Constraint Reasoning and Motion Planning in the SAUNA Project

Federico Pecora, Marcello Cirillo, Dimitar Dimitrov*

<name>.<surname>@oru.se

Center for Applied Autonomous Sensor Systems (AASS)
Örebro University, SWEDEN

* *Work funded by project SAUNA*
(Safe Autonomous Navigation)
Coordinator: Dimiter Driankov

AGVs for Industrial Automation

- Autonomous Ground Vehicles (AGVs) becoming paramount to industrial automation

mining (e.g., Atlas-Copco)

construction (e.g., Volvo)

logistics (e.g., Kollmorgen)

- Several key processes are still ad-hoc and labor-intensive
  - AGV paths often pre-defined and hand-crafted
  - crude planning/allocation/scheduling heuristics
  - conflicts “avoided off-line” rather than resolved on-line
  - lack of **flexibility** and **reconfigurability**

What If. . .

- Site operators could **post high-level requirements**
  - new tasks, vehicles
    e.g., "pick up new incoming loads"
  - spatial constraints
    e.g., "zone A is now off-limits"
  - temporal constraints
    e.g., "complete mission of vehicle A within 10 minutes"
- Vehicles could **adapt** their current and scheduled trajectories accordingly
  - vehicles coordinate automatically in response to posted requirements
- **Performace of vehicles** could be automatically taken into account by **other vehicles**

The Core Issue of Coordination

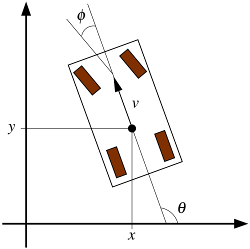
- Coordination is necessary to ensure absence of **collisions** and **deadlocks**
- **Current practice:** compute trajectories for all vehicles **before** coordination occurs
  - collisions and deadlocks are dealt with locally
  - overall fleet requirements cannot be guaranteed
- **Our view:** all decisions regarding vehicle trajectories can be seen as **constraints on trajectories**
  - some constraints **known in advance**, some need to be **inferred**
- **Least commitment:** impose increasigly tight constraints on trajectory, but **do not commit** to specific solution until **execution**

Outline

1. Trajectory Envelopes
2. Conflict Resolution as Refinement of Trajectory Envelopes
3. Control with Trajectory Envelopes
4. Evaluation
5. Conclusions

Non-Holonomic Vehicles

- Vehicles are non-holonomic $\Rightarrow$ no sideways motion
- "Car-like" vehicles



Kinematic Model

$$\dot{q} = f(q,v) = (v\cos(\theta), v\sin(\theta), \frac{v}{l}\tan(\phi), \dot{\theta})$$

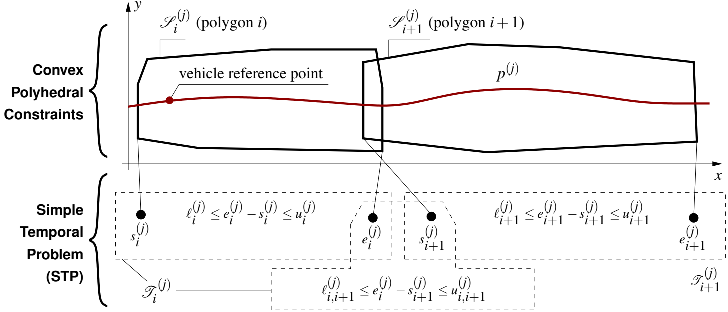
$q = (x,y,\theta,\phi) \in \mathbb{R}^4$ (state vector)

$v = (v,\dot{\theta}) \in \mathbb{R}^2$ (control vector)

- **Path:** $p : [0,1] \rightarrow \mathbb{R}^2$ (parametrized using arc length)
- **Trajectory:** $p(\sigma(t))$ (σ = time history along path)

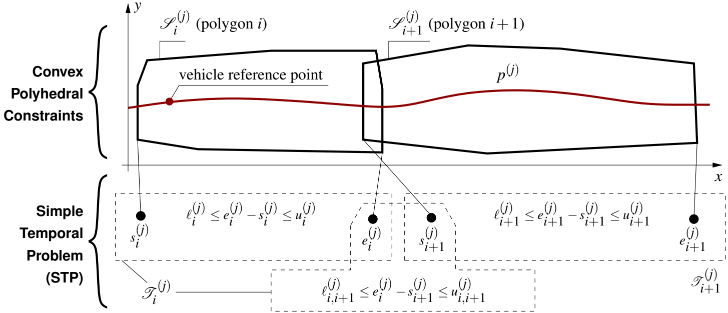
Trajectories and Trajectory Envelopes (I)

