UCI University of California, Irvine

Managing Autonomous Mobility on Demand Systems for Better Passenger Experience

Wen Shen and Cristina Lopes
Department of Informatics
University of California, Irvine

Outline

- Introduction
- Motivation and Objective
- Methodology
- Results
- Conclusion
- Future Work

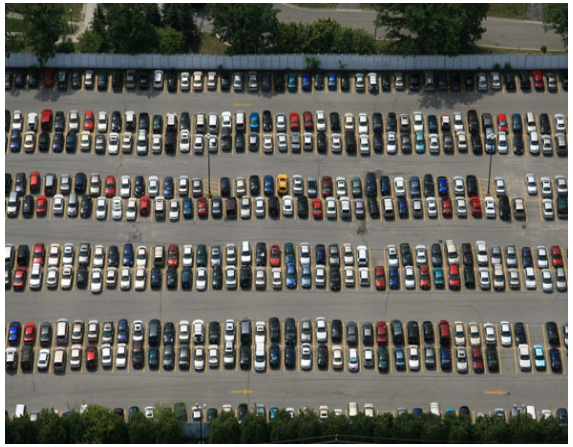
What are Autonomous Mobility on Demand Systems (AMoD systems)?

+

(Photograph by Google, Uber, and Zack Kanter, respectively)

Why AMoD systems?

Traffic congestion on San Diego Freeway



A commuter lot in Toronto, Canada

Google LIDAR for Self-Driving Cars

The XchangeE concept self-driving car by Rinspeed

(Photograph by Dan Chung, Tom Podolec, Rinspeed and Google, respectively)

Motivation and Objective

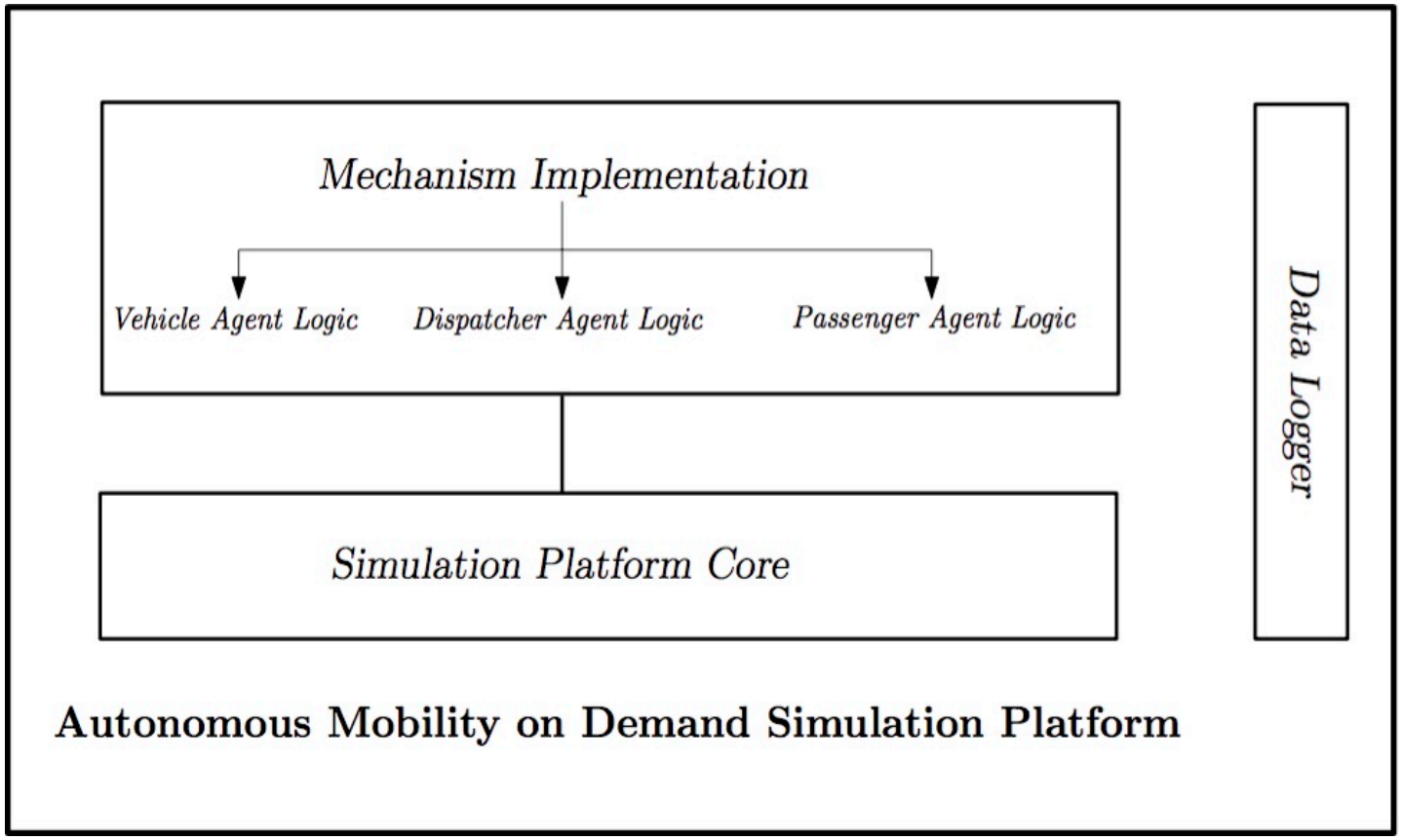
How to manage a fleet of self-driving cars to satisfy passengers' mobility demand?

