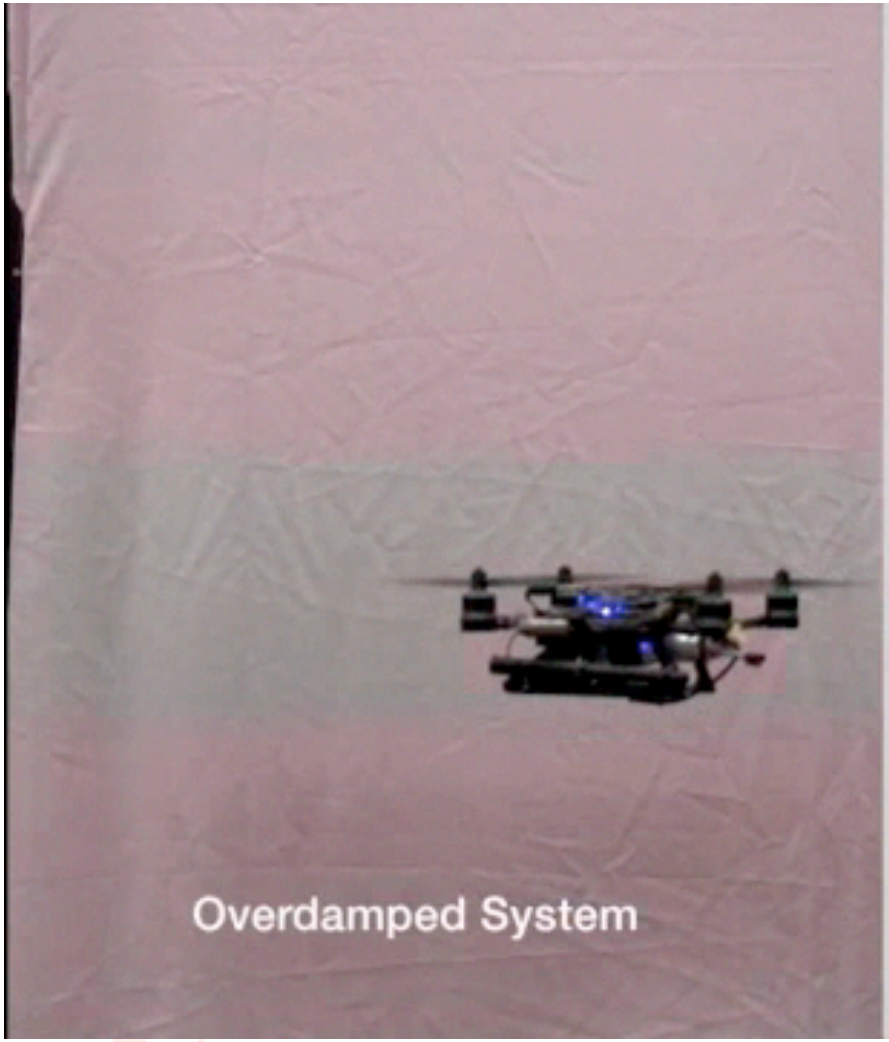
Quadrotors in the Laboratory

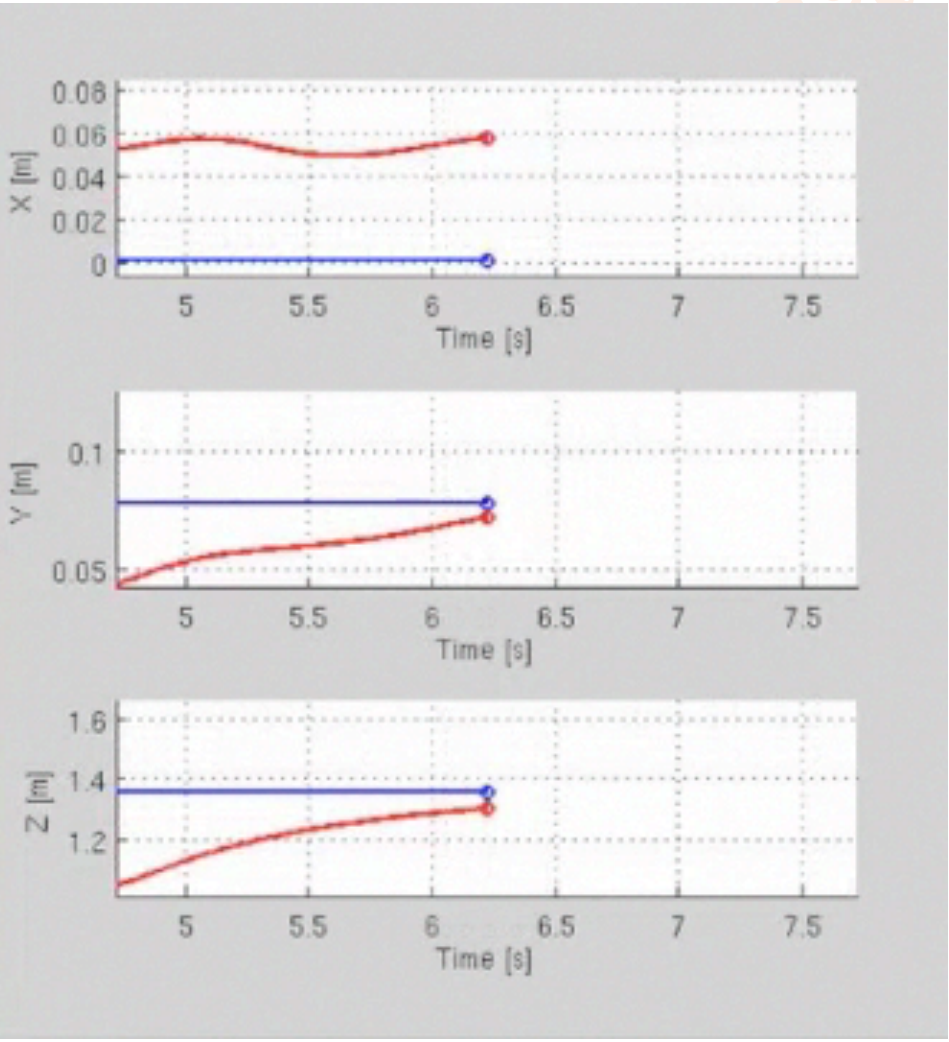


Quadrotors in the Laboratory

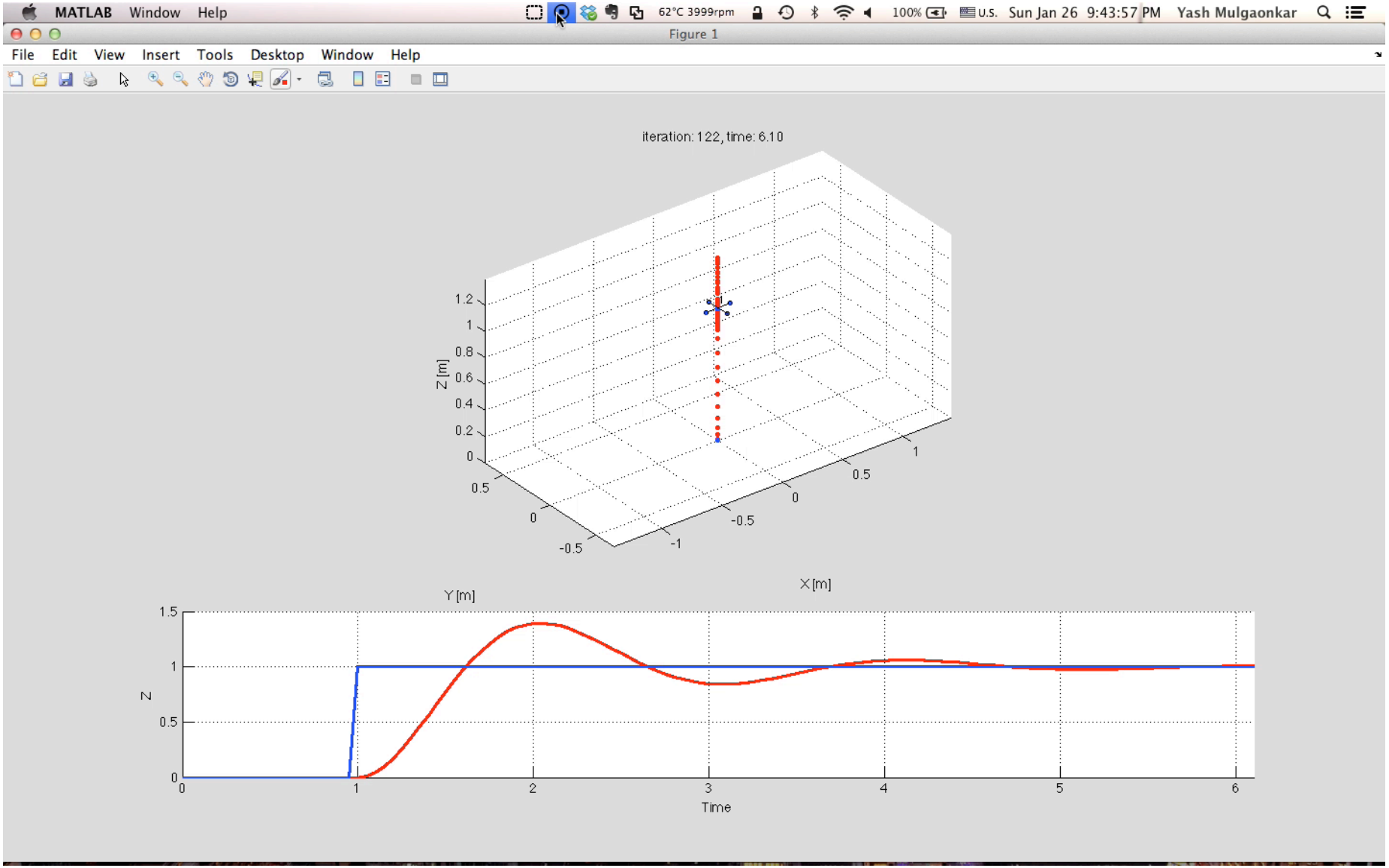


Quadrotors in the Laboratory

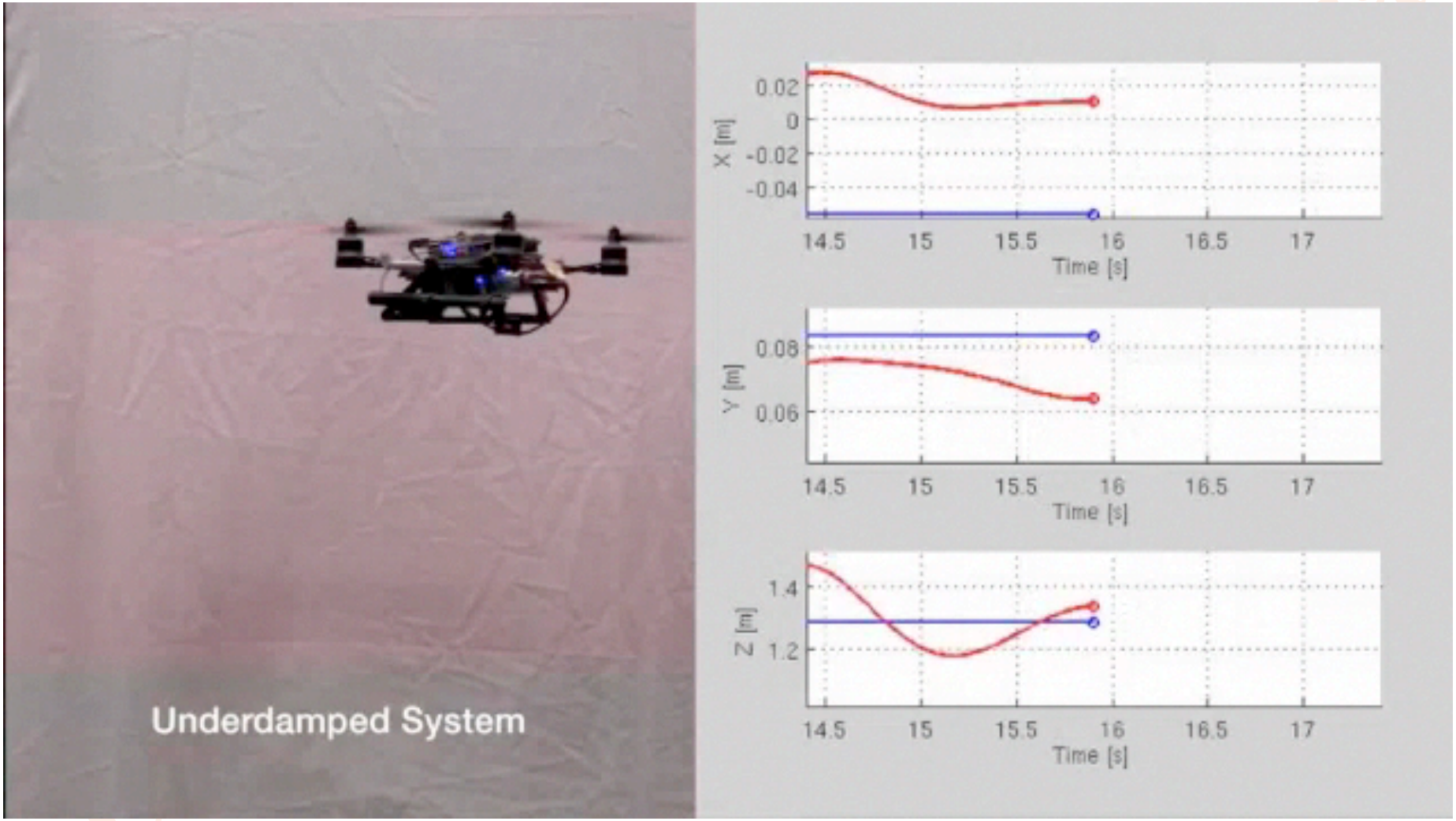




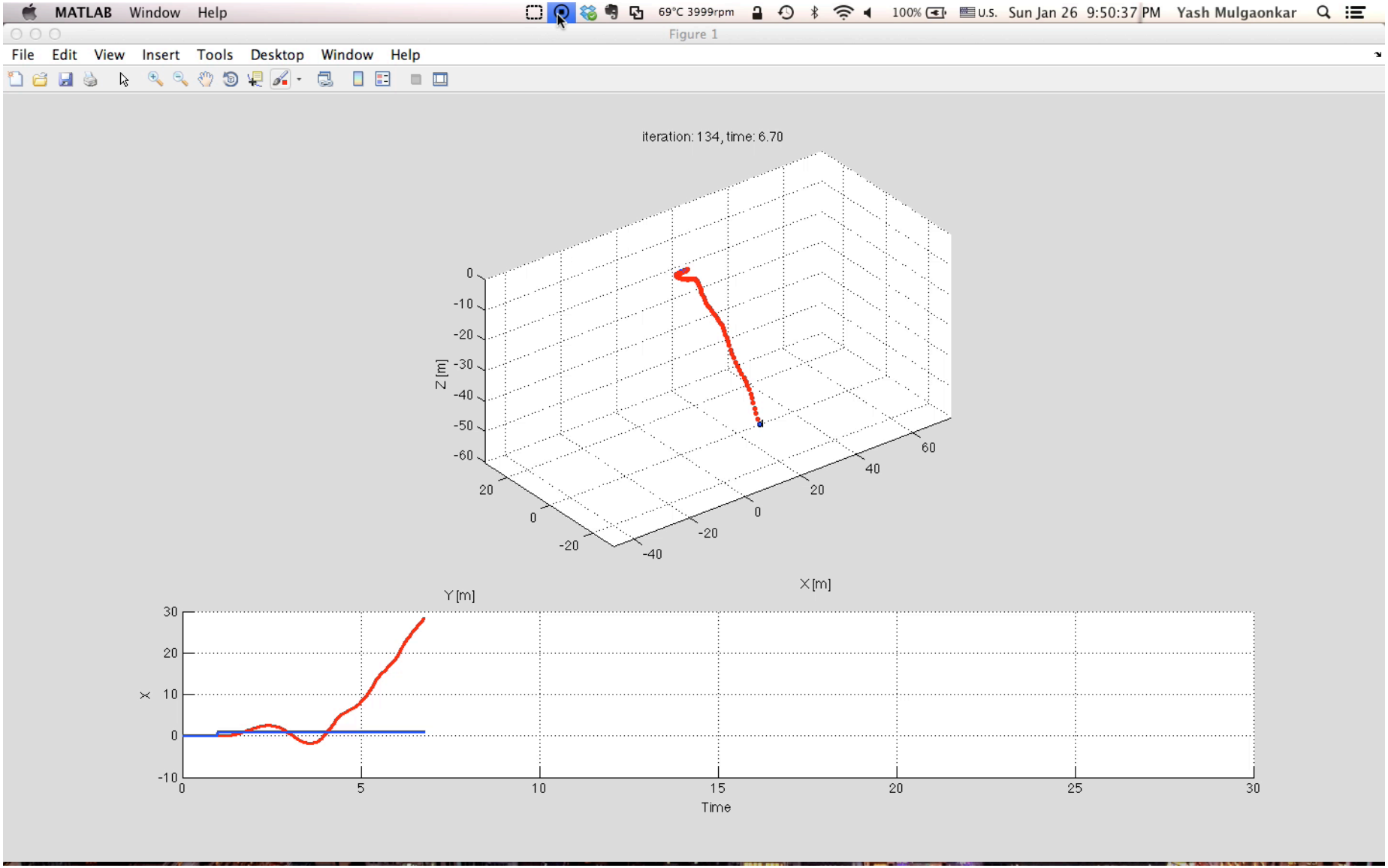
Quadrotors in the Laboratory



Quadrotors in the Laboratory



Quadrotors in the Laboratory



Break

Outline – Section III

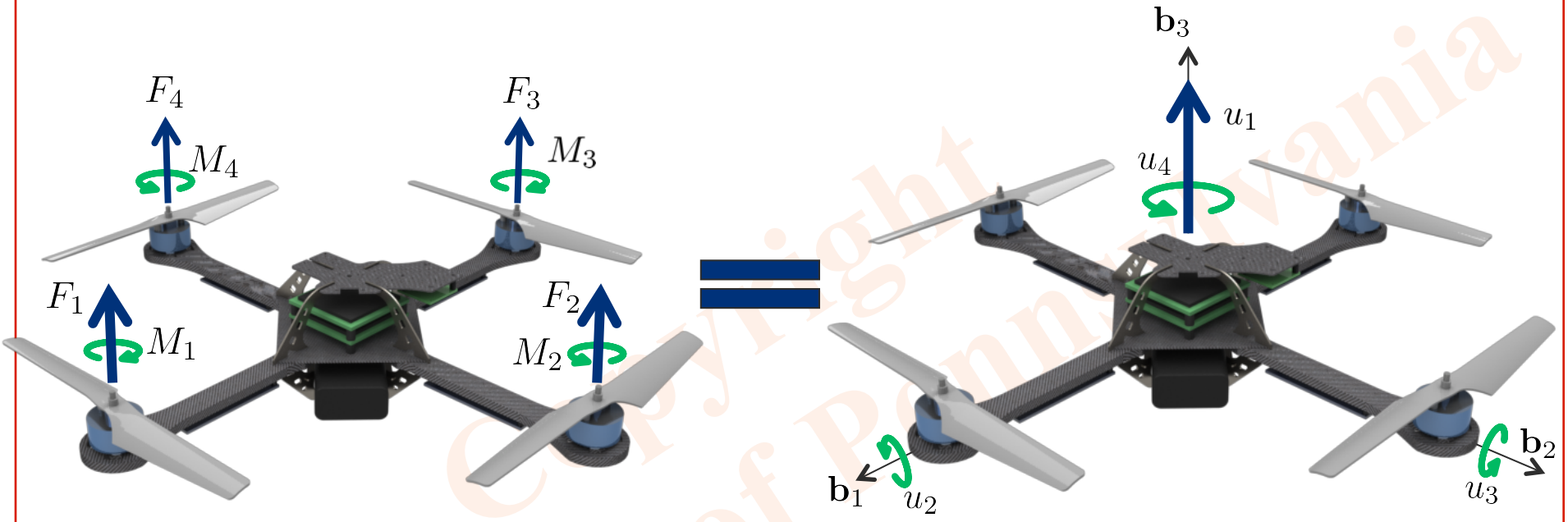
- Demonstration
- Trajectory Generation (Justin Thomas)
  - Differential Flatness and its impliciations
- Quadrotors in the Real World
  - Sensors Part II (Justin Thomas)
  - Planning Algorithms (Vijay Kumar)
    - Dijkstra, A*, RRTs
  - Estimation (Vijay Kumar)
    - Filtering
  - Vision-based localization and control
- Discussion & Close

Trajectory Generation

109

Trajectory Generation

Quick Review



Force: $F_i = k_f \omega_i^2$

Moments: $M_i = \pm k_m \omega_i^2$

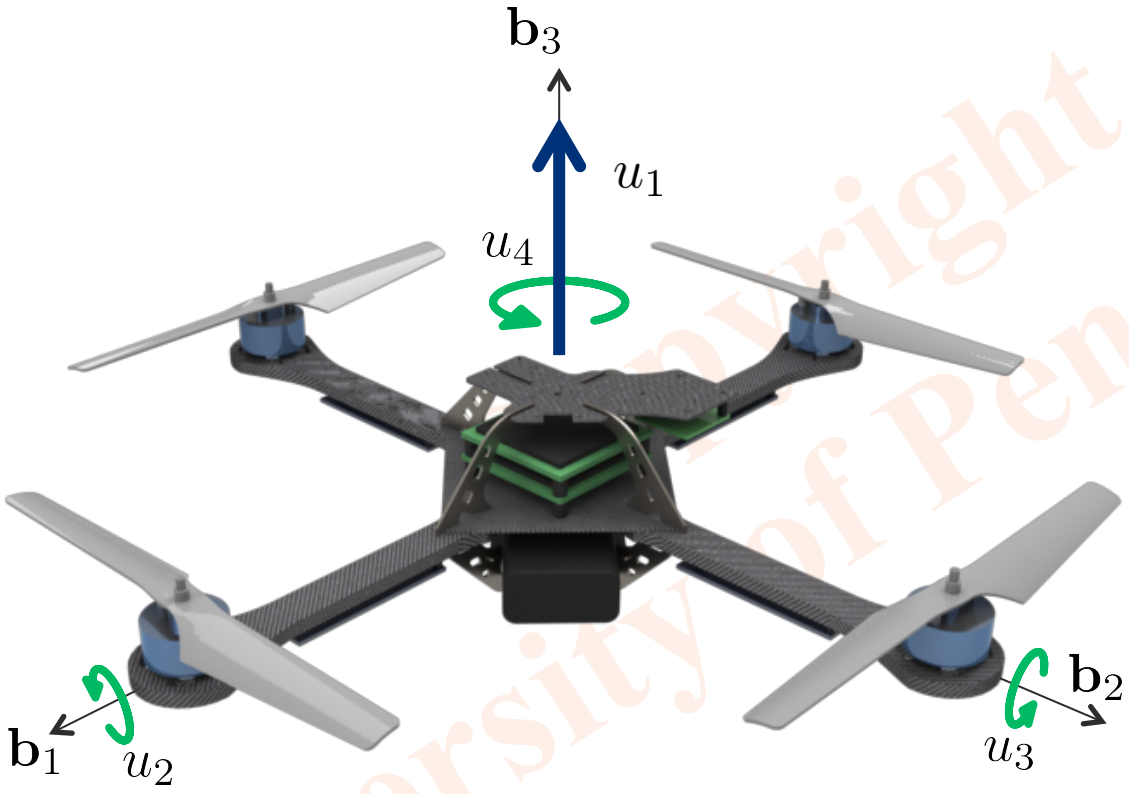
Inputs

Propeller Speeds

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f \\ 0 & lk_f & 0 & -lk_f \\ -lk_f & 0 & lk_f & 0 \\ k_m & -k_m & k_m & -k_m \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

Arm Length: l

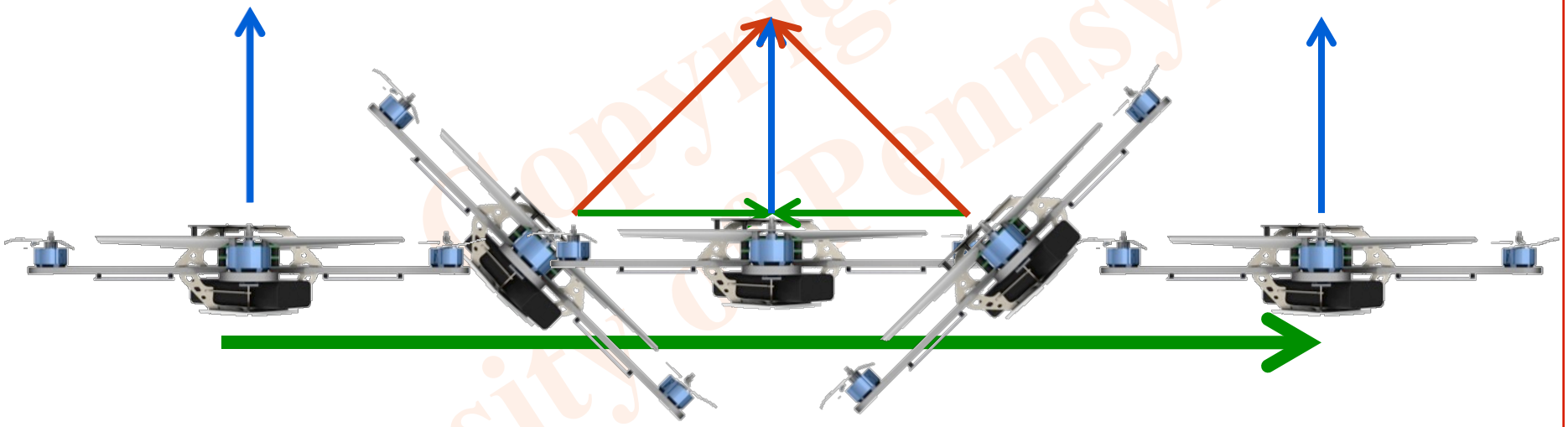
State Vector



$$\mathbf{q} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ \dot{r}_1 \\ \dot{r}_2 \\ \dot{r}_3 \\ \phi \\ \theta \\ \psi \\ {}^{\mathcal{B}}\Omega_1^{\mathcal{B}} \\ {}^{\mathcal{B}}\Omega_2^{\mathcal{B}} \\ {}^{\mathcal{B}}\Omega_3^{\mathcal{B}} \end{bmatrix}$$

12 Dimensions!

Example



$$f(t) = ?$$

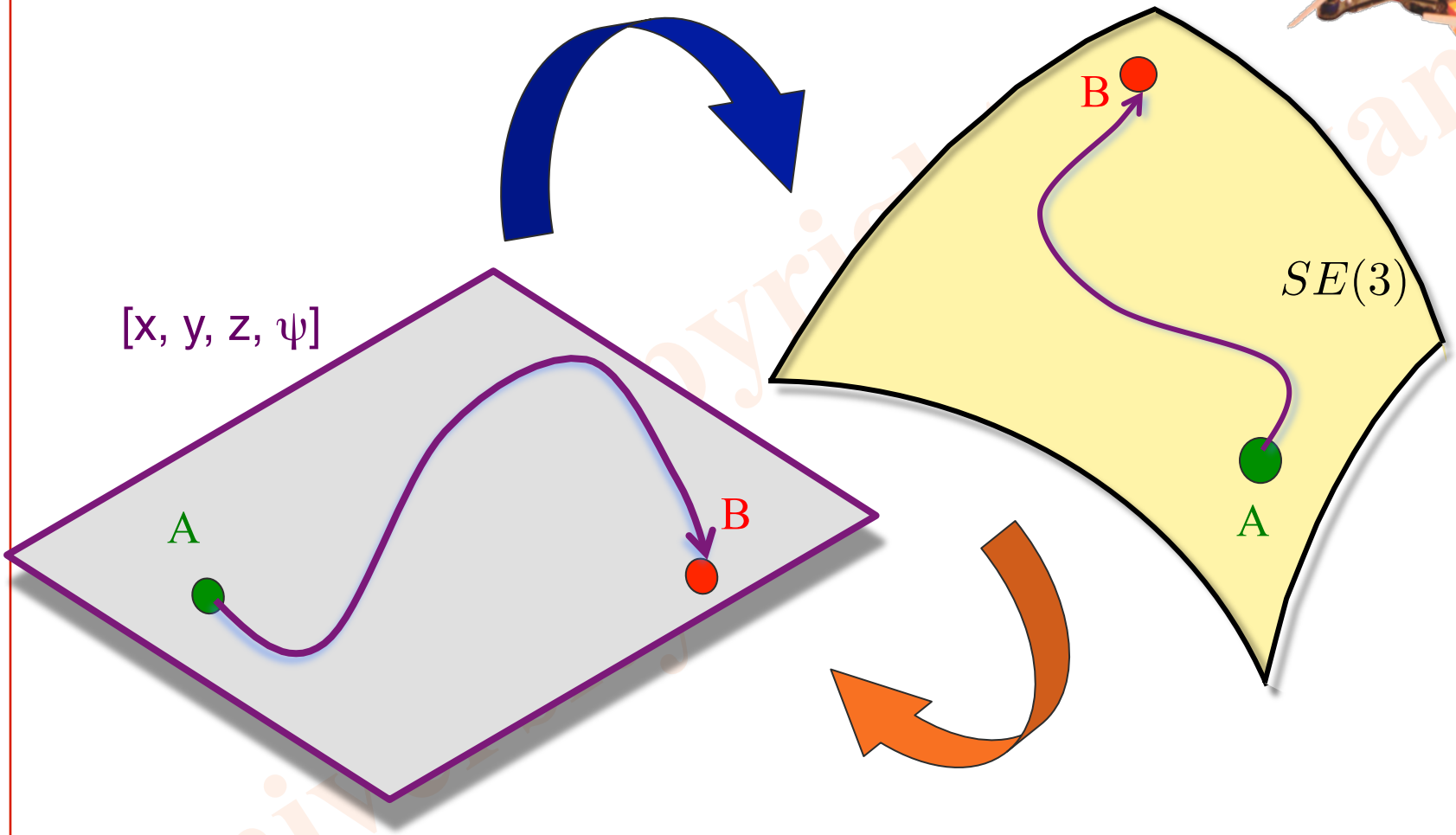
$$\tau(t) = ?$$

Quadrotor: A Differentially Flat System

Penn Engineering

[Fliess et al. 1995], [Murray et al. 1995], [Mellinger, Kumar 2011]

Differentially Flat System



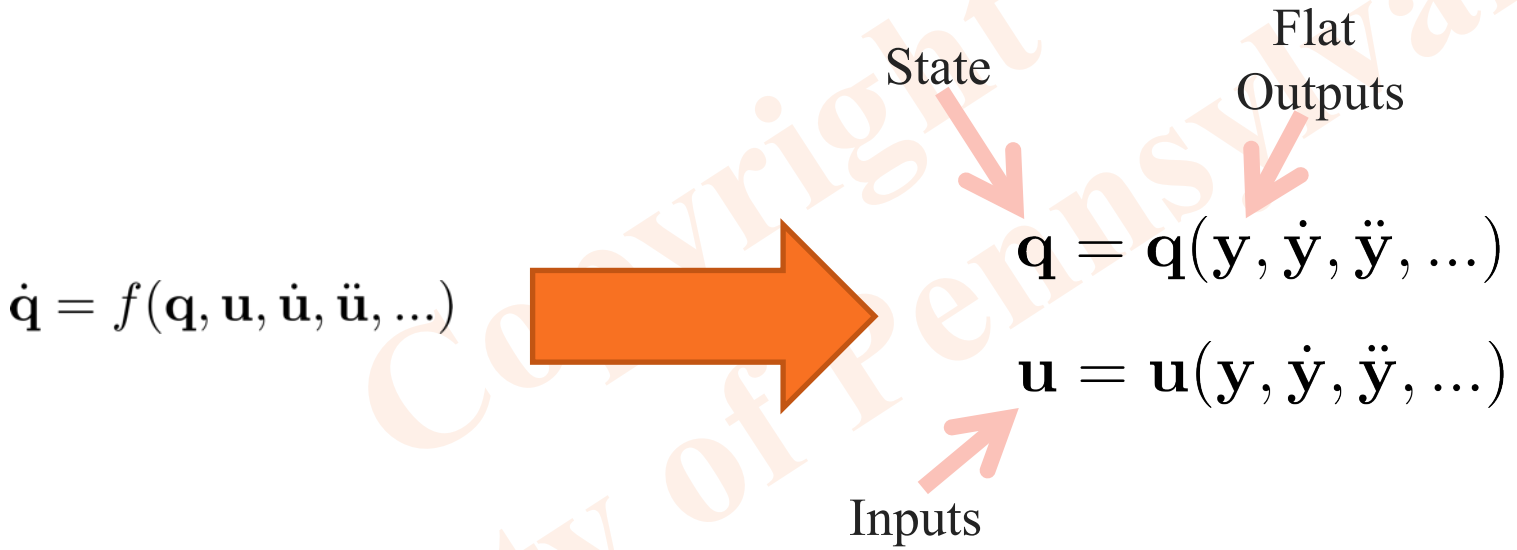
$$\sigma(t) : [0, T] \rightarrow R^3 \times SO(2)$$

$$\begin{bmatrix} x \\ y \\ z \\ \theta \\ \phi \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix}$$

[Mellinger and Kumar, 2011]

Requirements for a Flat System

Flat outputs (y) are a function of the state and inputs



For a Quadrotor:

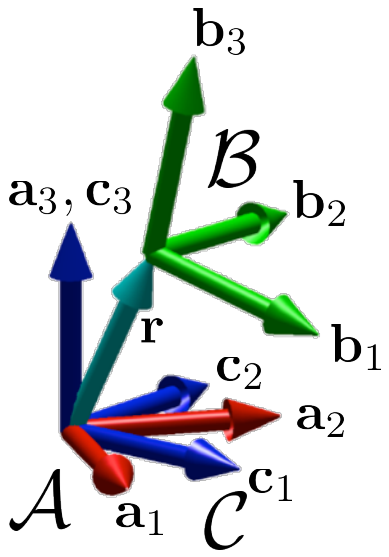
$$\mathbf{y} = \begin{bmatrix} \mathbf{r} \\ \psi \end{bmatrix}$$

$\mathbf{r}$: position

ψ : yaw (rotation about $\mathbf{a}_3$)

[Murray et al. 1995]

Equations of Motion



Angular velocity of frame $\mathcal{B}$ in $\mathcal{A}$ expressed in coordinates of frame $\mathcal{B}$

$${}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$$

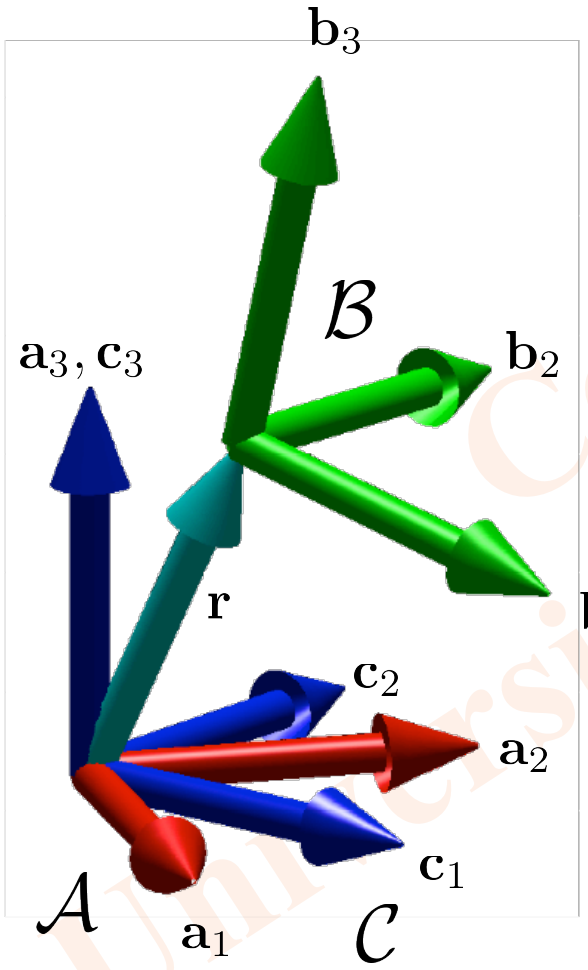
Acceleration — Net Force

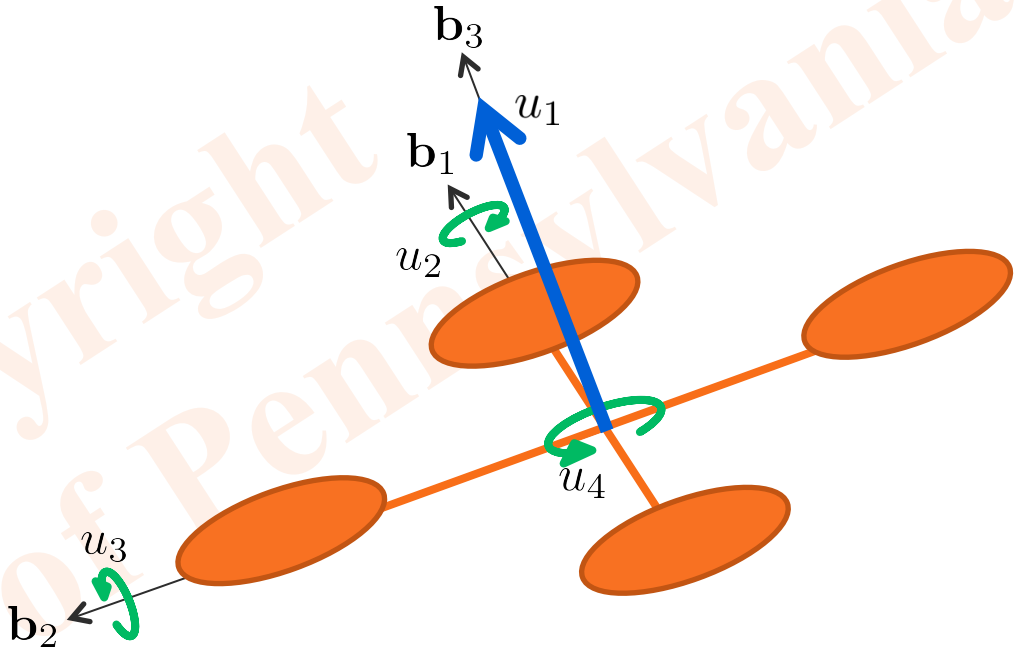
$$m\ddot{\mathbf{r}} = u_1\mathbf{b}_3 - mg\mathbf{a}_3$$

Inertia Tensor — Moments

$$J\,{}^{\mathcal{B}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}} = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} - {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}} \times J\,{}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$$

