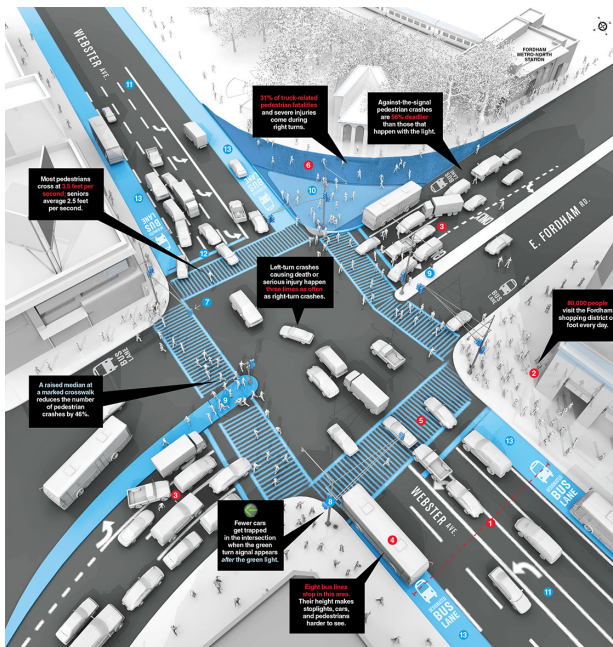


# EE 144/244: Modeling Automated Intersections

Albert Ou   Praagya Singh   Ross Yeager

University of California, Berkeley

13 May 2013



Source: <http://nymag.com/news/features/worst-nyc-traffic-intersections-2012-12/>

# Autonomous Intersections

- ▶ Future roadways face congestion issues due to increasing volume and population density, which would likely lead to greater accident rates
- ▶ Autonomous cars will become much more prevalent in future

## Objective

Model an intelligent intersection system with a centralized controller (as opposed to a peer-to-peer protocol) to route autonomous vehicles

# Original Vehicle Model

Originally, we assumed an intersection environment that contained both human and autonomously controlled vehicles where:

- ▶ Human controlled vehicles respond only to START and STOP commands from the controller
- ▶ Autonomous vehicles respond to START, STOP, CHNG, and SLWDN
- ▶ Controller analyzes all vehicles in system and sends out commands to optimize throughput.
- ▶ The intersection is a hybrid system containing both discrete and continuous components.

# Controller Knowledge

Vehicles are equipped with embedded telemetry units that continually notify the controller with certain information while within the intersection environment:

- ▶ Current location
- ▶ Current speed
- ▶ Planned route
- ▶ Entrance lane

# Deviations from Original Intent

Changes in intersection system model:

- ▶ Shift of focus from optimization towards correct modeling
- ▶ Constant velocity:  $a(t) = 0$
- ▶ Changes in velocity can occur instantaneously
- ▶ System contains autonomous vehicles only
- ▶ Vehicle reaction to controller command is instantaneous
- ▶ Intersection turns have no effect on velocity

# Rationale for Changes

- ▶ Optimization of the system may be intractable for application
- ▶ Human-controlled vehicles would be a rarity in the described system
- ▶ Constant velocity model reasonable within limited intersection distances
- ▶ Limitations with dynamic actor instantiation in Ptolemy modeling
- ▶ Learning curve involved with modeling tool (Ptolemy)

# Modeling Objectives

Let  $c$  denote the event in which a collision event has occurred at time  $t$ , and let  $e$  denote the event in which car  $v$  has entered a route within the intersection.

