Granting Agencies: Department of Science and Technology,
Indian Space Research Organization

Time Optimal Feedback in Multi-Agent Systems

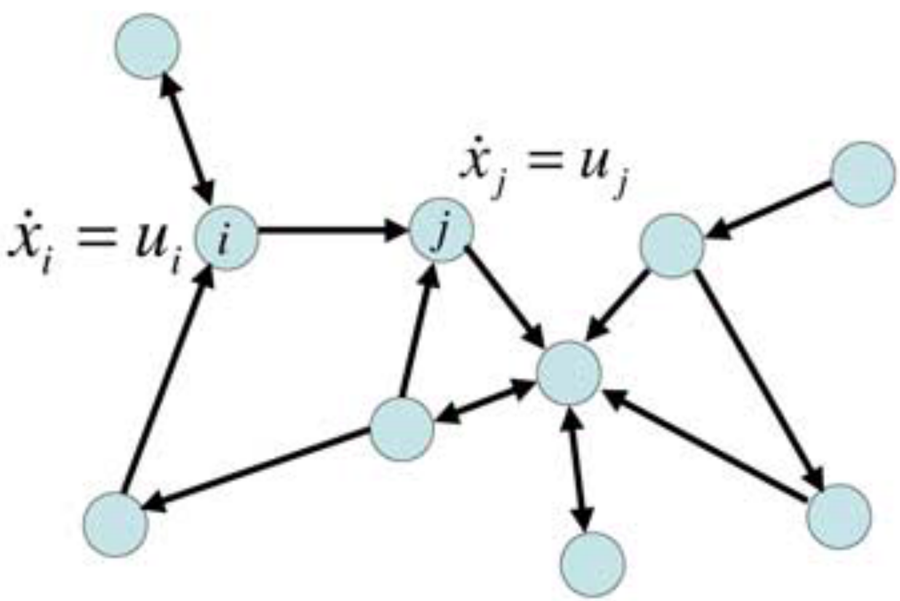
Joint Work with
Deepak Patil, Ameer Mulla, and Sujay Bhatt

Debraj Chakraborty

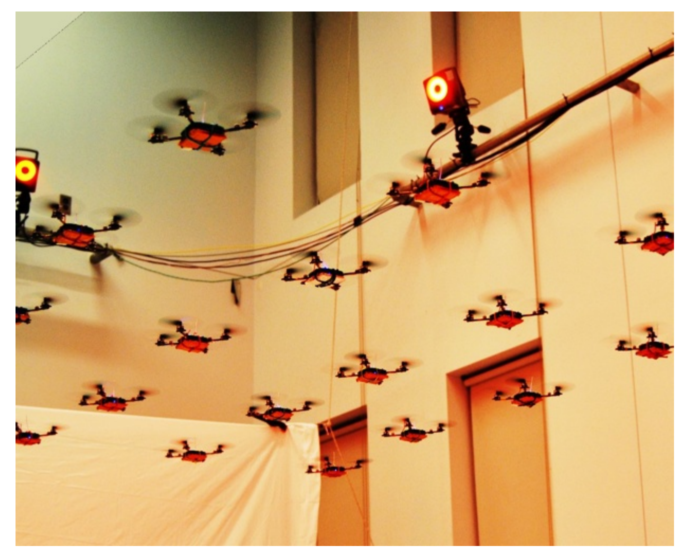
Department of Electrical Engineering, Control and Computing Group

Question

- Given a collection of autonomous dynamical systems (or 'agents') communicating with each other over (undirected/directed, time invariant/time varying) graph(s), how do we bring them to a consensus/ synchronize them in minimum time?



Olfati-Saber et al, Proceedings of IEEE, 2007



GRASP Lab, UPenn

We solve two sub-questions

Computation of Time Optimal **Feedback** using Groebner Basis

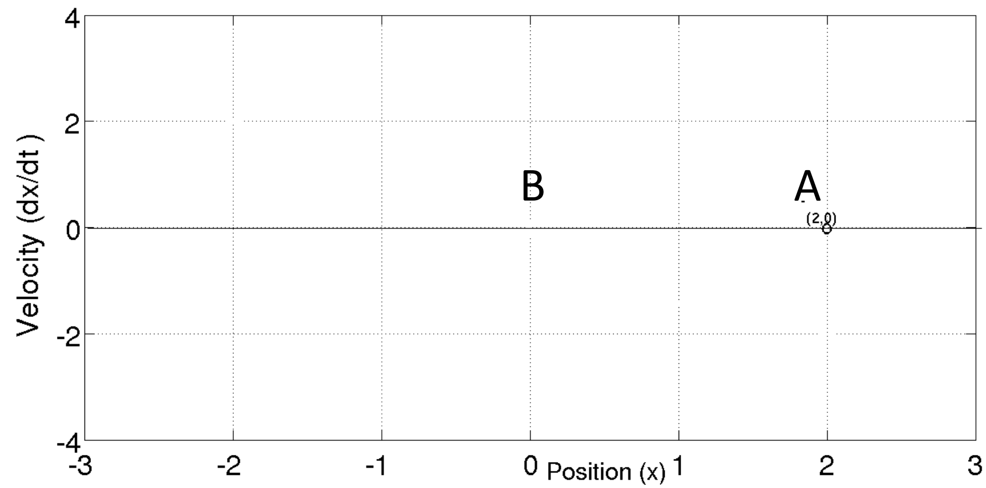
(Feedback) Pursuit-Evasion Games

Time Optimal Multi - agent Consensus (complete graph)

Time Optimal Leader Tracking in Multi-agent systems (directed graphs)

TIME OPTIMAL **FEEDBACK**

Time Optimal Feedback

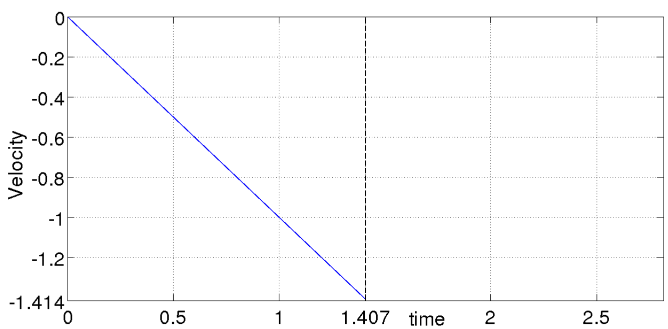


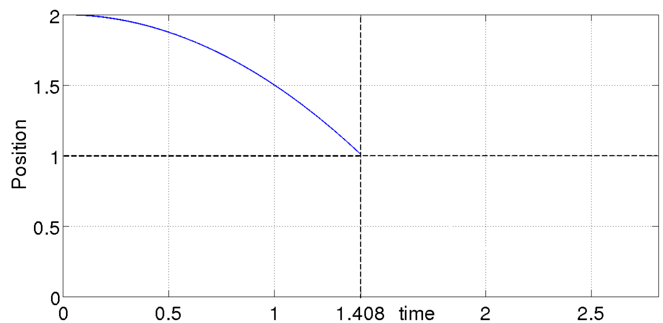
Problem: Go from A to B in minimum time with maximum allowed acceleration/deceleration = ± 1

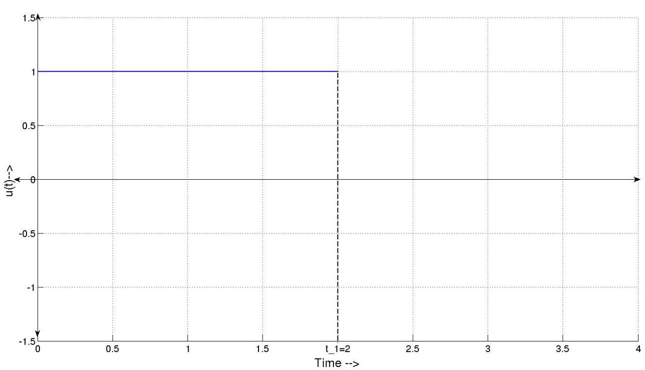
$$\dot{p} = v; \ \dot{v} = u$$

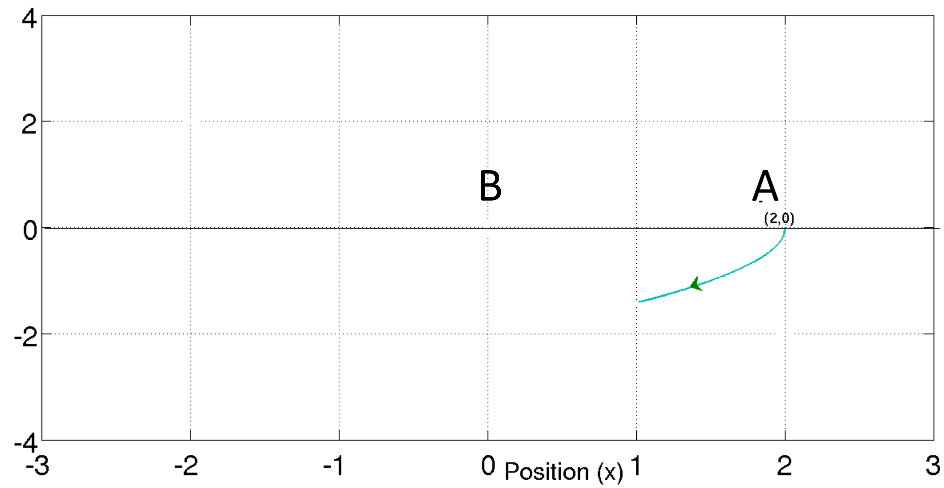
$$|u| \leq 1$$

Time Optimal Feedback







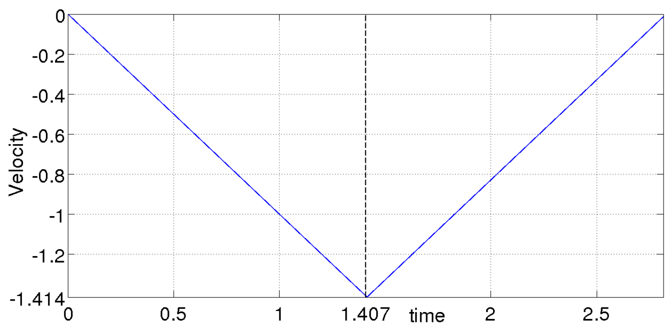


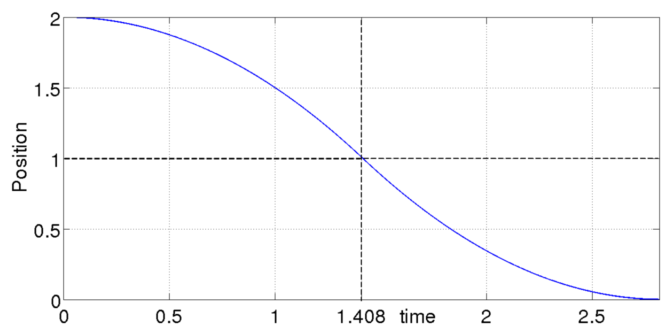
Problem: Go from A to B in minimum time with maximum allowed acceleration/deceleration = ± 1

