

A Decentralized Framework to Support UAS Merging and Spacing Operations in Urban Canyons

International Conference on Unmanned Aircraft Systems, Athens, Greece

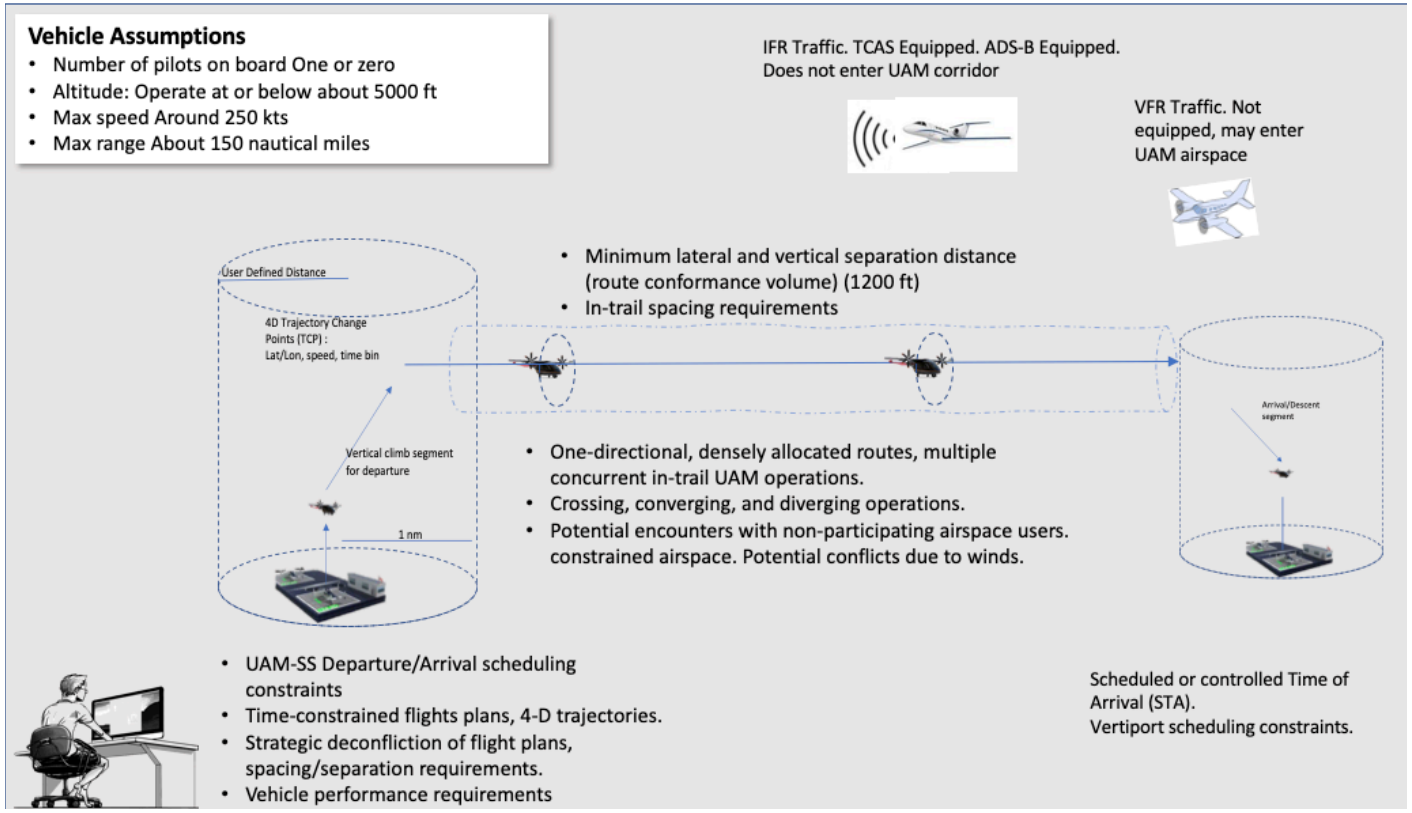
Swee Balachandran
National Institute of Aerospace,
Hampton, VA