- **Trajectory envelope: spatial** requirements on p and **temporal** requirements on $\sigma(t)$
  - **temporal envelope**, **spatial envelope**



Trajectories and Trajectory Envelopes (II)

- Trajectory $p(\sigma(t))$ is **admissible** if
  - it can be obtained from the evolution of the **kinematic model**
  - p / $\sigma(t)$ lies within the **spatial** / **temporal** envelope

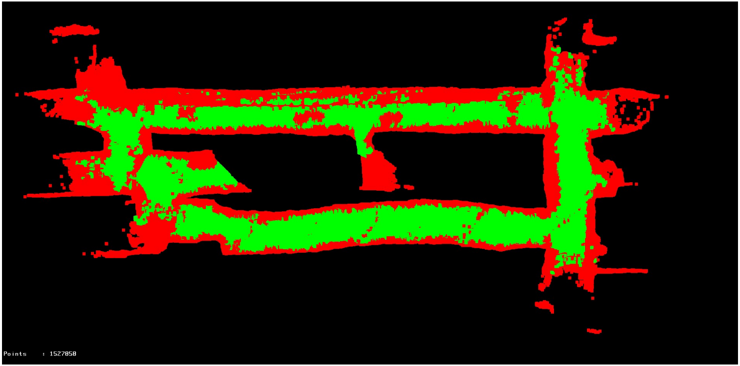


Explicit Requirements as Constraints

- Avoiding **fixed obstacles** = imposing **spatial constraits** on trajectory envelopes
  - drivable area encoded as spatial constraints
  - any additional spatial requirements can be posted as well
- High-level **temporal requirements** = imposing **temporal constraints** on trajectory envelopes
  - deadlines, durations, and any other temporal requirements can be posted directly on trajectory envelopes
- We can constrain **admissible trajectories** to account for all high-level spatial and temporal requirements
  - however, **collisions** between vehicles and **deadlocks** are still possible. . .

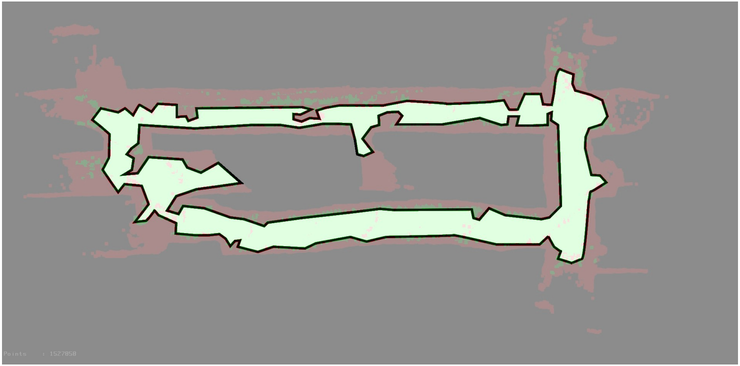
Computing Spatial Envelopes (I)

- Paths computed over drivable area using ARA^*
  - anytime path planning algorithm [Likhachev et al., 2003]
  - uses kinematically feasible motion primitives



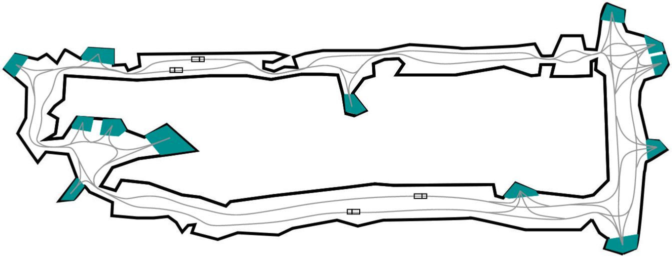
Computing Spatial Envelopes (I)

- Paths computed over drivable area using ARA^*
  - anytime path planning algorithm [Likhachev et al., 2003]
  - uses kinematically feasible motion primitives



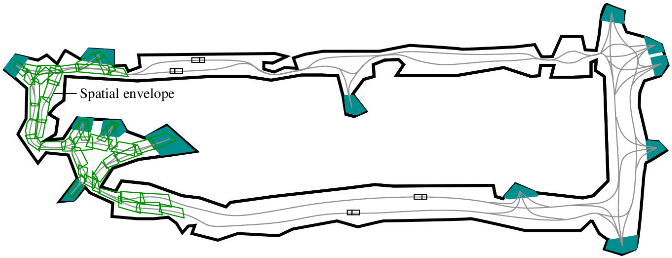
Computing Spatial Envelopes (I)