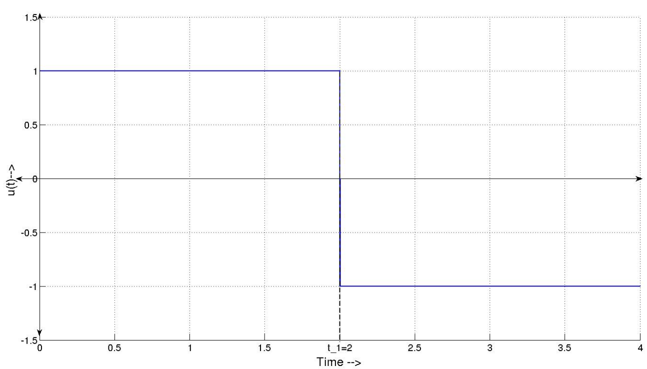
$$\dot{p} = v; \;\; \dot{v} = u$$

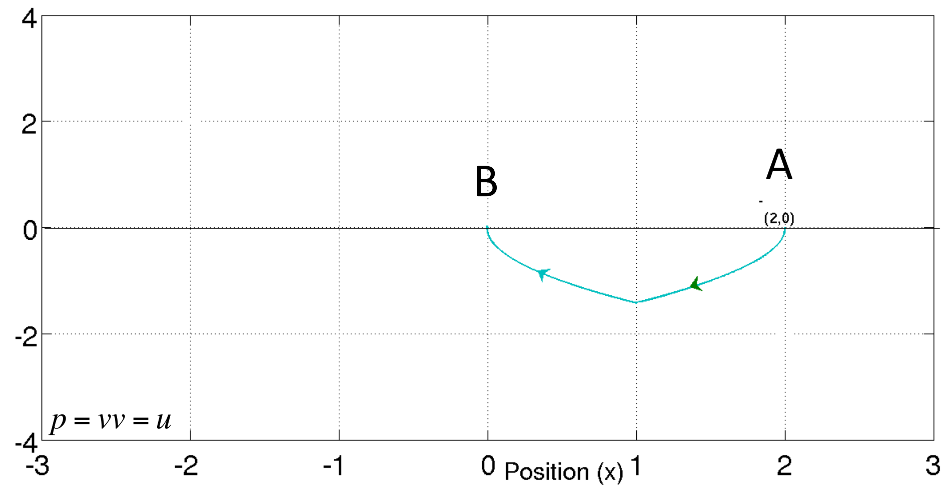
$$|u| \leq 1$$

Time Optimal Feedback







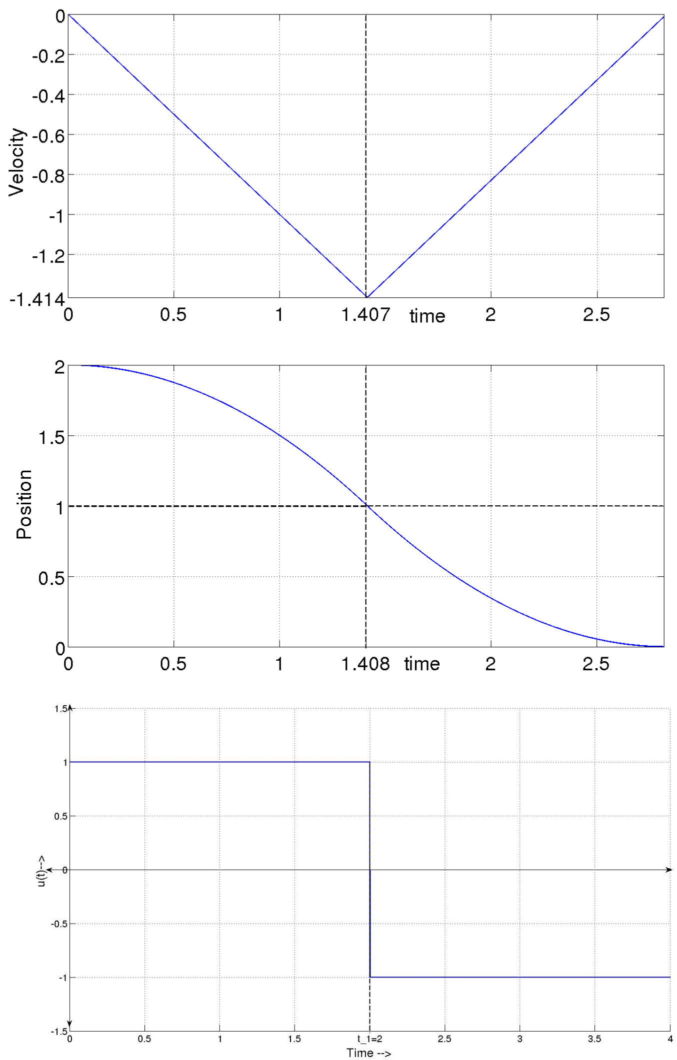


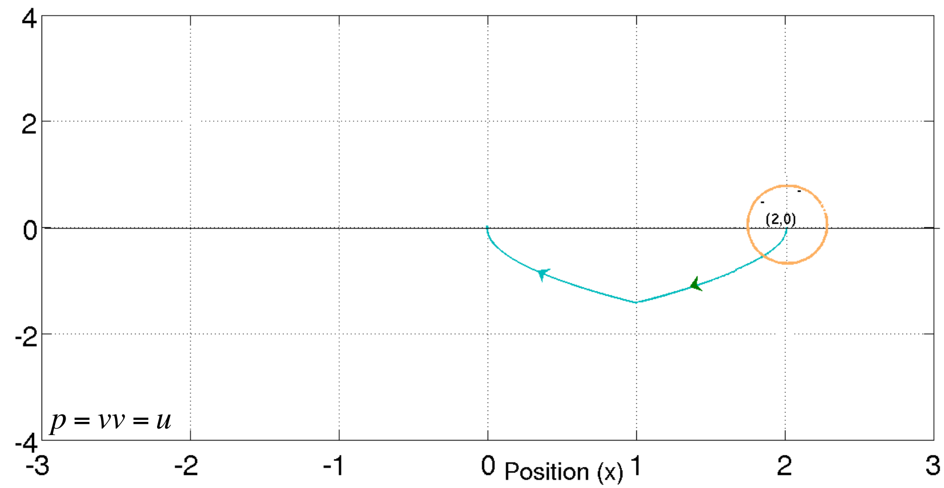
Problem: Go from A to B in minimum time with maximum allowed acceleration/deceleration = ± 1

$$\dot{p} = v;\ \dot{v} = u$$

$$|u| \leq 1$$

Time Optimal Feedback





Problem: Go from A to B in minimum time with maximum allowed acceleration/deceleration = ± 1

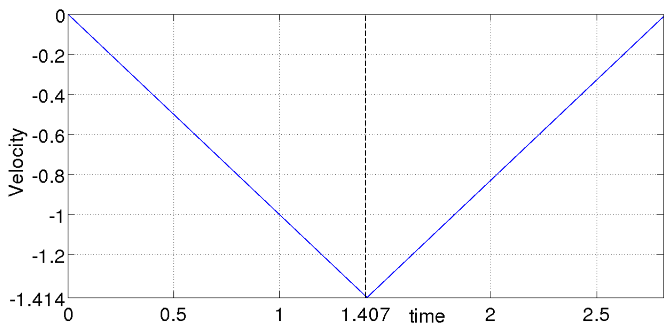
$$\dot{p} = v;\ \dot{v} = u$$

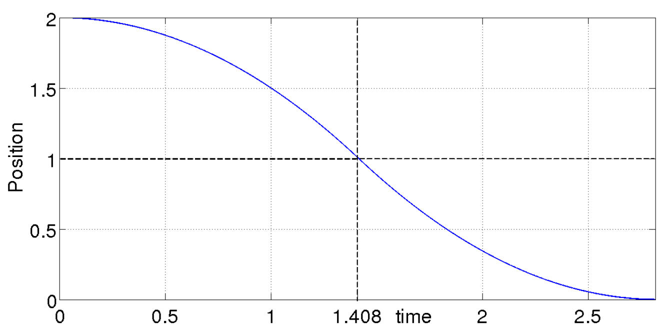
$$|u| \leq 1$$

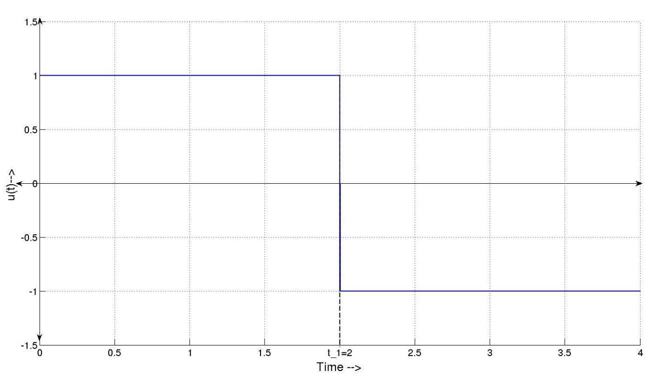
Q. What if A/B is perturbed?

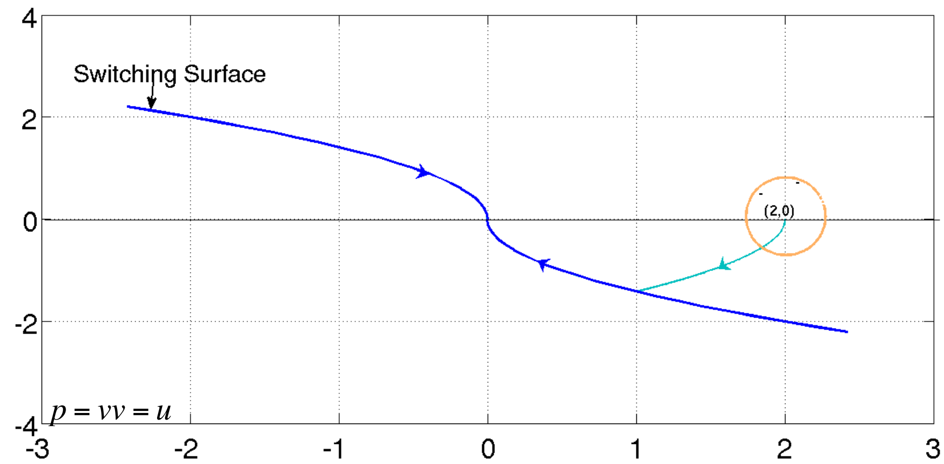
- Looks like we have to re-compute the switching instance all over again

Time Optimal Feedback









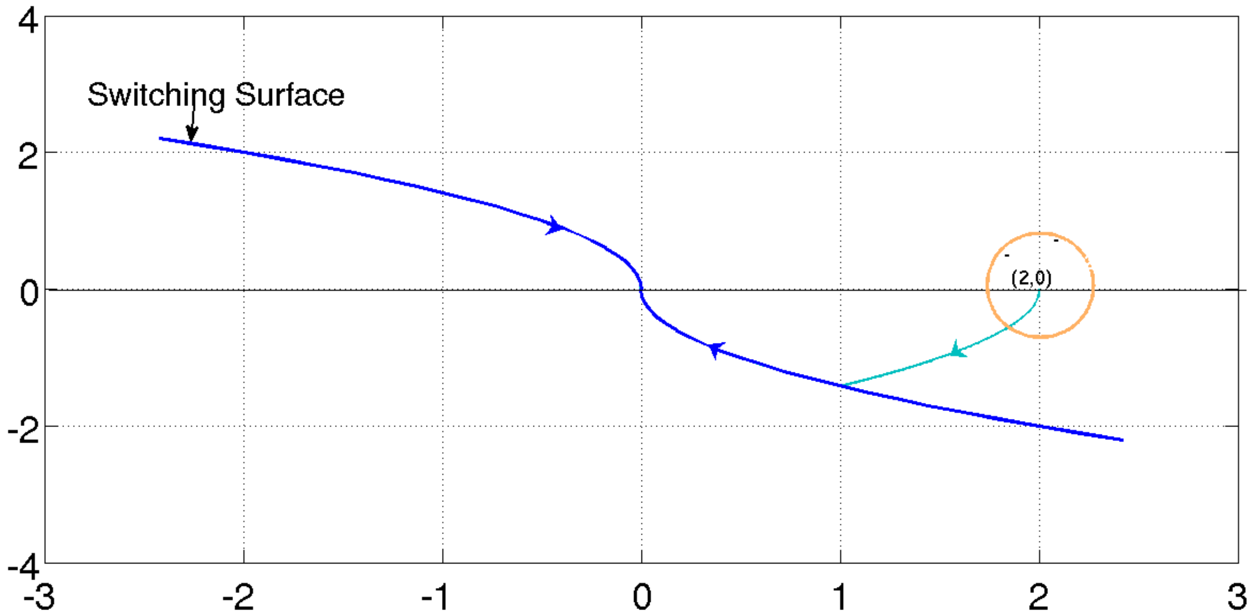
Q. What if A/B is perturbed?

