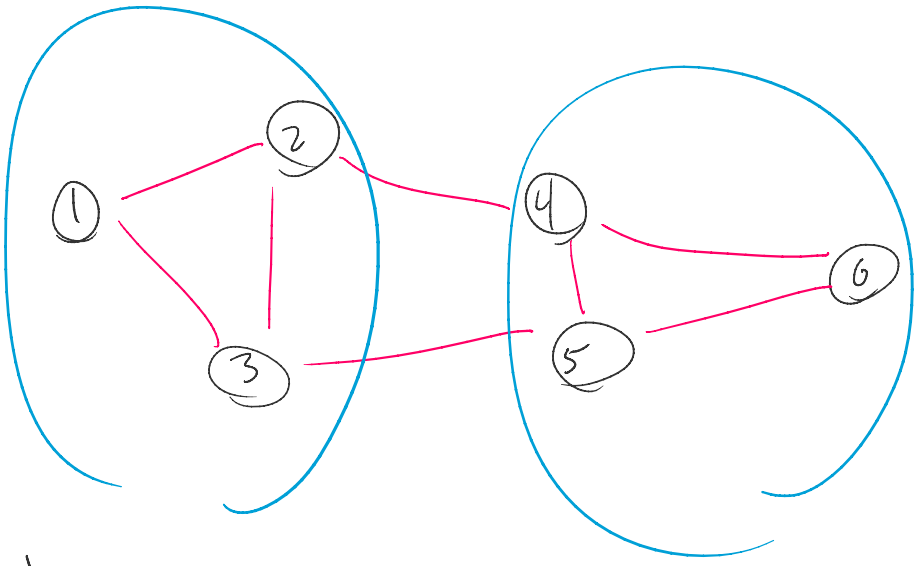
Coord control

Tuesday, August 30, 2022 7:43 PM



i, j — PAIR OF ROBOTS IN QUESTION

$p = \begin{cases} 2 & \text{if PLANAR} \\ 3 & \text{if AERIAL} \end{cases}$ (DOF)

$\mathbb{R}^p$ 2D GROUND
3D AERIAL

N — # OF ROBOTS

$\vec{x}$ — POSITIONS OF ROBOTS IN $\mathbb{R}^p$

$\mathcal{E}$ — ERROR IN POSITIONS

G — GRAPH STRUCTURE OF ROBOT TEAM

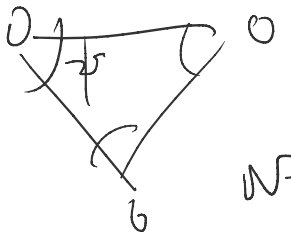
→ V — VERTEX SET

→ E — EDGE SET

L — GRAPH LAPLACIAN

DEGREE MATRIX — ADJECENTCY MATRIX

ϕ — OFFSET ANGLE



$N=3$

If $\phi = \frac{\pi}{N}$ ⟶ CYCLIC PURSUIT

If $\phi < \frac{\pi}{N}$ → SPIRAL INWARD

If $\phi > \frac{\pi}{N}$ → SPIRAL OUTWARD

CONSENSUS EQUATION $\dot{x} = -Lx$

WHAT IS x INITIALIZED AS?

CONSENSUS IS LOCAL AND SCALABLE

~~I DONT QUITE UNDERSTAND HOW THIS WORKS.~~ DRIVES CLOSER ALONG EDGES.

$$\dot{x}_i = R(-\phi)(x_{i+1} - x_i)$$

$$\dot{x}_N = R(-\phi)(x_1 - x_N)$$

← INITIALIZE IN DIRECTED CYCLE TOPOLOGY

FLOCKING $\dot{\phi} = -L\phi$

$$\frac{d\phi}{dt} = -\begin{bmatrix} \ddots & -1 & & \\ -1 & & -1 & 0 \\ & -1 & \ddots & -1 \\ & 0 & -1 & \end{bmatrix} \phi$$

The single integrator dynamics in conjunction with (2) can be represented as the Laplacian dynamics of the form

$$\dot{x}(t) = -\mathbb{L}(\mathcal{G})x(t), \tag{3}$$

where $x(t) = [x(t)_1^T,\ x(t)_2^T, \ldots, x(t)_n^T]^T$ denotes the aggregated state vector of the multi-agent system, $\mathbb{L}(\mathcal{G}) := \mathcal{L}(\mathcal{G}) \otimes I_d$, with I_d denoting the d-dimensional identity matrix, and $\otimes$ is the matrix Kronecker product [16]. In fact, if the dynamics of the agent's state is decoupled along each dimension, the behavior of the multi-agent system can be investigated one dimension at a time. Although our results can directly be extended to the case of (3), in what follows we will focus on the system

$$\dot{x}(t) = -\mathcal{L}(\mathcal{G})x(t), \tag{4}$$

capturing the multi-agent dynamics with individual agent states evolving in $\mathbb{R}$.

