Distributed robotics:
Dynamic Routing and Motion Coordination of Large-Scale Vehicle Networks

Emilio Frazzoli
LIDS, Aeronautics and Astronautics
Massachusetts Institute of Technology

thanks to:
A. Arsie, M. Pavone (MIT), J. J. Enright (UCLA)
M. Savchenko, V. Sharma, P. Voulgaris (UIUC)
K. Savla, F. Bullo (UCSB)
Sonia Martinez (UCSD), Jorge Cortes (UCSC)

TDS group
March 9, 2007
Large-scale biological groups
Man-made mobile networks
What are the benefits of scale?

- Large groups of animals often exhibit sophisticated collective behaviors, arising from the local interaction of “agents” with individual goals, and limited ability to exchange information.
  - The emergence of complex collective behaviors from local interactions is a fascinating research topic.
  - Today large-scale teams or “swarms” of robots are becoming increasingly feasible; growing interest in motion coordination algorithms, scalable to large groups.

- Should we design/field “large” robotic teams?
  - Are there any tasks that are better suited to large teams?
  - What are the advantages of numbers? i.e., how well can a task be performed by $n+1$ agents, as opposed to $n$ agents?
  - Are there any disadvantages to numbers? What are the tradeoffs?
  - Given a task, can we determine what is the most appropriate size of a robotic team for best efficiency?
Outline

• Modeling Framework

• Dynamic vehicle routing

• Dynamic vehicle routing with differential constraints

• Collision avoidance for air traffic control
Control System

- A **control system** is a tuple \( V = (X, U, X_0, f) \) consisting of
  1. \( X \) is a differentiable manifold, called the **state space**;
  2. \( U \) is a compact subset of \( \mathbb{R}^m \) containing \( 0 \), called the **input space**;
  3. \( X_0 \) is a subset of \( X \), called the **set of allowable initial states**;
  4. \( f : \mathbb{R} \times X \times U \rightarrow TX \) is a \( C^\infty \)-map with \( f(t, x, u) \in T_xX \) for all \( (t, x, u) \in \mathbb{R} \times X \times U \), representing the dynamics of the system, i.e., \( \dot{x} = f(t, x, u) \)

- We refer to \( x \in X \) and \( u \in U \) as a **state** and an **input** of the control system, respectively.

- If \( f \) does not depend on time, the control system is said **time-invariant**.

- We will often consider **control-affine systems**, i.e., control systems with \( f(x, u) = f_0(x) + \sum_{a=1}^{m} f_a(x) u_a \). In such a case, we represent \( f \) as the ordered family of \( C^\infty \)-vector fields \( (f_0, f_1, \ldots, f_m) \) on \( X \).
A robotic agent is a tuple $A = (V, W, W_0, L, \text{stf}, \text{msg}, \text{ctl})$ consisting of:

1. A control system $V$;
2. A finite-dimensional set of internal states $W$ ($W$ needs not be countable).
3. A set of starting states $W_0 \subseteq W$.
4. A communication language $L = \{a, b, \ldots\} \cup \text{null}$. Elements of $L$ are called messages. The null message is a placeholder for “no message.”
5. A control function $\text{ctl} : \mathbb{R} \times X \times W \to U, (t, x, w) \mapsto u$.
6. A state transition function $\text{stf} : \mathbb{R} \times X \times W \times 2^L \to W, (t, x, w, \{m_1, m_2 \ldots\}) \mapsto w^+$.
7. A message-generation function $\text{msg} : \mathbb{R} \times X \times W \to 2^L$.

The additional components with respect to a control system can be referred to as control and communication law.

For convenience, let us assume that each agent can send a message to itself (e.g., a snapshot of its state at the end of the previous round).
A mobile robotic network $S$ is a tuple $(\mathcal{A}, \mathcal{E}, \mathcal{T}, L)$ consisting of

1. A finite collection of $n$ robotic agents $\mathcal{A} = \bigcup_{i=1}^{n} A_i$;
2. A set of communication links, i.e., relation $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$; if $(A_i, A_j) \in \mathcal{E}$, then agents $A_i$ and $A_j$ can exchange information.
3. A communication schedule $\mathcal{T} = \{t_l : l \in \mathbb{N}\} \subset \mathbb{R}$, consisting of an increasing sequence of time instants.

