2017-05-31

Introduction Contents

- What / how / why

What?

Robust Decentralized Control of Cooperative Multi-robot Systems

- *Simultaneous* control of *multiple* systems
- Decentralized: *no* central control authority
- Robust: uncertainty *is* present
- Cooperative: an agent-binding, resource-sharing task

So, what is it that I will be to you talking about? Let's break it down:

1. First of all we are interested in controlling multiple systems, that operate in the same space, at the same time...
2. ...these systems do not obey the commands of a central, shared control authority, but decide their actions on their own ...
3. ...under the presence of a disturbance of some sort
4. Lastly, these systems are somehow binded in a shared activity

Cooperation realizations (1/2)

Figure : Controller—Relay—End effector (Goals-setter — autonomous agents)

Sources: `http://bit.ly/2pTljEf`, `http://bit.ly/2qt1kOW`

For instance, DARPA is interested in controlling swarms of drones whose members may need to act as relays of information, due to the fact that communication ranges are not limitless.

Cooperation realizations (2/2)

Figure : Controller–Relay–End effector (Goals-setter – autonomous agents)

Prerequisites-tier:

- Connectivity maintenance
- Collision avoidance (agents plus obstacles)

In any case, we find that there are two fundamental prerequisites of cooperation in decentralized settings: an arbitrary agent needs to maintain connectivity with a set of agents, and avoid colliding with them, and any obstacle that it may come across.

In this context, there are two interesting areas of control:

Control in what sense? (1/2)

Reference tracking

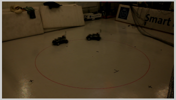


Figure : Vehicles tracking a desired trajectory in tandem.

The first is trajectory tracking, where either an agent all the whole multi-agent system needs to follow a given trajectory as close as possible, and stabilization. . .

Control in what sense? (2/2)

Stabilization

source: http://www.popularmechanics.com/military/weapons/a18362/tank-carry-beer/

where for instance each agent is assigned a desired configuration to which he must arrive at some point and stay, regardless of disturbances affecting his dynamic behaviour. I found this video of stabilization on the internet to be quite impressive.

And this is what the goal of this thesis is: to stabilize a multi-agent system subject to disturbances to predefined configurations for all agents. More or less we are interested in achieving this:

What we see here are three agents that have to somehow bypass two obstacles 1. without the blue agent losing connectivity with the other two, and 2. without all of them crashing into agents or obstacles

prior to stabilizing themselves in some chosen configuration.

How?

How?

an inter-constraint Receding Horizon approach

- ordinary Receding Horizon / Model Predictive Control strategy
- cooperating agents are *inter-constrained*

Now, how did we achieve this?
We used the control strategy of Receding Horizon (or Model Predictive) Control, where the connectivity and collision avoidance mandates are incorporated as constraints

Why MPC?

- direct incorporation of connectivity / collision constraints
- direct incorporation of input / state constraints
- theoretic guarantees of stability
- effective in aspects other methods are not

Why choose MPC over any other strategy?
MPC has the unique advantage that it can directly incorporate
1. the type of constraints we are interested in imposing 2. input / state constraints

a. we can procure theoretic guarantees of stability b. it can be effective in aspects other methods are not

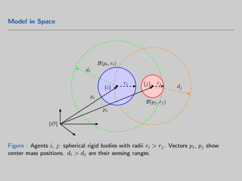
Prior approaches

- Navigation Functions
- Cost-coupled Decentralized MPC

For instance the same problem was approached using navigation functions, but this way proved to be not as robust as MPC.
The superiority of MPC has been exploited previously, and some approaches consider imposing the kinds of constraints we are interested in by coupling systems through the cost functions that MPC uses, instead of through their constraints.

The specifics: Contents

OK so next I'll talk about the problem and start formalizing it



We model agents in space as spherical rigid bodies, of radius r, which have a limited connectivity range of d, moving in space with p denoting their position through time.

Dynamics

Agents $i \in \mathcal{V} = \{1, 2, \dots, N\}$, with radii r_i and ranges d_i

Each agent $i \in \mathcal{V}$ is described by a continuous-time non-linear model

$$\dot{\mathbf{x}}_i = f_i(\mathbf{x}_i, \mathbf{u}_i) + \boldsymbol{\delta}_i$$

$\mathbf{x}_i \in X_i$, $\mathbf{u}_i \in U_i$

with *unknown* disturbance $\boldsymbol{\delta}_i \in \Delta_i : \|\boldsymbol{\delta}_i\|_\infty = \overline{\delta}_i$

In terms of dynamics, all agents are described by continuous-time, non-linear differential equations that include additive and bounded disturbance terms that are unknown.