34

CHAIN RULE

MIDPOINT

$$\mathcal{E}(x) = \frac{1}{2}\sum_{i=1}^{N}\sum_{(i,j)\in E}\|x_i - x_j\|^2$$

DESIGNED SO DERIVATIVE IS SIMPLE

$$\frac{\partial \mathcal{E}(x)}{\partial x_i} = \sum_{(i,j)\in E}(x_i - x_j), \quad i = 1, \ldots, N.$$

$$\dot{x}_i = -\sum_{(i,j)\in E}(x_i - x_j).$$

$$\mathcal{E}(x) = \sum_{i=1}^{N}\sum_{(i,j)\in E}\mathcal{E}_{ij}(\|x_i - x_j\|).$$

$$\frac{\partial \mathcal{E}_{ij}(\|x_i - x_j\|)}{\partial x_i} = \frac{\partial \mathcal{E}_{ij}(\|x_i - x_j\|)}{\partial \|x_i - x_j\|}\frac{(x_i - x_j)}{\|x_i - x_j\|} = w_{ij}(\|x_i - x_j\|)(x_i - x_j),$$

$$\dot{x}_i = -\frac{\partial \mathcal{E}}{\partial x_i} = -\sum_{(i,j)\in E} w_{ij}(\|x_i - x_j\|)(x_i - x_j).$$

$$\dot{x} = \frac{dx}{dt} = \frac{-\partial \mathcal{E}}{\partial x} = \sum_{i=1}^{N} -\frac{\partial \mathcal{E}}{\partial x_i}$$

OVER TIME, ERROR IN POSITION IS DECREASING

FIND $\dfrac{\partial}{\partial x_i}\,\mathcal{E}_{ij}(\|x_i - x_j\|)$, $u = \|x_i - x_j\|$

$$= \frac{\partial}{\partial u}\,\mathcal{E}_{ij}(\|x_i - x_j\|) \cdot \frac{\partial}{\partial x_i}\,\|x_i - x_j\|$$

$$= \frac{\partial\, \mathcal{E}_{ij}(\|x_i - x_j\|)}{2\,\|x_i - x_j\|} \cdot \frac{2}{\partial x_i}\sqrt{(x_{i_1} - x_{j_1})^2 + \cdots + (\cdots)^2}$$

$$= \cdots \quad \frac{1}{2}\left[(x_{i_1} - x_{j_1})^2 + \cdots + (\cdots)^2\right]^{-1/2} 2(\cdots)$$

$$= \cdots \quad \frac{(x_i - x_j)}{\|x_i - x_j\|}$$

IF $\mathcal{E}_{ij}(u) = u$

$\rightarrow \dfrac{\partial}{\partial u}\mathcal{E}_{ij}(u) = \dfrac{\partial}{\partial u}u$

$= 1$ (?)

$\mathcal{E}(\|x_i - x_j\|)$ → ERROR BY WHICH ROBOTS ARE CONVERGING

$\dfrac{\partial \mathcal{E}}{\partial x_i}$ → RATE ERROR IN DISTANCE IS CHANGING BY FROM THE PERSPECTIVE OF i ROBOTS

ERROR CHAIN ALL ROBOTS

i.e., the partial derivative is a scalar function of the inter-robot distance times the relative displacement. As such, the gradient descent rule is given by a weighted consensus protocol,

$$\dot{x}_i = -\frac{\partial \mathcal{E}}{\partial x_i} = -\sum_{(j,i)\in E} w_{ij}(\|x_i - x_j\|)(x_i - x_j).$$

$$\frac{d\mathcal{E}}{dt} = \frac{\partial \mathcal{E}}{\partial x}\dot{x} = \sum_{i=1}^{N} \frac{\partial \mathcal{E}}{\partial x_i}\dot{x}_i = -\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2.$$

POSITION / TIME

REDUCING ERROR IN POSITION

SUM EDGES

WEIGHT

DISTANCE

DISPLACEMENT

COLOR THM

CHAIN RULE

ALL ROBOTS

PROPERTIES OF DOT PRODUCT

$$\sum \frac{\partial \mathcal{E}}{\partial x_i} \cdot -\frac{\partial \mathcal{E}}{\partial x_i}$$

$$-\sum \vec{u}_i \cdot \vec{u}_i \,, \quad u = \frac{\partial \mathcal{E}}{\partial x}$$

$$-\sum_{i=1}^{N} \|\vec{u}_i\|^2$$

$$\frac{d\mathcal{E}}{dt} = -\|\vec{u}\|^2$$

$$\frac{d\mathcal{E}}{dt} = -\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2$$

★ WHAT IS w ?