- Paths computed over drivable area using ARA^*
  - anytime path planning algorithm [Likhachev et al., 2003]
  - uses kinematically feasible motion primitives



Computing Spatial Envelopes (II)

- Spatial envelopes computed over obtained paths
  - sample paths with given $\Delta\sigma \propto (1/\text{curvature of the path})$
  - calculate polyhedron $\mathscr{S}_i^{(j)}$ enclosing the sampled point



Computing Temporal Envelopes

- Impose temporal bounds on **spatial envelope traversal**

$$\ell_i^{(j)} \leq e_i^{(j)} - s_i^{(j)} \leq u_i^{(j)}$$
$$\ell_{i,i+1}^{(j)} \leq e_i^{(j)} - s_{i+1}^{(j)} \leq u_{i,i+1}^{(j)}$$

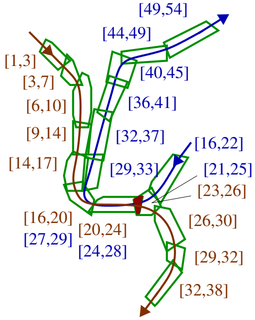
- Bounds computed with **slowest** and **fastest** speed traversal of reference path

$$\ell_i = e_i^{\text{fast}} - s_i^{\text{slow}}, \quad u_i = e_i^{\text{slow}} - s_i^{\text{fast}},$$
$$\ell_{i,i+1} = e_i^{\text{fast}} - s_{i+1}^{\text{slow}}, \quad u_{i,i+1} = e_i^{\text{slow}} - s_{i+1}^{\text{fast}}$$

Outline

Spatio-Temporal Conflicts

- Polygons that overlap in **time** and **space** constitute a **conflict**
- **Eliminating the temporal overlap** resolves a conflict



- **Spatio-temporal** overlap:

$$\mathscr{S}_i^{(j)} \cap \mathscr{S}_k^{(m)} \neq \emptyset \wedge$$
$$\left[s_i^{(j)}, e_i^{(j)}\right] \cap \left[s_k^{(m)}, e_k^{(m)}\right] \neq \emptyset$$

- Semantics of **temporal** overlap:

$$\left[s_i^{(j)}, e_i^{(j)}\right] \cap \left[s_k^{(m)}, e_k^{(m)}\right] \neq \emptyset \text{ if}$$
$$\max\left(\underline{s}_i^{(j)}, \underline{s}_k^{(m)}\right) \leq \min\left(\underline{e}_i^{(j)}, \underline{e}_k^{(m)}\right)$$

Conflict Resolution as a Meta-CSP (I)

- Spatio-temporal conflicts solved by **posting temporal constraints** [Cesta et al., 2002]
  - remove temporal overlap between polygons
- Search for resolving constraints **cast as a CSP**
  - **variables:** pairs of spatio-temporally overalpping polygons
  - **values:** temporal constraints that eleminate temporal overlap
- Collisions are avoided by **refining the trajectory envelopes**
  - adding temporal constraints to eliminate temporal overlap
- **Note:** temporal envelope refinement can be alternated with **spatial envelope refinement**
  - adding spatial constraints to eliminate spatial overlap

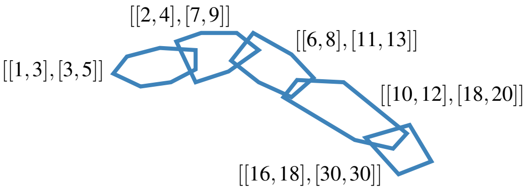
Conflict Resolution as a Meta-CSP (II)

- Search for resolving temporal constraints performed using standard backtracking search algorithm
- **Variable ordering heuristic** based on **spatial** features
  - prefer pairs of polygons that are **spatially closer** to other conflicting pairs
- **Value ordering heuristic** based on **temporal** features
  - prefer orderings that least impact overall **temporal flexibility**
- Conflict resolution also eliminates **deadlocks**
  - backtracking search breaks cycles on resource usage (resources = intersections of temporally overlapping polygons)

Outline

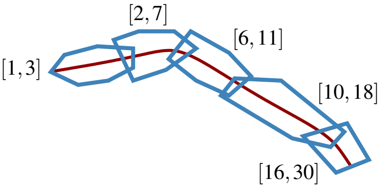
From Trajectory Envelopes to Control Actions (I)