Solution requirements (0/III)

Design $\mathbf{u}_i \in \mathcal{U}_i$ such that

- Collisions are avoided for all $i, j \in \mathcal{V},\ \ell \in \mathcal{L},\ t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > \underline{d}_{ij,a}$$
$$\|\mathbf{p}_i(t) - \mathbf{p}_\ell\| > \underline{d}_{i\ell,o}$$

  $\mathbf{p}_i$ component of state $\mathbf{x}_i$

- Neighbours $i \in \mathcal{N}_j$, $j \in \mathcal{N}_i$ maintain connectivity at all times $t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < \min(\overline{d}_i, \overline{d}_j)$$

- (Closed-loop) System is stable at desired state

$$\|\mathbf{x}_i(\infty) - \mathbf{x}_{i,\text{des}}\| = 0$$

Formally then, the problem lies in designing a control regime such that all input signals stay bounded, no collisions occur, all neighbours stay well-connected at all times, and that all agents are stabilized in steady-state

Solution requirements (I/III): constraint design

Solution requirements (I/III): constraint design

Design $\mathbf{u}_i \in \mathcal{U}_i$ such that

- Collisions are avoided for all $i, j \in \mathcal{V},\ \ell \in \mathcal{L},\ t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > \underline{d}_{ij,a}$$
$$\|\mathbf{p}_i(t) - \mathbf{p}_\ell\| > \underline{d}_{i\ell,o}$$

  $\mathbf{p}_i$ component of state $\mathbf{x}_i$

- Neighbours $i \in \mathcal{N}_j,\ j \in \mathcal{N}_i$ maintain connectivity at all times $t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < \min(\overline{d}_i, \overline{d}_j)$$

- (Closed-loop) System is stable at desired state

$$\|\mathbf{x}_i(\infty) - \mathbf{x}_{i,\text{des}}\| = 0$$

I will first focus on the second requirement and then go through the other two.

State constraint set

Encode all state requirements on agent i through time-varying set $\mathcal{Z}_i$.

$$\begin{aligned} \mathcal{Z}_i = \{\mathbf{x}_i(t) \in X_i : & \|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > \underline{d}_{ij,a}, & \forall j \in \mathcal{R}_i(t), \\ & \|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < \overline{d}_i, & \forall j \in \mathcal{N}_i, \\ & \|\mathbf{p}_i(t) - \mathbf{p}_\ell\| > \underline{d}_{i\ell,o}, & \forall \ell \in \mathcal{L}\} \end{aligned}$$

$\mathcal{R}_i(t)$: agents within range of agent i at time t
$\mathcal{N}_i$: assigned neighbours of agent i

The problem requires $\mathbf{x}_i(t) \in \mathcal{Z}_i$ for $t \geq 0$

The good thing with MPC as I mentioned is that we can encode all constraints in a set, say Z, which in our case is time-varying. The problem merely requires that the trajectories of each agent are kept within Z.

error-wise

The *error* of agent $i \in \mathcal{V}$ with respect to $\mathbf{x}_{i,\text{des}}$ is

$$\mathbf{e}_i(t) = \mathbf{x}_i(t) - \mathbf{x}_{i,\text{des}}$$

Described by a continuous-time non-linear model

$$\dot{\mathbf{e}}_i = g_i(\mathbf{e}_i, \mathbf{u}_i) + \boldsymbol{\delta}_i$$

$$\begin{aligned} &\mathbf{e}_i \in \mathcal{E}_i \equiv \mathcal{Z}_i \ominus \mathbf{x}_{i,\text{des}} \\ &\mathbf{u}_i \in U_i \\ &\boldsymbol{\delta}_i \in \Delta_i \end{aligned}$$

where $\boldsymbol{A} \ominus \mathbf{b} = \{\mathbf{a} - \mathbf{b}, \mathbf{a} \in \boldsymbol{A}\}$ is the Minkowski (Pontryagin) difference operation

If we look at the system in terms of error, we can get its dynamics and express the corresponding constraint set by using the minkowski set difference. We will be speaking in terms of error for the following slides.