The Chain Rule tells us that

$$\frac{\partial \mathcal{E}_{ij}(\|x_i - x_j\|)}{\partial x_i} = \frac{\partial \mathcal{E}_{ij}(\|x_i - x_j\|)}{\partial \|x_i - x_j\|}\frac{(x_i - x_j)}{\|x_i - x_j\|} = w_{ij}(\|x_i - x_j\|)(x_i - x_j),$$

So, $w = \frac{\partial \mathcal{E}_{ij}}{\partial u}\frac{1}{u}$, $u = \|x_i - x_j\|$ DISTANCE

$$\frac{\partial \mathcal{E}_{ij}(u)}{\partial u}\frac{u^2}{u} = w_{ij}\, u \cdot u^2$$

$$= \left(\frac{\partial \mathcal{E}_{ij}}{\partial u}\frac{1}{u}\right) u \cdot u^2$$

if $\mathcal{E}_{ij}(\|x_i - x_j\|) = f$

$$\longrightarrow w_{ij} = \frac{\partial f}{\partial x_i}$$

f CAN BE DESIGNED

$$\longrightarrow \text{OR IS } w_{ij} = \frac{1}{u} ?$$

$$\mathcal{E}_{ij}(\|x_i - x_j\|) = \frac{1}{2}(\|x_i - x_j\| - \delta)^2 \Rightarrow w_{ij} = \frac{\|x_i - x_j\| - \delta}{\|x_i - x_j\|}.$$

DRIVES TO MIDPOINT WITH MINIMUM DISTANCE

MIDPOINT OF

"1

MIDPOINT OF

SQUARE OF CUR DIST — MIN DIST

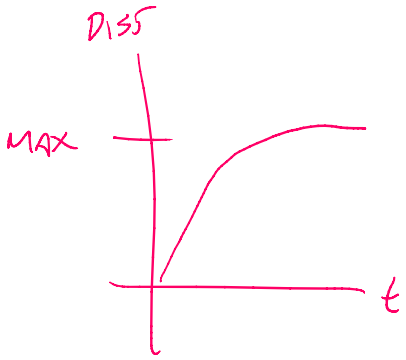
↑ KEEP POSITIVE

$$\mathcal{E}_{ij}(\|x_i - x_j\|) = \frac{\|x_i - x_j\|^2}{\Delta - \|x_i - x_j\|} \Rightarrow w_{ij} = \frac{2\Delta - \|x_i - x_j\|}{(\Delta - \|x_i - x_j\|)^2}.$$

PENALIZES SEPARATION

↑

SQUARE OF DISTANCES / MAX − CURRENT → PERCENTAGE OF SPACE TILL MAX



A combined formation control and connectivity maintenance protocol could thus become

$$\mathcal{E}_{ij}(\|x_i - x_j\|) = \frac{1}{2(\Delta - \delta)}\left(\frac{\|x_i - x_j\| - \delta}{\Delta - \|x_i - x_j\|}\right)^2$$
$$\Rightarrow w_{ij} = \frac{1 - \frac{\delta}{\|x_i - x_j\|}}{(\Delta - \|x_i - x_j\|)^3},$$

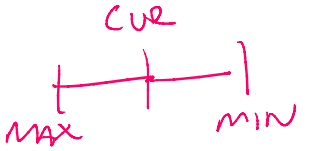
ENCOURAGES CLOSENESS

⇒ CLOSE TO MIN DIST δ

$$\frac{1}{2}\frac{(CUR - MIN)^2}{(MAX - CUR)^2}\,(MAX - MIN)^2$$

↓

MIDPOINT of RATIO of (DIST TO MIN (POSITIVE)) / (DIST TO MAX) / TOTAL DIST (POSITIVE) ALLOWABLE



TO MINIMIZE THIS . . .

TOP RATIO → CUR CLOSE TO MIN

BOTTOM ↔ LARGE

Lloyd's Algorithm

$$V_i(x) = \{ p \in \mathcal{D} \mid \|x_i - p\| \leq \|x_j - p\|, \forall j \neq i \}$$

- $V_i(x)$: Voronoi cell
- DIST BETWEEN ROBOT i AND p
- DIST BETWEEN ROBOT j AND p
- ROBOT i MANAGES ITS OWN CELL

p – POINTS IN WORK SPACE

$\mathcal{D}$ – ENTIRE WORKSPACE

D – VORONOI CELL SPACE

$$\phi : D \to \mathbb{R} \rightsquigarrow \phi(p) = y, \; y \in \mathbb{R}$$

ϕ IS A MEASURE OF THE SPACE IN A VORONOI CELL RELATIVE TO ITSELF

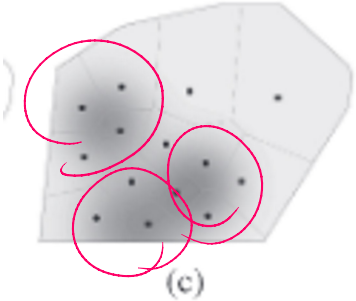
Here, the function $\varphi : D \to \mathbb{R}$ measures the relative importance of points in the environment, i.e., if $\varphi(p) > \varphi(q)$, then the point p is more important than the point q for the robot ensemble. As

$$\mathcal{E}(x) = \sum_{i=1}^{N} \int_{V_i(x)} \|x_i - p\|^2 \phi(p) \, dp$$

- $\sum_{i=1}^{N}$: FOR ALL ROBOTS
- $\int_{V_i(x)}$: SUM IN CELL
- $\|x_i - p\|^2$: L2 NORM OF DIST p
- $\phi(p)$: IMPORTANCE OF p IN CELL