(Photograph by Google)

Methodology

- Agent-based simulation
- Different Scheduling Strategies
- Expand and Target Algorithm
- Experiments

Simulation Platform

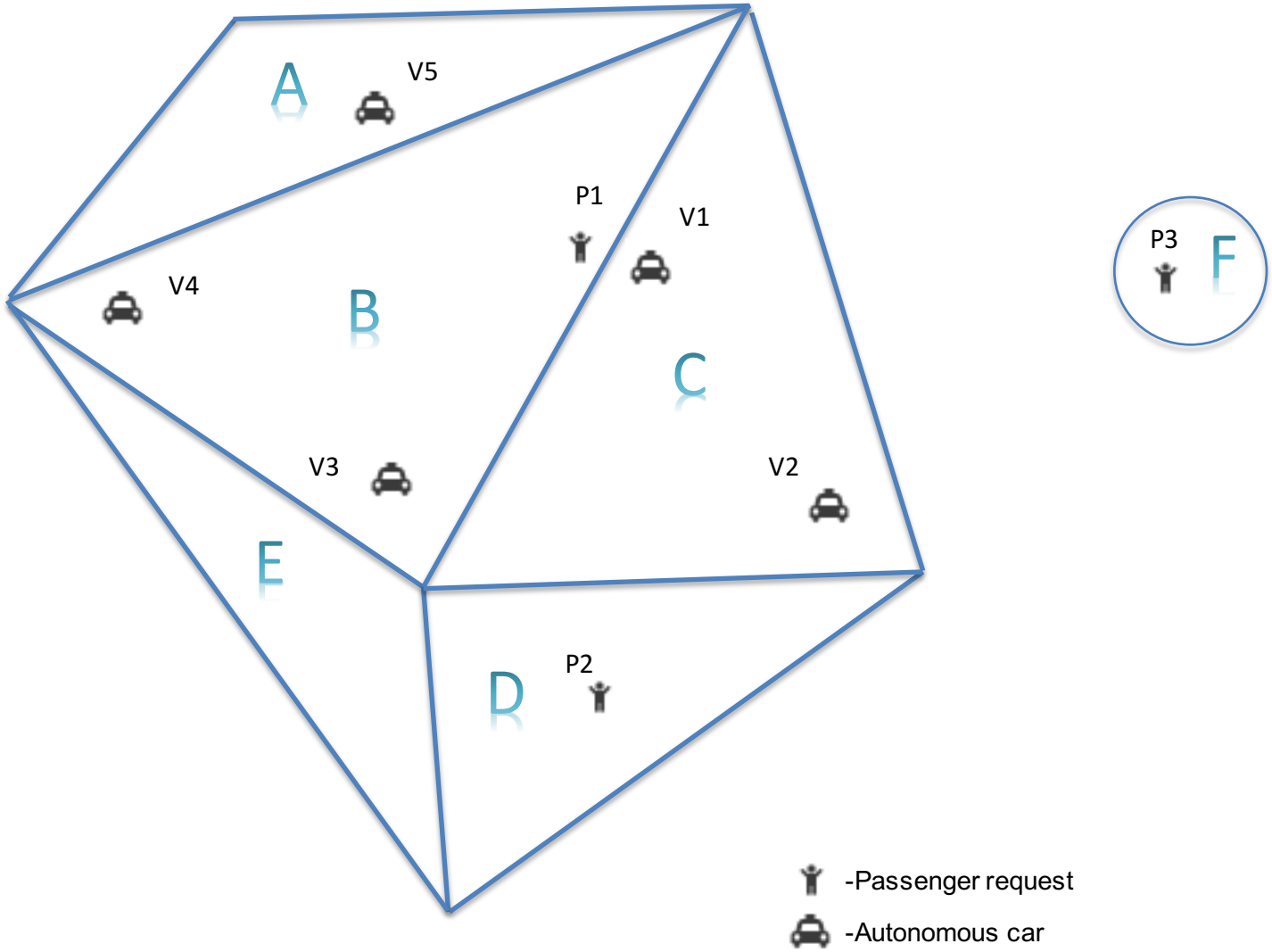


This simulation platform was implemented on top of the MobiltiyTestbed by Certicky et. al.

Scheduling Strategies

- No-Scheduling Strategy
- Static-Scheduling Strategy
- Online-Scheduling Strategy

The Expand and Target Algorithm



The Expand and Target Algorithm

Algorithm 1 *Expand and Target*

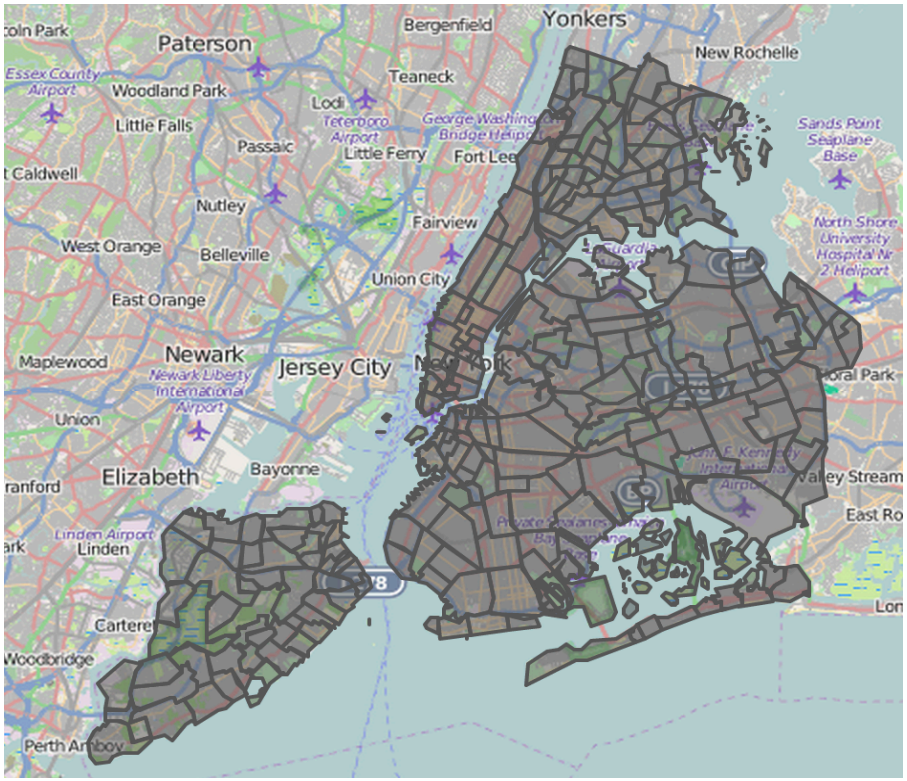
Precondition: c -an incoming call, V - autonomous vehicles in operation, and A-dispatching adjacency schedule

```
1: procedure EXPAND AND TARGET(c, V, A)
2:     identify c's dispatching area a_c ∈ A
3:     if area a_c has adjacent areas (immediate neighbors) B̃_{a_c} ⊆ A then
4:         //begin to expand:
5:         B_{a_c} ← a_c ∪ B̃_{a_c}
6:         //begin to target:
7:         if there are available vehicles in area B_{a_c} then
8:             in area B_{a_c}, search for the nearest available vehicle v ∈ V
9:             return assignment pair (v, c)
10:        else
11:            while a in B_{a_c} do
12:                continue to expand within A
13:            end while
14:            return reject the call c
15:        end if
16:    else
17:        if there are available vehicles in area a_c then
18:            in area a_c, search for the nearest vehicle v ∈ V
19:            return assignment pair (v, c)
20:        else
21:            if there are available vehicles in area A then
22:                in area A, search for the nearest vehicle v ∈ V
23:                //begin to update:
24:                set the dispatching area v as a neighbor of area a_c
25:                return assignment pair (v, c)
26:            else
27:                reject the call c
28:            end if
29:        end if
30:    end if
31: end procedure
```