- ▶ Safety:  $\forall t : G(\neg c)$
- ▶ Fairness:  $\forall v : Fe$

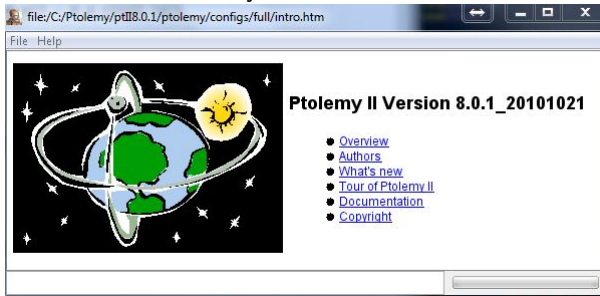


# Vehicle Properties

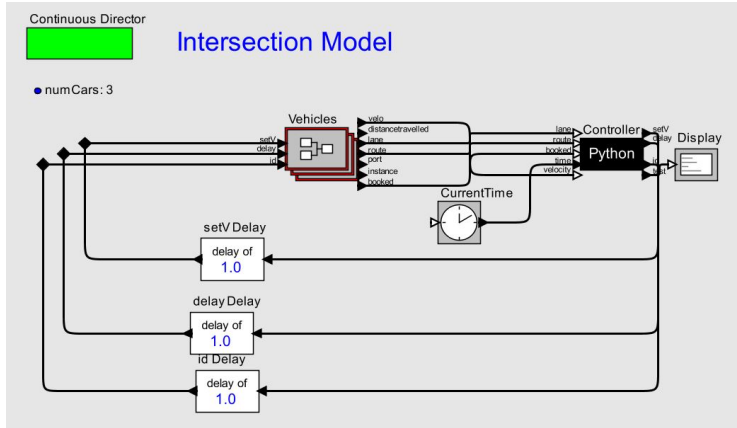
- ▶ Instantaneous change in velocity
- ▶ Constant velocity within intersection
- ▶ Velocity (MPH):  $V \rightarrow V \in [20, 40]$
- ▶ Route intention constant throughout intersection traversal
- ▶ Randomized vehicle entry times
- ▶ Receives velocity change or delayed start time from controller

# Modeling Tool

## Ptolemy II Version 8.0.1



# Intersection Model



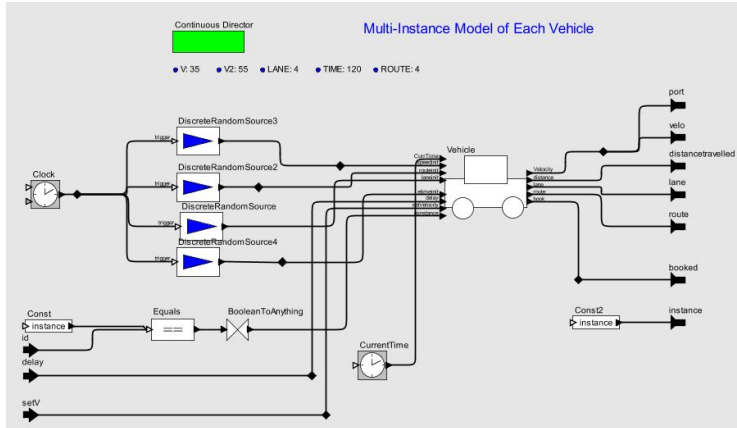
# Modeling Continuous Vehicle Flow

Ptolemy limitations prevent dynamic instantiation of actors.

## Multi-Instance Composite Actor

- ▶ Vehicle instances accessed by ID encoding
- ▶ Create  $N$  vehicle Modal Model instances
- ▶ Generate random:
  - ▶ Intersection entry time: 1 – 45 seconds
  - ▶ Initial velocity (m/s):  $V_0 \in [20, 40]$
  - ▶ Direction:  $N, S, E, W$
  - ▶ Route:  $L, S1, S2, R$
- ▶ Implement as many instances of the car model as desired
- ▶ Interacts with Python-based controller
- ▶ Car can “re-enter” system only after it exits (re-use vehicles vs. generating new vehicles)

