Stacking object detection on top of tracking is poor computationally (frame by frame)

Methods on the old car might not be appropriate now since they are simple object recognition

All new object tracking system needed ourselves probably

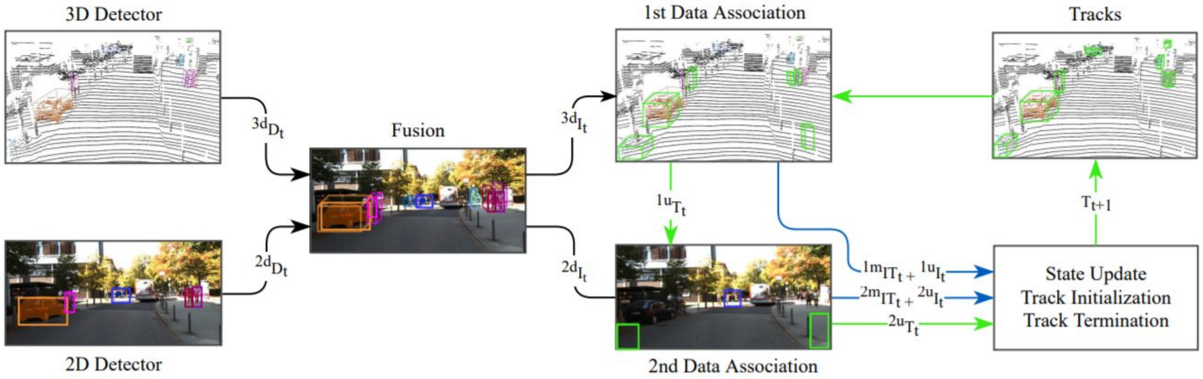
EagerMOT - https://arxiv.org/pdf/2104.14682v1.pdf