- Looks like we have to re-compute the switching instance all over again

NOT REALLY – On state space, switching occurs based on the SWITCHING SURFACE – the blue line

Switching Surface for Feedback

- If *S* (the switching surface) is known feedback control can be synthesized



Feedback Algorithm:

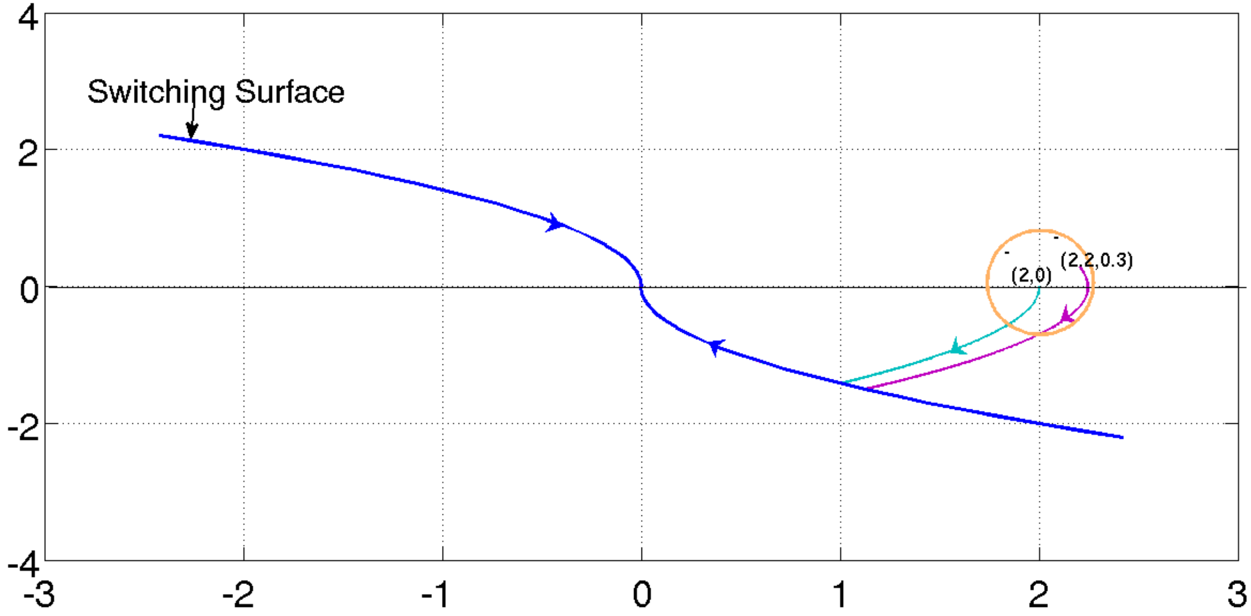
$$u = \begin{cases} +1 \text{ if } S < 0 \\ -1 \text{ if } S > 0 \end{cases}$$

And change sign as soon as

$$S = 0$$

Switching Surface for Feedback

- If *S* (the switching surface) is known feedback control can be synthesized



Feedback Algorithm:

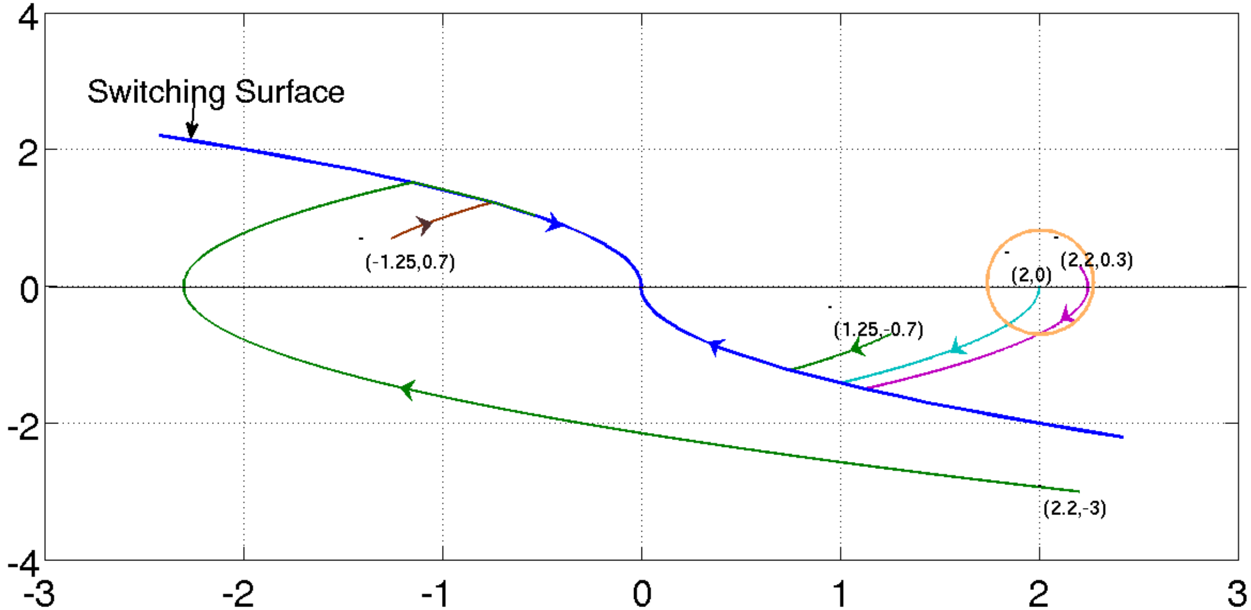
$$u = \begin{cases} +1 \text{ if } S < 0 \\ -1 \text{ if } S > 0 \end{cases}$$

And change sign as soon as

$$S = 0$$

Switching Surface for Feedback

- If S (the switching surface) is known feedback control can be synthesized



Feedback Algorithm:

$$u = \begin{cases} +1 \text{ if } S < 0 \\ -1 \text{ if } S > 0 \end{cases}$$

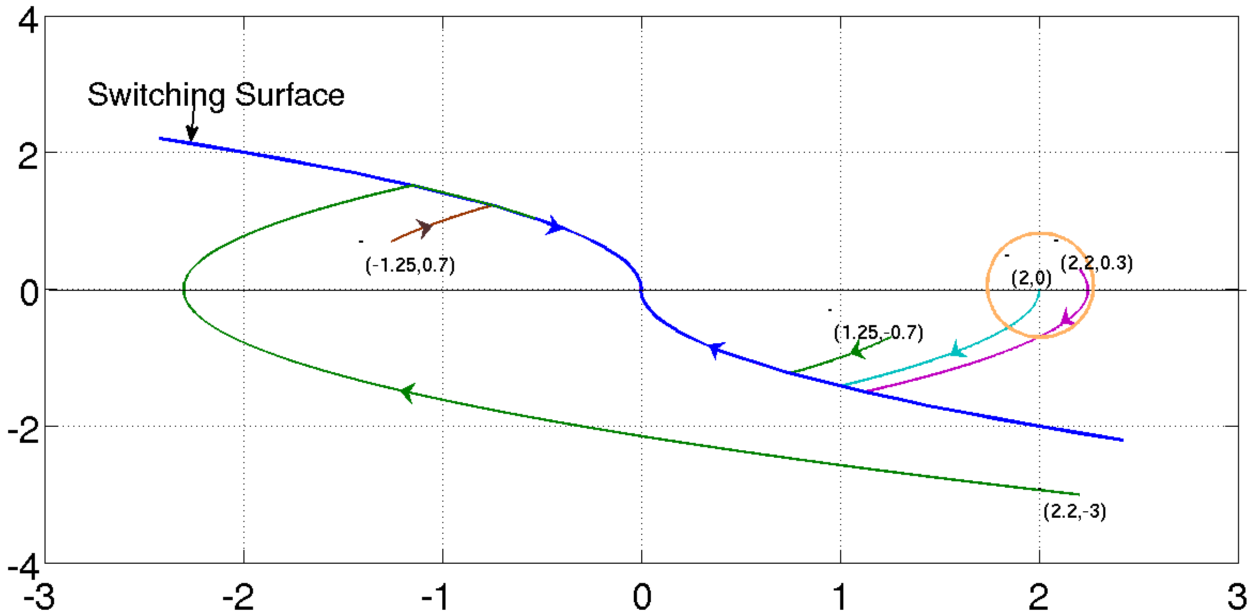
And change sign as soon as

$$S = 0$$

- The virtues of feedback over open loop are many – In fact, the initial motivation for this research was ISRO RLV RCS thruster control design

Switching Surface for Feedback

- **But for this we need an IMPLICIT expression i.e. $S(x_1,x_2) = 0$ equation for the switching surface**



Feedback Algorithm:

$$u = \begin{cases} +1 \text{ if } S < 0 \\ -1 \text{ if } S > 0 \end{cases}$$

And change sign as soon as

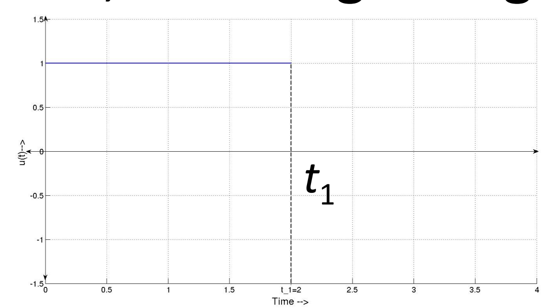
$$S = 0$$

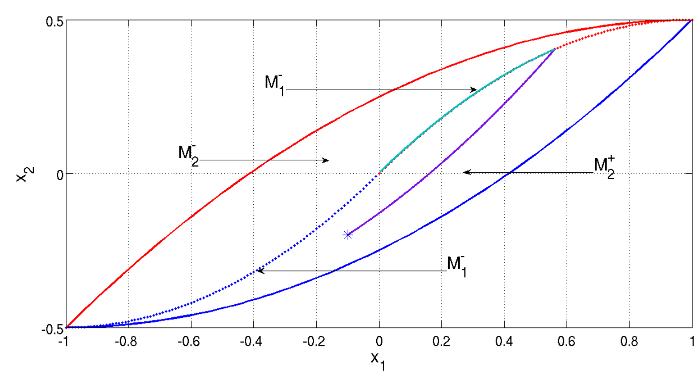
Basic Idea

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}; \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \begin{cases} \dot{x}_1 = x_1 + u \\ \dot{x}_2 = 2x_2 + u \end{cases}$$

Parametric Equations for the Switching Surface are easy – just solve above equations (for no switch, with origin target)

$$\left. \begin{aligned} 0 &= x_1 e^{t_1} \pm e^{t_1} \int_0^{t_1} e^{-\tau}\, d\tau \\ 0 &= x_2 e^{2t_1} \pm e^{2t_1} \int_0^{t_1} e^{-2\tau}\, d\tau \end{aligned} \right\}$$





***t_1* is unknown and to be eliminated.**

$$0 \le t_1 < \infty$$

Basic Idea

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}; \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \begin{cases} \dot{x}_1 = x_1 + u \\ \dot{x}_2 = 2x_2 + u \end{cases}$$

Solving:

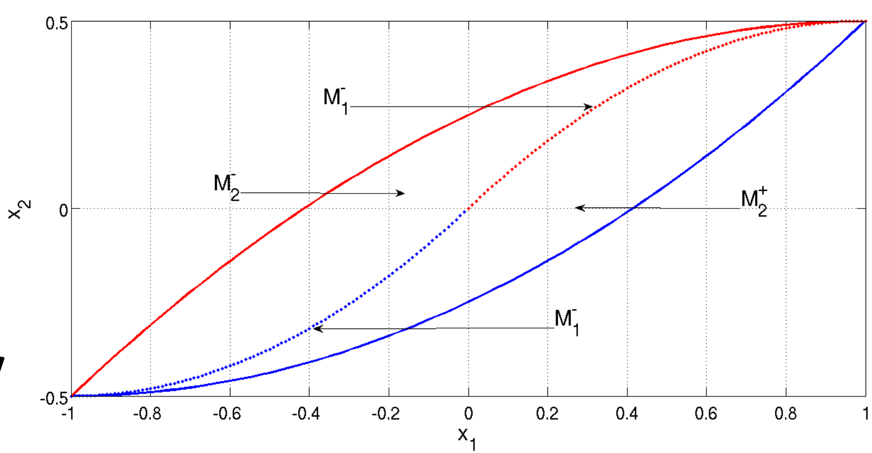
$$\left.\begin{aligned} x_1 &= \pm\left(e^{-t_1} - 1\right) \\ x_2 &= \pm\frac{(e^{-2t_1} - 1)}{2} \end{aligned}\right\} \{x_1, x_2\} =: M_1$$

are the points from which we can go to the origin without further switching i.e.

Substitute: $z_1 = e^{-t_1}$

Switching Surface: $$x_2 = \pm\frac{x_1^2}{2} + x_1$$

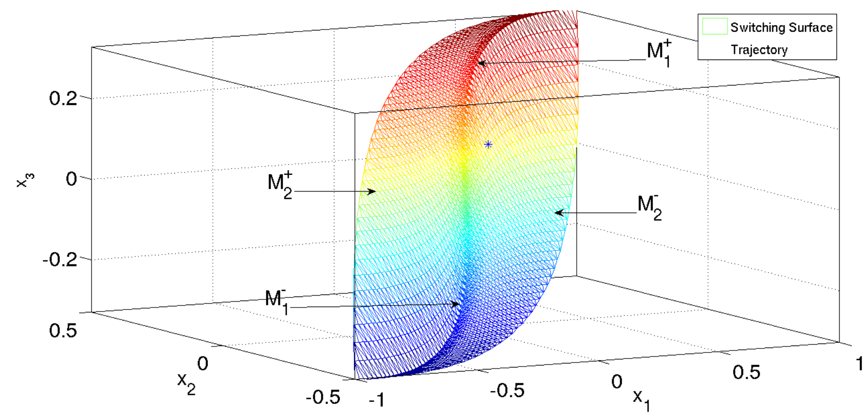
Elimination not always this easy



How to eliminate?