Coordinate System





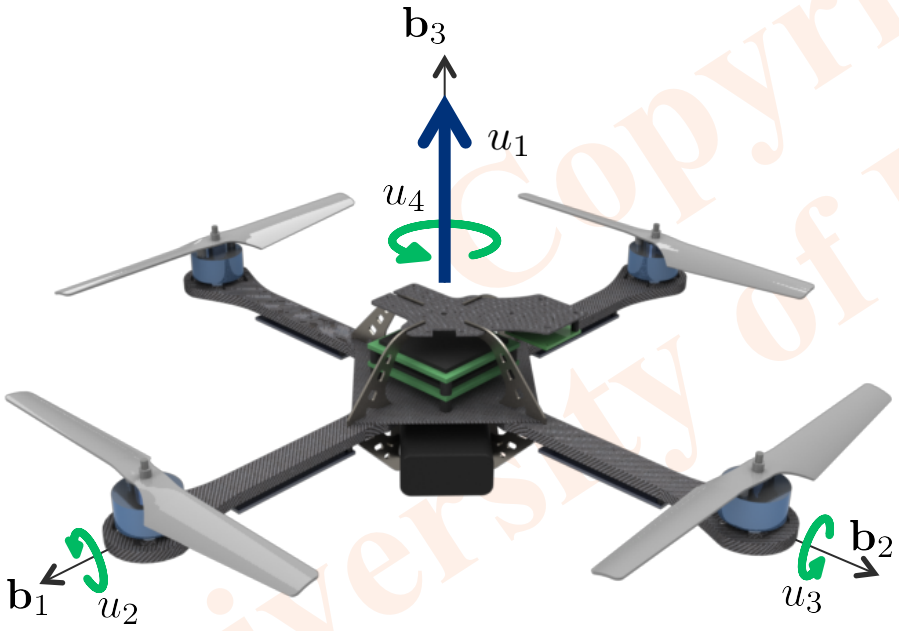
R rotates vectors in $\mathcal{B}$ to vectors in $\mathcal{A}$:

$$\mathbf{b}_3 = R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Thrust and Orientation from Flat Outputs

Total thrust: u_1

$\mathbf{r}$ is in the flat output space, $\mathbf{y}$



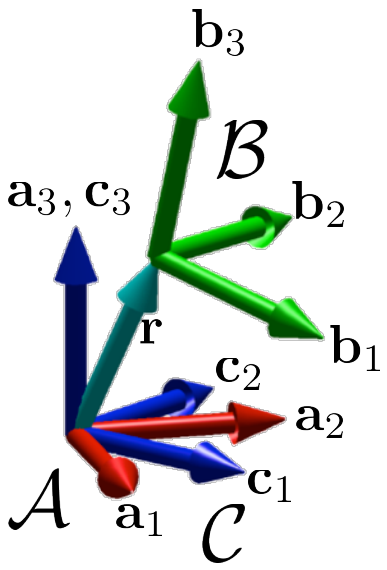
$$m\ddot{\mathbf{r}} = u_1\mathbf{b}_3 - mg\mathbf{a}_3$$

$$\mathbf{b}_3(\mathbf{y}) = \frac{\ddot{\mathbf{r}} + g\mathbf{a}_3}{\|\ddot{\mathbf{r}} + g\mathbf{a}_3\|}$$

$$u_1 = m\,\|\ddot{\mathbf{r}} + g\mathbf{a}_3\|$$

[Mellinger, Kumar 2011]

Use frame C to determine $\mathbf{b}_1$ and $\mathbf{b}_2$



$$\mathbf{c}_1 = \begin{bmatrix} \cos y_4 \\ \sin y_4 \\ 0 \end{bmatrix}, \quad \mathbf{c}_2 = \begin{bmatrix} -\sin y_4 \\ \cos y_4 \\ 0 \end{bmatrix}, \quad \mathbf{c}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$y_4 = \psi$$

Construction for an orthogonal vector:

$$\mathbf{b}_2(\mathbf{y}) = \frac{\mathbf{b}_3(\mathbf{y}) \times \mathbf{c}_1(\mathbf{y})}{\|\mathbf{b}_3(\mathbf{y}) \times \mathbf{c}_1(\mathbf{y})\|}$$

Finish the basis:

$$\mathbf{b}_1(\mathbf{y}) = \mathbf{b}_2(\mathbf{y}) \times \mathbf{b}_3(\mathbf{y})$$

[Mellinger, Kumar 2011], [Lee et al. 2010], [Lee et al. 2011]

Determine Attitude (angular orientation)

R is defined by flat outputs:

$$R = \begin{bmatrix} \mathbf{b}_1(\mathbf{y}) & \mathbf{b}_2(\mathbf{y}) & \mathbf{b}_3(\mathbf{y}) \end{bmatrix}$$

Z-Y-X Rotation allows us to determine Euler angles:

$$R(\phi,\theta,\psi) = \begin{bmatrix} c\theta c\psi & c\psi s\theta s\phi - c\phi s\psi & c\phi c\psi s\theta + s\phi s\psi \\ c\theta s\psi & c\phi c\psi + s\theta s\phi s\psi & -c\psi s\phi + c\phi s\theta s\psi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix}$$

$$s\theta : \ sin(\theta) \qquad c\theta : \ cos(\theta)$$

Translational Dynamics (jerk)

$$m\ddot{\mathbf{r}} = u_1 \mathbf{b}_3 - mg\mathbf{a}_3$$

Differentiate Newton's equations of motion in $\mathcal{A}$

$$m\dddot{\mathbf{r}} = \dot{u}_1 \mathbf{b}_3 + {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1 \mathbf{b}_3$$

Inner product with $\mathbf{b}_3$

$$\dot{u}_1 = \mathbf{b}_3 \cdot m\dddot{\mathbf{r}}$$

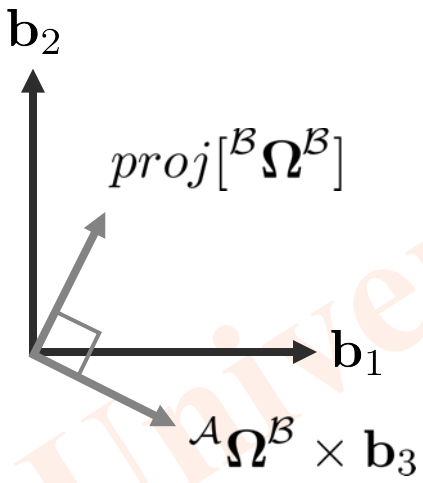
[Mellinger, Kumar 2011]

Determine ${}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$

$$\dot{u}_1 = \mathbf{b}_3 \cdot m\ddot{\mathbf{r}}$$

$$m\ddot{\mathbf{r}} = \dot{u}_1 \mathbf{b}_3 + {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1 \mathbf{b}_3$$

Rearrange: $${}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times \mathbf{b}_3 = \frac{m}{u_1}\left(\ddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right)$$



$$\left|{}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times \mathbf{b}_3\right| = \left|proj\left[{}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}\right]\right|$$

$${}^{\mathcal{B}}\Omega_1^{\mathcal{B}} = -\mathbf{b}_2 \cdot \left(\frac{m}{u_1}\left(\ddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right)\right)$$

$${}^{\mathcal{B}}\Omega_2^{\mathcal{B}} = \mathbf{b}_1 \cdot \left(\frac{m}{u_1}\left(\ddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right)\right)$$

Determine ${}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$

$$ {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}} = \left(R^T(t)\dot{R}(t) \right)^{\vee} $$

$$ \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}^{\vee} $$

Express in this form:

$$ {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} $$

$$ {}^{\mathcal{B}}\Omega_3^{\mathcal{B}} = \dot{\psi}\cos\theta\sec\phi - {}^{\mathcal{B}}\Omega_2^{\mathcal{B}}\tan\phi $$

[Bullo, Lewis 2005]

Translational Dynamics (snap)

$$m\ddot{\ddot{\mathbf{r}}} = \dot{u}_1\mathbf{b}_3 + {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3$$

Differentiate…

$$m\mathbf{r}^{(4)} = \ddot{u}_1\mathbf{b}_3 + 2{}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times \dot{u}_1\mathbf{b}_3 + {}^{\mathcal{A}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}} \times u_1\mathbf{b}_3 + {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3$$

Inner product with $\mathbf{b}_3$:

$$\mathbf{b}_3 \cdot m\mathbf{r}^{(4)} = \mathbf{b}_3 \cdot \ddot{u}_1\mathbf{b}_3 + \mathbf{b}_3 \cdot 2{}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times \dot{u}_1\mathbf{b}_3 + \mathbf{b}_3 \cdot {}^{\mathcal{A}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}} \times u_1\mathbf{b}_3 + \mathbf{b}_3 \cdot {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3$$

Solve for $\ddot{u}_1$:

$$\ddot{u}_1 = \mathbf{b}_3 \cdot m\mathbf{r}^{(4)} - \mathbf{b}_3 \cdot \left({}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times {}^{\mathcal{A}}\boldsymbol{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3\right)$$

Moment Inputs

Recall:

Angular velocity of frame $\mathcal{B}$ in $\mathcal{A}$ in coordinates of frame $\mathcal{B}$

J : Inertia tensor

$${}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$$

Rearrange Newton-Euler equations of motion and solve for moments:

$$\begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} = J\,{}^{\mathcal{B}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}} + {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}} \times J\,{}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}$$

The Flat System

$$\boldsymbol{\xi}_i = \begin{bmatrix} y_i \\ \dot{y}_i \\ \ddot{y}_i \\ \dddot{y}_i \end{bmatrix} \text{ for } i = 1, 2, 3 \qquad \boldsymbol{\xi}_4 = \begin{bmatrix} y_4 \\ \dot{y}_4 \end{bmatrix}$$

$$\dot{\boldsymbol{\xi}}_i = \begin{bmatrix} 0_{3\times1} & I_{3\times3} \\ 0_{1\times1} & 0_{1\times3} \end{bmatrix} \boldsymbol{\xi}_i + \begin{bmatrix} 0_{3\times1} \\ 1 \end{bmatrix} y_i^{(4)} \qquad \dot{\boldsymbol{\xi}}_4 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{\xi}_4 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} y_4^{(2)}$$

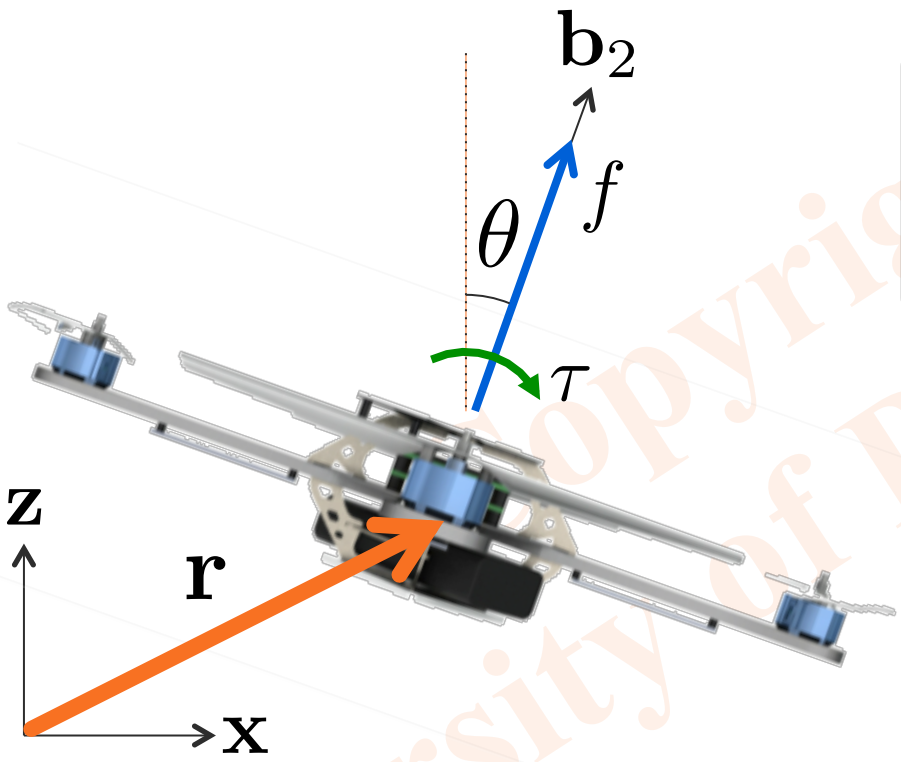
Minimize Snap

Minimize Acceleration

Chain of Integrators!

[Murray et al. 1995]

The Planar Case



$$m\ddot{\mathbf{r}} = fR\mathbf{e}_2 - mg\mathbf{e}_2$$

$$\ddot{\theta} = \frac{\tau}{I}$$

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

$$\mathbf{u} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

The Planar Case

Determine the thrust (first input)

$$\mathbf{u} = \mathbf{u}\left(\mathbf{y}, \dot{\mathbf{y}}, \ddot{\mathbf{y}}, \ldots\right)$$

$$\mathbf{u} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} x \\ z \end{bmatrix}$$

$$m\ddot{\mathbf{r}} = fR\mathbf{e}_2 - mg\mathbf{e}_2$$

$$fR\mathbf{e}_2 = m\ddot{\mathbf{r}} + mg\mathbf{e}_2$$

$$\|fR\mathbf{e}_2\| = m\|\ddot{\mathbf{r}} + g\mathbf{e}_2\|$$

$$f = m\|\ddot{\mathbf{r}} + g\mathbf{e}_2\|$$

The Planar Case

Determine the attitude (second input)

Since: $\mathbf{b}_2 = R\mathbf{e}_2$

$$f\mathbf{b}_2 = m\ddot{\mathbf{r}} + mg\mathbf{e}_2$$

$$\mathbf{b}_2 = \frac{m\ddot{\mathbf{r}} + mg\mathbf{e}_2}{f} = \frac{\ddot{\mathbf{r}} + g\mathbf{e}_2}{\|\ddot{\mathbf{r}} + g\mathbf{e}_2\|}$$

And we know…

$$\mathbf{b}_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{b}_2$$

$$R = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \end{bmatrix}$$

The Planar Case

$$m\ddot{\mathbf{r}} = fR\mathbf{e}_2 - mg\mathbf{e}_2$$

$$m\dddot{\mathbf{r}} = fR\dot{\hat{\theta}}\mathbf{e}_2 + \dot{f}R\mathbf{e}_2$$

$$\dot{R} = R\dot{\hat{\theta}}$$

$$\hat{a} = \begin{bmatrix} 0 & a \\ -a & 0 \end{bmatrix}$$

Projection along $R\mathbf{e}_2$

$$m\mathbf{e}_2^T R^T \dddot{\mathbf{r}} = f\mathbf{e}_2^T \dot{\hat{\theta}}\mathbf{e}_2 + \dot{f}$$

$$\dot{f} = m\mathbf{e}_2^T R^T \dddot{\mathbf{r}}$$

Projection along $R\mathbf{e}_1$

$$m\mathbf{e}_1^T R^T \dddot{\mathbf{r}} = f\mathbf{e}_1 \dot{\hat{\theta}}\mathbf{e}_2 + \dot{f}\mathbf{e}_1^T\mathbf{e}_2$$

$$\dot{\theta} = m\frac{\mathbf{e}_1^T R^T \dddot{\mathbf{r}}}{f}$$

The Planar Case

$$m\ddot{\mathbf{r}} = f R\dot{\hat{\theta}}\mathbf{e}_2 + \dot{f} R\mathbf{e}_2$$

$$m\ddot{\mathbf{r}} = f R\ddot{\hat{\theta}}\mathbf{e}_2 + \dot{f} R\dot{\hat{\theta}}\mathbf{e}_2 + f R\dot{\hat{\theta}}\dot{\hat{\theta}}\mathbf{e}_2 + \dot{f} R\dot{\hat{\theta}}\mathbf{e}_2 + \ddot{f} R\mathbf{e}_2$$

$$m\ddot{\mathbf{r}} = f R\ddot{\hat{\theta}}\mathbf{e}_2 + 2\dot{f} R\dot{\hat{\theta}}\mathbf{e}_2 + -\dot{\theta}^2 f R\mathbf{e}_2 + \ddot{f} R\mathbf{e}_2$$

$$m\ddot{\mathbf{r}} = f R\ddot{\hat{\theta}}\mathbf{e}_2 + 2\dot{f} R\dot{\hat{\theta}}\mathbf{e}_2 + \left(\ddot{f} - \dot{\theta}^2 f\right) R\mathbf{e}_2$$

The Planar Case

$$m\dddot{\mathbf{r}} = f R\dot{\hat{\theta}}\mathbf{e}_2 + \dot{f} R\mathbf{e}_2$$

$$m\ddddot{\mathbf{r}} = f R\ddot{\hat{\theta}}\mathbf{e}_2 + 2\dot{f} R\dot{\hat{\theta}}\mathbf{e}_2 + \left(\ddot{f} - \dot{\theta}^2 f\right) R\mathbf{e}_2$$

Projection along $R\mathbf{e}_2$

$$\ddot{f} = m\mathbf{e}_2^T R^T \ddddot{\mathbf{r}} - \dot{\theta}^2 f$$

Projection along $R\mathbf{e}_1$

$$\ddot{\theta} = \frac{m\mathbf{e}_1^T R^T \ddddot{\mathbf{r}} - 2\dot{f}\dot{\theta}}{f}$$

The Planar Case

- Recall the dynamics:

$$\ddot{\theta} = \ddot{\theta}\left(\ddot{\mathbf{r}}, \dddot{\mathbf{r}}, \ddddot{\mathbf{r}}\right) \quad \Rightarrow \quad \tau = \tau\left(\ddot{\mathbf{r}}, \dddot{\mathbf{r}}, \ddddot{\mathbf{r}}\right)$$

- The input to our system is a function of the 4th derivative of position (i.e. "snap")

- Other assumptions result in different results
  - What if we assume that attitude is a control input?

Trajectory Generation

What order?

$$n \geq 7$$

Why?

- Boundary conditions (8)
- The system requires continuous derivatives below snap

Trajectory Generation

Optimization

- Minimize Snap of Position
- Minimize Acceleration of Yaw

Basis

- Polynomials

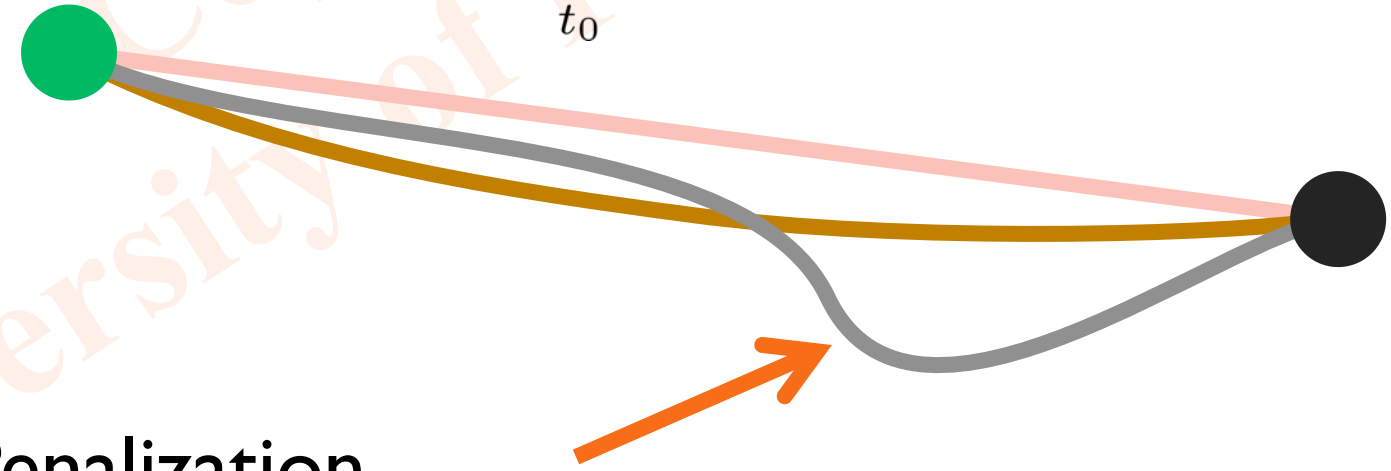
$$\mathbf{g} = \begin{bmatrix} 1 \\ t \\ t^2 \\ \vdots \\ t^n \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix}$$

$$\mathbf{g}^T\mathbf{c} = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \ldots + \alpha_n t^n$$

Cost Functionals

$$J = \int_{t_0}^{t_f} \left\| y_i^{(4)}(t) \right\|^2 dt \quad \text{for} \quad i = 1, 2, 3$$

$$J = \int_{t_0}^{t_f} \left\| y_4^{(2)}(t) \right\|^2 dt$$



Quadratic Penalization

[Mellinger, Kumar 2011]

Euler Lagrange Solution

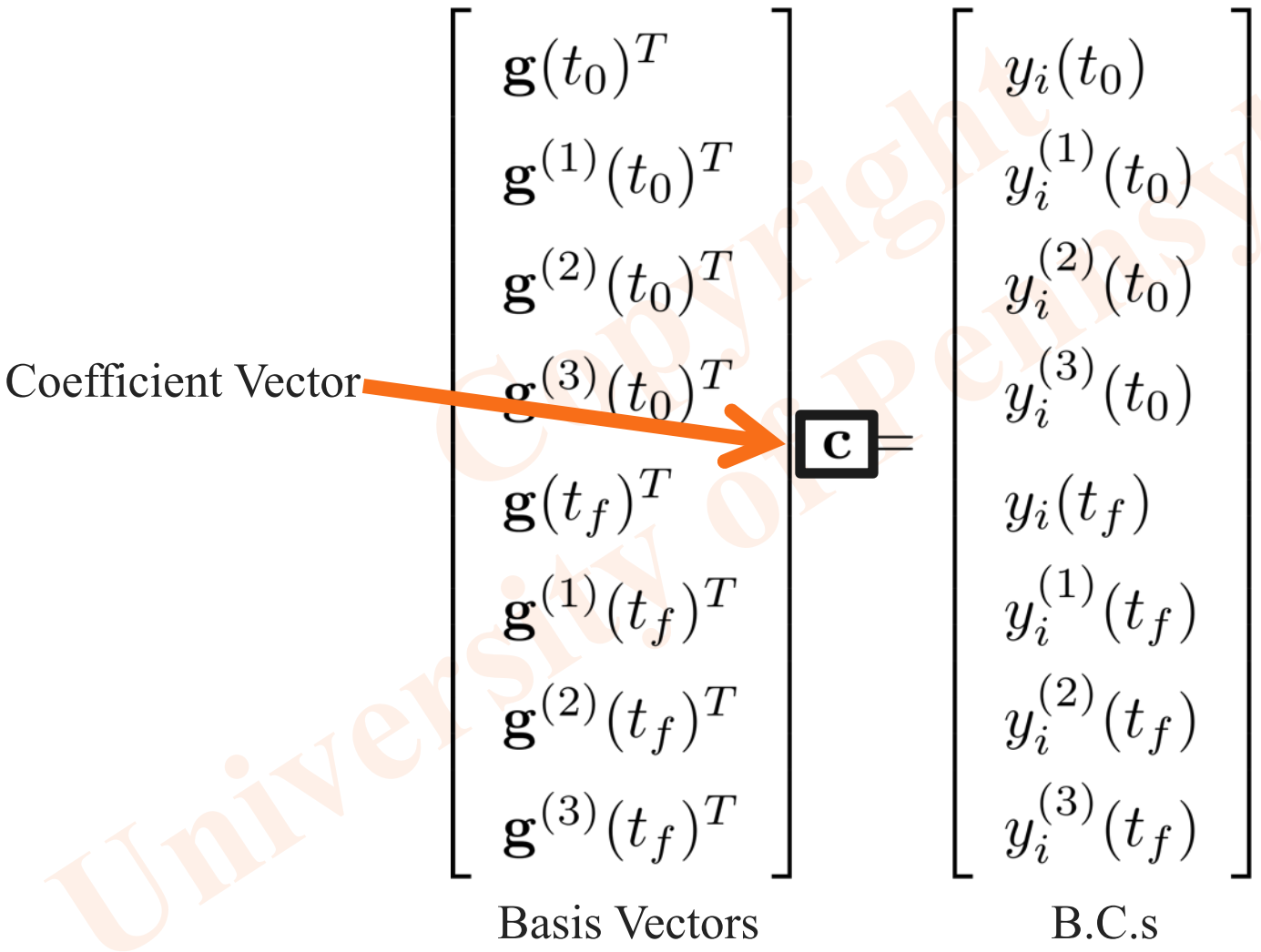
Euler Lagrange Equation

$$\frac{\partial \mathcal{L}}{\partial f} + \sum_{i=1}^{n} (-1)^i \frac{d^i}{dt^i} \frac{\partial \mathcal{L}}{\partial f^{(i)}} = 0$$

$$f = y_i \quad \text{and} \quad \mathcal{L} = \left(y_i^{(4)}\right)^2 \xrightarrow{i = 1, 2, 3} y_i^{(8)} = 0$$

$$f = y_4 \quad \text{and} \quad \mathcal{L} = \left(y_4^{(2)}\right)^2 \longrightarrow y_4^{(4)} = 0$$

Euler Lagrange Solution



$t_0 \neq t_f$

Quadratic Programming: Algebraic Solution

Increase Flexibility

- Intermediate conditions

Constraints: $H\mathbf{c} = \mathbf{d}$

$$J = \int_{t_0}^{t_f} \left\| y_i^{(4)}(t) \right\|^2 dt$$

- Algebraic Solution

Define the cost functional using the basis vectors:

$$J = \int_{t_0}^{t_f} \left\| y_i^{(4)}(t) \right\|^2 dt = \mathbf{c}^T \left[\int_{t_0}^{t_f} \left[\frac{d^4\mathbf{g}}{dt^4} \frac{d^4\mathbf{g}^T}{dt^4} \right] dt \right] \mathbf{c}$$

$$G = \int_{t_0}^{t_1} \frac{d^4\mathbf{g}}{dt^4} \frac{d^4\mathbf{g}^T}{dt^4} dt$$

$$\begin{bmatrix} 2G & H^T \\ H & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{d} \end{bmatrix}$$

Quadratic Programming: Numerical Solution

Constraints: $H\mathbf{c} = \mathbf{d} \quad M\mathbf{c} \leq \mathbf{p}$

- Inequality constraints allow us to constrain positions, velocities, control inputs, etc.

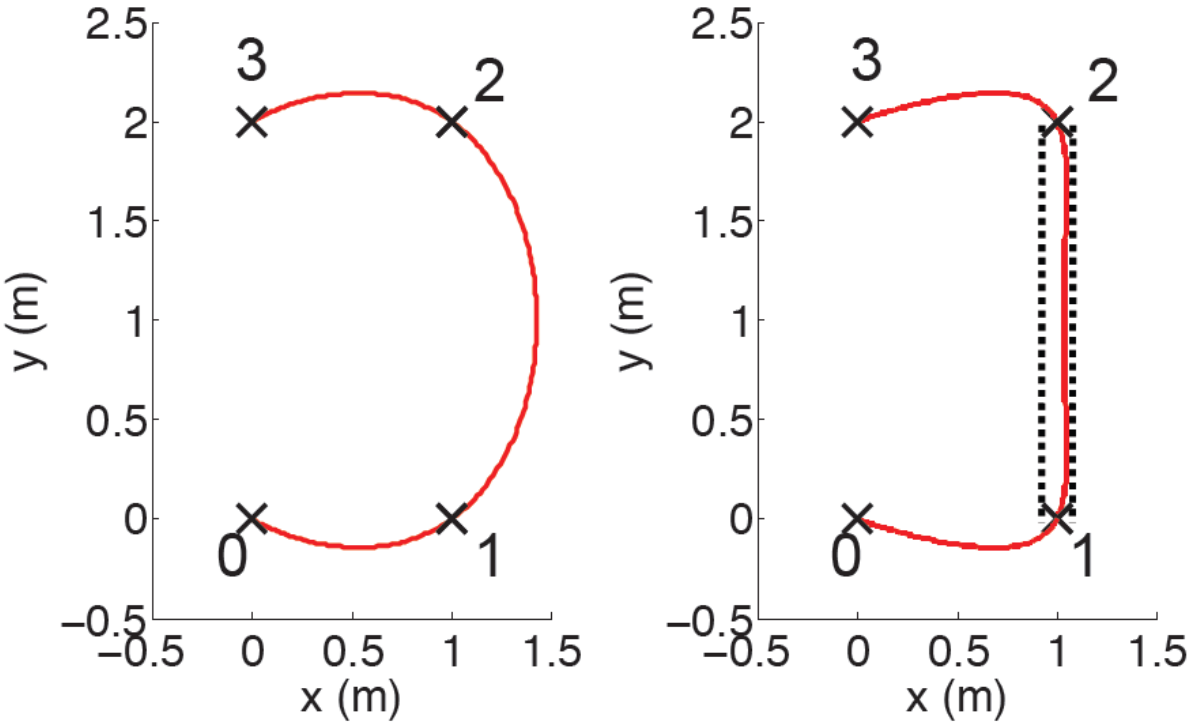
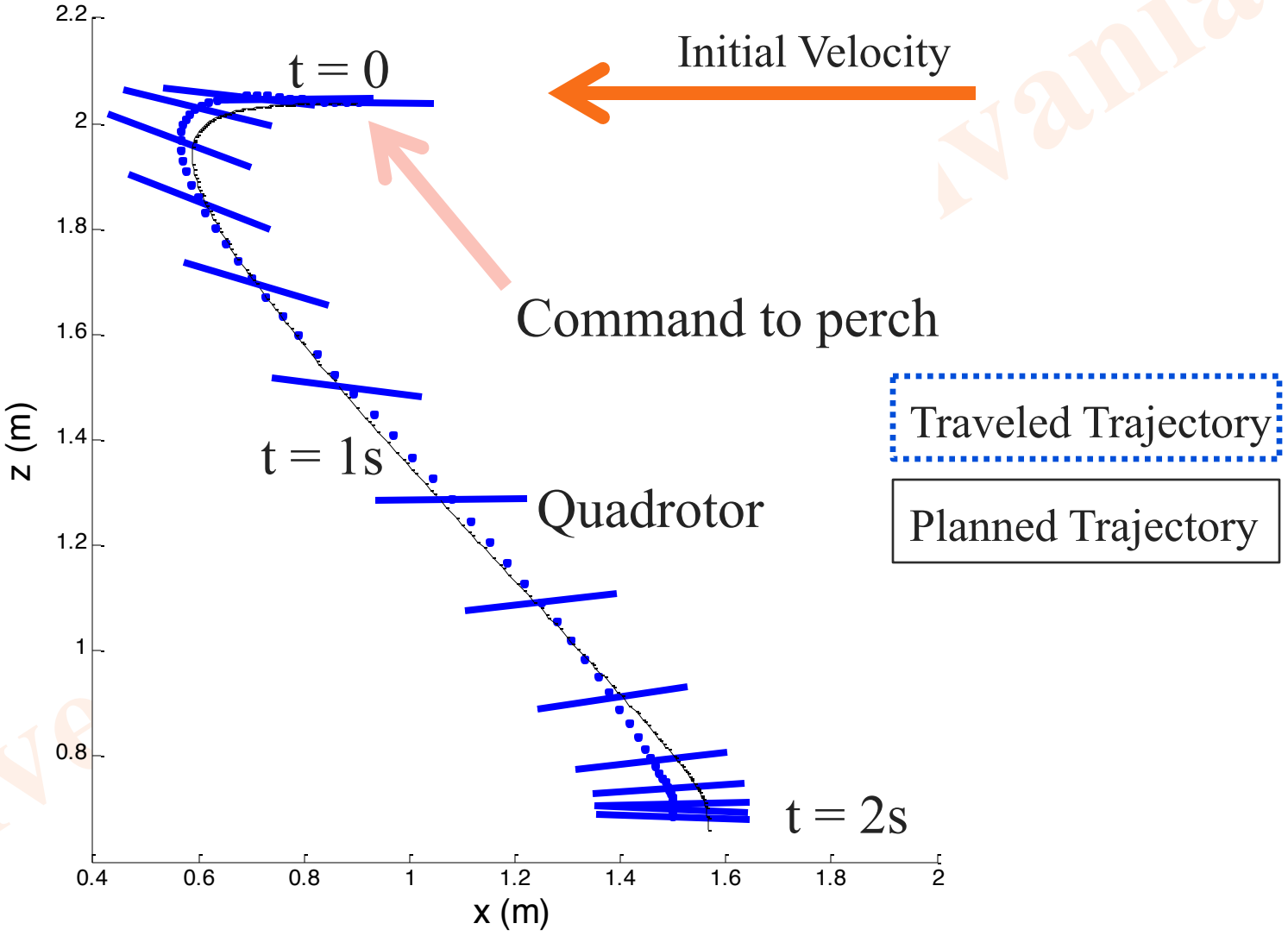
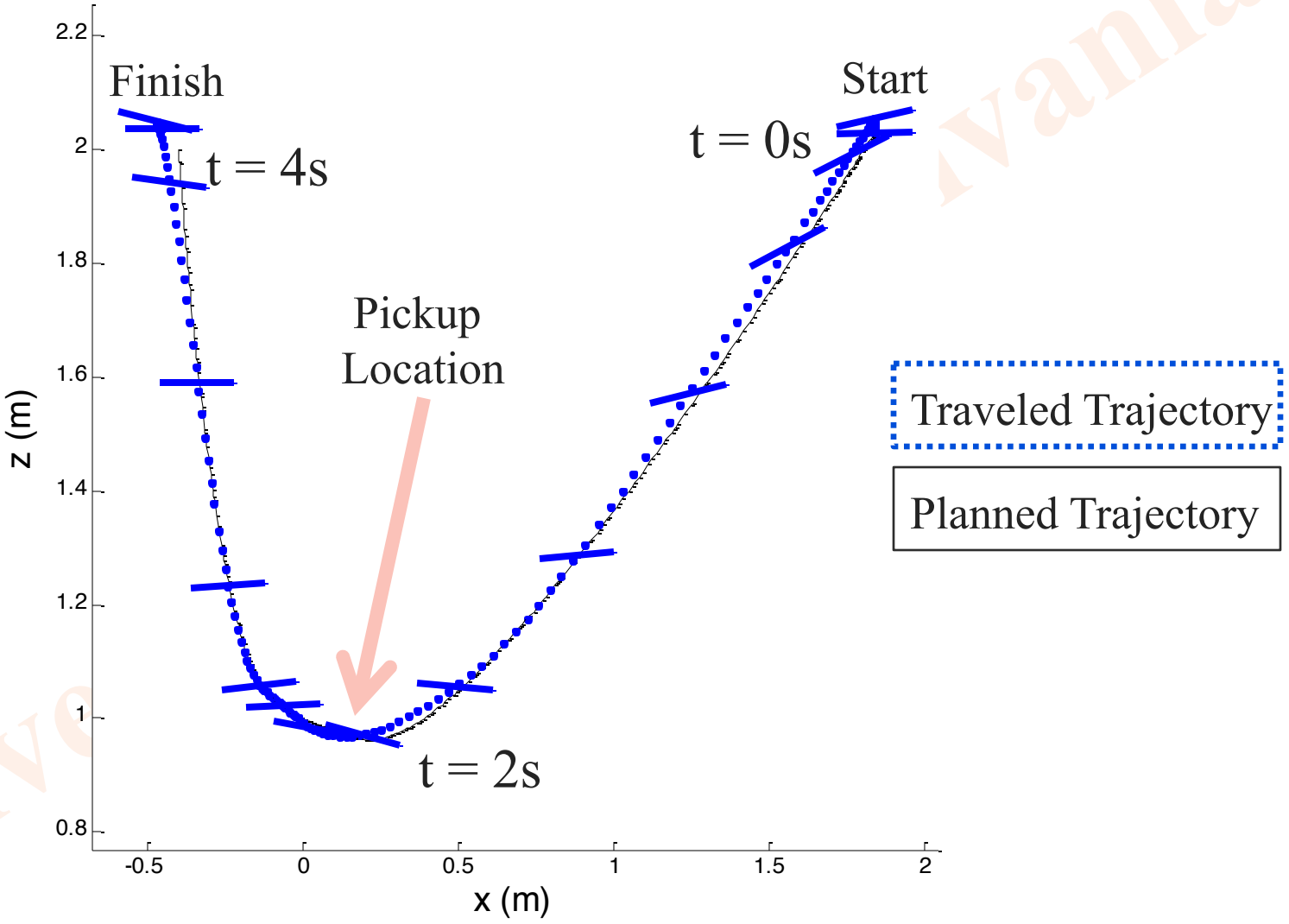


Image from [Mellinger, Kumar 2011]

Perching (using a real quadrotor)



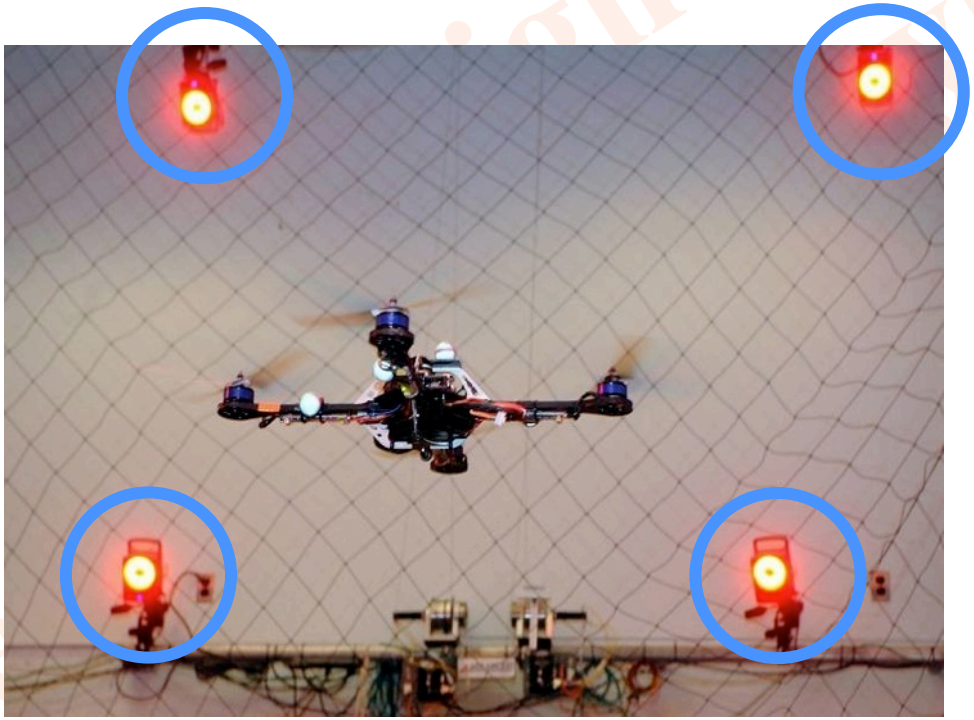
Grasping (using a real quadrotor)