Experimental Design

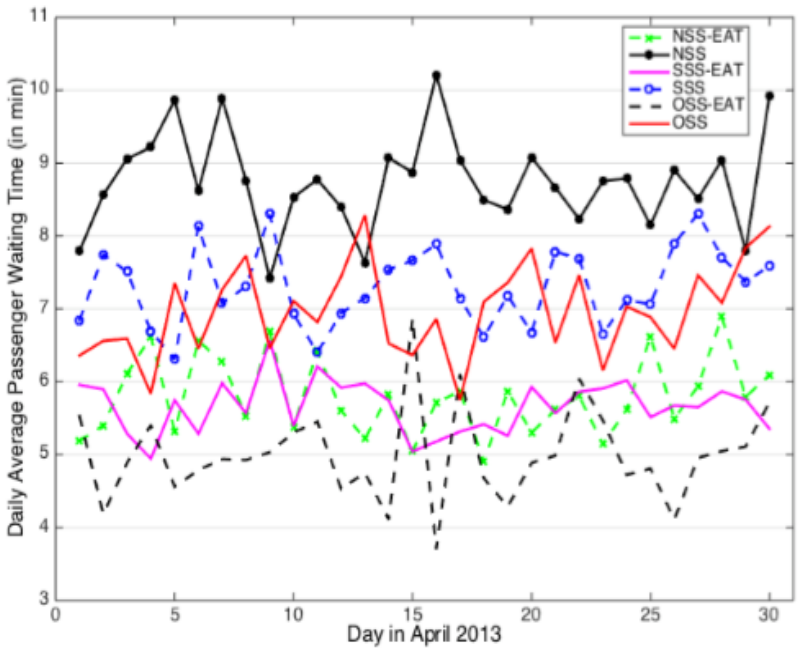
- Data:
  - New York Taxi Data 2013 (124,178,987trips, 12, 216 AVs)
  - OpenStreetMap New York City (OSM XML data ~ 2.19GB)
- Metrics:
  - Average Passenger Waiting Time
  - Trip Success Rate
- Groups:
  - No Scheduling Strategies (NSS)
  - No Scheduling Strategy with Expand and Target (NSS-EAT)
  - Static Scheduling Strategy (SSS)
  - Static Scheduling Strategy with Expand and Target (SSS-EAT)
  - Online Scheduling Strategy (OSS)
  - Online Scheduling Strategy with Expand and Target (OSS-EAT)