- At each time instant in $\mathcal{T}$, each robotic agent can communicate a message in $L$ to its neighbors, as defined by $\mathcal{E}$.
- Note that in practical applications, messages from $A_i$ to $A_j$ can be either actively sent by $A_i$, or measured from $A_j$. In other words, the message-generating function could be computed either by the source or by the destination of a message.
- If all the agents in $\mathcal{A}$ are identical (possibly apart from an identifier), the mobile network is said uniform.
In each interval in the communication schedule $\mathcal{T}$ (i.e., the time between $t_i$ and $t_{i+1}$), the following happens:

1. **[at $t_i^+$]:** Each agent collects all incoming messages and updates its own internal state applying the state-transition function $\mathrm{stf}$. The communication links are cleared after this step.

2. **[between $t_i$ and $t_{i+1}$]:** Each agent evolves under the closed-loop dynamics $\dot{x} = f(t, x, \mathrm{ctl}(t, x, w))$.

3. **[at $t_{i+1}^-$]:** Each agent computes and sends messages to its neighbors, using the message-generating function $\mathrm{msg}$. 
Examples of communication graphs

**Delaunay** the “dual” of the Voronoi partition; i.e., the intersection of the Voronoi regions of the endpoints of Delaunay edges is not empty.

**$r$-disk** all edges have length no greater than $r$.

**$r$-Delaunay** Intersection of Delaunay and $r$-disk.

**$r$-limited Delaunay** Like Delaunay, but Voronoi neighbors included only if the boundary between regions is no more than distance $r$ from either generator.

**Gabriel** All edges are such that minimum-radius circles drawn through endpoints do not contain any other node.

**Euclidean MST** The minimum-length tree through all the nodes.
Complexity notions

- Task specification \( H : \mathcal{A} \rightarrow \{ \text{true}, \text{false} \} \)

- **Time Complexity (TC):** Total number of rounds needed until \( H(\mathcal{A}) = \text{true} \), counted from the time the first agent in the network is active (wake-up).

- **Communication complexity**
  - Define \( cc : 2^L \rightarrow \mathbb{R}^+ \) as a measure of the communication cost of sending a certain number of messages at a time step. Assume that \( cc(\emptyset) = cc(\text{null}) = 0 \), and \( cc(M_1 \cup M_2) \leq cc(M_1) + cc(M_2) \) (i.e., the communication cost function is *subadditive*).
  - **Total communication complexity (CC):** sum of the costs of all messages exchanged until \( H(\mathcal{A}) = \text{true} \).
  - **Mean communication complexity (MCC):** average cost per round of all messages exchanged until \( H(\mathcal{A}) = \text{true} \). Note that \( CC = MCC \cdot TC \).

- **Space complexity:** Minimum dimension of the internal state \( W \).
The “merry-go-round” problem

- Consider \( n \) mobile agents (children) who can only move on a circle, with bounded speed.
- The children are blindfolded and have earplugs... i.e., they can only sense other children by touching their hands. The arms of the children are extended and have a given length \( 2\pi/n < r < 2 \).
- The children want to move in the same direction on the circle, and be uniformly spaced.
- There is a symmetry-breaking mechanism, e.g., a total order (on weight, height, age, stubbornness, etc.)
Formalization

- Robotic agents:
  - $X = S^1$, i.e., the unit circle. A coordinate is, e.g., $\theta$, with the position given as $(\cos \theta, \sin \theta)$.
  - The motion of each agent is given by $\dot{x} = (-u \sin \theta, u \cos \theta)$, with $u \in [-1, 1]$.
  - Communication graph: $r$-Delaunay (i.e., talk to neighbors within $r$).
Control and communication law

- Communication schedule: $\mathcal{T} = \{1, 2, \ldots \}$.
- Internal state $w = (\text{dir}, \text{priority})$
- The variable dir is initially set to either $\text{cw}$ or $\text{ccw}$. The variable priority is initially set to a unique value from a totally ordered set.
- Each agent sends a message containing $(\theta, \text{dir}, \text{priority})$ to its neighbors.
- If an agent receives a message with higher priority, it updates its dir and priority variables with the ones contained in the message.
- An agent sets its (absolute) velocity $|u|$ to $0 < k < 1/2$ times the distance to the nearest neighbor in direction dir (or to $kr$ if there are no neighbors in that direction). The sign of the velocity is decided based on the value of dir.
Simulation results
Performance analysis