Dynamic High Speed Grasping

Quadrotors in the Real World

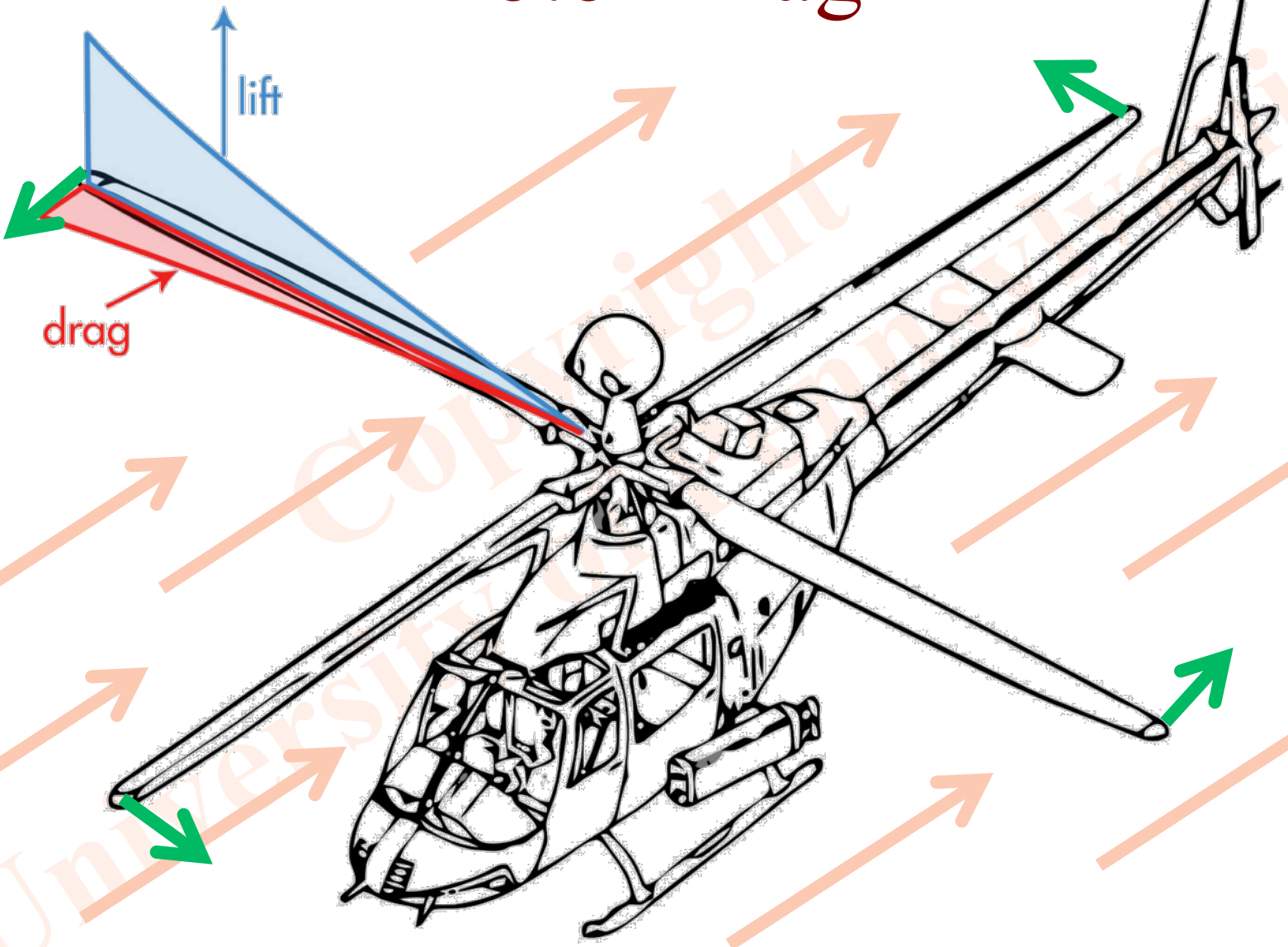
This is not the real world…



Many Challanges

- Onboard Sensing & State Estimation
- External Disturbances (wind, ground/ ceiling effects)
- Obstacle Avoidance
- Interaction with the environment
- Limited flight time

Rotor Drag

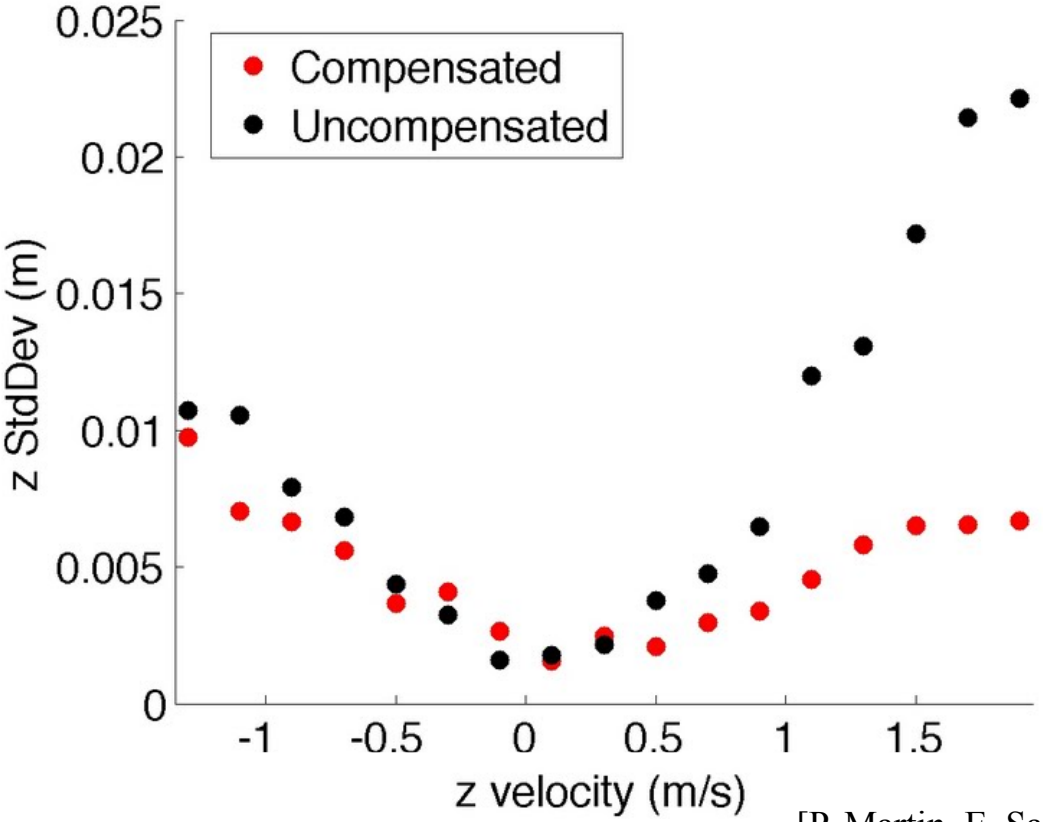


cesarvandevelde-constructieleer.blogspot.com/2011/01/forces-acting-on-helicpoter.html

Aerodynamics

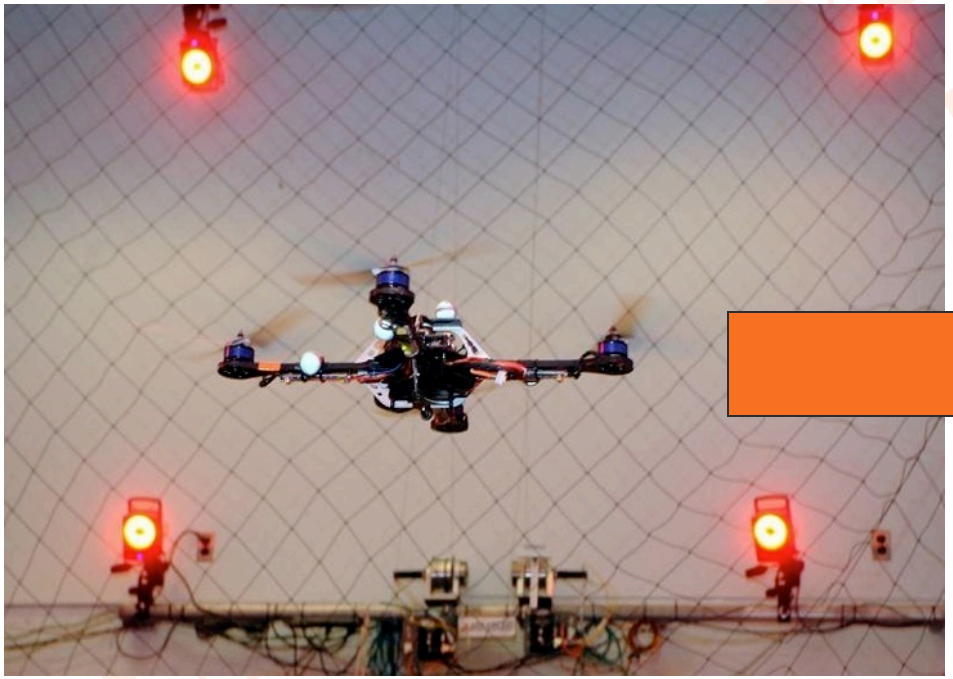
In practice, consideration of rotor drag is insignificant for accelerometer feedback

Free Stream Velocity



[P. Martin, E. Salaun, 2010],[Powers et al. 2012]

Sensors – Part II



Many types of sensors

- Inertial Measurement Unit (IMU)
- Motion Capture System

- Ultrasound
- Monocular Vision
- Stereo vision
- Laser Scanner
- RGBD
- GPS
- Other

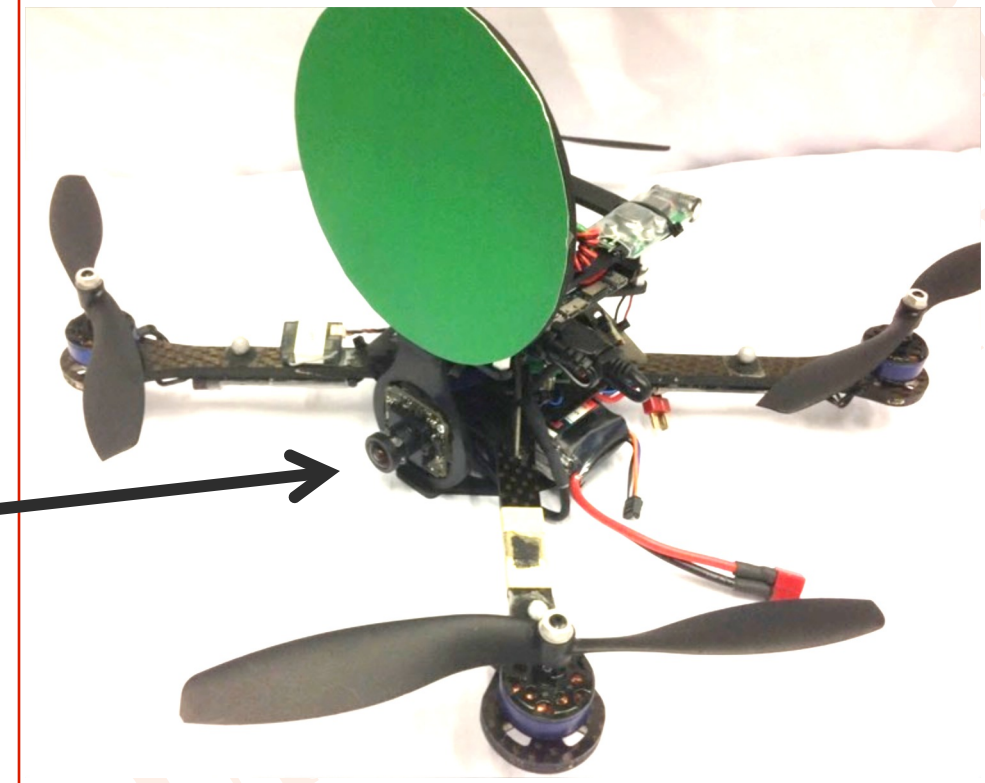
Ultrasound

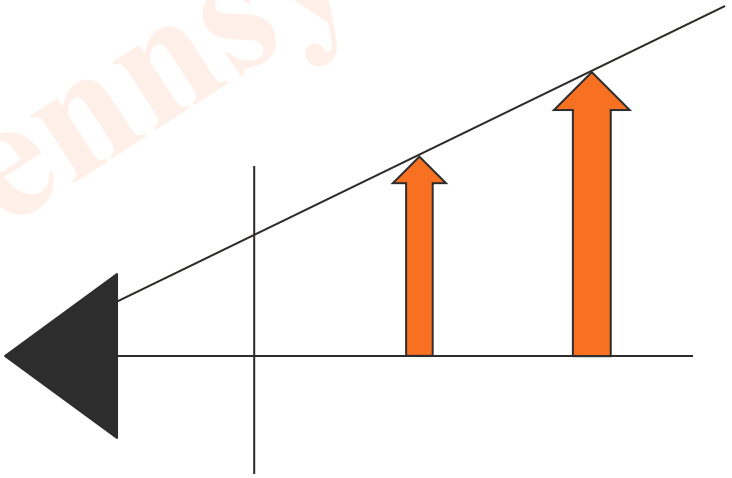
- Emits a sound
- Time until reflection is measured
- Used to determine distance to target

- Used mainly to estimate height

Monocular Vision

- Single camera
  - Cannot directly measure scale





Visual Servoing

Position Based

- Pose of the robot is estimated in the inertial frame
- Controllers use the pose as feedback

\+ Typical filters and estimators can be used

\- Vulnerable to calibration errors

Image Based

- Robot pose is not explicitly estimated
- Controllers use image features as feedback

\- Filters must be mapped to image features

\+ Desired pose can be set and achieved despite rough calibrations

\- Trajectory planning in the image feature space is difficult

Position Based Visual Servoing (PBVS)

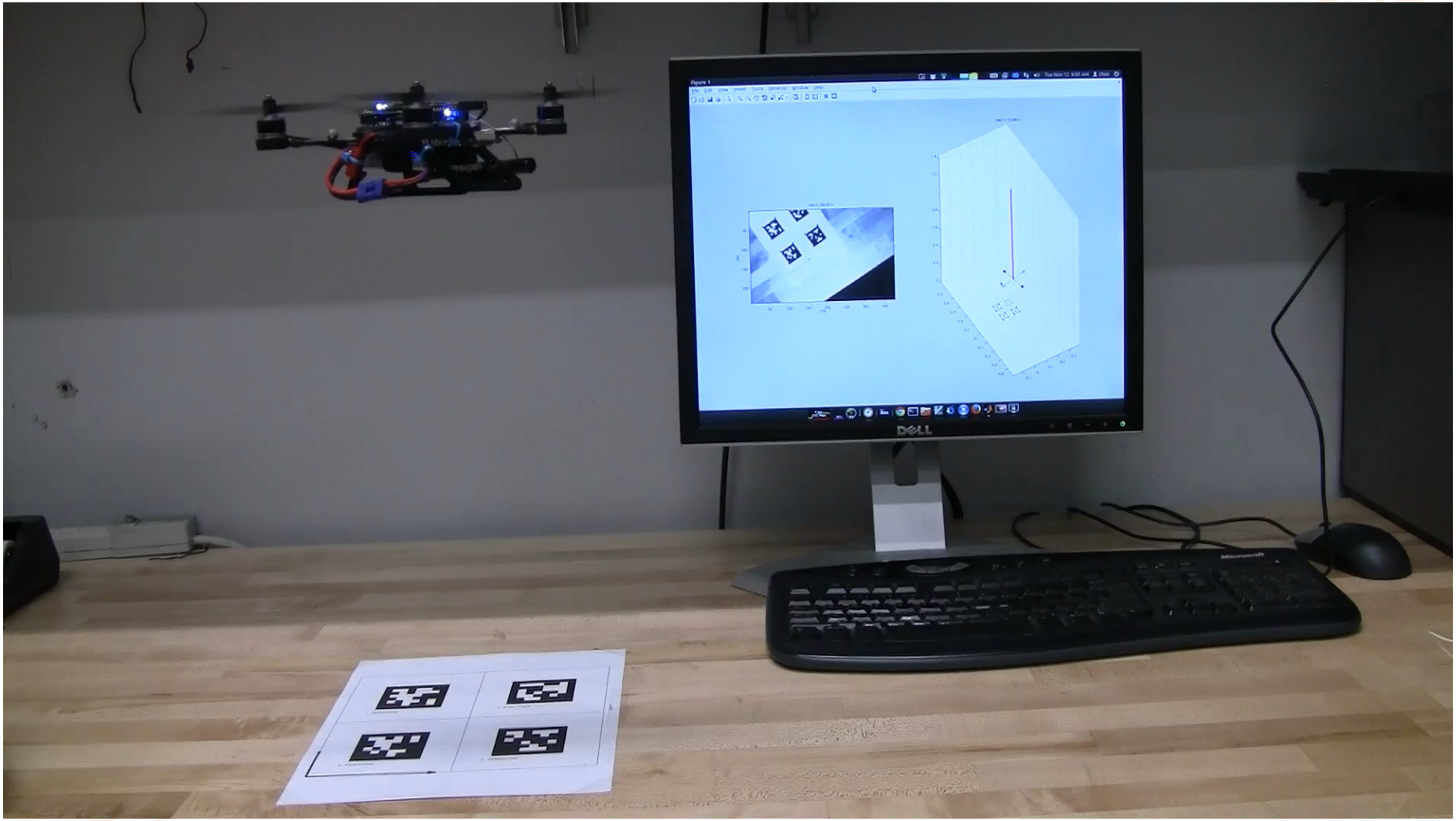
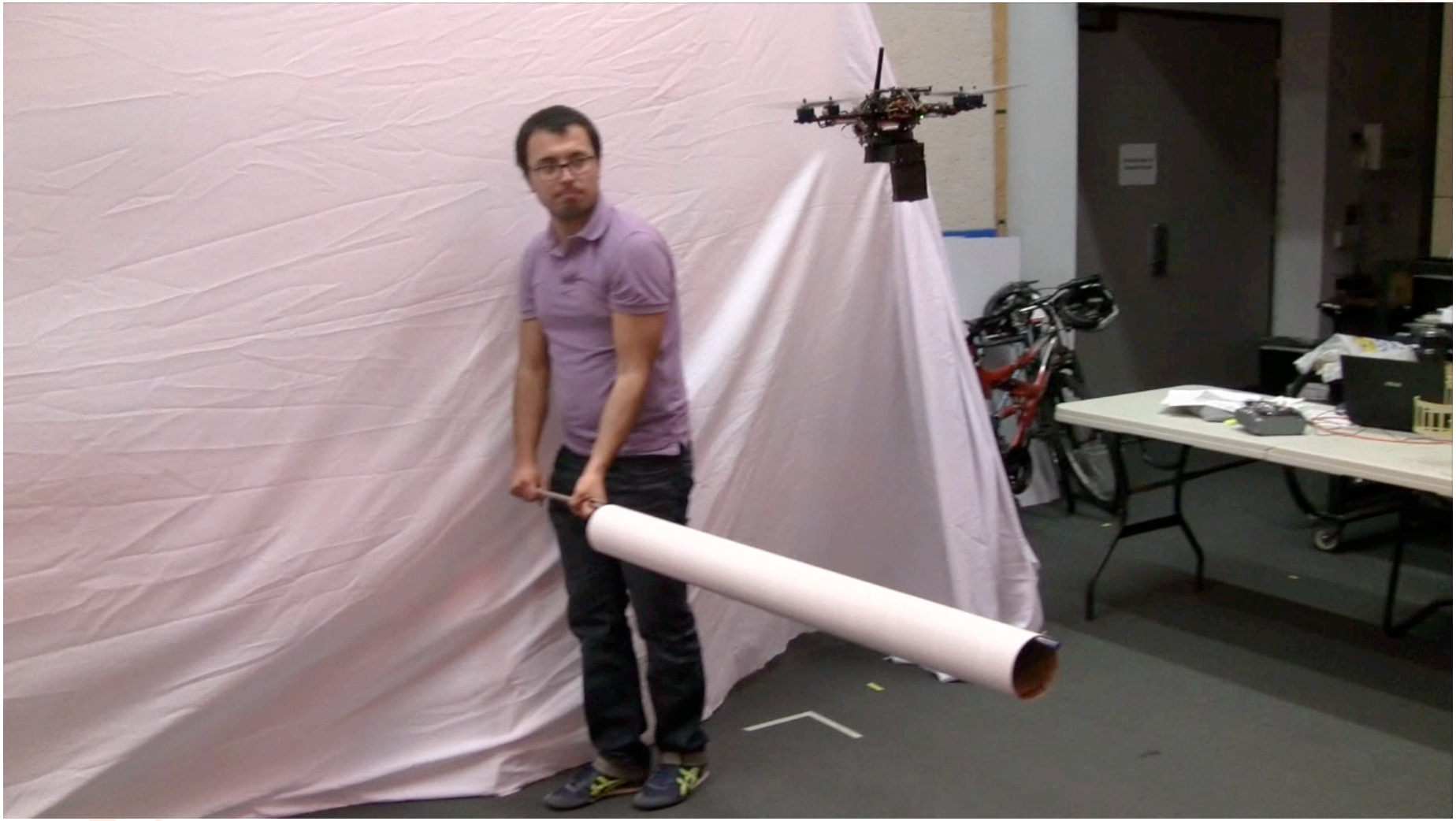
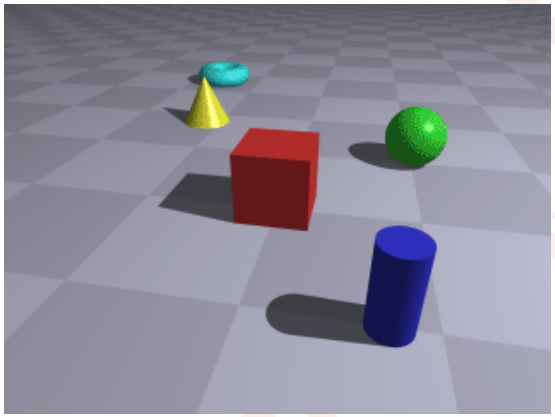


Image Based Visual Servoing (IBVS)

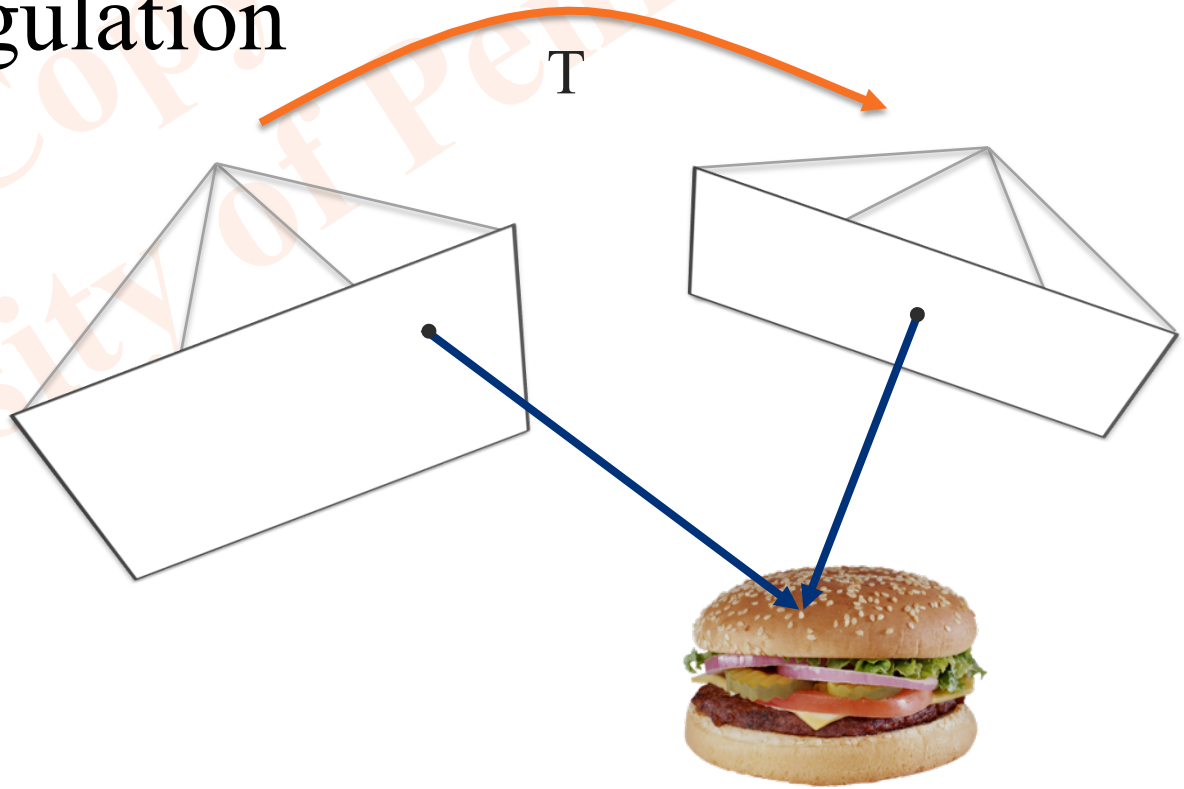


Stereo Vision

- Two or more cameras with a known transformation between frames
- Depth of an image feature determined using triangulation

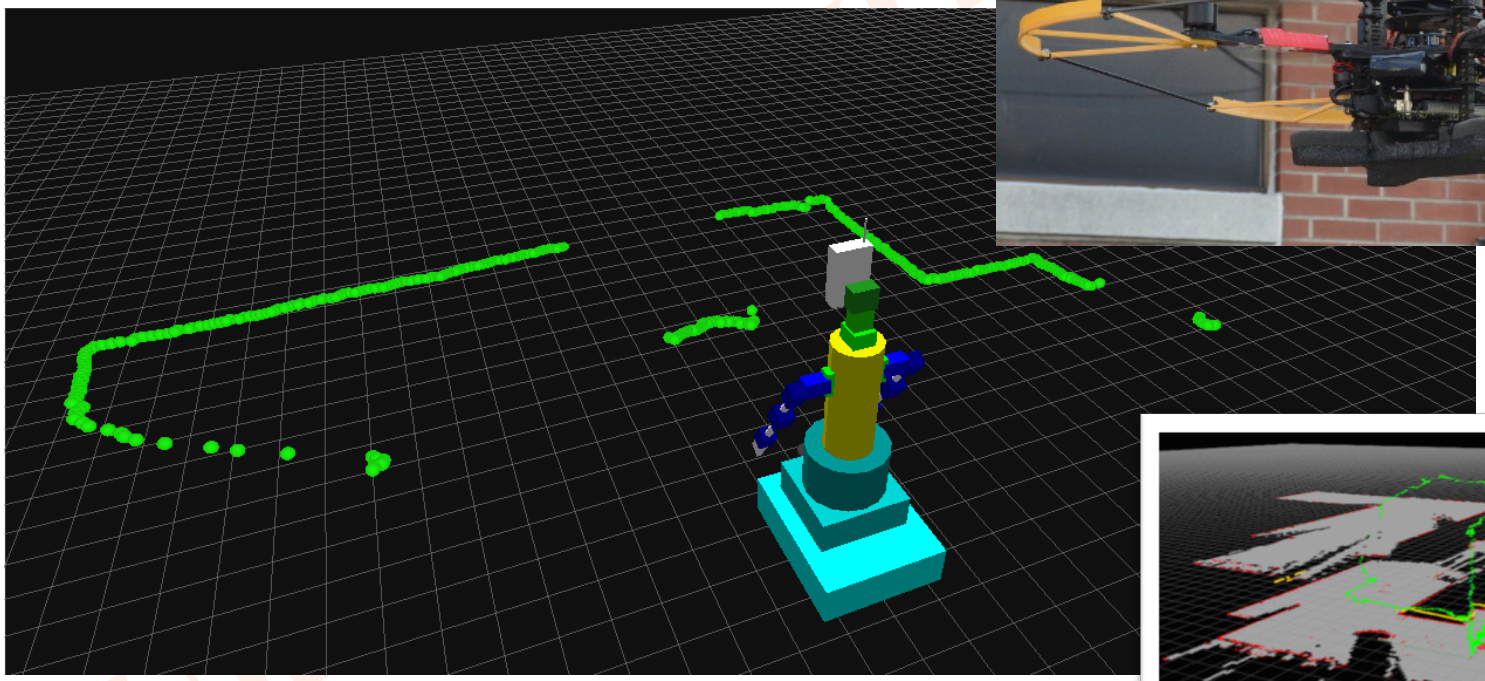


http://en.wikipedia.org/wiki/Wiggle_stereoscopy#mediaviewer/File:Stereo_wiggle_3D.gif

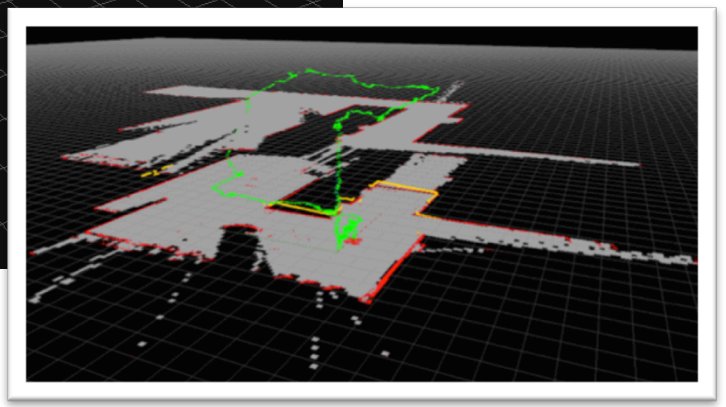


Laser

- Provides a 2D scan of distances to obstacles

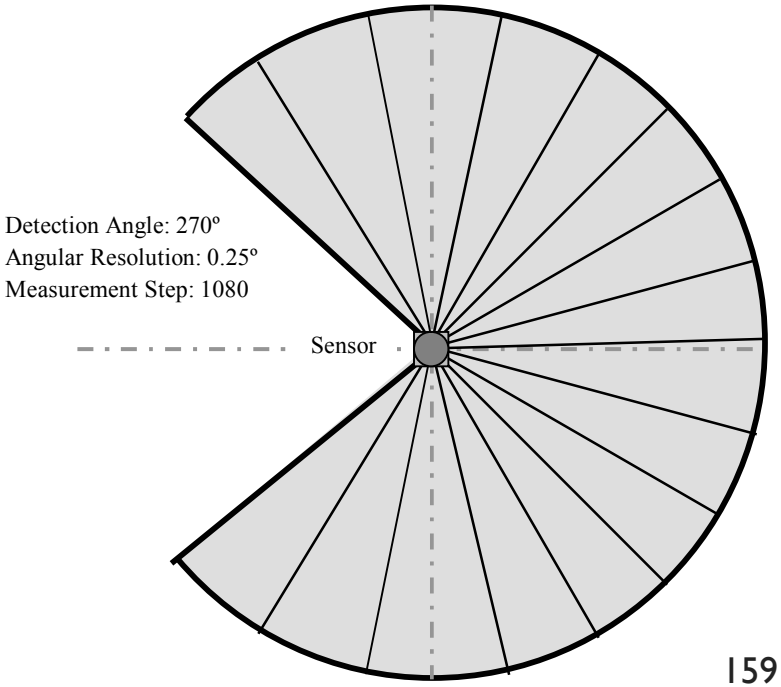


http://www.pirobot.org/blog/0015/laser-1.png

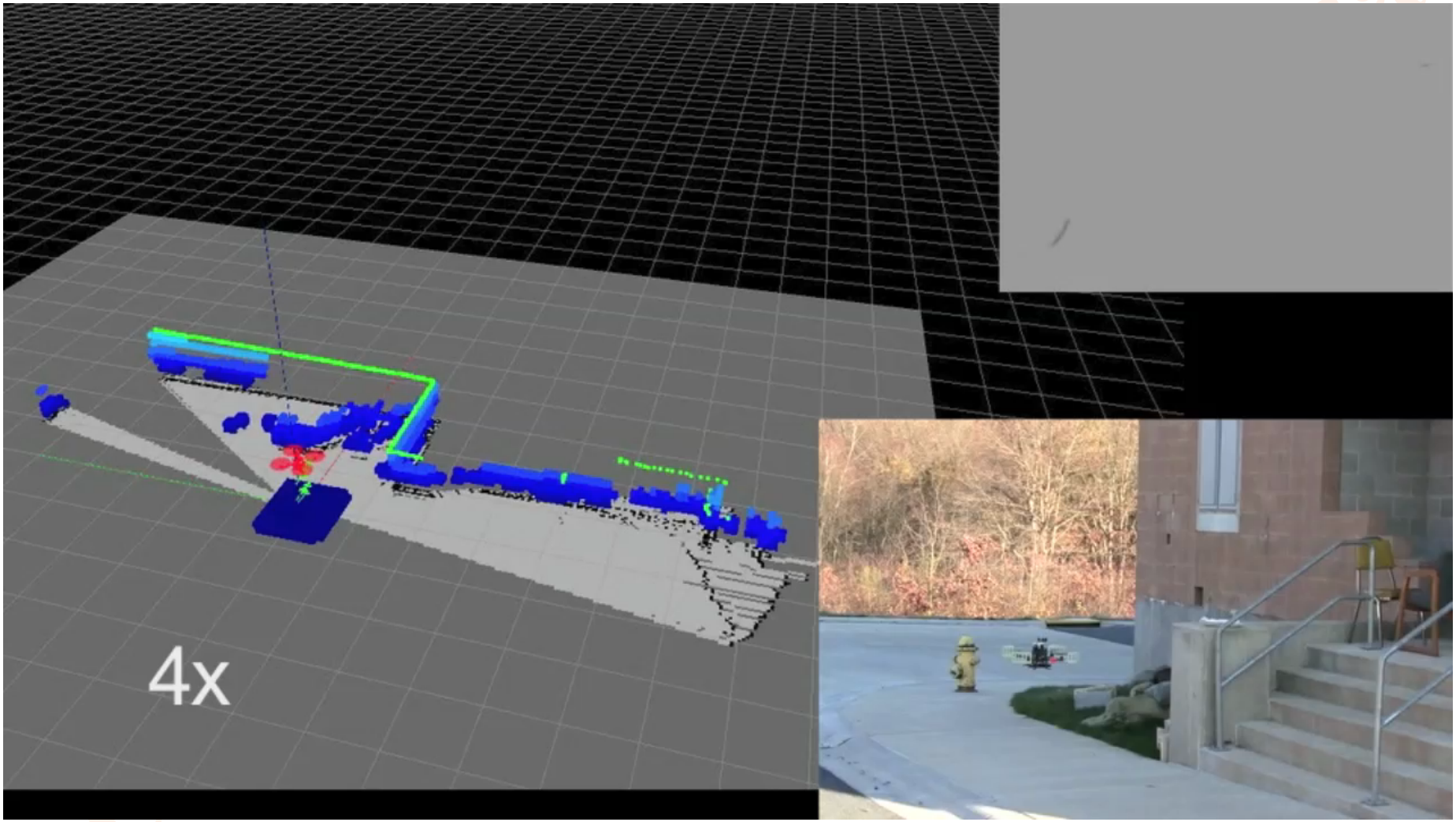


Laser

Hokuyo UTM-30LX
Scanning Laser
Rangefinder

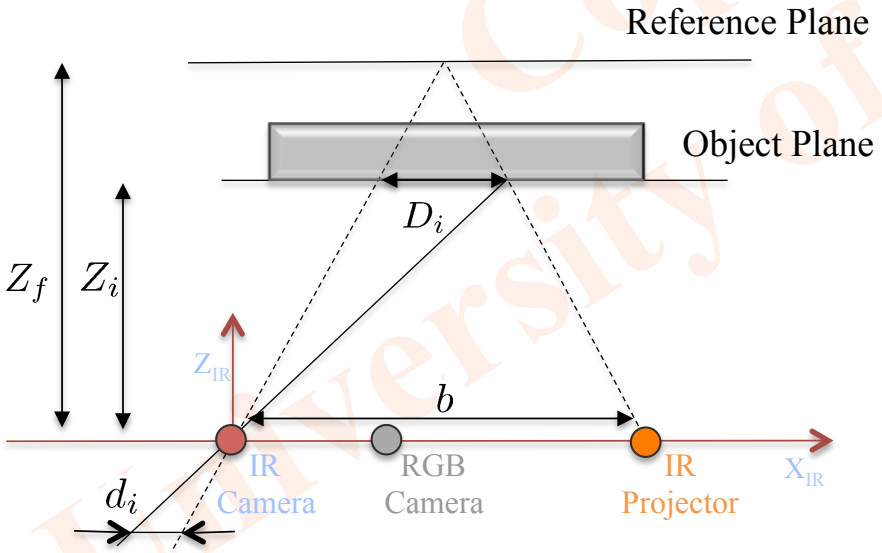


Laser



RGBD

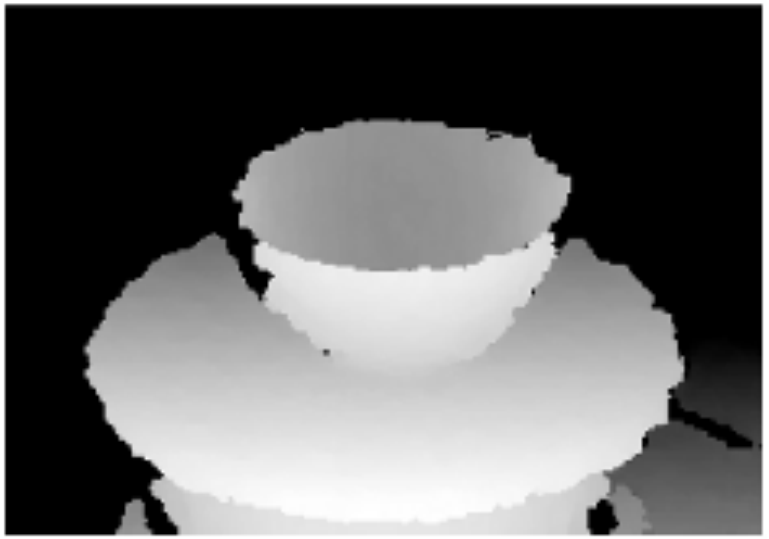
RGB Camera + IR Projector + IR Camera = RGB Image with Depth



RGBD

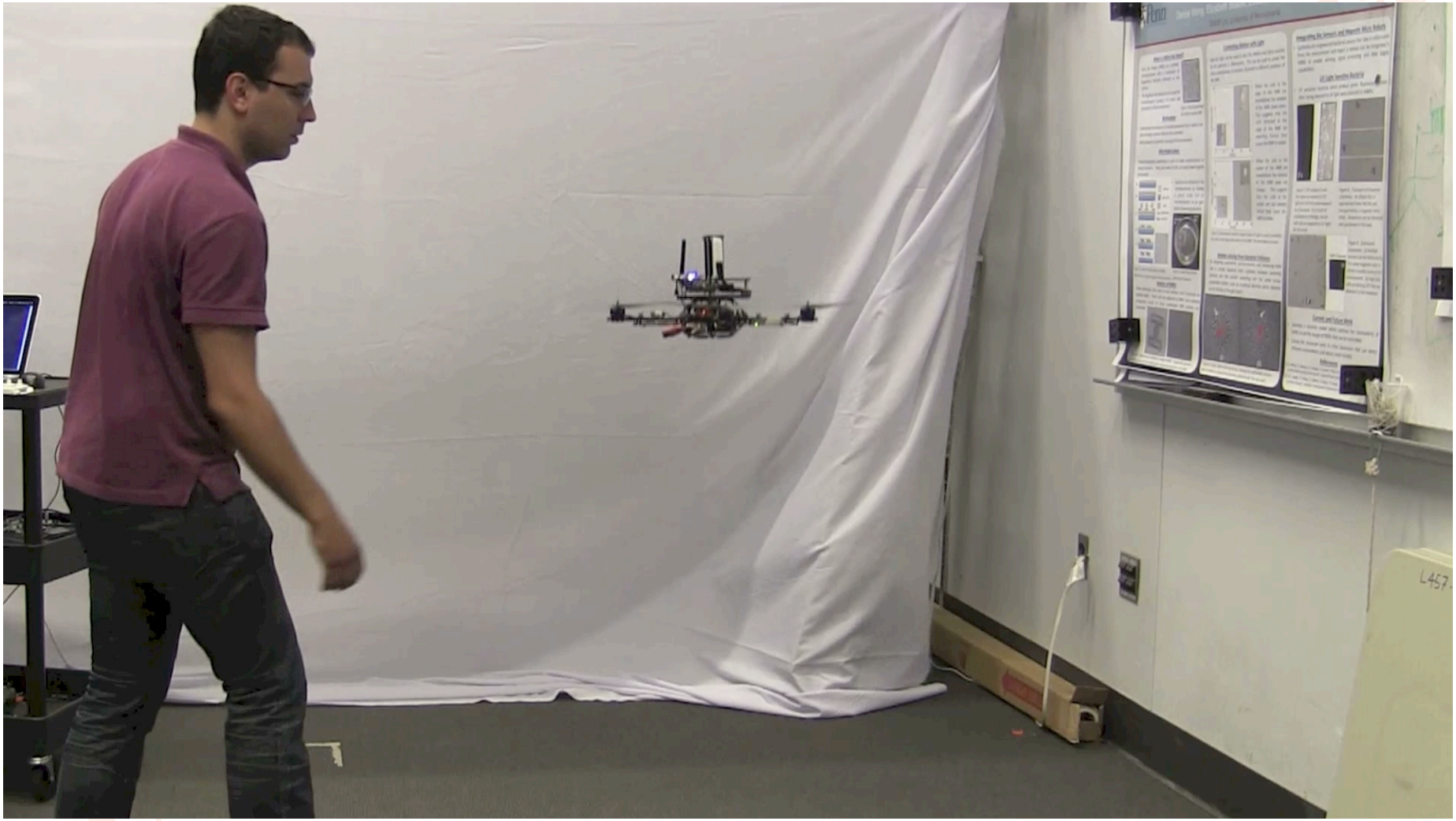
RGB Image

Depth Map



http://rgbd-dataset.cs.washington.edu

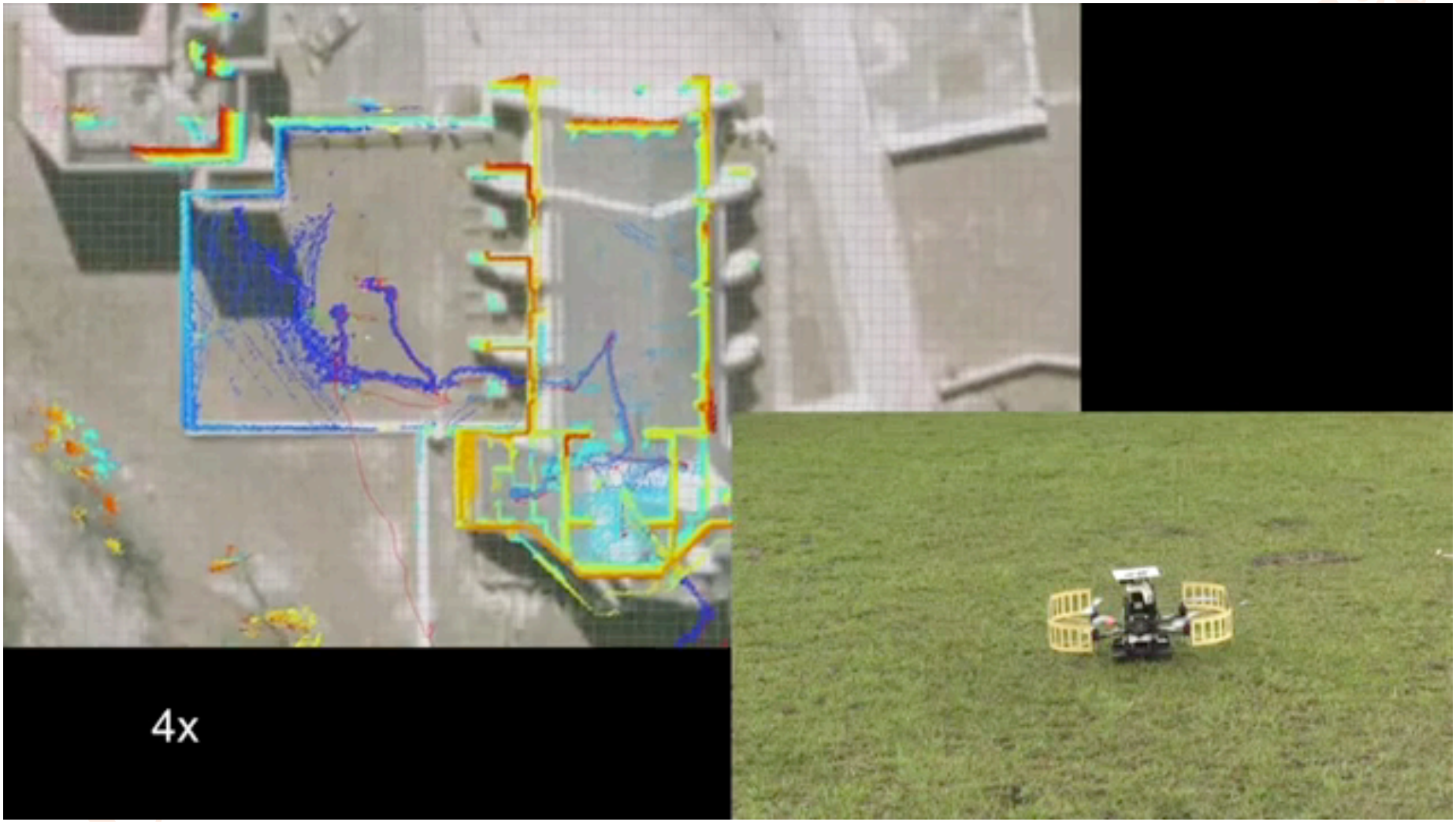
RGBD



GPS

- Problem:
  - Affected by cities, weather, etc…
  - Not accurate enough for flying in tight formations or close proximity to obstacles

Sensor Fusion Video



Central Question

How to plan paths/trajectories/motion in an known environment?