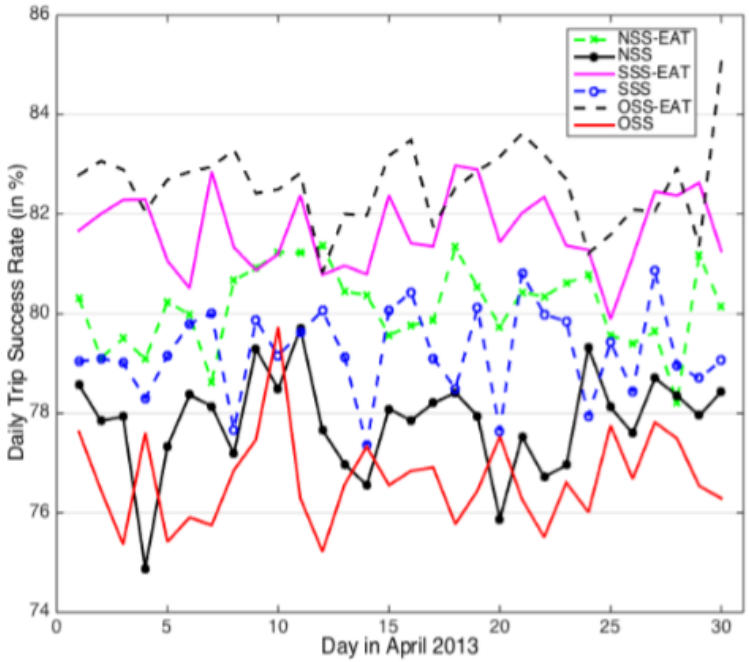
Experimental Design(cont'd)



