System Level Synthesis: Introduction, Tutorial and Application to Satellite Constellations

1st Devan Tormey
1st Hari Krishna Hari Prasad
1st Kate Williams
1st Shikhar Varshney
School of Aeronautics and Astronautics
School of Mechanical Engineering
University of Washington
Seattle WA, USA

***Abstract*—This paper is a condensed tutorial to the System Level Synthesis (SLS): a new method to characterize and synthesize stabilizing controllers for a distributed systems, its software implementation, and application to a Satellite constellation model. The initial part of the paper is an elucidation of concepts from [1] and [2]; provides a brief introduction and discusses the benefits of SLS as a method of Distributed Control problem formulation, which is then followed by proofs of the fundamental theorems (discussed briefly in [1]) and it's comparison to previous, less generalized methods. Following this, synthesis of Distributed controllers by minimizing various norms over the transfer function of interest is discussed. Next, implementation of such a controller to a system where a node (can be thought of a sub-system with its own state-space representation) is hit with an impulse and the subsequent propagation of the disturbance is studied and illustrated through heat maps. Finally, a viable Satellite Constellation model is introduced, construction discussed, formulation of Distributed dynamic control law and finally its performance analysis is conducted.**

***Index Terms*—Control, SLS, SLA, SLP, Distributed Control, Quadratic Invariance, H_2 norm, L_1 norm, Youla Parameterization, Control Law, Locality Constraints, Sparsity Constraints, Actuation Delay, Communication speed, Satellite Constellation, Orbital Phases, Diagraphs**

INTRODUCTION

As attention switched from centralized to distributed optimal control, the parameterization approaches which were helpful for the centralized setting were no longer useful. Distributed systems are usually spread-out, physically distributed and interconnected. This in turn calls for a distributed set of sub-controllers with its own actuators and sensors. And these sub-controllers share information locally through a communication network (such as sensor measurements and applied control actions), which forms information sharing constraints on the system and makes distributed optimal controller synthesis problem extremely challenging.

With the introduction of quadratic invariance (QI), it was shown that for a large set of practically relevant LTI systems, such an internal structure could be integrated with the Youla parameterization maintaining the convexity of the problem. Informally, a system is QI if the sub-controllers exchange information with each other faster than their control actions propagate, which happens to be a necessary and sufficient condition for subspace constraints on the controller to be enforceable via convex constraints on the Youla Parameter. But, such a controller is very hard to synthesize and for strongly connected systems where sub-controllers have access to only a subset of the system wide-data at any given time, the problem isn't convex anymore.

With the advent of a powerful method known as System Level Synthesis (SLS) to parameterize, constrain, and synthesize controllers for the problems mentioned earlier, structural constraints are imposed on the whole system response which is characterized by maps from process to measurement disturbance to states and control actions; in contrast to the previous methods which looked at the feedback loops between sensors and actuators [2]. This structure carries over to the stabilizing controllers synthesized through this method which still admit a convex characterization [1].

MOTIVATION

The System Level Approach (SLA) presented in [1] is inspired from System level thinking. One Important distinction between SLA and Youla is that the prior - models structure of the whole system whereas the latter hid the internal structure of the controller and focused on the input-output behavior. The latter approach was natural for problems which had a logically centralized controller which has access to all measurements and actuators. However, in a distributed setting where each individual sub-controller is physically distanced, instantaneous access over all measurements is highly unrealistic. Hence, SLA includes the communication and information sharing structure as shown in figure below.

SALIENT FEATURES OF SLA

The following points refer to the important features of SLA as in [1].

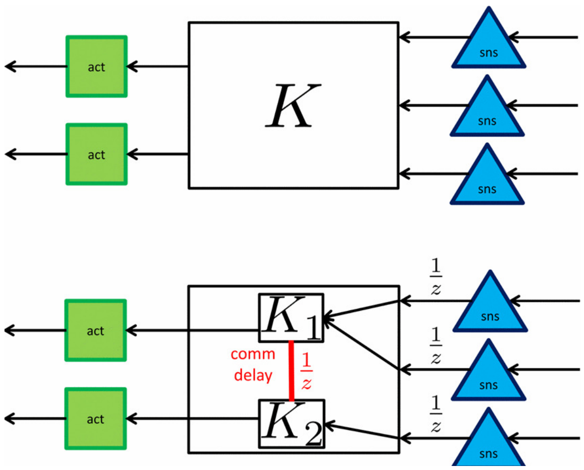


Fig. 1. Top: A classical centralized controller mapping sensor measurements to actuator commands. In this setting there are no communication constraints. Bottom: A distributed controller composed of two sub-controllers that can share information with a delay of 1 time-step.

- System Level Parameterizations (SLP) are used to parameterize all stabilizing controllers and closed-loop responses achieved by them. They constrain the system response to lie in arbitrary sets which are termed as System Level Constraints (SLC). If the SLCs admit a convex representation, then the underlying system responses also admit a convex representation.
- The SLC imposed on the system response in turn imposes a SLC on the internal structure of the controller.
- SLA provides the broadest known class of constrained controllers that admit a convex parameterization. The SLCs and SLPs together formulate a System Level Synthesis (SLS) problem which can be solved using convex programming.
- All previous methods to characterize distributed control problems can be framed as a SLS problem. Hence, it is a strict superset.

Problem Statement

A closed-loop plant can be represented by the Linear Fractional Transformation (LFT) block diagram shown below:

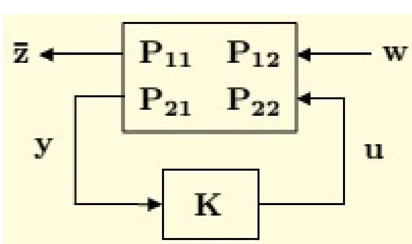


Fig. 2. Interconnection between the controller K and plant P.

This can be represented as a DT-LTI system which takes the following form

$$x[t+1] = Ax[t] + B_1 w[t] + B_2 u[t] \tag{1}$$

$$\bar{z}[t] = C_1 x[t] + D_{11} w[t] + D_{12} u[t] \tag{2}$$

$$y[t] = C_2 x[t] + D_{21} w[t] + D_{22} u[t] \tag{3}$$

where x, u, w, y, $\bar{z}$ are the state vector, control action, exogenous signal (it can be a disturbance or reference input coming in) and regulated output, respectively. In the partitioned form

$$\mathbf{P} = \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix} \tag{4}$$

where $\mathbf{P}_{ij} = C_i(zI - A)^{-1}B_j + D_{ij}$ is a the corresponding transfer matrix in the frequency or z-domain with P being the open-loop plant model. It is further assumed that the system is sparse (meaning there are a lot of zeroes in the state-space matrices since the total system is made up of n-subsystems).