Open sourced code
https://github.com/aleksandrkim61/EagerMOT

LIDAR integration

Compiles multiple sensor data

Problem - multiple cameras in their model



OpenCV

Lots of already implemented libraries

All image based

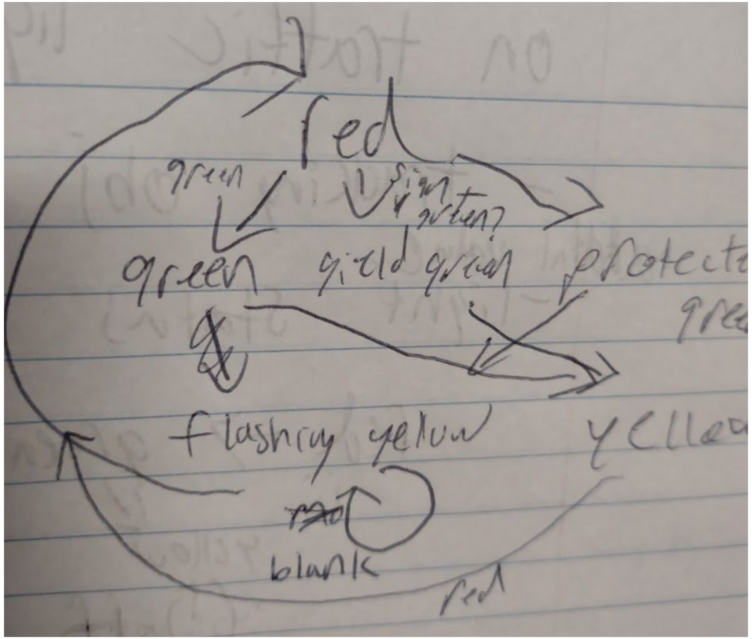
Subclasses of found objects

Given bounding box can function as a state machine

Traffic Light

Break flow takes 2? - becomes noise resistant

Normal flow takes 1

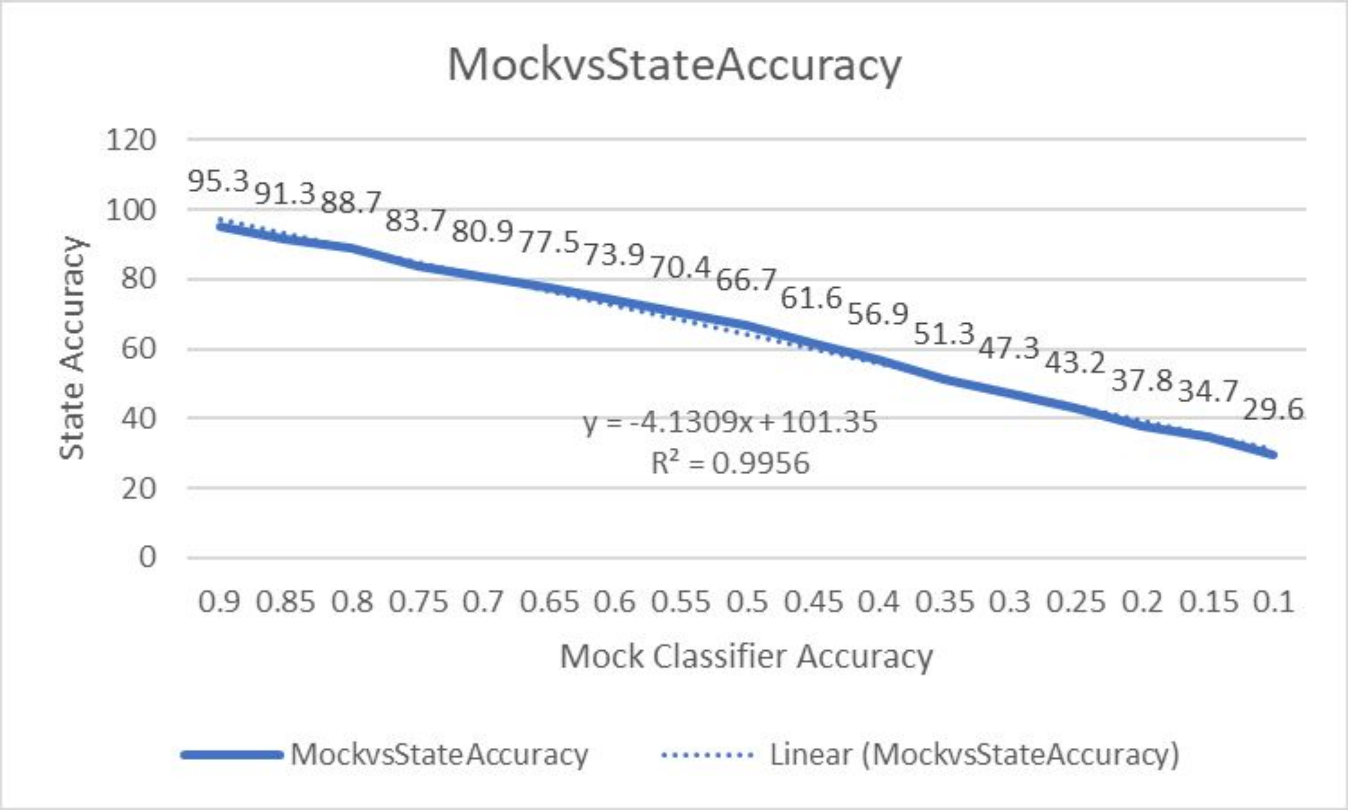


Traffic Cones

Data Analysis

- 10,000 samples from 0.9 to 0.1 accuracy, totaling 160k samples
- Data shows performance improvements for noise and resiliency
- Too many sheets in an excel doc

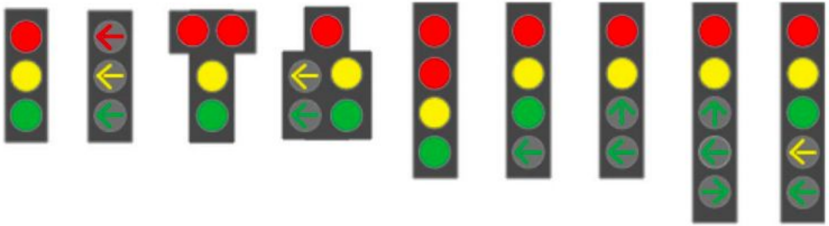
- every 1% drop in classifier accuracy results in 0.8% drop in state accuracy
- 10 points better @ 70% accuracy, 20 points better @ 0.1 (essentially random guessing)



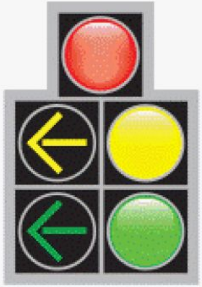
How Traffic Lights Work

Colors

Light Options

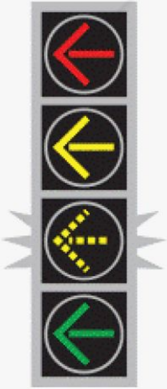


CURRENT CONFIGURATION . . .