The motion planning problem

The Basic Problem

- Euclidean world, R^N, N=2 or 3
- Obstacles O_1, O_2, … O_p, all closet subsets of R^N
- Rigid Body (robot) A

Reference

- Planning algorithms, Lavalle (Section 4.3)

Rigid Body *A*

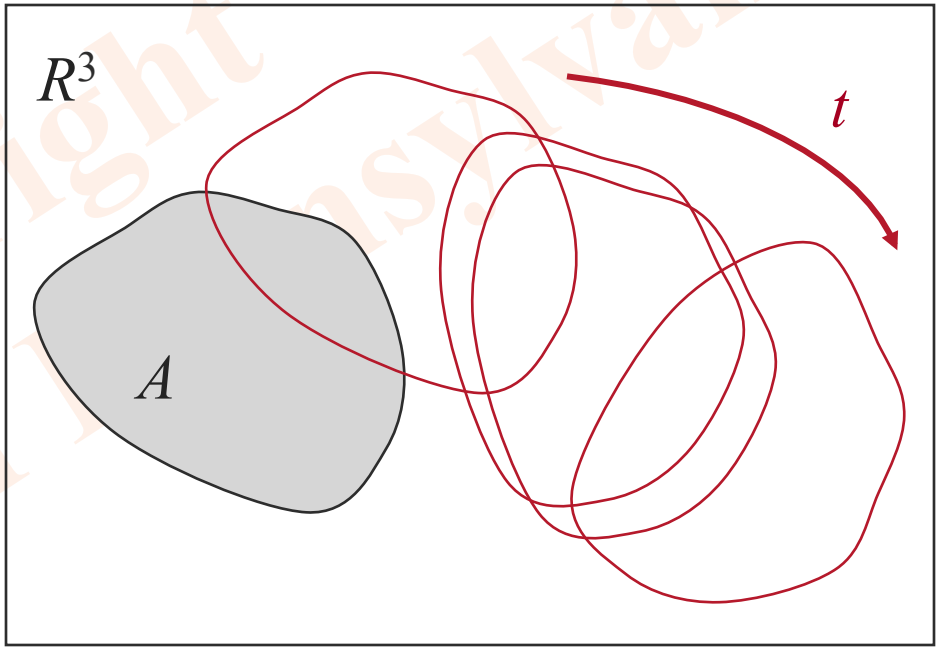
Body

$$A \subset R^3$$

Rigid Body Displacement

Map

$$g : A \rightarrow R^3$$



Rigid Body Motion

Continuous family of maps

$$g(t) : A \rightarrow R^3$$

Each displacement is a new pose (position + orientation)

The Basic Problem

- Euclidean world, R^N
- Obstacles O_1, O_2, … O_p, all closet subsets of R^N
- Robot (Rigid Body) A
- Given initial and final position/orientation (pose) of A, find a continuous (and legal) sequence of poses

Given

$$g(t_I) : A \rightarrow R^3$$

$$g(t_F) : A \rightarrow R^3$$

Find a safe, continuous family of maps

$$g(t) : A \rightarrow R^3$$

Configuration Space (C-space)

The motion planning problem is best formulated in configuration space (denote by C)

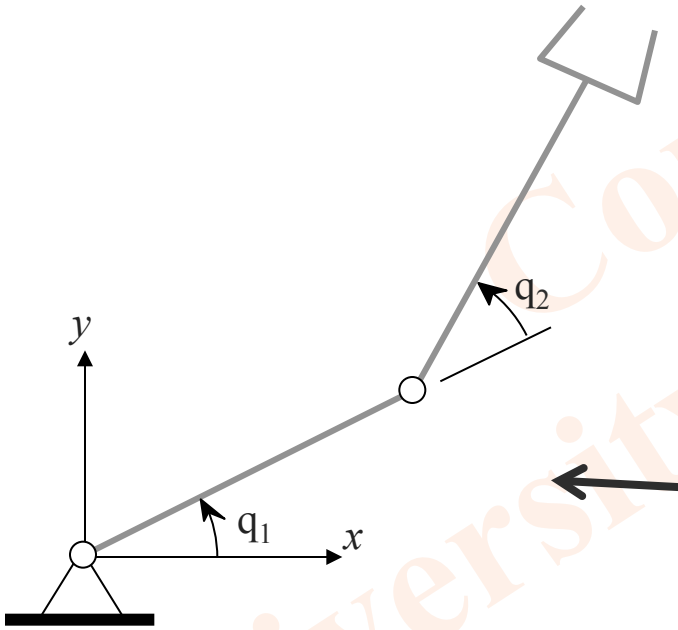
- $C \neq R^N$
- C is the set of all position/orientations

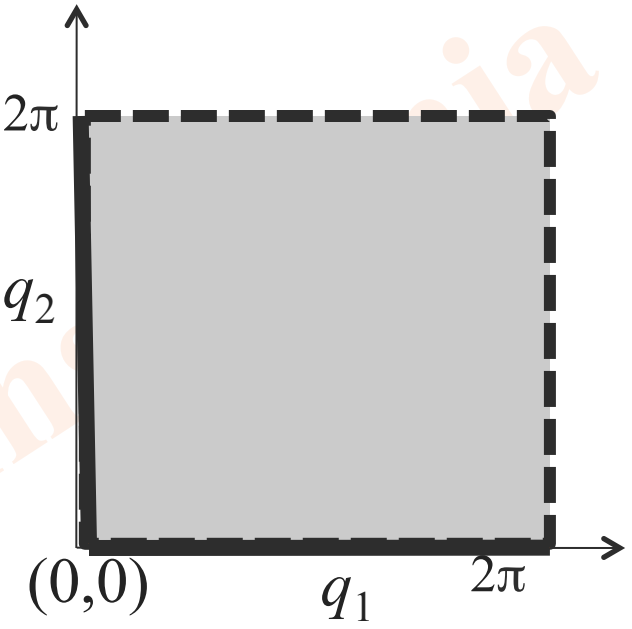
  - all possible maps $g : A \rightarrow R^3$

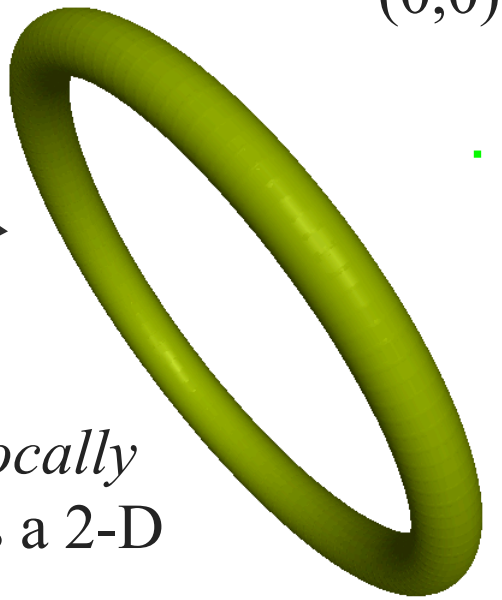
Examples of configuration space

- Point robot in N-dimensional space R^N Ans: R^N
- Rectangular robot in R^2 Ans: $SE(2)$
- Fixed robot arm with 2 revolute joints

C-space for a 2-R arm (2 rigid bodies)







C-space: $S^1 \times S^1$

The configuration space *locally* looks like R^2, and hence is a 2-D *manifold*

Configuration Space (C-space)

The motion planning problem is best formulated in configuration space (denote by C)

- $C \neq R^N$
- C is the set of all position/orientations

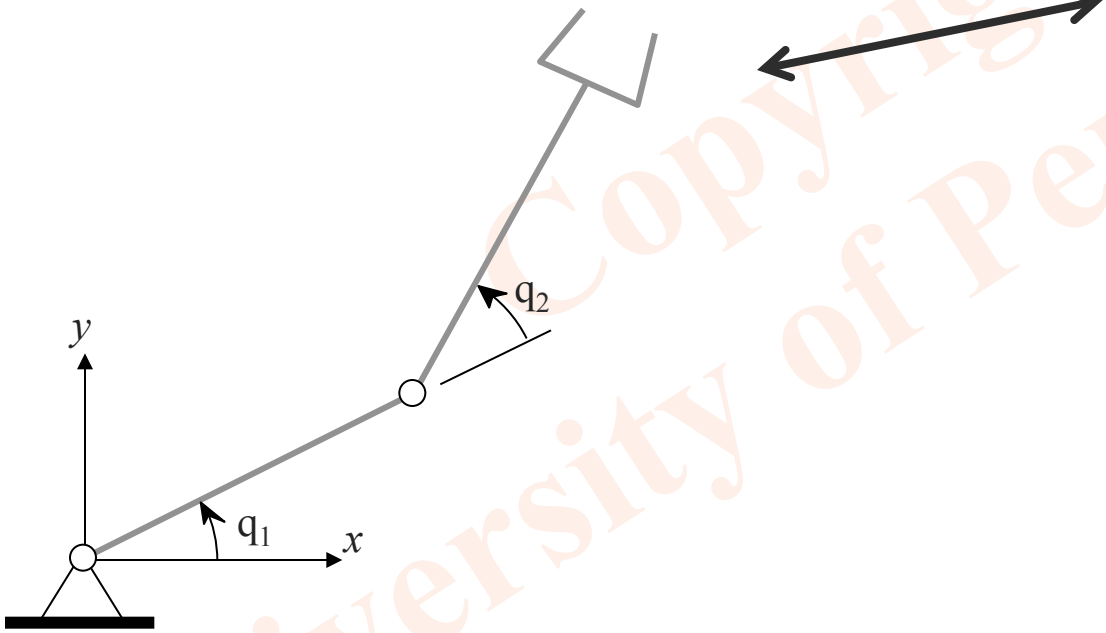
  - all possible maps $g : A \rightarrow R^3$

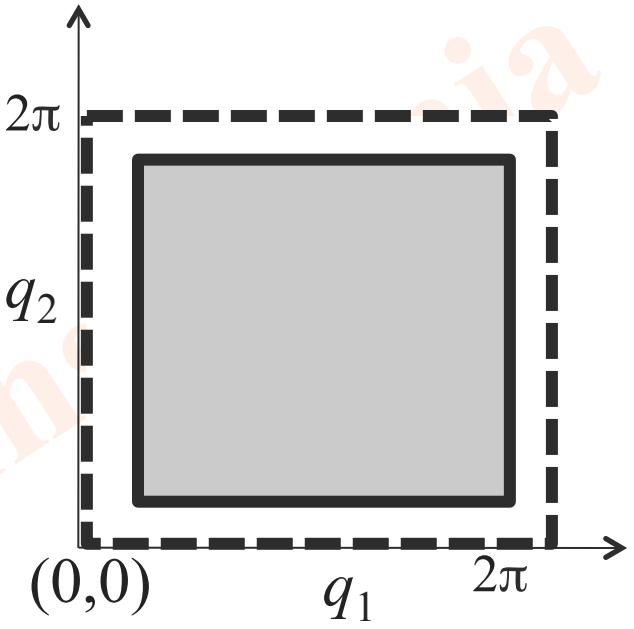
Examples of configuration space

- Point robot in N-dimensional space R^N Ans: R^N
- Rectangular robot in R^2 Ans: $SE(2)$
- Fixed robot arm with 2 revolute joints Ans: $S^1 \times S^1$

- Fixed robot arm with 2 revolute joints each with limits

C-space for a 2-R arm with two rigid bodies (2 rigid bodies)





C-space: *subset of* R^2

The configuration space *locally* looks like R^2, and hence is a 2-D *manifold*

Configuration Space (C-space)

The motion planning problem is best formulated in configuration space (denote by C)

- $C \neq R^N$
- C is the set of all position/orientations
  - all possible maps $g : A \rightarrow R^3$

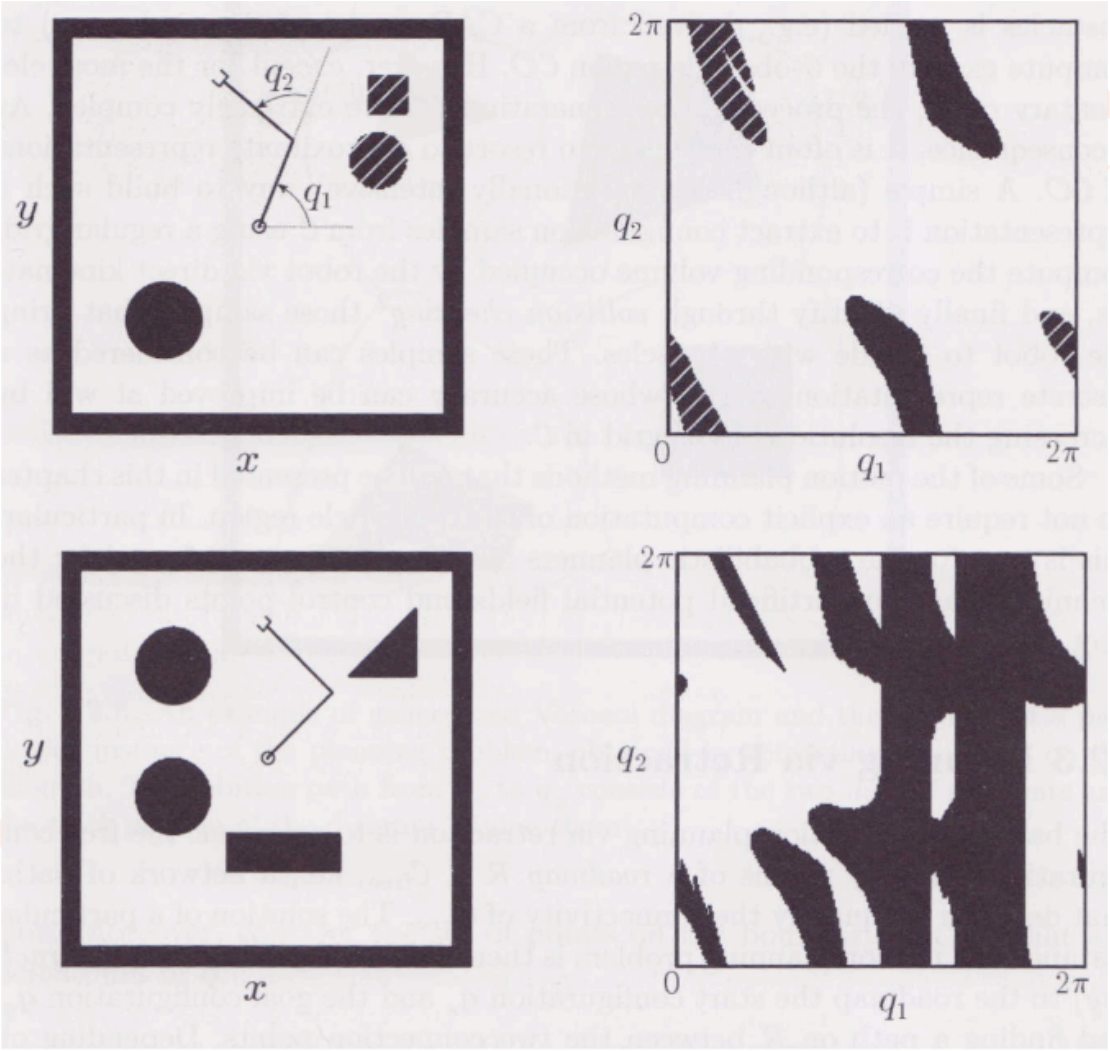
Examples of configuration space

- Point robot in N-dimensional space R^N Ans: R^N
- Rectangular robot in R^2 Ans: $SE(2)$
- Fixed robot arm with 2 revolute joints Ans: $S^1 \times S^1$

- Fixed robot arm with 2 revolute joints each with limits

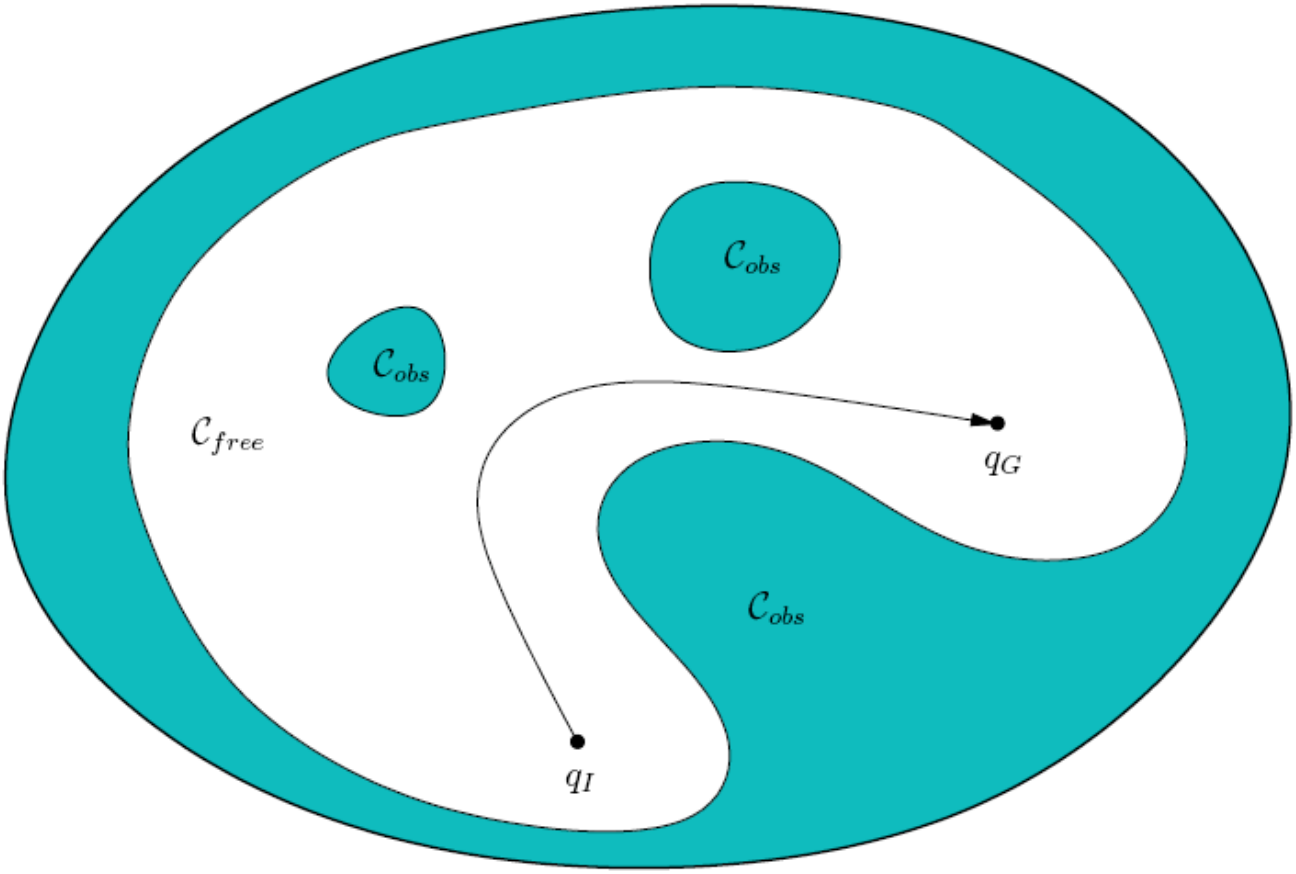
Fixed robot arm with 2 revolute joints but with obstacles

Is there a solution to the motion planning problem for any pair of initial and goal configurations?



[Siciliano, Sciavicco, Villani, Oriolo, 09]

The Basic Motion Planning Problem

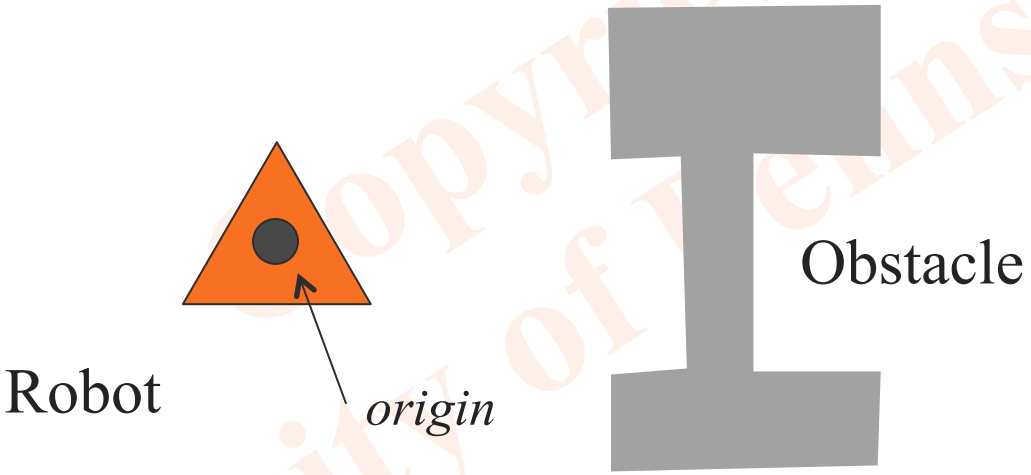


There exists a motion plan from q_I to q_G *iff* q_I and q_G belong to the same connected component of C_{free}

Lavalle, 4.3.1

Modeling Obstacle Regions and Free Space for a Robot with Finite Extent

Example: A single-rigid-body robot that can only translate in R^2 (configuration space is R^2)



The *obstacle region*, $\mathcal{C}_{obs} \subseteq \mathcal{C}$, is defined as

$$\mathcal{C}_{obs} = \{q \in \mathcal{C} \mid \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\}$$

Modeling Obstacle Regions and Free Space for a Robot with Finite Extent

Example: A single-rigid-body robot that can only translate in R^2

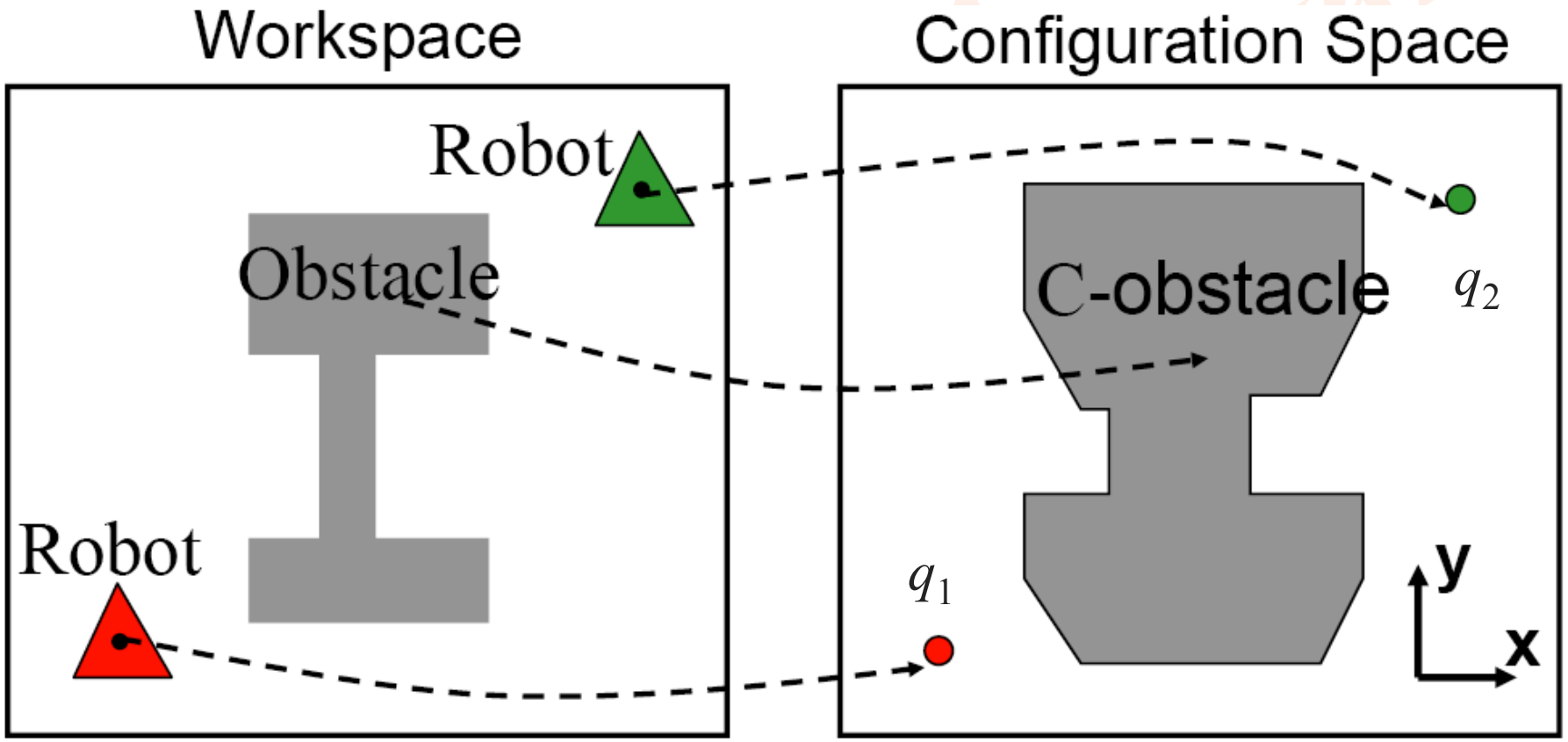
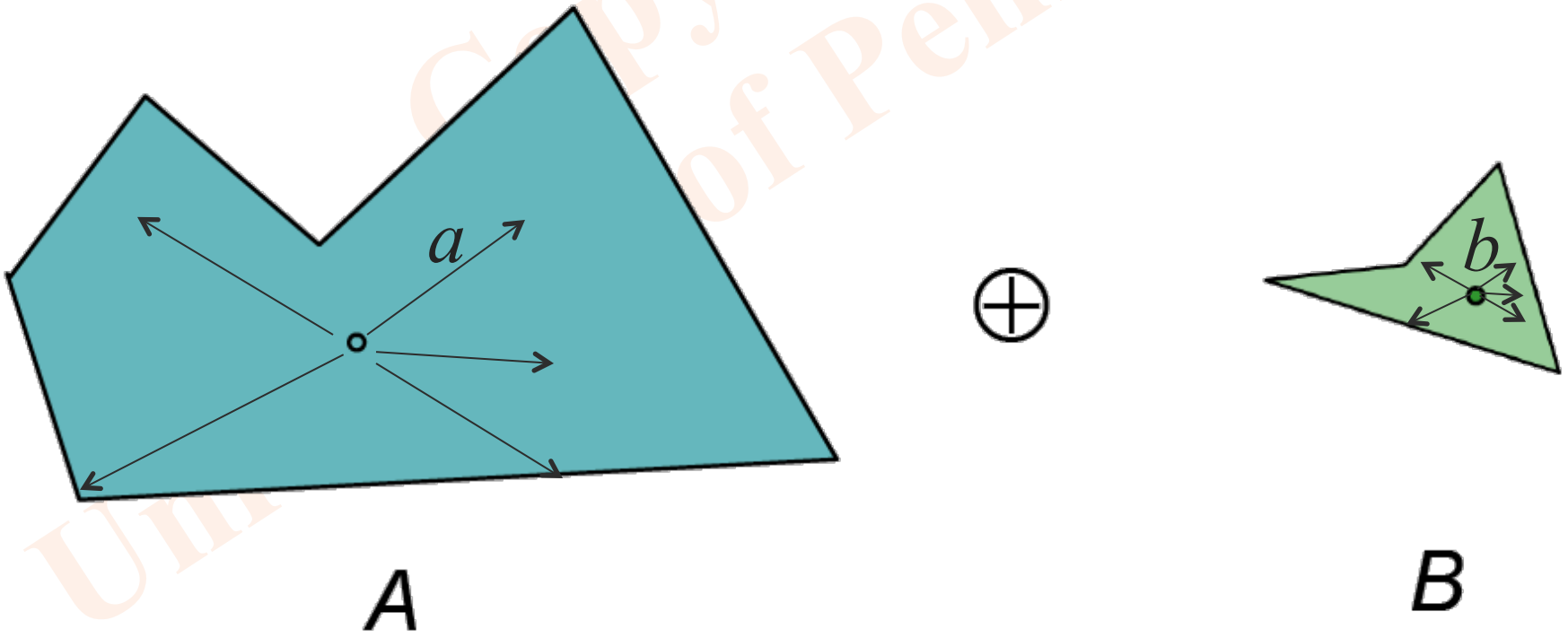