- With our model of synchronous robotic network, we were able to prove the following:
  - Time needed to agree on a direction is $\Theta(1/r)$.
  - Time needed to converge to a $\epsilon$-uniform distribution is $O(n^2 \log n)$ and $\Omega(n^2)$.
  - Total comm. complexity to agree on direction is $\Theta(n^2/r)$.
  - Total comm. complexity to converge to a $\epsilon$-uniform distribution is $O(n^3 \log n)$ and $\Omega(n^3)$. 
Dynamic Vehicle Routing Problems

- **“User” model:**
  - Exogenous process generating “service requests”
  - Static or dynamic.
  - Stochastic or adversarial.
  - Complex task specs (e.g., LTL).

- **“System” model**
  - Vehicles subject to algebraic, differential, and integral constraints.
  - Local sensing and communications.
  - Limited computational resources.

- **Performance Criterion**
  - Quality of Service.

- Traveling Repairperson
- Dial-A-Ride
- Environmental Monitoring
- Mobile sensor networks
- Surveillance
- Search and Rescue
- Area Denial
- Crime prevention
- Security
- Network connectivity
- Emergency Relief
- Traffic congestion management.
User model
Service requests generated by a (spatio-temporal) Poisson point process, with:
- Time intensity $\lambda > 0$
- Probability density $\varphi > 0$ in $Q$.
- Service requests are fulfilled when visited by a vehicle.

System model
A single vehicle, moving on the plane with bounded speed, $\|\dot{p}\|_2 \leq V$.

Performance Criterion
QoS: Average waiting time $T$ between issuance of service request and its fulfillment.

- First formulated by Psaraftis (1988). Bertsimas and coworkers (DTRP, 1990-’93) provided lower bounds on the average waiting time and centralized policies providing optimal performance.
Single-Vehicle Euclidean Traveling Repairperson Problem

**Control law**

**sRH**: pick the **densest cluster** including at least a fraction $\eta \leq 1$ of all outstanding targets, visit all targets in the cluster in min time (solve a “local TSP”). Repeat. If there are no targets, move to the median (aka Fermat-Torricelli or Weber point).

**Properties**

- Optimal in the limit $\lambda \to 0$:

$$T = \frac{1}{AV} \min_p \int_Q \| p - q \|_2 \phi(q) dq$$

- Recovers the performance of the best known policy in the limit $\lambda \to \infty$.

$$T = \frac{\beta^2}{2 - \eta} \frac{A\lambda}{V^2} \quad (\beta = 0.7120 \pm 0.0002)$$

[Frazzoli and Bullo ’04]
Single-Vehicle Euclidean Traveling Repairperson Problem

\[ T_{\text{sRH}} < T_{\text{NN}} \text{ for } \eta < 0.7 \]
\[ T_{\text{sRH}} \approx 0.63 \, T_{\text{NN}} \text{ for } \eta \ll 1 \]

Nearest neighbor

\[ \lambda = 10 \]
\[ \lambda = 50 \]
\[ \lambda = 250 \]
\[ \lambda \rightarrow \infty \]
Multi-Vehicle Euclidean Traveling Repairperson Problem

System model
A team of $m$ vehicles, moving on the plane with bounded speed, $\|\dot{p}\|_2 \leq V$.

Control law - Preliminaries

Task assignment: Associate a virtual generator $g_i$ to each agent; let

$$
\mathcal{V}_i(g) := \{ q \in \mathcal{Q} : \| q - g_i \|_2 \leq \| q - g_j \|_2, \forall i \neq j \}
$$

be the $i$-th agent’s Voronoi cell (region of dominance/responsibility).

Fixed generators: Assign the generators arbitrarily, and let each agent run the sRH control law within its own region. Then, the QoS is:

$$
T = \frac{1}{AV} \sum_{i=1}^{m} \min_p \int_{\mathcal{V}_i(g)} \| q - p \|_2 \, dq \quad \text{for } \lambda \rightarrow 0
$$

$$
T = \frac{\beta^2 \lambda}{(2 - \eta)A^2V^2} \sum_{i=1}^{m} \text{Area}[\mathcal{V}_i(g)]^3 \quad \text{for } \lambda \rightarrow \infty
$$
Multi-Vehicle Receding Horizon control law