Five-section “Doghouse” signal

NEW CONFIGURATION



Steady Red Arrow

Drivers turning left must stop and wait (except where permitted by law).

Steady Yellow Arrow

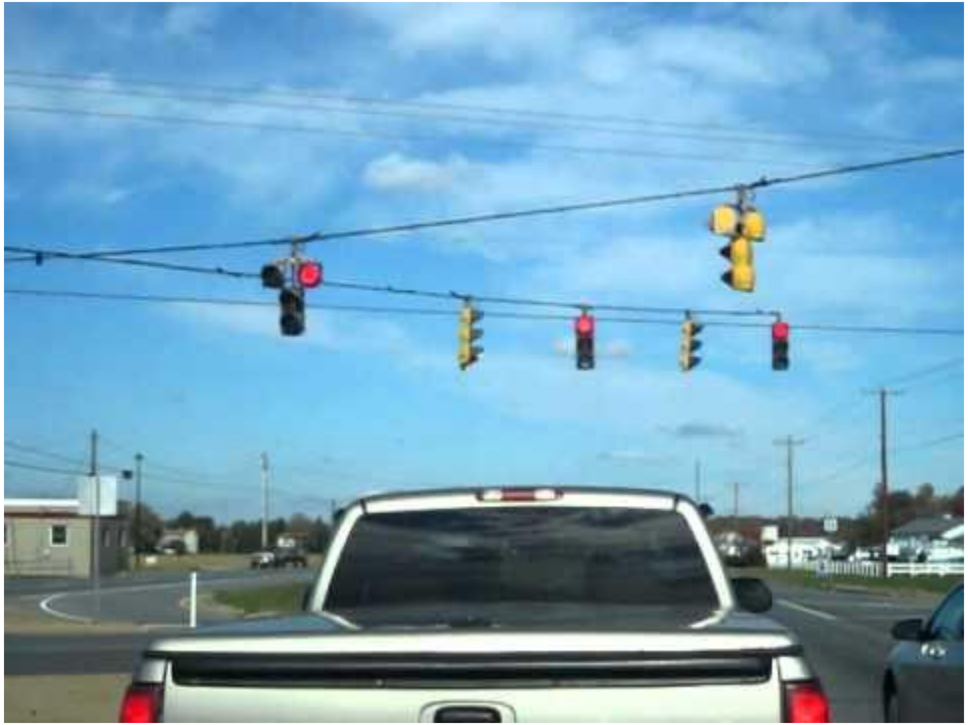
Stop, if you can do so safely.

Flashing Yellow Arrow

Proceed with left turn after yielding to oncoming traffic and pedestrians.