Figure by Dinesh Manocha, UNC

Key Idea: Minkowski Sum

The *Minkowski* sum of two sets A and B

$$A \oplus B = \{a + b \mid a \in A, b \in B\}$$



Penn Engineering

Figure by Dinesh Manocha, UNC

$$A \oplus B = \{a + b \mid a \in A, b \in B\}$$

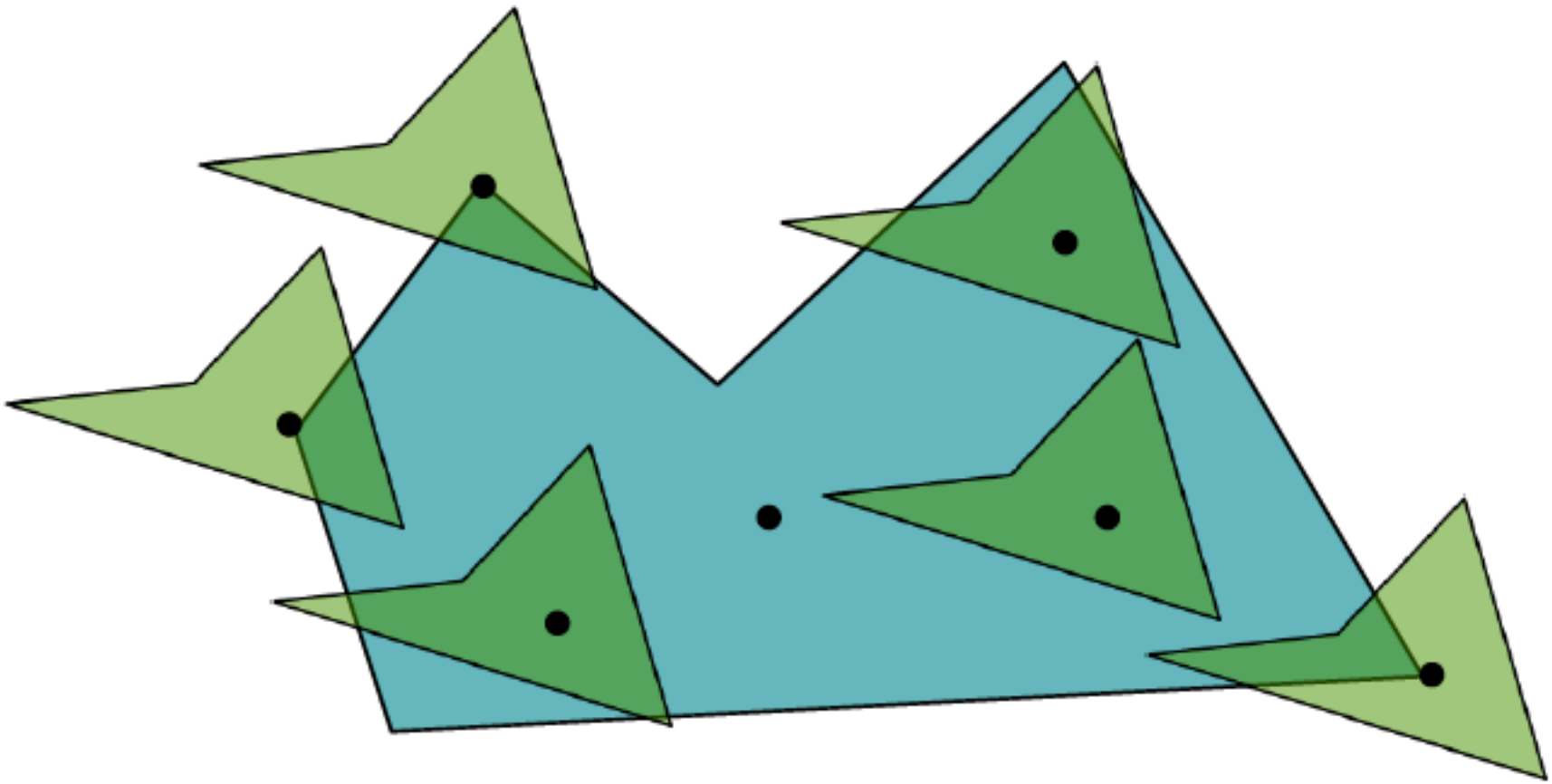


Figure by Dinesh Manocha, UNC

$$A \oplus B = \{a + b \mid a \in A, b \in B\}$$

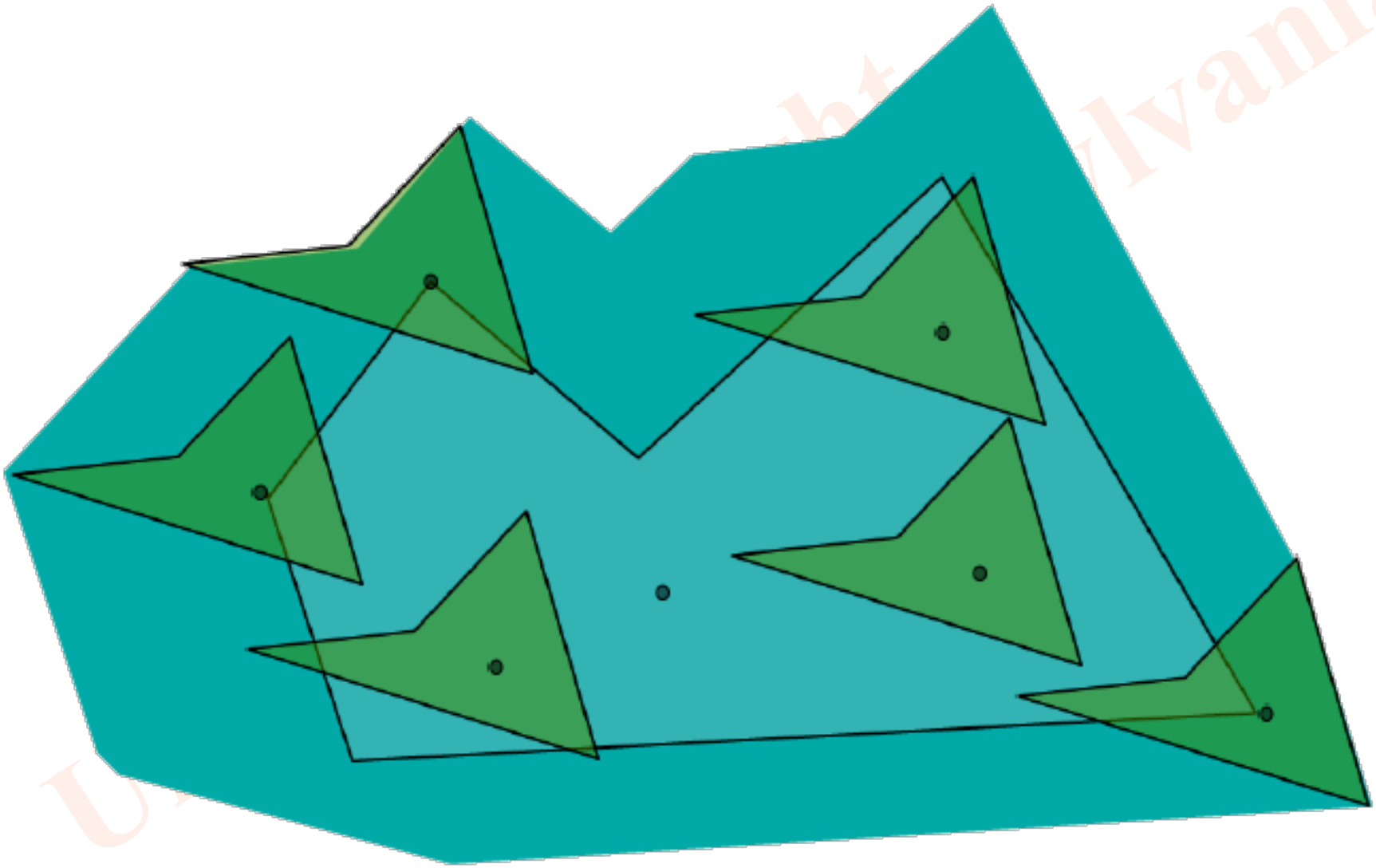


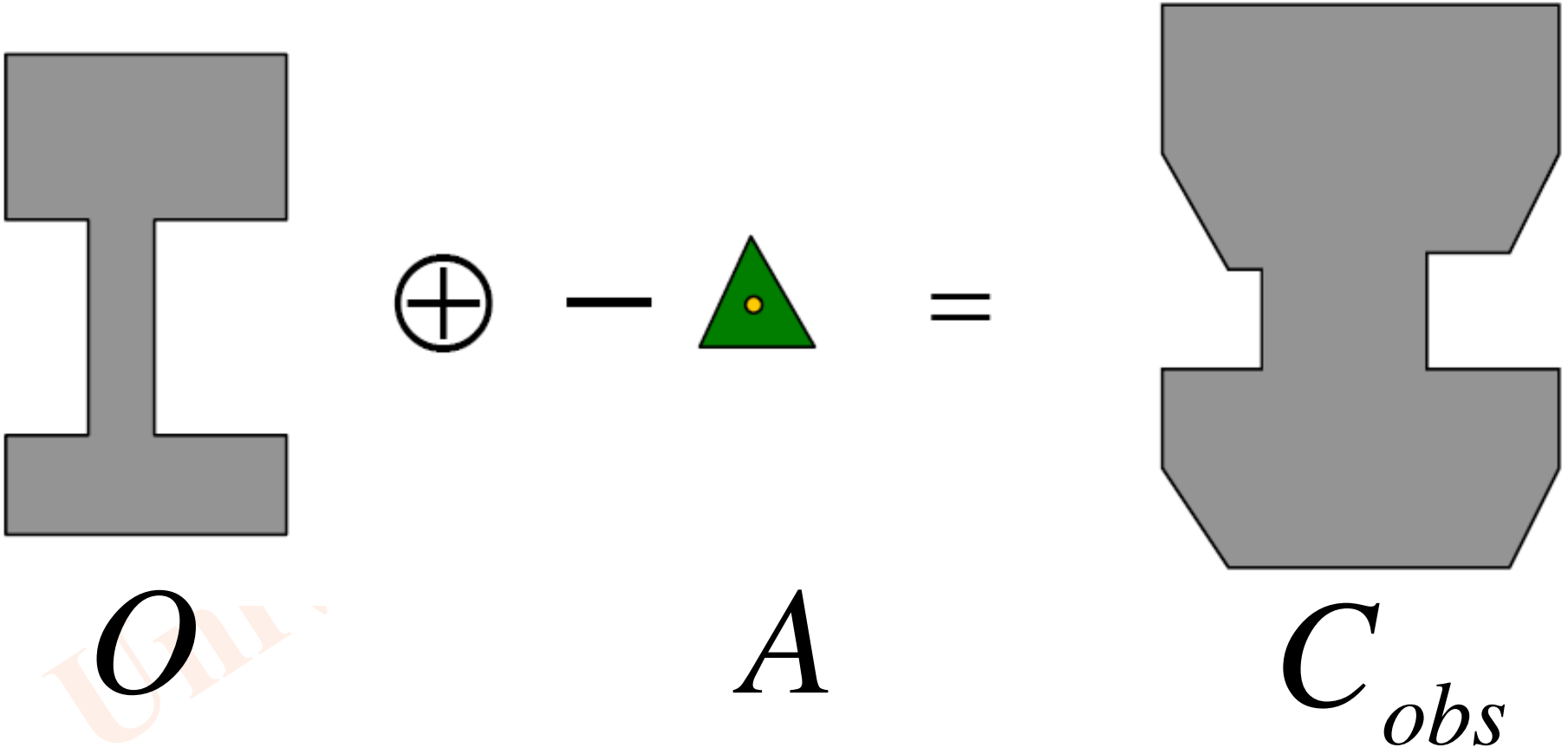
Figure by Dinesh Manocha, UNC

$$A \oplus B = \{a + b \mid a \in A, b \in B\}$$

Figure by Dinesh Manocha, UNC

Lozano-Perez and Wesley, 1979

$$C_{obs} = O \oplus -A$$



Complexity

O – 2D convex polygon, m vertices

A – 2D convex polygon, n vertices

Minkowski sum is a convex polygon of $m+n$ vertices

Run time ~ $O(n+m)$

Nonconvex case is much harder

- decompose into convex polygons
- compute Minkowski sums
- take unions

Run time ~ $O(n^2m^2)$

Run time ~ $O(n^3m^3)$ in 3D

Obstacle Regions for Translation

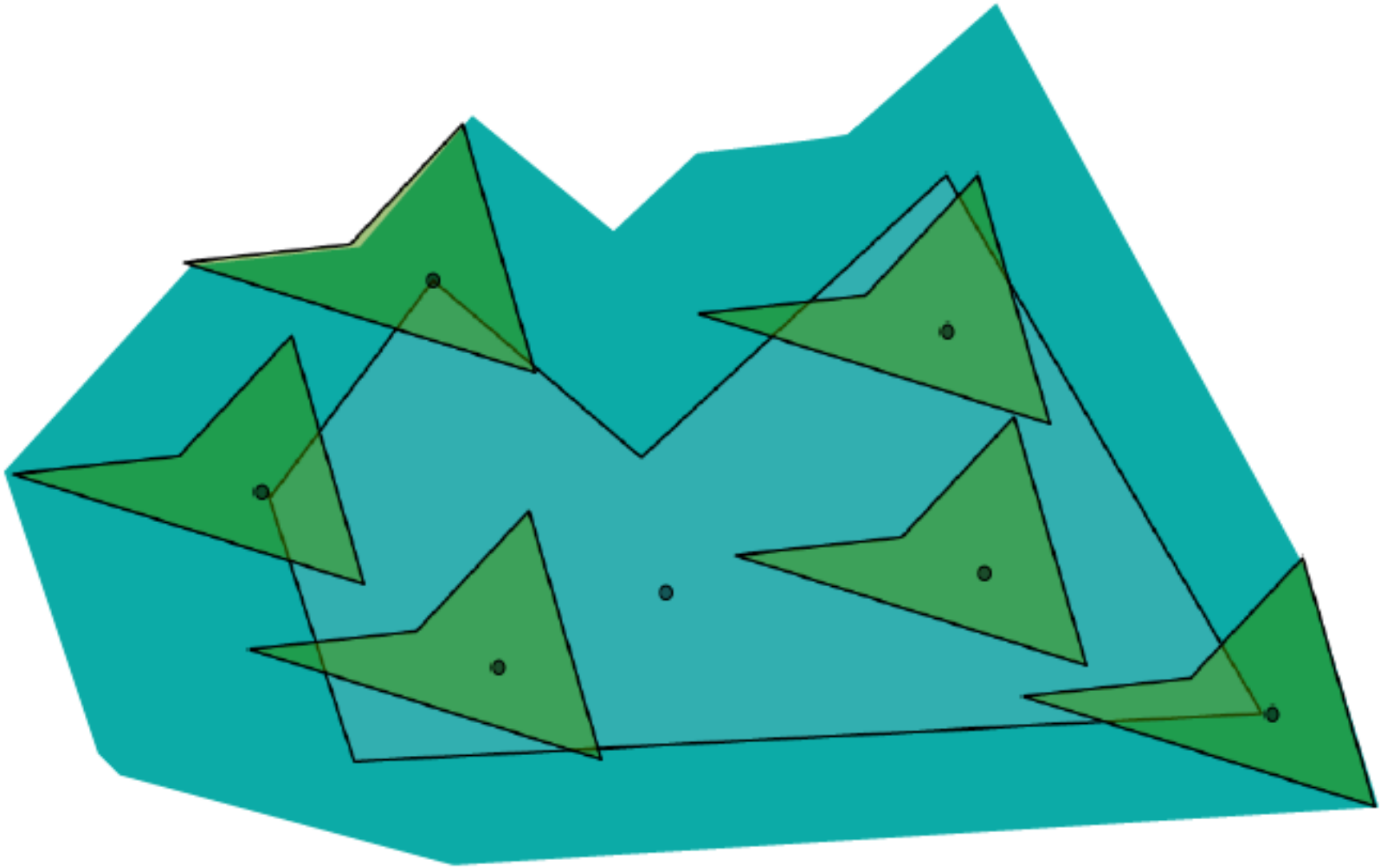
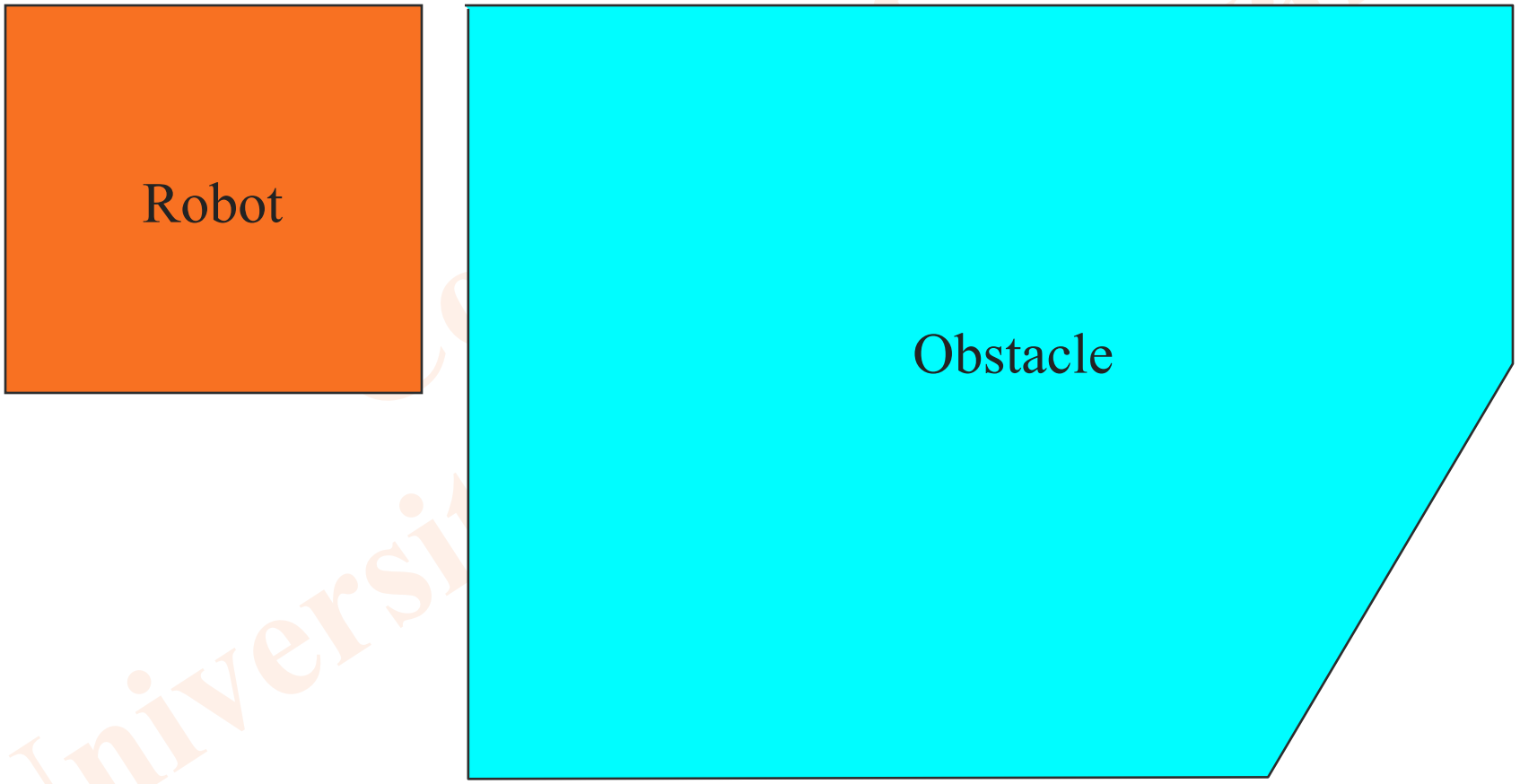
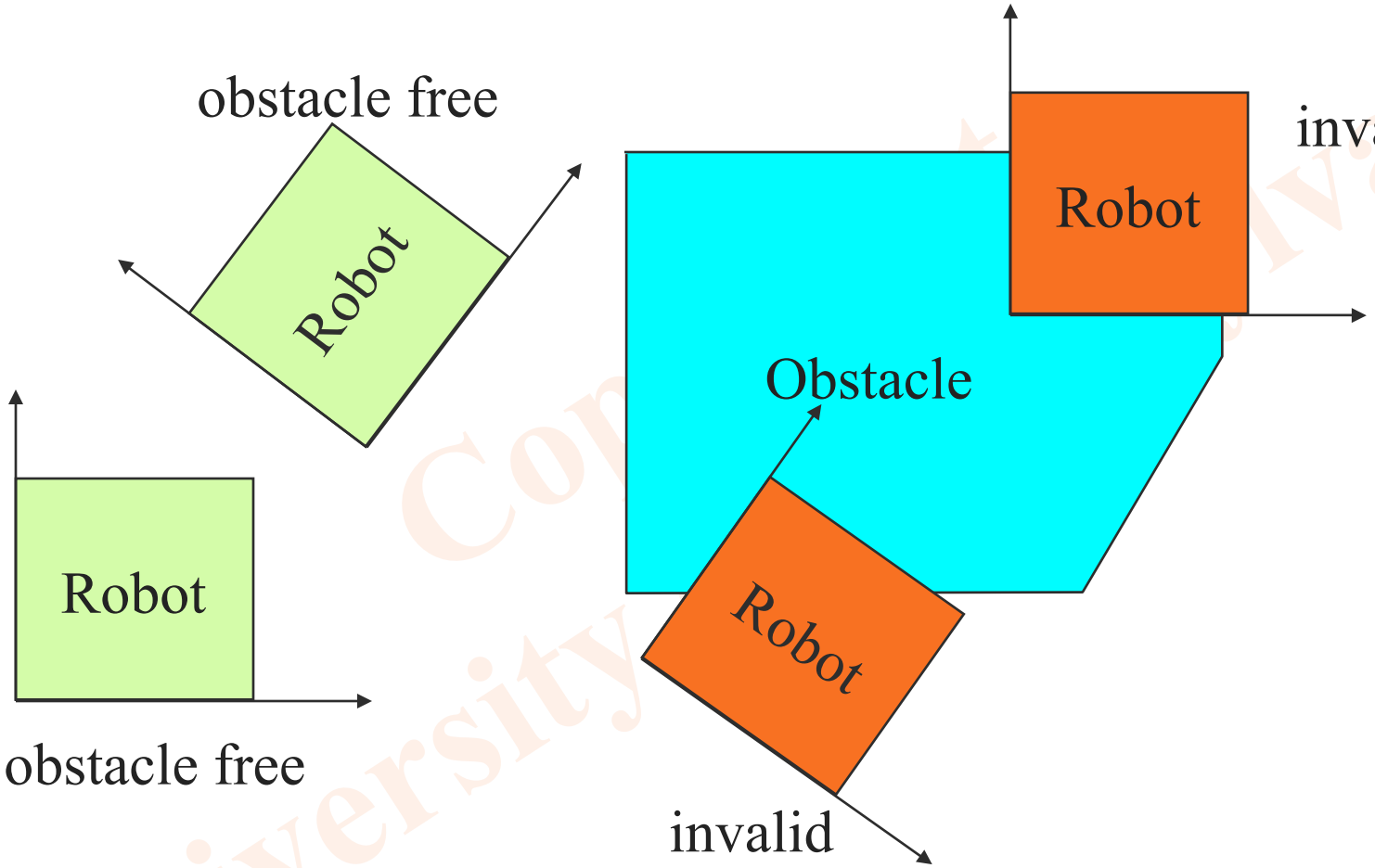


Figure by Dinesh Manocha, UNC

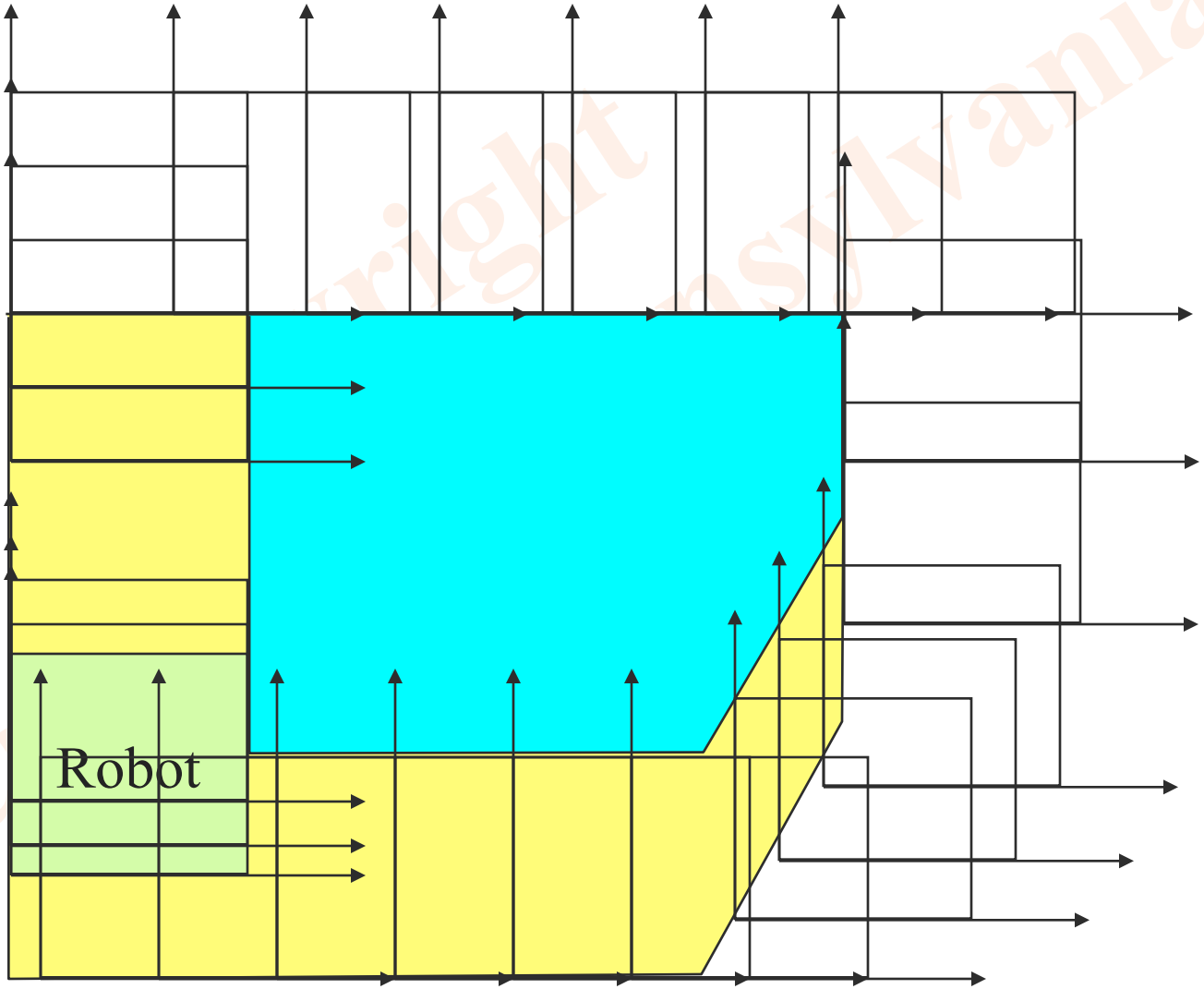
What if the robot is rigid body that can translate and rotate in the plane?



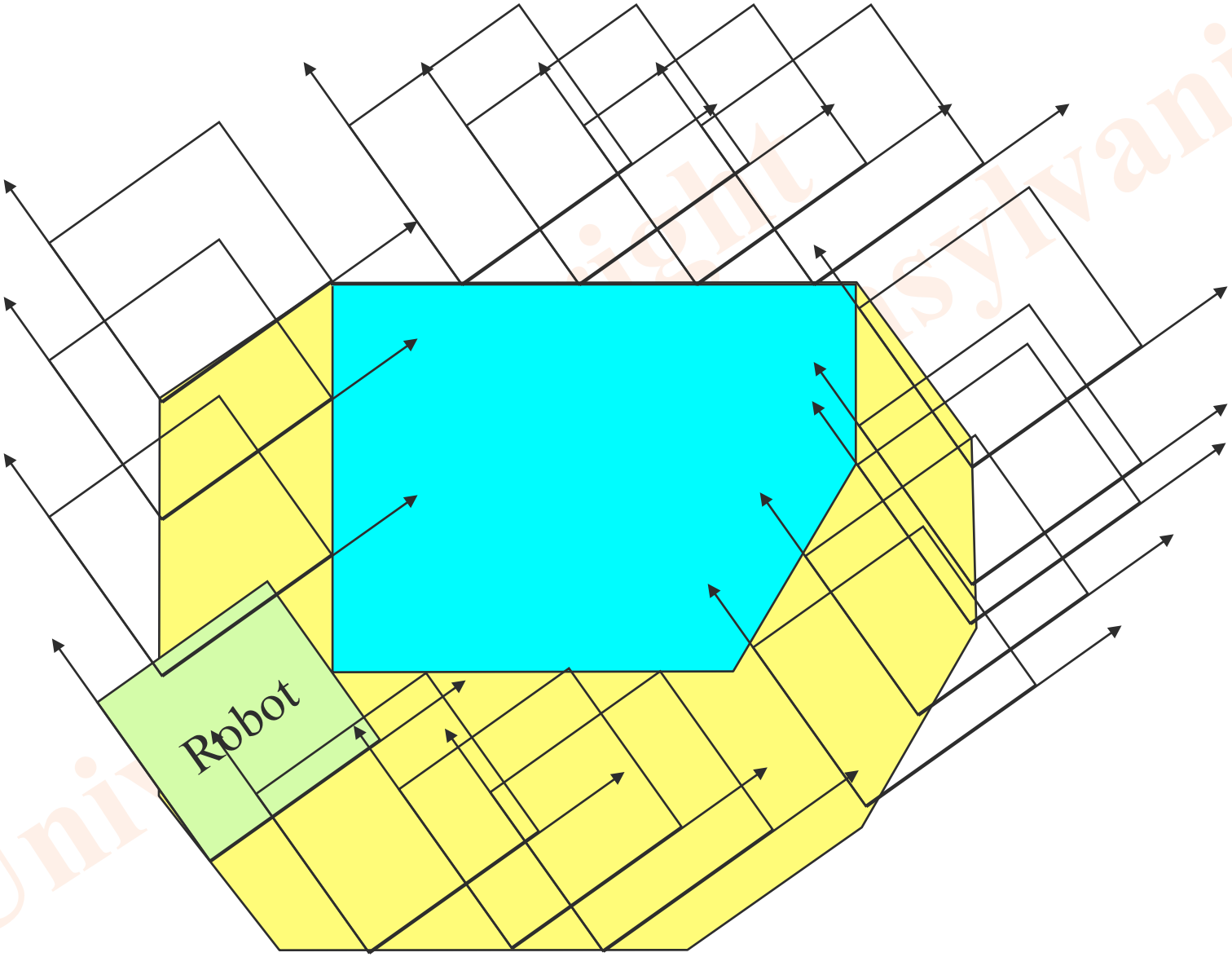
C-space

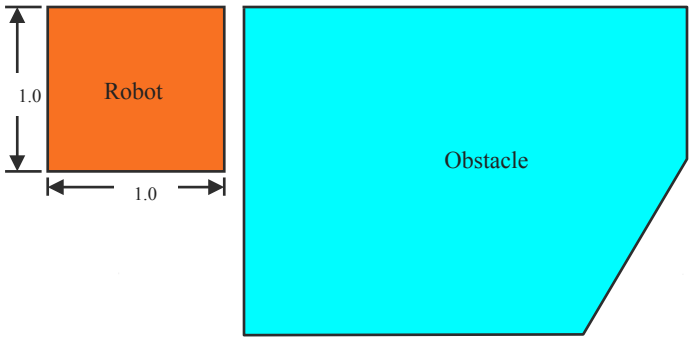


C_{obs} (θ=0)

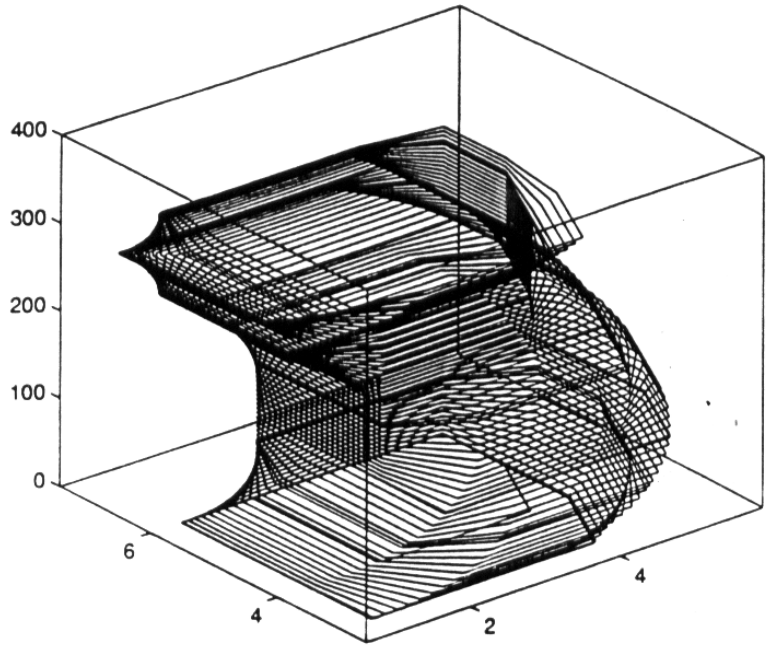


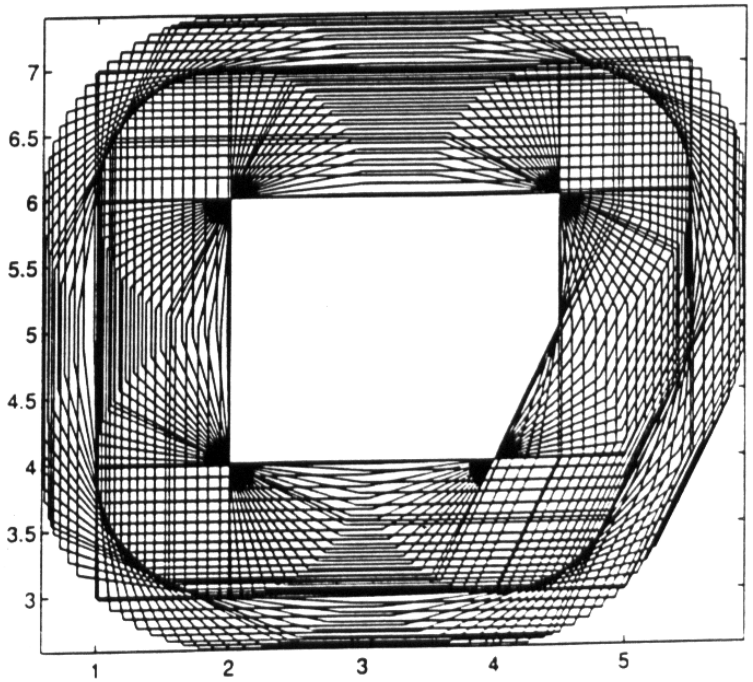
C_{obs} ($\theta=\theta_0\neq 0$)





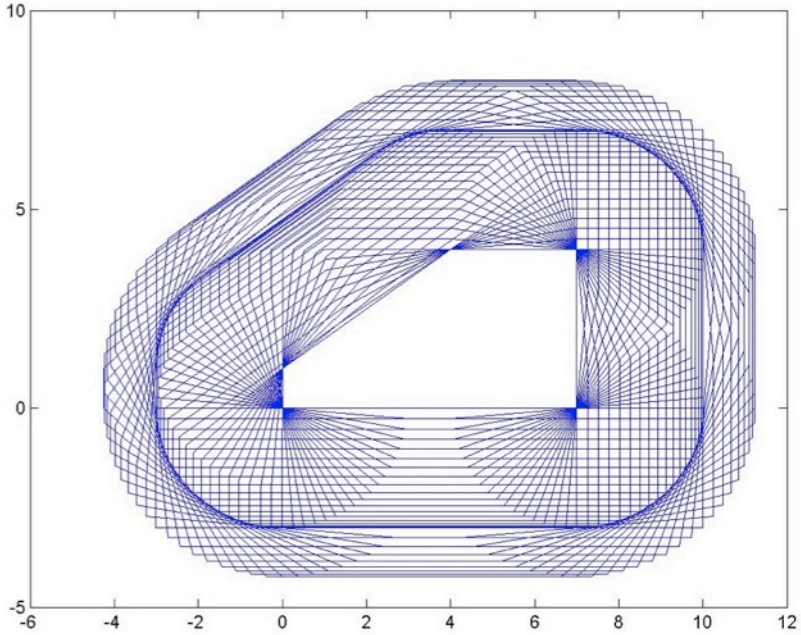
Slices of C_{obs}

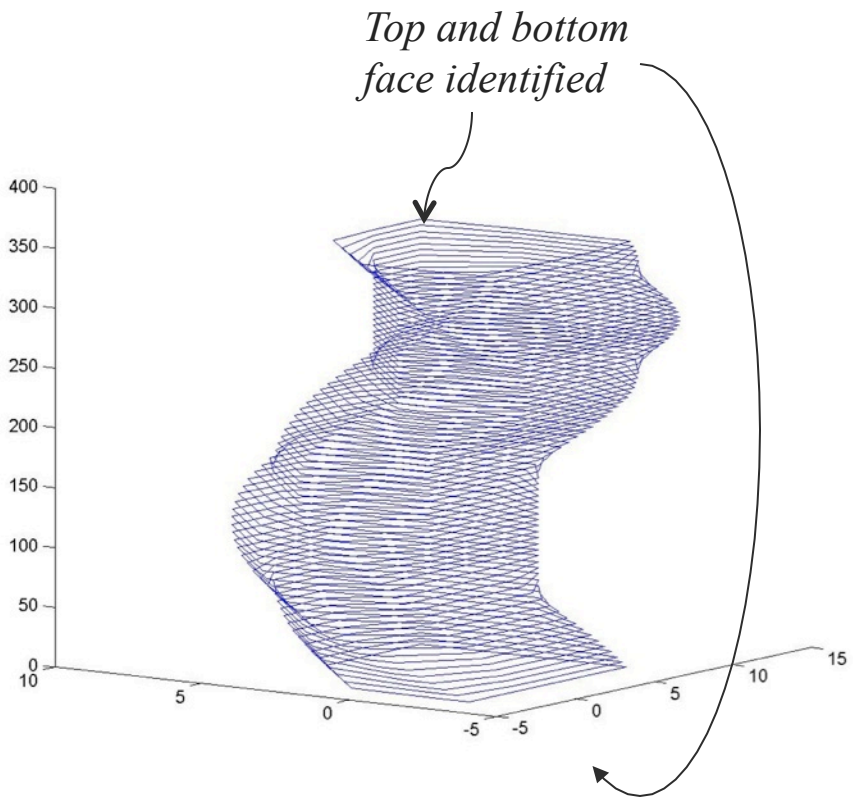




Each slice is a convex polygon of $\leq$ 4+5 vertices

C_{obs}





Details on computation with polygonal objects/robots available in Lavalle (3.1.1, 4.3)

C_{obs} is modeled as a union of semi-algebraic sets (Lavalle 3.1.2)

Challenges in Motion Planning

Designing a complete motion planning algorithm

A **complete** algorithm finds a path if one exists and reports no otherwise

- Dimensionality (2, 3 for point robots in Euclidean spaces, 3, 6 with orientations)
- Presence of obstacles

  - having to find narrow corridors
- Articulated chains (even in 2-D)
- Topology – C_{free} can be really complicated
- Representation of C_{free}

  -finite memory

  -reason with finite processing capability
- Constraints (kinematic, dynamic)

  - Ground mobile robot may have nonholonomic constraints

  - Aerial robots must obey laws of motion and actuator constraints

Four Approaches

1. Decompose C_{free} into cells

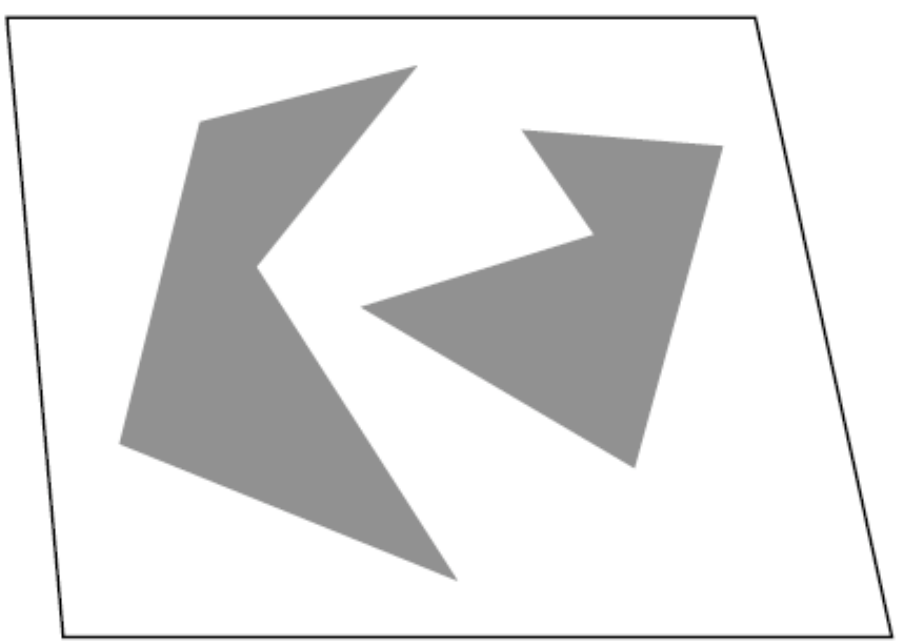
   Exact methods

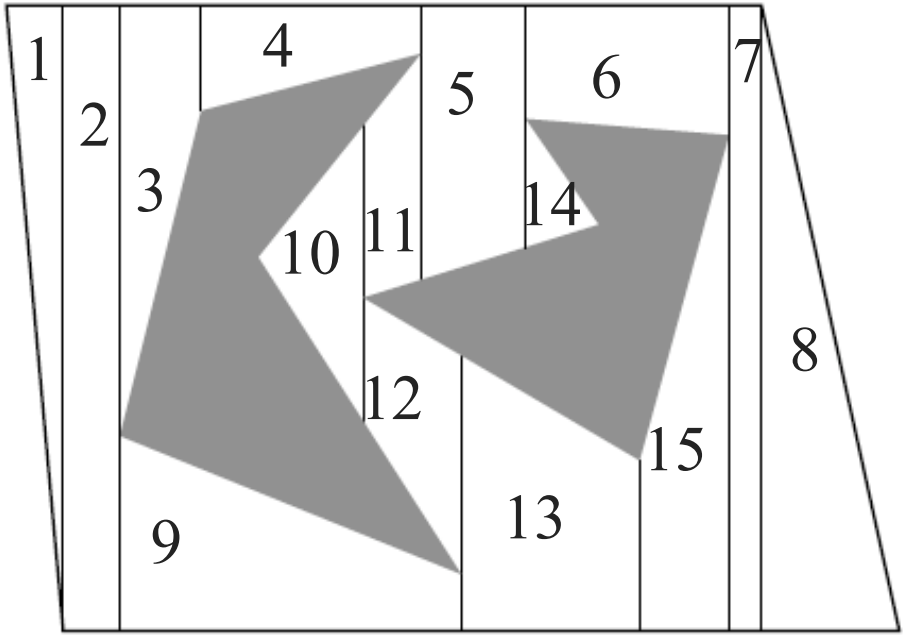
   Approximate methods

2. Use roadmaps to represent C_{free}
3. Probabilistic methods
4. Artificial potential functions

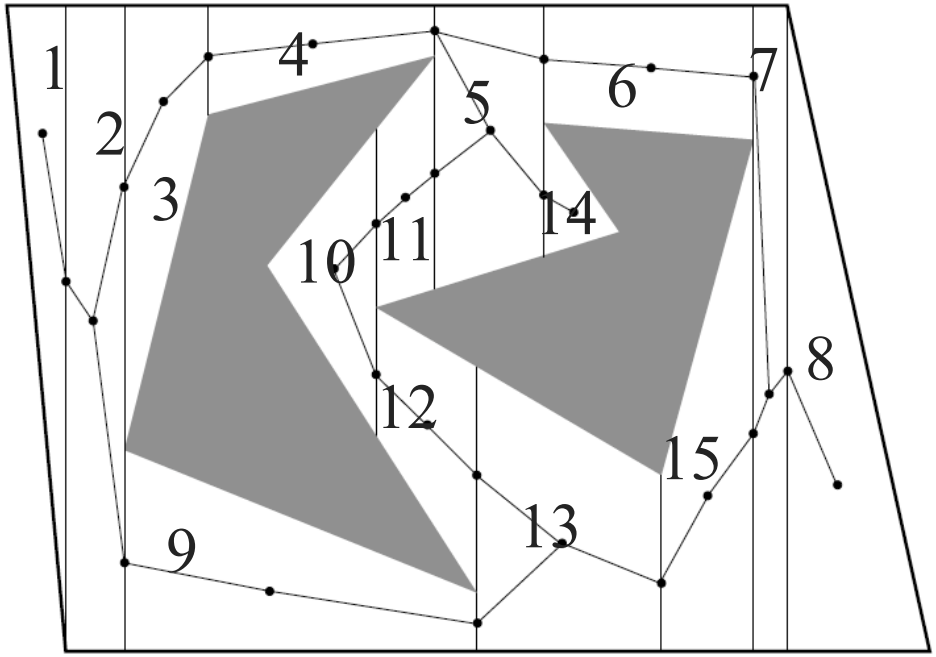
1. Cell Decomposition (Exact)