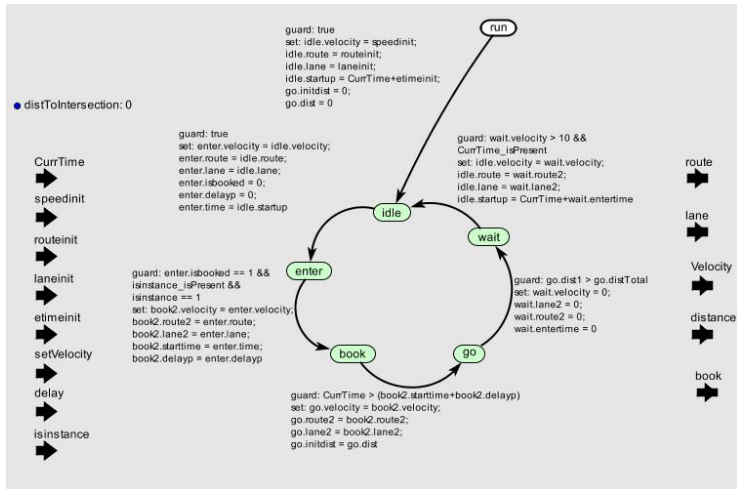
# Vehicle Modeling



# Vehicle States

- ▶ *RUN*
- ▶ *IDLE*
- ▶ *ENTER*
- ▶ *BOOK*
- ▶ *GO*
- ▶ *WAIT*

# Vehicle FSM Actor



# ENTER State

Abstraction:

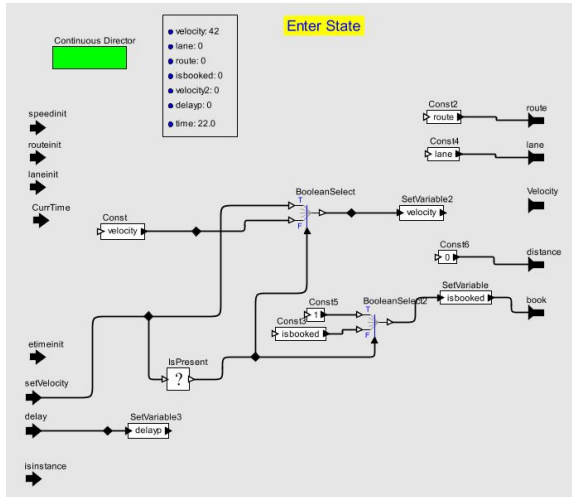
- ▶ Equivalent to pre-conflict entrance approach
- ▶ Initial velocity used to calculate time until intersection
- ▶ Stops at intersection if instructions not received from controller

Guard:

- ▶  $t_{\text{GLOBAL}} > t_{\text{ENTER}}$



# ENTER State



# GO State

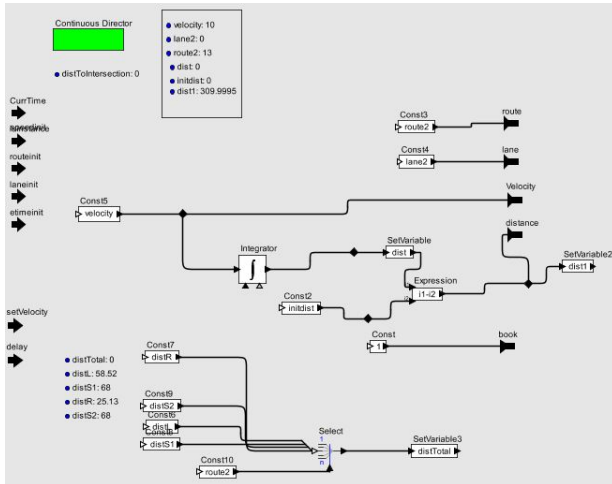
Abstraction:

- ▶ Equivalent to the approach through the intersection
- ▶ Tracks when the vehicle leaves the system
- ▶ Stops at intersection if instructions not received from controller

Guard:

- ▶  $t_{\text{GLOBAL}} > t_{\text{STRAIGHTAWAY}}$

# GO State



# Collision Detection Schemes

The primary rationale for an automated intersection is to prevent traffic accidents. Methods to predict collisions:

- ▶ Tiling – Autonomous Intersection Management (AIM) project at UT Austin
- ▶ Euclidean distance between three-dimensional trajectories  
 $f(x, y, t)$

We would like an alternative that is less computationally intensive and amenable to optimization.