In order to implement the communication and information sharing constraints, we will deal with dynamic input-output feedback control law of the form **u = Ky**, where **K** denotes the mentioned control law and has the state-space realization

$$\xi[t+1] = A_k \xi[t] + B_k y[t] \tag{5}$$

$$u[t] = C_k \xi[t] + D_k y[t] \tag{6}$$

with internal state ξ. The transfer matrix of **K** is given by $\mathbf{K} = C_k(zI - A)^{-1}B_k + D_k$. It is also assumed that the the interconnection shown in Figure 2 is well posed - the matrix $(I - D_{22}D_k)$ is invertible, along-with both plant and controller realizations are stabilizable and detectable. Now the final step is selecting a controller **K** which minimizes a chosen norm on the closed-loop map.

Figure 1 is an example of the information sharing constraints imposed on controller **K** by the communication network. There is a delay of one time-step for measurements to reach the sub-controller and another time-step delay for measurements from one sub-controller to reach the other. The mapping from sensor to actuation can be written as:

$$\begin{bmatrix} u_1(z) \\ u_2(z) \end{bmatrix} = \left(\frac{1}{z}\begin{bmatrix} * & * & 0 \\ 0 & 0 & * \end{bmatrix} \bigoplus \frac{1}{z^2}\begin{bmatrix} * & * & * \\ * & * & * \end{bmatrix} \bigoplus \ldots\right) \begin{bmatrix} y_1(z) \\ y_2(z) \\ y_3(z) \end{bmatrix} \tag{7}$$

where $*$ denotes an unspecified non-zero entry. Such an communication constraint can be encoded in the subspace constraint $\mathcal{C}$.

System Level Approach

A. *System Level Parameterization*

For a DT-LTI system described in (1),(2) and (3), the *system response* can be defined by maps **R**,**M**,**N**,**L** from disturbances to state vector and control actions:

$$\begin{bmatrix} x \\ u \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{N} \\ \mathbf{M} & \mathbf{L} \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} \tag{8}$$

We can notice a *key difference* between the Youla and SLA framework - state vector is an important part of the system response.

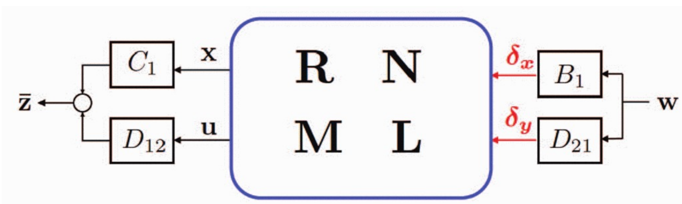


Fig. 3. The closed-loop response mapping w to $\bar{z}$. The SLP maps part of the closed-loop response that we design is the map from disturbances to state vector and control action.

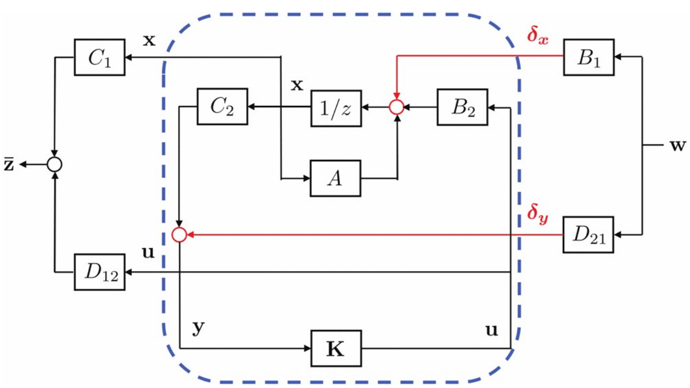


Fig. 4. Representation of the closed loop map in terms of the state space matrices from (1),(2),(3) and the controller **K**.

Theorem 1: For the output feedback problem with $D_{22} = 0$ in (3) the system response **R**,**M**,**N**,**L** transfer matrices from (8) are:

$$\begin{aligned} \mathbf{R} &= (zI - A - B_2\mathbf{K}C_2)^{-1} \\ \mathbf{M} &= \mathbf{K}C_2\mathbf{R} \\ \mathbf{N} &= \mathbf{R}B_2\mathbf{K} \\ \mathbf{L} &= \mathbf{K} + \mathbf{R}C_2\mathbf{R}B_2\mathbf{K}. \end{aligned} \tag{9}$$

and the following are true:
(a) The affine subspace described by:

$$\begin{bmatrix} zI - A & -B_2 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{N} \\ \mathbf{M} & \mathbf{L} \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix} \tag{10}$$

$$\begin{bmatrix} \mathbf{R} & \mathbf{N} \\ \mathbf{M} & \mathbf{L} \end{bmatrix} \begin{bmatrix} zI - A \\ -C_2 \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{11}$$

$$\mathbf{R}, \mathbf{M}, \mathbf{N} \ \epsilon \ \frac{1}{z} RH_\infty \tag{12}$$

parameterizes all system responses achievable by an internally stabilzing controller **K**.

(b) For an transfer matrices **R**,**M**,**N**,**L** satisfying (10),(11),(12) the controller $\mathbf{K} = \mathbf{L} - \mathbf{M}\mathbf{R}^{-1}\mathbf{N}$ is internally stabilizng and achieves the desired response [1].

Proof: (a) First, let us start off by deriving the expressions for the closed-loop responses **R**,**M**,**N**,**L** from equations (1),(2) and (3). Taking the z-transform, we have:

$$\begin{aligned} x(z) &= (zI - A)^{-1}B_1w(z) + (zI - A)^{-1}B_2u(z) \\ \bar{z}(z) &= C_1x(z) + D_{11}w(z) + D_{12}u(z) \\ y(z) &= C_2x(z) + D_{21}w(z) \end{aligned} \tag{13}$$

As we already know, these equations represent the open-loop response of the plant. Substituting $u = \mathbf{K}y$, closing the loop, we have:

$$x(z) = (zI-A)^{-1}B_1w(z)+(zI-A)^{-1}B_2\mathbf{K}(C_2x(z)+D_{21}w(z))$$

Simplifying the above equation (take $(zI - A)^{-1}$ to the other side and group common terms), we have the following expression for the state vector in z-domain:

$$x(z) = (zI-A-B_2\mathbf{K}C_2)^{-1}B_1w(z)+(zI-A-B_2\mathbf{K}C_2)^{-1}B_2\mathbf{K}D_{21}w(z) \tag{14}$$

