Coordinated Construction by a Distributed Multi-Robot Sys. 1

[Ph.D. of Seung-kook Yun, MIT]

- Groups of robots that complete a complex assembly task using the maximal amount of parallelism, in a way that can adapt to the amount of construction material.

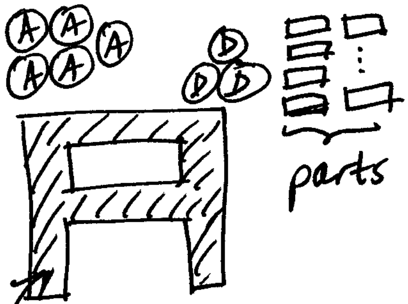
- 2 types of robots: part delivery robots (D) that locate + deliver parts, and assembly robots (A) that join the parts into desired objects.

- The target structure is given by a blueprint, a material-density function that encodes the object geometry and is known to all robots.

- The assembly robots partition the structure adaptively into subassemblies, and each robot is responsible for completion of a partition region.
  - Robots locally compute a Voronoi partition, weighted by the mass of all parts in the partition, and perform a gradient descent algorithm to balance the masses of the regions.

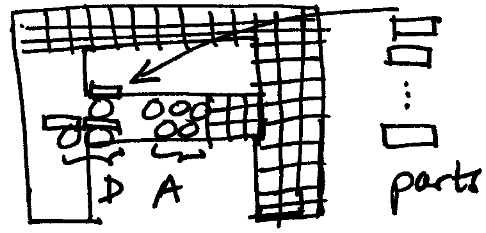
- The delivery robots locate parts in a cache and bring them to the assembly robots.
  - Want the assembly robots to work at approx. the same pace
  - Delivery occurs according to the demanding mass for each subassembly, the amount of remaining work (measured in the # of components that have to be added).

Ex



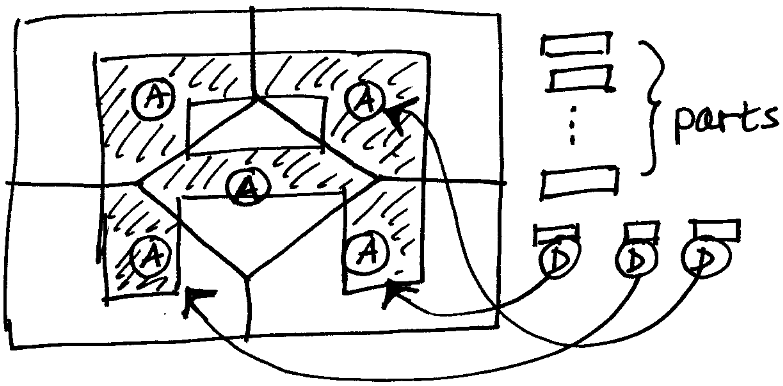
blueprint (fill with the parts)

Incremental solution



- Each robot travels the entire blueprint (slow)
- Requires knowledge of exact construction plan + placement of each part before execution

Proposed solution using <u>Voronoi cells</u>:



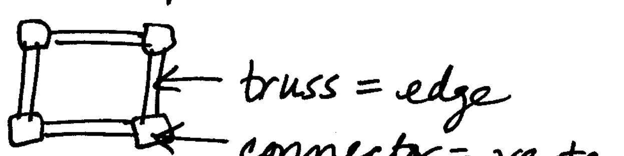
- Each assembly robot builds within its own Voronoi cell.

- Decentralized controllers are proposed for (1) partitioning, (2) part delivery, and (3) assembly.

- Construction involves <u>discrete</u> components.

ex) Truss structure:

- Can be modeled as a graph



<u>Problem Formulation</u>

N_d = # of D $\quad$ N_a = # of A

- Robots can communicate locally with other robots within their communication range.

- The domain is $Q \subset \mathbb{R}^N$ ($N=2$ or 3) or a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

(a) Domain is $Q \subset \mathbb{R}^N$:

- Robots are given a target density fn. $\phi_t : Q \to \mathbb{R}$

Density of construction material

- If components can be built independently + an assembling robot is capable of assembling all of them, then:

$$\phi_t = \sum_{u=1}^{z} \beta_u \psi_u ,$$

β_u weights the importance of uth component, z = # of components that can be assembled by Ⓐ.

(Importance = time to assemble the piece, time until piece is needed in the assembly, etc.)

- To represent truss structures, ϕ_t is defined point-wise on the grid that corresponds to the truss. Point density $\propto$ # of possible truss connections at the point.

(b) Domain is $\mathcal{G} = (\mathcal{V}, \mathcal{E})$:

$p_i \in \mathcal{V}$ is the position node (vertex) of robot i

$d(\cdot,\cdot) : \mathcal{E} \to \mathbb{R}^+$ = shortest distance measure between 2 vertices $[d(s,t) = \infty$ when $(s,t) \notin \mathcal{E}]$

$\phi_t(v)$ = vertex weight denoting the importance of a task at vertex v (target density fn.)