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix};\ B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Things get complicated fast



$$\left.\begin{aligned} x_1 &= 2e^{-t_1} - e^{-t_2} - 1 \\ x_2 &= e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2} \\ x_3 &= \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3} \\ 0 &\le t_1 \le t_2 < \infty \end{aligned}\right\}$$

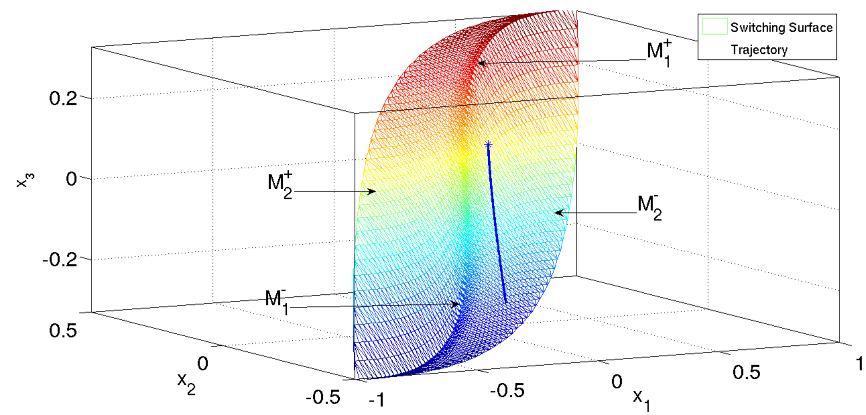
- Set of points which can reach origin in ONE switch (colored surface above)
- Parametric representation of Switching Surface

Q. How to eliminate t_1 and t_2?

How to eliminate?

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}; \; B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Things get complicated fast



$$x_1 = 2e^{-t_1} - e^{-t_2} - 1$$

$$x_2 = e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2}$$

$$x_3 = \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3}$$

$$0 \le t_1 \le t_2 < \infty$$

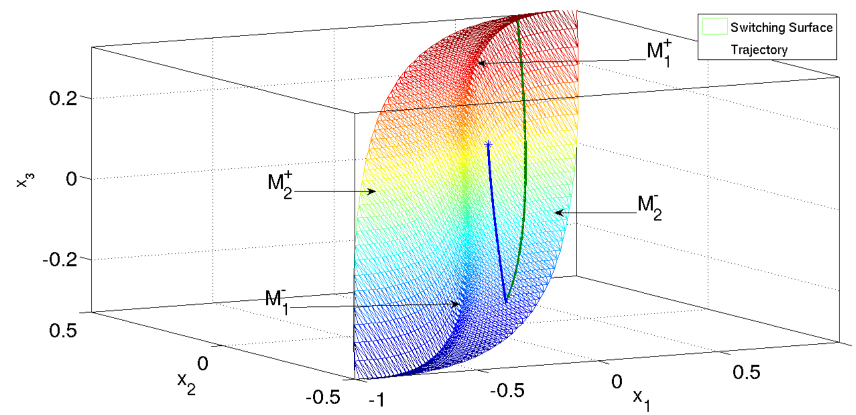
- Set of points which can reach origin in ONE switch (colored surface above)
- Parametric representation of Switching Surface

Q. How to eliminate t_1 and t_2?

How to eliminate?

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix};\ B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Things get complicated fast



$$\left.\begin{aligned} x_1 &= 2e^{-t_1} - e^{-t_2} - 1 \\ x_2 &= e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2} \\ x_3 &= \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3} \\ 0 &\le t_1 \le t_2 < \infty \end{aligned}\right\}$$

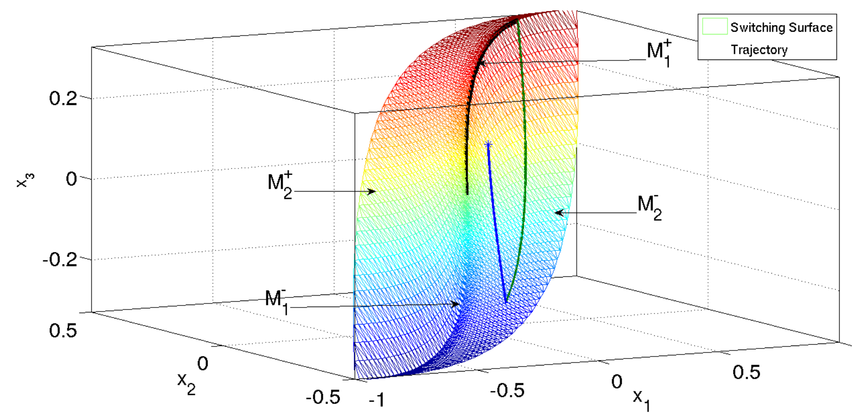
- Set of points which can reach origin in ONE switch (colored surface above)
- Parametric representation of Switching Surface

Q. How to eliminate t_1 and t_2?

How to eliminate?

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix};\ B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Things get complicated fast



$$\left.\begin{aligned} x_1 &= 2e^{-t_1} - e^{-t_2} - 1 \\ x_2 &= e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2} \\ x_3 &= \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3} \\ 0 &\le t_1 \le t_2 < \infty \end{aligned}\right\}$$

- Set of points which can reach origin in ONE switch (colored surface above)
- Parametric representation of Switching Surface

Q. How to eliminate t_1 and t_2?

How to eliminate?

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}; \; B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Substitution to polynomials

$$x_1 = 2e^{-t_1} - e^{-t_2} - 1$$

$$x_2 = e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2}$$

$$x_3 = \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3}$$

$$\underbrace{0 \le t_1 \le t_2 < \infty}_{\text{Set of points which can reach origin in ONE switch}}$$

$$z_1 = e^{-t_1}$$

$$z_2 = e^{-t_2}$$

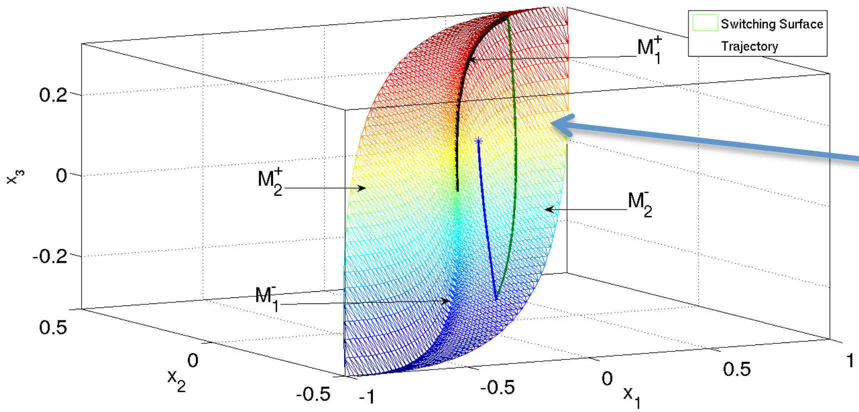
$$x_1 = 2z_1 - z_2 - 1$$

$$x_2 = z_1^2 - \frac{1}{2}z_2^2 - \frac{1}{2}$$

$$x_3 = \frac{2}{3}z_1^3 - \frac{1}{3}z_2^3 - \frac{1}{3}$$

$$\underbrace{0 < z_2 \le z_1 \le 1}_{\text{Polynomial Parametric representation of Switching Surface}}$$

How to eliminate?



$$g(x_1, x_2, x_3) = 0$$

+ the inequalites

Eliminate z's

$$x_1 = 2e^{-t_1} - e^{-t_2} - 1$$

$$x_2 = e^{-2t_1} - \frac{1}{2}e^{-2t_2} - \frac{1}{2}$$

$$x_3 = \frac{2}{3}e^{-3t_1} - \frac{1}{3}e^{-3t_2} - \frac{1}{3}$$

$$0 \le t_1 \le t_2 < \infty$$

Set of points which can reach origin in ONE switch

$$z_1 = e^{-t_1}$$

$$z_2 = e^{-t_2}$$

$$x_1 = 2z_1 - z_2 - 1$$

$$x_2 = z_1^2 - \frac{1}{2}z_2^2 - \frac{1}{2}$$

$$x_3 = \frac{2}{3}z_1^3 - \frac{1}{3}z_2^3 - \frac{1}{3}$$

$$0 < z_2 \le z_1 \le 1$$

Polynomial Parametric representation of Switching Surface

Elimination Algorithm

- Form an Ideal:

$$J = \left\langle x_1 - 2z_1 + z_2 + 1, x_2 - z_1^2 + \frac{1}{2}z_2^2 + \frac{1}{2}, x_3 - \frac{2}{3}z_1^3 + \frac{1}{3}z_2^3 + \frac{1}{3} \right\rangle$$

- Compute Groebner basis *G* of *J* with lexicographic ordering $z_1 \succ z_2 \succ x_1 \succ x_2 \succ x_3$.
- The element $g \in G \cap Q[x_1, x_2, x_3]$ defines the smallest variety containing the parametric representation of the switching surface

- Inequality constraints: z_1 and z_2 can be computed in terms of the states (skipped here)

Example

Example

$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$ and $B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$.

- Form an ideal $J = \langle x_1 - 2z_1 + z_2 + 1, x_2 - z_1^2 + \frac{1}{2}z_2^2 + \frac{1}{2}, x_3 - \frac{2}{3}z_1^3 + \frac{1}{3}z_2^3 + \frac{1}{3}\rangle$.
- Using Elimination Algorithm compute $g_2^+(x_1,x_2,x_3) = 0$.
- Also compute $z_1 = \frac{-(-x_1^3-3x_1^2-3x_1+3x_3)}{(3x_1^2+6x_1-6x_2)}$ and $z_2 = \frac{-(x_1^3+3x_1^2-6x_1x_2-6x_2+6x_3)}{(3x_1^2+6x_1-6x_2)}$
- Thus $M_2^+ = \{(x_1,x_2,x_3) : g_2^+(x_1,x_2,x_3) = 0, 0 < z_2 \leq z_1 \leq 1\}$

Guarantees

- $g(x_1,x_2,x_3)$ can be 'cut-out' to recover the actual switching surface.
- Switching based on $g(x_1,x_2,x_3)$ works.
- Inaccurate/practical switching converges to arbitrary neighborhood of origin
- The null controllable set can be algebraically computed.
- Limit cycles occur for most non-origin targets - time period can be computed

The Good: Time Optimal + works for entire null controllable region + feedback control

The Bad – only works for rational/imag eigenvalues-
recently some hope of removing this limitation

Plan

Computation of Time Optimal **Feedback** using Groebner Basis

(Feedback) Pursuit-Evasion Games

Time Optimal Multi - agent Consensus (complete graph)

Time Optimal Leader Tracking in Multi-agent systems (directed graphs)

Pursuit Evasion Games

Time Optimal (*Feedback*) Pursuit Evasion

- Optimal Feedback strategy was hard to compute: can be computed now (for rational/imaginary eigenvalues)

$$\dot{x}_e = Ax_e + Bu_e; \quad |u_e| \le \alpha$$

$$\dot{x}_p = Ax_p + Bu_p \quad |u_p| \le \beta$$

Problem: 'e' tries to maximize and 'p' tries to minimize the time T when

$$x_e(T) = x_p(T)$$

Pursuit Evasion Games - Assumptions

- P and E do not know each others strategies
- Each needs to guard against worst possible strategies of the other
- Proposed pursuer control strategy (similarly for evader):

$$u_p^*(t) = \arg \min_{|u_p| \leq \beta} \left(\max_{|u_e| \leq \alpha} T(u_p, u_e) \right)$$

$T(u_p, u_e)$ is capture time

$u_p^*(t)$: min-max control strategy for pursuer

Trick: Difference System

- Difference System:

$$\dot{x}(t) = Ax(t) + Bu_{ep}(t)$$

where, $x(t) = x_p(t) - x_e(t)$ and $u_{ep}(t) = u_p(t) - u_e(t)$

- Capture condition: $x_p(t) = x_e(t) \implies x(t) = 0$ for some $t \geq T$
- Objective function:

$$J = \int_0^T 1dt = T(u_p, u_e)$$

- Min-max strategies: u_p^* and u_e^* such that

$$J^* = T(u_p^*, u_e^*) = \min_{|u_p| \leq \beta} \max_{|u_e| \leq \alpha} T(u_p, u_e)$$

Bryson and Ho (1969)

- Hamiltonian: $H = \lambda^T(Ax + B(u_p - u_e)) + 1$
- Necessary condition for stationarity of J

$$\dot{\lambda} = -\frac{\partial H}{\partial t} = -A^T\lambda \qquad \lambda(0) = \lambda_0$$

$$H^* = \min_{|u_p|\leq\beta} \max_{|u_e|\leq\alpha} (\lambda^T(Ax + B(u_p - u_e)) + 1)$$

- Optimal inputs:

$$u_e^*(t) = \arg\max_{u_e} H(u_p, u_e) = -\alpha sign(\lambda_0^T e^{-At} B)$$

$$u_p^*(t) = \arg\min_{u_p} H(u_p, u_e^*) = -\beta sign(\lambda_0^T e^{-At} B)$$

- u_p^* and u_e^* should have same sign and switch according to same switching function.