Christopher Manderino
University of Pittsburgh,
Pittsburgh, Pennsylvania

César Muñoz, María Consiglio
NASA Langley Research Center,
Hampton, VA

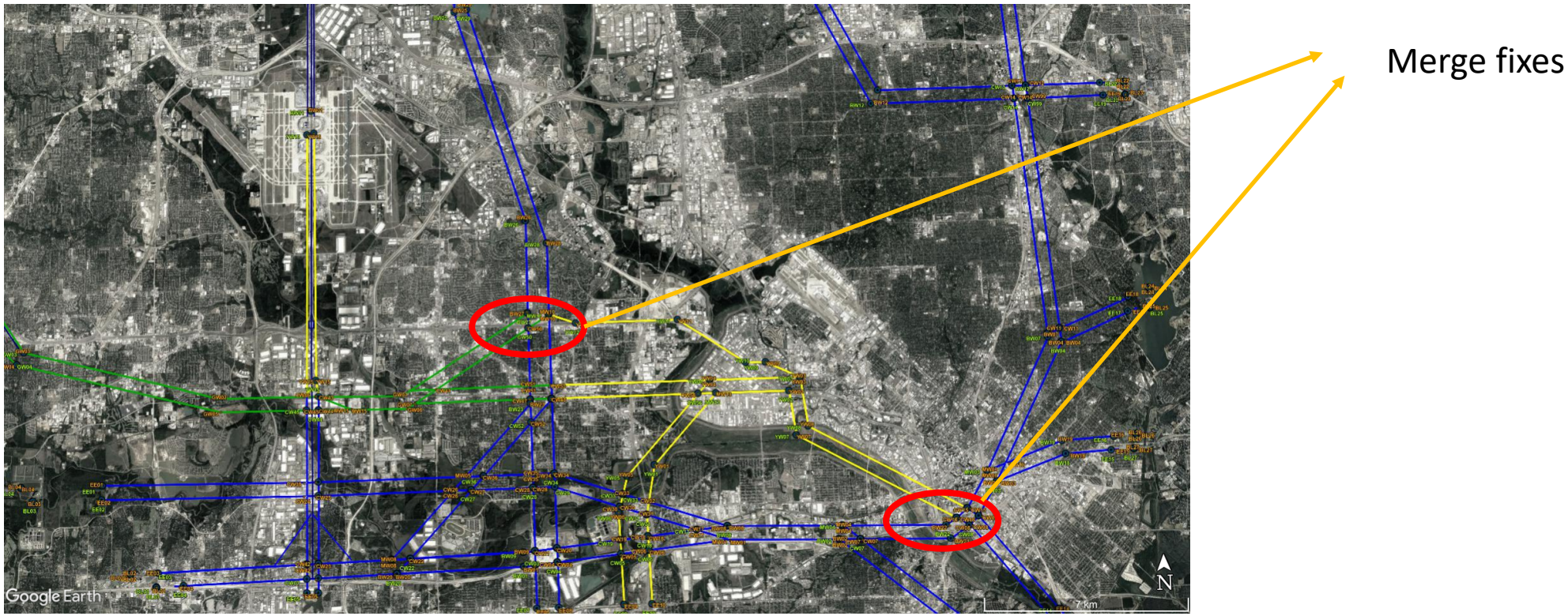
Motivation

- Urban Air Mobility (UAM) envisions the utilization of airspace below 5000 ft in an urban environment for air taxi operations using Vertical Takeoff and Landing (VTOL) type aircraft.
- Vertiports connected by a network of routes (air corridors).



UAM Airspace

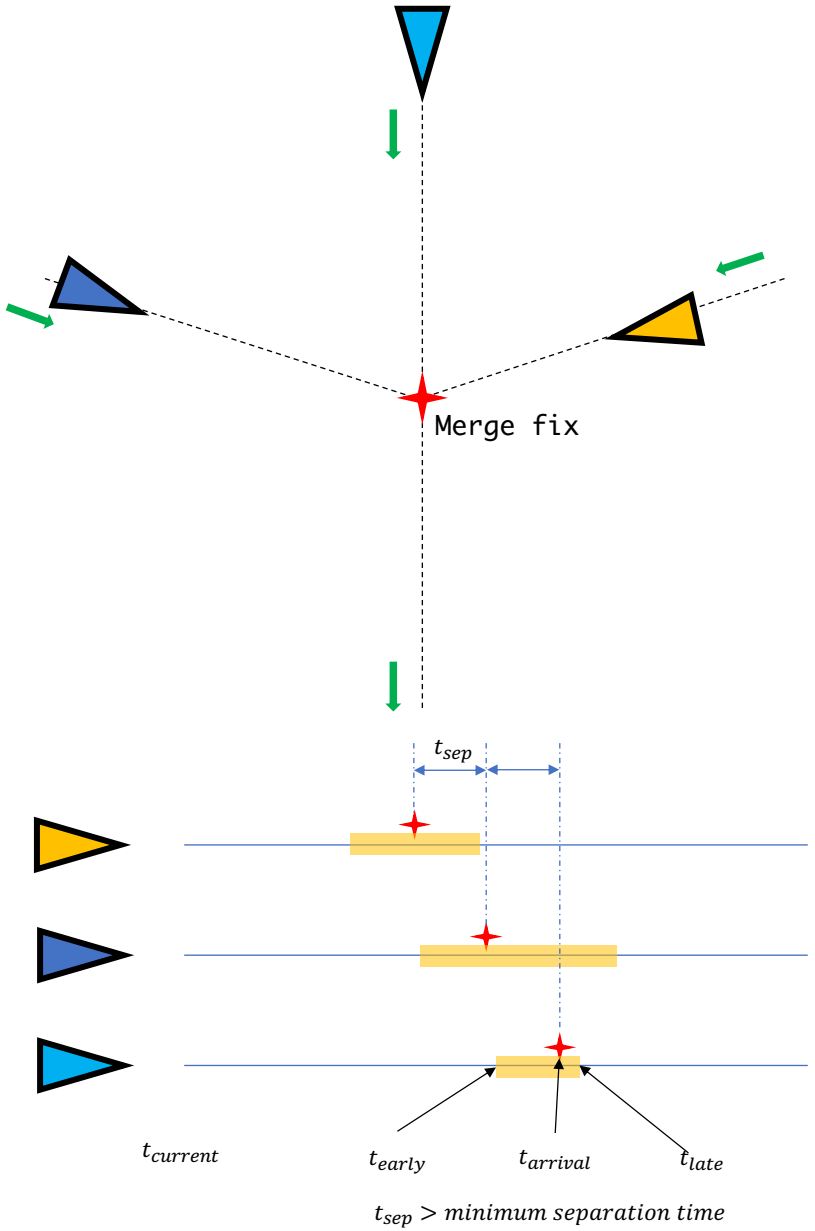
- Relying on Air Traffic Controllers (ATC) to ensure coordination among UAM vehicles may not be feasible amidst busy terminal airspace operations.
- Frequent merging/crossing operations.
- Use of air corridors requires sufficient spacing between vehicles.