The above expression denotes the closed-loop transfer matrices which map the state-disturbance 'δ_x' and measurement noise 'δ_y' to the state vector 'x'. Hence the closed loop maps **R** and **M** are given by

$$\mathbf{R} = (zI - A - B_2\mathbf{K}C_2)^{-1} \tag{15}$$

$$\mathbf{N} = (zI - A - B_2\mathbf{K}C_2)^{-1}B_2\mathbf{K} \tag{16}$$

Similarly substituting for x and y in $u = \mathbf{K}y$, we obtain the following expression for control action 'u'

$$u(z) = \mathbf{K}C_2R\delta_x + (\mathbf{K} + \mathbf{K}C_2RB_2\mathbf{K})\delta_y \tag{17}$$

Finally, we have the following expressions for the closed-loop transfer matrices which map δ_x and δ_y to u

$$\begin{aligned} M &= \mathbf{K}C_2R \\ N &= \mathbf{K} + \mathbf{K}C_2RB_2\mathbf{K} \end{aligned} \tag{18}$$

(b) The affine subspace constraints are easily proved by substituting the *system response* transfer matrices in (10), (11) and (12):

$$(1a.) \qquad (zI - A)R - B_2 N = I$$

Consider the LHS:

$$(zI - A)R - B_2\mathbf{K}C_2 R = (zI - A - B_2\mathbf{K}C_2)R = R^{-1}R = I$$

$$(1b.) \qquad (zI - A)N - B_2 L = 0$$

Consider the LHS:

$$(zI - A)RB_2\mathbf{K} - B_2\mathbf{K} - B_2\mathbf{K}C_2 RB_2\mathbf{K}$$

$$= (zI - A - B_2\mathbf{K}C_2)RB_2\mathbf{K} - B_2\mathbf{K}$$

$$= B_2 K - B_2 K = 0$$

$$(2a.) \qquad R(zI - A) - NC_2 = I$$

Consider the LHS:

$$R(zI - A) - RB_2\mathbf{K}C_2$$

$$R(zI - A - B_2\mathbf{K}C_2) = RR^{-1} = I$$

$$(2b.) \qquad M(zI - A) - LC_2 = I$$

Consider the LHS;

$$\mathbf{K}C_2 R(zI - A) - (\mathbf{K} + \mathbf{K}RB_2\mathbf{K})C_2$$

$$\mathbf{K}C_2 R(zI - A - B_2\mathbf{K}C_2) - \mathbf{K}C_2$$

$$\mathbf{K}C_2 - \mathbf{K}C_2 = 0.$$

Theorem 2: For the state feedback system, where (3) becomes $y[t] = x[t]$, the following are true:

(a) The affine subspace defined by

$$\begin{bmatrix} zI - A & -B_2 \end{bmatrix} \begin{bmatrix} \mathbf{R} \\ \mathbf{M} \end{bmatrix} = I \tag{19}$$

$$\mathbf{R}, \mathbf{M} \;\epsilon\; \frac{1}{z} RH_\infty \tag{20}$$

parameterizes all system response from δ_x to (x, u) as defined earlier, achievable by an internally stabilizing state feedback controller $\mathbf{K}$.

(b) For any transfer matrices $\mathbf{R}$,$\mathbf{M}$ satisfying the above conditions, the controller $\mathbf{K} = \mathbf{M}\mathbf{R}^{-1}$ is internally stabilizing and achieves the desired system response.

I. Implementation

In order to better understand any concept it's important to be able to pick apart and understand an example. In this section we will dive into a fairly simple but important example of how to implement SLS. For this example we are going to need a few assumptions,

- (1) The system is composed of n loosely-coupled subsystems.
- (2) The system is capable of returning full-state feedback.
- (3) There is no feed-through.

A. *Dynamics*

We will allow for disturbances to enter in to the state as well as in form of a reference. This can better be understood by expanding of Figure 3 into Figure 5. In this figure we have the references entering as δ_y, δ_X, and δ_u. For this discussion lets focus on δ_x. One thing that this diagram helps us dissect is the interaction between the plant and controller dynamics. One thing to keep in mind here is the fact that this diagram is in the frequency domain (z-transform). What we are doing inside the controller is mapping the controller estimate of the state disturbance, $\hat{\delta}_x$, to some control output u. This is achieved by feedback of some reference state, $\hat{x}$, via the R matrix. The frequency space equations are,

$$\hat{\delta}_x = x - \hat{x} \tag{21}$$
$$u = zM\hat{\delta}_x \tag{22}$$
$$\hat{x} = (zR - I)\hat{\delta}_x \tag{23}$$

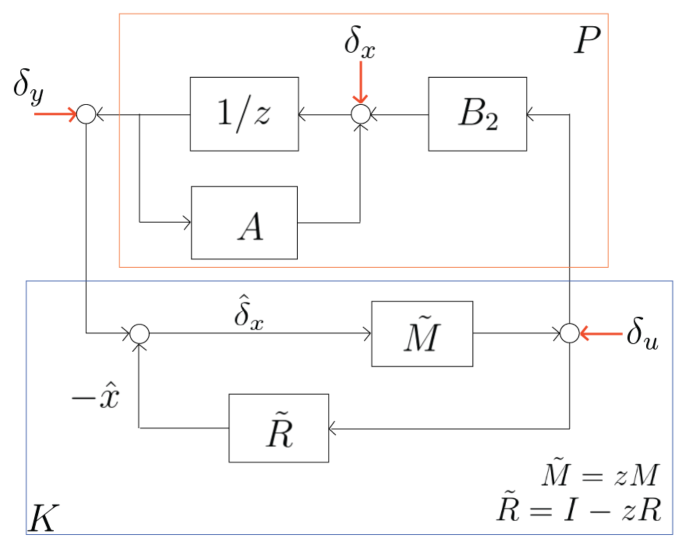


Fig. 5. The two dotted boxes separate the plant dynamics from the controller dynamics. $\tilde{M}$ and $\tilde{R}$ are introduced for simplicity

These equations show the power of the system level parameterization. By mapping the system as a whole from input disturbance to control output we end up with seeming straightforward definitions. However due to the fact that this is the frequency domain the discrete-time domain will equations appear,

$$\hat{\delta}_x[t] = x - \hat{x} \tag{24}$$

$$u[t] = \sum_{\tau=0}^{T-1} M[\tau+1]\hat{\delta}_x[t-\tau] \tag{25}$$

$$\hat{x}[t] = \sum_{\tau=0}^{T-2} R[\tau+2]\hat{\delta}_x[t-\tau] \tag{26}$$

So clearly from the fact that these equations were multiplication in the frequency domain they have become convolution in the time domain. From these equation and the diagram in Figure 5 we can reason that M is acting similarly to a proportional controller taking in our error measurement and generating a control output. The convolution is what allows it to select and generate controls individually for each subsystem. Secondly, we can see that R is propagating those error dynamics through to show us how our estimate $\hat{x}$ is changing as a result of this control. With this understand we can now address the real problem here, how do we generate these R and M matrices?