# Conflict Points

## Definition

A *traffic conflict point* is the point at which multiple traffic flows may cross or merge.

- ▶ Require that all vehicles adhere to designated lanes when traversing the intersection, so that trajectories can be bounded exactly.
- ▶ The set of conflict points are determined by the intersections between curves representing all legal paths of vehicles.

# Intersection Geometry

- ▶ Four-way symmetric intersection with orthogonal arms and perfect alignment
- ▶ Open space without obstructions (e.g., channelization features, pedestrians)
- ▶ Total number of lanes at any exit approach equals the total number of lanes at any entrance approach
- ▶ Traffic lanes at an exit approach can only serve as the exit of lanes with the same lane number relative to the center line
- ▶ No exclusive right/left turn lanes

# Path Equations

$N$  number of lanes per direction

$M$  number of left turn lanes per direction ( $M < N$ )

$w$  lane width

$c$  corner radius

$d$  minimum buffer distance between vehicles in potential conflict

$k$  lane number

Quantities defined for convenience:

$$\alpha_k = \left(k - \frac{1}{2}\right) w$$

offset from center line

$$\beta_k = \frac{1}{\sqrt{2}} \left[ \frac{d}{2} + \left(M - k + \frac{1}{2}\right) w \right]$$

projected offset from origin

$$\gamma = Nw + c$$

half of intersection width

# Path Equations

► Straight:

$$y = \pm \alpha_k \quad k = 1, \dots, N \quad (1)$$

$$x = \pm \alpha_k \quad k = 1, \dots, N \quad (2)$$

- Right turns: Circular quadrants of radius  $\frac{w}{2} + c$   
( $k = M + 1, \dots, N$ )

Direction	Center	Entrance	Exit
S $\rightarrow$ E	$(\gamma, -\gamma)$	$(\alpha_k, -\gamma)$	$(\gamma, -\alpha_k)$
N $\rightarrow$ W	$(-\gamma, \gamma)$	$(-\alpha_k, \gamma)$	$(-\gamma, \alpha_k)$
E $\rightarrow$ N	$(\gamma, \gamma)$	$(\gamma, \alpha_k)$	$(\alpha_k, \gamma)$
W $\rightarrow$ S	$(-\gamma, -\gamma)$	$(-\gamma, -\alpha_k)$	$(-\alpha_k, -\gamma)$



## Path Equations: Left Turns

- Circular arcs fitted to three control points ( $k = 1, \dots, M$ ):

Direction	Entrance	Midpoint	Exit
S $\rightarrow$ W	$(\alpha_k, -\gamma)$	$(-\beta_k, -\beta_k)$	$(-\gamma, \alpha_k)$
N $\rightarrow$ E	$(-\alpha_k, \gamma)$	$(\beta_k, \beta_k)$	$(\gamma, \alpha_k)$
E $\rightarrow$ S	$(\gamma, \alpha_k)$	$(\beta_k, -\beta_k)$	$(-\alpha_k, -\gamma)$
W $\rightarrow$ N	$(-\gamma, -\alpha_k)$	$(-\beta_k, \beta_k)$	$(\alpha_k, \gamma)$

- Outermost left turn lane is diagonally offset  $\frac{d+w}{2}$  from origin to avoid conflicts with oncoming left turn traffic.
- For simplicity, U-turns are disallowed.