UAM airspace corridors around Dallas Fort Worth and Dallas Love Field Airspace

Merging Operations

- Vehicles pass through a common merge fix while maintaining sufficient separation with other vehicles.
- Each merge fix is a resource.
- Only a single vehicle can consume the resource in each time interval.
- Each vehicle is constrained by its earliest and latest arrival times at the intersection.

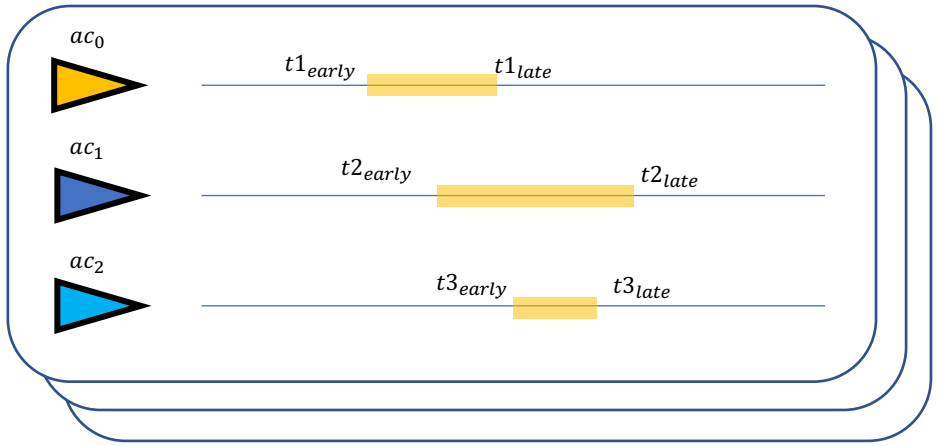


Solution Approach

- Constraint Satisfaction Problem (CSP): Match users (vehicles) to resources (merge fixes) while satisfying constraints.
- Useful techniques to compute solution:
  - Constraint Solvers: SMT Solvers
  - Scheduling algorithm
- Challenges:
  - No reliance on centralized entity to schedule usage of a merge fix.
  - Scheduling is done onboard each vehicle.
  - Achieve distributed consensus.
  - Vehicle to Vehicle communication network.

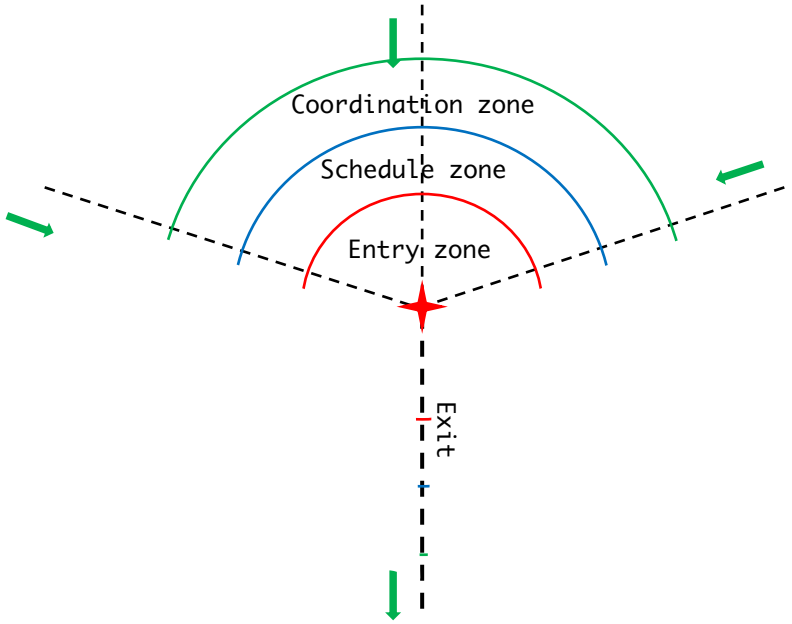
Distributed Consensus

- Vehicles exchange arrival time information among each other.
- Use of a distributed consensus algorithm to exchange information among nodes ensure each vehicle receives consistent information.
- Distributed consensus algorithms:
  - Paxos, Multi-paxos, Raft
- Inspired by Raft consensus:
  - Leader election.
  - Leader synchronizes data across other follower nodes via a heartbeat mechanism.
- Each vehicle computes a schedule based on information on which consensus has been established.