B. Controller Synthesis

Once the system has been parameterized and properly constrained, we are able to move on the last stage of the System Level Approach - System Level Synthesis. In the case that we have set forward with our constraints, system level synthesis is stated as the following convex minimization problem,

$$\begin{aligned} \underset{\{R,M\}}{\text{minimize}} \quad & g(R,M) \\ \text{subject to} \quad & \{R,M\} \,\epsilon\, S \end{aligned} \tag{27}$$

where $g(R,M)$ is some notion of norm placed on the system described by the interaction of plant and controller, and S is system level constraints placed on the system. More specifically if we restrict this norm to H_2, and wish to look at localizable, quadratically invariant systems under a finite time horizon. The problem statement becomes,

$$\begin{aligned} \underset{\{R,M\}}{\text{minimize}} \quad & ||C_1R + D_{12}M|| \\ \text{subject to} \quad & \{R,M\} \,\epsilon\, C \cap L \cap F_T \end{aligned} \tag{28}$$

In order to properly understand this example it is important we break down the constraints placed on our synthesis. This are whats known as system level constraints.

*1) **Quadratic Invariance**:* First off the biggest differentiation between the Youla Parameterization and SLS is the fact that these constraints are being placed on the system as a whole. They incorporate such things as the disturbance propagation speed, the actuator reaction speed, and the interconnectedness of the system. The first constraint C is this notion of quadratic invariance. Essentially this constraint is a comparison of the relative rate of the communication speed and the speed at which actuation passes through the system. In short, a system is quadratically invariant if subcontrollers are able to exchange information with each other faster than their control actions propagate through the system.

*2) **Locality Constraint**:* The second constraint L is known as the Locality constraint. This constraint comes from the comparison of the the actuation delay, communication speed, and disturbance propagation speed. Essentially, a system is localizable if, via the ratio of these three attributes, a disturbance can be dissipated such that it can be constrained to a small "area" or region of our state space model. In other word if the disturbance was a fire started in the forest we need to be able to communicate and react at such a speed and geometry as to contain that fire to only one area of the forest and not let the forest as a whole burn down.

*3) **FIR Constraint**:* The last constraint F_T is the Finite Impulse Horizon Constraint. This constraint correspond to the amount of time you allowing the actuators to act over. The more relaxed this constraint is the longer the controllers have to dissipate a disturbance. The tighter the constraint the less time. This, in a sense, becomes one of the key tuning parameters that go into controller synthesis as it determines the relative intensity that these controller will actuate with.

With these parameters understood, it is easier to see now that controllers will be shaped and tuned via the tightening and relaxation of these constraints.

II. RESULTS: EXAMPLE

In this example the system is composed of 50 states, with each state corresponding to a separate subsystem. As per the assumption in the previous section these subsystem are loosely coupled and we are provided full-state feedback.

A. H_2 Controller Synthesis

The following examples come from the code provided by Doyle as part of his CDC presentation.

*1) **Open Loop**:* In order to understand the coming examples it's important that we analyze the open loop response of the system. This can be seen in figure 6 .

This heat map allows us to understand the time varying response of the system as a whole. The y-axis of the graph represent the states of the system, or in this case the state of each subsystem. The x-axis is discrete time, this accounts for the blocky nature of the graph. Finally the response of the system is detailed via the heat map. The darker red the color the more intense the response of the system. For all of these graphs you may notice that the first dark red spot (left to right) is at the center of the states. That is because for every simulation we introduce a disturbance at the 25th state. This is in contrast to later in the report when the disturbance is introduced to the first state. Doyle's code is quite dynamic in that it allows us to choose various locations at which

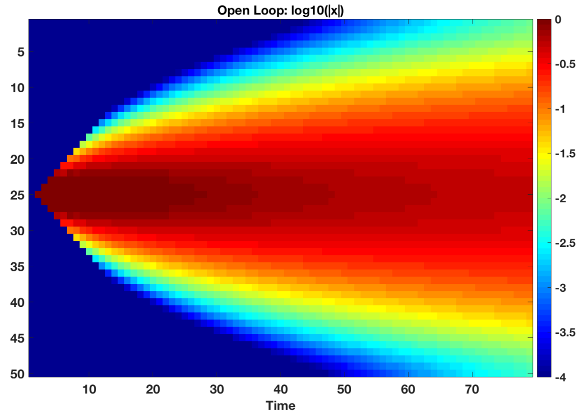


Fig. 6. This plot details the open loop response of system.

to introduce the disturbance as well as many other tuning parameters.

*2) **Actuation Density**:* In this section we will first look at the most basic of system level synthesis and then vary the actuation density parameter. For the most simple case we assume that we have no delay in communication and an actuator for every state x. The result of this can be seen in Figure 7

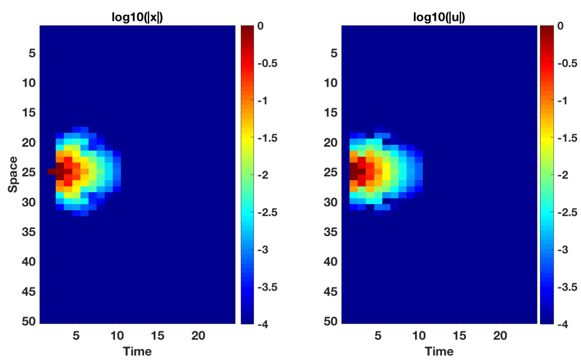


Fig. 7. Notice the very flat shape of the “nose” of the plot.

This plot, while still informative, may not be the most realistic. We can see here that while it takes one time step for the disturbance to propagate from the center state outward all of our actuators are able to act instantly. This reflects the lack of delay in the system. I feel it is important to place the delay constraints on the system before we can properly understand the effect of actuation as the two share some transient effects. In Figure 8 we introduce communication delays to the system.