- Conflict resolution **refines the temporal envelope** to obtain conflict- and deadlock-free trajectory envelopes
- We now have available **many alternative trajectories** that can be sampled from these envelopes
  - spatial constraints designate the polygons in which vehicles should be
  - temporal constraints determine flexible bounds on each polygon



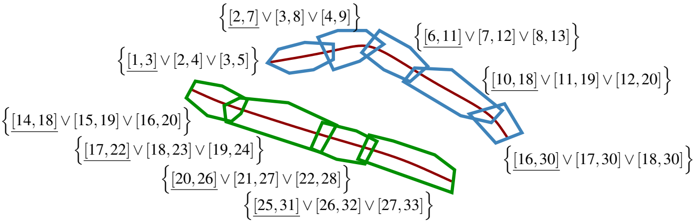
From Trajectory Envelopes to Control Actions (II)

- Each vehicle has a **tracking controller**
  - computes **kinematically feasible** control actions for vehicle
  - minimizing divergence from a **reference trajectory**
  - while accounting for **spatial constraints**
  - based on efficient QP solver [Dimitrov et al., 2011]
- **Controller input** = **nominal path** + **fixed temporal bounds** + **spatial envelope**



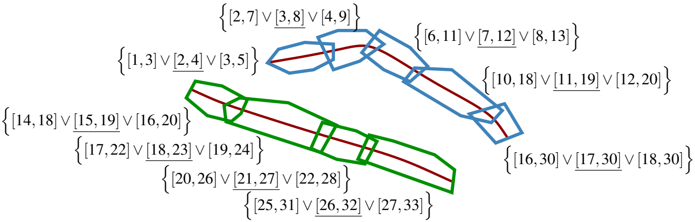
Sampling Trajectory Envelopes (I)

- Due to efficient formulation, controller can compute control actions for **several alternative** reference trajectories
  - **choose** “best” one to follow according to **trakcing performance**
- Alternative bounds can be computed in **polynomial time**
  - controllers choose current alternative **consistently** w/ other vehicles (executed temporal profiles must be solution of the STP)



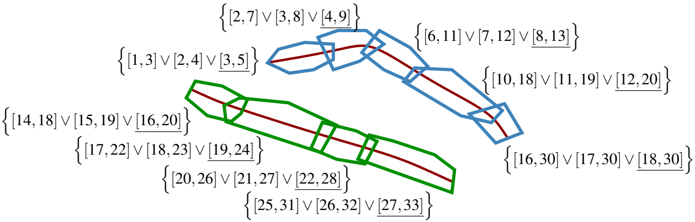
Sampling Trajectory Envelopes (I)

- Due to efficient formulation, controller can compute control actions for **several alternative** reference trajectories
  - **choose** “best” one to follow according to **trakcing performance**
- Alternative bounds can be computed in **polynomial time**
  - controllers choose current alternative **consistently** w/ other vehicles (executed temporal profiles must be solution of the STP)



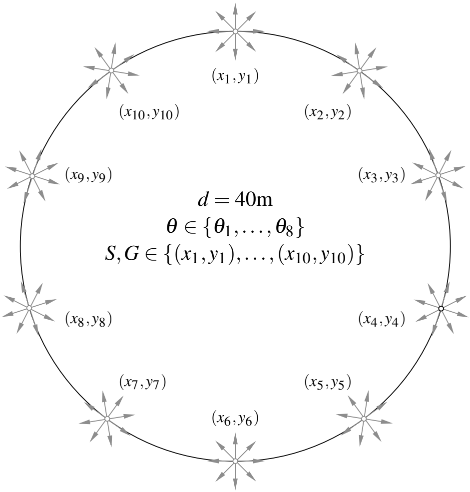
Sampling Trajectory Envelopes (I)

- Due to efficient formulation, controller can compute control actions for **several alternative** reference trajectories
  - **choose** "best" one to follow according to **trakcing performance**
- Alternative bounds can be computed in **polynomial time**
  - controllers choose current alternative **consistently** w/ other vehicles (executed temporal profiles must be solution of the STP)