Switching Surface

$$u_e^*(t) = \arg\max_{u_e} H(u_p, u_e) = -\alpha sign(\lambda_0^T e^{-At} B)$$

$$u_p^*(t) = \arg\min_{u_p} H(u_p, u_e^*) = -\beta sign(\lambda_0^T e^{-At} B)$$

A switching surface corresponding to this switching function can be computed by considering the difference system

- The "difference" system:

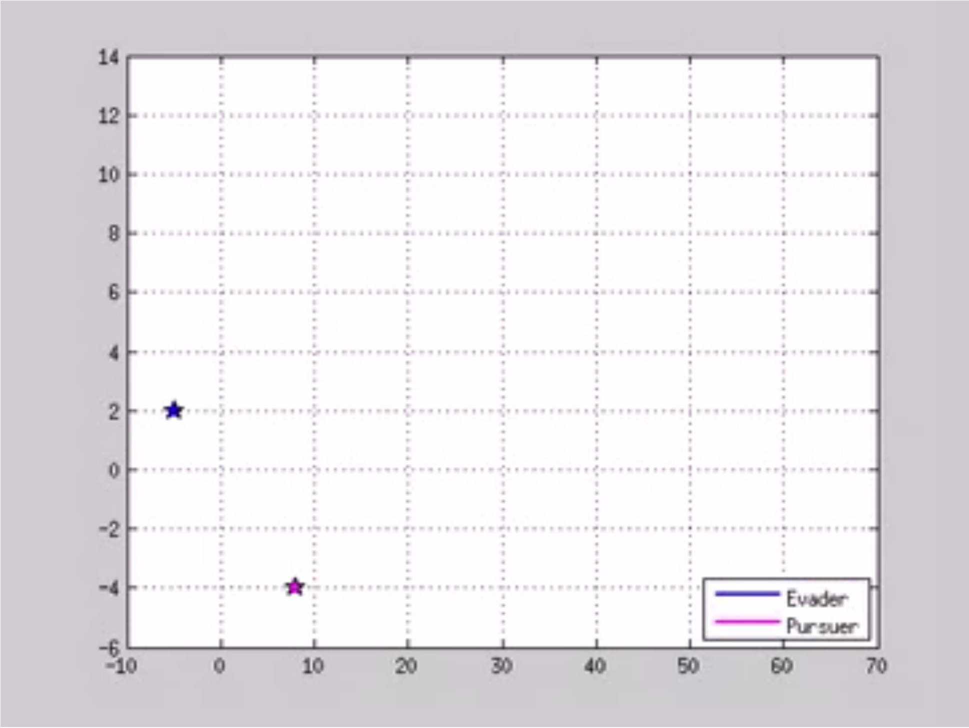
$$D : \dot{x}_p - \dot{x}_e = A\left(x_p - x_e\right)x_e + B\left(u_p - u_e\right); \quad \left|u_p - u_e\right| \le \beta - \alpha$$

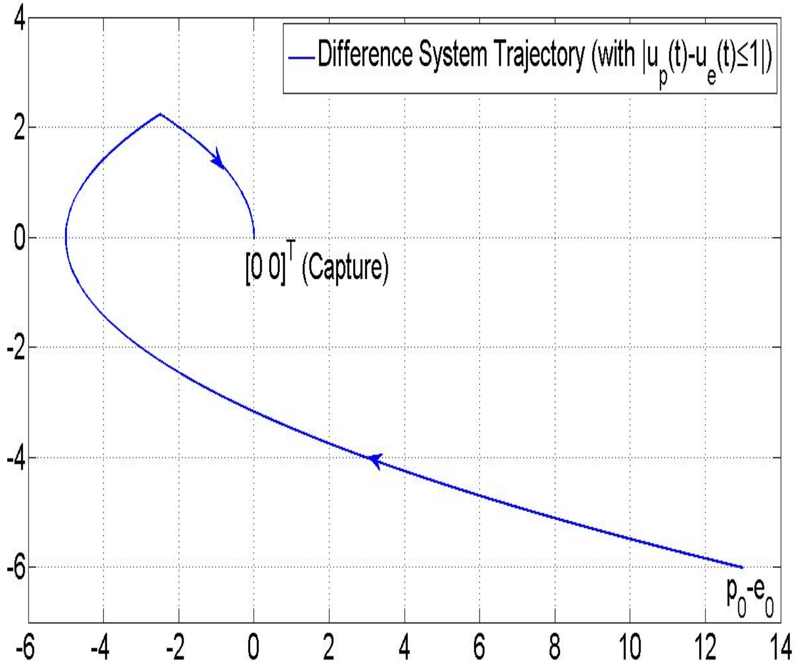
- Capture when D reaches origin = Time Optimal transfer to origin with the changed input bound
- Feedback pursuit-evasion strategies can be computed
- Capture can be guaranteed if $\alpha < \beta$

Example Pursuit Evasion

$$\dot{p}_p = v_p;\ \dot{v}_p = u_p$$
$$|u_p| \le 2$$

$$\dot{p}_e = v_e;\ \dot{v}_e = u_e$$
$$|u_e| \le 1$$





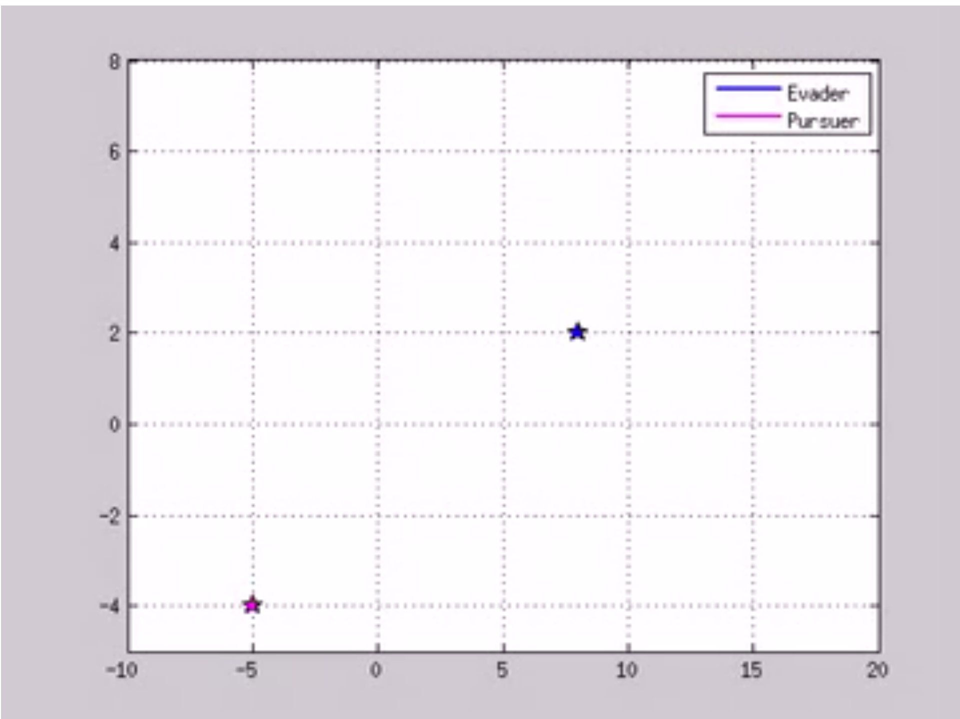
?

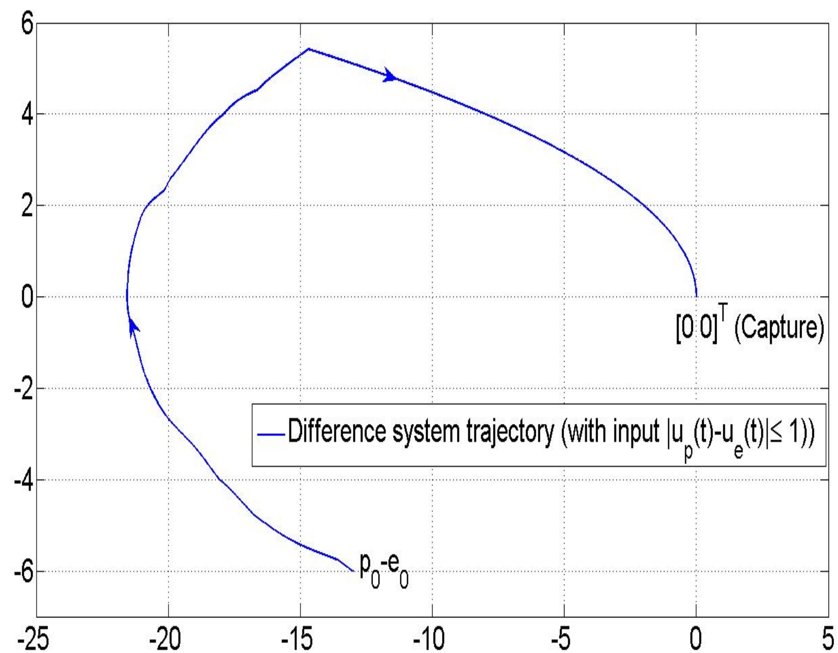
‘p’ plays **min-max feed**back while e plays **max-min feedback** strategy, but still gets captured.

Example Pursuit Evasion

$$\dot{p}_p = v_p;\ \dot{v}_p = u_p \qquad \dot{p}_e = v_e;\ \dot{v}_e = u_e$$

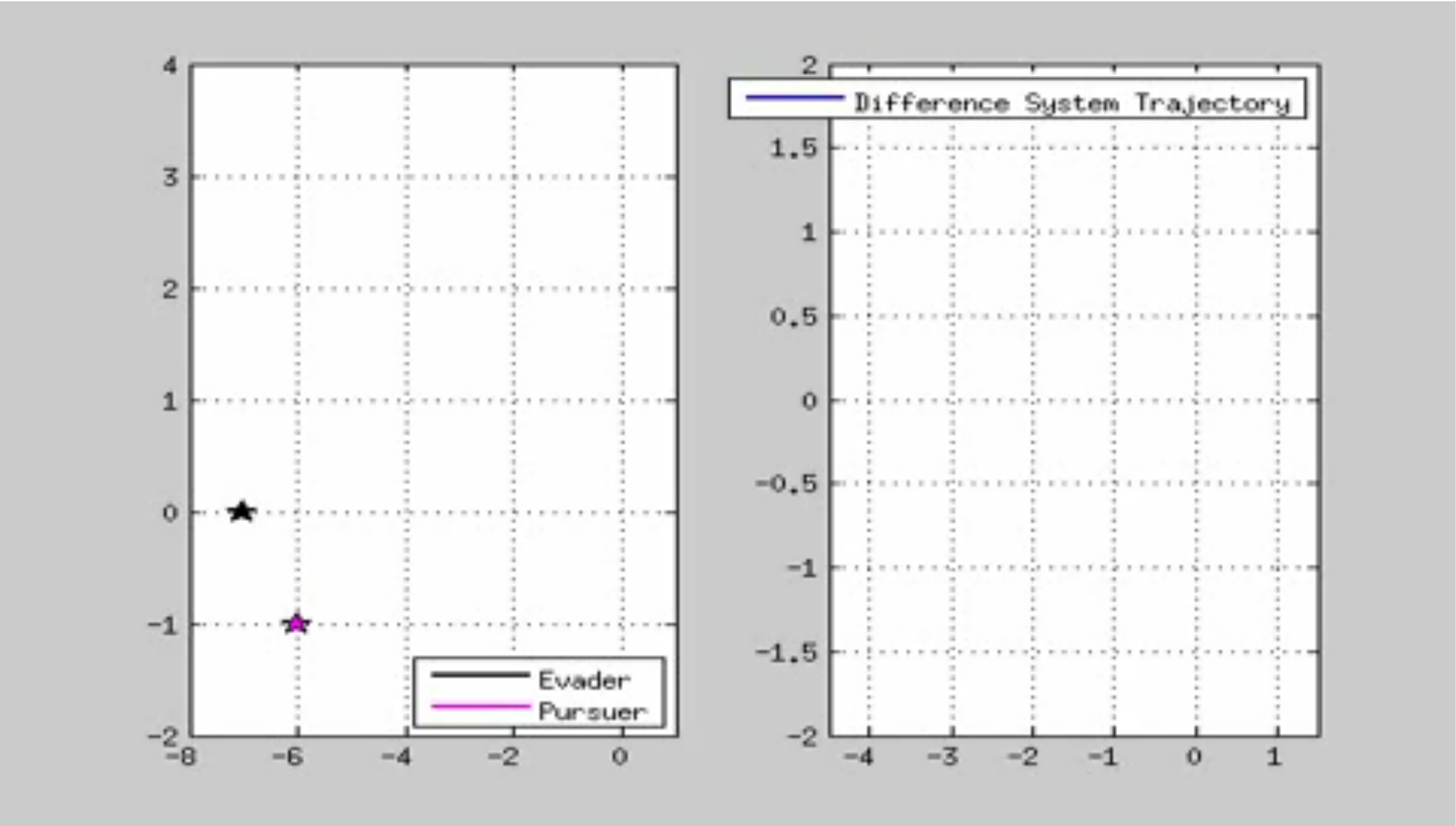
$$|u_p| \le 2 \qquad |u_e| \le 1$$





'p' plays **min-max feed**back while e plays **NON-OPTIMAL** strategy, gets captured earlier.