These communication delays mean that the actuators cannot instantly begin to respond to the disturbance. The result is that we have this conical shape to the response as well as a longer impulse response horizon. Now that we have seen the effect of communication delay let’s look at what happens when we start to vary the actuation density. This can be seen in action

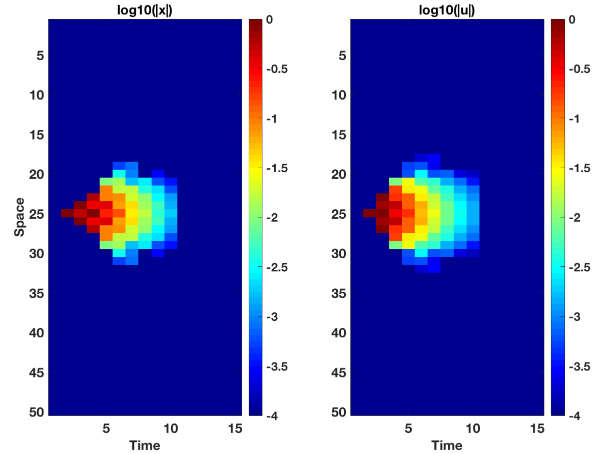


Fig. 8. The communication delay introduces a more conical shape in the step response.

in Figure 9

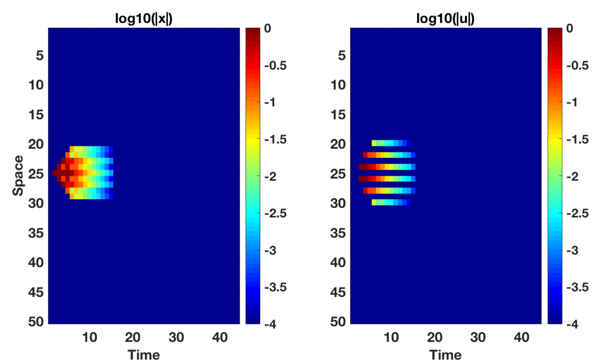


Fig. 9. The actuation density has been reduced as can be seen in the cone pattern on the right hand graph.

Clearly the reduction of the number of actuators will reduce the effectiveness of your controllers. A more interesting question become how many is enough? I picture it as a scenario in which you are tasked with catching falling sand. When you see the sand your hands are apart and open. You then have to bring your hands together and close all of the gaps in your fingers in order to catch all of the sand before it hits the ground. At some point your hands are too far apart, and then what happens to the sand? well some of it will fall through your fingers and fall to the floor. Eventually physics will cause the sand to stop moving but you will only be able to catch what doesn’t slip through your grasp. This effect can be seen in Figure 10

In Figure 10 we can see that while some of the disturbance has been properly mitigated some of the disturbance has leaked through. But due to fact that we have given our system an Impulse or control horizon long enough eventually the actuators can catch up. But what happens when this is violated?

*3) **Impulse Horizon**:* The most succinct explanation of the effect that impulse horizon has on actuation is intensity. It’s

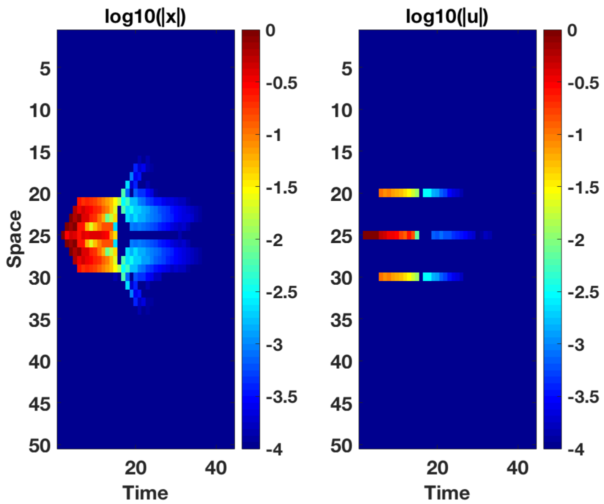


Fig. 10. This graph shows the effect of limited actuation density.

fairly straightforward to see that if you restrict the amount of time that a controller has to dissipate a disturbance the controller will have to actuate more intensely. The two are proportionately inversely related. This can be seen by first look at Figure 11

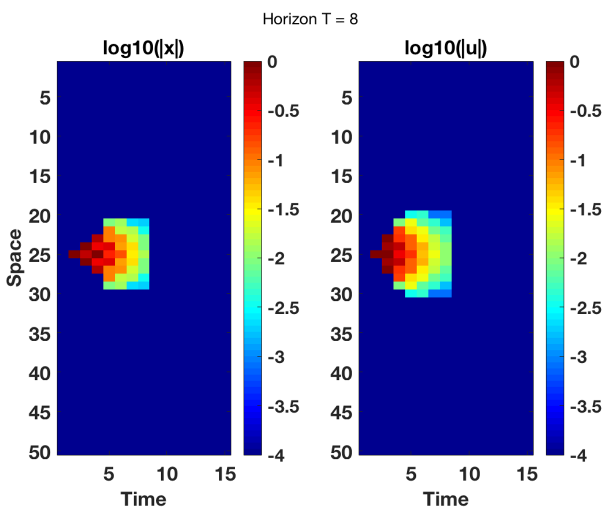


Fig. 11. This graph shows the result of impulse horizon of 8 discrete time steps.

In this example we have given the synthesis framework a reasonable (obviously relatively) horizon over which to control. Now observe what happens when you restrict this horizon by nearly a factor of 2. This can be seen in figure 12

While the time that it took to mitigate the disturbance has been shortened this was at the cost of a much increased actuation intensity. While this may seem simple it is important to understand that this constraint is one of the key parameters we can tune in order to alter the performance of our distributed control system.

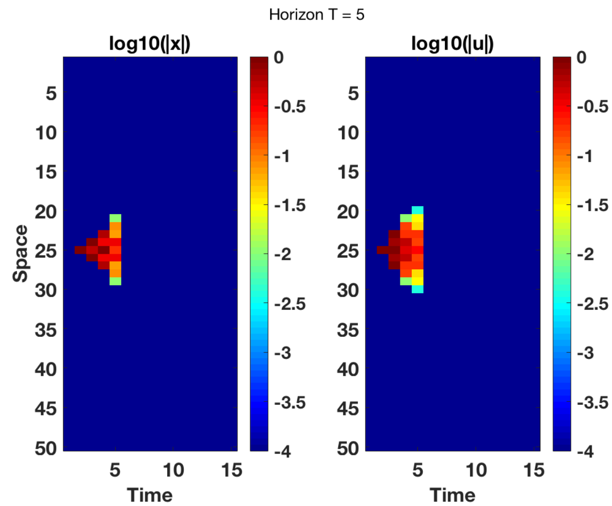


Fig. 12. This graph shows the result of impulse horizon of 5 discrete time steps.