- Divide $\mathcal{G}$ into graph Voronoi partitions:

$$V_i = \{ v \in \mathcal{V} \mid d(v, p_i) < d(v, p_j),\ \forall j \neq i \}$$

- The nearest robot to v will execute the task at v. Each robot is allocated the task that includes its Voronoi partition V_i in G.
- Need to clarify assignment of a vertex with same distance to multiple robots; give priority to robot with the minimum ID:

$$v \in V_i \Rightarrow i = \min \{ j \mid d(v, p_i) = d(v, p_j) \}$$

- Let w_i be a weight; larger $w_i \Rightarrow$ larger region
Generalized Voronoi partition:

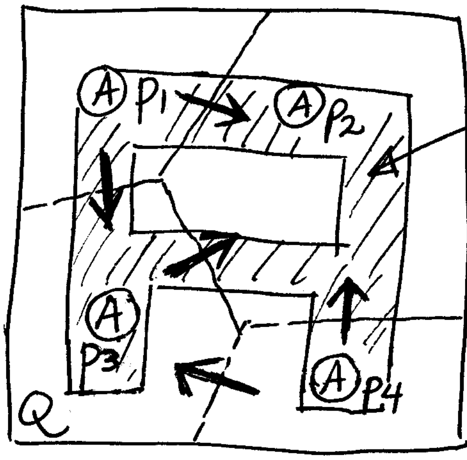
$$V_i = \{ v \in V \mid (d(v, p_i) - w_i) < (d(v, p_j) - w_j), \forall j \neq i \}$$

- Assumptions: (1) G, $\phi_t(v)$ is given to each robot
(2) $\phi_t(v)$ is fixed
(3) robots do not know locations of other robots ($\Rightarrow$ can't precompute the optimal config.)
(4) robots precompute the distance matrix $\underline{D}$ of G as a $|V| \times |V|$ symmetric matrix where $\underline{D}_{ij} = d(v_i, v_j)$.

Construction Algorithm:

① Deploy Ⓐ in Q or V
② Place Ⓐ at optimal task locations in Q or V (uses a distributed coverage controller)
Repeat until task completed or out of parts:
Ⓓ: carry source parts to Ⓐ with max. demanding mass
Ⓐ: assemble delivered parts after determining the optimal placement of the part in the assembly

Ex



Equal-mass partitioning

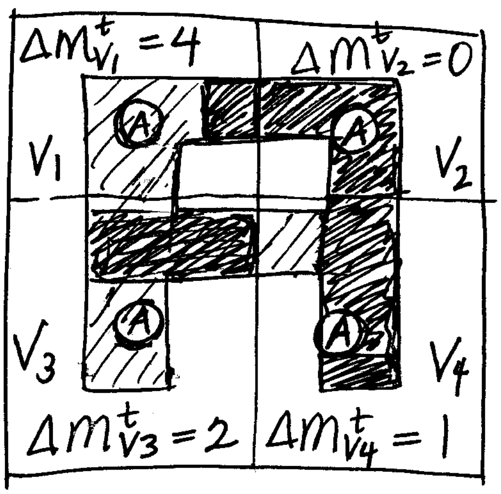
$\phi_t : Q \to \mathbb{R}$

p_i = position of A

Mass of robot i = size of total shaded region in its Voronoi region

→ = direction of motion of robot; component of normal to edge of Voronoi region

(combine to get resultant direction of robot's motion)



▩ = completed assembly

$\Delta m^t_{V_i}$ = demanding mass of region i
= area of ▨ in region i
− area of ▩ in region i

D is in V_4: among the neighboring regions (V_2, V_3), V_3 has a higher demanding mass
⇒ D moves to V_3.

D is in V_3: among the neigh. regions (V_1, V_4), V_1 has a higher demanding mass
⇒ D moves to V_1.

Decentralized equal-mass partitioning controller

- Given $q \in Q$, the nearest robot to q will execute the assembly task at q.
- Each robot is allocated the assembly task in its Voronoi partition:

$$V_i = \{ q \in Q \mid \|q - p_i\| \le \|q - p_j\|, \ \forall j \neq i \}$$

ϕ_t = density of parts

$$m_{V_i} = \int_{V_i} \phi_t(q)\,dq$$

[Cost function: $\mathcal{H}_0 = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2}\|q - p_i\|^2 \phi(q)\,dq$]
[for max. sensor coverage]

- Want each robot to have the same amount of assembly work (same # of truss elements)

$$\mathcal{H}_0 = \left(\frac{1}{n}\sum_{i=1}^{n} M_{V_i}\right)^n \qquad \mathcal{H} = \mathcal{H}_0 - \prod_{i=1}^{n} m_{V_i}$$

