Estimation and Control for Autonomous Agents

Analytic and learning-inspired methods enabling real-time operations

Luis Rodolfo GARCIA CARRILLO

Unmanned Systems Laboratory
Texas A&M University - Corpus Christi

February 20, 2020

What can we do with Unmanned Agents?

Monitoring traffic, and hazardous disaster areas

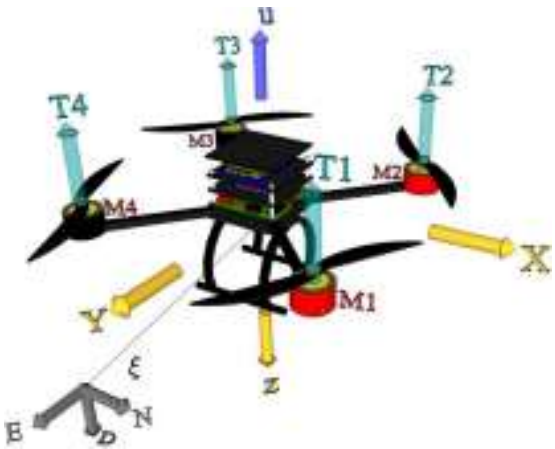
Search and rescue operations, and package (goods) delivery

Things we (humans) can do, but unmanned agents are more reliable and efficient. . . if they are **effectively controlled**

Motivation

Requirement(s) for **effective control** of a robotic agent?

- **accurate model** of that system



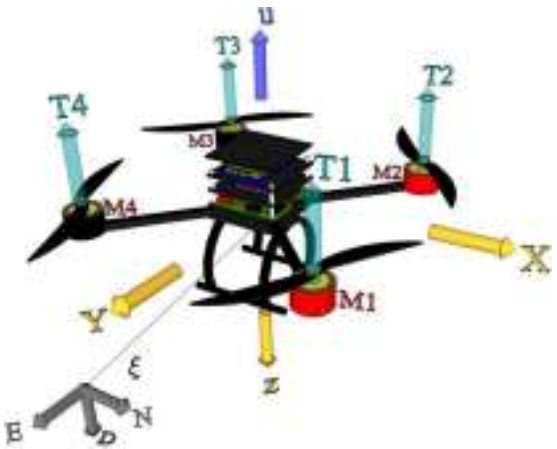
Dynamic model

$$\dot{\xi} = v$$
$$m\dot{v} = m\bar{g}e_3 + \mathcal{R}F$$
$$\dot{\mathcal{R}} = \mathcal{R}\mathrm{sk}(\Omega)$$
$$\mathbb{I}\dot{\Omega} = -\Omega \times \mathbb{I}\Omega + \Gamma$$

Motivation

Requirement(s) for **effective control** of a robotic agent?

- **accurate model** of that system



Dynamic model

$$\dot{\xi} = v$$
$$m\dot{v} = m\bar{g}e_3 + \mathcal{R}F$$
$$\dot{\mathcal{R}} = \mathcal{R}\mathrm{sk}(\Omega)$$
$$\mathbb{I}\dot{\Omega} = -\Omega \times \mathbb{I}\Omega + \Gamma$$

Simplified models are often used

$$m\ddot{x} = -u(\cos(\psi)\sin(\theta)\cos(\phi) + \sin(\psi)\sin(\phi))$$
$$m\ddot{y} = -u(\sin(\psi)\sin(\theta)\cos(\phi) - \cos(\psi)\sin(\phi))$$
$$m\ddot{z} = -u(\cos(\theta)\cos(\phi)) + mg$$

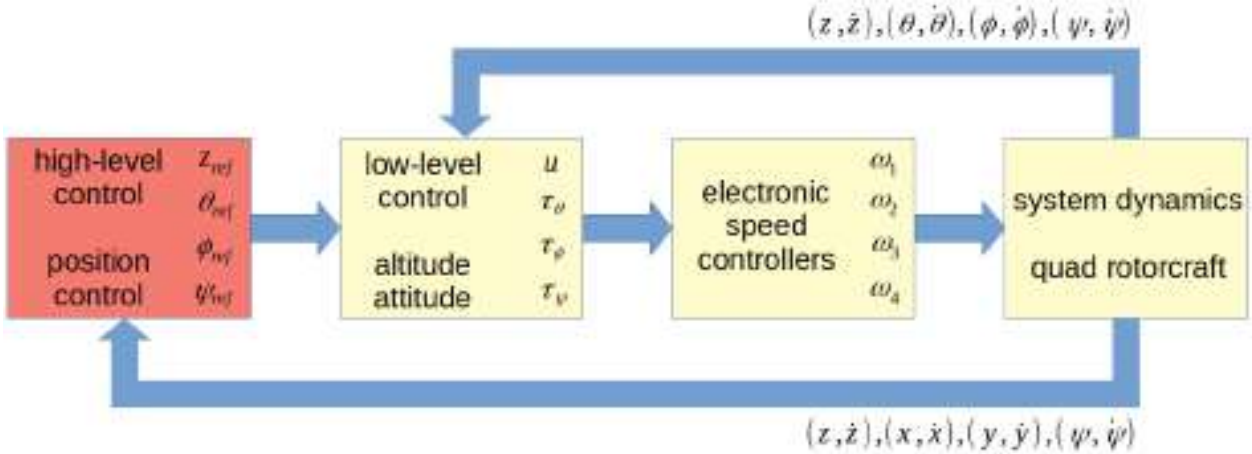
$$\ddot{\theta} = \tilde{\tau}_\theta$$
$$\ddot{\phi} = \tilde{\tau}_\phi$$
$$\ddot{\psi} = \tilde{\tau}_\psi$$

Not as accurate, but captures **critical aspects** of the system

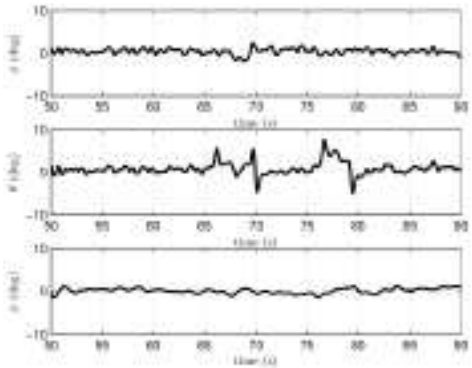
Motivation

Quad rotorcraft controller well suited for **real-time implementation**

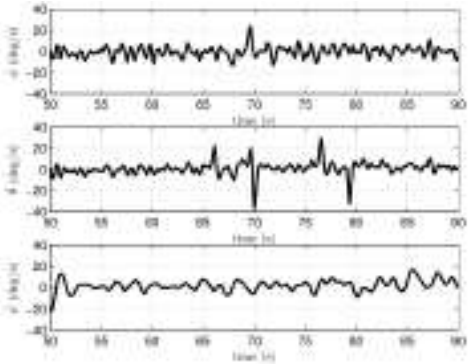
- inner-loop control: attitude and altitude
- **outer-loop control**: translational dynamics



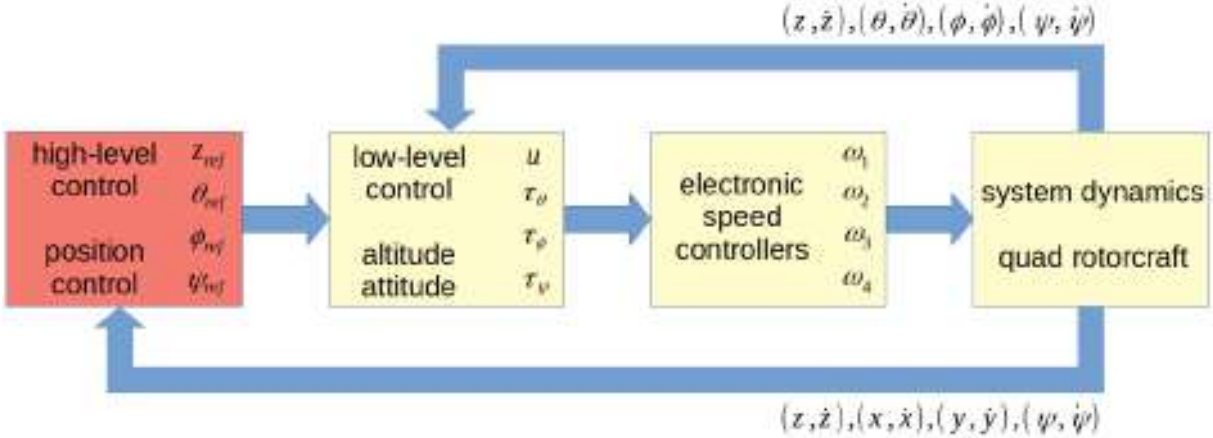
Attitude



Angular rates



Motivation



Outer-loop control requires proper/accurate information in order to enable the robot to efficiently execute missions, unfortunately

- **vision-based sensors:** exhibit large errors when a visual landmark is temporarily obstructed or misinterpreted
- **RF/acoustic sensors:** prone to reporting false measurements due to multi-path reflections
- **Global Positioning System (GPS):** suffer from spoofing and jamming threats, as well as lack of coverage
- **unknown/uncertain parameters** in the model of the agent

Advanced estimation and control techniques are needed in order to overcome these challenges

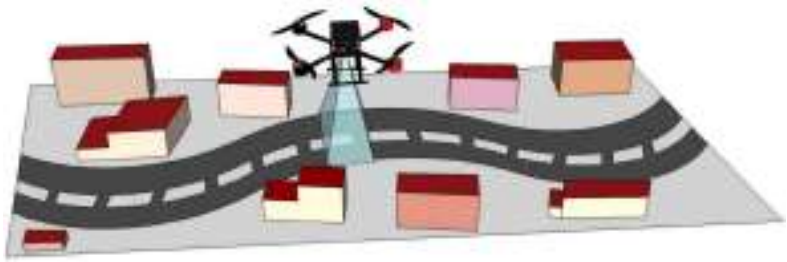
Table of contents

Robust trajectory tracking: a switched systems approach

Problem statement: autonomous trajectory tracking

How to perform an autonomous trajectory tracking mission?

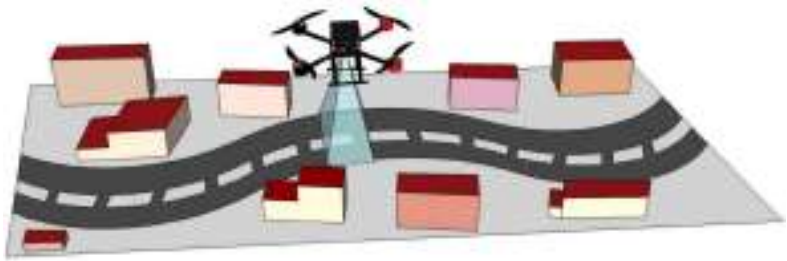
- take-off: altitude $z \to z_d$
- heading angle: $B_x \to R_x$
- forward speed: $\dot{x}$
- lateral error ≈ 0
- landing: $z \to 0$



Problem statement: autonomous trajectory tracking

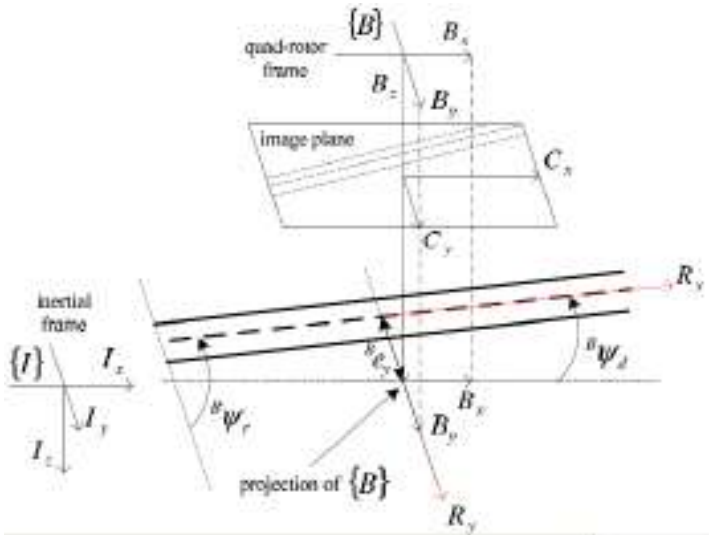
How to perform an autonomous trajectory tracking mission?