AS p BECOMES LESS IMPORTANT THE RESULT OF THE INTEGRAL BECOMES SMALLER

★ THIS EFFECTIVELY EMPHASIZES STAYING CLOSE TO IMPORTANT AREAS



(c)

p is more important than the point q for the robot ensemble. As before, taking the partial derivative of this locational cost gives

$$\frac{\partial \mathcal{E}}{\partial x_i} = 2\int_{V_i(x)} (x_i - p)\varphi(p)dp.$$

The reason why the application of Leibniz rule at the area over which the integral is evaluated does not seem to matter is because whatever area is moved into V_i by the infinitesimal movement of x_i, exactly the same area is lost in some other cell, i.e., the effects cancel out.

In calculus, the **Leibniz integral rule** for differentiation under the integral sign, named after Gottfried Leibniz, states that for an integral of the form

$$\int_{a(x)}^{b(x)} f(x,t)\,dt,$$

where $-\infty < a(x), b(x) < \infty$ and the integral are functions dependant on x, the derivative of this integral is expressible as

$$\frac{d}{dx}\left(\int_{a(x)}^{b(x)} f(x,t)\,dt\right) = f(x,b(x))\cdot\frac{d}{dx}b(x) - f(x,a(x))\cdot\frac{d}{dx}a(x) + \int_{a(x)}^{b(x)}\frac{\partial}{\partial x}f(x,t)\,dt,$$

where the partial derivative $\frac{\partial}{\partial x}$ indicates that inside the integral, only the variation of $f(x,t)$ with x is considered in taking the derivative.[1]

LEIBNIZ RULE IS FOR INTEGRALS WHERE THE LIMITS OF INTEGRATION ARE FUNCTIONS.

WE USE LEIBNIZ RULE HERE BECAUSE THE SHAPE OF THE VORONOI CELL IS CHANGING AS THE FORMATION ADJUSTS LIMITS OF INTEGRATION ARE THE BOUNDARIES OF THE CELL

Following the program laid out in the previous paragraphs of using a gradient descent flow as a way of enabling LaSalle Invariance Principle to be applicable, gives us

$$\dot{x}_i = 2\int_{V_i(x)} (p - x_i)\varphi(p)dp = 2m_i(x)(x_i - \rho_i(x)),$$

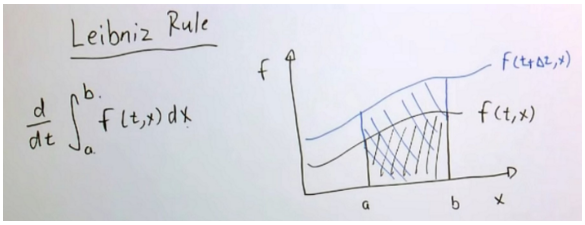
where $m_i(x) = \int_{V_i(x)} \varphi(p)dp$ and $\rho_i(x)$ are, respectively, the mass and center of mass of the i:th Voronoi cell.

$m_i(x) = \int_{V_i(x)} \phi(p)dp$ — MASS OF CELL i

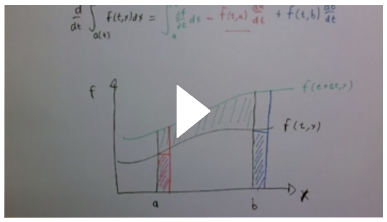
$\rho_i(x)$ — COM OF CELL i

$$\rightarrow \dot{x}_i = 2\int_{V_i(x)} (p - x_i)\,\phi(p)dp$$

$$= 2m_i(x)(x_i - \rho_i(x))$$



Leibniz integral rule



$$\rightarrow = 2\left[\int_{V_i(x)} \phi(p)dp\right]\left[(x_i - \rho_i(x))\right]$$

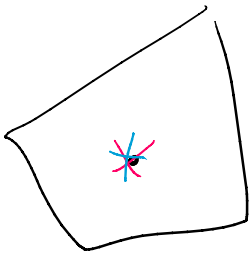
MASS — DIST FROM COM

$$\star \quad p - x_i = x_i - \rho_i(x)$$

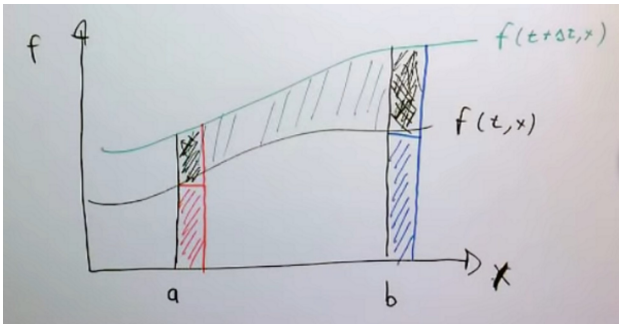
DIST BTWN POINT IN WORKSPACE & ROBOT = DIST BTWN ROBOT & COM

$$\frac{d}{dt}\int_{a(t)}^{b(t)} F(t,x)dx = \int_a^b \frac{\partial f}{\partial t}dx - f(t,a)\frac{da}{dt} + f(t,b)\frac{db}{dt}$$