# Circumcircle of a Triangle

$$a = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

$$c = - \begin{vmatrix} (x_1^2 + y_1^2) & x_1 & y_1 \\ (x_2^2 + y_2^2) & x_2 & y_2 \\ (x_3^2 + y_3^2) & x_3 & y_3 \end{vmatrix}$$

Circumcenter:

$$x_c = -\frac{b_x}{2a} \quad (3)$$

$$y_c = -\frac{b_y}{2a} \quad (4)$$

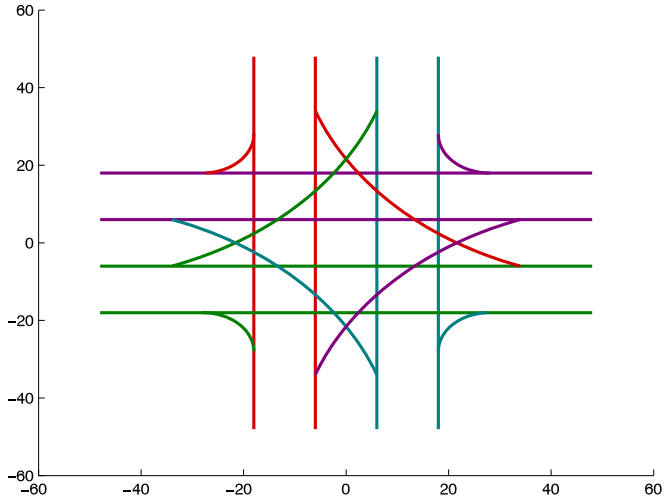
$$b_x = - \begin{vmatrix} (x_1^2 + y_1^2) & y_1 & 1 \\ (x_2^2 + y_2^2) & y_2 & 1 \\ (x_3^2 + y_3^2) & y_3 & 1 \end{vmatrix}$$

$$b_y = \begin{vmatrix} (x_1^2 + y_1^2) & x_1 & 1 \\ (x_2^2 + y_2^2) & x_2 & 1 \\ (x_3^2 + y_3^2) & x_3 & 1 \end{vmatrix}$$

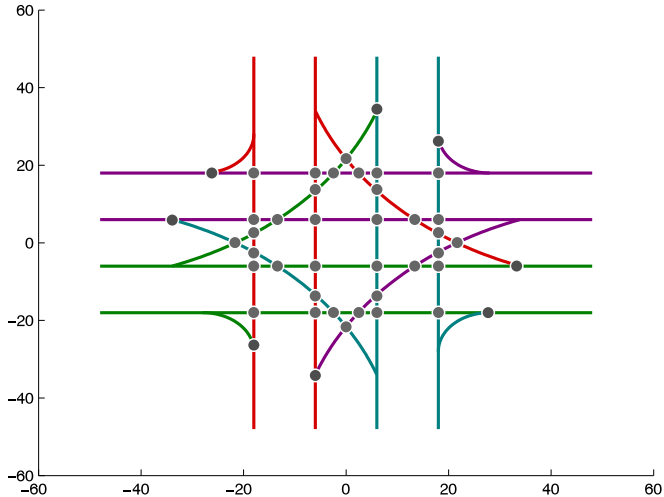
Circumradius:

$$r = \frac{\sqrt{b_x^2 + b_y^2 - 4ac}}{2|a|} \quad (5)$$

# Conflict Points of Model Intersection



# Conflict Points of Model Intersection



# Conflict Zones

- ▶ In reality, vehicles are not point masses – they occupy area.
- ▶ **Safety property:** Enforce a minimum separation distance  $d$  at all times between vehicles traveling on conflicting paths.

## Definition

A *traffic conflict zone* is the segment of a given path within distance  $d$  of a conflict point.

- ▶ The initial and terminal points of a conflict zone can be expressed as an ordered pair of offsets/displacements along the curve, relative to the start of the path.