- take-off: altitude $z \to z_d$
- heading angle: $B_x \to R_x$
- forward speed: $\dot{x}$
- lateral error ≈ 0
- landing: $z \to 0$



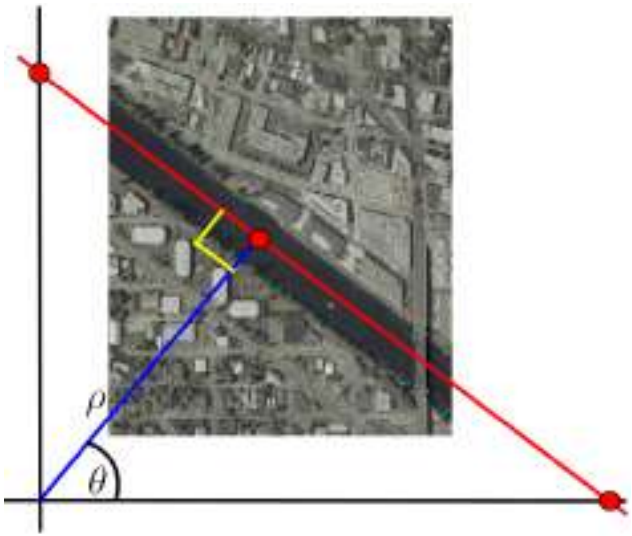
Possible solution: use data coming from complementary sensors

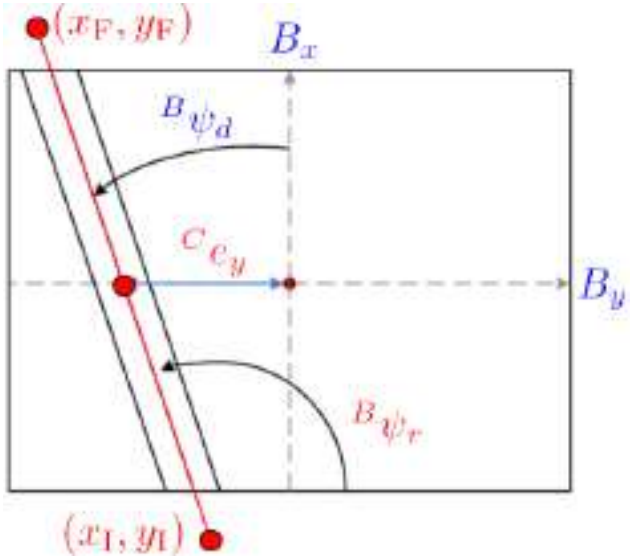
- imaging sensor (camera): detect the road
- ultrasonic sensor: altitude
- inertial measurement unit (IMU): attitude
- camera, ultrasonic, IMU: optical flow (velocity)



Desired trajectory: detection of road

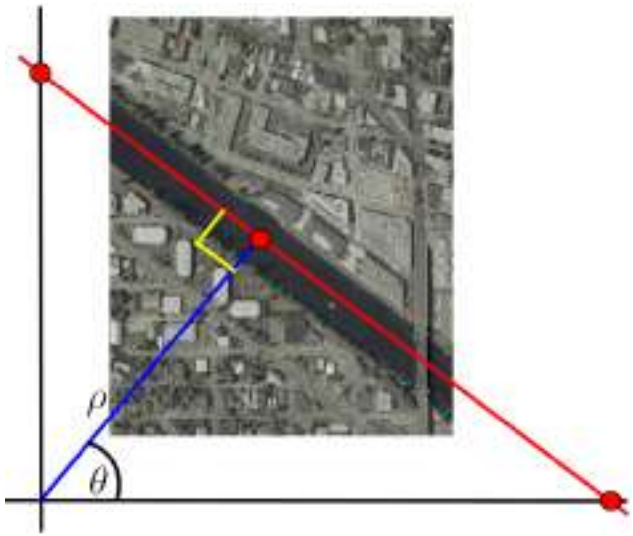
Based on optimized openCV **Hough's Transform**

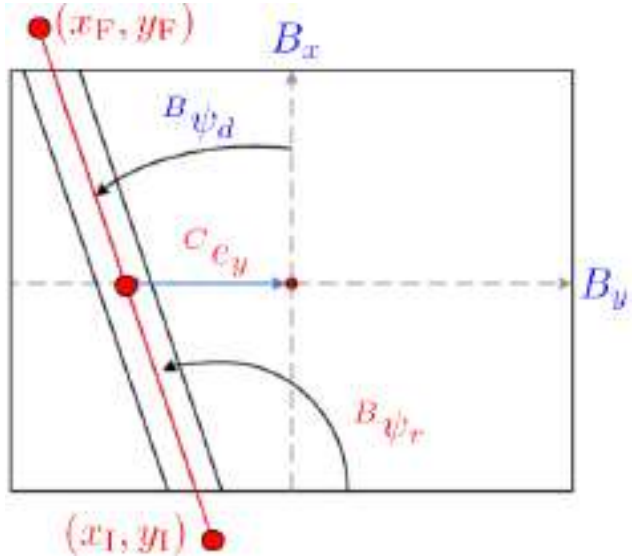




Desired trajectory: detection of road

Based on optimized openCV **Hough's Transform**





Vehicle's desired heading angle

$$ {}^B\psi_r = \arctan(y_F - y_I, x_F - x_I) $$

$$ {}^B\psi_d = {}^B\psi_r - \frac{\pi}{2} $$

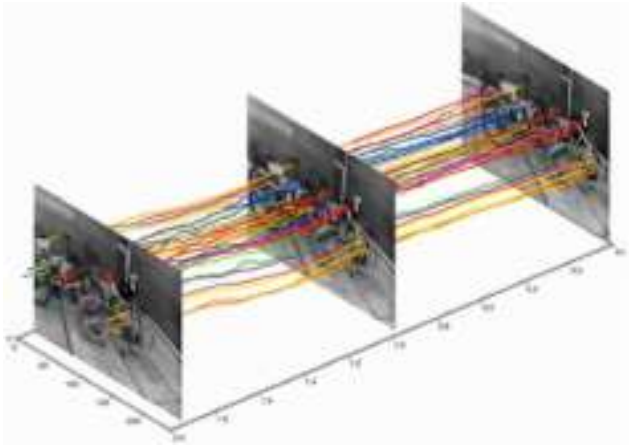
Relative lateral position

$$ {}^C e_y = \left(\frac{x_I - x_F}{2} + x_F \right) - \frac{C_W}{2} $$

$$ e_y = z \frac{{}^C e_y}{\alpha_y} $$

Translational Velocities: optical flow–based approach

Optical Flow based on optimized openCV **Good Features to Track** and **Lucas-Kanade feature tracker** functions

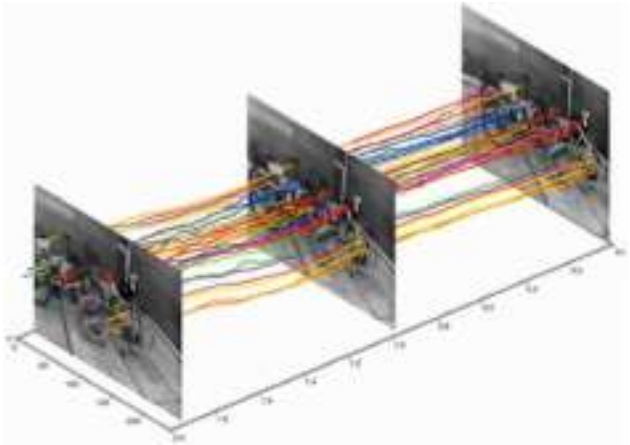


Challenge: flow can be generated by

- Translational Motion
- Rotational Motion
- Combination of both

Translational Velocities: optical flow–based approach

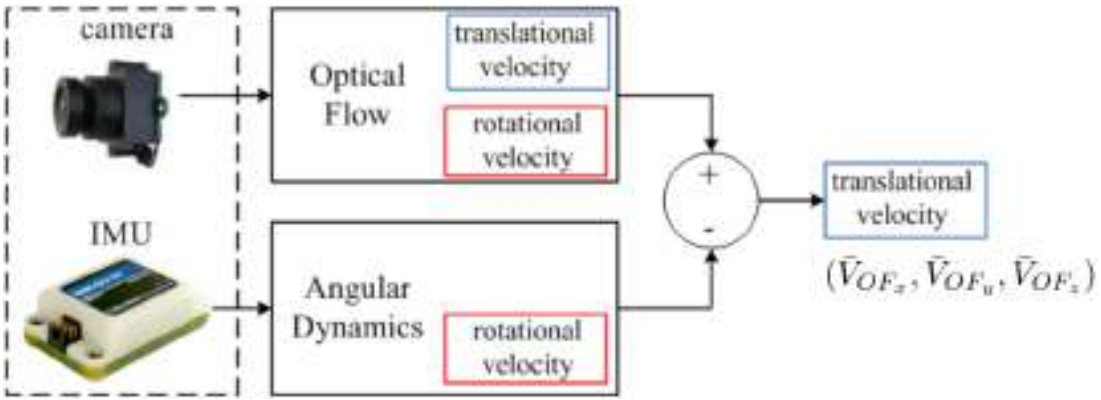
Optical Flow based on optimized openCV **Good Features to Track** and **Lucas-Kanade feature tracker** functions



Challenge: flow can be generated by

- Translational Motion
- Rotational Motion
- Combination of both

Complementary filter to compensate rotational optical flow



$$-z\frac{\bar{V}_{OF_x}}{\alpha_x} = \dot{x}$$

$$-z\frac{\bar{V}_{OF_y}}{\alpha_y} = \dot{y}$$

$$z\bar{V}_{OF_z} = \dot{z}$$

Challenge: proposed technique is not 100% error-free

Desired heading angle (${}^{B}\psi_d$) and lateral position (e_y) depend on image processing techniques (Hough's Transform)

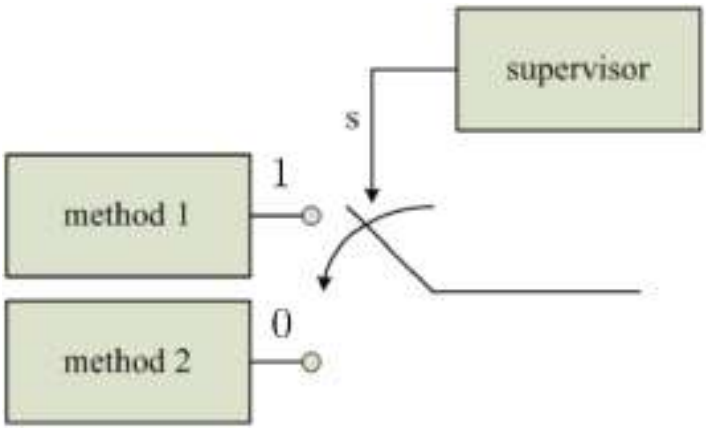
- data is not generated if trajectory is not detected/outside of FOV

Challenge: proposed technique is not 100% error-free

Desired heading angle (${}^B\psi_d$) and lateral position (e_y) depend on image processing techniques (Hough's Transform)

- data is not generated if trajectory is not detected/outside of FOV

Proposed solution: switching between estimation and control strategies



Define a switching signal

$$s : [0, \infty) \to \{0, 1\}$$

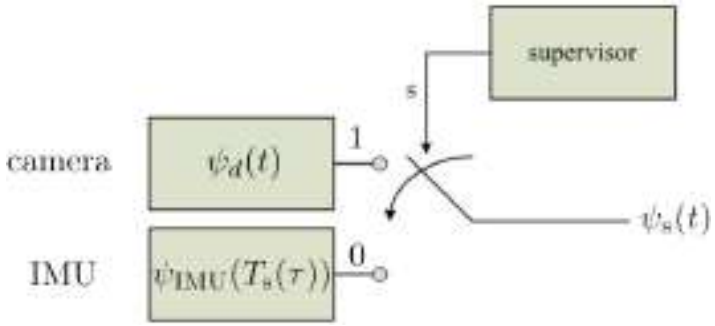
$$s(t) := \begin{cases} 0 & \text{trajectory not detected at time } t \\ 1 & \text{camera detects trajectory at time } t \end{cases}$$