Example: Vertical Cell Decomposition in R^2

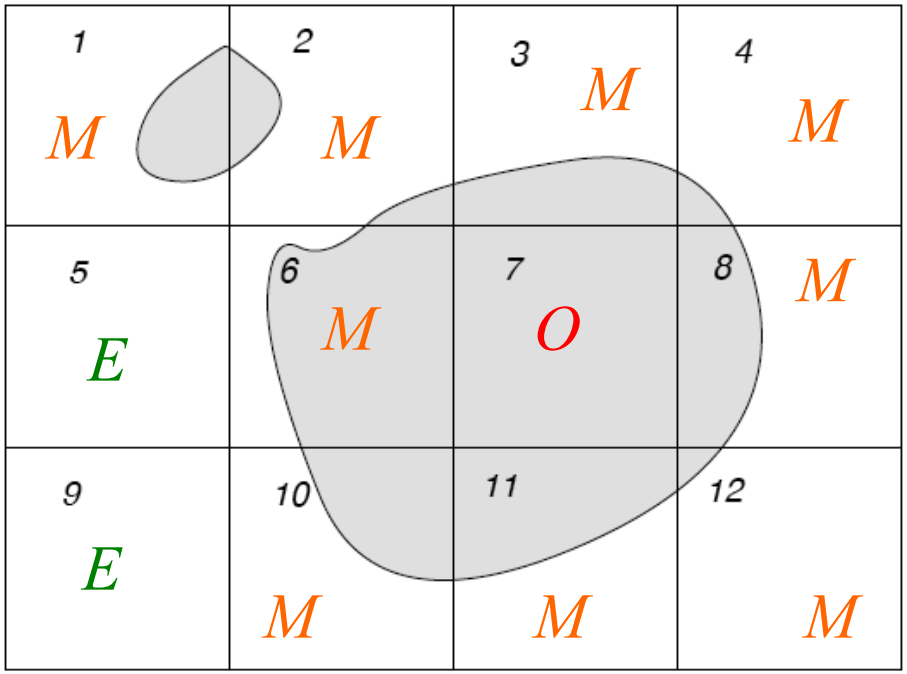


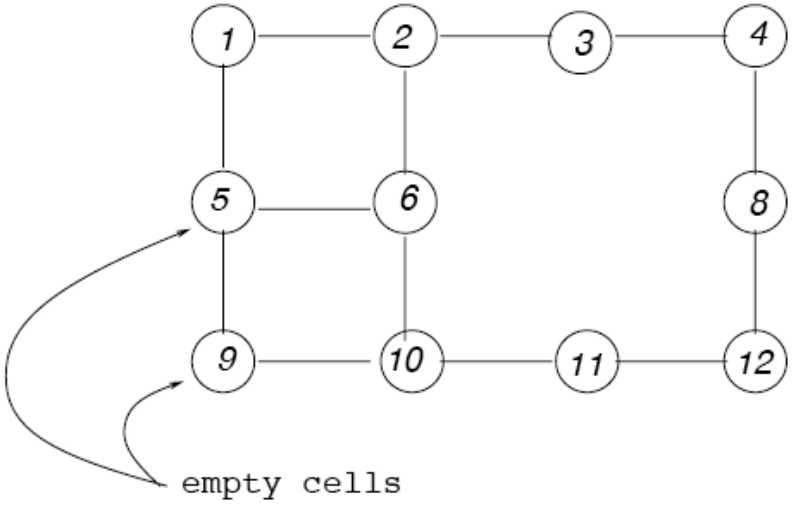


Motion planning can be reduced to a discrete graph search problem



1. Cell Decomposition (Approximate)





E - empty

M - mixed

O - occupied

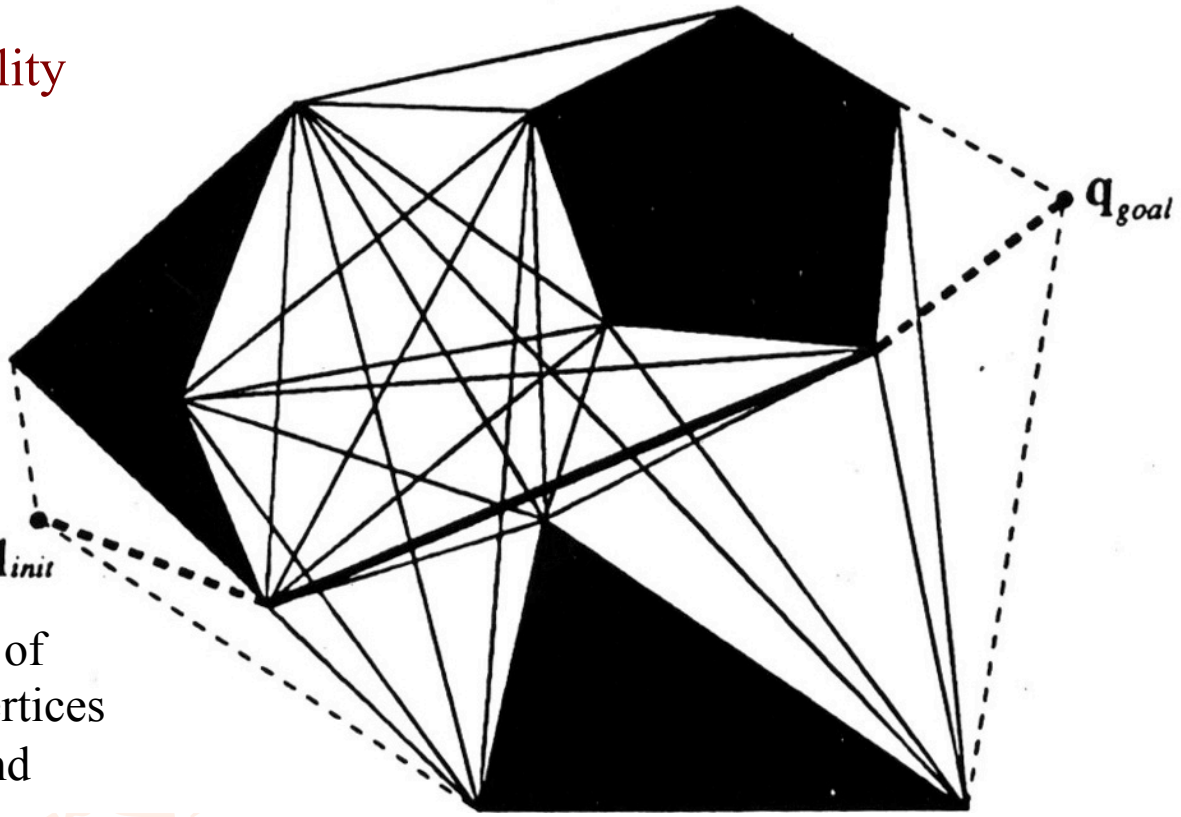
Search Algorithms

- Approximate cell decomposition yields a discrete representation of the world
- Efficient search techniques allow online computation of motion plans

  (leveraging 40 plus years of work on graph search)

2. Roadmaps

Example 1: Visibility Graphs in R^2



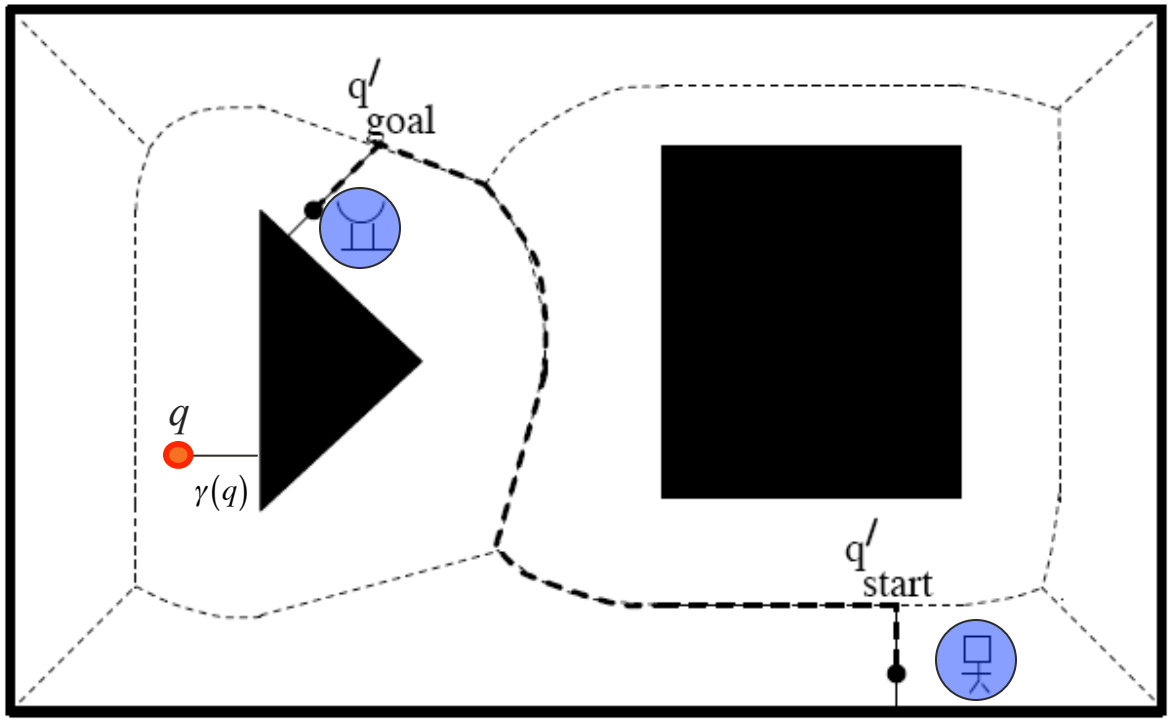
- Join all pairs of vertices: *n* vertices plus initial and final starting position
- Eliminate edges that intersect obstacles

- Run Dijkstra's algorithm to solve the **single-source shortest paths** problem

[Latombe 91]

2. Roadmaps

Example 2: Generalized Voronoi Diagram in R^2 [Choset, 1996]



1. Clearance function

$$\gamma(q) = \min_{s \in \partial C_{free}} \|q - s\|$$

2. Neighbor(s)

$$N(q) = \{ s \in \partial C_{free} : \|q - s\| = \gamma(q) \}$$

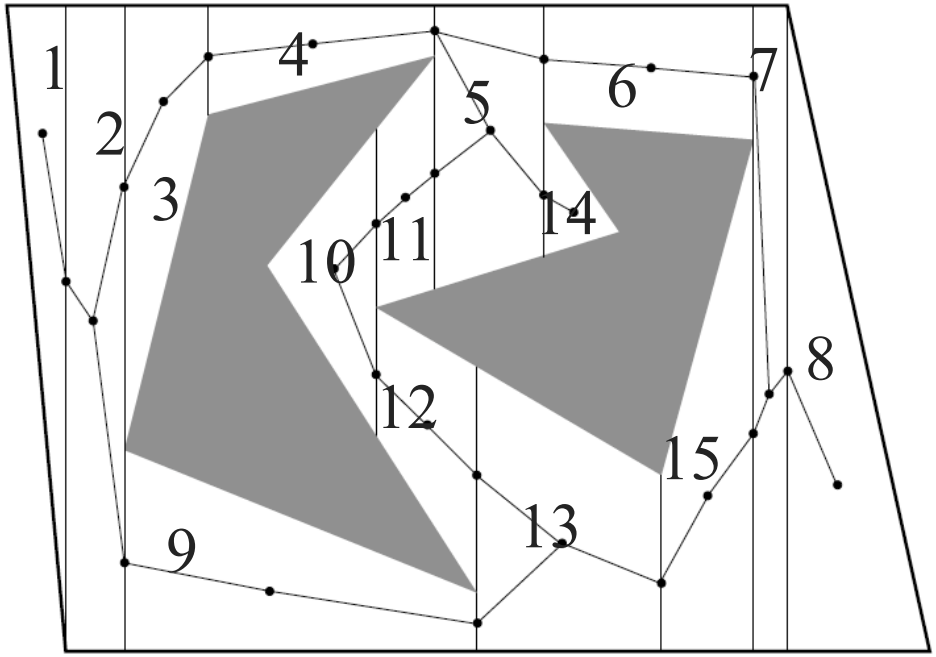
3. Voronoi Diagram of C_{free}

$$V(C_{free}) = \{ q \in C_{free} : card(N(q)) > 1 \}$$

RM is a roadmap of C_{free} if it is accessible and connected

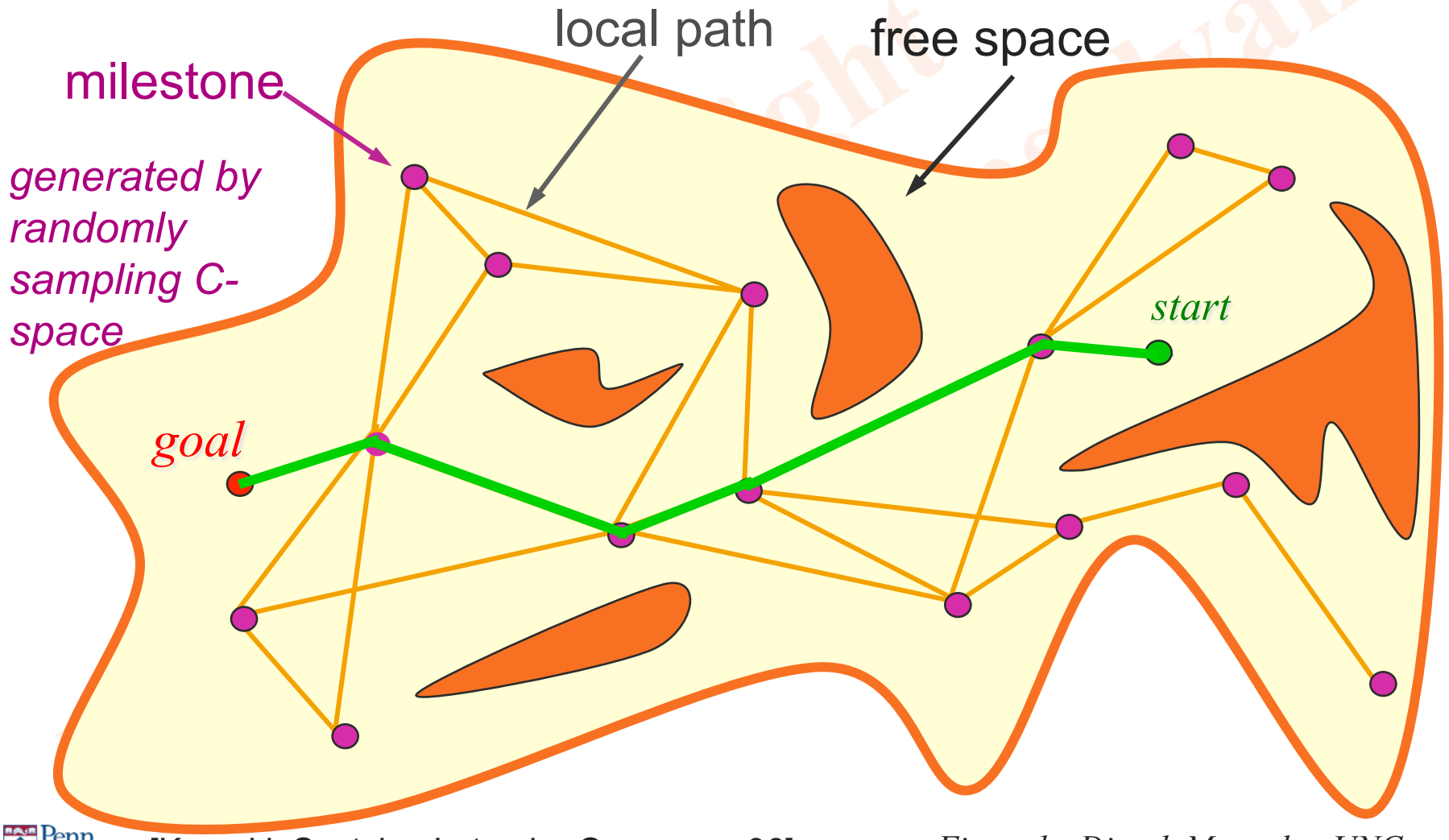
Possible to go from any q to the *RM*

Example 3: Roadmap via Cell Decomposition in R^2



3. Probabilistic Methods

Example: Probabilistic Roadmap in R^2

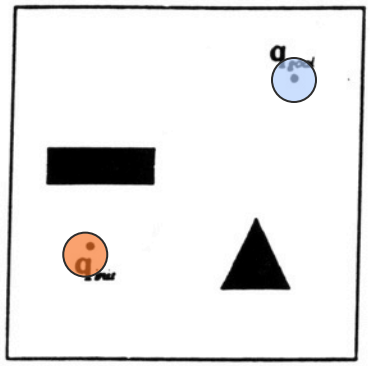


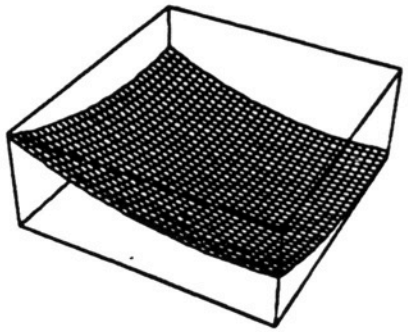
[Kavraki, Svetska, Latombe,Overmars, 96]

Figure by Dinesh Manocha, UNC

4. Artificial Potential Function

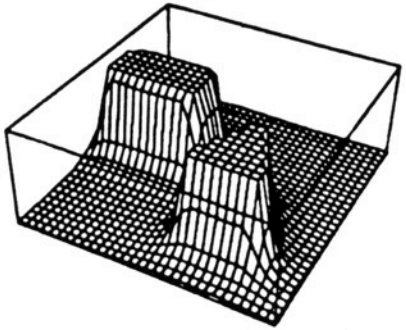
[Latombe 91]

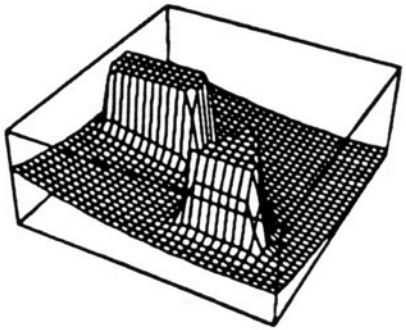




Attractive potential field for goal

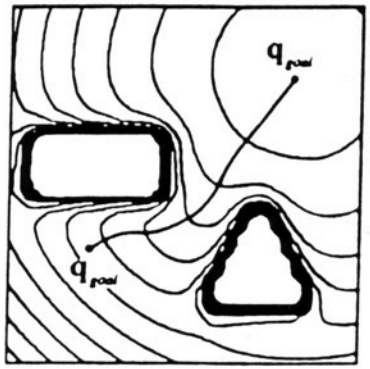
Repulsive potential field for obstacles

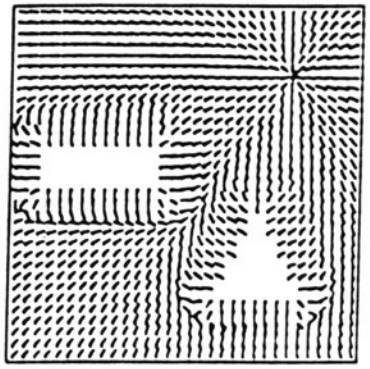




Attractive + repulsive potential fields

Equi-potential contours





Force field

- Force on a particle is given by $f = -grad\ (V)$
- Command robot velocity according to the following control law (policy)

Comparison

	Exact Cell Decomposition	Approx Cell Decomposition	Roadmap methods	Probabilistic methods	Potential field methods
Practical above 2 or 3 dimensions	Very slow, memory	Slow, memory (m^N requirement*)	Very slow	Fast	Fast
Practical above 5 dimensions	No	Perhaps not, not clear	No	Yes	Yes
Optimality	Yes	Yes, up to resolution	Yes	Probabilistic guarantees	No
Easy to implement	No	Yes	No	Yes	Yes
Completeness	Yes	Yes, up to resolution	Yes	Probabilistically complete	No**

If there is a solution path, the algorithm will find it with high probability

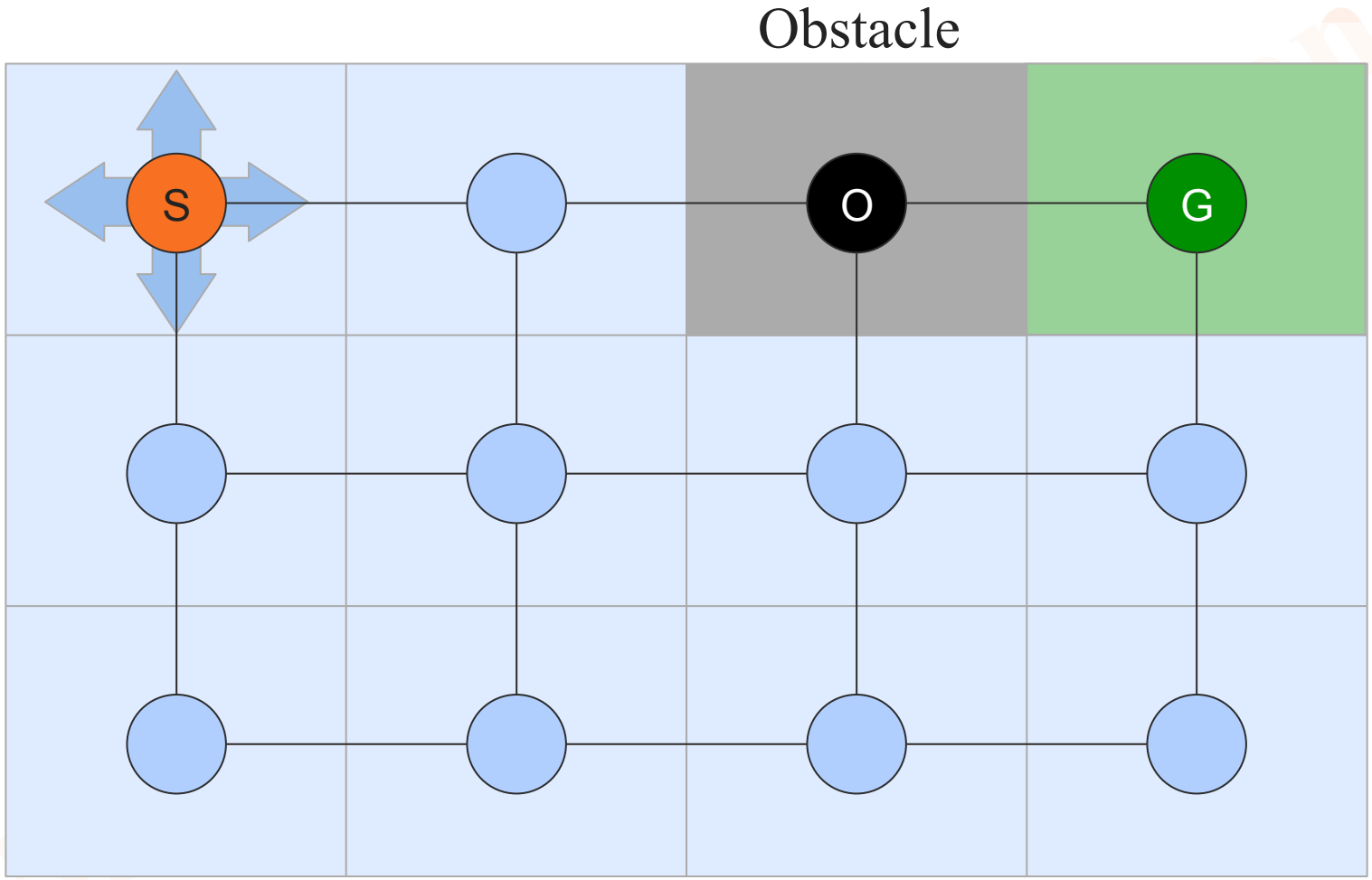
**m* grid cells along each of *n* axes

***Navigation functions in certain classes of environments provide theoretical guarantees

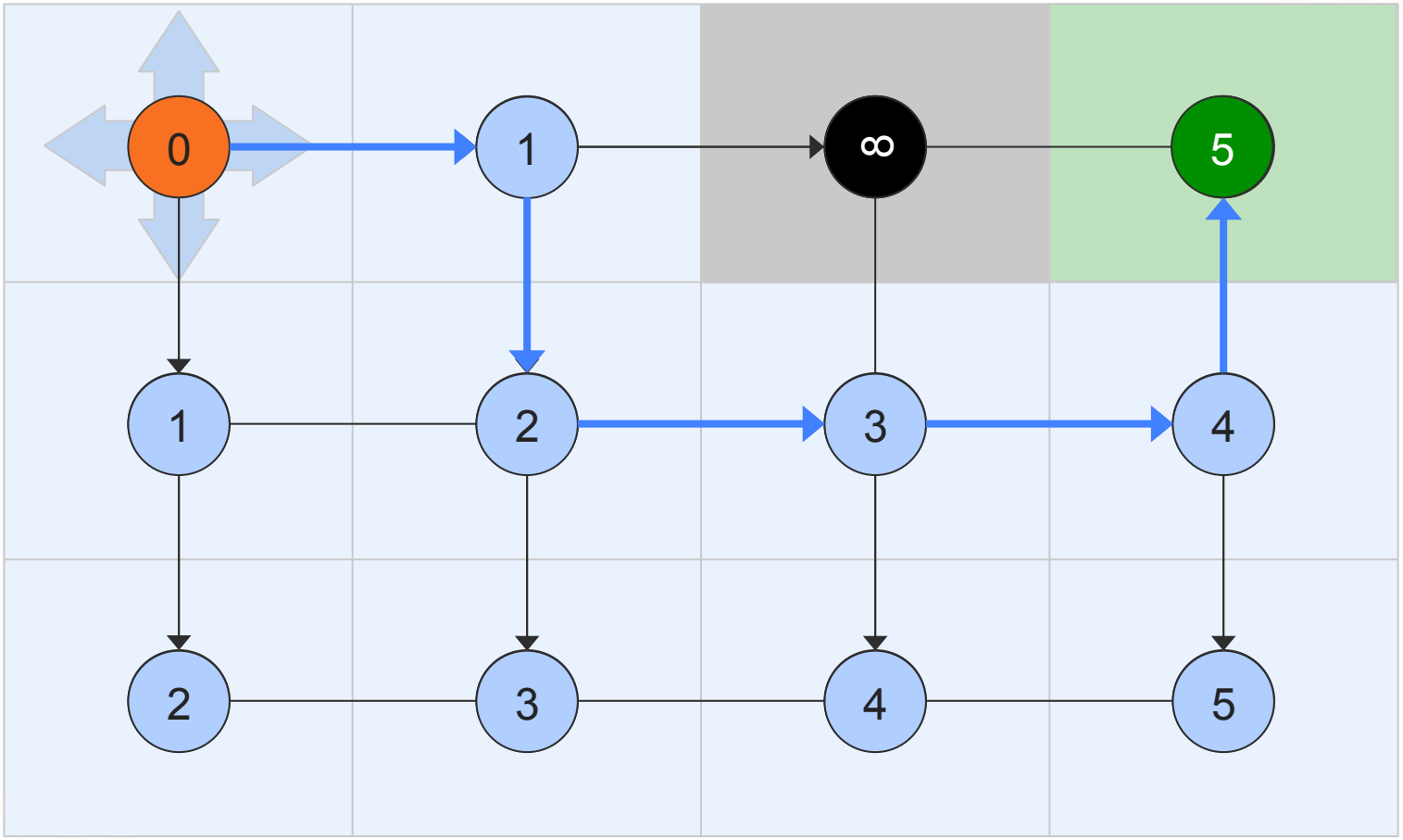
Approximate Cell Decomposition and Graph Search

- Dijkstra's algorithm
- A* algorithm

Dijkstra's Algorithm – An example



Dijkstra's Algorithm – An example



Dijkstra's Algorithm – Animation

Penn Engineering

http://upload.wikimedia.org/wikipedia/commons/2/23/Dijkstras_progress_animation.gif

A*

http://en.wikipedia.org/wiki/A*_search_algorithm#mediaviewer/File:Astar_progress_animation.gif

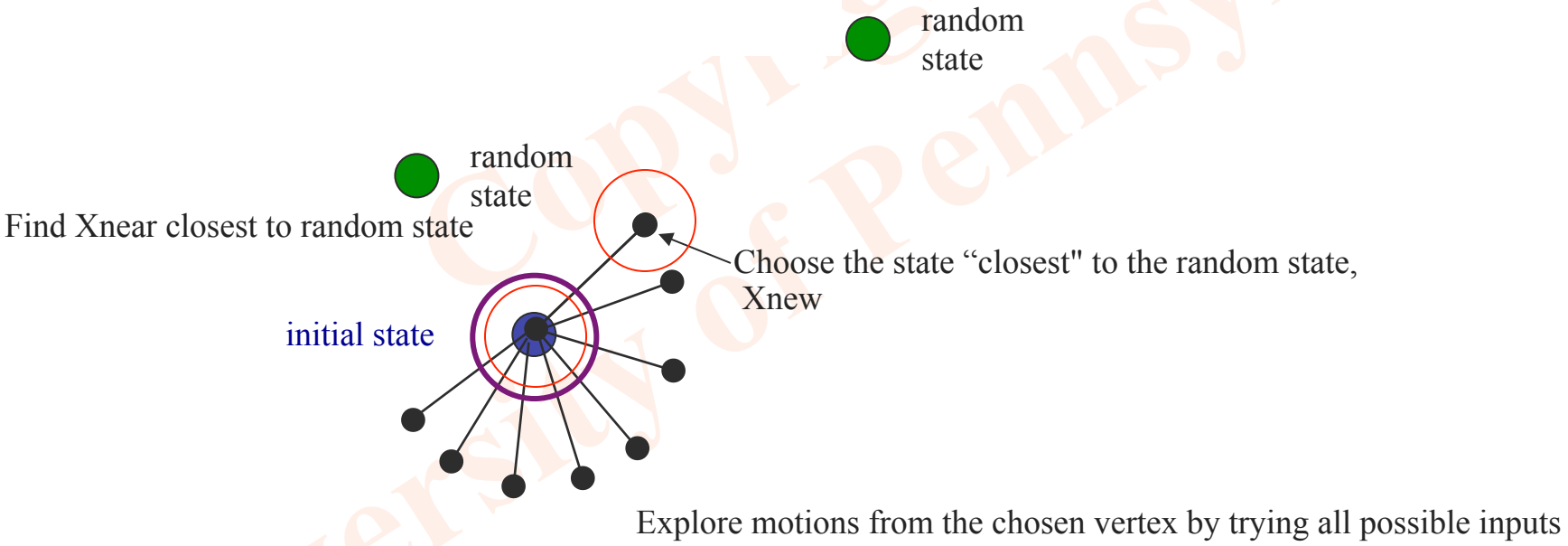
RRT
Rapidly-exploring Random Tree

(Probabilistic Technique)

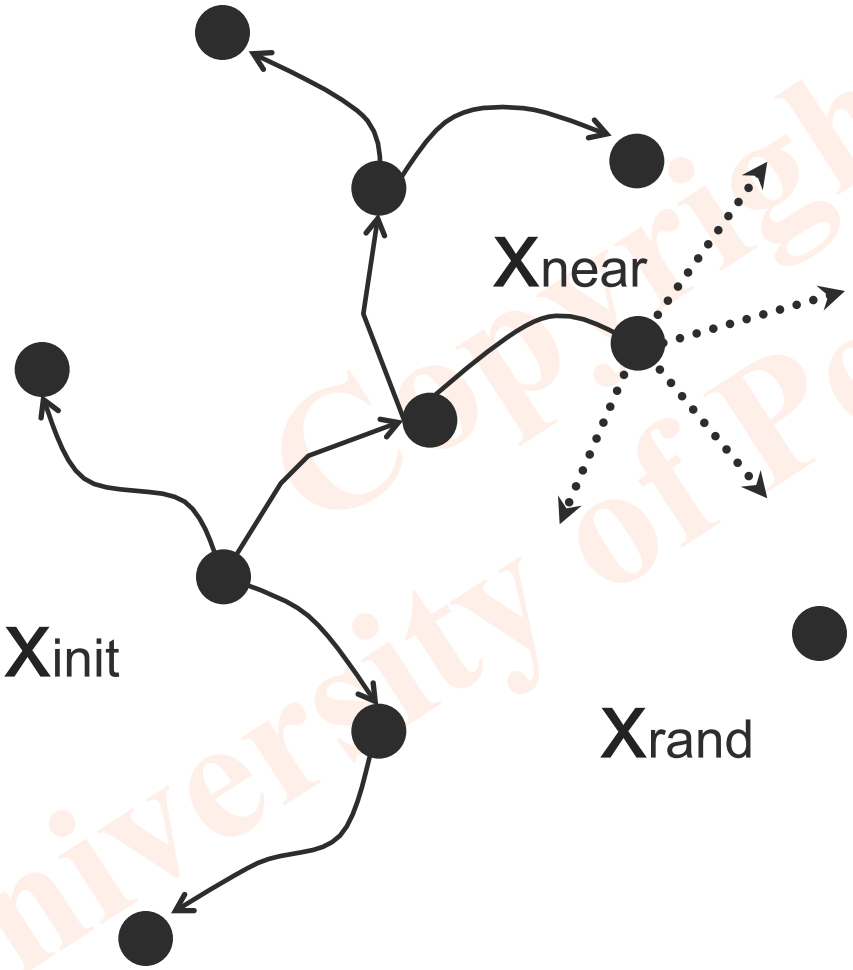
Lavalle and co-workers, 2000-2003

LaValle, S., Planning Algorithms. Cambridge University Press, 2006.