WORKSPACE & ROBOT



ROBOT COM POINT IN WORKSPACE



$$\frac{1}{2}\frac{1}{m}\frac{\partial \mathcal{E}}{\partial x} = x - p^{com}$$

IF ROBOT POSITION REACHES COM OF VORONOI CELL THEN ERROR → ∅

SUM FOR EACH ROBOT

(A) VECTOR FROM CUR POS TO CUR COM

(B) VECTOR FROM CUR VELOCITY TO CHANGE IN COM/TIME

Note that the evolution of this function along the robot trajectories can be written as

$$\frac{d}{dt}(\mathcal{E}(x,t)) = \sum_{i=1}^{N}(x_i - \rho_i(x,t))(\dot{x}_i - \frac{\partial \rho_i}{\partial t}(x,t) - \sum_{j=1}^{N}\frac{\partial \rho_i}{\partial x_j}(x,t)\dot{x}_j) =$$

$$(x - \rho(x,t))^T(I_N - \frac{\partial \rho}{\partial x}(x,t))\dot{x} - \frac{\partial \rho}{\partial t}(x,t)),$$

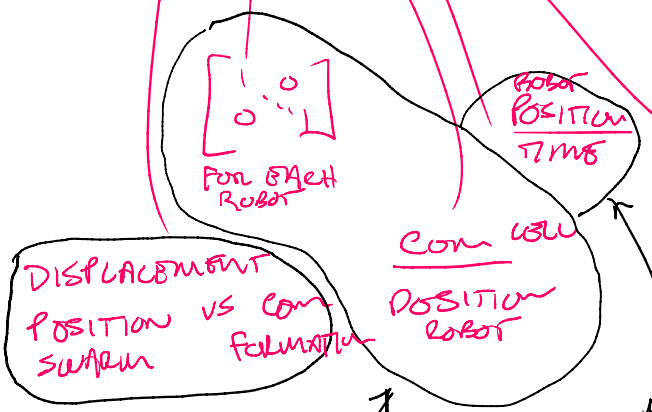
where, for simplicity, we use the short-hand notation $\rho(x,t) = (\rho_1(x,t),\ldots,\rho_N(x,t))$. This computation reveals how the design of the coordination strategy above should be modified to deal with time-varying functions φ. Specifically, if one sets

OBJECTIVE

GAIN

(C) VECTOR FROM ABOVE TO SUM OF PAIRED ROBOTS' VELOCITY OF COM/ORIGIN ROBOT'S POSITION MULTIPLIED BY CUR VELOCITY

IF THIS STOPS MOVING, THIS TERM GOES TO ZERO



SWARM MOVEMENT

IF SWARM STOPS MOVING THEN THIS GOES TO ZERO

→ SWARM HAS MOVED INTO CORRECT FORMATION POSITION.

→ STOP MOVING

FORMATION MOVEMENT

IF FORMATION STOPS MOVING THEN THIS GOES TO ZERO

→ FORMATION ARRIVED AT TARGET.

→ THE SHAPE OF THE FORMATION IS ABSTRACT AND MOVES AHEAD OF SWARM... BUT ONLY SO FAR AHEAD

ROBOT IN CELL

GAIN. TELLS EACH ROBOT WHICH WAY

GAIN. TELLS EACH ROBOT WHICH WAY TO MOVE TO GET TO COLL CENT.

POSITION.

→ STOP MOVING INDIVIDUAL ROBOTS, MOVE THE ENTIRE FORMATION

SWARM . . . BUT ONLY SO FAR AHEAD

→ THE FORMATION CHANGES VELOCITY TOWARDS GOAL AS SWARM ALIGNS WITH DESIRED FORMATION

SWARM VS FORMATION

STOP MOVING WHEN SWARM IS IN FORMATION

$$\mathcal{E}(x,t) = \frac{1}{2}\sum_{i=1}^{N} \|x_i - \rho_i(x,t)\|^2.$$

$(\rho_1(x,t),\ldots,\rho_N(x,t))$. This computation reveals how the design of the coordination strategy above should be modified to deal with time-varying functions φ. Specifically, if one sets

$$\left((I_N - \frac{\partial \rho}{\partial x}(x,t))\dot{x} - \frac{\partial \rho}{\partial t}(x,t)\right) = k(\rho(x) - x),$$

then the evolution of the error function $\mathcal{E}$ takes the form

$$\frac{d}{dt}(\mathcal{E}(x(t),t)) = -(x - \rho(x,t))k(x - \rho(x,t)) = -2k\mathcal{E}(x(t),t),$$

and hence $\mathcal{E}(x(t),t) = \mathcal{E}(x(0),0)\exp(-2kt)$, guaranteeing exponential convergence. The implementation of this design, however, is challenging from a distributed viewpoint, because it requires the inversion of the matrix $(I_N - \frac{\partial \rho}{\partial x}(x,t))$ to compute the robots' motion. The matrix is sparse, but its inversion is not. One can tackle this, for instance, by approximating the inverse matrix with the Taylor series expansion which, as the matrix sparse, is amenable to distributed implementation [57]. An example of this approach is shown in Fig. 3 (bottom row), where a team of robots execute the dynamic coverage control algorithm.