## Position as Arc Length

- ▶ Recall that all turns are modeled as circular arcs.
- ▶ Central angle with vertex  $(x_c, y_c)$ , endpoint  $(x, y)$  on the circle, and one side parallel to the x-axis:

$$\theta(x, y) = \left| \arccos \left( \frac{x - x_c}{r} \right) \right| = \left| \arcsin \left( \frac{y - y_c}{r} \right) \right| \quad (6)$$

- ▶ Arc length (displacement) from the initial point  $(x_0, y_0)$  to the terminal point  $(x_1, y_1)$ :

$$s = r |\theta(x_1, y_1) - \theta(x_0, y_0)| \quad (7)$$

- ▶ Only one coordinate needs to be known for each endpoint.

# Conflict Point Displacements

- ▶  $w = 12$  ft,  $c = 10$  ft,  $d = 15$  ft
- ▶ Merge conflict points (highlighted in red) are the exit points of turns, which also indicate the total length of each path.

	→					↓					←			
	L	S1	S2	R		L	S1	S2	R		L	S1	S2	R
L	44.62	34.44	18.12	–		–	23.99	40.30	–		13.80	58.42	–	–
S1	–	28.00	16.00	–		47.38	–	–	–		20.62	40.00	52.00	68.00
S2	68.00	28.00	16.00	–		36.39	–	–	–		31.61	40.00	52.00	–
R	–	–	25.13	–		–	–	–	–		–	–	–	–

# Conflict Zone Partitioning

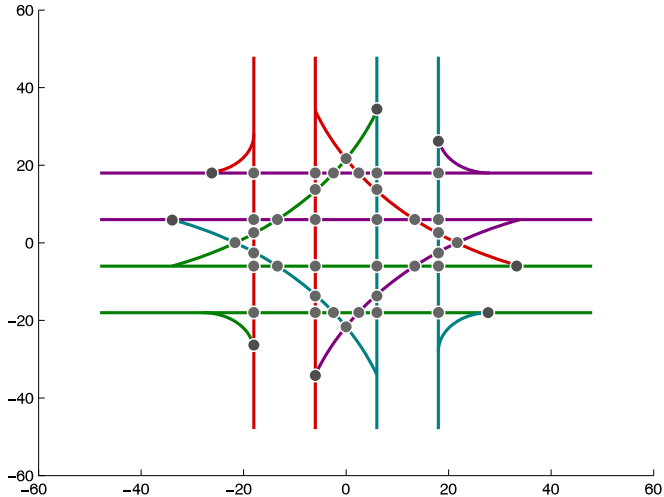
Rigorous method is to calculate intersection points between a circle of radius  $d$ , centered at conflict point  $(x, y)$ , and involved path curves – more complicated than necessary.

- ▶ Conflict zones for straight paths are simply  $x \pm d$  for  $E \rightleftharpoons W$  and  $y \pm d$  for  $N \rightleftharpoons S$ .
- ▶ The chord length approximates the circular arc length when the subtended angle remains relatively small. For left turns where  $d \ll r$ , the extent of a conflict zone can be treated as  $s \pm d$ .
- ▶ The entire right turn path is considered a conflict zone.

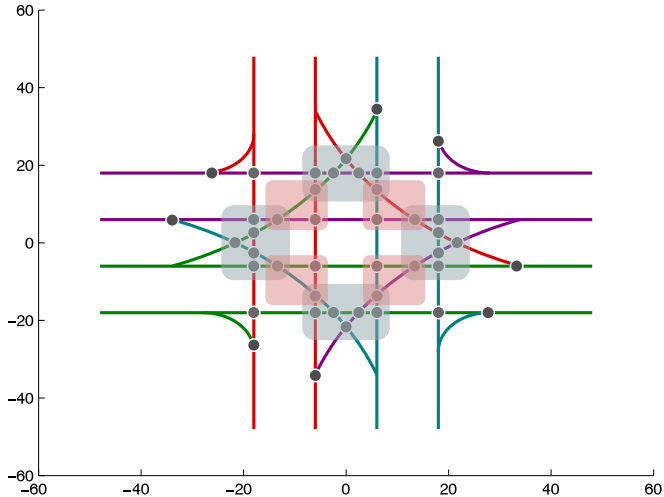
**Aggregation:** Group conflict points into multiple overlapping zones for coverage and flexibility