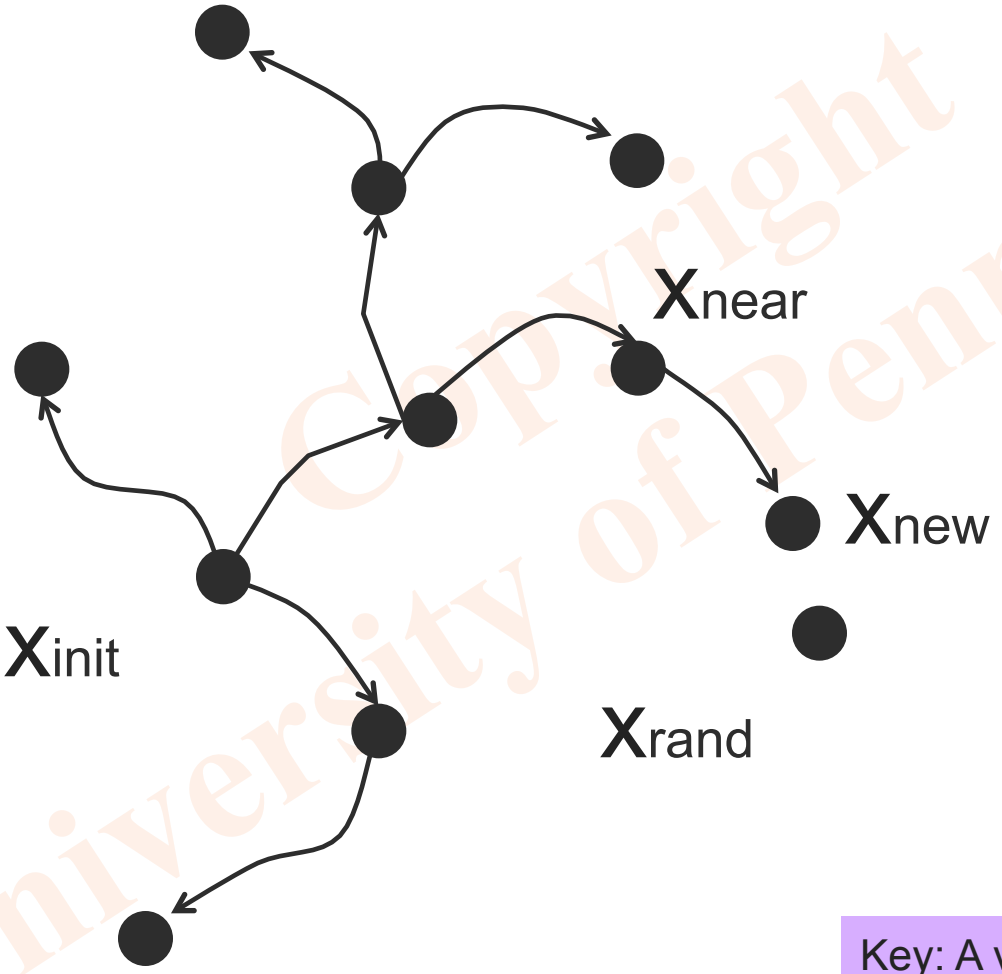
Overview of RRT method



kth edge



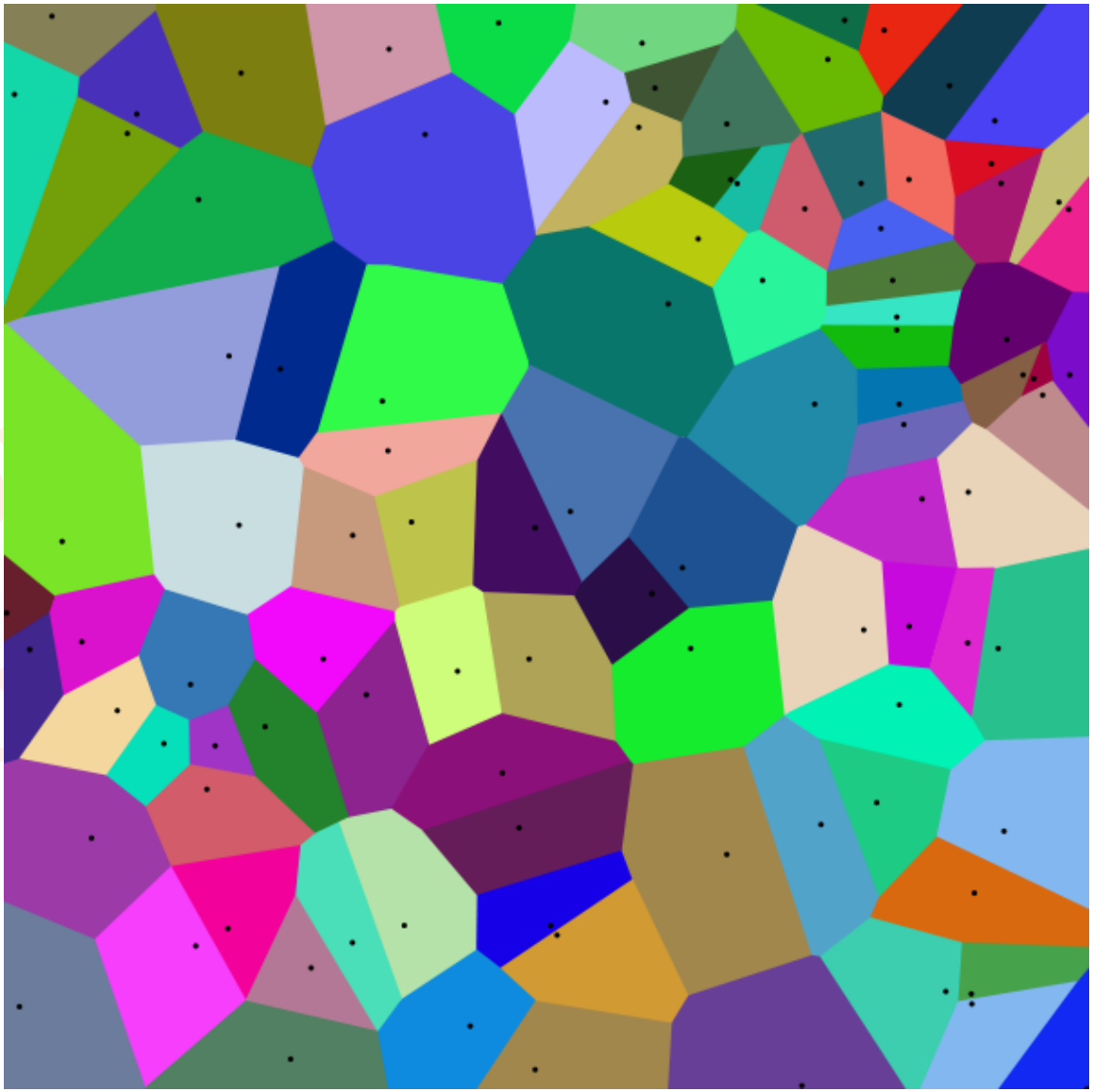
kth edge



Key: A vertex with a larger Voronoi region has higher probability of being chosen as X_{near}

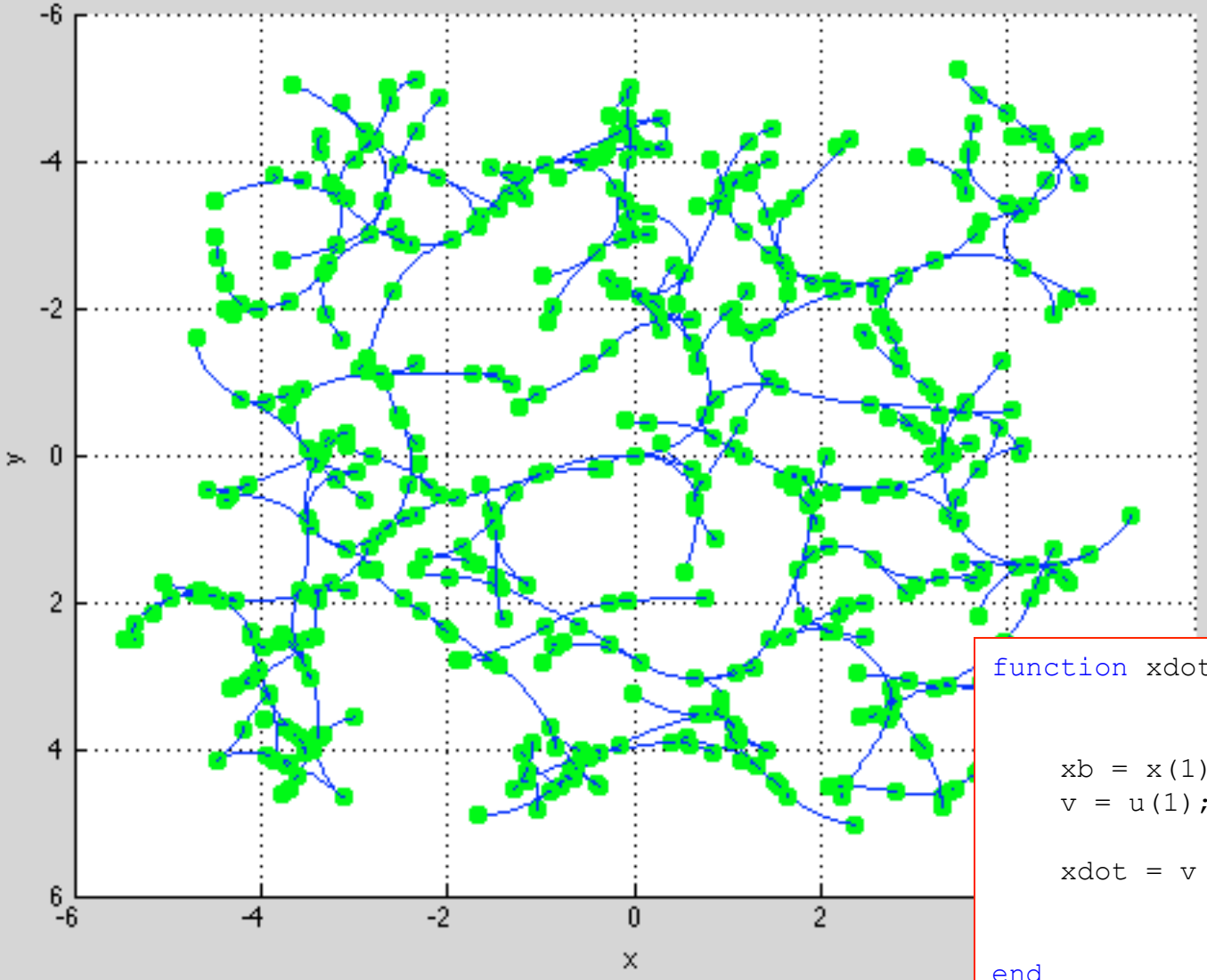
Voronoi Diagram

Probability of a randomly chosen point to lie in a set is proportional to the area of the set



http://en.wikipedia.org/wiki/Voronoi_diagram

Edges in tree satisfy constraints on input and dynamics



Cart model (3 states, 2 inputs)

No collision checking

P. Corke Matlab Toolbox

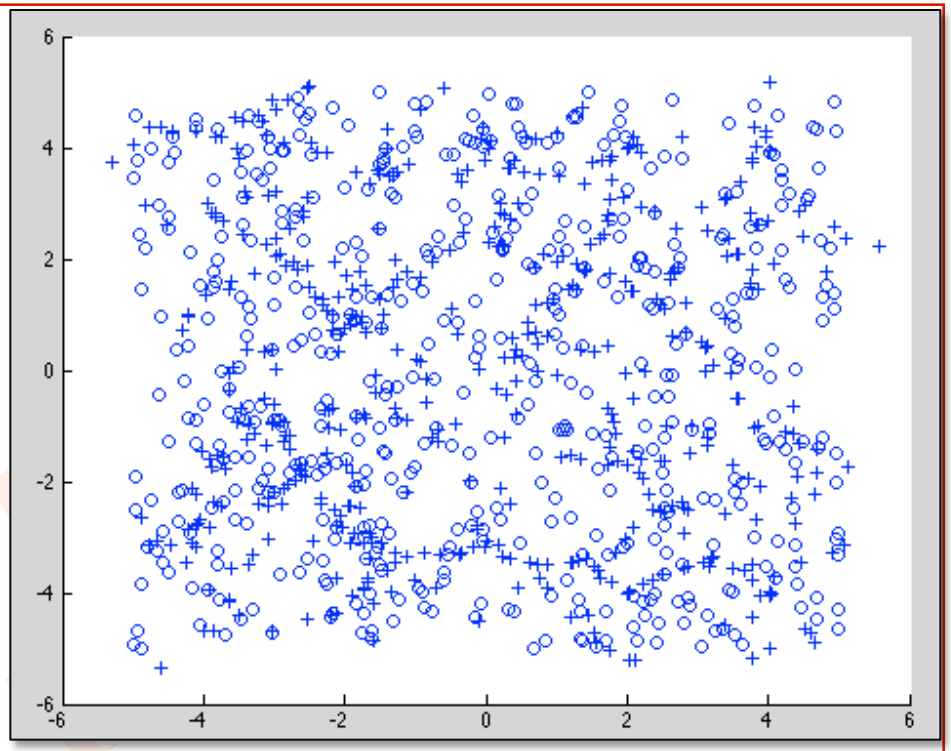
```
function xdot = bicycle(t, x, u, L)


    xb = x(1); yb = x(2); thb = x(3);
    v = u(1); gamma = u(2);

    xdot = v * [ cos(thb)
                 sin(thb)
                 tan(gamma) / L ];
end
```

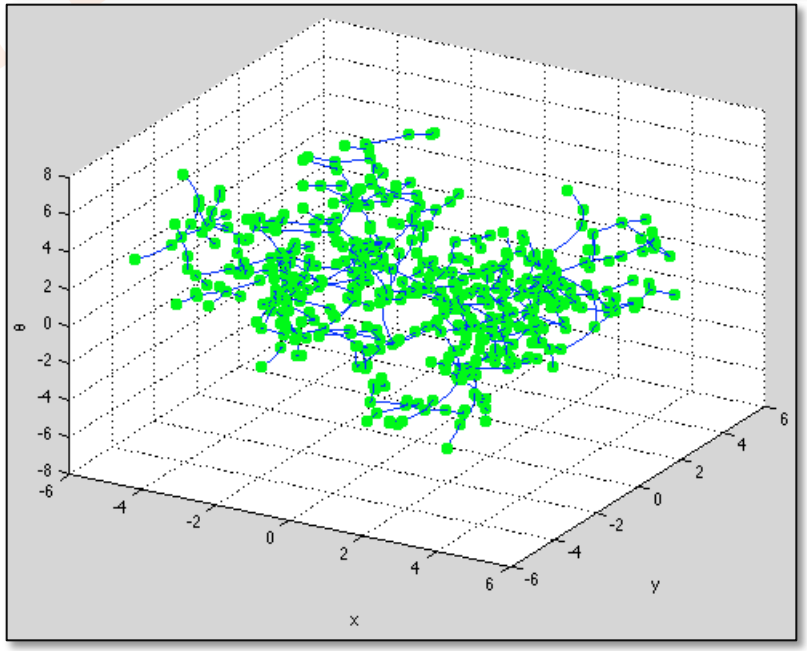
RVC Toolbox*

Bicycle Model**

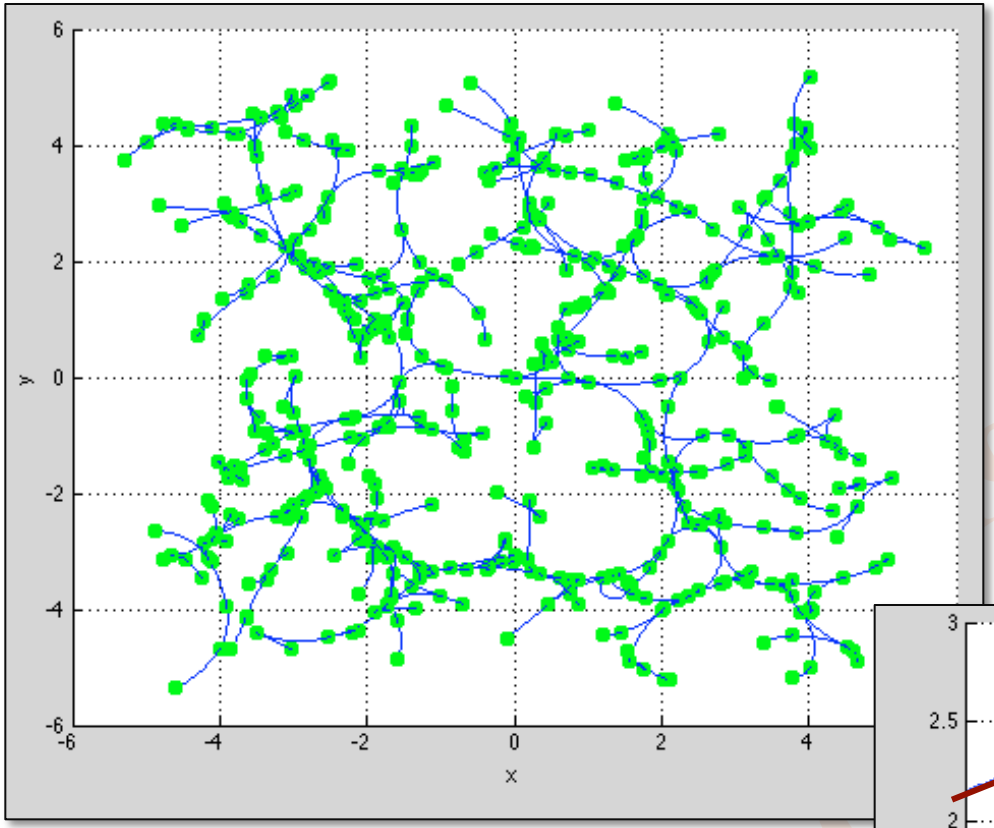
```
function xdot = bicycle(t, x, u, L)


    xb = x(1); yb = x(2); thb = x(3);
    v = u(1); gamma = u(2);

    xdot = v * [ cos(thb)
                 sin(thb)
                 tan(gamma) / L ];
end
```

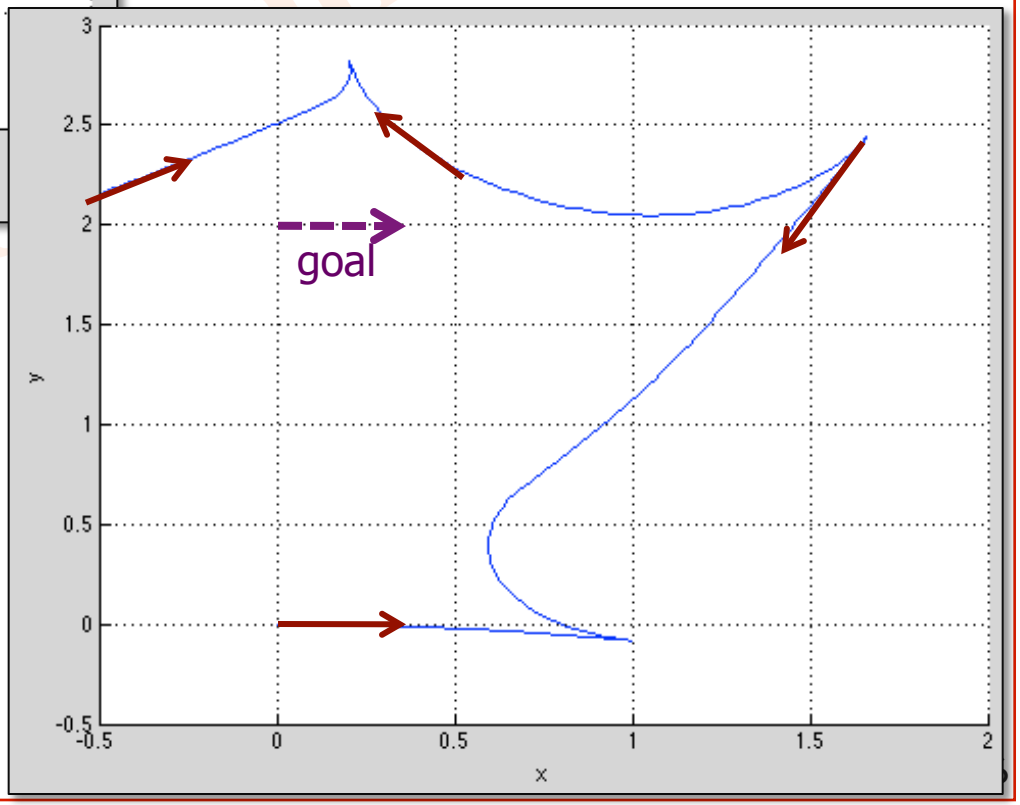
+/- 1 m/s
+/- 1.2 rad steering angle



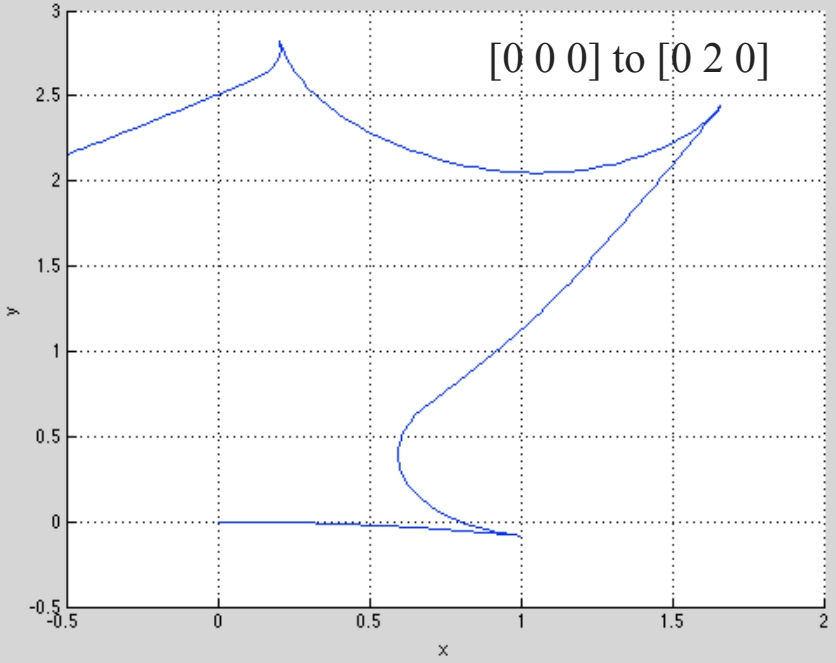


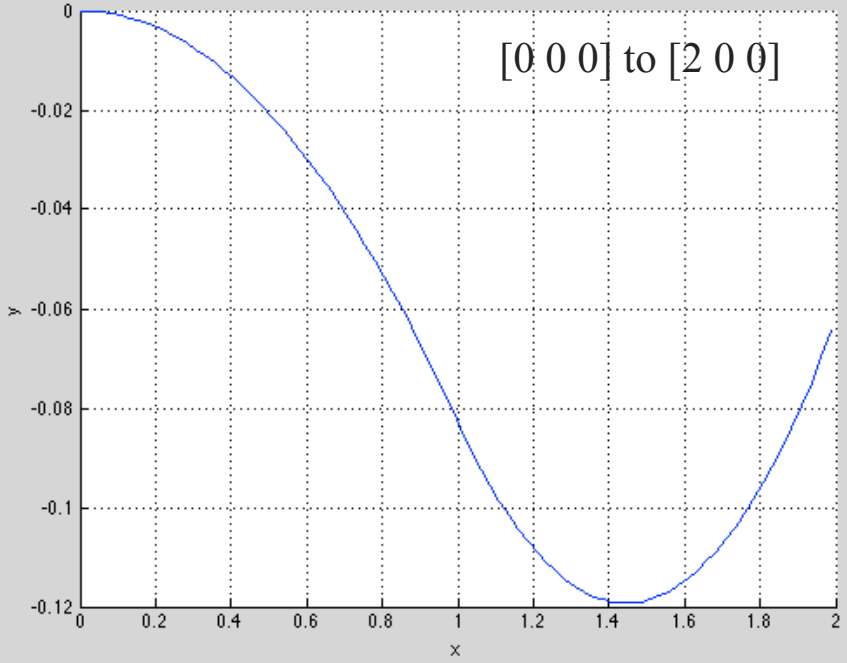
*RRT subclasses the Navigation class
**Bicycle model is hardwired into the class

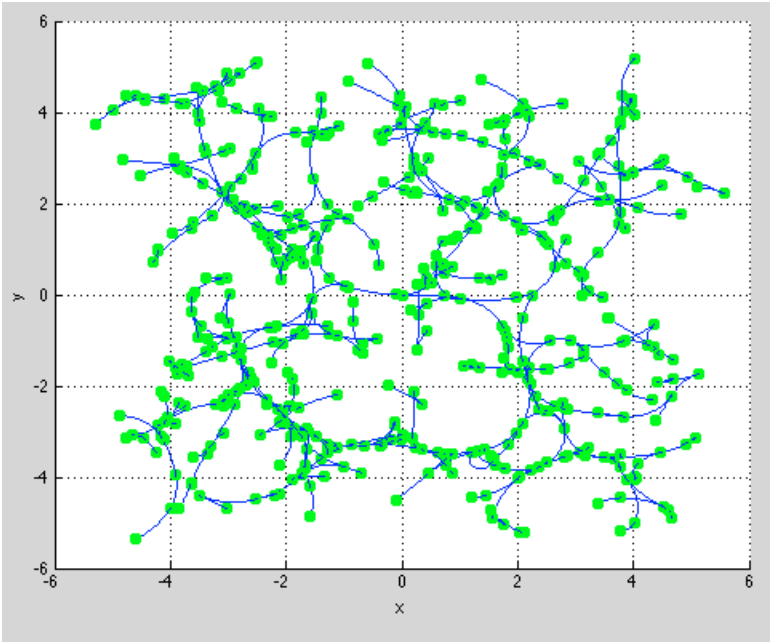


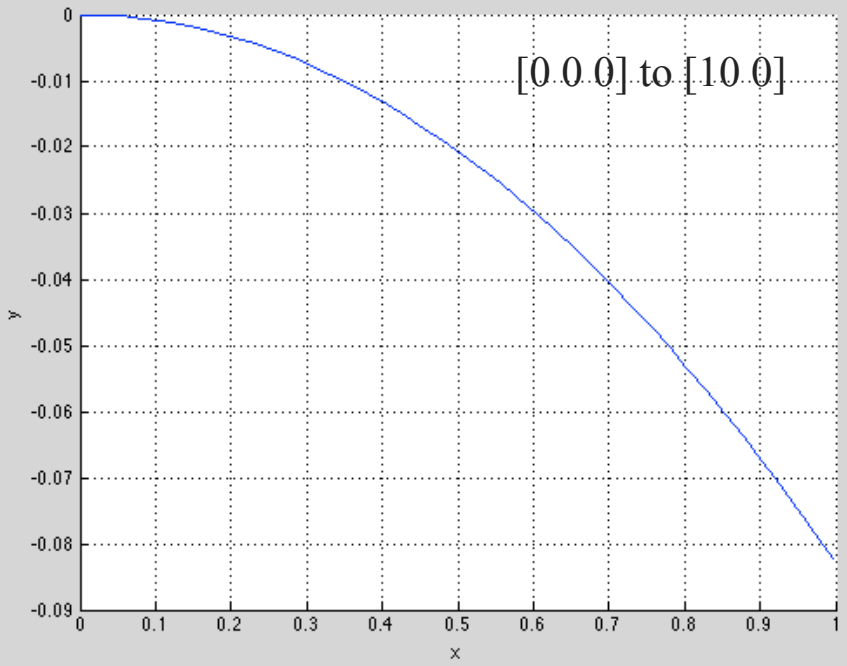


Penn Engineering



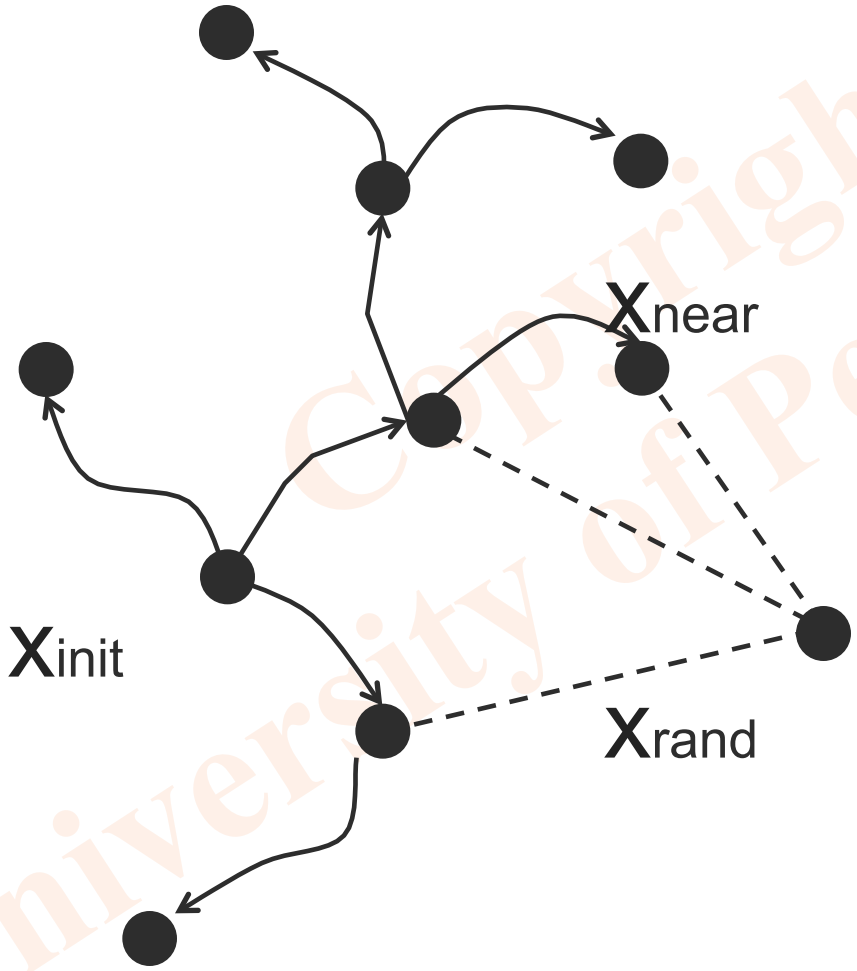






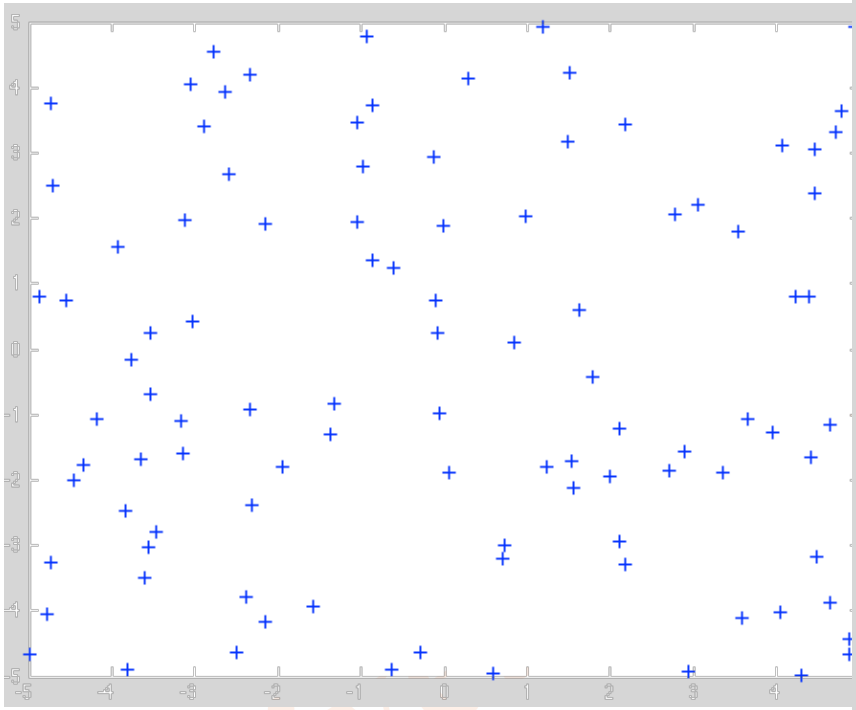
Engineering

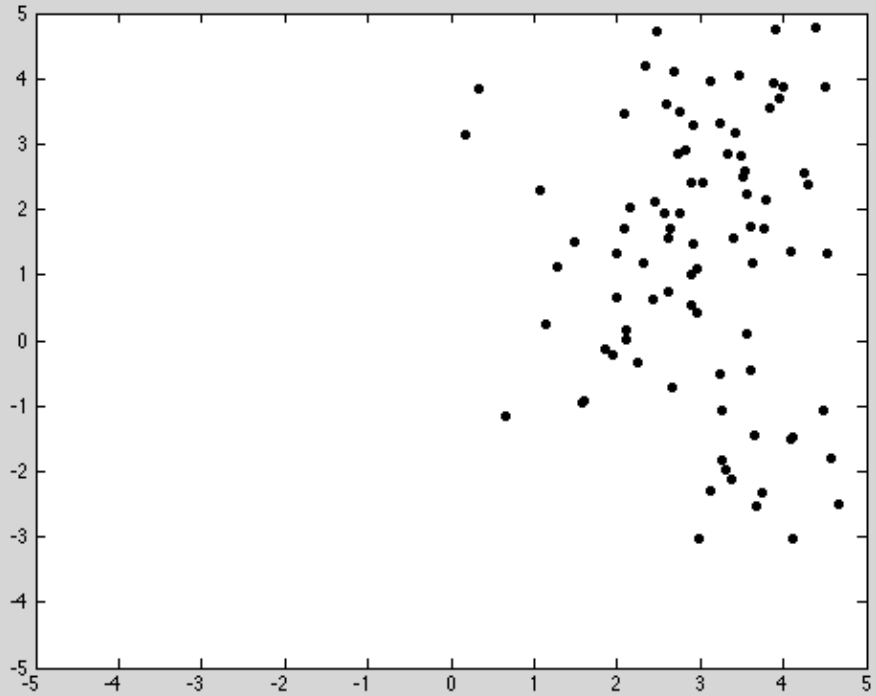
1. Metric in state space



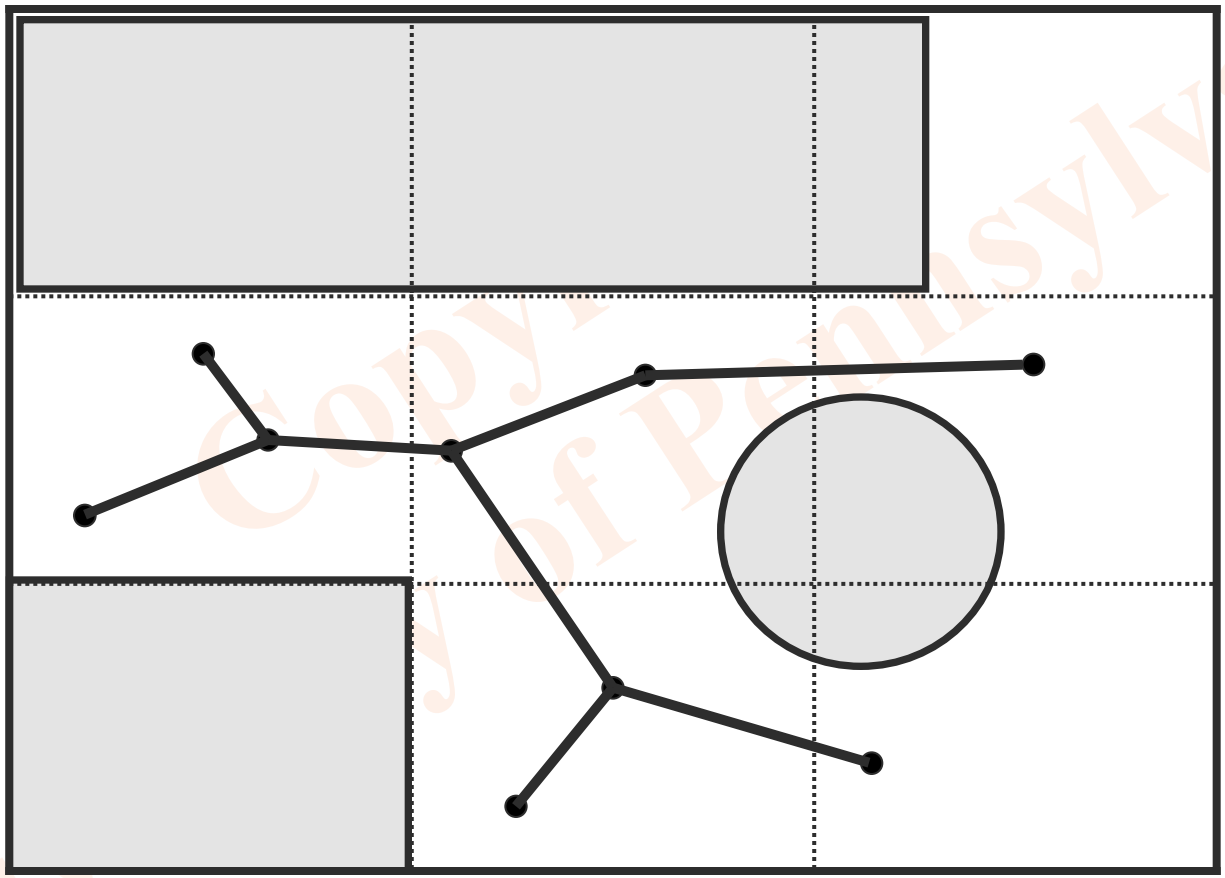
2. Sampling Strategy

Goal at (3, 2). Believe that there is a narrow corridor along the y-axis leading to the goal.

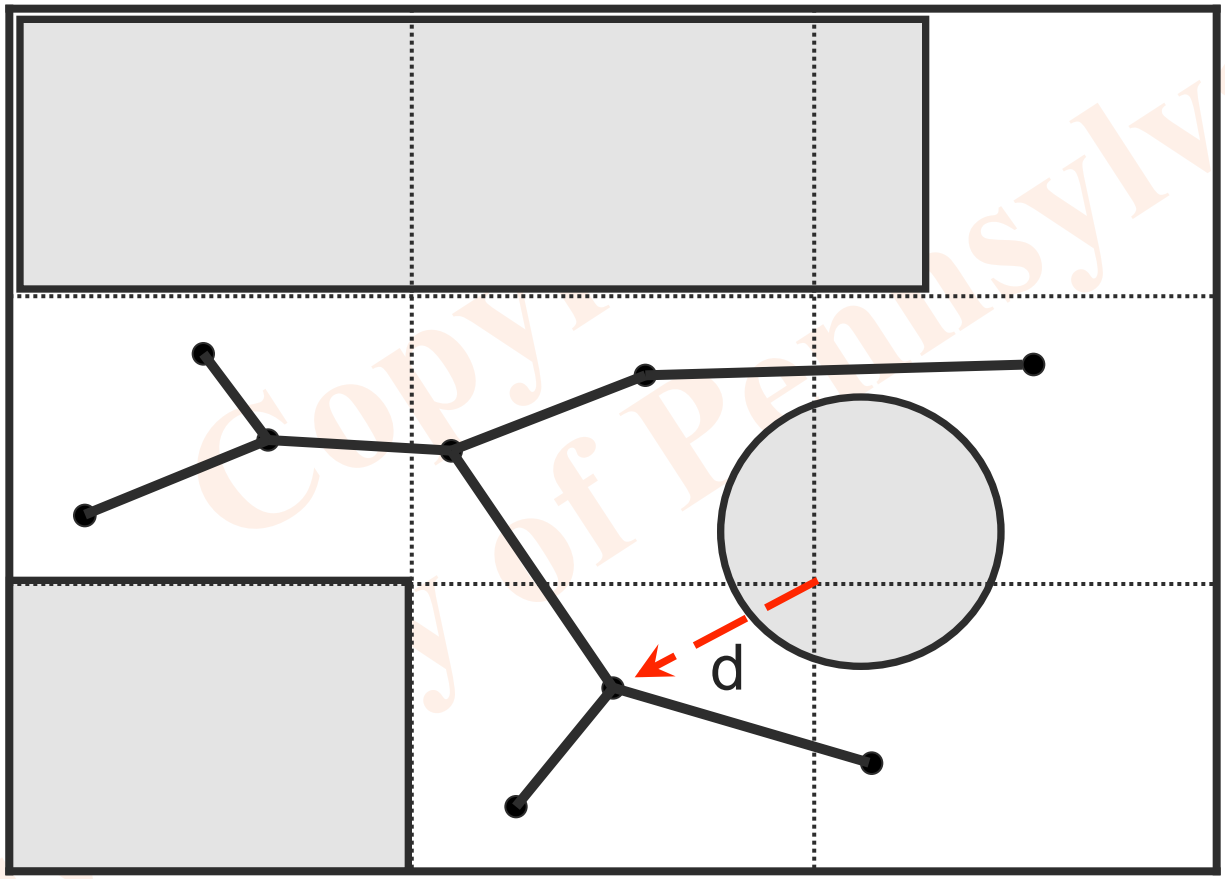




3. Measures of Coverage and Growth



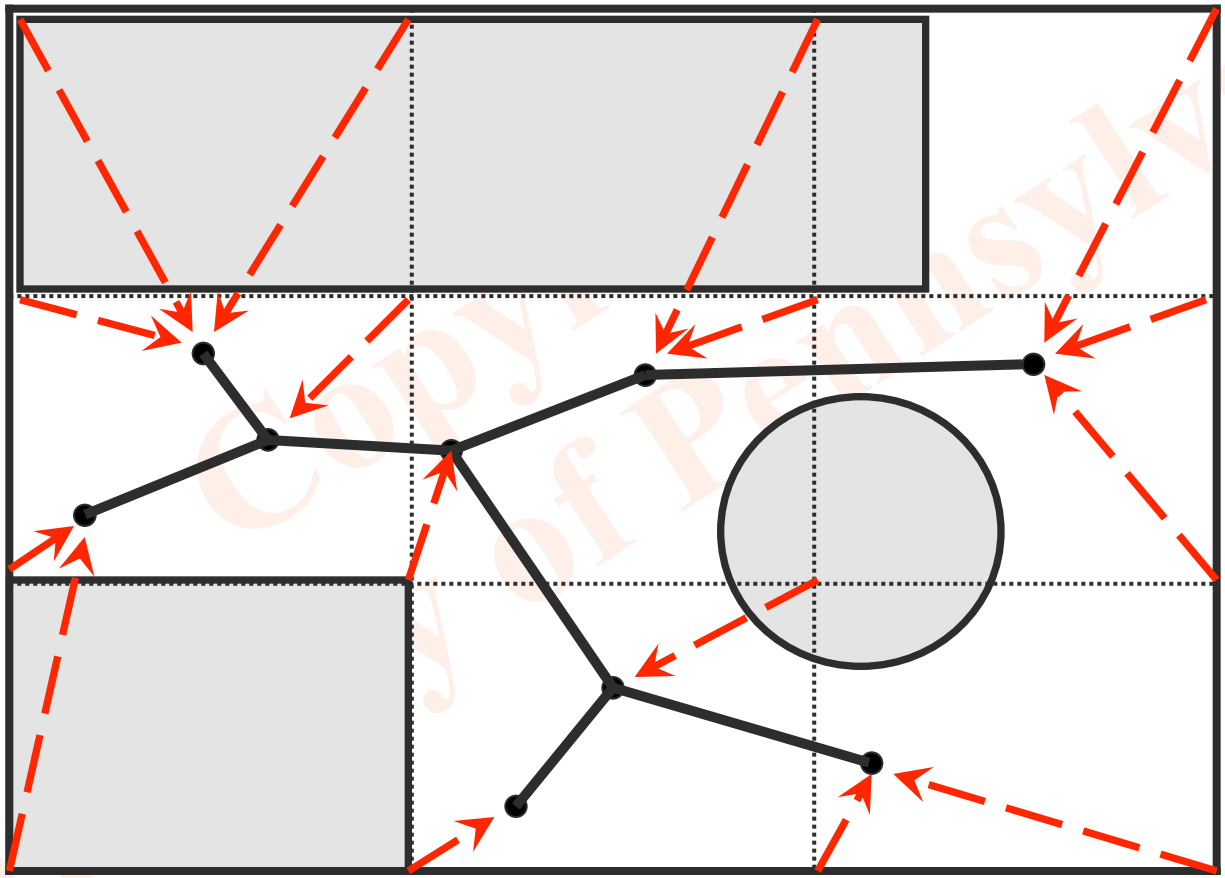
Distance from a grid point



n_g grid points

i nearest node to the *jth* grid point

d_{ij} distance of nearest node to the *jth* grid point



Measures of Coverage and Growth

Coverage

$$C(T) = \frac{1}{\delta} \sum_{j=1}^{n_g} \frac{\min(d_{ij}, \delta)}{n_g}$$

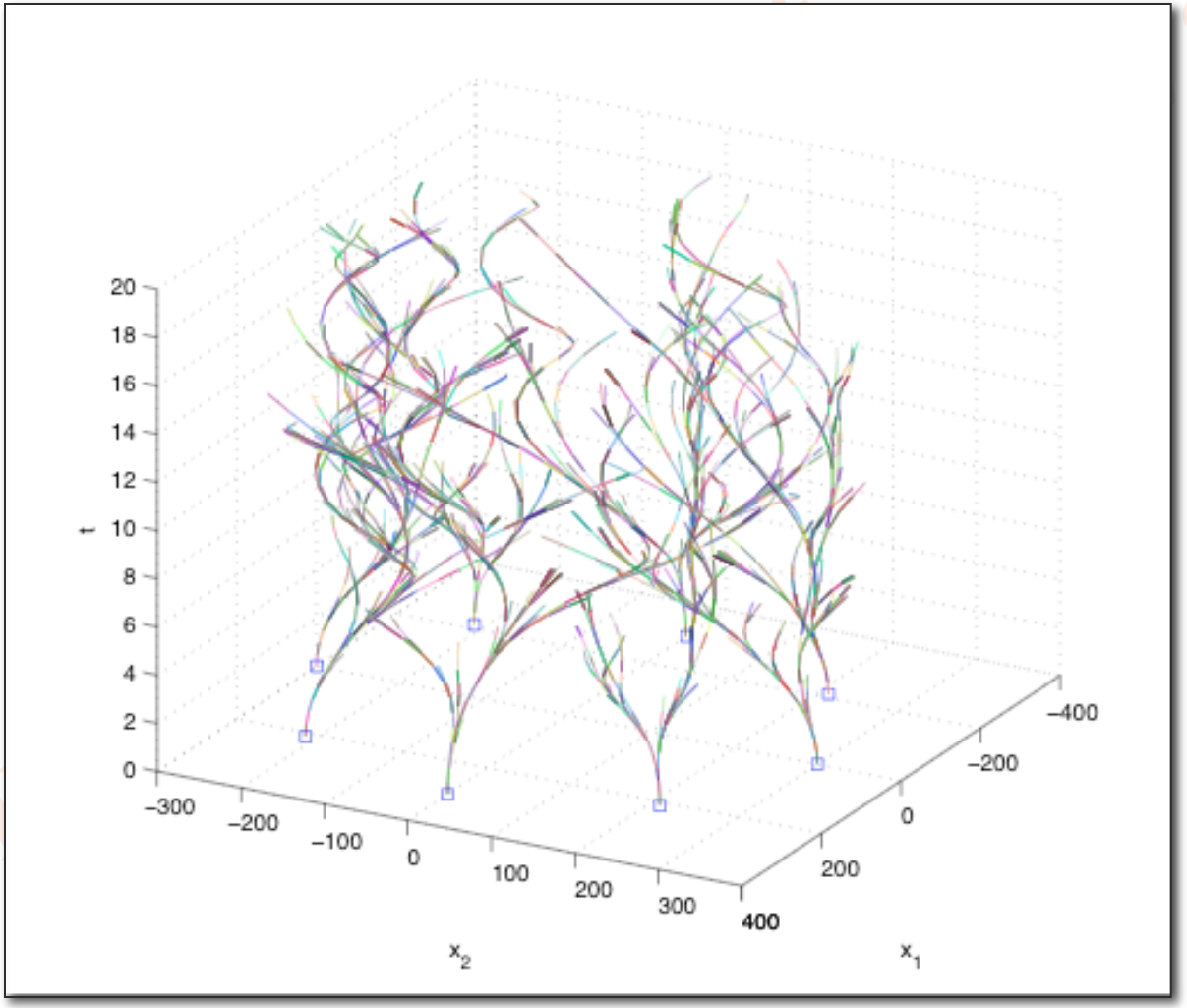
- Measure will monotonically decrease
- Asymptotically approach a constant