Solution requirements (II/III): control input

Design $\mathbf{u}_i \in \mathcal{U}_i$ such that

- Collisions are avoided for all $i, j \in \mathcal{V},\ \ell \in \mathcal{L},\ t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > \underline{d}_{ij,a}$$
$$\|\mathbf{p}_i(t) - \mathbf{p}_\ell\| > \underline{d}_{i\ell,o}$$

  $\mathbf{p}_i$ component of state $\mathbf{x}_i$

- Neighbours $i \in \mathcal{N}_j$, $j \in \mathcal{N}_i$ maintain connectivity at all times $t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < \min(\overline{d}_i, \overline{d}_j)$$

- (Closed-loop) System is stable at desired state

$$\|\mathbf{x}_i(\infty) - \mathbf{x}_{i,\mathrm{des}}\| = 0$$

If we now focus on the control input requirements. . .

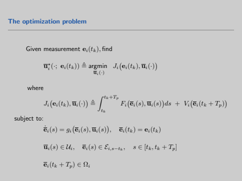
Solution requirements (II/III): control input

Solution requirements (II/III): control input

Designed control input $\mathbf{u}_i$ must

- result in states that satisfy the state constraints
- drive trajectory to / stabilize at desired state (how close?)
- exist and be feasible $\forall t \geq 0$ ($\mathbf{u}_i \in \mathcal{U}_i$)

...Such an input can be obtained by solving an optimization problem at regular sampling intervals.



it is the input that minimizes a standard MPC criterion over a period of T_p time units. This is the typical formulation of the MPC solution. However, there is a big difference here. The disturbance is unaccounted for in the dynamics of the solution because it is unknown.

Issue #1

The disturbance is unknown to the controller.

How can we guarantee that $\mathbf{e}_i \in \mathcal{E}_i$?

Somehow the disturbance needs to be accounted for within the optimization problem.

And so the first question we are asking is: how can we guarantee that the error does not escape the set in which we want it to be?

The optimization problem

Given measurement $\mathbf{e}_i(t_k)$, find

$$\overline{\mathbf{u}}_i^{\star}(\cdot;\ \mathbf{e}_i(t_k)) \triangleq \underset{\overline{\mathbf{u}}_i(\cdot)}{\operatorname{argmin}} \ J_i\big(\mathbf{e}_i(t_k), \overline{\mathbf{u}}_i(\cdot)\big)$$

where

$$J_i\big(\mathbf{e}_i(t_k), \overline{\mathbf{u}}_i(\cdot)\big) \triangleq \int_{t_k}^{t_k+T_p} F_i\big(\overline{\mathbf{e}}_i(s), \overline{\mathbf{u}}_i(s)\big)ds \ + \ V_i\big(\overline{\mathbf{e}}_i(t_k+T_p)\big)$$

subject to:

$$\dot{\overline{\mathbf{e}}}_i(s) = g_i\big(\overline{\mathbf{e}}_i(s), \overline{\mathbf{u}}_i(s)\big), \quad \overline{\mathbf{e}}_i(t_k) = \mathbf{e}_i(t_k)$$

$$\overline{\mathbf{u}}_i(s) \in \mathcal{U}_i, \quad \overline{\mathbf{e}}_i(s) \in \mathcal{E}_{i,s-t_k}, \quad s \in [t_k, t_k+T_p]$$

$$\overline{\mathbf{e}}_i(t_k+T_p) \in \Omega_i$$

What we do in this case is manipulate the set in which the error should lie in within the solution of the optimization problem. And that is what this (shows red) constraint is. At each step we progressively tighten the original constraint set during the solution of the optimization problem, so that, irrespective of the actual magnitude of the disturbance affecting the real system, its real trajectory always stays within the prescribed constraints. And we can do this because we know the supremum of the disturbance

Restricted constraint set (1/5)

$\mathcal{E}_{i,s-t_k} \equiv \mathcal{E}_i \ominus \mathcal{B}_{i,s-t_k}$

$\mathcal{B}_{i,s-t_k} \equiv \{\overline{\mathbf{e}}_i : \|\overline{\mathbf{e}}_i\| \leq \zeta(\overline{\delta}_i, s)\},\ \ \zeta(\nearrow, \nearrow)$

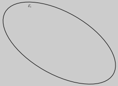


Figure : The set where all real trajectories must lie in.