Successful Escape

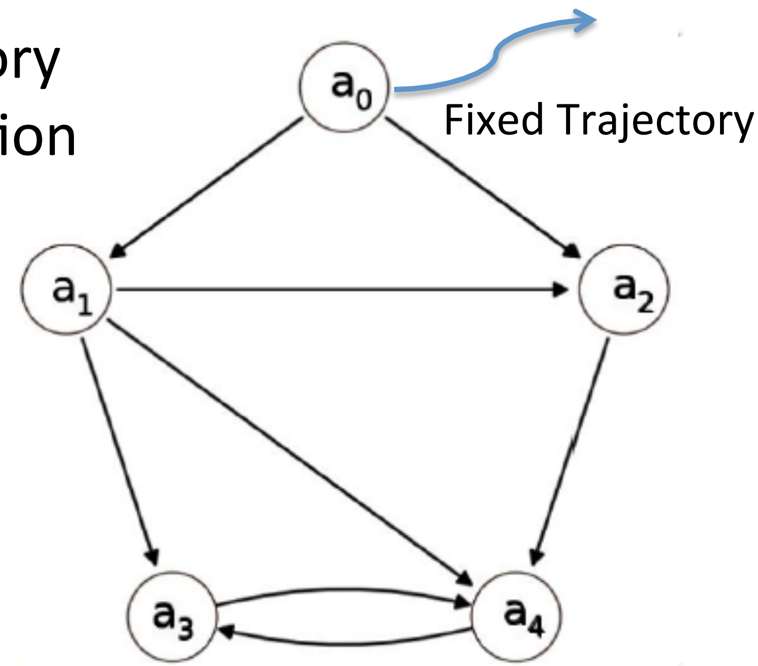


Time Optimal Leader Tracking in Multi-agent systems

Consensus Tracking for Multiple Agents

Assumptions:

- All agents are stable with identical dynamics and input bounds
- a_0 is the leader
- a_0 moves along a given fixed trajectory
- State information flows in the direction of the arrows (directed graph)

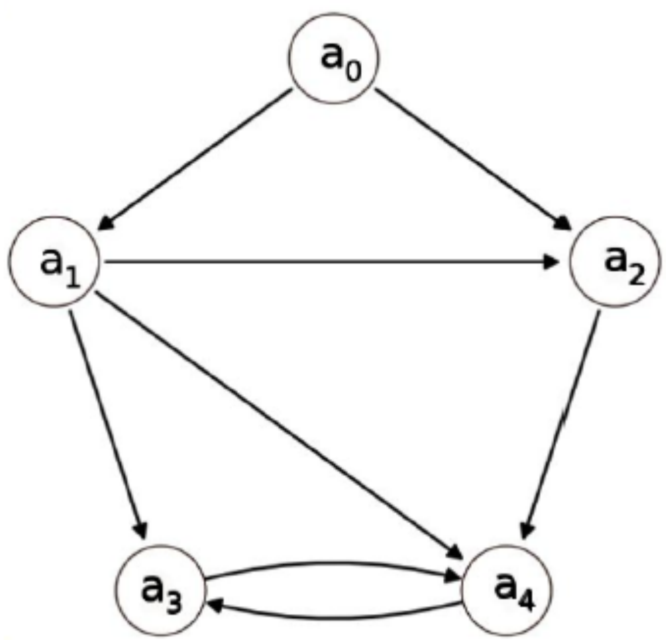


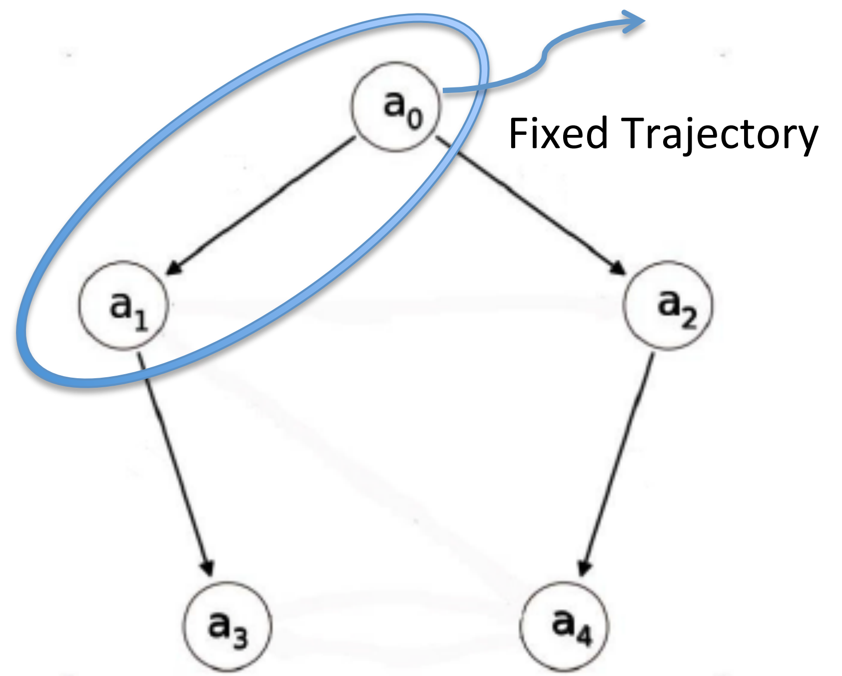
Problem: Find the *local* control laws for $a_1,\ldots,a_4$ such that all of them track a_0's trajectory in the minimum time possible.

Assumption: a_0 is "capturable" by the followers

Min-Max Pursuit

- Identify a directed spanning tree rooted at the leader (later)
- Apply the min-max pursuit policy for each follower
- For example: consider (a_0,a_1) pair and apply the min-max pursuit policy for a_1



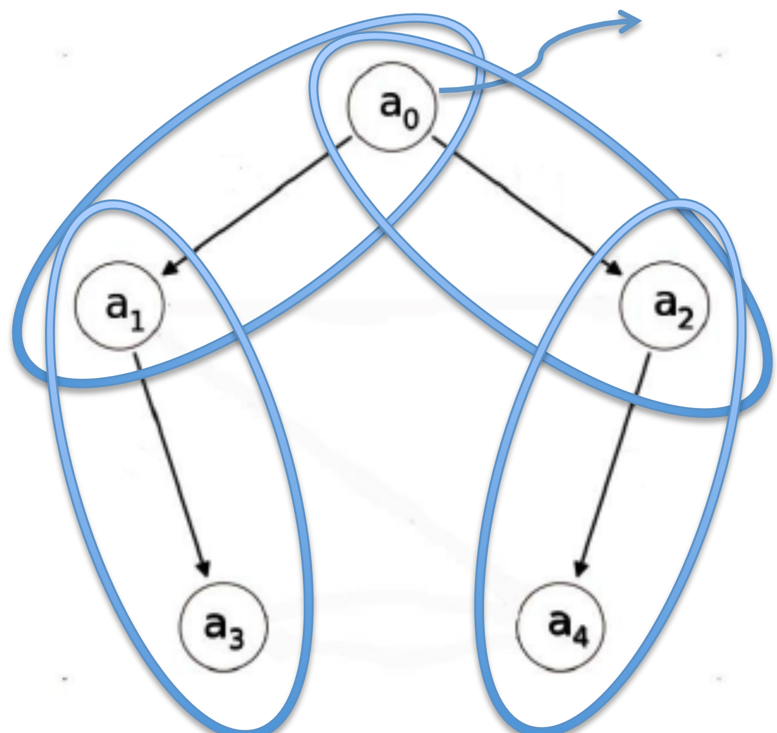


Min-Max Pursuit

- Identify a directed spanning tree rooted at the leader (later)
- Apply the min-max pursuit policy for each follower
- For example: consider (a_0,a_1) pair and apply the min-max pursuit policy for a_1
- Similarly for all pairwise leader-follower pairs
- For each pair the upper bound on capture time is given by:

$$\bar{t}_{ij} = \min_{|u_i| \le \beta_i} \max_{|u_j| \le \beta_j} T(u_i, u_j)$$

- But there is no upper bound for identical bounds on the leader and follower



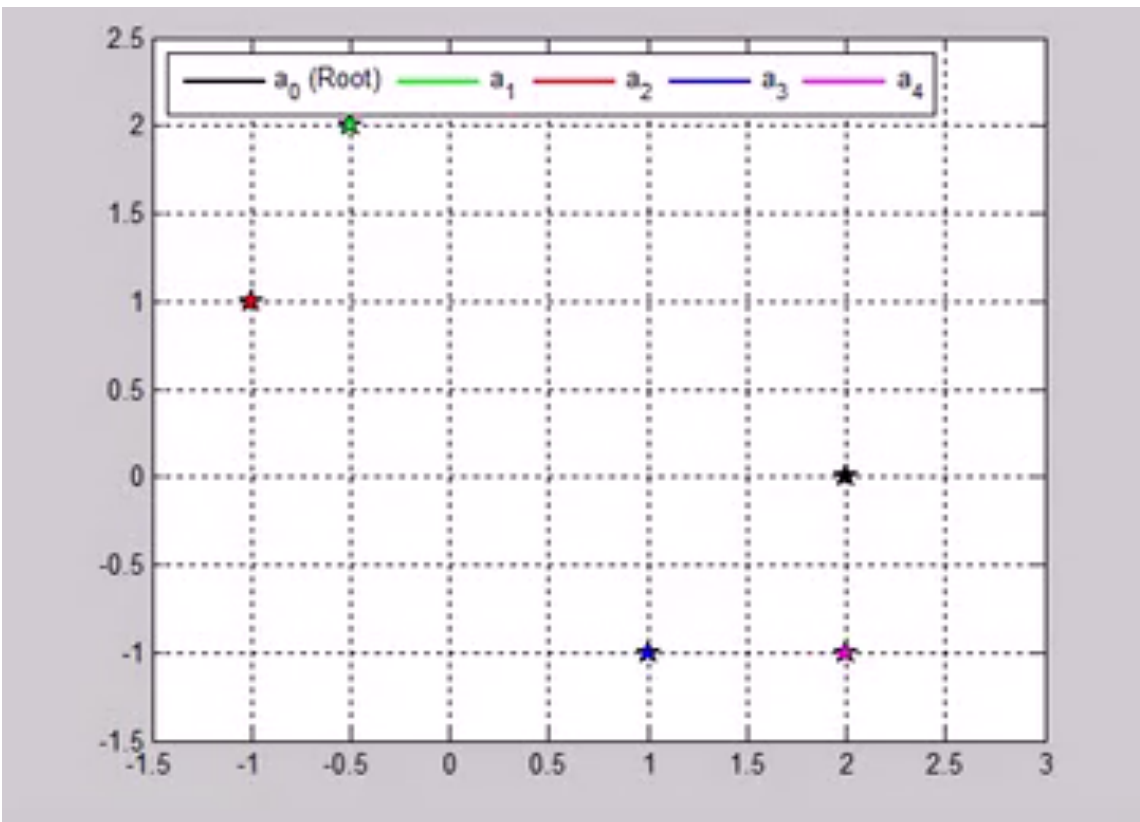
Min Time Leader Tracking

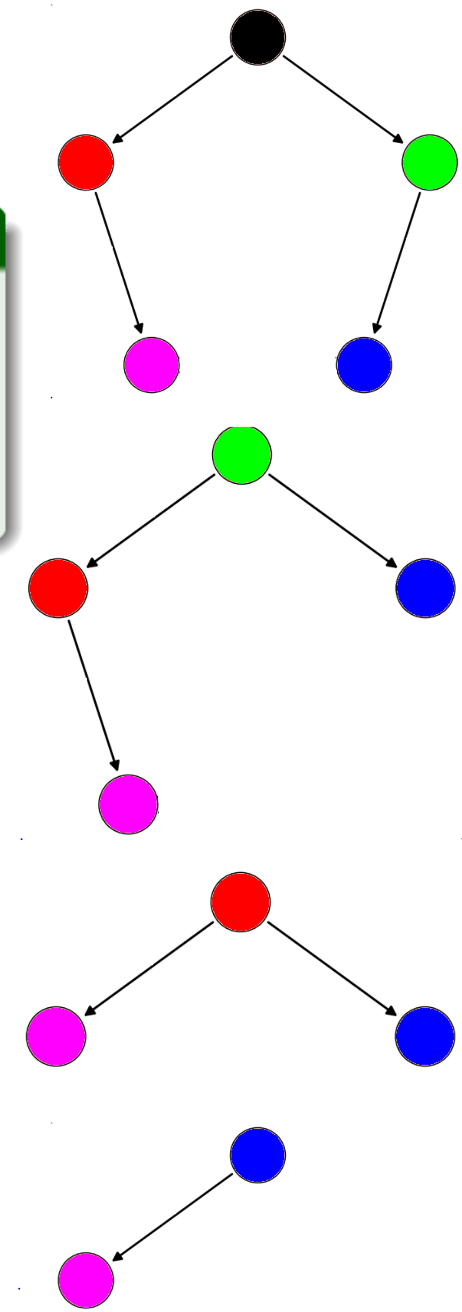
Example

5-agent systems communicating over a tree. Agent dynamics is given by

$$\dot{x}_i(t) = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} x_i(t) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u_i(t) \qquad \text{for } i = 0, 1, \ldots, 4$$

$|u_0(t)| \leq 1$ and $|u_i(t)| \leq 3$ for $i = 1, \ldots, 4$





Selection of Directed Spanning Tree

- We have an algorithm which does this with local information (skipped here)
- How does the selection of the spanning tree affect time to consensus?
- Does using information from multiple leaders help reduce time to consensus?
- How do cycles (if allowed to remain) affect time to consensus?