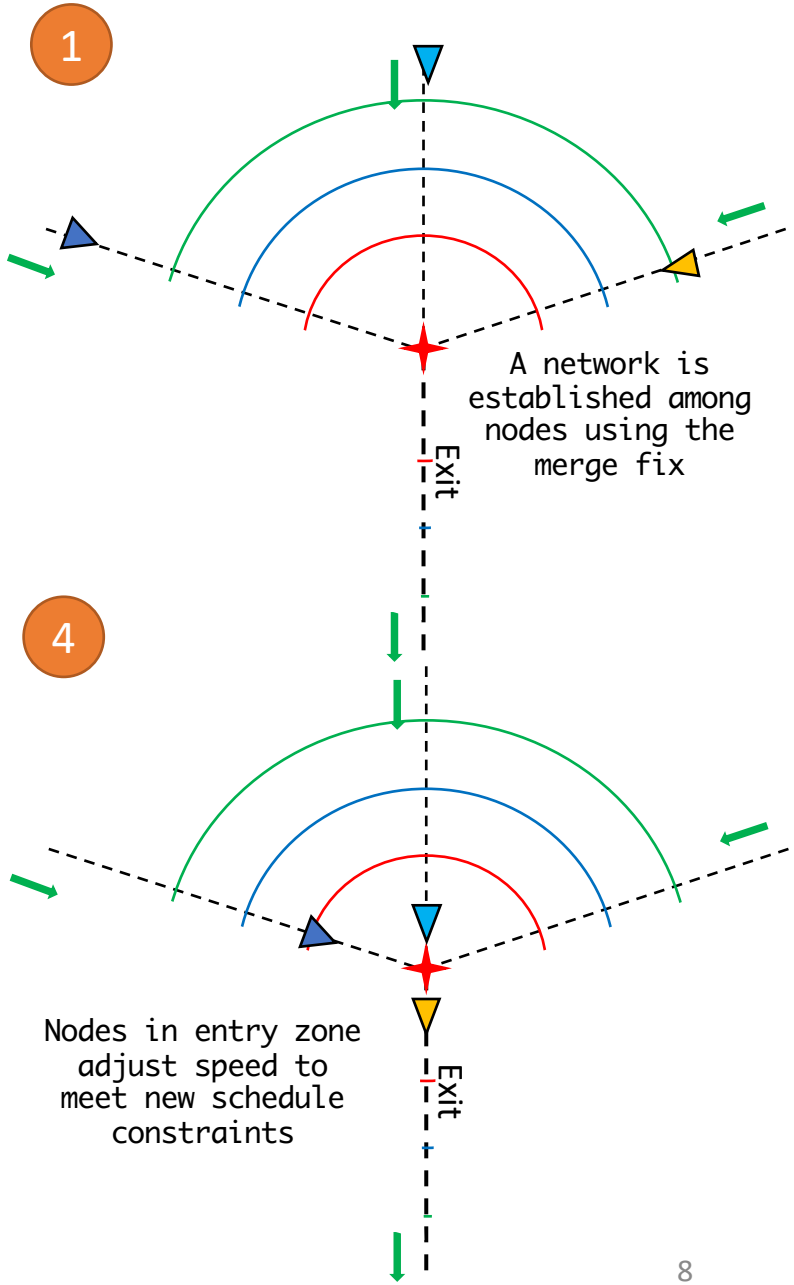
Merge Fix Geometry

- Synchronize *schedule computations* and *execution*
- Coordination zone
  - Network established among nodes approaching the network
- Scheduling zone
  - Start schedule computation
- Entry zone
  - Adjust speed to meet new schedule requirements
- Vehicles cannot change schedules after entering entry zones