What is the logic behind this manipulation? Let's depict the real constraint set of the error with this ellipse. When solving the optimization problem, we consider the evolution of the trajectory of the error for a number of time units. If disturbances are present, then, as time goes by, the true state becomes more and more unpredictable as more and more disturbance has affected it in the course of time, since the input given to the system does not know anything about the disturbances. The more time goes by and the more disturbance is present, the more unpredictable the system becomes, and the more likely it is that its states will escape the constraint set. If instead we restrict the predicted trajectory in a reverse manner at each step to be futher and further away from the boundary of the constraints, then we can guard the system from violating its constraints, provided that the horizon has been selected so that the last constrained set is not empty.

Restricted constraint set (2/5)

$\mathcal{E}_{i,s-t_k} \equiv \mathcal{E}_i \ominus \mathcal{B}_{i,s-t_k}$

$\mathcal{B}_{i,s-t_k} \equiv \{\bar{\mathbf{e}}_i : \|\bar{\mathbf{e}}_i\| \leq \zeta(\bar{\delta}_i, s)\}, \;\; \zeta(\nearrow, \nearrow)$

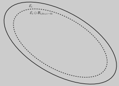


Figure : The set where all *predicted* trajectories are restricted in, when $t = t_k + h$, dashed.

In the first step, when $t = t_{k+1}$, the real constraint set is reduced to the states that will not violate the constraints if at $t = t_k$ the disturbance hits its maximum.

Restricted constraint set (3/5)

Restricted constraint set (3/5)

$\mathcal{E}_{i,s-t_k} \equiv \mathcal{E}_i \ominus \mathcal{B}_{i,s-t_k}$

$\mathcal{B}_{i,s-t_k} \equiv \{\bar{\mathbf{e}}_i : \|\bar{\mathbf{e}}_i\| \leq \zeta(\bar{\delta}_i, s)\}, \ \zeta(\nearrow, \nearrow)$

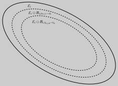


Figure : The set where all *predicted* trajectories are restricted in, when $t = t_k + 2h$, dashed.

In the second step the same happens for t_{k+2}

Restricted constraint set (4/5)

Restricted constraint set (4/5)

$\mathcal{E}_{i,s-t_k} \equiv \mathcal{E}_i \ominus \mathcal{B}_{i,s-t_k}$

$\mathcal{B}_{i,s-t_k} \equiv \{\bar{\mathbf{e}}_i : \|\bar{\mathbf{e}}_i\| \leq \zeta(\bar{\delta}_i, s)\}, \ \zeta(\nearrow, \nearrow)$

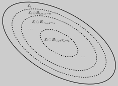
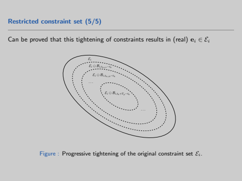


Figure : The set where all *predicted* trajectories are restricted in, when $t = t_k + T_p$, dashed.

until we reach the horizon



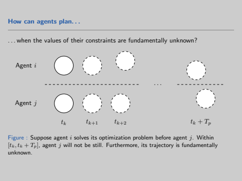
We can prove that if we restrict the predicted trajectory in this way within the optimization problem, then the real state will never violate the constraints.

Issue #2

In principle:

1. agents > 1
2. agents need to cooperate

This would be sufficient if we were only considering one agent. But we are considering multiple agents that need to cooperate with each other.



suppose for instance that agent i solves its optimization problem and he must sustain connectivity with agent j. within the optimization problem, agent j moves, and its position needs to be known to agent i at every step

They access the open-loop trajectories of agents in $\mathcal{R}_i$

When at time t_k agent i solves a finite horizon optimization problem, he has access to

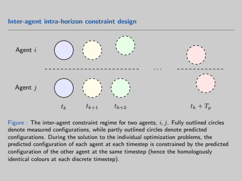
1. *measurements* of the states $\mathbf{x}_j(t_k)$

   of all agents $j \in \mathcal{R}_i(t_k)$ within its sensing range at time t_k

2. the *predicted states* $\overline{\mathbf{x}}_j(\tau)$, $\tau \in (t_k, t_k + T_p]$

   of all agents $j \in \mathcal{R}_i(t_k)$ within its sensing range

therefore, at time t_k when an agent solves its optimization problem, he needs to know where all agents within its sensing range are, and where they predict they will be for the next T_p seconds.