Define also a time interval

$$T_s(\tau, t) : \text{ amount of time that } s = 0$$

Proposed switching estimation and control techniques

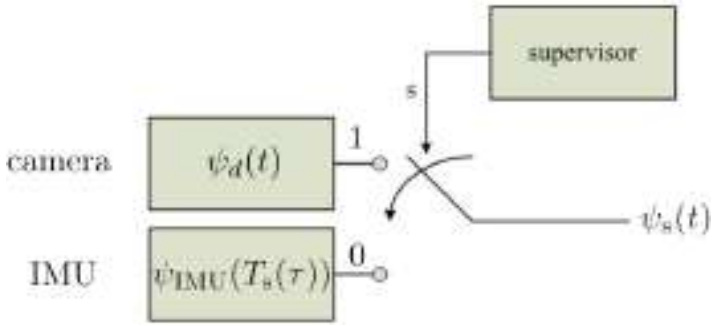
Switching strategy for estimation of desired **heading angle**



$$\psi_s(t) = s(t)\psi_d(t) + (1 - s(t))\psi_{\text{IMU}}(T_s(\tau))$$

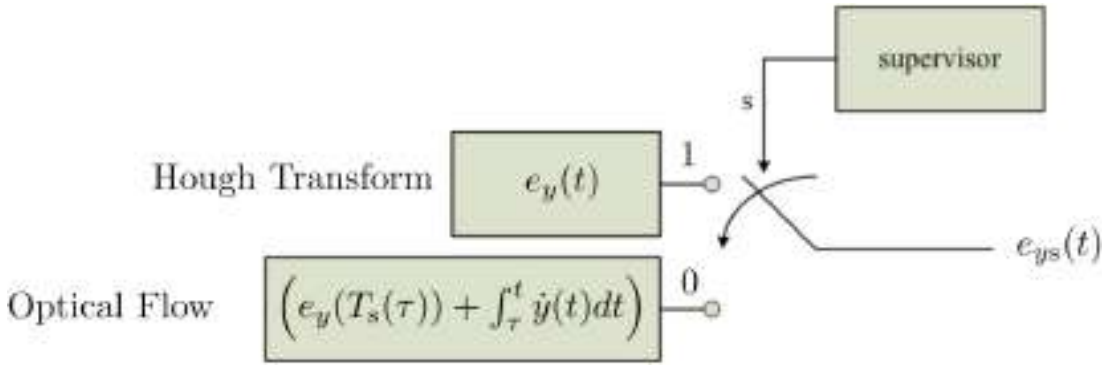
Proposed switching estimation and control techniques

Switching strategy for estimation of desired **heading angle**



$$\psi_s(t) = s(t)\psi_d(t) + (1 - s(t))\psi_{\text{IMU}}(T_s(\tau))$$

Switching strategy for **lateral position** estimation



$$e_{ys}(t) = s(t)e_y(t) + (1 - s(t))\left(e_y(T_s(\tau)) + \int_\tau^t \dot{y}(t)dt\right)$$

Proposed switching estimation and control techniques

Translational displacement: subactuated control strategy approach

- orientate thrust vector towards desired x and y displacement

How? use θ_d and ψ_d as virtual controllers for translational dynamics

Altitude Dynamics

$$u = k_{pz}(z - z_d) + k_{vz}(\dot{z} - \dot{z}_d) + mg$$

Longitudinal Displacement

$$\theta_d = \frac{k_{px}(x - x_d) + k_{vx}(\dot{x} - \dot{x}_d)}{u}$$

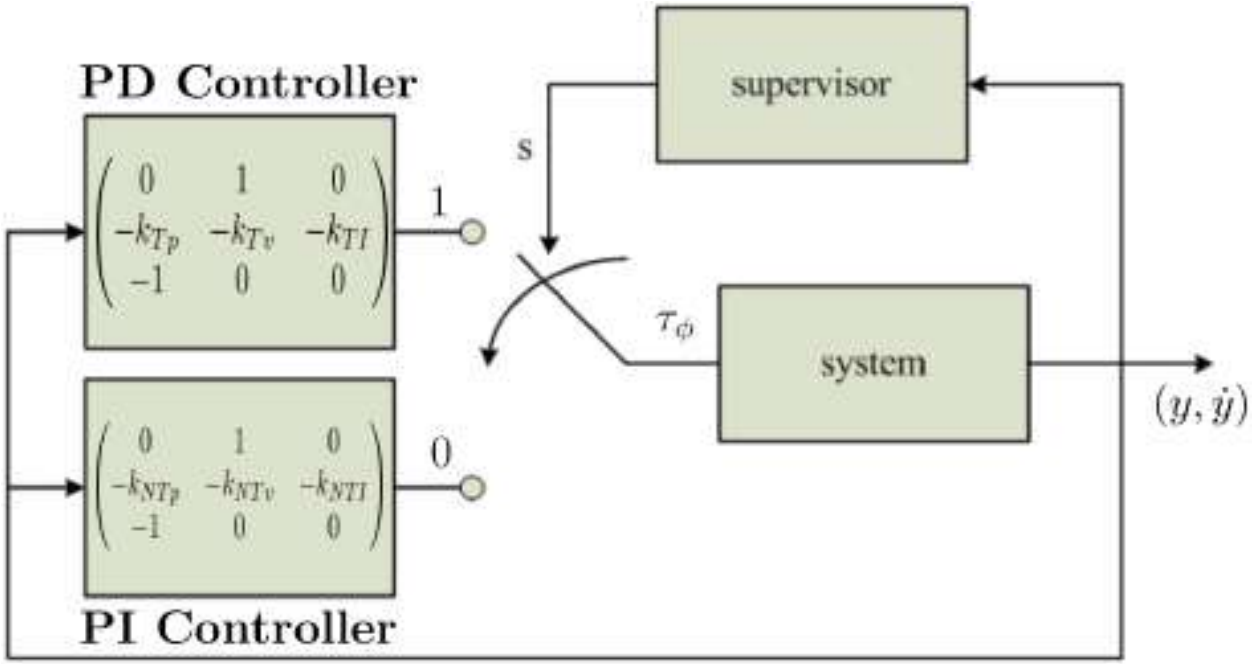
Lateral Displacement

$$\phi_d = \frac{-k_{\mathcal{P}_p} y - k_{\mathcal{P}_v} \dot{y} - k_{\mathcal{P}_I} \int_0^t \xi_y(\tau) d\tau}{u}$$

Lateral dynamics' error

$$\dot{e}_y = A_{\mathcal{P}} e_y, \text{ with } e_y = (y, \dot{y}, \xi_y) \text{ and } \mathcal{P} \in \{T, NT\}$$

Proposed switching estimation and control techniques



$$\tau_\phi = \begin{cases} A_T e_y & \text{if } s = 1 \\ A_{NT} e_y & \text{if } s = 0 \end{cases}$$

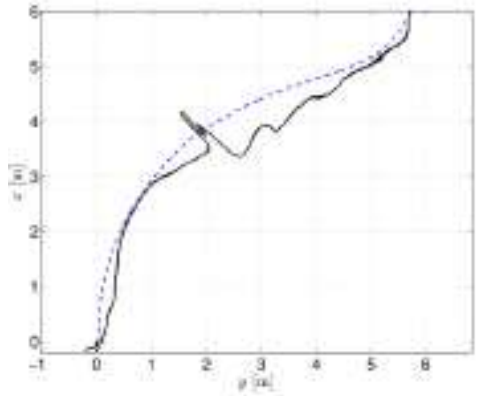
Switched systems stability theorem [Liberzon, 2003]

- If A_T and A_{NT} are Hurwitz, trajectory of the systems tends to zero exponentially

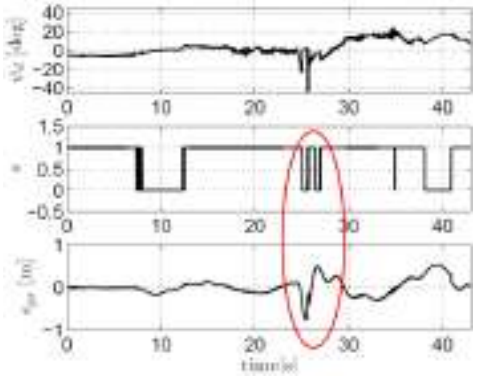
Experimental Application

Video of Experiment

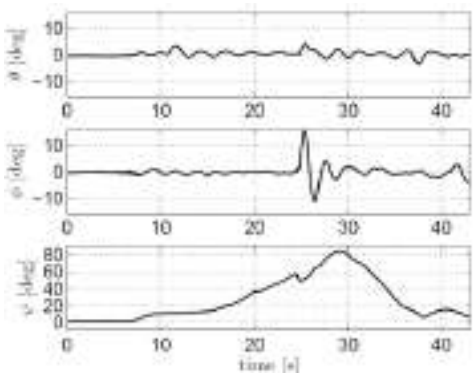
Trajectories



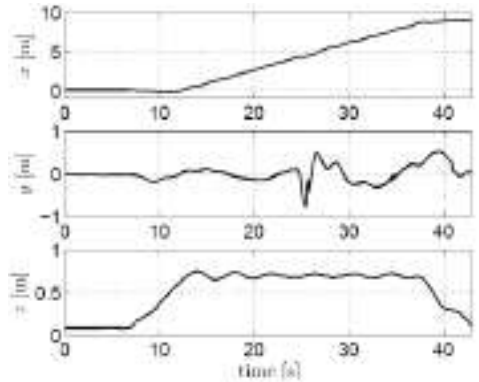
Switching Signal



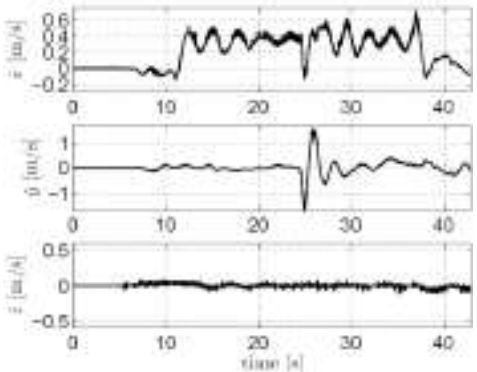
Attitude Angles



Translational States



Velocities



To know more

[1] **L.R. Garcia Carrillo**, A. Dzul, R. Lozano and C. Pegard. "Quad Rotorcraft Control: Vision- Based Hovering and Navigation". Springer, ISBN-10: 1447143981, September 2012.

[2] **L.R. Garcia Carrillo**, I. Fantoni, E. Rondon, and A. Dzul. "3-dimensional Position and Velocity Regulation of a Quad-rotor Using Optical Flow". IEEE Transactions on Aerospace and Electronic Systems, vol. 52, issue 1, pp. 358-371, January 2015.

[3] **L.R. Garcia Carrillo**, G. Flores, G. Sanahuja, and R. Lozano. "Quad Rotorcraft Switching Control: An Application for the Task of Path Following". IEEE Transactions on Control Systems Technology, vol. 22, issue 4, pp. 1255-1267, July 2014.