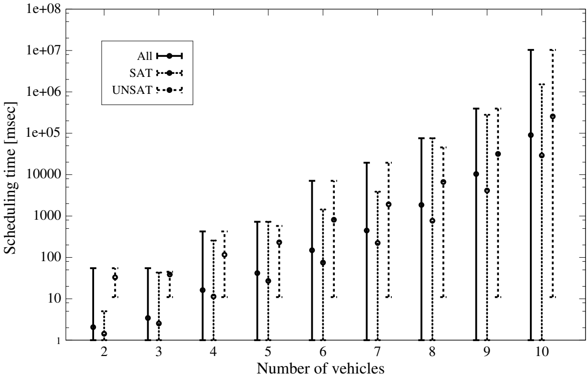
Outline

Quantitative Evaluation on Artificial Problems



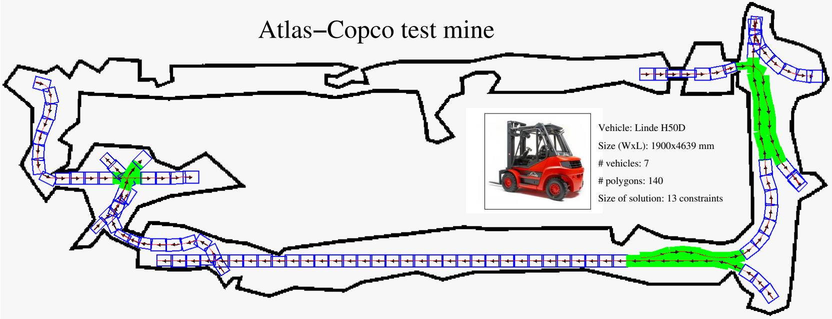
- 900 problems
- 9 test sets (2–10 vehicles)
- randomly generated paths
- 100 trials per test set
- Note: artificially difficult problems

Quantitative Evaluation on Artificial Problems



- Average conflict resolution time < 1 sec with up to 8 vehicles
- Maximum time to calculate an alternative temporal profile < 50 msec

Realistic Test Run

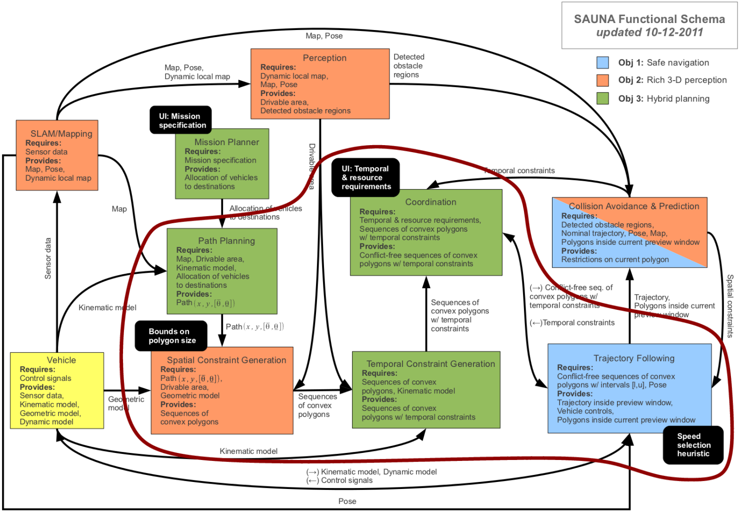


- 7 Linde vehicles, 140 polyhedra, pre-assigned start/destination poses
- Path planning: 5 sec / Conflict resolution: 34 sec (13 temporal constraints)
- Time to calculate alternative temporal profiles: 250 msec

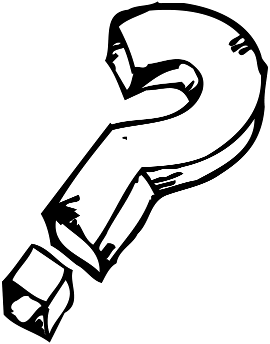
Conclusions

- Fleet management typically consists of **several processes**
  - all processes impose requirements on trajectories
- **Constraint-based formulation** provides processes with a **uniform model**
  - perception, planning, scheduling, control all contribute to refining trajectory envelopes
- Not all proccesses are necessarily **automated**
  - human operators, a pre-existing dispatching strategy, . . .
  - this is important for industrial application
- Modular approach maximizes the ability to **react at different levels**
  - modules impose increasingly low-level constraints on overall solution
  - commitment to specific trajectories done **as late as possible**

SAUNA Functional View



Thank You!



References

Cesta, A., Oddi, A., and Smith, S. F. (2002).
A constraint-based method for project scheduling with time windows.
Journal of Heuristics, 8(1):109–136.

Dimitrov, D., Sherikov, A., and Wieber, P.-B. (2011).
A sparse model predictive control formulation for walking motion generation.
In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2292–2299.

Likhachev, M., Gordon, G., and Thrun, S. (2003).
ARA*: Anytime A* with provable bounds on sub-optimality.
Advances in Neural Information Processing Systems, 16.