Now that agent i knows where agent j thinks he will move, we can constrain the states of agent i at each time step by the predicted state of agent j at the same time step. then, the constraint tightening takes care of the rest.

Steering to the desired state

Remember. . . the designed control input $\mathbf{u}_i$ must

- result in states that satisfy the state constraints
- drive trajectory to / stabilize at the desired state (how close?)
- exist and be feasible $\forall t \geq 0$ $(\mathbf{u}_i \in \mathcal{U}_i)$

That takes care of the state feasibility. The question now is: how does the input steer the state to the equilibrium?

Road-trip to equilibrium (1/7)

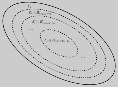


Figure : Progressive tightening of the original constraint set $\mathcal{E}_i$.

The constraint set tightening concludes at some set

Road-trip to equilibrium (2/7)

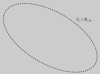


Figure : The furthest the predicted trajectories must be restricted to: $\mathcal{E}_{i,t_k+T_p-t_k}$

and suppose that this is it

Road-trip to equilibrium (3/7)

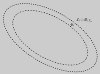


Figure : If linearized g_i is stabilizable, there exists $\overline{\mathbf{e}}_i \in \Phi_i$: $\mathbf{u}_i = h(\overline{\mathbf{e}}_i) \in \mathcal{U}_i$.

if this set is not empty, and if the linearized model of the system is stabilizable, then we can find a feasible input whose form is linear feedback with respect to the state, which

Road-trip to equilibrium (4/7)

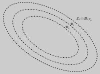


Figure : If this $\mathbf{u}_i$ is applied to a state in Ψ_i...

$$\Psi_i = \{\mathbf{e}_i \in \mathcal{E}_i : V_i(\mathbf{e}_i) = \mathbf{e}_i^\top \boldsymbol{P} \mathbf{e}_i \leq \varepsilon_{\Psi_i}, \quad \varepsilon_{\Psi_i} > 0\}$$

if we apply it to a state that lives in a set Ψ

Road-trip to equilibrium (5/7)

Road-trip to equilibrium (5/7)

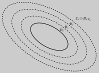


Figure : If this $\mathbf{u}_i$ is applied to a state in Ψ_i, the resulting state will end up in Ω_i.

$$\Omega_i = \{\mathbf{e}_i \in \mathcal{E}_i : V_i(\mathbf{e}_i) = \mathbf{e}_i^\top \boldsymbol{P}\mathbf{e}_i \leq \varepsilon_{\Omega_i}, \quad \varepsilon_{\Omega_i} \in (0, \varepsilon_{\Psi_i})\}$$

it will move the state to the terminal set

Road-trip to equilibrium (6/7)

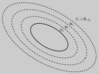


Figure : Once inside Ω_i, the trajectory is trapped there (Ω_i is invariant).

and since Ω is a subset of Ψ, it will always stay inside Ω.

Road-trip to equilibrium (7/7)

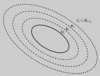


Figure : Once inside Ω_i, the trajectory is trapped there (Ω_i is invariant).

Provided that $\overline{\delta}_i \leq \xi(\varepsilon_{\Psi_i} - \varepsilon_{\Omega_i}, T_p)$, $\xi(\nearrow, \searrow)$, the trajectory never escapes Ω_i.

the size of the terminal set depends on the how strong the disturbance is and on the hierarchy of all of these sets. Provided that the supremum is bounded by a certain threshold that depends on the size of the terminal set and Ψ, and the length of the horizon.

Recursive feasibility

Remember... the designed control input $\mathbf{u}_i$ must

- result in states that satisfy the state constraints
- drive the trajectory / stabilize at the desired state (how close?)
- exist and be feasible $\forall t \geq 0$ ($\mathbf{u}_i \in \mathcal{U}_i$)

Can show that
if the solution is feasible at $t = 0$, then there are past controls and an auxiliary feedback controller $\forall t > 0$, both in $\mathcal{U}_i$.

as for the feasibility of the input, we can patch together past control sequences with the feedback controller and come up with a feasible control signal

Solution requirements (III/III): stabilization

Design $\mathbf{u}_i \in \mathcal{U}_i$ such that

- Collisions are avoided for all $i, j \in \mathcal{V},\ \ell \in \mathcal{L},\ t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > \underline{d}_{ij,a}$$
$$\|\mathbf{p}_i(t) - \mathbf{p}_\ell\| > \underline{d}_{i\ell,o}$$

  $\mathbf{p}_i$ component of state $\mathbf{x}_i$

- Neighbours $i \in \mathcal{N}_j$, $j \in \mathcal{N}_i$ maintain connectivity at all times $t \geq 0$

$$\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < \min(\overline{d}_i, \overline{d}_j)$$

- (Closed-loop) System is stable at desired state

$$\|\mathbf{x}_i(\infty) - \mathbf{x}_{i,\mathrm{des}}\| = 0$$

Now the last thing is the actual stabilization.