Commercially available agents: system identification and control

Motivation and Problem Statement

It is convenient (and sometimes less expensive) to use ready-to-fly commercially-available robotic platforms

AR.Drone ($90) **Bebop** ($150) **Sumo** and **Mambo** ($30)



Challenge: manufacturers of commercial systems do not make available

- equations describing the **dynamic model** of their products
- details about **control strategies** running onboard

Motivation and Problem Statement

It is convenient (and sometimes less expensive) to use ready-to-fly commercially-available robotic platforms

AR.Drone ($90) **Bebop** ($150) **Sumo** and **Mambo** ($30)



Challenge: manufacturers of commercial systems do not make available

- equations describing the **dynamic model** of their products
- details about **control strategies** running onboard

Developing control strategies for commercial systems requires

- collecting input-output system information data
- system identification techniques: Extended Least-Squares (ELS)
- developing a controller taking into account the identified model

Collecting input-output system data

Ground Station computer generates **control input** signals

- programmed in **Robot Operating System (ROS)** software

Motion Capture System is used for recording **state information**

- UAS is equipped with spheres reflecting infrared light

Extended Least-Squares for model identification

Consider an n-th order difference equation model of the form

$$A_q y_k = B_q u_k + \epsilon_k, \qquad y_k = z_k + \nu_k$$

with $\epsilon_k = A_q \nu_k$. The optimal least square (OLS) solution is

$$\hat{\theta}_{\text{OLS}} = (\Phi^T \Phi)^{-1} \Phi^T y, \qquad y = \Phi\theta + \epsilon$$

Extended Least-Squares for model identification

Consider an n-th order difference equation model of the form

$$A_q y_k = B_q u_k + \epsilon_k, \qquad y_k = z_k + \nu_k$$

with $\epsilon_k = A_q \nu_k$. The optimal least square (OLS) solution is

$$\hat{\theta}_{\text{OLS}} = (\Phi^T \Phi)^{-1} \Phi^T y, \qquad y = \Phi\theta + \epsilon$$

To eliminate the bias induced by the noise in $\hat{\theta}_{OLS}$, ϵ_k is modeled as $C_q \epsilon_k = e_k$, with e_k a random sequence

The term ϵ_k can be estimated from

$$\hat{\epsilon}_k = \hat{A}_q y_k - \hat{B}_q u_k$$

with $\hat{A}$ and $\hat{B}$ the respective estimation of A and B

Extended Least-Squares for model identification

Combining the estimation problems

$$y = [\Phi\ \Omega] \begin{bmatrix} \theta \\ \Pi \end{bmatrix} + e$$

whose solution is provided as follows:

$$\hat{\theta} = \hat{\theta}_{OLS} - \hat{\theta}_{\text{Bias}}, \qquad \hat{\theta}_{\text{Bias}} = (\Phi^T\Phi)^{-1}\Phi^T\Omega\hat{\Pi}, \qquad \hat{\Pi} = [\Omega^T\Omega]^{-1}\Omega^T\epsilon$$

This problem is nonlinear and must be solved iteratively

1. first we find $\hat{\theta}_{\text{OLS}}$
2. then we compute $\hat{\epsilon}$
3. finally we calculate $\hat{\theta}$

Recursive procedure is repeated until the norm of the difference of two $\hat{\theta}$ corresponding to two consecutive iterations is small enough

Dead-Time identification

Common issue on real-time system identification

- hardware/software induced time-delayed measurements

Supposing an output time-delay of τ samples

$$y_k = z_{k-\tau} + \nu_k$$

we obtain the time-delay combining the ELS with the solution of

$$\min_{\tau_i} \|y - \hat{y}_{(i)}\|^2,$$

where

- y is ground truth
- $\hat{y}_{(i)}$ is the output of the i-th model with a τ_i time-delay
- $\tau_i = \tau_{\min} + i, \quad i = 0, 1, 2, ..., (\tau_{\max} - \tau_{\min})$

The model with the minimum error norm is chosen

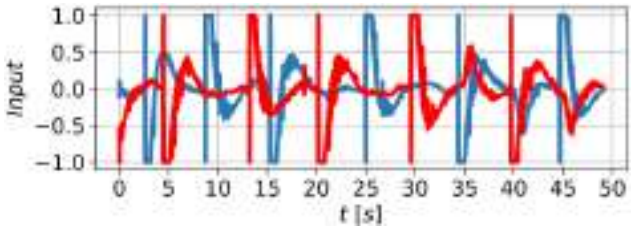
Practical considerations for model identification

- **Signal scaling:** Better numerical condition if inputs and outputs have similar order of magnitude
- **Down-Sampling:** If a down-sample can be applied to the signals, it helps to filter the signals
- **Dealing with known parameters:** If one or more poles/zeros of the process are known, imposing that structure simplifies the problem
- **Quality of Fit:** Mean-Square Error (MSE) normalized by the Mean-Square Output (MSO) as

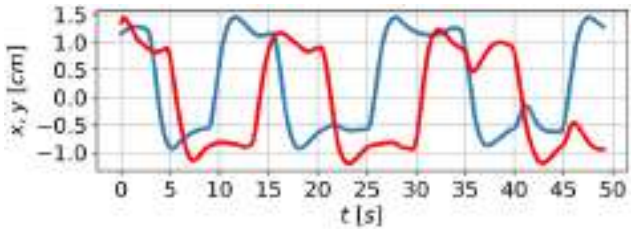
$$\frac{MSO}{MSE} = \frac{\|\Phi\hat{\theta} - y\|^2}{\|y\|^2}$$

Identification experiment in the X–Y plane

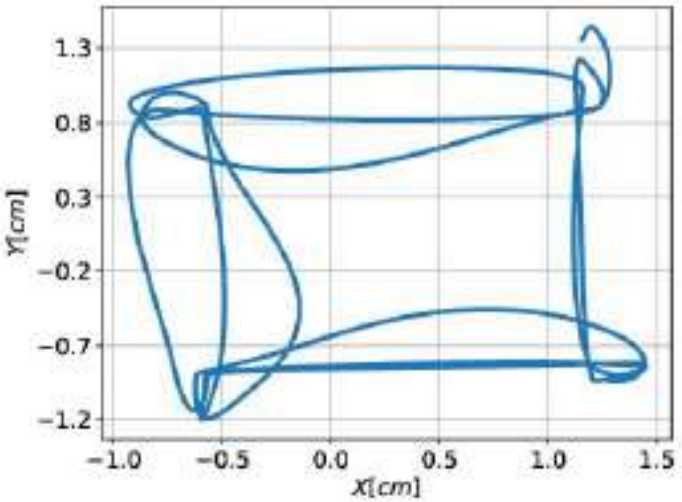
X–Y control inputs (PD)



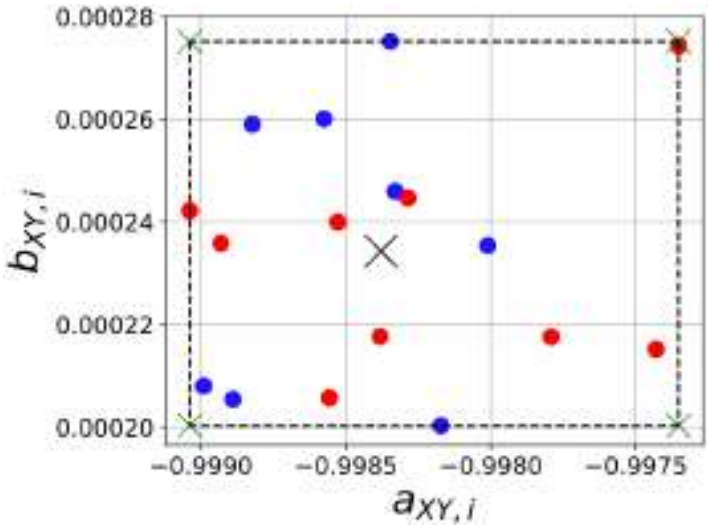
X–Y states



X–Y trajectories



Identification of multiple models (multiple UASs)



Imposing structure: integrator plus a first order system

Defining the state vector $\bar{x}_k = [x_k, \Delta x_k]$

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k, \qquad y_k = C\bar{x}_{k-\tau}$$

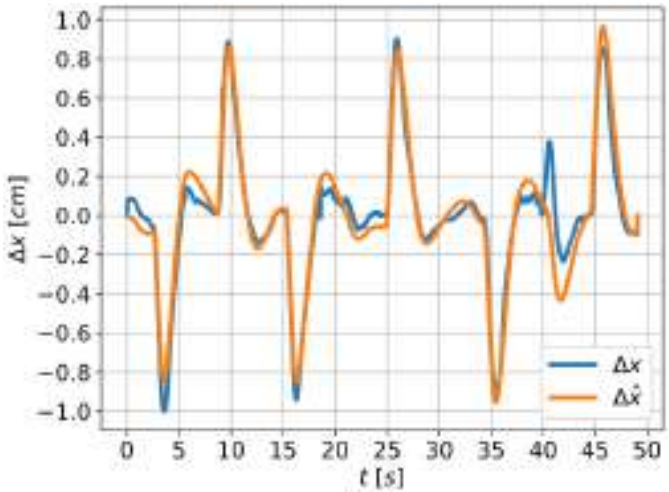
$$A = \begin{bmatrix} 1 & 1 \\ 0 & -a_m \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ b_m \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

where $\tau = 17$ samples, a_m and b_m are the nominal model, with $a_{\min} < a_m < a_{\max}$ and $b_{\min} < b_m < b_{\max}$

$$a_m = -0.99837, \quad a_{\max} = -0.99735 \quad a_{\min} = -0.99903$$

$$b_m = 0.2343 \cdot 10^{-3}, \quad b_{\max} = 0.2751 \cdot 10^{-3} \quad b_{\min} = 0.2003 \cdot 10^{-3}$$

Validation - velocity dynamics



Robust control strategy: H_∞ controller synthesis

Parameter–dependent linear time variant (LTV) discrete-time system

$$\begin{aligned} x_{k+1} &= A(\xi_k)x_k + B(\xi_k)u_k + E(\xi_k)d_k \\ y_k &= C(\xi_k)x_k + D(\xi_k)u_k + F(\xi_k)d_k \end{aligned}$$

The H_∞ performance is defined by the l_2-to-l_2 gain

$$\|H\|_\infty = \mu = \sup_{\|d_k\|_2 \neq 0} \frac{\|y_k\|_2}{\|d_k\|_2}$$

Robust control strategy: H_∞ controller synthesis

Parameter–dependent linear time variant (LTV) discrete-time system

$$\begin{aligned} x_{k+1} &= A(\xi_k)x_k + B(\xi_k)u_k + E(\xi_k)d_k \\ y_k &= C(\xi_k)x_k + D(\xi_k)u_k + F(\xi_k)d_k \end{aligned}$$

The H_∞ performance is defined by the l_2-to-l_2 gain

$$\|H\|_\infty = \mu = \sup_{\|d_k\|_2 \neq 0} \frac{\|y_k\|_2}{\|d_k\|_2}$$

Lemma 1: The system is poly-quadratically stabilizable with H_∞ performance bound μ, iff $\exists\, Q_i = Q_i^T \succ 0$ and X, L such that

$$\begin{bmatrix} X + X^T - Q_i & \star & \star & \star \\ 0 & \mu I & \star & \star \\ A_i X + B_i L & E_i & Q_j & \star \\ C_i X + D_i L & F_i & 0 & \mu I \end{bmatrix} \succ 0$$

The robust control law is given by $u_k = K x_k$, with $K = L X^{-1}$.

Numerical values H_∞ controller

We propose to down-sample the model to a number equal to τ

- make time delay equal to the sampling time

$$A_d = A^\tau, \quad B_d = \sum_{j=0}^{\tau-1} A^j B, \quad C_d = C \quad \text{and} \quad D_d = D$$

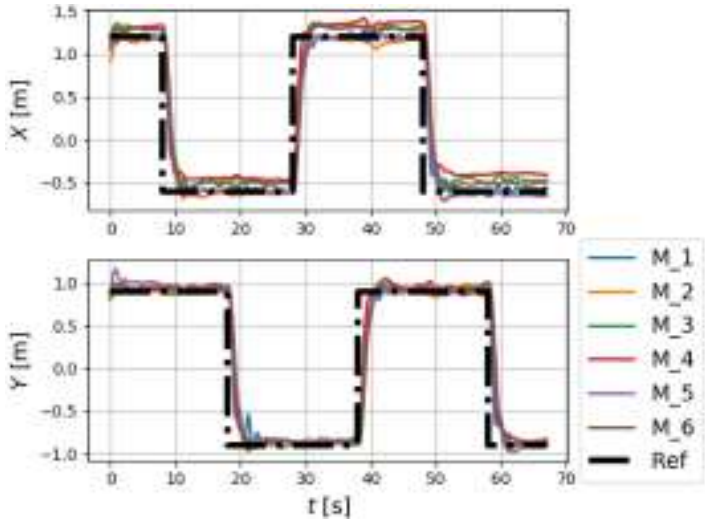
- We assume $E = [0\ 1]^T$ and $F = [1]$
- To minimize the energy we replaced the D with $D_{\text{opt}} = 0.075$

The following controller gain matrix is obtained minimizing μ

$$K = [-3.52718, -269.24932], \quad \mu = 57.7053$$

Closed-loop time response for X dynamics for all models

Validation in Real–time experiments



Recently published in IEEE GLOBECOM, December 2019

Current directions of this research

- online disturbance estimation to feed-forward the perturbation information

$$d_k = x_k - (Ax_{k-1} + Bu_{k-1})$$

Application: autonomous tracking of suspicious activity

Single Agent

The more, the better?