Plan

Computation of Time Optimal **Feedback** using Groebner Basis

(Feedback) Pursuit-Evasion Games

Time Optimal Multi - agent Consensus (complete graph)

Time Optimal Leader Tracking in Multi-agent systems (directed graphs)

Multi Agent: Minimum Time Consensus

Consensus: Many 'agents' try to reach a previously unspecified point autonomously

Min Time Consensus

- **Problem**: Consider N double integrator 'agents' communicating over a complete graph

$$\dot{x}_i(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A} x_i(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{b} u_i(t) \qquad i = 1, \ldots, N$$

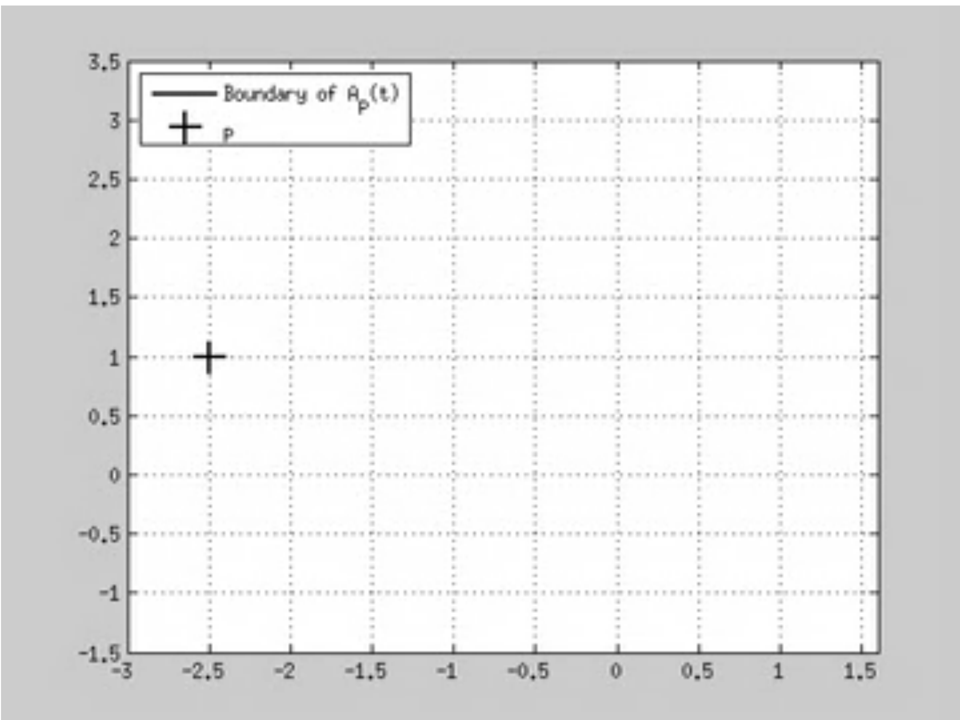
with $x_i(t) = \begin{bmatrix} r_i(t) \\ v_i(t) \end{bmatrix}$, $x_i(0) = x_{i0} = \begin{bmatrix} r_{i0} \\ v_{i0} \end{bmatrix}$ and $|u_i(t)| \leq 1$.

$$\boxed{\begin{array}{l} \text{Find } \bar{x} \text{ and min } \bar{t} \text{ such that, for all } i,j \\ x_i(\bar{t}) = \bar{x} \text{ and } x_i(t) = x_j(t) \text{ for all } t \geq \bar{t} \end{array}}$$

Attainable Set

<u>Attainable Set from p at time t</u>

$$\mathscr{A}_p(t) = \left\{ x : x = e^{At}p + \int_0^t e^{A(t-\tau)} b u(\tau) d\tau, \;\; \forall u(t) : |u(t)| \le 1 \right\}$$



- Each point on the boundary can be reached using bang-bang time optimal control.
- **Polynomial Expressions for the boundaries can be obtained**

Main Idea

- For consensus, it would seem that the attainable sets of all the agents need to intersect, i.e. for consensus at time *t*

$$\bigcap_{1\le i\le N} \mathscr{A}_i(t) \neq \phi \;\; (\mathscr{A}_i(t) := \mathscr{A}_{x_{i0}}(t))$$

- Solution requires solving large set of coupled polynomial equations and inequalities
- Computation cannot be distributed between the agents

Helly's theorem comes to the rescue

Let F be a finite family of **convex sets** in $\mathbb{R}^n$, containing **at least** $n+1$ elements. **If every** $n+1$ sets of F have a point in common, **then all the sets of** F have a point in common.

Parallel Computation

$\bar{t}_{ijk}$: Minimum time to consensus for agents $\{a_i, a_j, a_k\}$

Lemma: $\bar{t} = \max_{1 \leq i,j,k \leq N} \bar{t}_{ijk}$

Theorem:

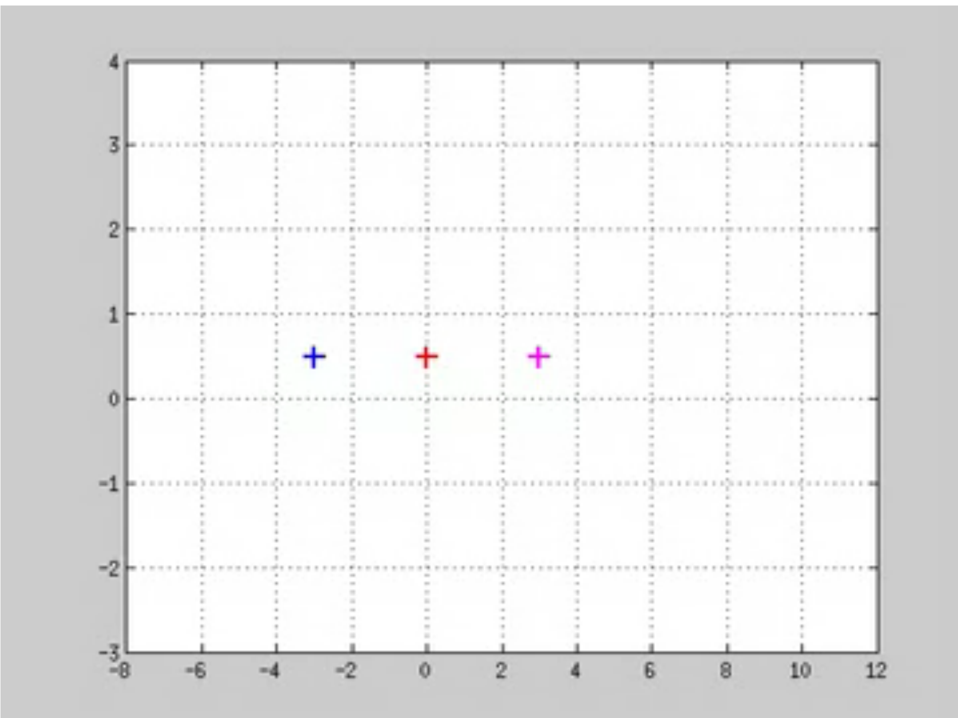
Let $\{a_p, a_q, a_r\}$ be the triple of agents such that $\bar{t}_{pqr} = \max_{1 \leq i,j,k \leq N} \bar{t}_{ijk}$. Then the minimum time to consensus $\bar{t} = \bar{t}_{pqr}$ and the corresponding consensus point $\bar{x} = \bar{x}_{pqr}$.

This means:

- We have to check ${}^{N}C_3$ combinations for the max.
- But each of these computations are decoupled from the other – can be distributed between the agents

Two ways to three agent consensus

Case 1: $\bar{t}_{ijk} = \bar{t}_{ij} = \max\{\bar{t}_{ij}, \bar{t}_{jk}, \bar{t}_{ik}\}$ i.e $\bar{x}_{ij} \in \mathscr{A}_k(\bar{t}_{ij})$



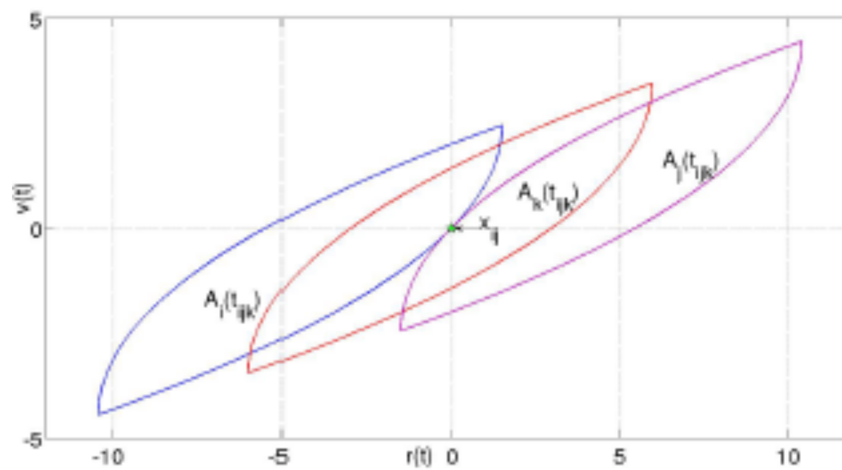
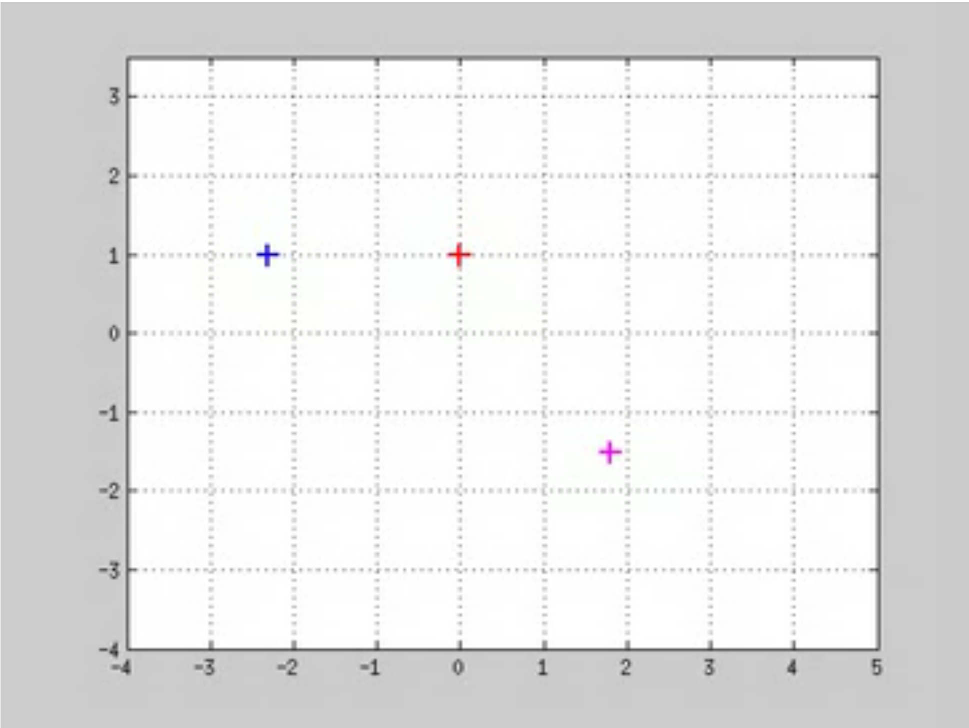
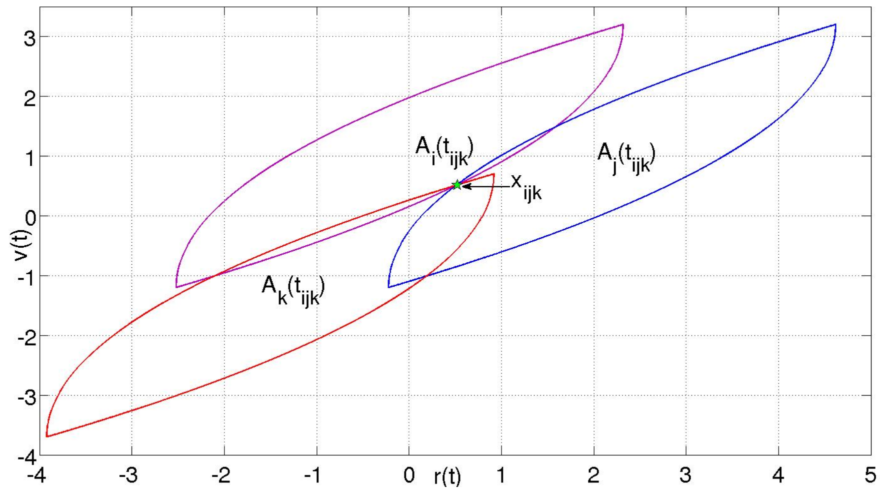


