A Projected Lloyd's Algorithm for Coverage Control Problems

Yoav Palti, Supervisor: Associate Professor Daniel Zelazo

Faculty of Aerospace Engineering, Technion, Haifa, Israel

M.Sc. Seminar

December 10, 2018

Table of Contents

Table of Contents

About Me

- Yoav Palti, B.Sc in Aerospace Engineering, Technion, 2012
- Aeronautical algorithms engineer, IAF, since 2013
- M.Sc student since 2014

Motivation

- Covering an area - (relatively) easy

Motivation

- Covering an area - (relatively) easy
- Covering an area with not sufficient amount of sensors - not so easy
  - *Requires better definition of behaviour*

Motivation

- Covering an area - (relatively) easy
- Covering an area with not sufficient amount of sensors - not so easy
  - *Requires better definition of behaviour*
- Maintain contact with home base (at least in steady state) - hard

Problem Formulation

- There is some area $A \in \mathbb{R}^2$ That we aim to cover
- We have set of sensors $S = \{s_1 \ldots s_n\}$ located in positions $p_i \in \mathbb{R}^2$ (for $i = 1, \ldots, n$) at time t
  - Each sensor has coverage radius R (assuming all sensors are identical)
  - Each sensor can cover a disk $D\left(p_i\left(t\right), R\right) \subset \mathbb{R}^2$, centred at $p_i\left(t\right)$
- Thus, the coverage:

$$D\left(p_i\left(t\right), R\right) = D_i\left(t\right) = \{x \in \mathbb{R}^2 \mid \|x - p_i\left(t\right) \leq\|_2 R\}. \quad (1)$$

- We also assume $D_i\left(t\right) < A$

Problem Formulation

Coverage Constraint:

- A given area inside the area $A_m \subset A$ must be covered always (e.g. ground station).

A *configuration*: A configuration c at time t is the stack of the sensor positions at time t,

$$c(t) = \begin{bmatrix} p_1^T(t) & \cdots & p_n^T(t) \end{bmatrix}^T \in \mathbb{R}^{2n}. \tag{2}$$

Notice

Since $D_i(t) < A$, there is no one configuration that can cover the entire area A at once

Problem Formulation

Our goal is to find the set of configuration C which contains configurations that all together provide full coverage of the area A, and yet maintains the coverage of A_m.

Problem

Find the set $\mathtt{C} = \begin{bmatrix} c_1^T(t_1) & \cdots & c_n^T(t_n) \end{bmatrix}^T$ *that provide coverage at time* i *to some area* $A_c(t_i)$*, such that:*

1. *after time* n*, each point of* A *was visited at least once,*
2. *At each time there was coverage to some area* $A_s m \subset A_m$*.*

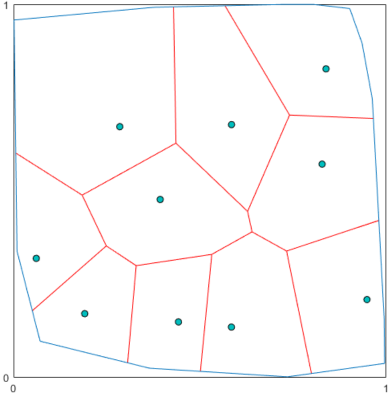
Literature Review

Table of Contents

Lyaponov Stability

Voronoi Partitioning

Let's start with a simple intuitive explanation...



Voronoi Partitioning

While being a method to partition an area with some cost function, the is a widely-used representation in the coverage problem [1] [2] [3].

[1] Cortes, J. and Martinez, S. (2004). Coverage control for mobile sensing networks. Robotics and Automation, Vol. 20, No. 2, 2004, p. 13, doi:10.1109/TRA.2004.824698

[2] Hussein, I. I. and Stipanovic, D. M., "Effective Coverage Control for Mobile Sensor Networks With Guaranteed Collision Avoidance," IEEE Transactions on Control Systems Technology, Vol. 15, No. 4, 2007, pp. 642–657, doi:10.1109/TCST.2007.899155

[3] Du, Q., Faber, V., and Gunzburger, M., "Centroidal Voronoi Tessellations: Applications and Algorithms," SIAM Review, Vol. 41, No. 4, 1999, pp. 637–676, doi:10.1137/ S0036144599352836

Voronoi Partitioning

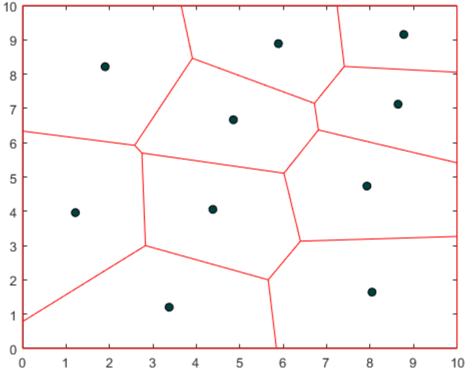
The Voronoi Diagram of a region $\Omega \subset \mathbb{R}^2$ is the set of partitions $\mathcal{V} = \{V_i \mid \cup V_i = \Omega\}$, generated by the generators $\mathcal{Z} = \{z_1, \ldots, z_n \mid z_i \in \Omega\}$, such that

$$V_i = \{q \in \Omega \mid \|q - z_i\| \leq \|q - z_j\| \forall z_i, z_j \in \mathcal{Z}\}, \tag{3}$$

where V_i corresponds to the i-th element of $\mathcal{Z}$, and $\|\cdot\|$ denotes the Euclidean distance.

Central Voronoi Tessellations

And yet again, let's have an intuitive explanation...