Control law

\textbf{mRH}: In parallel (possibly asynchronously), do the following:
- Execute the \textbf{sRH control law} within own Voronoi cell.
  \textit{(Note that the sRH does not require the explicit knowledge of the Voronoi cell!)}
- Update own generator according to the following \textbf{gradient descent} scheme:

\[
\dot{g}_i = -k(\lambda) \frac{\partial}{\partial g_i} \left\{ \frac{\beta^2 \lambda}{(2 - \eta)AV} \sum_{j=1}^{m} \text{Area}[V_j(g)]^3 + \sum_{j=1}^{m} \int_{V_j(g)} \|q - g_j\| \, dq \right\}
\]

\textbf{Note}: the above is a \textbf{spatially decentralized} computation (in the Delaunay sense); in the general case the complexity of comm./computation is $O(1)$.

\[
\dot{g}_i = k(\lambda) \left\{ \frac{\beta^2 \lambda}{(2 - \eta)AV} \sum_{j \in N_i} l_{ij} \left( A_j^2 - A_i^2 \right) \frac{g_j - g_i}{\|g_j - g_i\|_2} + \int_{V_i(g)} \frac{q - g_i}{\|q - g_i\|_2} \, dq \right\}
\]
Multi-Vehicle DVRP Performance

- **Theorem:** the mRH policy (locally) recovers the performance of the best known centralized policies:

  - In the light-load limit $\lambda \rightarrow 0$:
    
    \[
    T_{\text{mRH}} = \frac{1}{AV} \min_g \int_Q \min_i \|q - g_i\|_2 \varphi(q) dq \\
    = \frac{1}{AV} \min_g \sum_{i=1}^{m} \int_{V_i(g)} \|q - g_i\|_2 \varphi(q) dq = \Theta \left( \frac{1}{\sqrt{m}} \right)
    \]

  - In the heavy-load limit $\lambda \rightarrow \infty$:
    
    \[
    T_{\text{mRH}} = \frac{\beta^2}{2 - \eta} \frac{A\lambda}{m^2V^2}
    \]

- **Note:** In the heavy-load case, the waiting time decreases as the square of the number of agents!
Optimal Coordination with No Communication?

System model
- Agents can sense one another’s position, but cannot communicate.
- Agents do not know the target distribution (i.e., $\varphi$).

Control law
If there is a target: Move towards target(s) for which self is the nearest agent.
If there are no outstanding target: Each agent moves to the position that minimizes the average distance to previously serviced targets.

Properties
- The above control law is (locally) optimal if $\lambda < \lambda^*(m)$, i.e., the loitering points converge to the median points of $Q$ in the light load case.
- The solution is a pure Nash equilibrium in a non-cooperative game setting.
- Same results if agents cannot sense others, and speeds are different.
- Generalizes MacQueen’s algorithm for Vector Quantization.

[Arsie and Frazzoli, ‘06]
Vehicle Routing with Differential Constraints

- **What happens if the vehicles are subject to non-integrable differential constraints on their motion?**
  - Minimum turn radius, constant speed (UAVs, Dubins’ cars)
  - Minimum turn radius, able to reverse (Reeds-Shepp’s cars)
  - Differential drive robots (e.g., tanks).
  - Bounded acceleration vehicles (e.g., helicopters, spacecraft).
  - Systems on manifolds/Lie groups (e.g., space telescopes)

- Fundamentally different problems, combining **combinatorial task specifications** with differential geometry and optimal control.

- Decompose the problem, study the asymptotic cases:
  - Heavy load: the **“Dubins Traveling Salesperson Problem.”**
  - Light load: **optimal loitering patterns.**
The Dubins Traveling Salesperson Problem

**Problem Statement**

“Find the shortest closed curve with bounded curvature through \( n \) points in the plane”

- Open problem, present in most UAV routing applications.
- Current solutions rely on uncharacterized heuristics and/or brute force.
- NP-hardness a consequence of the NP-hardness of the Euclidean TSP.

- Does the cost of the DTSP increase sublinearly with \( n \)?
- Is there a polynomial-time algorithm that returns a tour of length \( o(n) \)?
What is the quality of the solution?
ETSP vs. DTSP

• The Euclidean TSP (ETSP) is one of the prototypical “hard” combinatorial optimization problems.
  - The exact solution is extremely hard to compute.
  - Good approximations are “easy” to obtain.