A series of different examples of employing this method for going from $u \in \mathbb{R}^2$ via (v,ω) to (ω_r,ω_l) are shown in Fig. 3 for a number of the coordinated controllers discussed in this paper.

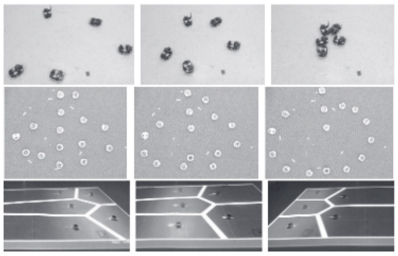
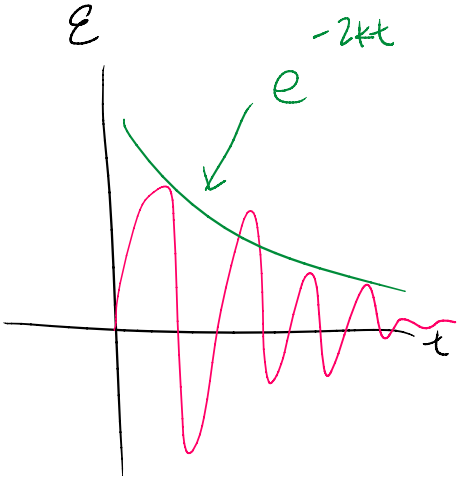


Fig. 3 Top: Five robots are solving the rendezvous problem. Middle: 15 robots are forming a "G" by executing an energy-based formation control strategy resulting in a weighted consensus equation. Bottom: the robots move to cover a time-varying density function by minimizing the locational cost.



SOLVE DIFF EQ

$$\frac{d}{dt}\mathcal{E} = -2k\mathcal{E}$$

$$\rightarrow \mathcal{E} = \mathcal{E}_0 e^{-2kt}$$

$$\left((I_N - \frac{\partial \rho}{\partial x}(x,t))\dot{x} - \frac{\partial \rho}{\partial t}(x,t)\right) = k(\rho(x) - x),$$

LET $a = \frac{\partial \rho}{\partial x}$, $b = \frac{\partial \rho}{\partial t}$

$$\rightarrow (I - a)\dot{x} - b = k(\rho - x)$$

?

$$\dot{x} - a\dot{x} - b = k\rho - kx$$

SO THIS IS HOW WE KNOW K MUST BE INVERTED

AND WHY WE SHOULD USE A TAYLOR SERIES EXPANSION.

A different take on the locational problem discussed in Section 3.1 is to consider the optimization by the robots of the locational cost subject to fair partitioning of the areas of the environment tasked to each of them [58]–[60]. For instance, in case of heterogeneous robots, some with more mobility than others, the fast robots may be tasked with larger regions than the slow ones. For the case of homogeneous teams, it makes sense to

FAST ROBOTS CAN TAKE UP MORE SPACE THAN SLOW ONES

cational cost subject to fair partitioning of the areas of the environment tasked to each of them [58]–[60]. For instance, in case of heterogeneous robots, some with more mobility than others, the fast robots may be tasked with larger regions than the slow ones. For the case of homogeneous teams, it makes sense to prescribe an equitable partition among the robots, where the mass of each region is the same for all. Interestingly enough, this is not guaranteed by the centroidal Voronoi configurations achieved by the Lloyd's algorithm. To illustrate how to deal

MORE SPACE THAN SLOW ONES
→ CHANGES VORONOI CELL
AND CHANGES CENTROID.

Integrator

From Wikipedia, the free encyclopedia

HERE = EFFECT OF ACTIONS OVER TIME

For the business function, see systems integrator.

An **integrator** in measurement and control applications is an element whose output signal is the time integral of its input signal. It accumulates the input quantity over a defined time to produce a representative output.

able to go from integrators to full-blown robot kinematics in order to actually deploy these control laws. The standard manner in which this is done is to use the velocities resulting from the coordinated control algorithms as "plans" and then wrap nonlinear controllers around these plans in order to deploy them on real robotic systems. Rather than characterizing all the different

INTEGRATOR → KINEMATICS
W/ DIRECT CONTROL OVER VELOCITY

WRAP NON-LINEAR CONTROLLER
AROUND VELOCITY "PLAN"

Control design techniques for nonlinear systems also exist. These can be subdivided into techniques which attempt to treat the system as a linear system in a limited range of operation and use (well-known) linear design techniques for each region:

- Gain scheduling

Those that attempt to introduce auxiliary nonlinear feedback in such a way that the system can be treated as linear for purposes of control design:

- Feedback linearization

And Lyapunov based methods:

- Lyapunov redesign
- Control-Lyapunov function
- Nonlinear damping
- Backstepping
- Sliding mode control

https://en.wikipedia.org/wiki/Nonlinear_control

MPC?
RL?

real robotic systems. Rather than characterizing all the different ways in which this has been done on a large number of different types of platforms, we illustrate here this on two standard classes of robots, namely wheeled, differential drive robots and aerial quadcopters. The common denominator for both robots is the use of the concept of differential flatness [62],[63] to carry out the control design.