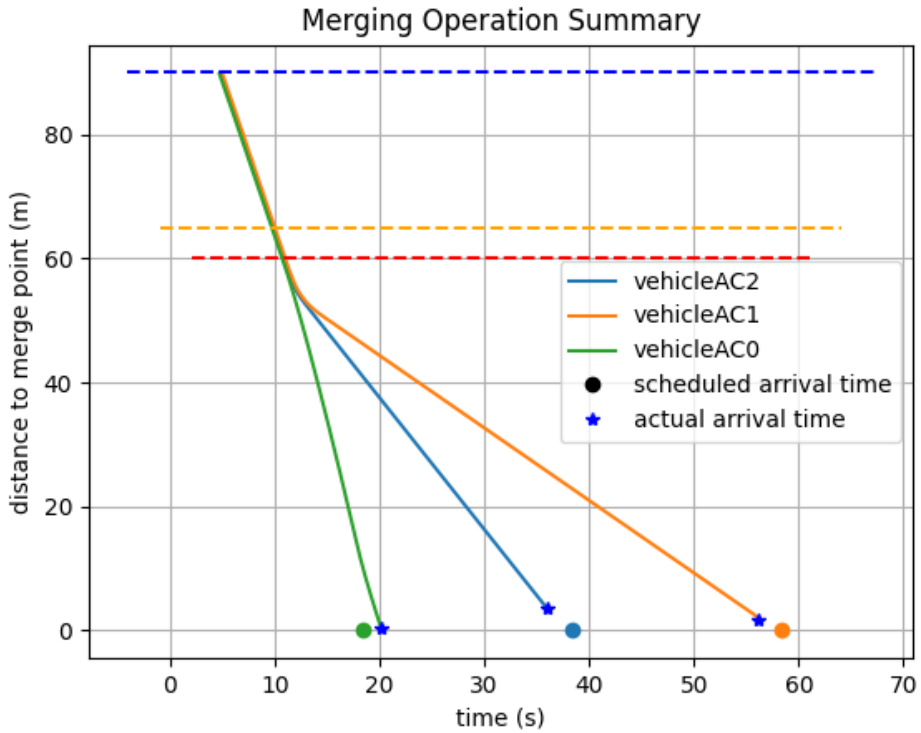
Summary: Merging Procedure

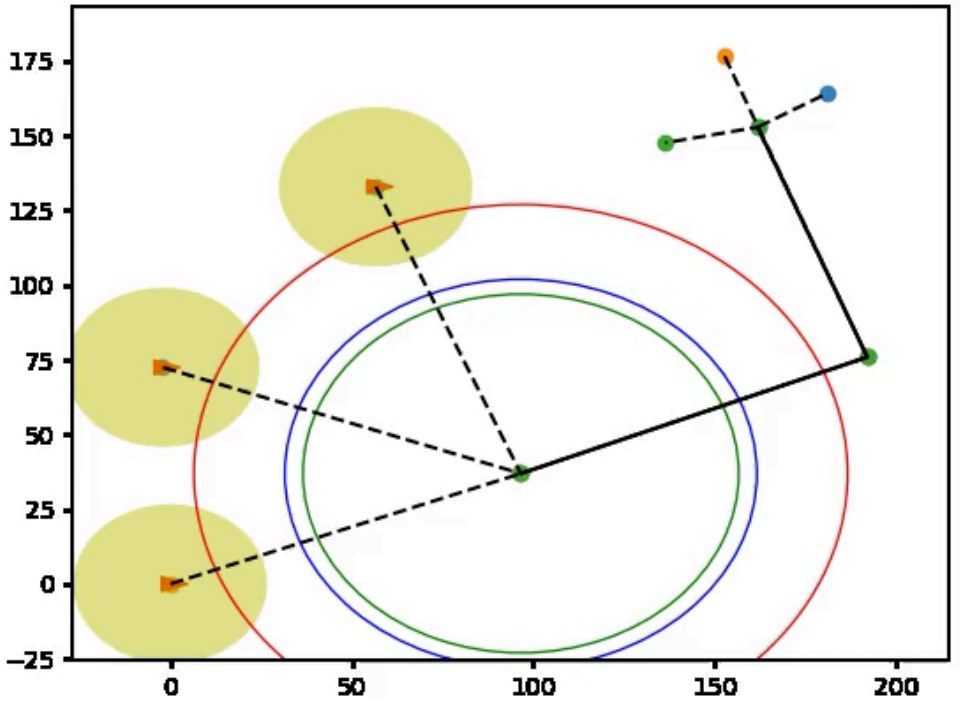
- Step 1: Identify vehicles (nodes) participating in the merging operation.
- Step 2: Exchange arrival time information among participating nodes.
- Step 3: Compute schedule when appropriate
- Step 4: Make changes to trajectory to achieve the computed schedule.



Results

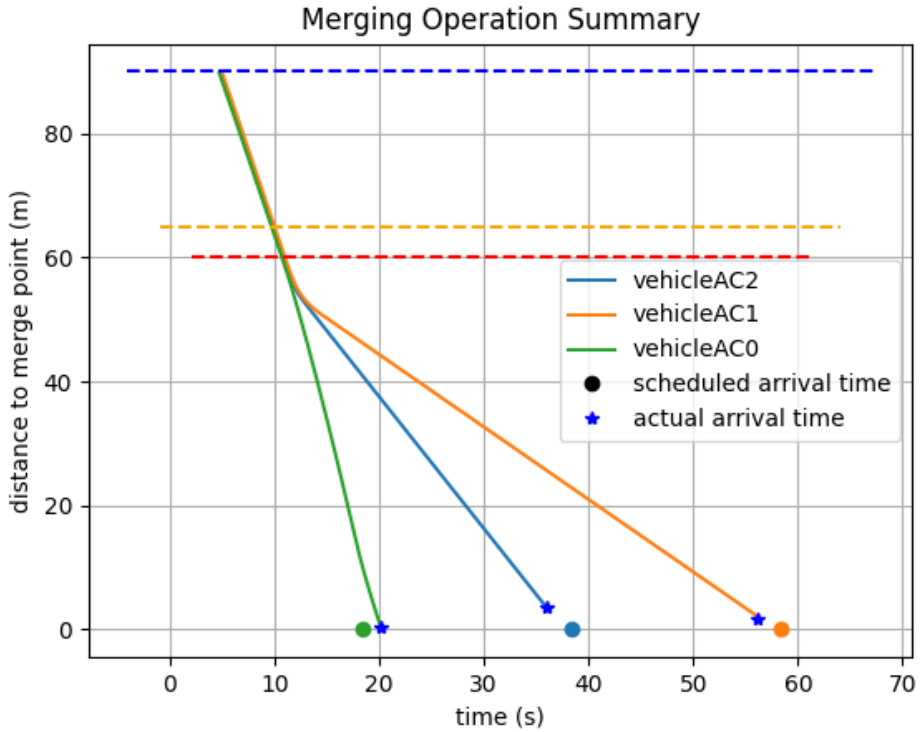
- Algorithm implemented in ICAROUS
- Batch simulation framework
- Single merge fix scenario
- 3 vehicles (entering the merging zone at 5 m/s)
- Coordination, Schedule, Entry zones: 90, 65, 60 m