Application: autonomous tracking of suspicious activity

Single Agent

The more, the better?

Coordinating the actions of the agents and managing the information these systems retrieve **is crucial to prevent**

- loss of dynamic agent network deployment advantage
- damaging collisions or interference

Multi-Agent Systems
Consensus and Flocking Control

What is needed to coordinate a team of agents?

Communication/Perception: detect neighbors, obstacles, targets

Computation: monitor/control the physical processes.

Actuation: motion control using online measurements

Challenges

- Limited communication, sensing, computation, actuation
- Model uncertainty, and non-linearity
- Loss of measurements, large noise, low accuracy
- Fusion of different sensing modalities
- Computational complexity

What is needed to coordinate a team of agents?

Communication/Perception: detect neighbors, obstacles, targets

Computation: monitor/control the physical processes.

Actuation: motion control using online measurements

Challenges

- Limited communication, sensing, computation, actuation
- Model uncertainty, and non-linearity
- Loss of measurements, large noise, low accuracy
- Fusion of different sensing modalities
- Computational complexity

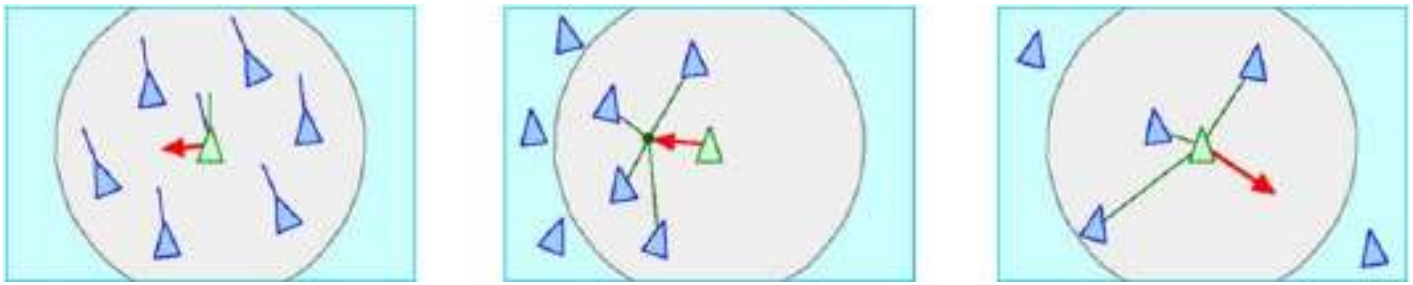
Coordinated behavior in nature: foraging, schooling, flocking



Decoding nature's coordinated behavior

Motion is derived from weighted combination of force vectors

- alignment: steer towards average heading of neighbors
- attraction: steer towards average position of neighbors
- repulsion: steer to avoid crowding neighbors (separation)



Each agent can only see other agents in its neighborhood

- global system exhibits intelligent coordinated behavior

Mathematical models to emulate coordinated behavior

Assume n agents with second order dynamics evolving in an m dimensional space ($m = 2, 3$). Motion of each agent i is

$$\begin{cases} \dot{q}_i = p_i \\ \dot{p}_i = u_i \end{cases}, \quad i = 1, 2, \ldots, n$$

$\{u_i,\, q_i,\, p_i\} \in \mathbb{R}^m$ are control input, position, and velocity of agent i

Mathematical models to emulate coordinated behavior

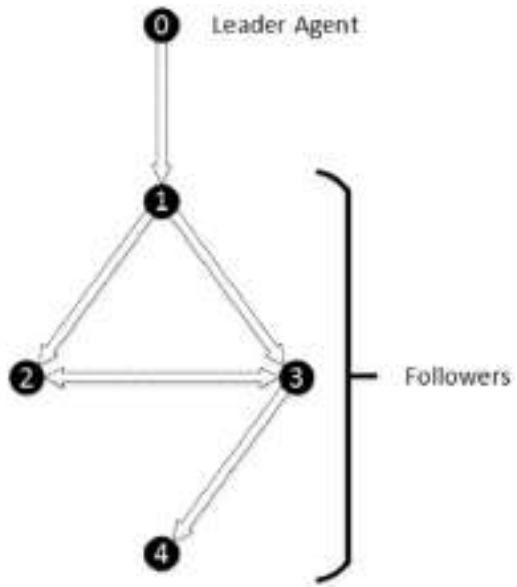
Assume n agents with second order dynamics evolving in an m dimensional space ($m = 2, 3$). Motion of each agent i is

$$\begin{cases} \dot{q}_i = p_i \\ \dot{p}_i = u_i \end{cases}, \quad i = 1, 2, \ldots, n$$

$\{u_i,\, q_i,\, p_i\} \in \mathbb{R}^m$ are control input, position, and velocity of agent i

Consider a dynamic graph $G(\upsilon, \varepsilon)$ consisting of a set of

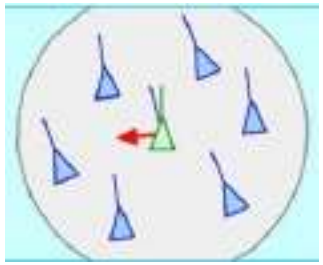
- **vertices** $\upsilon = \{1,\ 2,\ \ldots,\ n\}$, each one an **agent** i
- **edges** $\varepsilon \subseteq \{(i, j) : i, j \in \upsilon, j \neq i\}$, each one a communication **link** between agents i and j



Mathematical models to emulate coordinated behavior

The neighborhood set of agent i is

$$N_i^{\alpha} = \{j \in \upsilon_{\alpha} : \|q_j - q_i\| < r, j \neq i\}$$

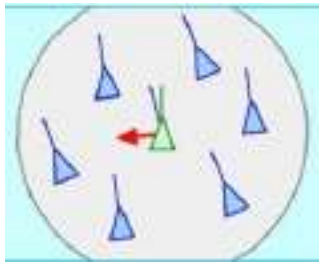


where r is the range of interaction between agents i and j

Mathematical models to emulate coordinated behavior

The neighborhood set of agent i is

$$N_i^\alpha = \{j \in \upsilon_\alpha : \|q_j - q_i\| < r, j \neq i\}$$



where r is the range of interaction between agents i and j

To find the geometric model of the flock, i.e., the α–lattice, we solve

$$\|q_j - q_i\|_\sigma = d_\alpha \quad \forall j \in N_i^\alpha$$

where $d_\alpha = \|d\|_\sigma$ is the sigma norm (differentiable everywhere) of the positive constant d (distance between neighbors i and j)

A smooth collective potential function $\psi_\alpha(z)$ can be obtained as

$$V(q) = \frac{1}{2} \sum_i \sum_{j \neq i} \psi_\alpha(\|q_j - q_i\|_\sigma)$$

Mathematical models to emulate coordinated behavior

Potential functions allow creating **flocking control algorithms** for avoiding obstacles and tracking a target, while making all agents to form an α–lattice configuration [Olfati-Saber, 2006]

$$u_i = u_i^{\alpha} + u_i^{\beta} + u_i^{\gamma}$$

with

$$u_i^{\alpha} = c_1^{\alpha} \sum_{j \in N_i^{\alpha}} \phi_{\alpha}(\|q_j - q_i\|_{\sigma}) \mathbf{n}_{i,j} + c_2^{\alpha} \sum_{j \in N_i^{\alpha}} a_{ij}(q)(p_j - p_i)$$

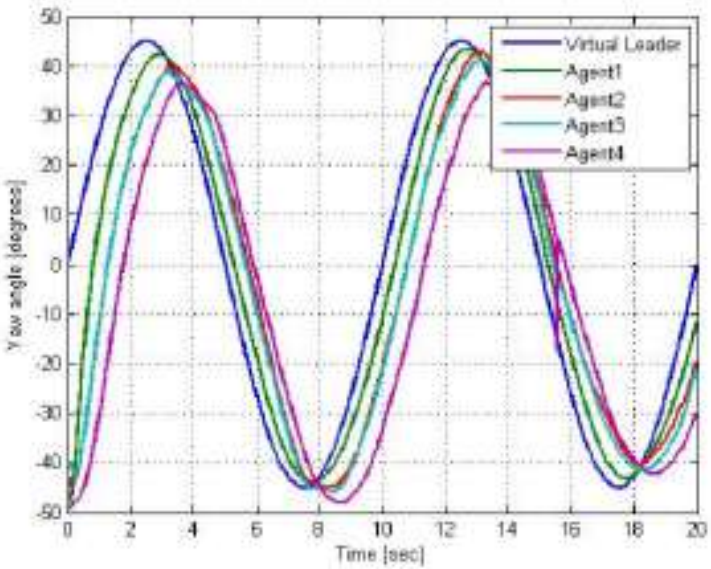
$$u_i^{\beta} = c_1^{\beta} \sum_{k \in N_i^{\beta}} \phi_{\beta}(\|\hat{q}_{i,k} - q_i\|_{\sigma}) \hat{\mathbf{n}}_{i,k} + c_2^{\beta} \sum_{k \in N_i^{\beta}} b_{i,k}(q)(\hat{p}_{i,k} - p_i)$$

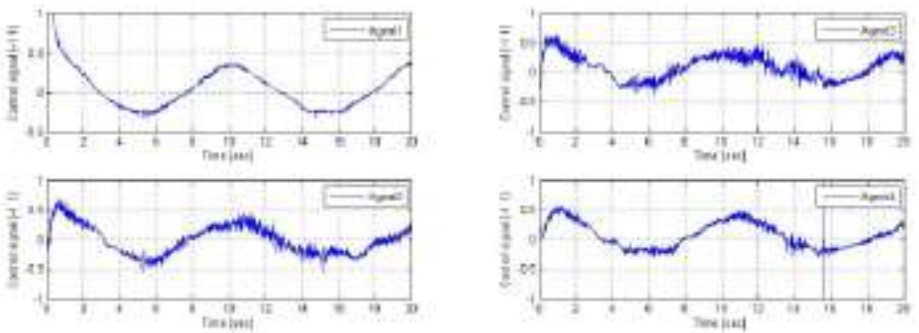
$$u_i^{\gamma} = -c_1^{\gamma}(q_i - q_r) - c_2^{\gamma}(p_i - p_r) - c_1^{\mathrm{sc}}\left(\frac{\sum_{i=1}^{n} q_i}{n} - q_r\right) - c_2^{\mathrm{sc}}\left(\frac{\sum_{i=1}^{n} p_i}{n} - p_r\right)$$

where

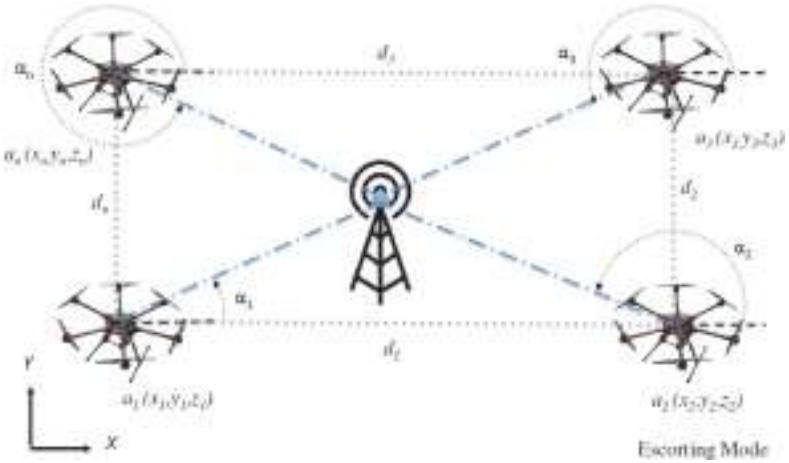
- c_1^{α}, c_1^{β}, c_1^{γ}, c_1^{sc}, c_2^{α}, c_2^{β}, c_2^{γ}, c_2^{sc} are positive constants
- (q_r, p_r) is the coordinates of a virtual leader of the MAS flock