*4) **Locality**:* Although the three prior parameters are important in the response of the system it is their combination and relative intensities that decides the ultimate outcome of the system response. This brings us to locality. Locality, informally, is the systems ability to mitigate a disturbance within a localized area. In essence a system is localized if for any given disturbance, the disturbance is not able to propagate throughout all states in a system. Think about this way, if there is a fire in a forest and we have a network of sensors and water dispensers able to put out a fire, it would be very important that we be able to sense, communicate, and act in such a time and fashion as to contain this fire before it spreads throughout the system. So while you may want to tune the Impulse Horizon such that the fire is stopped in time x, you may run up against the fact that the disturbance is propagating at some set speed and you can only communicate at speed y. In addition you must consider the fact that you may not have an actuator in every possible position. The summation and comparison of these things is what determines a systems localizability. The violation of this constraint is illustrated in Figure 13

In this figure we have given the system a maximum of 45 time steps to control the disturbance. As well as a fairly sparse actuator network. As we can see here the ability to mitigate the disturbance falls apart. And, even at time T_{max}, the disturbance is still propagating aggressively. The subject on how and when this constraint is violated could be (and is) a paper of its own. For the purpose of this project we need only understand that there is some region in which the Venn diagram of these three parameters overlap in which the system is localized.

*B. L_1 **Controller Synthesis***

The L1 norm is the sum of the magnitudes of the vectors in a space. It measures the distance between vectors by summing of absolute difference of the components of the vectors. For an asymptotically stable positive linear system denoting G as

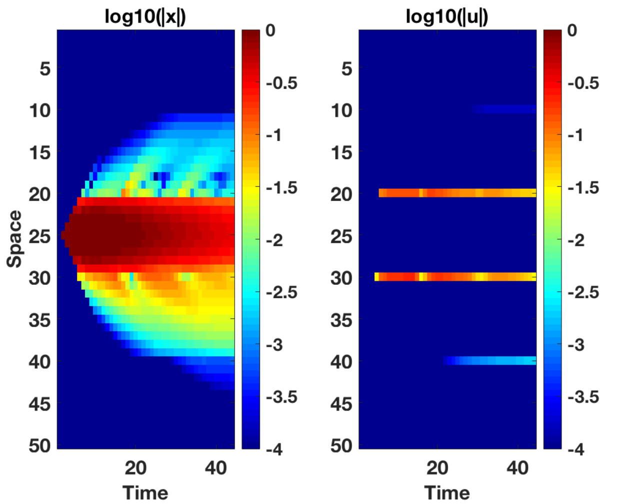


Fig. 13. This graph shows the result of violating the localizability constraint.

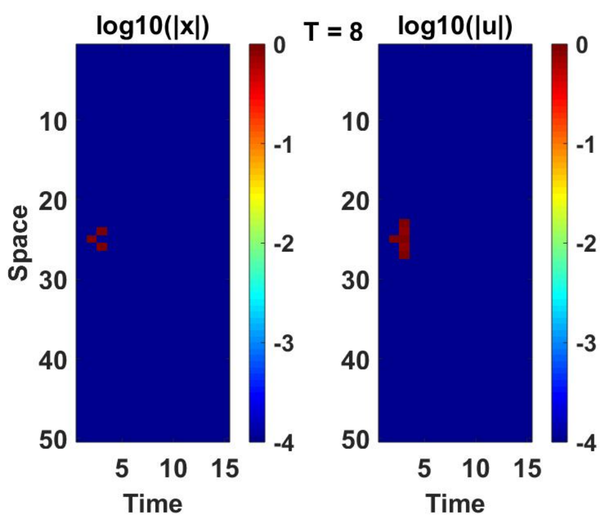
the system operator for input w to output y the L1 induced norm is given by

$$G_{L1->L1} = ||C(I-A)^{-1} - B + D|| \tag{29}$$

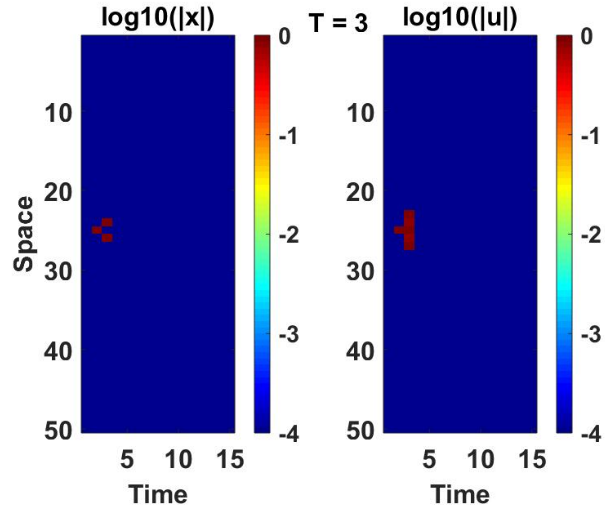
For SLS using the previously defined R and M matrices, equation 11 becomes

$$G_{L1->L1} = sum||C * R(t) + D * M(t)|| \tag{30}$$

Since our system is asymptotic and has positive eigen values we can use this norm for our sythensized system. We first completed an SLS analysis varying the finite impulse horizon but discovered that there was no change in actuation density. As shown below for different horizons the time for the disturbance to mitigate and at what actuation density this occurred did not vary.

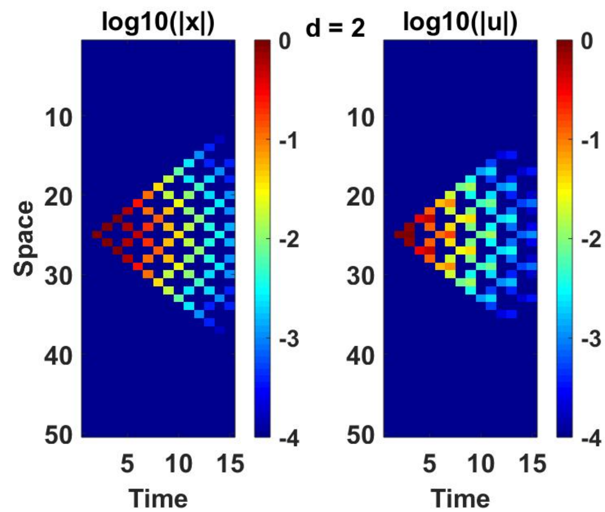


Next we varied actuation from 1 to 0.2 and found that as it decreased the disturbance began to leak, similarly with the H2



norm, and finally resulted in a ”star ship looking” propagation at a very limited actuation of 0.2.