Figure : Case 1

Two ways to three agent consensus

Case 2: $\bar{t}_{ijk} > \max\{\bar{t}_{ij}, \bar{t}_{jk}, \bar{t}_{ik}\}$ i.e. $\bar{x}_{ij} \notin \mathscr{A}_k(\bar{t}_{ij})$

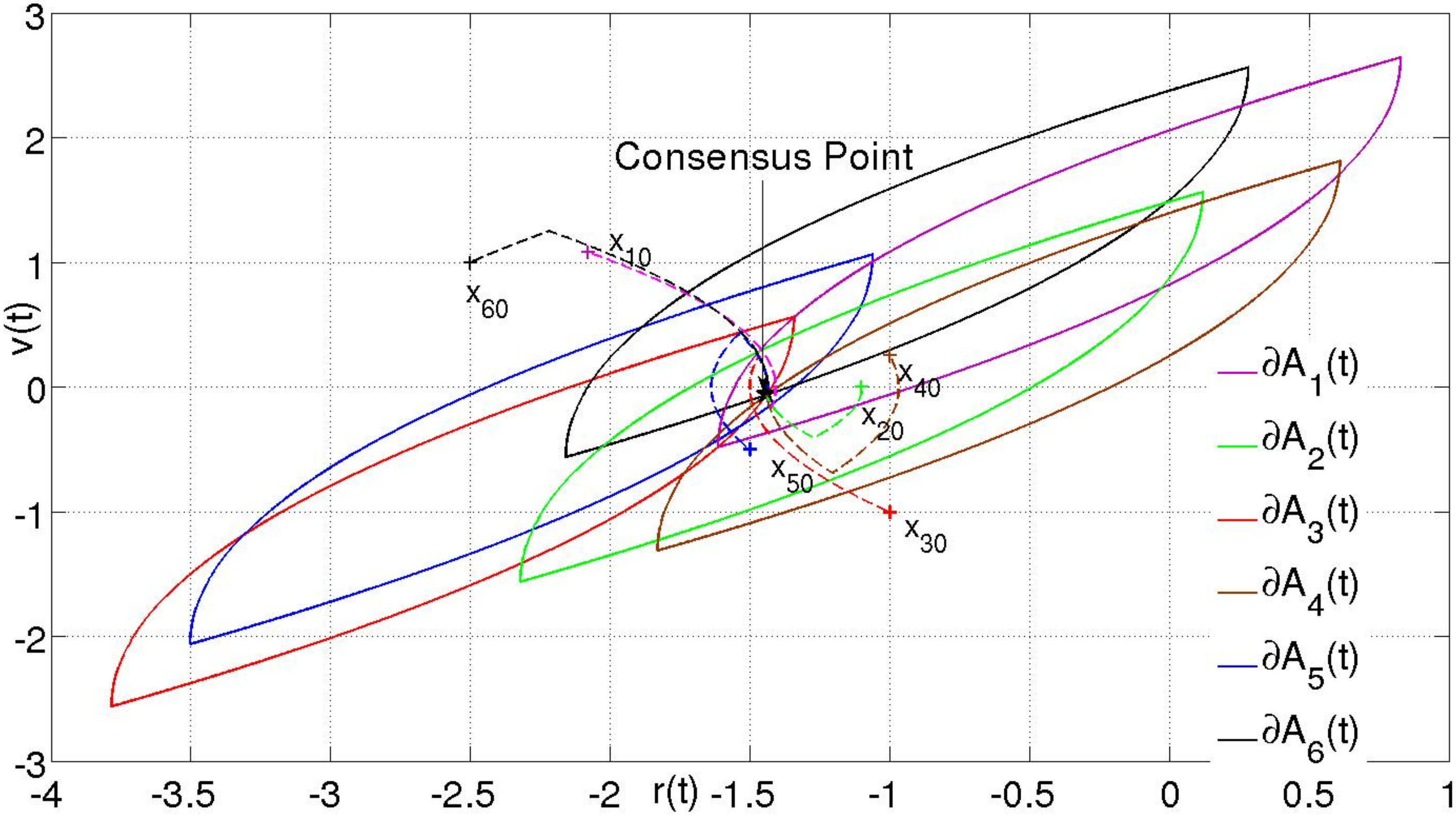




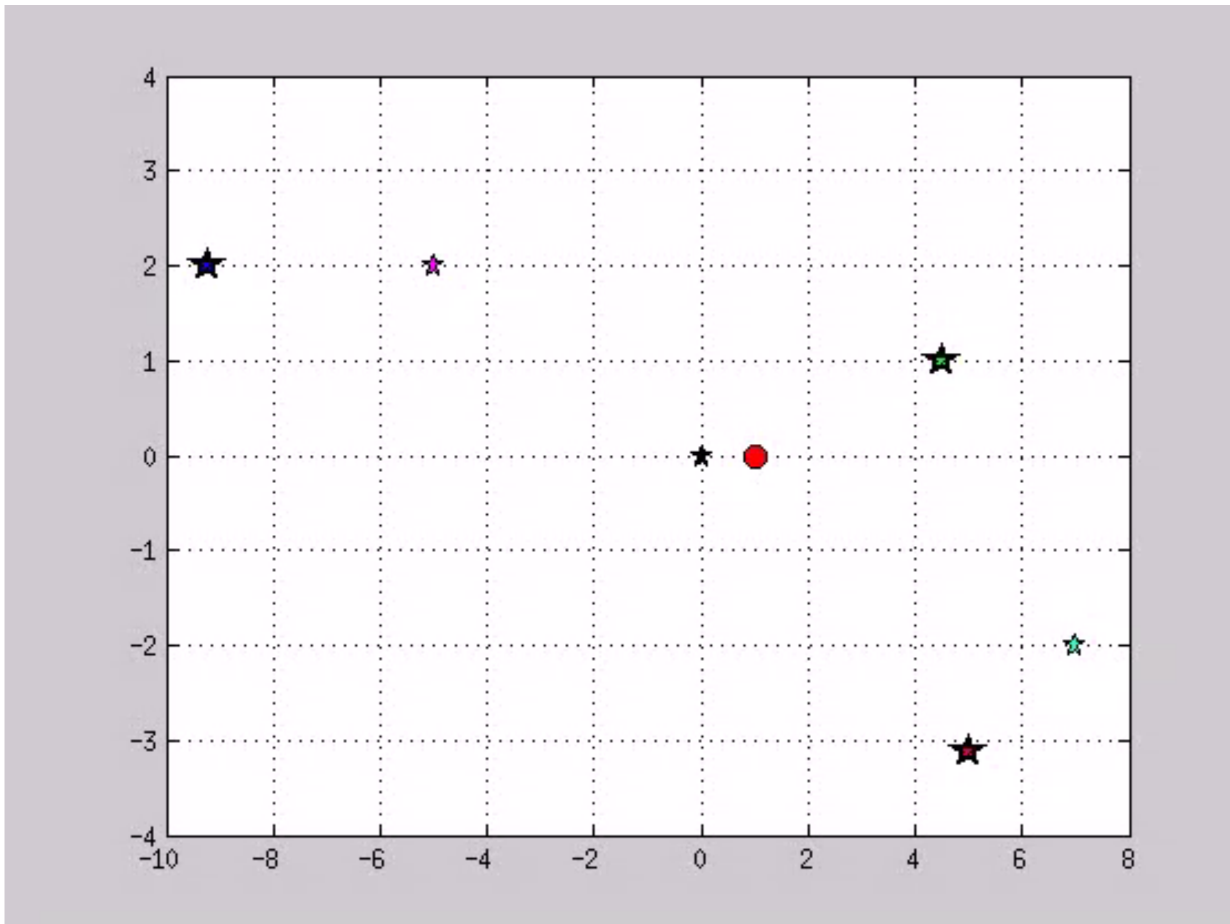
Computation

- Algebraic formula for computation in both cases have been derived.
- Can be used to directly compute the min time and the consensus point based on the current states
- Proposed algorithm can handle disturbances to the agents by dynamically (feedback) re-computing the target point
  - Then full computation (${}^{N}C_3/N$) needs to be done only once at the beginning

Six agents min time consensus

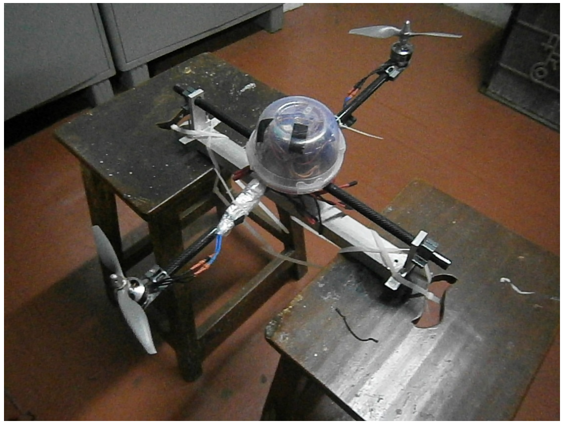


Min time consensus on R^1



Anything useful?

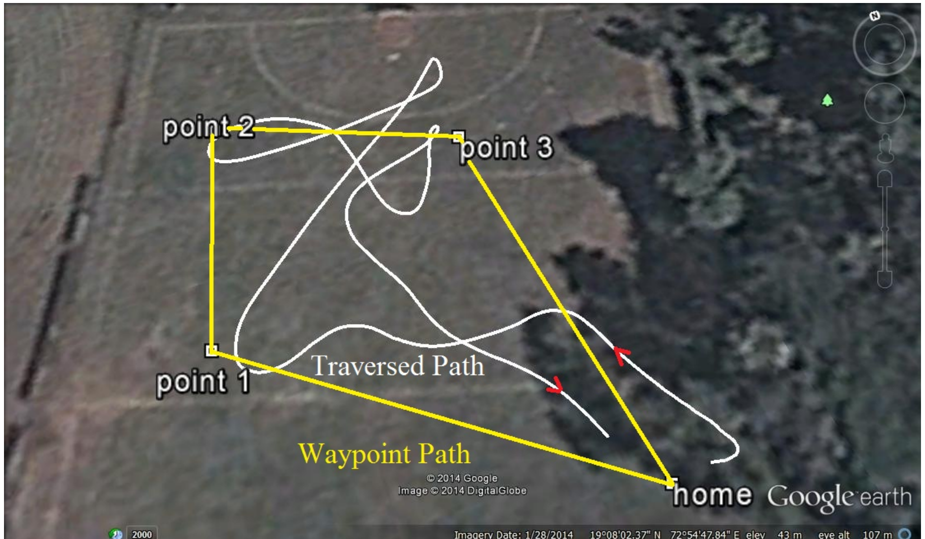
Quadcopter testbed



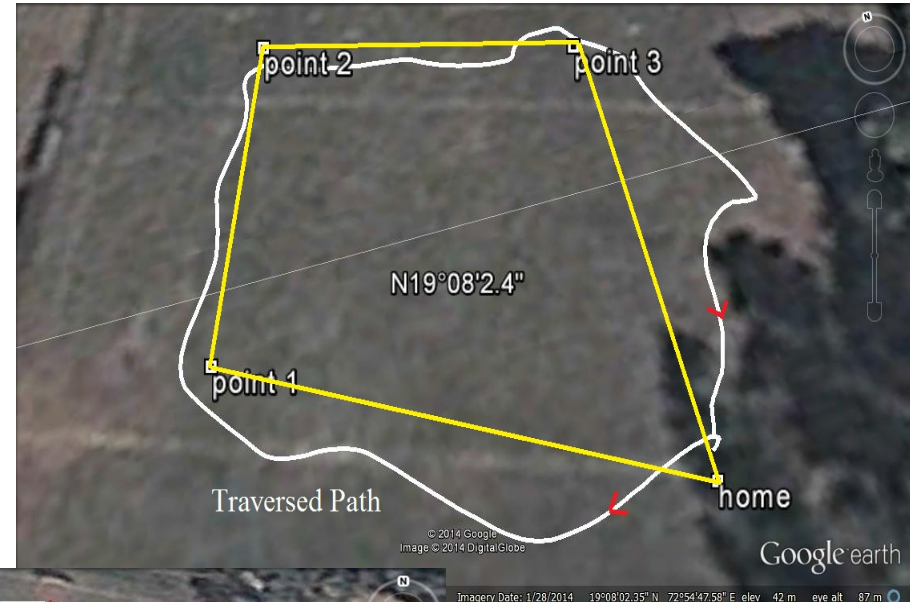
2012

2012

GPS waypoint-Leader Follower



2013



2013

2014

Video: Leader Follower - 1

Video: Leader Follower - 2

Still a long way to go before we can catch up with the leopard, duck or even cows

Thank You