I THINK THIS MEANS THAT IF WE CAN DETERMINE THE PATH TO THE DESTINATION → THEN WE CAN ALSO DETERMINE ALL OF THE OUTPUTS ∀ TIME THAT ACHIEVES THIS.

AND I ASSUME THIS WORKS ESPECIALLY WELL FOR ROBOTS WHICH CAN BE CONTROLLED AS PARTICLES.

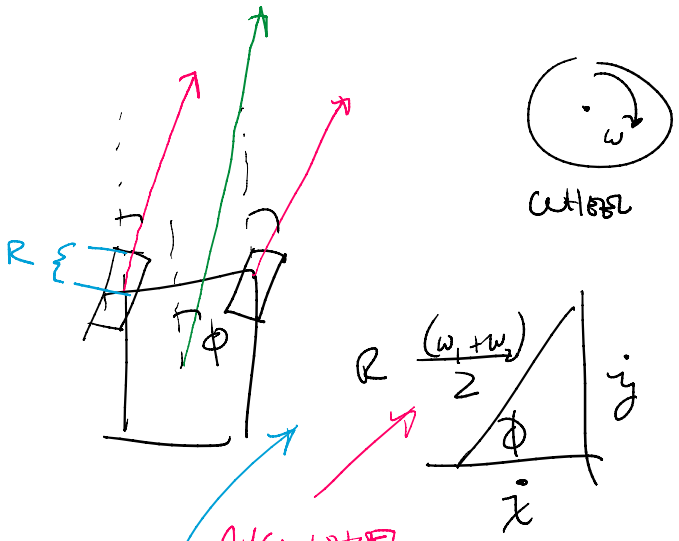
↑

ACHIEVES THIS.

One of the most commonly used robotic platforms is the wheeled, differential-drive ground robot. It is equipped with two independently controlled wheels of radius R, where the control inputs are the angular velocities of the right ω_r and left ω_l wheels. If the wheel axis has length L, then the kinematics of the differential drive robot is

$$\dot{x} = \frac{R}{2}(\omega_r + \omega_l)\cos\phi,$$
$$\dot{y} = \frac{R}{2}(\omega_r + \omega_l)\sin\phi,$$
$$\dot{\phi} = \frac{R}{L}(\omega_r - \omega_l).$$

ϕ IS HEADING



AVG WHEEL VELOCITIES

$$\omega = \frac{\theta}{t}$$

$$v = \frac{s}{t}$$

$$v = \frac{r\theta}{t}$$

$$v = r\omega$$

$$\bar{x} = x + \ell\cos\phi,$$
$$\bar{y} = y + \ell\sin\phi.$$

In fact, these variables work as flat outputs. Their derivative is given by

$$\dot{\bar{x}} = v\cos\phi - \ell\omega\sin\phi,$$
$$\dot{\bar{y}} = v\sin\phi + \ell\omega\cos\phi.$$

WHAT THE HECK DOES FLAT ACTUALLY MEAN?

ABLE TO BE CONTROLLED LINEARLY ON A MANIFOLD?

https://roar.me.columbia.edu › research-projects › differential-flatness

Differential Flatness | Roar Lab - Columbia University

For differentially flat systems, the planning and control problems can be solved effectively using the linear and controllable forms. These methods have applications in the control of under-actuated robots, nonholonomic robots, and space robots. Design of Systems Contact Us sa3077@columbia.edu

SO DOES THIS MEAN THAT THE MAPPING OF WHEEL VELOCITIES, ETC. TO $\mathbb{R}^2$ ARE CONTROLLABLE LINEARLY JUST FROM $\mathbb{R}^2$?

i and use x_{N_i} to denote the states of all robots adjacent to Robot i (however the adjacency relationship happens to be defined), then one can realize that all the costs already discussed were of the following form:

$$\mathcal{E}(x) = \sum_{i=1}^{N} F_i(x_i, x_{N_i}).$$

For example, in the weighted formation control costs, we had

$$F_i(x_i, x_{N_i}) = \sum_{(i,j)\in E} \mathcal{E}_{ij}(\|x_i - x_j\|),$$

while the locational cost for coverage control was

$$F_i(x_i, x_{N_i}) = \int_{V_i(x_i, x_{N_i})} \|x_i - p\|^2 \varphi(p) dp.$$

The reason why a gradient descent algorithm is particularly appropriate for coordinated control is that the adjacency relationship implied in the cost is made explicit by the descent algorithm in that

$$\frac{\partial \mathcal{E}(x)}{\partial x_i} = \frac{\partial F_i(x_i, x_{N_i})}{\partial x_i} + \sum_{(i,j)\in E} \frac{\partial F_j(x_j, x_{N_j})}{\partial x_i},$$

i.e., Robot i can evaluate this expression solely by having access to the states of adjacent agents. This preservation of adjacency

GRADIENT DESCENT FLOW

→ GRADIENT DESCENT THROUGH MOVEMENT OF DISTRIBUTED AGENTS??