Finally we also varied the dehops of the system but found it only stable for a dehops of 2 with full actuation. With this combination the system cancels out the disturbance in about 15 seconds. It should be noted that though the system did succeed it did so after only a long time. In the real world a disturbance that would take so long to eliminate would result in significant damage.



III. Satellite Constellation Model

After understanding how the theory and the implementation of SLS framework. We decided to use it on a problem of interest. We chose the Satellite constellation model. A satellite constellation is a group of artificial satellites working in a specific formation. These satellites are usually employed for a specific application and as such have similar orbits, eccentricity and inclination. Any kind of disturbance affects every satellite in a similar manner i.e. the same disturbance when affecting two satellites will cause similar change. A satellite constellation is shown in the Figure 14.

Fig. 14. A satellite constellation. Photo: ESA

A. Why this model?

This model is a practical model for SLS because:

- A distributed control system is preferred as all the satellites can be considered as a similar node.
- The QI constraint is satisfied as the communication is done at a light speed.
- The constellation model turns out to satisfy sparsity constraint as each satellite is depended upon only a few satellites for maintaining it's formation in the constellation.
- There are many applications for satellite models. These motivate the importance of a robust controller.
  - Global Positioning System or GPS
  - Weather monitoring
  - Surveillance
  - Telecommunication
    - SpaceEx Starlink program with about 12000 Satellites in a single constellation.
- Satellites are cool!

B. ***Modeling the system***

Ulybyshev in his paper on Long-Term formation keeping of Satellite Constellation provided a way to model Satellite constellations and also controlled the system with an LQR controller. [3]. We used his model for our SLS synthesis.

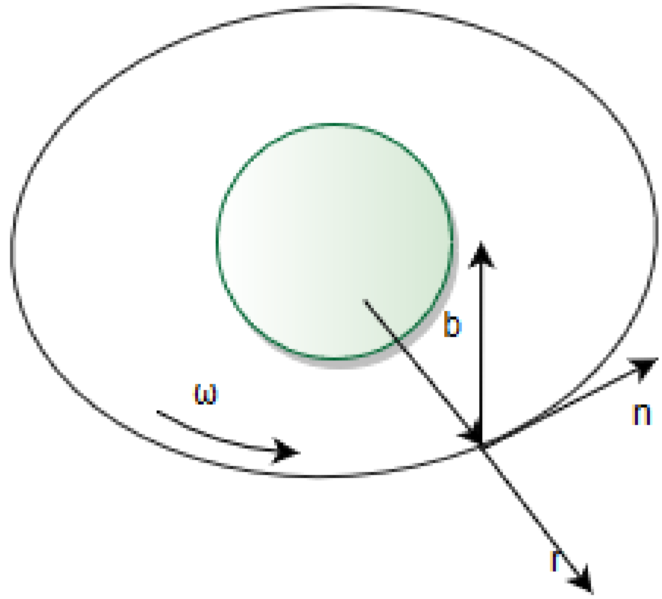


Fig. 15. Coordinate system

1) Coordinate System: The coordinate system we are using is shown in the Figure 15. n and r are the coordinates in the plane of the orbit. b is the out of orbit coordinate.

2) ***Modelling****:* The paper uses the Clohessy-Wiltshire equations to describe the motion of satellite.

$$\ddot{n} = -2\omega\dot{r} \tag{31}$$

$$\ddot{r} = 3\omega^2 r + 2\omega\dot{n} \tag{32}$$

$$\ddot{b} = -\omega^2 b \tag{33}$$

They arrive at N uncoupled discrete-time equations using the above equations which relate the N satellites to each other.:

$$\left\{ \begin{array}{c} \Delta n_i(k+1) \\ \Delta T_i(k+1) \end{array} \right\} = \left[\begin{array}{cc} 1 & -1 \\ 0 & 1 \end{array} \right] \left\{ \begin{array}{c} \Delta n_i(k) \\ \Delta T_i(k) \end{array} \right\} + \left[\begin{array}{c} 0 \\ 1 \end{array} \right] \Delta V_i(k)$$

i=1, . . . , N (34)

where N is the number of satellites, ΔT is the displacement of the satellite orbital period relative to reference orbit period, Δn are the relative displacements.

3) ***Diagraphs****:* Then, they describe a diagraph. This technique is able to map a complicated 3-D satellite constellation to a 2-D mapping. This uses prescribed inter satellite spacing of satellite constellations. This is used to establish a constant relationship constraint between the displacements and the time periods of different satellites according to the formation they are. A link from one satellite to another is called a 'direct edge' or a 'controlled arc'. Then one of the satellites is taken as the root of the diagraph. Each diagraph with N vertices has precisely $N-1$ edges. Each diagraph edge between two satellites is the prescribed spacing whereas for two satellites in different orbital planes, it is the inter-plane phase. The diagraph mapping shown in Figure 16.

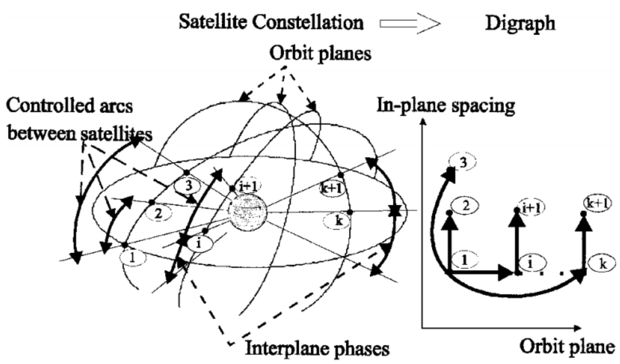


Fig. 16. Satellite Constellation to a diagraph mapping. [3]

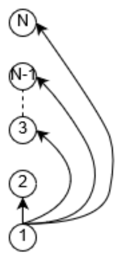


Fig. 17. The MLT diagraph

Diagraph.png

Fig. 18. The Chain diagraph or CG

Then they give two kinds of diagraphs: minimum length tree or MLT (Figure 17) and the chain graph or CG (Figure 18. In MLT all the nodes in diagraph directly relate to the root. Where as in the CG, each diagraph relates to the previous node. All diagraphs can be represented as a combination of the two. For the two diagraphs they provide corresponding incidence matrices describing the diagraphs. Coefficients of the incidence matrices are -1,0,1. The coefficient is zero when the node is not related to the other node, coefficient is +1 when the node is the start point of the branch, -1 when the node is the end point of the branch. The transposed incidence matrices A_t are given below with all of the unspecified elements equal to zero. The MLT and CG transposed indices matrices are given below respectively.