The adjacency schedule we used in the experiment

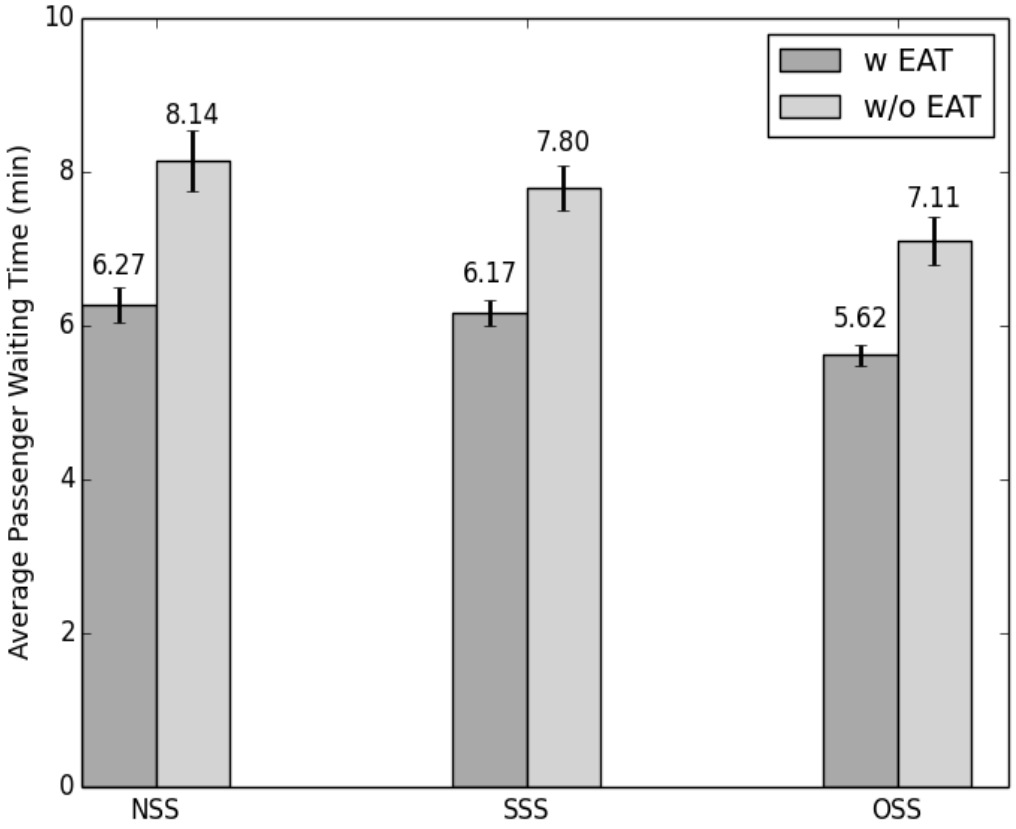
Results





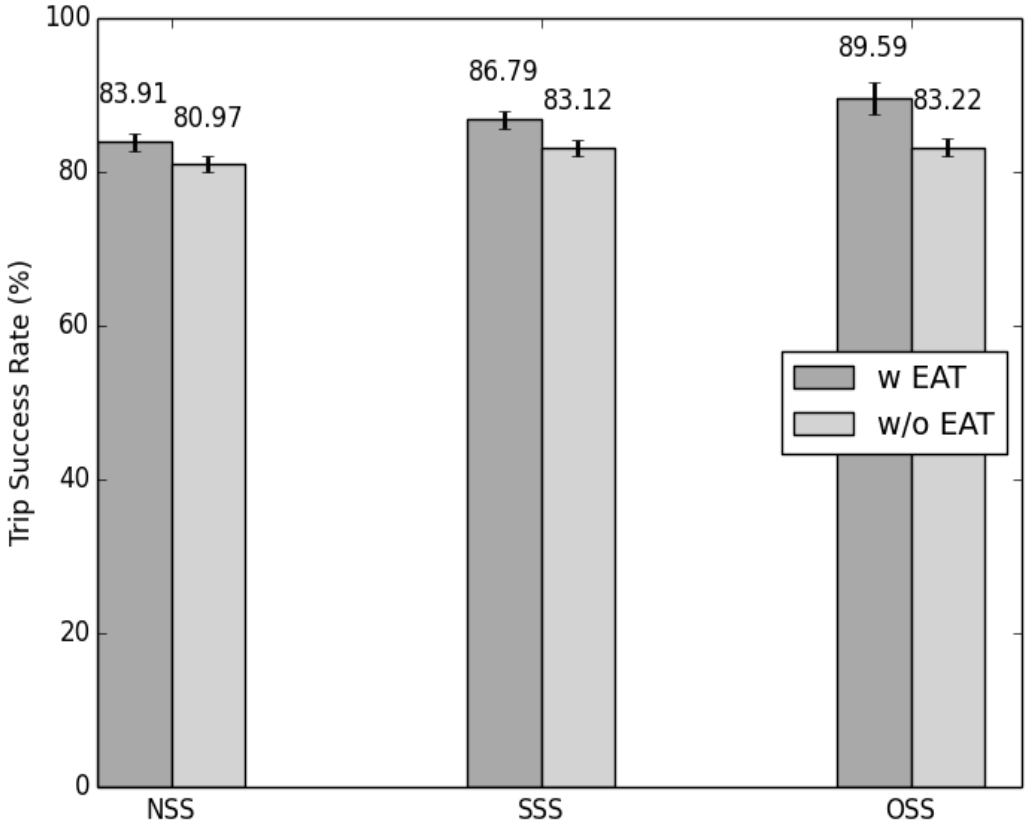
Daily average passenger waiting time (left) and daily trip success rate (right) of an AMoD system using six different strategies (New York Taxi data, April 2013).

Results (cont'd)



A comparison of the average passenger waiting time using different strategies (2013)

Results (cont'd)



A comparison of the trip success rates using different strategies (2013)

Conclusion

- AMoD simulation
- Expand and Target algorithm with three different scheduling strategies
- Performance Improvement
  - Up to 29.82% decrease on average waiting time
  - Up to 7.65% increase on trip success rate

Future Work

- Agent-based simulation platforms for AMoD systems

- Mechanism design for AMoD systems

Thank you !