• Stochastic ETSP [Beardwood et al., ’59]: let ETSP(n) be a random variable representing the minimum length of a tour through n points sampled from a uniform distribution in a $d$-dimensional set of measure 1.
  \[
  \lim_{n \to \infty} \frac{\text{ETSP}(n)}{n^{1 - 1/d}} = \beta_d, \quad \text{a.s.}
  \]

• The Dubins TSP (DTSP) is fundamentally different:
  - Non-metric problem: might not even be approximable.
  - No known reduction to a problem on a finite graph.
  - Introduced in [Enright and Frazzoli ’05], and [Savla, Frazzoli, Bullo ’05].
  - Contributions include: [Darbha ’05], [Le Ny and Feron ’05], [Itani and Dahleh ’05-’06], [Sivakumar, Darbha, Sengupta ’06].
  - Our latest results in [Savla, Frazzoli, Bullo ’06].
The Bead Tiling Algorithm 1/2

- Basic geometric construction: the “Bead”

- Properties of a bead of length \( l \):
  - A path of length \( l + o(l^2) \) always exists between the end points and an arbitrary point in the bead.
  - The “width” of the bead is \( l^2 + o(l^3) \).
  - The bead tiles the plane
The Bead Tiling Algorithm 2/2

• Tile the region of interest with beads such that: 
  \[ \text{Area}[B_\rho(l)] = \frac{\text{Area}[Q]}{2^n} \]

  Sweep the bead rows, visiting one target per non-empty bead.

• Iterate, using at the \( i \)-th phase a “meta-bead” composed of \( 2^{i-1} \) original beads.

• After \( \log n \) phases, visit the outstanding targets in any arbitrary order, e.g., with a greedy strategy.
Analysis of the BTA

**Theorem:** The cost of stochastic DTSP satisfies the following inequalities, *with high probability*

\[
\frac{3}{4} \left(3\rho WH\right)^{1/3} \leq \lim_{n \to \infty} \frac{\text{DTSP}_\rho(n)}{n^{2/3}} \leq 9.88 \sqrt[3]{\rho WH} \left(1 + \frac{7}{3} \pi \frac{\rho}{\max\{W, H\}}\right)
\]

**Outline of the proof:**
- Let \( v_i \) be the number of non-empty beads at the inception of the \( i \)-th phase.
- Show (by induction) that \( v_i \leq 2^{i-i} n \) w.h.p., for all \( i \leq i^* \leq \log_2 n \).
- Show that by the end of the \( i^*-th \) phase, almost all (i.e., \( n - O(\log n) \)) targets have been visited. The cost of visiting the \( O(\log n) \) leftovers is negligible.
- Show that the cost of the first \( i^* \) phases is a constant time the cost of the first phase, which in turn is \( O(n^{2/3}) \).
Numerical Experiment Results
The dynamic case

- The static DTSP results can be directly applied to the dynamic case, using a **queueing strategy**
  - Tile the plane with beads of length \( l = c/\lambda \).
  - Visit all beads with a non-empty intersection with \( Q \) in sequence, servicing one target per bead. Repeat.

**Theorem:**

\[
\beta^-(\rho)^3 \leq \lim_{\lambda \to +\infty} \frac{T^*}{\lambda^2} \leq c^* \beta^+(\rho)^3
\]

- Each bead can be treated as a separate queue, with arrival rate \( \lambda_B \approx \frac{l^3 \lambda}{16\rho WH} \)
- The vehicle visits each bead with a rate that is no smaller than \( \mu_B \approx \frac{l^2}{16\rho WH} \left( 1 + \frac{7}{3} \frac{\rho}{W} \right)^{-1} \)
- The system time is no greater than the system time for the M/D/1 queue.

**Notes:**

- First result showing the stability of DVRPs with Dubins vehicles.
- Stronger dependency on \( \lambda \) (quadratic) than in the Euclidean case (linear).
- Unlike the Euclidean case, such stability cannot be maintained for an adversarial target selection.
Multiple-vehicle cooperative policy design

- For an environment bounded by a $W \times H$ rectangle, the bead-tiling algorithm provides an upper bound of the form:

$$\lim_{\lambda \to \infty} \frac{T^*}{\lambda^2} \leq 9.88^3 \rho WH \left(1 + \frac{7}{3} \pi \frac{\rho}{\max\{W, H\}}\right)^3$$

- The area of the region is not the only important factor: the shape plays a major role.
- The “non-holonomy” penalty decreases as $W/\rho$ increases (for constant WH).