$$A_t = \begin{bmatrix} 1 & -1 & & & & \\ 1 & & -1 & & & \\ 1 & & & -1 & & \\ . & . & . & . & . & . \\ 1 & & & & -1 & \\ 1 & & & & & -1 \end{bmatrix} \tag{35}$$

$$A_t = \begin{bmatrix} 1 & -1 & & & & \\ & 1 & -1 & & & \\ & & 1 & -1 & & \\ . & . & . & . & . & . \\ & & & 1 & -1 & \\ 1 & & & & 1 & -1 \end{bmatrix} \tag{36}$$

C. State Space equation

From equations 35 and 36, we can see the matrices are sparse and therefore it satisfies the sparsity constraint of SLS framework. From the equation 34 the paper arrives at a discrete time state space model with state vector $\boldsymbol{x}^T = \left(\delta\boldsymbol{n}^T, \Delta\boldsymbol{T}^T\right) = (\delta n_1 \,, \delta n_2, \ldots, \delta n_{N-1}, \Delta T_1, \Delta T_2, \ldots, \Delta T_N)$, where $\delta n_j = \Delta n_i - \Delta n_{i+1}$.

And then the state matrix A is written as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I}_{N-1} & \boldsymbol{A}_t \\ \boldsymbol{0}_{N\times(N-1)} & \boldsymbol{I}_N \end{bmatrix} \tag{37}$$

where $\boldsymbol{I}_{N-1}$ and $\boldsymbol{I}$ are identity matrices and $\boldsymbol{0}_{N\times(N-1)}$ is a zero matrix. Paper proposes the long time behavior of the satellite to be described by the following multi-variable discrete time equation:

$$x(k+1) = Ax(k) + Bv(k) \tag{38}$$

where

$$B = \begin{bmatrix} 0_{(N-1)\times N} \\ I_N \end{bmatrix} \tag{39}$$

and $v^t = (\Delta V_1, \Delta V_2, \ldots, \Delta V_N)$ is the control vector. This is completely state controllable as the rank of controllability matrix is $2N-1$.

D. SLS Basic Implementation

Now, after understanding the model we are using. We needed to implement the SLS algorithm on our model. We implemented the SLS basic algorithm. Even with light speed communication, the SLS basic algorithm fails to accommodate actuator delays which maybe in the order of hours. Hence, this implementation is unrealistic but we use it due to it's simplicity and to verify that our code runs with our chosen model. Also, we minimized the H2 norm in this case. To run this we require the A and B matrices from the model, which we get from the equation a matrix and bmatrix. For the A_t in A we decided to use the chain graph matrix as that is closer to the examples given by Doyle, with each satellite related to it's neighbors. We plug these values in our code, disturb the first state and get Figure 19.

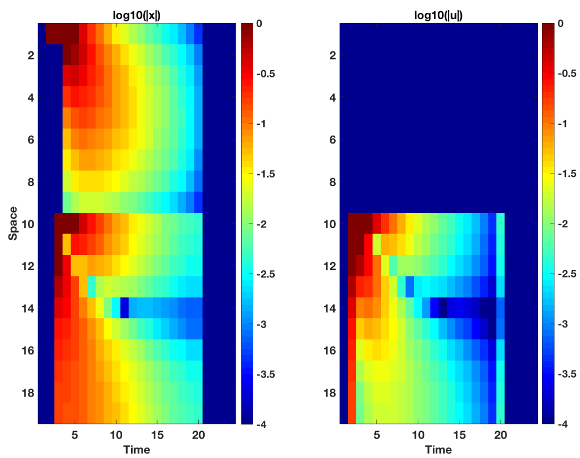


Fig. 19. The SLS Basic plot from our Model

In the state side we see the first state was disturbed first and the 11th state disturbed with it as expected the displacement and the orbital period were disturbed together. The plot in the bottom half is different as the number of orbital phases is more than the number of relative displacements as we choose one satellite to be the reference. The plot shows that the disturbance loops around. Now, for the control input heat map is only there for the bottom half of the plot. *This is because control input only directly affects the orbital phases.* We see the disturbance quickly dies down in the middle for the orbital phases. This seems to be due to the looping as discussed before, the input works according to the constraints in the model and corrective measures around to the middle states has a greater impact on the disturbance decay around node 14.

Conclusion and Future Work

So far, we have seen the benefits of SLS in framing Distributed Control problems. We started off with introduction, implementation of H_2 and L_1 stabilizing controllers and the eventual application of *SLS basic* algorithm to a satellite constellation model. The key drawback of this algorithm is that it assumes infinite communication speed, access of system-wide measurements and instantaneous actuation for corrective trajectories, which is basically ***Centralized control***. This isn't very realistic with respect to real systems, since you only have access over a *subset of the system-wide measurements* with *less than infinite communication speed* (in our case, the communication takes place at the speed of light, hence it is still *very fast compared to actuation delay*) and there is *delay in the actuation* (for our system, more in the order of a few hours for the corrective maneuver). We are currently working on *localizing* this system, introducing finite communication speeds and non-zero actuation delays. Future work can involve comparing the the L_1, H_2 and H_∞ distributed control laws by setting up some performance metrics such as actuation energy consumed (sum of the actuation values across nodes in the system), Disturbance decay horizon and minimum measurement subset required. Following which, we could draw patterns from these distributed controllers and see where each controller works best - ***classification of distributed controllers through SLS***.

References

[1] J. C. Doyle, N. Matni, Y.-S. Wang, J. Anderson, and S. Low, "System level synthesis: A tutorial," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. Melbourne, VIC: IEEE, Dec. 2017, pp. 2856–2867. [Online]. Available: http://ieeexplore.ieee.org/document/8264074/

[2] Y.-S. Wang, N. Matni, and J. C. Doyle, "A System Level Approach to Controller Synthesis," *IEEE Transactions on Automatic Control*, pp. 1–1, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8600332/

[3] Y. Ulybyshev, "Long-Term Formation Keeping of Satellite Constellation Using Linear-Quadratic Controller," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 1, pp. 109–115, Jan. 1998. [Online]. Available: http://arc.aiaa.org/doi/10.2514/2.4204