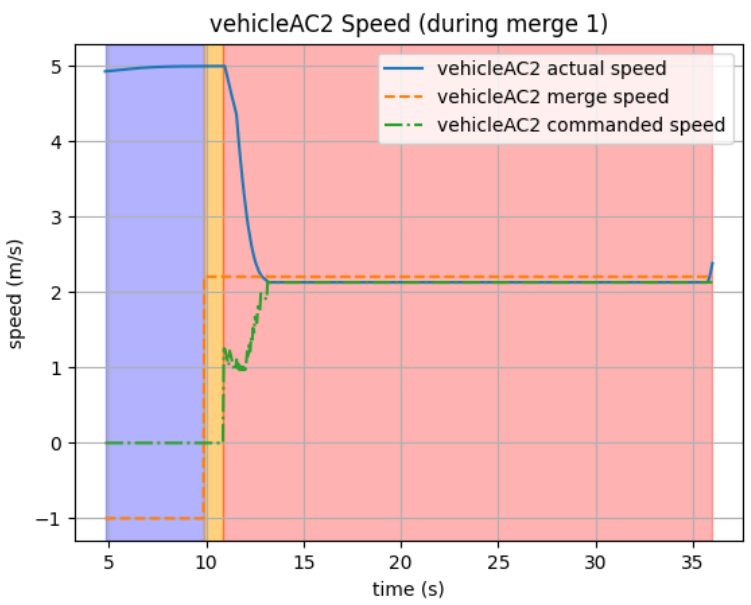


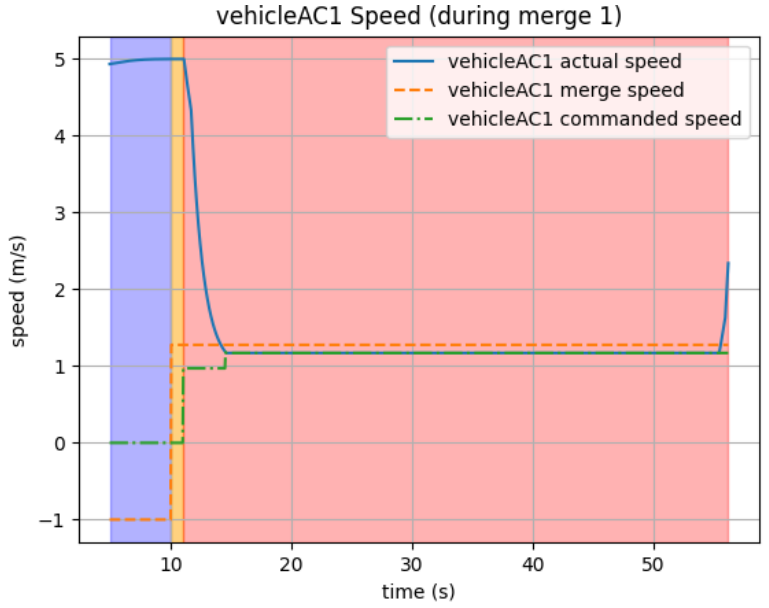


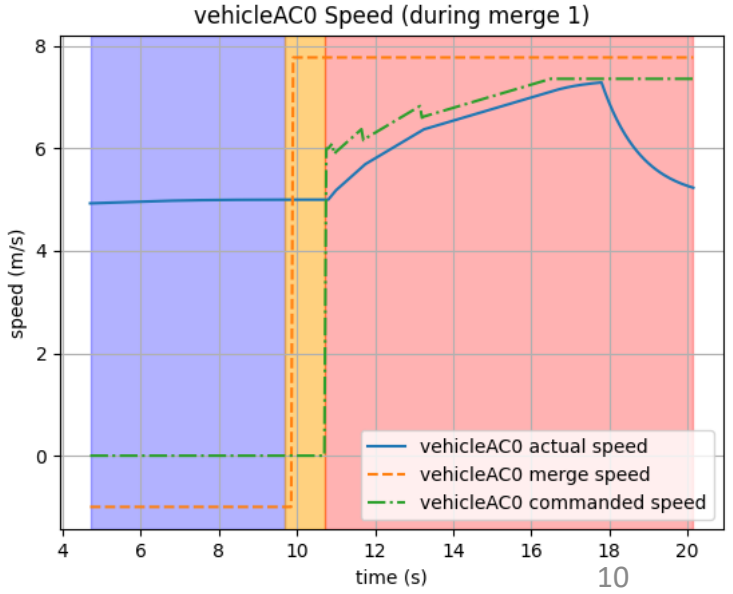
Results

- Algorithm implemented in ICAROUS
- Batch simulation framework
- Single merge fix scenario
- 3 vehicles (entering the merging zone at 5 m/s)
- Coordination, Schedule, Entry zones: 90, 65, 60 m