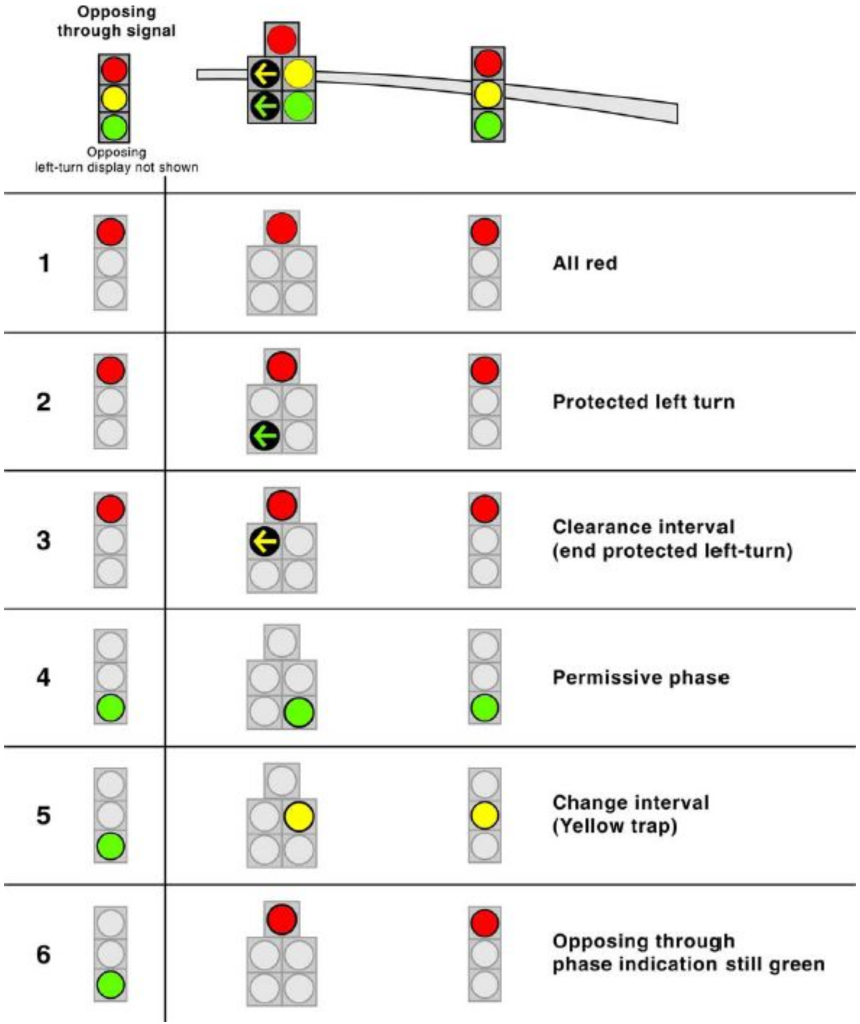
Steady Green Arrow

Proceed with left turn.