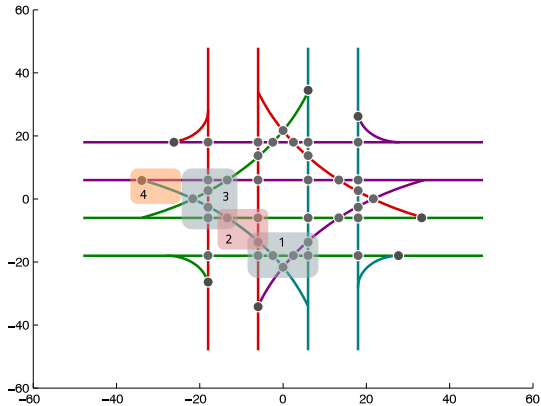
# Conflict Zone Aggregation



# Conflict Zone Aggregation

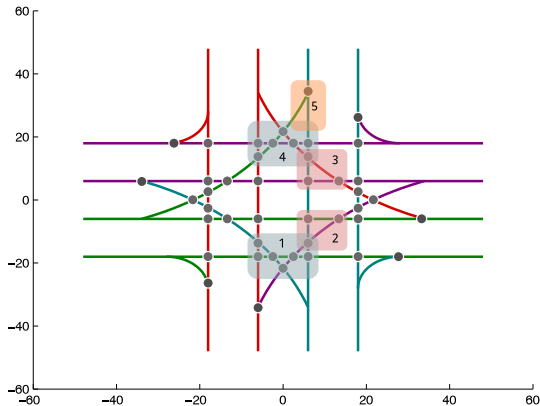


# Left Turn Path



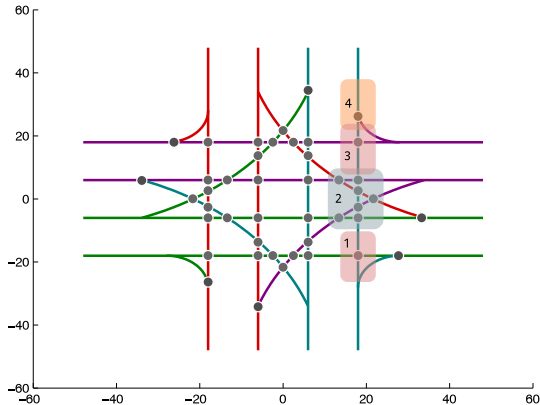
1	8.80	28.99
2	18.88	39.44
3	29.44	49.62
4	43.52	58.52

# Straight Path (Inner)



1	8.50	25.62
2	15.62	34.00
3	34.00	52.38
4	42.38	59.50
5	53.00	68.00

# Straight Path (Outer)



1	8.50	23.50
2	20.50	47.50
3	44.50	59.50
4	53.00	68.00

# Collision Detection Algorithm

*Given:* Intended route and lane assignment, initial velocity  $v_0$  at the intersection entrance, planned acceleration function  $a(t)$ , dimensions of vehicle

- 1 Determine the set of conflict zones along the path.
- 2 Numerically integrate to determine when endpoints of conflict zones are crossed, yielding a set of time intervals.
- 3 For each conflict zone, check whether the corresponding time interval overlaps with that of any other active vehicle.

# Observations

- ▶ Conflict zones are static and therefore need only to be calculated once during initialization.
- ▶ Runtime does not require knowledge about curvature of paths
  - only linear displacements matter.

## Scheduling

Reservations for a vehicle are scheduled as time intervals during which it has exclusive access to the conflict zones along its path.