Growth rate

$$G(T) = -\frac{dC(T)}{dn_v} = -\frac{\Delta C(T)}{\Delta n_v}$$

- Always positive
- Eventually approaches 0

4. Can start from multiple points or from start and goal



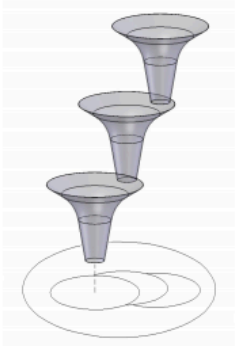
5. Can model noisy processes

LQR-Trees: Feedback Motion Planning on Sparse Randomized Trees

Russ Tedrake
Computer Science and Artificial Intelligence Lab
Massachusetts Institute of Technology
Cambridge, MA 02139
Email: russt@mit.edu

***Abstract*— Recent advances in the direct computation of Lyapunov functions using convex optimization make it possible to efficiently evaluate regions of stability for smooth nonlinear systems. Here we present a feedback motion planning algorithm which uses these results to efficiently combine locally valid linear quadratic regulator (LQR) controllers into a nonlinear feedback policy which probabilistically covers the reachable area of a (bounded) state space with a region of stability, certifying that all initial conditions that are capable of reaching the goal will stabilize to the goal. We investigate the properties of this systematic nonlinear feedback control design algorithm on simple underactuated systems and discuss the potential for control of more complicated control problems like bipedal walking.**

I. INTRODUCTION



LQG-MP: Optimized Path Planning for Robots with Motion Uncertainty and Imperfect State Information

Jur van den Berg
University of North Carolina at Chapel Hill
E-mail: berg@cs.unc.edu

Pieter Abbeel
University of California, Berkeley
E-mail: pabbeel@cs.berkeley.edu

Ken Goldberg
University of California, Berkeley
E-mail: goldberg@berkeley.edu

October 27, 2010

Comparison

	Exact Cell Decomposition	Approx Cell Decomposition	Roadmap methods	Probabilistic methods	Potential field methods
Practical above 2 or 3 dimensions	Very slow, memory	Slow, memory (m^N requirement*)	Very slow	Fast	Fast
Practical above 5 dimensions	No	Perhaps not, not clear	No	Yes	Yes
Optimality	Yes	Yes, up to resolution	Yes	Probabilistic guarantees	No
Easy to implement	No	Yes	No	Yes	Yes
Completeness	Yes	Yes, up to resolution	Yes	Probabilistically complete	No**

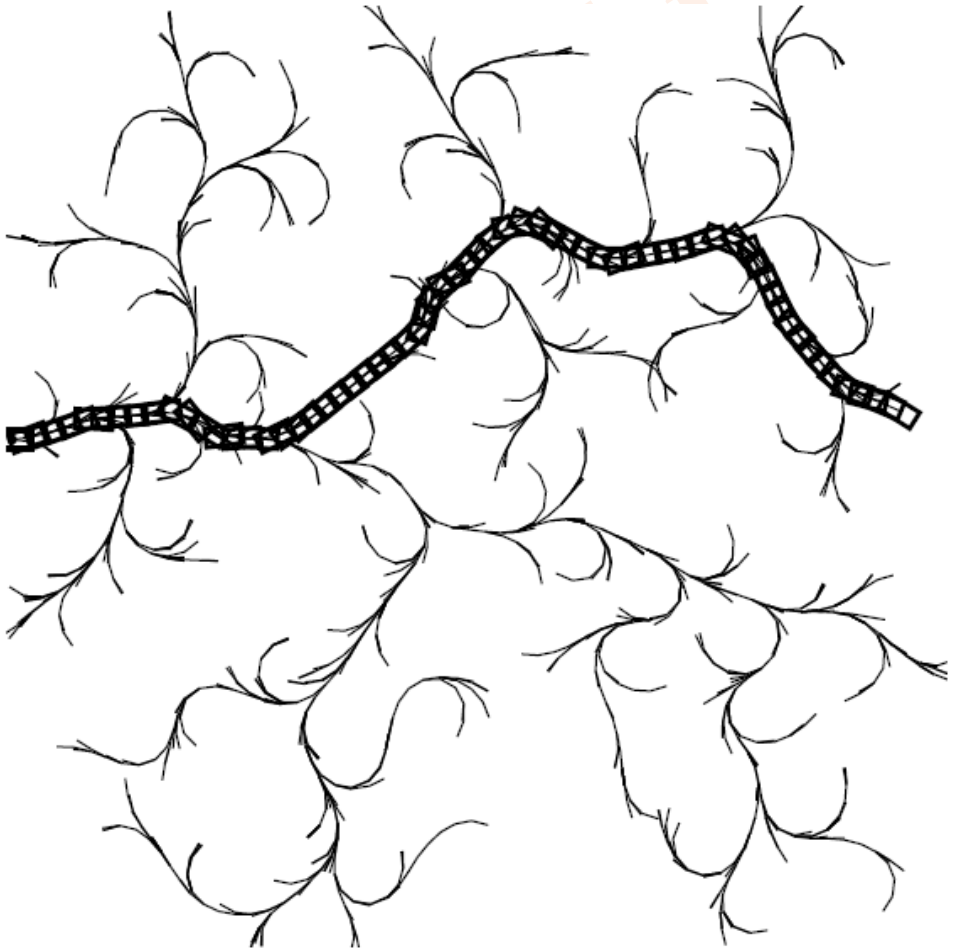
If there is a solution path, the algorithm will find it with high probability

**m* grid cells along each of *n* axes

***Navigation functions in certain classes of environments provide theoretical guarantees

Other Trajectory Generation Methods

Rapidly exploring Random Trees (RRT)

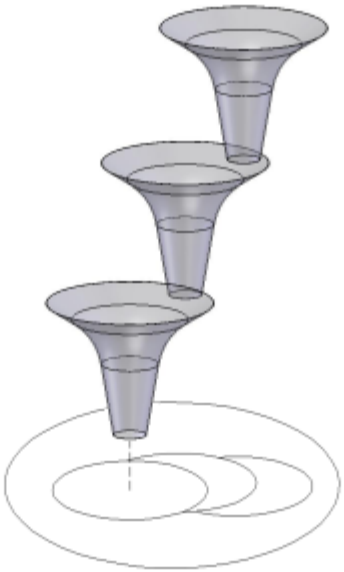


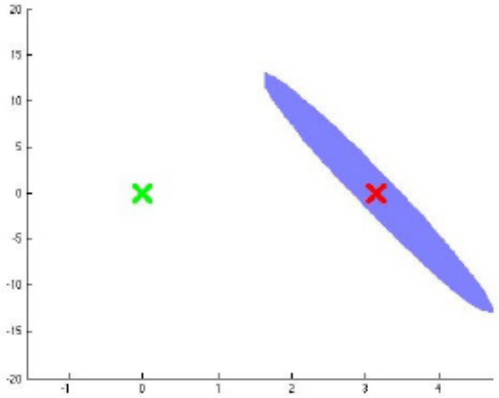
[LaValle, 1998]

Other Trajectory Generation Methods

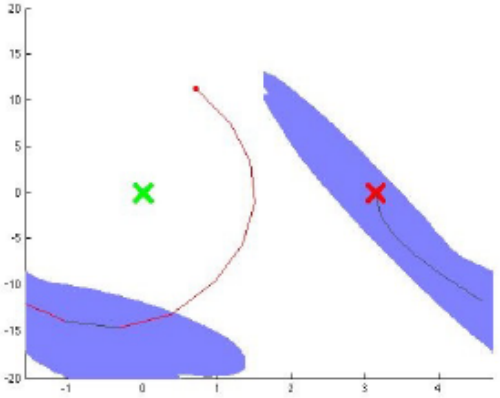
Linear Quadratic Regulator Trees (LQR T)

- Pendulum Example

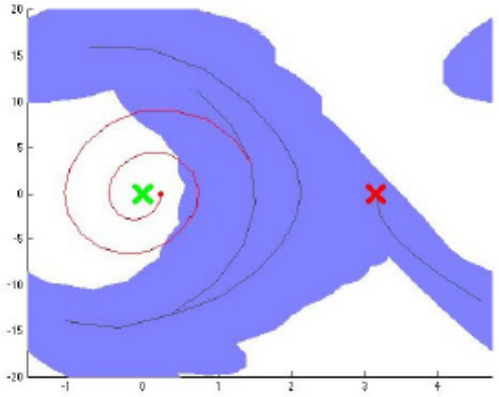




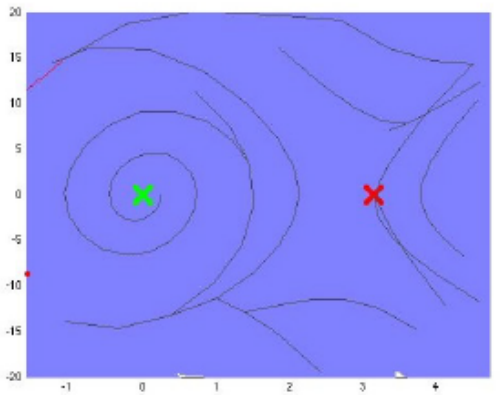
(a) 1 node



(b) 8 nodes



(c) 24 nodes

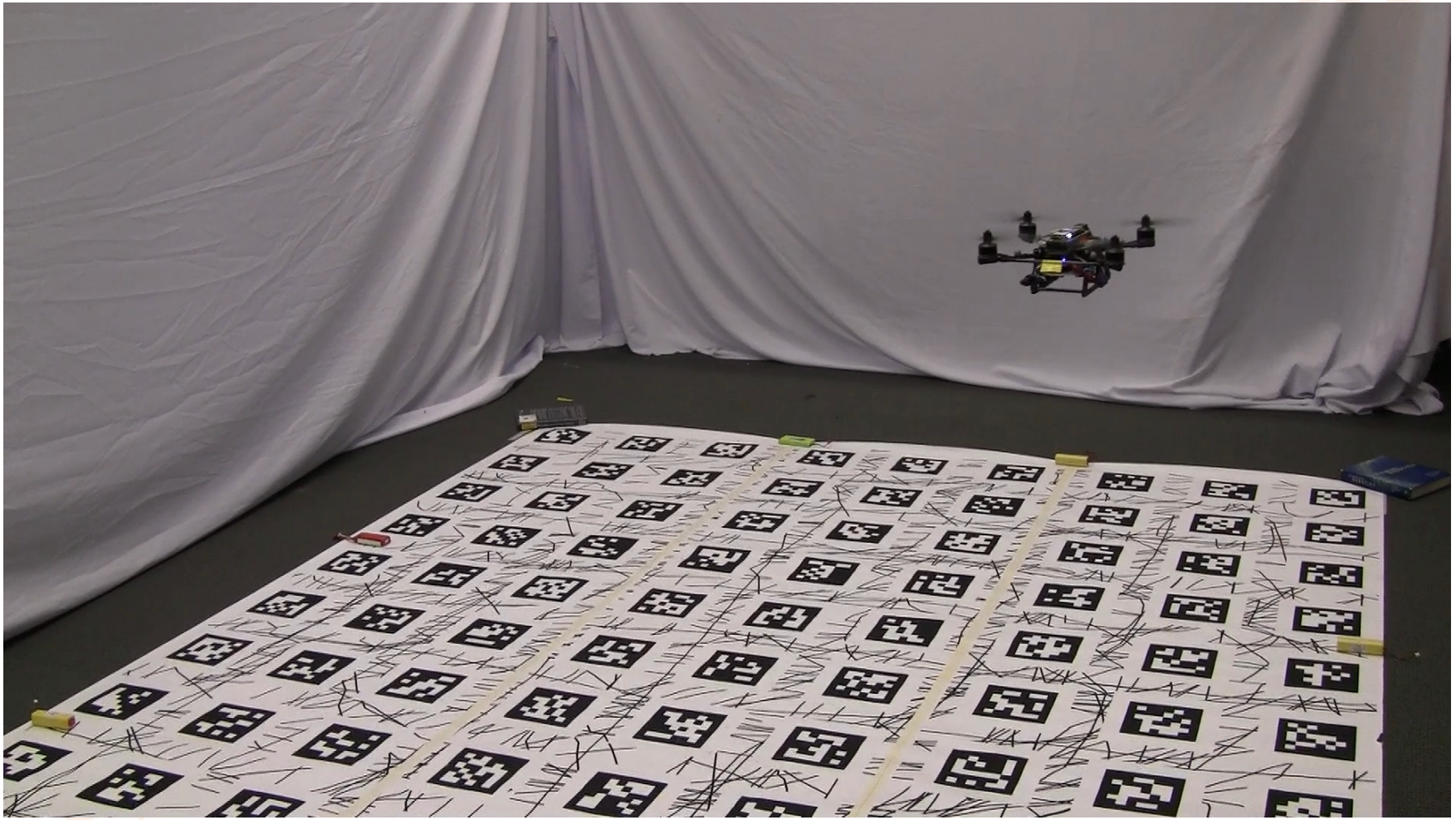


(d) 104 nodes

[Tedrake, 2009]

Filtering

Motivation



Basic Probability Theory

Probability Distribution

$$f_X(x) : \mathbb{R}^n \rightarrow \mathbb{R}$$

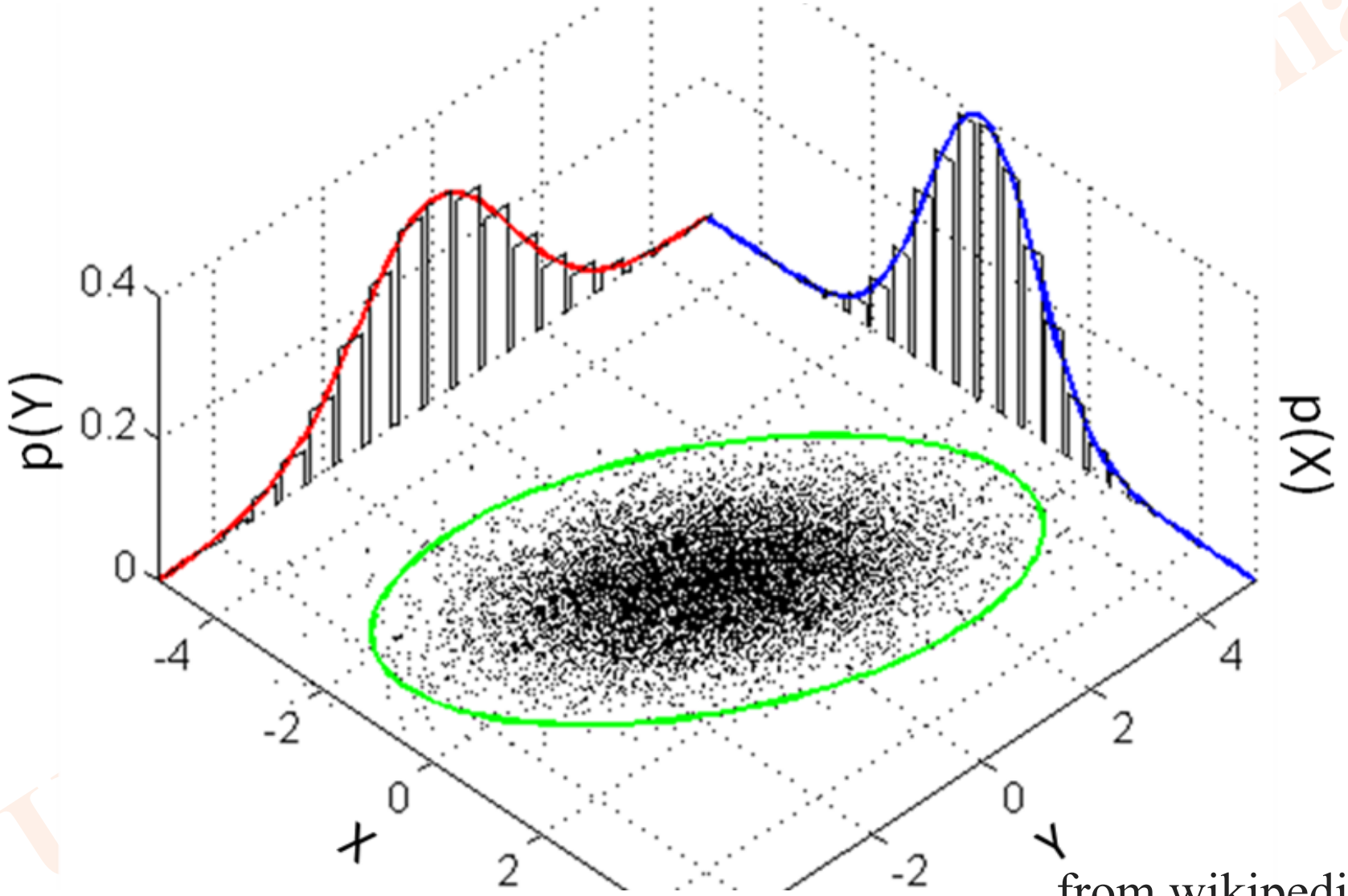
Bayes Rule

$$p(x|y) = \frac{p(x)p(y|x)}{p(y)}$$

Joint Normal Distribution

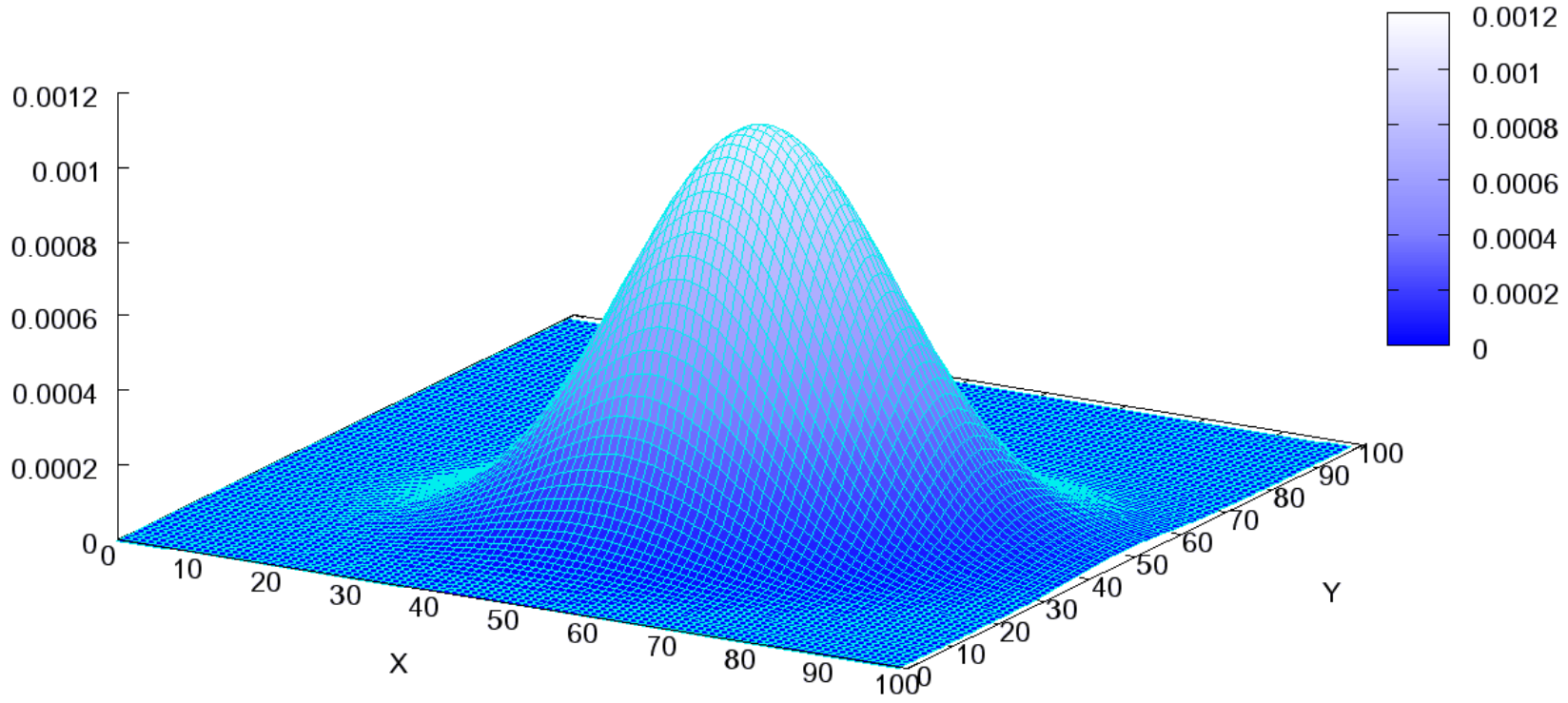
$$f_X(x) = \frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{\det \Sigma}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

Joint Normal Distribution



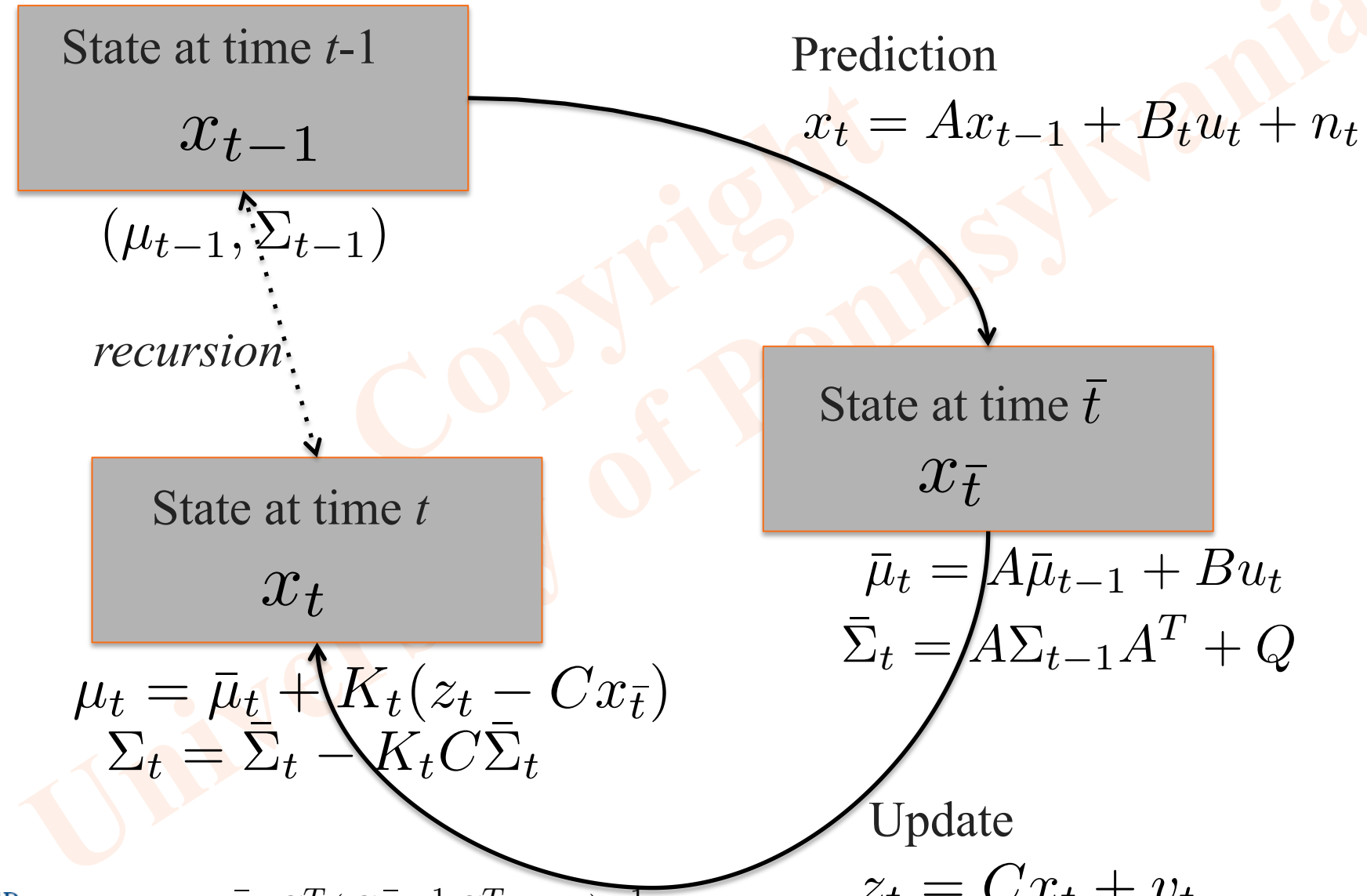
from wikipedia.org

Gaussian Noise Model



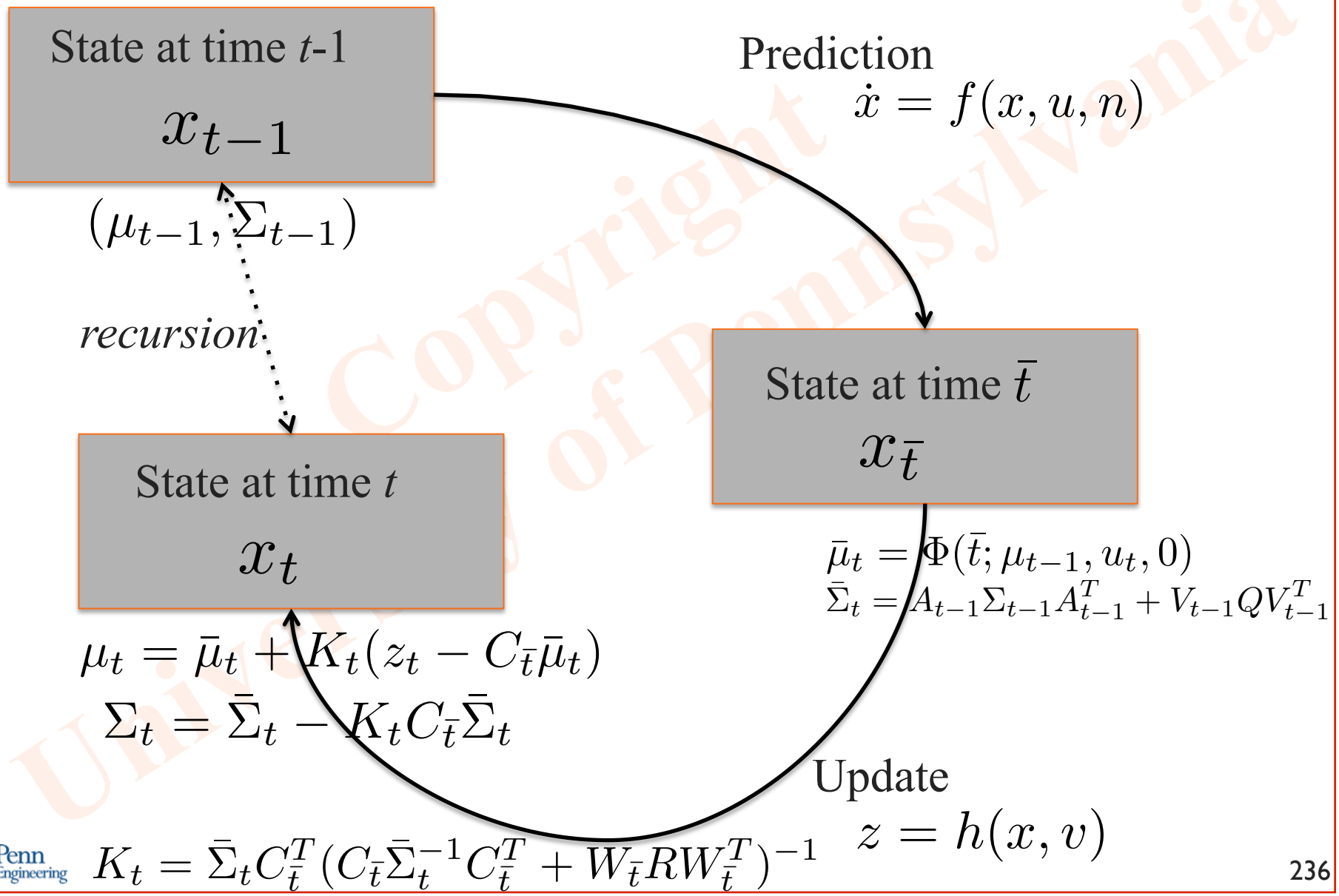
from wikipedia.org

Kalman Filter



Extended Kalman Filter

Extended Kalman Filter



Quadrotor Model

ate Vector

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \\ \dot{\mathbf{p}} \\ \mathbf{b}_g \\ \mathbf{b}_a \end{bmatrix}$$

Kinematics

$$R(\mathbf{q}) = \begin{bmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} c\theta & 0 & -c\phi s\theta \\ 0 & 1 & s\phi \\ s\theta & 0 & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

ω $\quad$ $G(\mathbf{q})$ $\quad$ $\dot{\mathbf{q}}$

Body frame angular velocity

yro Measurements

$$\omega_m = \omega + \mathbf{b}_g + \mathbf{n}_g$$

Vision measurements

$$\mathbf{z} = \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \\ \dot{\mathbf{p}} \end{bmatrix}$$

Accelerometer Measurements

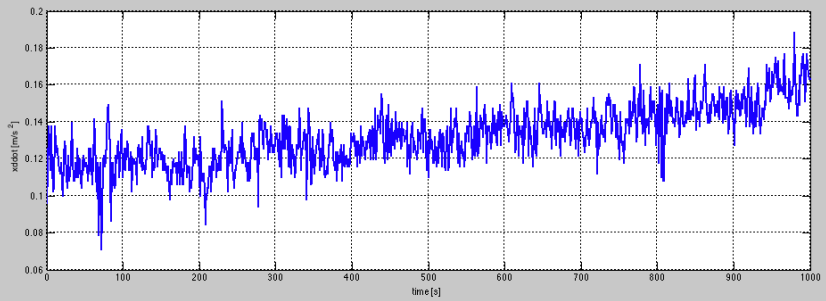
$$\mathbf{a}_m = R(\mathbf{q})^T(\ddot{\mathbf{p}} - \mathbf{g}) + \mathbf{b}_a + \mathbf{n}_a$$

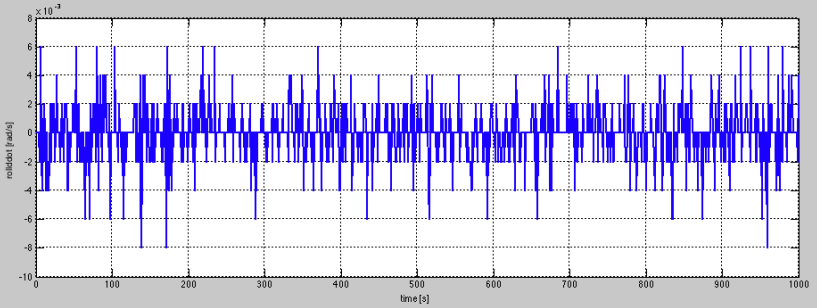
Accelerometer Measurements

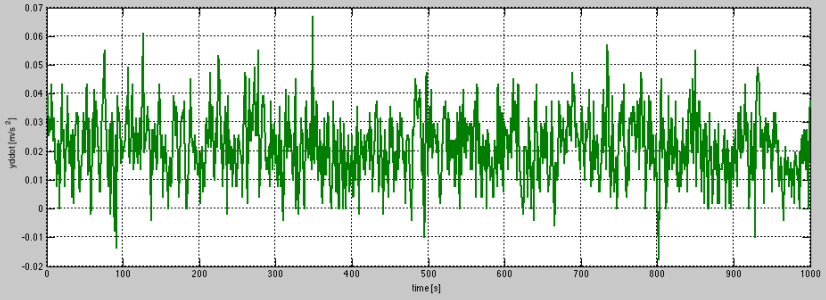
$$\mathbf{a}_m = R(\mathbf{q})^T(\ddot{\mathbf{p}} - \mathbf{g}) + \mathbf{b}_a + \mathbf{n}_a$$

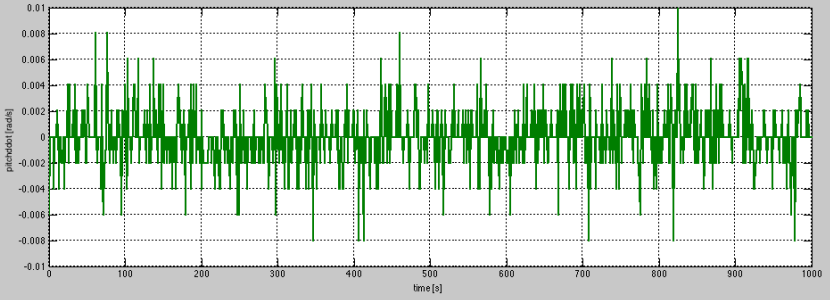
Gyro Measurements

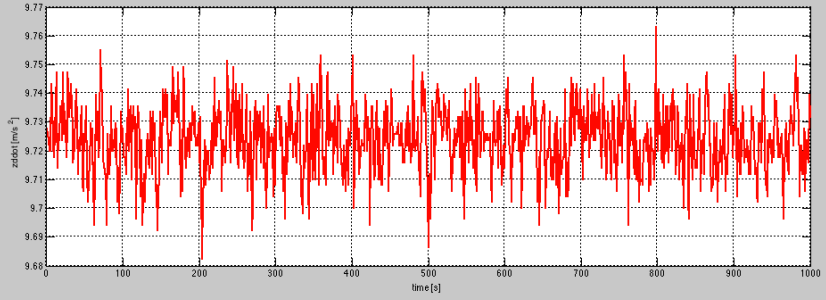
$$\omega_m = \omega + \mathbf{b}_g + \mathbf{n}_g$$

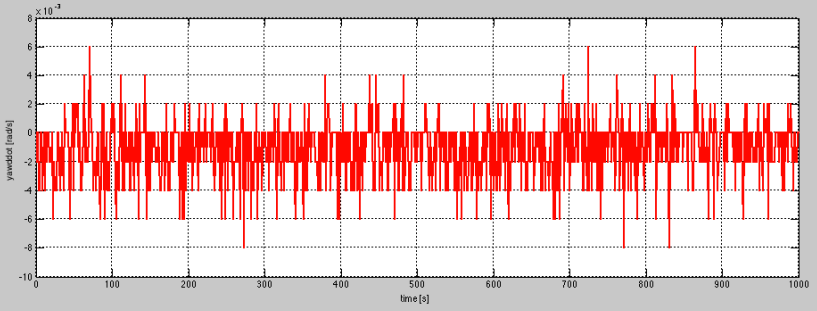












IDETC/CIE T3
Kinematics, Dynamics, and Control of Quadrotors

Vijay Kumar

Emails:

Justin Thomas

Mickey Whitzer

Yash Mulgaonkar

{jut, mwhitzer, yashm}@seas.upenn.edu