MAS consensus control in real-time





MAS flocking control in real-time



Solved if for each agent, and for any initial $x_i(0)$, there is a local control u_i, such that the closed-loop system satisfies

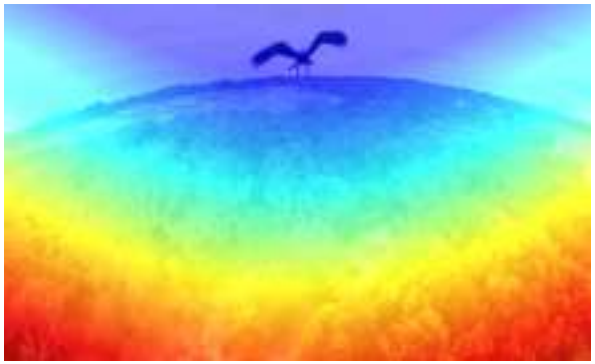
$$\lim_{t\to\infty} \| (x_i(t) - h_i) - x_0(t) \| = 0$$

$h_i \in \mathbb{R}^n$ is a formation vector $[h_{X,i} \;\; h_{Y,i} \;\; h_{\psi,i}]^\top$, associated with a desired position in the $\{X, Y\}$ plane, and a heading angle ψ

MAS - Current directions

Building a prototype for performing onboard distributed MAS consensus and flocking

Onboard detection of neighbors, obstacles, targets



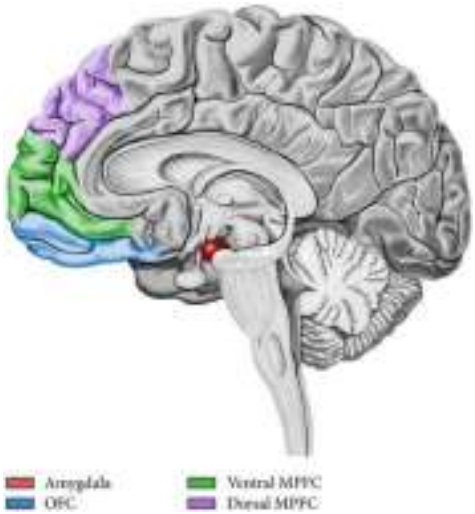
Development of learning–inspired estimation and control for MAS

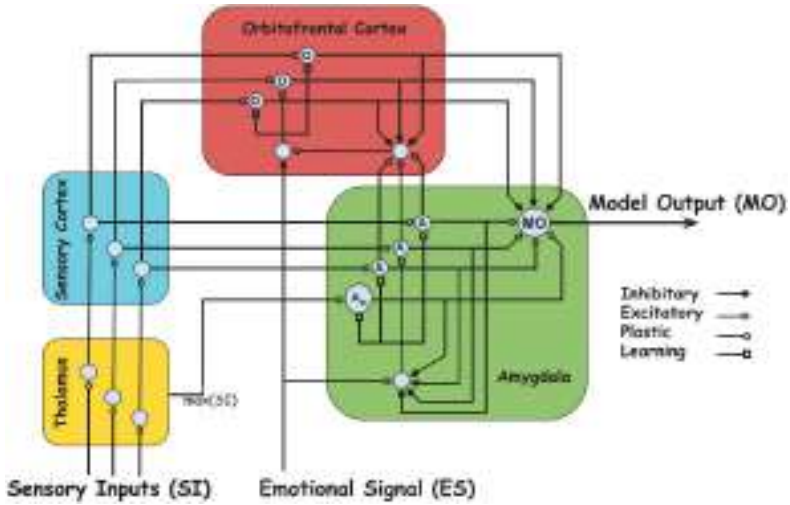
- address model uncertainties

Learning–inspired estimation and control

Brain Emotional Learning (BEL) computational model

Model mimics parts of the brain that are known to produce emotion





$$MO = A_{th} + \sum_{i=1}^{n} A_i - \sum_{i=1}^{n} OC_i$$

outputs of Amygdala and OFC are obtained as

$$A_i = V_i SI_i, \qquad OC_i = W_i SI_i$$

- V_i and W_i: plastic weights of Amygdala and OFC
- SI_i: the i_{th} sensory input

Brain Emotional Learning (BEL) computational model

Update law for V_i

$$\dot{V}_i = \alpha \cdot SI_i \cdot \max\left(0, ES - A_{th} - \sum_{i=1}^{n} A_i\right)$$

α learning rate parameter: $0 =$ no learning, $1 =$ instant adaptation

Weights V_i can not decrease

- once an emotional reaction is learned, it should be permanent
- it is the task of the OFC to inhibit this reaction if needed

Brain Emotional Learning (BEL) computational model

Update law for V_i

$$\dot{V}_i = \alpha \cdot SI_i \cdot \max\left(0, ES - A_{th} - \sum_{i=1}^{n} A_i\right)$$

α learning rate parameter: $0 =$ no learning, $1 =$ instant adaptation

Weights V_i can not decrease

- once an emotional reaction is learned, it should be permanent
- it is the task of the OFC to inhibit this reaction if needed

Update law for W_i: a function of the SI_i and reinforcer ES

$$\text{If } ES \neq 0 \qquad \dot{W}_i = \beta \cdot SI_i \left(\max\left(0, \sum_{i=1}^{n} A_i - ES\right) - \sum_{i=1}^{n} OC_i\right)$$

$$\text{If } ES = 0 \qquad \dot{W}_i = \beta \cdot SI_i \cdot \max\left(0, \sum_{i=1}^{n} A_i - \sum_{i=1}^{n} OC_i\right)$$

β is a learning rate parameter similar to α

Proposed approach: Feedback linearizing controller

Consider the class of nonlinear systems of order n described by

$$x^{(n)} = f(\underline{x}) + g(\underline{x})u + d(\underline{x}, t)$$

Variable depending on tracking error $e = x - x_d$ and its derivatives

$$s := s(\underline{x}) = e^{(n-1)} + \Delta_{n-1}e^{(n-2)} + \ldots + \Delta_1 e$$

Derivative of variable s is

$$\dot{s} = f(\underline{x}) + g(\underline{x})u + q_a(t) + d(\underline{x}, t)$$

with $q_a = -x_d^{(n)} + e^{(n-1)} + \Delta_{n-1}e^{(n-2)} + \ldots + \Delta_1 \dot{e}$

Proposed approach: Feedback linearizing controller

Consider the class of nonlinear systems of order n described by

$$x^{(n)} = f(\underline{x}) + g(\underline{x})u + d(\underline{x}, t)$$

Variable depending on tracking error $e = x - x_d$ and its derivatives

$$s := s(\underline{x}) = e^{(n-1)} + \Delta_{n-1}e^{(n-2)} + \ldots + \Delta_1 e$$

Derivative of variable s is

$$\dot{s} = f(\underline{x}) + g(\underline{x})u + q_a(t) + d(\underline{x}, t)$$

with $q_a = -x_d^{(n)} + e^{(n-1)} + \Delta_{n-1}e^{(n-2)} + \ldots + \Delta_1 \dot{e}$
Feedback linearization: solve tracking problem by selecting u such that

$$\dot{s} = -Ks + u_r$$

$K > 0$ a constant, u_r auxiliary input for improving performance
If $f(\underline{x})$ and $g(\underline{x})$ were known and $d(\underline{x}, t) = 0$, then we need

$$u^* = -g^{-1}(\underline{x})(f(\underline{x}) + q_a + Ks - u_r)$$

obtained by equating the right-hand sides of the $\dot{s}$ equations

Proposed approach: Control with robust integral action

State $\xi(t) = \int s(t)dt$ is introduced to augment the system and improve performance through an integral action

$$\begin{bmatrix} \dot{s} \\ \dot{\xi} \end{bmatrix} = \underbrace{\begin{bmatrix} -K & 0 \\ 1 & 0 \end{bmatrix}}_{A_e} \underbrace{\begin{bmatrix} s \\ \xi \end{bmatrix}}_{s_e} + \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{B_e} u_r$$

$u_r = -\frac{1}{r} B_e^T P_e s_e$ is obtained by solving the Ricatti equation

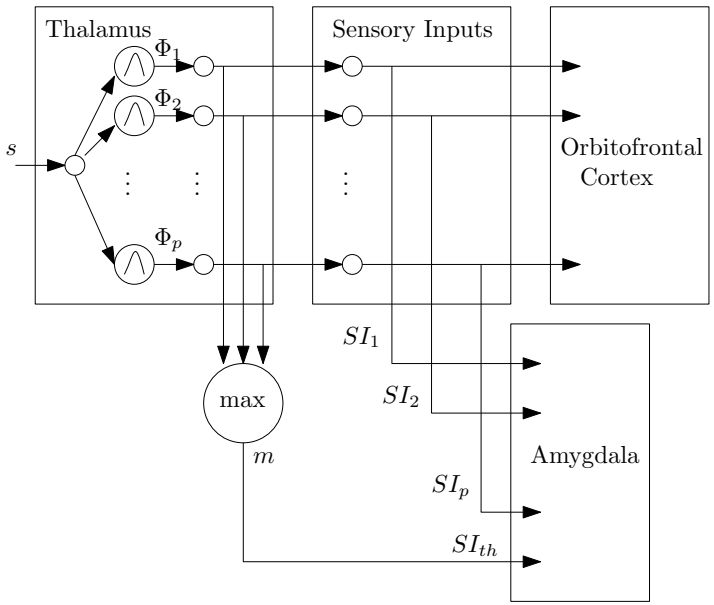
$$0 = A_e^T P_e + P_e A_e - P_e B_e R^{-1} B_e^T P_e + Q_e$$

where $Q_e = \text{diag}\{Q, Q_I\}$, $Q_e = Q_e^T \succ 0$, $R = \frac{\rho^2 r}{2\rho^2 - r}$ with $2\rho^2 > r$

Radial Basis Functions in Neural Networks

Approximate $f(\underline{x})$ and $g(\underline{x})$ by means of estimates $\hat{f}(\underline{x})$ and $\hat{g}(\underline{x})$ using a combination of Gaussian RBF's that emulates the BEL structure

$$\hat{f}(\underline{x}) := \hat{f}(\underline{x}, V_f, W_f) = V_f^T \Phi_A(s(\underline{x})) - W_f^T \Phi(s(\underline{x}))$$
$$\hat{g}(\underline{x}) := \hat{g}(\underline{x}, V_g, W_g) = V_g^T \Phi_A(s(\underline{x})) - W_g^T \Phi(s(\underline{x}))$$



Input is processed in the Thalamus by the RBFs given the SI's

$$\Phi_j = \exp\left(-\frac{(s-\mu_j)^2}{\sigma_j^2}\right)$$
$$m = \max([\Phi_1, \Phi_2, \ldots, \Phi_p])$$