As with majority of other CS problems, finding an optimal solution is much more difficult than checking a solution.

## Questions for Validation

- ▶ Is it possible for a vehicles with different velocities in the same lane to collide without being detected?
- ▶ Are static conflict zones optimal?
- ▶ Can collisions occur at locations other than conflict points?



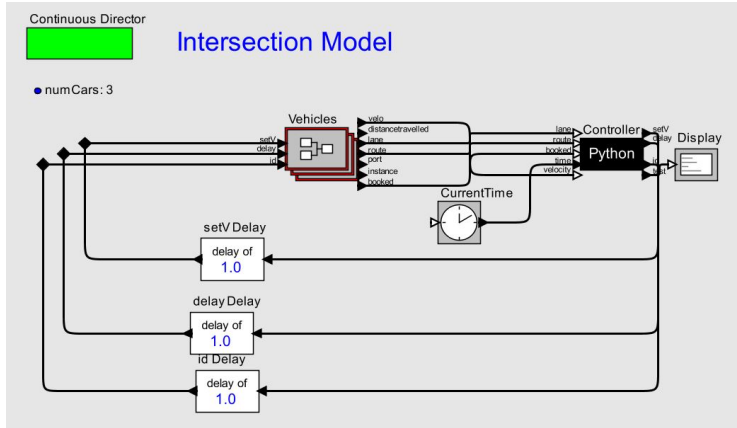
## Questions for Validation

- ▶ Is it possible for a vehicles with different velocities in the same lane to collide without being detected?
  - ▶ No, if conflict zones overlap, then the two vehicles will be in the same conflict zone when the incident occurs.
- ▶ Are static conflict zones optimal?
  - ▶ No, conflict zones must consider worst-case scenarios. In reality, vehicles may be at opposite ends of the same conflict zone and be sufficiently separated.
- ▶ Can collisions occur at locations other than conflict points?
  - ▶ Yes, trajectories can conceivably pass closely together but not intersect. Intersection topology must be designed to avoid this.

# Controller Properties

- ▶ Lack of multiport output (serial communication)
- ▶ Delayed feedback to vehicle
  - ▶ Prevent a zeno system
  - ▶ Models communication delay
- ▶ Receives requests and sends commands to system vehicles
- ▶ Inputs to controller: *velocity, booking, entertime*
- ▶ Python script actor

# Intersection Model



# Controller Algorithm

```
Initialize list of conflict zones and displacements along paths
for each timestep  $t$  do
   $current \leftarrow$  all tokens received from Vehicle modal model
  for each  $car$  in  $current$  do
    if  $car$  needs reservation then
       $v_{\max} \leftarrow$  maximum velocity  $\in [v_{\text{current}} \pm 10]$  that ensures safety
      if  $v_{\max}$  exists then
         $v_{\text{new}} \leftarrow v_{\max}$ 
         $t_{\text{delay}} \leftarrow 0$ 
      else
         $v_{\text{new}} \leftarrow v_{\text{current}} + 10$ 
         $t_{\text{delay}} \leftarrow$  end of latest reservation for all conflict zones
      end if
      broadcast( $v_{\text{new}}, t_{\text{delay}}, \text{id} \leftarrow car.\text{id}$ )
    end if
  end for
end for
```

# Technical Challenges

- ▶ Continuous car flow given static number of vehicles
- ▶ Delivering controller commands to specified car instance
- ▶ Continuous time solver time step evaluation issues
- ▶ Hybrid system causality issues
- ▶ Scalability

# Results

- ▶ Functional hybrid model
- ▶ Continuous vehicle flow with static vehicle set
- ▶ Centralized responsive controller
- ▶ Rigorous intersection path definitions
- ▶ Multiplexed communication to vehicles

## Further Work

- ▶ Investigate resource scheduling algorithms for optimization
- ▶ Introduce non-zero vehicle acceleration into the model
- ▶ Add support for vehicles with human drivers