Cost function

- $\mathcal{H}$ is continuously differentiable
- Minimizing $\mathcal{H}$ leads to equal-mass partitioning:

$$\underset{\text{(arithmetic mean)}}{\frac{1}{n}\sum_{i=1}^{n} M_{V_i}} \ge \underset{\text{(geometric mean)}}{\left(\prod_{i=1}^{n} m_{V_i}\right)^{1/n}}$$

$\ge$ is $=$ only if all m_{V_i} are the same (then $\mathcal{H} = 0$)

- Can guarantee that $\mathcal{H}$ converges to a local minimum under the controller

- The robot controller continuously decreases the cost fn:

$$\dot{\mathcal{H}} \le 0, \quad t > 0.$$

$$\dot{\mathcal{H}} = \sum_{i=1}^{n} \frac{\partial \mathcal{H}}{\partial p_i} \dot{p}_i \qquad p_i = \text{position of robot } i$$

$\mathcal{N}_i$ = set of neighbors of robot i

Can derive $\dot{\mathcal{H}}$ as:

$$\dot{\mathcal{H}} = -\sum_{i=1}^{n} \prod_{l \notin \{i, \mathcal{N}_i\}} M_{v_l} \underbrace{\sum_{j = i, \mathcal{N}_i} \frac{\partial M_{v_j}}{\partial p_i} \prod_{\substack{k \in \{i, \mathcal{N}_i\}, \\ k \neq j}}^{n} M_{v_k}}_{\underline{J}_i} \dot{p}_i$$

- Decentralized controller:

$$\dot{p}_i = k \frac{\underline{J}_i}{\|\underline{J}_i\|^2 + \lambda^2}$$

k = positive control gain

λ = constant to stabilize controller even around singularities where $\|\underline{J}_i\|^2 = 0$.

- Only depends on variables of neighboring robots.

- The controller guarantees that $\mathcal{H}$ converges to either a local maximum or a global maximum:

$$\dot{\mathcal{H}} = -k \sum_{i=1}^{n} \frac{\|\underline{J}_i\|^2}{\|\underline{J}_i\|^2 + \lambda^2} \prod_{l \notin \{i, \mathcal{N}_i\}} M_{v_l}$$

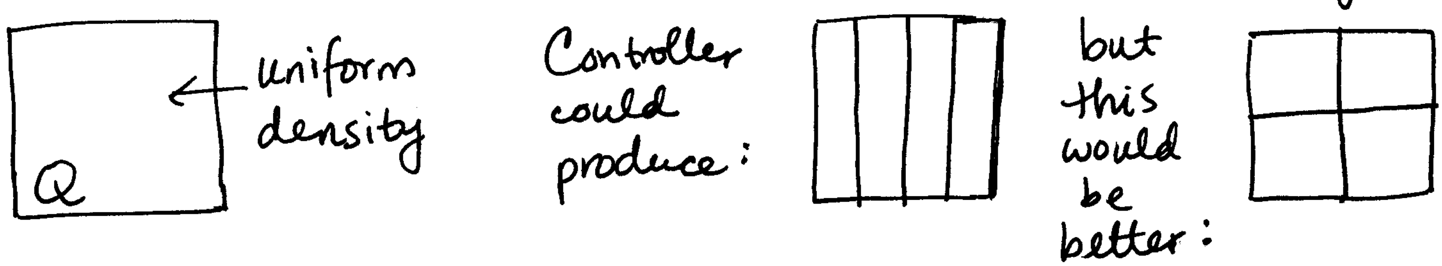
$k > 0$, $M_{v_l} > 0 \Rightarrow$ each term of $\dot{\mathcal{H}}$ is negative

Also, $\mathcal{H}$ is differentiable, and robot trajectories are bounded in Q

$\Rightarrow$ Controller keeps $\mathcal{H}$ decreasing until all $\underline{J_i = 0}$
(relocating the robots does not change $\mathcal{H}$)

Equal-mass partitioning with locational optimization ⊛

- Although the controller above leads robots to regions with equal masses, the region shapes may not be desirable in terms of robot travel time and communication range.



- Add locational optimization property to the controller:
  - A solution of loc. optim. is to locate robots at the centroids $\underline{C}_{V_i} = \dfrac{1}{M_{V_i}} \displaystyle\int_{V_i} \underline{q}\, \phi_t(\underline{q})\, d\underline{q}$ of the Voronoi cells
- Redefine $\mathcal{H}_0$ as: $\mathcal{H}_0 = \displaystyle\sum_{i=1}^{n} M_{V_i} \| \underline{C}_{V_i} - \underline{p}_i \|^2$
- New cost function: $\hat{\mathcal{H}} = \mathcal{H} + \gamma \mathcal{H}_0$ ($\gamma > 0$ can be tuned)
  - Can redefine $\dot{\underline{p}}_i$ to produce same convergence prop. for $\hat{\mathcal{H}}$.

⊛ Locational optimization: Optimization problems in operations research for placing facilities to minimize costs in terms of distance (transportation costs).