I THINK THIS CONTAINS AN IMPLIED RELATIONSHIP WITH OTHERS BEYOND THE IMMEDIATE PAIR

The second reason why the gradient descent flow is useful is that even through the introduction of a strictly positive (possibly state-dependent) gain

$$\dot{x}_i = -\gamma_i(x_i, x_{N_i}) \frac{\partial \mathcal{E}(x)}{\partial x_i}$$

leads to

$$\frac{d\mathcal{E}(x(t))}{dt} = -\frac{\partial \mathcal{E}(x(t))}{\partial x}^T \Gamma(x) \frac{\partial \mathcal{E}(x(t))}{\partial x} = -\left\| \frac{\partial \mathcal{E}(x(t))}{\partial x} \right\|^2_{\Gamma(x)} \leq 0,$$

where $\Gamma(x) > 0$ is a positive definite, diagonal matrix with the individual gains on the diagonal.

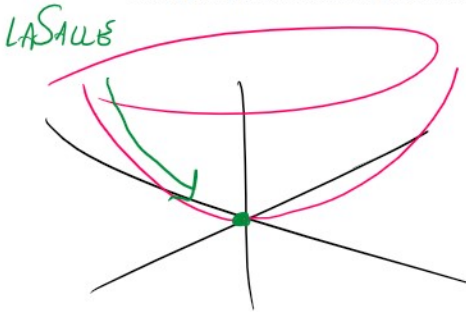
As a consequence, through LaSalle Invariance Principle, we can (subject to sufficient regularity assumptions on the cost) draw the conclusion that the state converges to the set of stationary points, i.e., points where the gradient of the cost is zero.

DIAGONALIZED SYMMETRIC MATRIX ⇒ UNIQUE EIGENVALUES

THIS REMINDS ME OF A SINGULAR VALUE DECOMPOSITION

$\Gamma(x)$ HAS INDIVIDUAL GAINS ON THE DIAGONAL

LASALLE

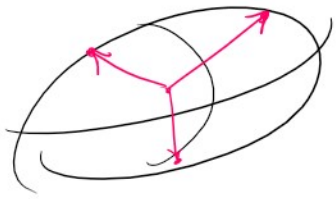


$\Gamma(x) \succ 0$

PD

THIS MEANS THAT GAINS WILL BRING SYSTEM TO CONVERGE AT ZERO

DIRECTIONS of GAINS



ENERGY ELLIPSOID

$$\frac{d\mathcal{E}(x(t))}{dt} = -\frac{\partial \mathcal{E}(x(t))}{\partial x}^T \Gamma(x) \frac{\partial \mathcal{E}(x(t))}{\partial x} = -\left\| \frac{\partial \mathcal{E}(x(t))}{\partial x} \right\|^2_{\Gamma(x)} \leq 0,$$

WHAT IS THIS?

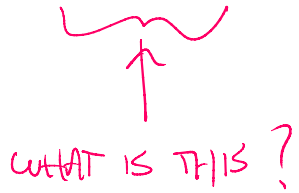
https://noisrucer.github.io/linearalgebra/2_5_Norms/

L2 Norm (Euclidean Norm)

We're more familiar with L_2 norm especially in machine learning fields. Using the above equation, L_2 norm is denoted as

$$L_2 \; norm : ||x||_2 = \sqrt{\sum_i |x_i|^2}$$

$dt \qquad \partial x \qquad \partial x \qquad \| \quad \partial x \quad \|_{\Gamma(x)}$



https://math.stackexchange.com/questions/3477017/matrix-in-norm-subscript-notation

1 Answer Sorte

2

If the P_i are positive definite, $\|x\|_{P_i}^2$ may be either

$$x^T P_i x \quad \text{or} \quad x^T P_i^{-1} x.$$

Share Cite Follow

L2 Norm (Euclidean Norm)

We're more familiar with L_2 norm especially in machine learning fields. Using the above equation, L_2 norm is denoted as

$$L_2 \; norm : ||x||_2 = \sqrt{\sum_i |x_i|^2}$$

and is also called **Euclidean norm** which is the Euclidean distance. Since L_2 norm is very frequently used in machine learning fields, it is simply denoted as $||x||$ without the subscript 2. The L_2 norm can be calculated by $x^\top x$.