Radial Basis Functions in Neural Networks

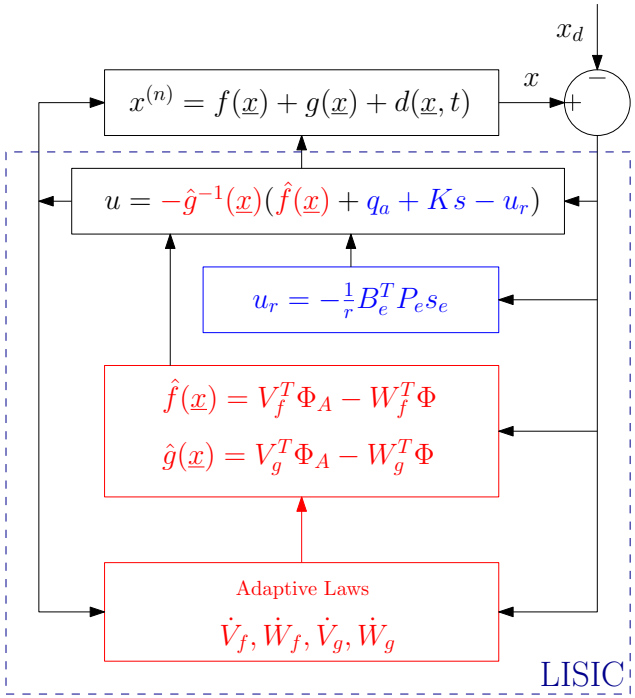
Proposed adaptation rules (similar to BEL)

$$\dot{V}_f = \alpha_f \Phi_A \max(B_e^T P_e s_e, 0), \quad \dot{W}_f = -\beta_f \Phi B_e^T P_e s_e,$$
$$\dot{V}_g = \alpha_g \Phi_A \max(B_e^T P_e s_e u, 0), \quad \dot{W}_g = -\beta_g \Phi B_e^T P_e s_e u$$

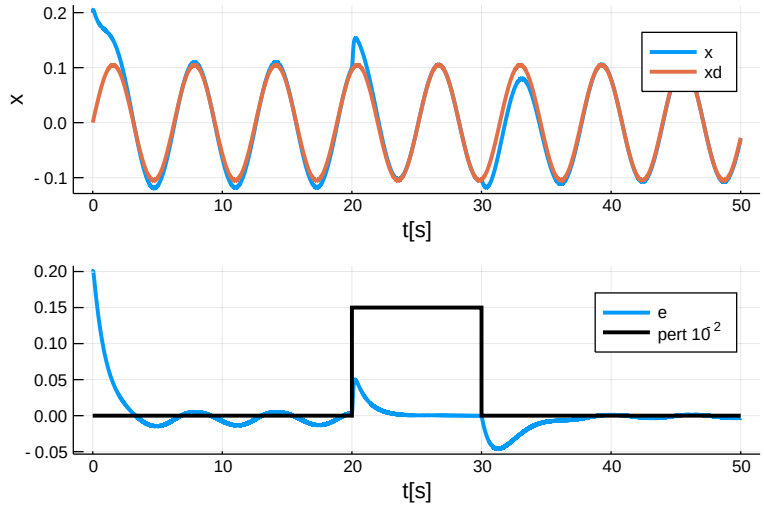
Following the universal approximation property exhibited by Gaussian RBF's, there exist optimal parameters $[V_f^*, W_f^*]$ and $[V_g^*, W_g^*]$ such that $\hat{f}(\underline{x})$ and $\hat{g}(\underline{x})$ can approximate $f(\underline{x})$ and $g(\underline{x})$ on a compact set Ω_x

$$[V_f^*, W_f^*] = \underset{V_f \in \Omega_{fv}, W_f \in \Omega_{fw}}{\arg\min} [\sup_{\tilde{\underline{x}} \in \Omega_x} |V_f^T \Phi_A(\tilde{\underline{x}}) - W_f^T \Phi(\tilde{\underline{x}}) - f(\tilde{\underline{x}})|],$$
$$[V_g^*, W_g^*] = \underset{V_g \in \Omega_{gv}, W_g \in \Omega_{gw}}{\arg\min} [\sup_{\tilde{\underline{x}} \in \Omega_x} |V_g^T \Phi_A(\tilde{\underline{x}}) - W_g^T \Phi(\tilde{\underline{x}}) - g(\tilde{\underline{x}})|],$$

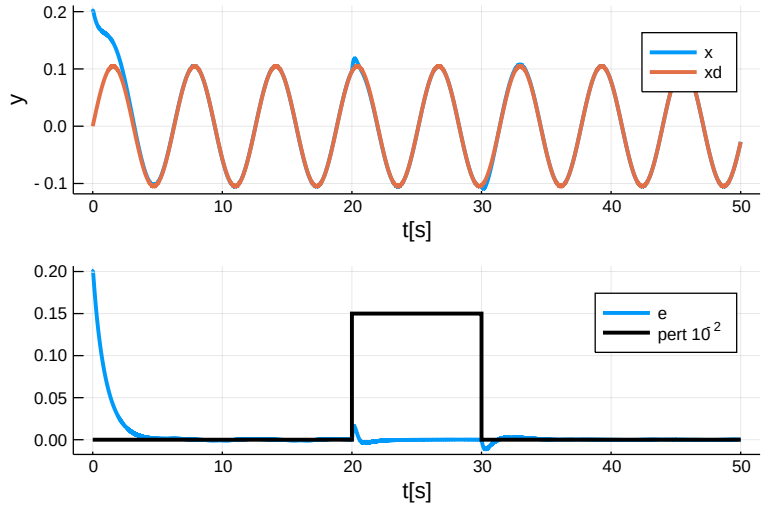
Emotional Learning Inspired Control - Performance



Conventional BEL with P feedback



Novel LISIC with a PI action



Stabilization of inverted pendulum

$$\ddot{x} = \frac{g\sin(x) - a_p m_p l \dot{x}^2 \sin(2x)/2}{4l/3 - a_p m_p l \cos(x)^2} + \frac{a_p \cos(x)}{4l/3 - a_p m_p l \cos(x)^2} u + d$$

$$y = x + \nu$$

Limbic system-inspired control (LISIC) for MAS

Challenges in real-time MAS control

- dynamics of the agents (aerial, ground, and water vehicles, or a combination of them) are usually not precisely known
- nonlinear protocol for interconnection of agents: nonlinear propagation of model uncertainties and external perturbations

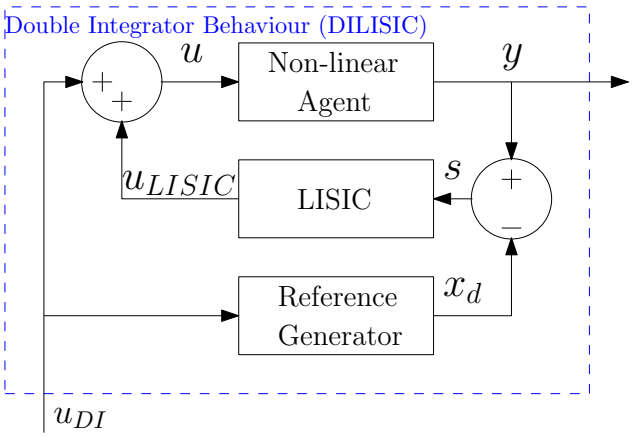
Limbic system-inspired control (LISIC) for MAS

Challenges in real-time MAS control

- dynamics of the agents (aerial, ground, and water vehicles, or a combination of them) are usually not precisely known
- nonlinear protocol for interconnection of agents: nonlinear propagation of model uncertainties and external perturbations

Proposed solution: Double Integrator–LISIC (DILISIC)

- composed by a nonlinear agent in closed-loop with a LISIC
- imitate double integrator dynamics (via the s variable)



Identify and compensate model differences between theoretical assumptions considered when tuning the consensus control, and the actual conditions encountered in the real-time system

Allows applying consensus techniques for double integrator agents

DILISIC structure imitating double integrator agents

Consider a reference model representing double integrator dynamics

$$\ddot{x}_d = u_{DI}$$

$(\cdot)_{DI}$: double integrator that the LISIC closed-loop should imitate

System output is compared with the reference model that represents the double integrator dynamics

$$e = x_d - y, \qquad x^{(n)} = f(\underline{x}) + g(\underline{x})(u_{LISIC} + u_{DI})$$

DILISIC structure imitating double integrator agents

Consider a reference model representing double integrator dynamics

$$\ddot{x}_d = u_{DI}$$

$(\cdot)_{DI}$: double integrator that the LISIC closed-loop should imitate

System output is compared with the reference model that represents the double integrator dynamics

$$e = x_d - y, \qquad x^{(n)} = f(\underline{x}) + g(\underline{x})(u_{LISIC} + u_{DI})$$

The DILISIC closed-loop system can now be rewritten as

$$x^{(n)} = f(\underline{x}) + g(\underline{x})u_{LISIC} + \underbrace{g(\underline{x})u_{DI} - u_{DI}^{(n-2)}}_{d(\underline{x},t)} + u_{DI}^{(n-2)}$$

For a second order system we have

$$\ddot{x} = f(\underline{x}) + g(\underline{x})u_{LISIC} + g(\underline{x})u_{DI} - u_{DI} + u_{DI}$$

If function $f(\underline{x}) = 0$ and $g(\underline{x}) = 1$, then $\ddot{x}_d = \ddot{x}$

- similar initial conditions, no compensation required: $u_{LISIC} = 0$

Robust Adaptive Control of MAS

Due to the DILISIC structure, each agent will inherit a nonlinear component that can be considered as a nonlinear function

Consider the MAS, but now with a nonlinear additive perturbation

$$\begin{cases} \dot{q}_i = p_i \\ \dot{p}_i = f(p_i) + u_i \end{cases} \qquad i = 1, 2, \ldots, n$$

nonlinear function $f(p_i)$ is the error produce by the DILISIC in the transformation of the original agent into a double integrator

Robust Adaptive Control of MAS

Due to the DILISIC structure, each agent will inherit a nonlinear component that can be considered as a nonlinear function

Consider the MAS, but now with a nonlinear additive perturbation

$$\begin{cases} \dot{q}_i = p_i \\ \dot{p}_i = f(p_i) + u_i \end{cases} \qquad i = 1, 2, \ldots, n$$

nonlinear function $f(p_i)$ is the error produce by the DILISIC in the transformation of the original agent into a double integrator

Distributed flocking algorithm

$$u_i = -\Delta_{r_i} V(r) + \sum_{j \in N_i} (a_{ij}(t) + \delta_{ij}(t))(p_j - p_i)$$

$\Delta_{r_i} V(r)$ is a gradient-based term of a collective potential function V

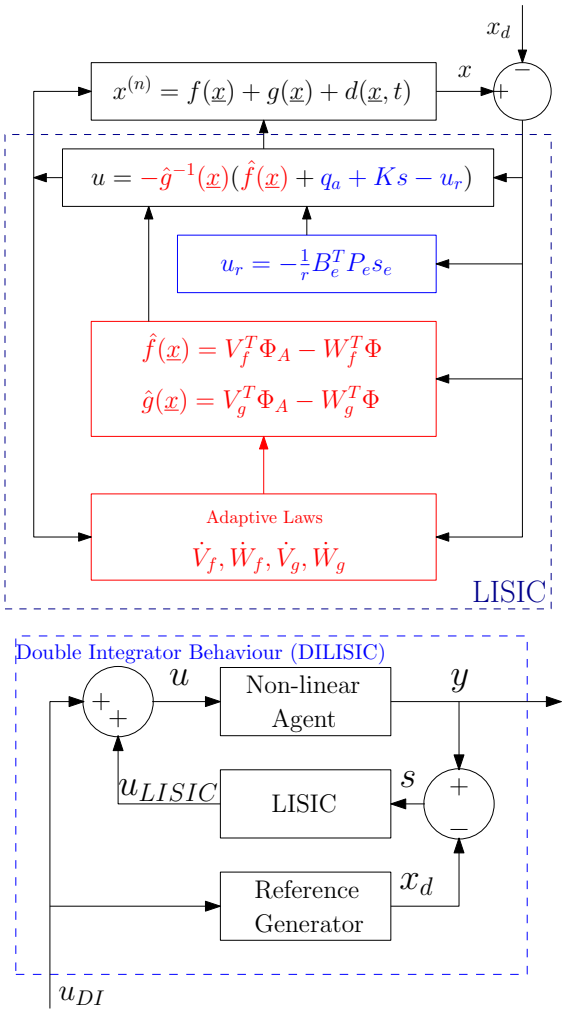
$$-\Delta_{r_i} V(r) = \phi_\alpha(\|q_j - q_i\|_\sigma)\mathbf{n}_{i,j}$$

the second term is the velocity consensus, with perturbation $\delta_{ij} \neq \delta_{ji}$