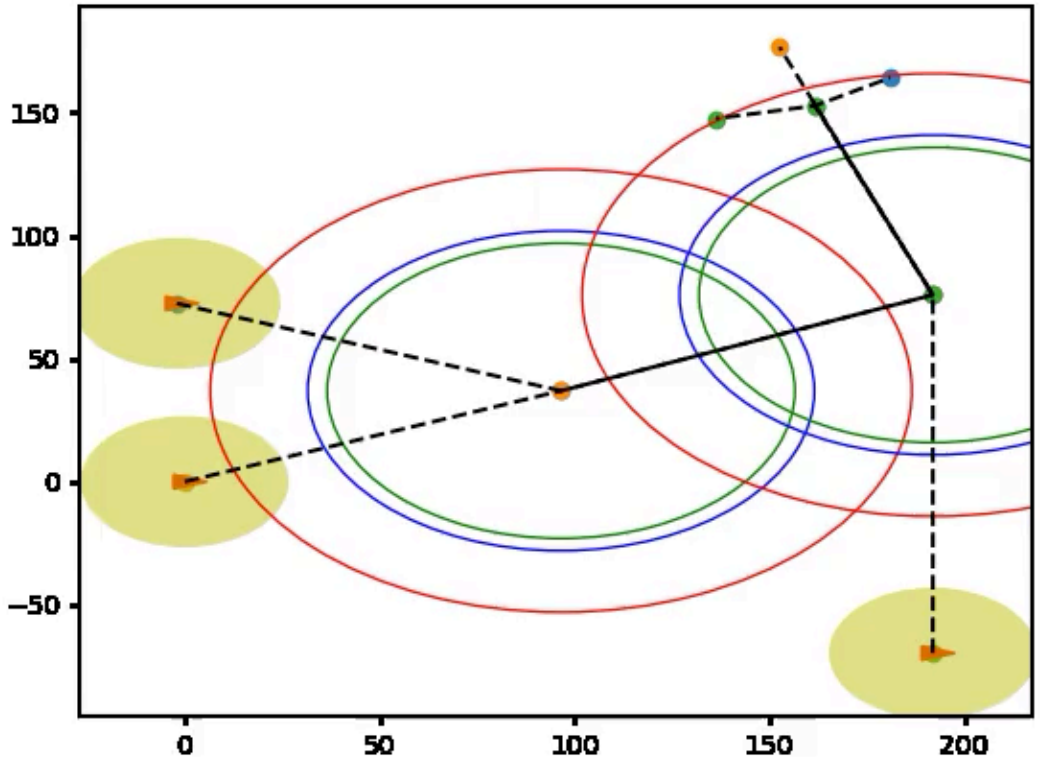






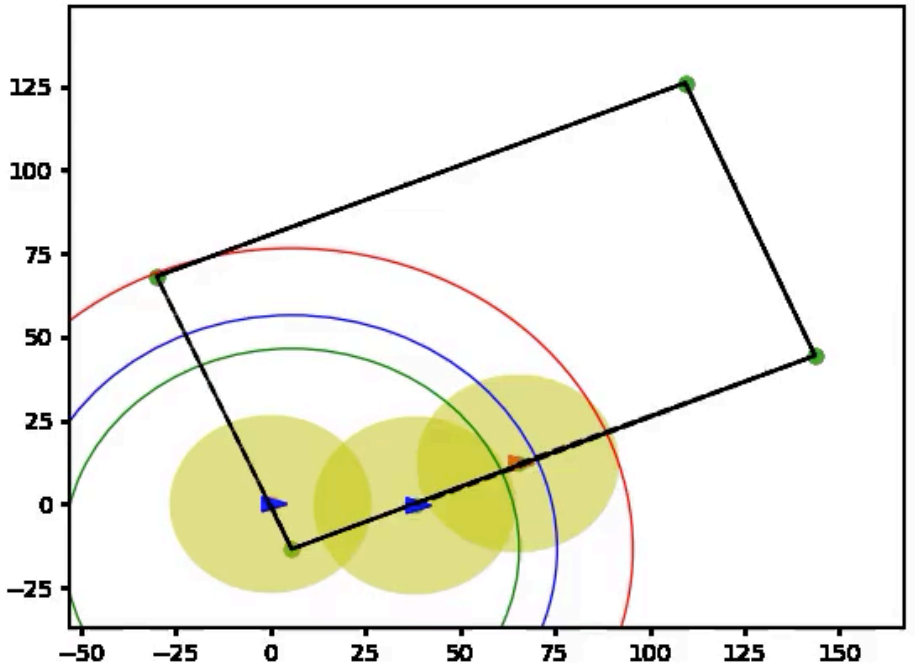
Multiple Merge Fixes

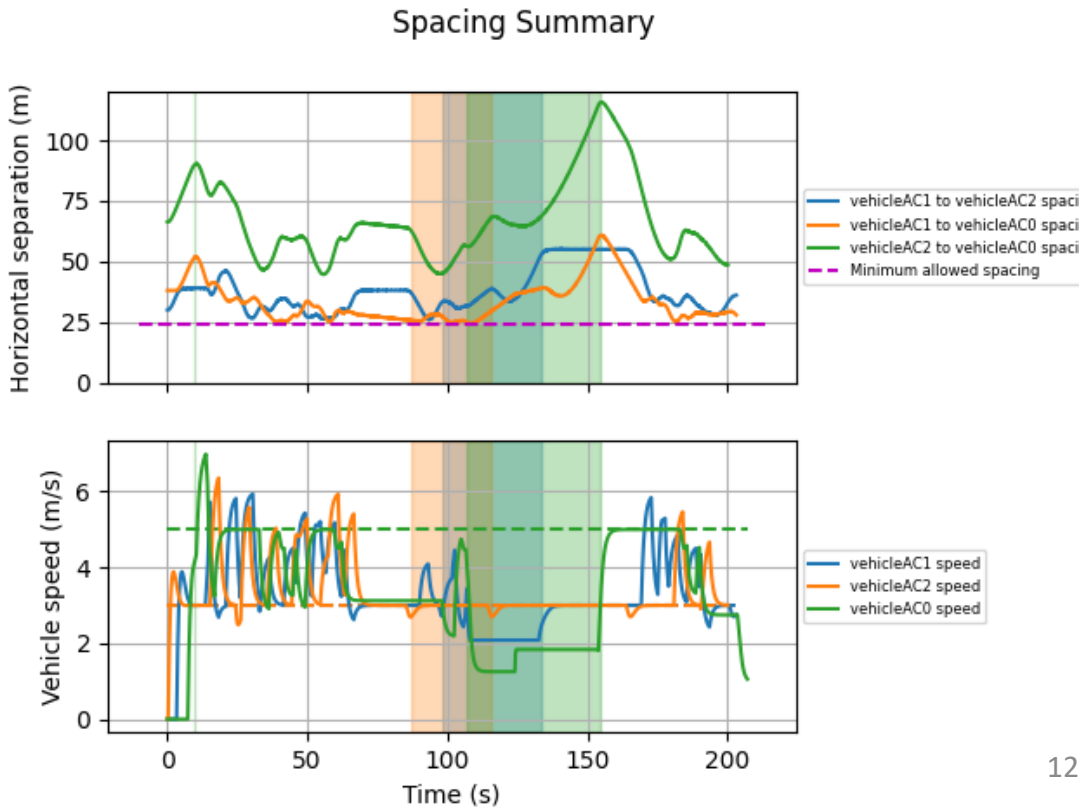
- Proposed approach deals with multiple intersections.
- Distributed coordination take places on a per intersection basis
- Vehicles coordinate with other vehicles utilizing the same intersection
- Vehicles seamlessly transition between merge fixes.