- If $m$ agents share the same region, the most efficient policy assigns distinct rows to each agent. The agent move roughly along parallel paths (“swarms”!).

  - $\max\{W, H\}$ is unchanged, $\min\{W, H\}$ and $\lambda$ scale down by $m$:

$$\lim_{\lambda \to \infty} \frac{T^*}{\lambda^2} \leq 9.88^3 \frac{\rho WH}{m^3} \left(1 + \frac{7}{3} \pi \frac{\rho}{\max\{W, H\}}\right)^3 = \left(\frac{\beta^+(\rho)}{m}\right)^3$$

For Dubins vehicles (and UAVs), the performance of multiple-vehicle systems increases with the cube of the number of agents.
Collision avoidance for Air Traffic

- Consider a set of $m$ aircraft, modeled as vehicles moving at unit speed along planar paths with bounded curvature:

  \[ \begin{align*}
  \dot{x}_i(t) &= \cos \theta_i(t) \\
  \dot{y}_i(t) &= \sin \theta_i(t) \\
  \dot{\theta}_i(t) &= \omega_i(t), \quad \omega_i(t) \in [-1/\rho, 1/\rho], \quad \forall t
  \end{align*} \]

- **Dubins** model, with additional constraint of constant speed.
- Each aircraft must reach a target configuration:

  \[ g_{f,i} = (x_{f,i}, y_{f,i}, \theta_{f,i}) \in SE(2) \]

- Conflict arise then the distance between two aircraft is less than a given safety distance

  \[ d_s \geq 2\rho \]
Problem formulation

• Control policy: \( \pi_i : (g_i, z_i, \text{Neigh}(g_i)) \rightarrow \omega_i \)

• It is desired that the conflict avoidance policy be
  - **Spatially decentralized**, with respect to a proximity graph.
  - **Scalable**: the amount of information processed by a single agent should be independent from the number of agents in the environment.
  - **Sensor-based**: Reliant only on position and heading, not on pilot’s intentions. (Robust and transparent implementation, communication via on-board transponders.)

• We restrict our attention to Cooperative Policies, i.e., all agents rely on the fact that other agents follow a certain set of rules.

• We would like the control policy to formally guarantee:
  - **Safety**: Nothing bad ever happens (no conflicts)
  - **Liveness**: Something good eventually happens (land at destination)
(Some, recent) related work

- Path coordination for multiple holonomic robots:
  - Prioritization-based schemes [LaValle, Hutchison ’98, Peng, Akella ‘02]
  - Pareto-optimal coordination on graphs [LaValle, Ghrist, et al. ‘04-’05]

- Path coordination for multiple non-holonomic robots with variable speed (and able to stop).
  - “Decentralized” Navigation Functions [Kyriakopulous et al. ‘03-05]

- Algorithms for air traffic control
  - SDP-, MILPS-based centralized optimization [Frazzoli et al. ‘01, Pallottino et al. ‘02]
  - Canonical paths for multiple Dubins vehicles: [Pallottino, Bicchi ‘01]
  - Decentralized optimization: [Stipanovic, Tomlin et al, ‘03]

- Main sources of inspiration for this work:
  - The multi-BUG Algorithm [Lumelski et al. ‘95]: safety is guaranteed, liveness is not.
  - The Roundabout policy [Tomlin et al. ‘98-] : safety guaranteed for n <= 3 vehicles.
Reserved Region

- Can we use Lumelski’s approach for the **multi-BUG** algorithm in the context of non-holonomic, constant-speed vehicles?

  - **Reserved region**: union of safety disks when the vehicle is performing a max. curvature turn in the CW direction.
    - Each vehicle claims exclusive ownership of its reserved region.
    - The vehicle is always able to maintain its safety region inside the reserved region.