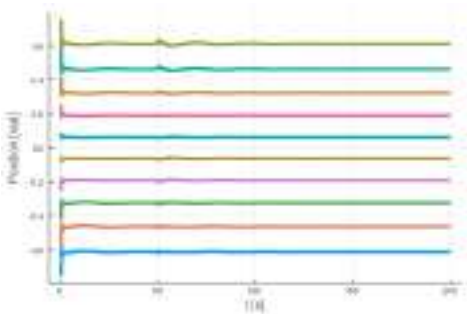
DILISIC - Performance for a 10-agent MAS

Dynamics of each pendulum

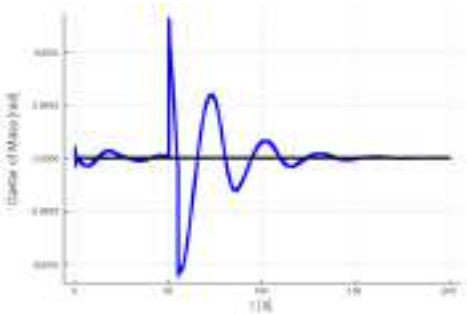
$$\ddot{x} = \frac{g\sin(x) - a_p m_p l \dot{x}^2 \sin(2x)/2}{4l/3 - a_p m_p l \cos(x)^2} + \frac{a_p \cos(x)}{4l/3 - a_p m_p l \cos(x)^2} u + d, \qquad y = [x,\ \dot{x}]^T$$



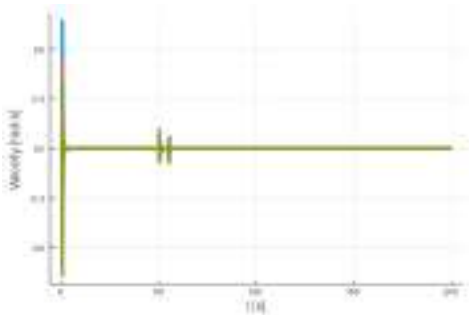
Positions of 10-agents



CoM of the MAS

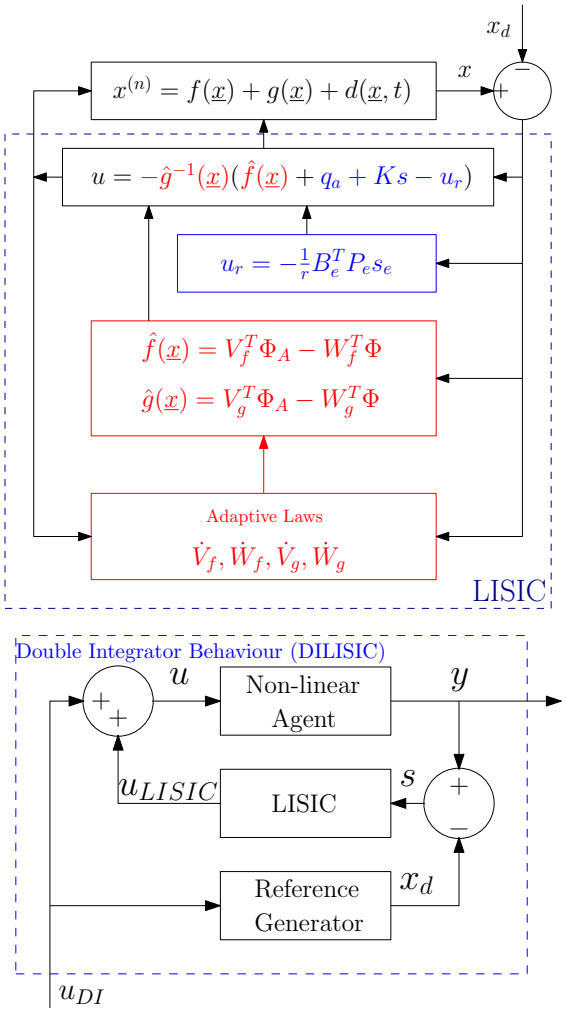


Angular velocity, disturbance at $t = 50$sec

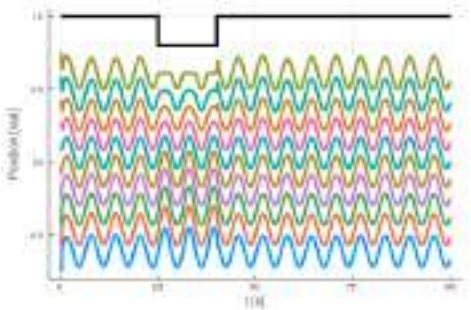


DILISIC - Performance for a 10-agent MAS

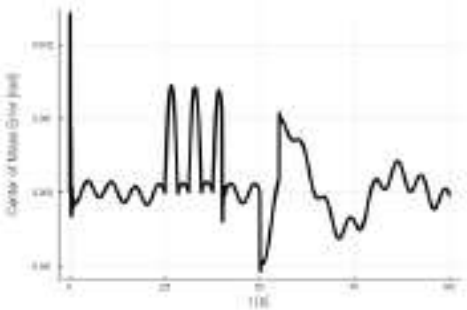
Same MAS, but now flockig: consensus, obstacle avoidance, tracking



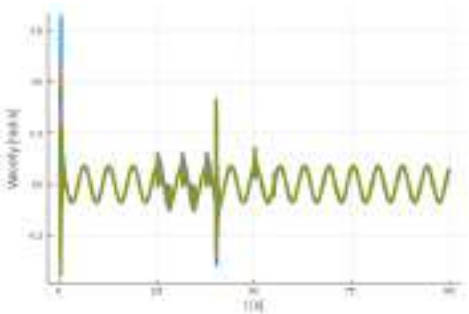
Positions of 10-agents



CoM of the MAS



Angular velocity, with obstacle and dist.



Learning–inspired estimation and control

The development of LISIC was sponsored by the Army Research Office

Network Sciences Division

Grant W911NF1810210

Multi-Agent Network Control
A Brain Emotional Learning-Inspired Approach

Current status of this research

- LISIC paper accepted in ACC 2020
- DILISIC paper submitted to conference publication
- Working on stability proofs
- Working on real–time implementation of both methods

Concluding Remarks and Future Directions

Concluding Remarks

What this presentation has covered

- Analytic high–level control for robust trajectory tracking
- Model identification/control for commercial robotic agents
- Multi–Agent Systems
- Learning inspired estimation and control for single and MAS

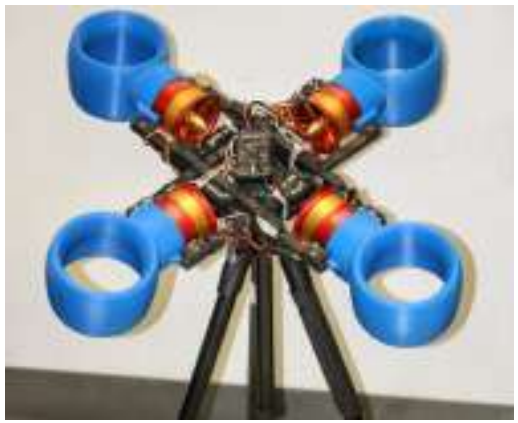
Concluding Remarks

What this presentation has covered

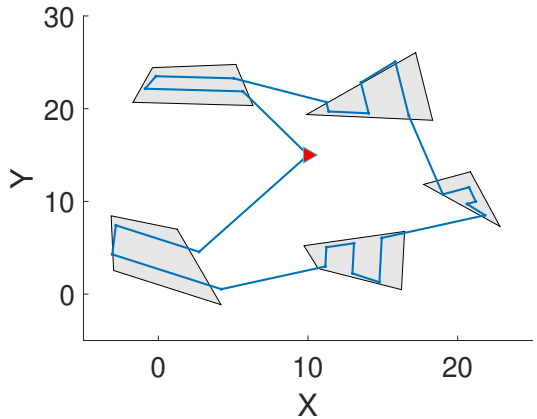
- Analytic high–level control for robust trajectory tracking
- Model identification/control for commercial robotic agents
- Multi–Agent Systems
- Learning inspired estimation and control for single and MAS

What this presentation has not covered

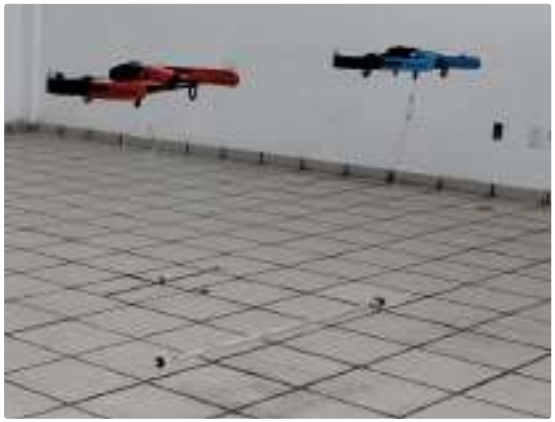
Novel Prototypes



TSP + CPP



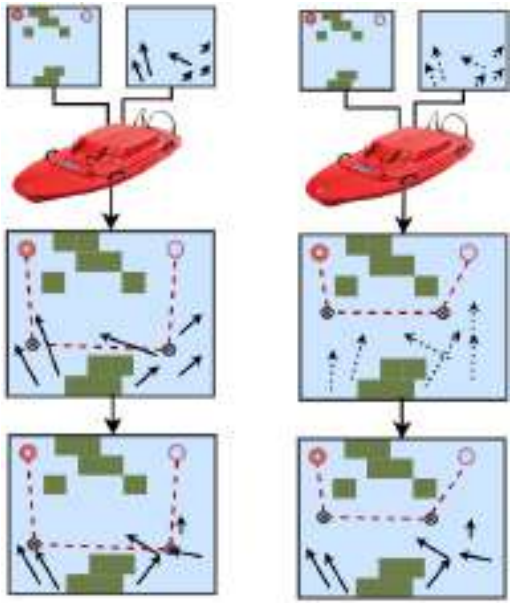
Load Transportation



Additional Current and Future Directions

Game theory-inspired navigation using weather forecasts

Naive vs. Strategic



- Game theory: applied to dynamic programming motion planning for strategic planning in uncertain environments

Strategy makes use of real data

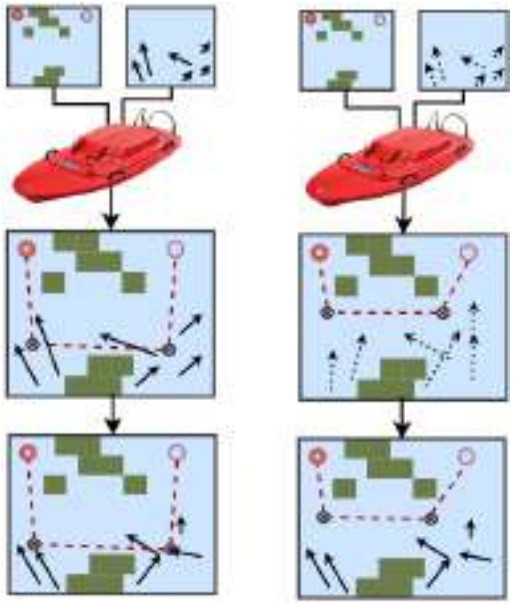
- Large marine region and online water forecasts

To appear in ACC 2020, Denver, CO, USA

Additional Current and Future Directions

Game theory-inspired navigation using weather forecasts

Naive vs. Strategic



- Game theory: applied to dynamic programming motion planning for strategic planning in uncertain environments

Strategy makes use of real data

- Large marine region and online water forecasts

To appear in ACC 2020, Denver, CO, USA

Game theory-inspired framework for designing optimal persistent inputs for learning–inspired control

Formulate game theory-inspired persistent control inputs to overcome the exploration/exploitation dilemma occurring in artificial learning mechanisms. Perform simultaneous identification and control

Joao
UCSB

Rogelio
CNRS & IPN

Kyriakos
GTech

Junfei
CalState SD

Ignacio
UN Rosario

Gabriel
TAMU-CC

Evan
TAMU-CC

End of Presentation

Estimation and Control for Autonomous Agents

Questions?