Central Voronoi Tessellations

Let us define a density function, ρ_i, for each Voronoi partition V_i. Then, we can define the center of mass for each partition as

$$z_i^* = \frac{\int_{V_i} y\rho(y)dy}{\int_{V_i} \rho(y)dy}. \tag{4}$$

If a generator $z_i = z_i^* \, \forall \, V_i$, we call this partitioning a *centroidal Voronoi tessellation* (CVT).

Lloyd's Algorithm

Now that we know what Central Voronoi Tessellations are, we need to know how to calculate them.
Stuart P. Lloyd, an Electrical Engineer, invented an algorithm that deals with PCM quantization [4].
It appears that the algorithm is very useful for calculating CVT's.
Moreover, It is possible to build a controller based on this algorithm.

---

[4] Lloyd, S., "Least squares quantization in PCM," IEEE Transactions on Information Theory, Information Theory, IEEE Transactions on, IEEE Trans. Inform. Theory, Vol. 28, No. 2, 1982, pp. 129–137, doi:10.1109/TIT.1982.1056489

Lloyd's Algorithm

Algorithm 1 Lloyd's Algorithm

1: Calculate the Voronoi diagram for the current agents positions.
2: Calculate the center of mass for every cell.
3: Move the agents to the center of mass.

According to Cortes et al., if we define agent i position as p_i and the i's partition centroid as C_{V_i}, then for some proportional constant k_{prop}, the controller can be defined as:

$$u_i = -k_{prop}\left(p_i - C_{V_i}\right) \tag{5}$$

Moreover, this controller is locally asymptotically stable.

Formation Control

Projection Operator

The projection linear operator is defined as a linear transformation P from a vector space to itself such as $P^2 = P$. In other words, the transformation P is idempotent.

Table of Contents

Projected Lloyd's Algorithm

We should supply a solution for the "Coverage Constraint"

Projected Lloyd's Algorithm

We should supply a solution for the "Coverage Constraint"

Reminder

A given area inside the area $A_m \subset A$ must be covered always

Projected Lloyd's Algorithm

We should supply a solution for the "Coverage Constraint"

Reminder

A given area inside the area $A_m \subset A$ must be covered always

We came up with a rather simple solution for this problem.

Projected Lloyd's Algorithm

Algorithm 2 Projected Lloyd's Algorithm (PLA)

1: Calculate the Voronoi diagram for the current agents positions.
2: Calculate the center of mass for every cell.
3: Project the center of mass of every cell to the area constraint limiting polygon.
4: Move the agents the projected center of mass.
5: Repeat until converge.

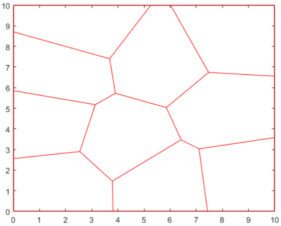
Writing this algorithm as a controller:

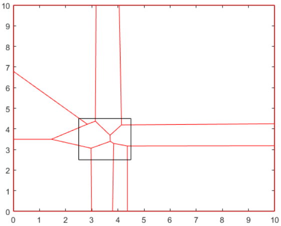
$$u_i = -k_{prop}\left(p_i - \mathbf{proj}\left(C_{V_i}\right)\right) \quad (6)$$

Projected Lloyd's Algorithm

In the following example:

- Left - regular CVT, build with the original Lloyd's Algorithm.
- Right - A partitioning built with the PLA.





Projected Lloyd's Algorithm

Theorem

The projected Lloyd's Algorithm is locally asymptotically stable

As the projection is a linear operator, the controller is also locally asymptotically stable, and the proof is virtually the same as was given by Cortes et al. The proof is based on a proposal of Lyaponov function, and then using the direct Lyaponov method to prove the stability

Proof of stability

As the projection is a linear operator, the controller is also locally asymptotically stable, and the proof is virtually the same as was given by Cortes et al.

Problem Solution Algorithm

So far, we've given solution for:

- Covering a given area using Voronoi partitioning
- Partition and area such that the coverage constraint is fulfilled.

Therefore, we are ready for the problem solution algorithm...

Problem Solution Algorithm

Algorithm 3 Problem Solution Algorithm

1: Using some random initial guess, partition the whole area using PLA.

2: For each partition (assuming that the agents can actually cover each partition with their coverage radius), calculate the CVT. The initial positions for the CVT calculation is the previous partition CVT.

Lloyd's Algorithm and Formation Control

- Problem solved.

Lloyd's Algorithm and Formation Control

- Problem solved.
- Make it more interesting...

Lloyd's Algorithm and Formation Control

- Problem solved.
- Make it more interesting...
- Combine Lloyd's Algorithm with distance-based formation control!

Lloyd's Algorithm and Formation Control

- Problem solved.
- Make it more interesting...
- Combine Lloyd's Algorithm with distance-based formation control!
  - create and maintain spatial properties partially or fully (We do not provide a condition where this combination meets the requirements).

Lloyd's Algorithm and Formation Control

As both of the controllers are convex, we propose to simply combine them with some coefficient:

TODO: INSERT THE CONTROLLER (already written in the "Deployment and Formation controller" document)

Lloyd's Algorithm and Formation Control

As both of the controllers are convex, we propose to simply combine them with some coefficient:
TODO: INSERT THE CONTROLLER (already written in the "Deployment and Formation controller" document)

Theorem

The combined controller is Locally Asymptotically Stable

Lloyd's Algorithm and Formation Control

Proof.

TODO (already written in the "Deployment and Formation controller" document) □

Table of Contents

Some Simulation

List of simulations to create:

- 3 agents, 5 big partitions, no formation, no PLA
- 3 agents, 5 big partitions, no formation, PLA
- 10 agents, 5 big partitions, no formation, PLA
- 6 agents, 5 big partitions, some formation, PLA