- At the cost of some conservatism on conditions guaranteeing safety, we gain the following:
  - The motion of the reserved region can be stopped at will.
  - Motion can be resumed in any direction, provided the vehicle waits long enough to adjust the heading.
Basic strategy: straight/hold

- Let us consider a simple (non-optimal) strategy for steering vehicles in a free environment:
  - Steer the center of the reserved region towards the center of the reserved region at the final configuration.
  - “Right-turn-only” paths
Overcoming “obstacles”

• What to do when the vehicle encounters another vehicle?
• Reserved disks are not allowed to intersect.
• Each vehicle will attempt to make its reserved disk “roll” counterclockwise on a neighbor’s reserved disk, in a manner compatible with all the contact constraints.
Rolling on non-stationary disks

- If the reserved disk on which a vehicle is rolling over is itself moving, contact between the two reserved regions will be lost.
  - Note: the reserved disks are guaranteed to remain openly disjoint.
- In the case of unexpected loss of contact during a “roll” maneuver, vehicles continue the turn at maximum curvature, for a maximum of 360 degrees
  - need an internal timer state
The Generalized Roundabout Policy

- Encode the control logic as a Finite State Machine.
  - Combination of continuous dynamics and discrete logic gives rise to a **Hybrid Control System**.
  - Extend to the multiple-vehicle system in the natural way, through parallel composition.

- Is the hybrid system
  - Well-posed?
  - Safe?
  - Deadlock/livelock-free?
Well-Posedness

**Proposition:** the Generalized Roundabout Policy is well posed for all initial conditions such that reserved disks are openly disjoint.

**Proof outline:**
- In each mode the continuous dynamics are Lipschitz.
- The maximum number of simultaneous discrete switches is three. Feasible paths on the Hybrid Automaton FSM do not allow loops.
- There is no accumulation point of switching times: the time to the next (non-simultaneous) event is bounded away from zero---assuming a finite number of vehicles.
Safety

- Straightforward to show the following safety invariance property: *if the reserved disks are openly disjoint at the initial time, they will always be openly disjoint.*
  - Safety is guaranteed for arbitrary numbers of aircraft.
  - The size of the input to each vehicle is uniformly bounded, for any number of aircraft. More specifically, each vehicle needs to know the current position and heading of at most six other vehicles.
  - The maximum distance of a neighbor is $4\rho + d_s$. 
Liveness analysis

• **Proposition:** If no disk of radius $d_s + 2\rho$ contains 3 or more target centers, the Gen. Roundabout Policy is **deadlock-free**

• **Proof outline:** Consider the set of reserved region centers. Agents whose reserved region center lies on the convex hull of such set will switch to a different mode within finite time $t_s$:
  - Agents in the straight mode: $t_s \leq$ distance from target center.
  - Agents in the roll2 mode: $t_s \leq 2\pi$.
  - Agents in the hold mode: n.a.
  - Agents in the roll mode: $t_s \leq 4\pi$
    
    *if the hypothesis is verified.*
Liveness analysis

- **Conjecture:** A sufficient condition for liveness is that the centers of target reserved regions satisfy the additional following *sparsity criterion*:

\[ \text{No disk of radius } d_s + 2 \rho \cot(\pi/m) \text{ contains } m \text{ target centers or more}. \]

- **Formal proof** available for the **two-** and **three-vehicle** case;
  - Proof is laborious and non-interesting: step through all possible cases.
  - work in progress on the general case.
Liveness counterexample

- Symmetric arrangement of initial configurations, containing all targets.
- Does the set of initial conditions leading to livelock have zero measure?
  - Monte-Carlo simulations have not yet produced an instance leading to livelock.
  - Probabilistic verification: 99% confidence the sparsity condition is correct within a 1% approximation.
Simulation example

- Initial conditions: 37-airplane cluster
Simulation example

- Random initial conditions, 50 aircraft
Experimental demo

- Low-cost, limited-space indoor mobile robotics testbed.
  - Approx. $30 per car.
  - Ceiling mounted video cameras provide sensor feedback.
Conclusions

- The **algorithmic analysis and design of mobile robotic networks**
  - requires new tools combining **systems & control theory, combinatorial optimization, differential geometry, distributed computing, probability and stochastic systems**, 
  - offers new insight into the design and performance evaluation of autonomous systems---and possibly beyond (social networks, bio systems).

- **Lessons learned:** Rediscover the “original motivation” for **Hybrid Systems**
  - **Discrete/combinatorial task specification + dynamical systems:** control algorithms will likely maintain such a dual nature.
  - **Do not lose sight of the original control design problem:** algorithmic analysis tools can provide effective guidance on the controller structure. Correct modeling and problem formulation are key steps in this.
  - Focus on **effective control synthesis techniques:** design control systems in such a way that the analysis of the closed-loop behavior is easy. *(As opposed to imposing and studying arbitrary feedback structures.)*

[More info available at [http://ares.lids.mit.edu](http://ares.lids.mit.edu)]