Merging and Spacing

- Perform spacing outside the merge zone.
- Make speed adjustments to maintain suitable separation from other vehicles.
- Speed adjustments are computed by the DAIDALUS library (SC-228 DAA reference implementation)
- Passive spacing.





Flight Test Evaluations

- Merging algorithm implemented in ICAROUS.
- Two real vehicles (DJI s1000, ArduCopter autopilot)
- One simulated vehicle (ArduCopter SITL)
- Multiple scenarios
  - Varying initial conditions
  - Varying entry speeds



Flight Test Results of a Distributed Merging Algorithm for Autonomous UAS Operations

Andrew Peters
National Institute of Aerospace
Hampton, VA, USA
andrew.peters@nianet.org

Swee Balachandran
National Institute of Aerospace
Hampton, VA, USA
swee.balachandran@nianet.org

Brendan Duffy
National Institute of Aerospace
Hampton, VA, USA
brendan.dufy@nianet.org

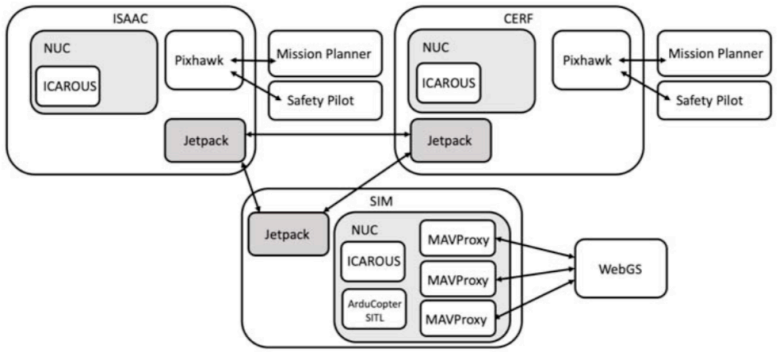
Kyle Smalling
National Institute of Aerospace
Hampton, VA, USA
kyle.smalling@nianet.org

María Consiglio
NASA Langley Research Center
Hampton, VA, USA
maria.c.consiglio@nasa.gov

César Muñoz
NASA Langley Research Center
Hampton, VA, USA
cesar.a.munoz@nasa.gov

***Abstract* - This paper reports the flight test results of an on-demand, distributed, consensus based merging algorithm for autonomous multi-agent coordination used to regulate flow of air traffic through a common intersection or merge fix in a given airspace. Distributed merging is enabled by vehicle-to-vehicle (V2V) communication technology, a distributed consensus algorithm and a scheduling algorithm to coordinate the arrival**

initial capabilities of a merging application developed to enable on-demand deconfliction of intersecting traffic flows at merge fixes. The merging application was integrated into the ICAROUS software suite and Vehicle to Vehicle (V2V) communication technology enabled vehicles to exchange the information required to achieve arrival time deconfliction. A distributed consensus and a scheduling algorithm work in



To appear in Digital Avionics Systems Conference (DASC) 2020

Formal Verification

- Algorithm specified in TLA+ and the Prototype Verification System (PVS).
- TLA+ supports usage of Linear Temporal Logic (LTL), e.g.,

$$\forall i \in Nodes: initialCondition(i) \wedge \Box(NextCondition(i))$$

- TLA+ model checking tool was also used to identify counter example traces of inconsistencies in specification.
- Key properties are verified using PVS's interactive theorem prover.
- Assumptions are made regarding communication constraints to simplify specification.
- Key safety properties of the algorithm are encoded as invariance properties of the specification and proven in PVS, e.g.,

$$\forall i, j \in Nodes: i \neq j \wedge zoneStatus(i) = ENTRY \wedge zoneStatus(j) = ENTRY$$
$$\Rightarrow exitTime(i) \neq exitTime(j)$$

Future work

- Develop contingency procedures when no schedules are available.
- Batch simulation evaluations:
  - Large scale UAM operations in an urban environment (e.g., Dallas Fort Worth airspace)
  - Integration of detect and avoid operations along with merging and spacing operations.
- Study interoperability between preflight scheduling services provided by UAM Service Suppliers
- Study the impact of UAM airspace constraints and assumptions:
  - Relation between merge fix airspace and throughput
  - Max/min vehicle speeds allowed for a given merge

Conclusions

- Merging and crossing of UAM routes as well as mutually exclusive resource sharing are challenges that will require creative multi-tier solutions
- Onboard technologies such as ICAROUS will resolve tactical conditions due to uncertainty in UAM flight trajectories, environmental conditions (e.g., wind), and other factors.
- Ability to perform these operations efficiently and reliably without relying on a centralized entity will contribute to the success of the UAM operations.
- Careful understanding of airspace constraints is essential:
  - Relation between merge fix airspace and throughput
  - Max/min vehicle speeds allowed for a given merge fix airspace
- The experimental results show that the integration of merging and spacing into ICAROUS enables autonomous decentralized deconfliction for UAM operations.