How is stabilization achieved?

$$\|\mathbf{x}_i(\infty) - \mathbf{x}_{i,\text{des}}\| \not\to 0$$

$$\mathbf{e}_i^\top \boldsymbol{P} \mathbf{e}_i \leq \varepsilon_{\Omega_i}$$

it turns out that we cannot drive the system to complete equilibrium, but we can only trap it in a neighbourhood of it. this neighbourhood is smaller than that which it would have been if the controller didn't manage to attenuate the disturbances.

How is stabilization achieved?

Can be shown that the optimal cost $J_i^\star = J_i(\mathbf{e}_i, \overline{\mathbf{u}}_i^\star)$

is an *Input-to-state* Lyapunov function in $\mathcal{E}_i$ for all $\boldsymbol{\delta}_i : \|\boldsymbol{\delta}_i\|_\infty \leq \overline{\delta}_i$

In essence:

- the *trajectory* is bounded in Ω_i: it does not escape Ω_i
- the *desired state* is not asymptotically stable unless the disturbance is decaying

We can show that the cost that results from the application of the suggested control input, the optimal cost, rather than monotonically deacreasing, is bounded. In this case we can prove that the optimal cost is an input-to-state lyapunov function, which makes the closed-loop system input-to-state stable within its constraints set

Example system: the unicycle

Unicycle dynamics

$$\dot{x}_i(t) = v_i(t)\cos\theta_i(t) + \delta_i(t)$$

$$\dot{y}_i(t) = v_i(t)\sin\theta_i(t) + \delta_i(t)$$

$$\dot{\theta}_i(t) = \omega_i(t) + \delta_i(t)$$

$$\delta_i(t) = \overline{\delta}_i \sin 2t$$

States: x_i, y_i, θ_i

Inputs: v_i, ω_i

Disturbance: $\overline{\delta}_i = 0.1$

$i \in \{1, 2, 3\}$

From this point on I will only show you how the control regime that I have described to you performs in simulations. We will use the standard unicycle model moving in $x - y$ and rotating around the z axis. Its states are $xy\theta$ and its inputs are the longitudinal velocity and the angular velocity

Simulation example I/II: 2D Trajectories

2D trajectories of agent 1 (blue, middle), agent 2 (yellow, below) and agent 3 (red, above). Black circles are obstacles. Agents 2 and 3 must keep connectivity with agent 1. Disturbances are absent. Trajectories with and without disturbances are more or less similar. Only steady-state changes.

Here we see three agents trying to get to their desired positions marked with x, that have to bypass these two obstacles. The blue one needs to be within certain bounds from the other two, and they all must not hit the obstacles or each other.

Simulation example II/II: Equilibrium close-up

The effect of unattenuated disturbances on the steady-state configurations. Left: no disturbance. Right: unattenuated additive bounded disturbances with $\overline{\delta}_i = 0.1$.

If we look at what happens around the desired equilibrium, we see on the left that we can totally stabilize the system when there are no disturbances. On the righthand side we see the effect of disturbances which have a supremum of 0.1, that have not been attenuated to the trajectories of the systems around the equilibrium.

Simulation example II/II: Equilibrium close-up

Steady-state configurations. Left: no disturbance. Middle: unattenuated additive bounded disturbances with $\overline{\delta}_i = 0.1$. Right: attenuated additive bounded disturbances with $\overline{\delta}_i = 0.1$ as a result of the proposed control regime.

if we design the control system the way i described you here today, then we can make the system get closer to its desired state.

Conclusion

It is possible to design a feasible control regime based on the principle of Receding Horizon Control such that, a multi-agent system constrained with connectivity and collision constraints on the states and input constraints, is stable under the influence of disturbances.

It is possible to design a feasible control regime based on the principle of Receding Horizon Control such that, a multi-agent system constrained with connectivity and collision constraints on the states and input constraints, is stable under the influence of disturbances.