https://www.youtube.com/watch?v=4r45aJQ0zDg Source

Will Typically refer to left 2 lanes and be for indicating left turns



Opposing through signal

Opposing left-turn display not shown

1 All red

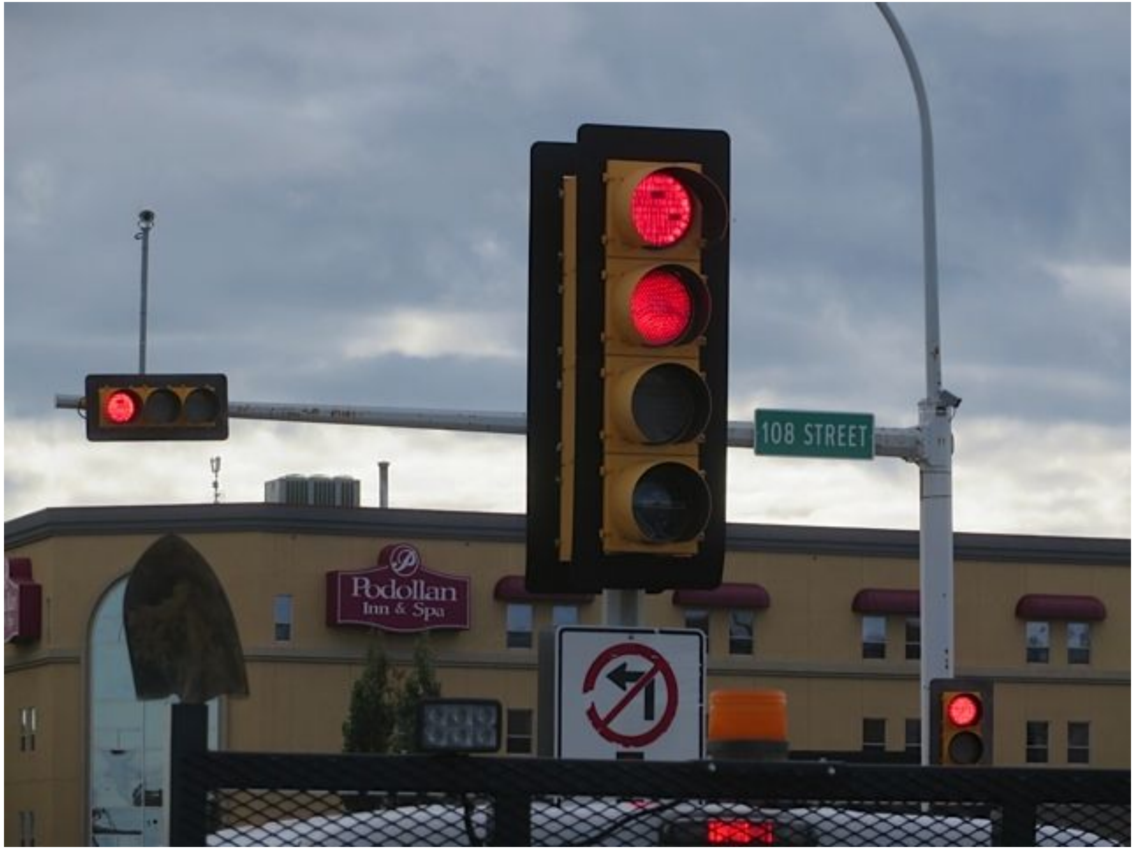
2 Protected left turn

3 Clearance interval (end protected left-turn)

4 Permissive phase

5 Change interval (Yellow trap)

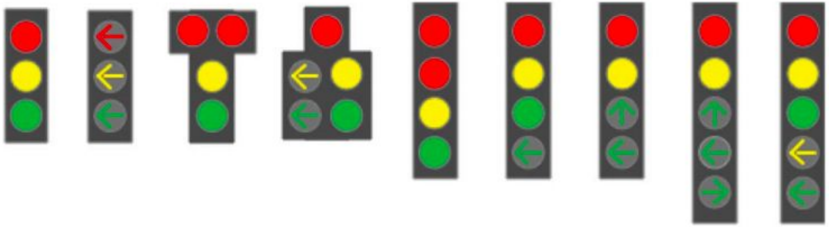
6 Opposing through phase indication still green



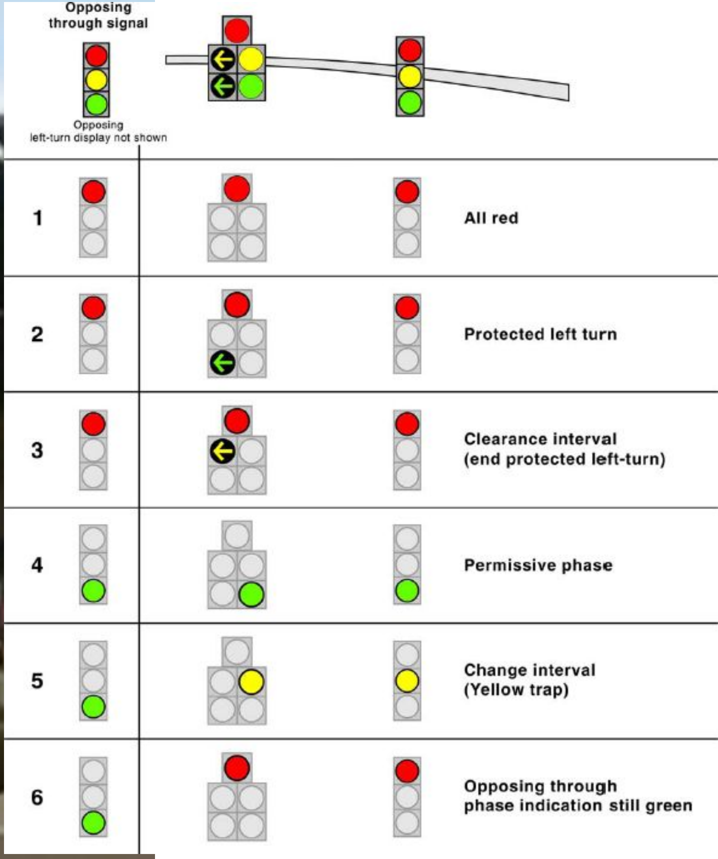
Functions differently in Texas sometimes. Can just be left turn indicator or be there for more light in rural areas

The double-red ball aspect is used in Saskatchewan and Alberta, Canada, to indicate a protected-prohibited left turn signal. A sign with the universal no left turn symbol and a depiction of the double red light is mounted near the signal to indicate that no left turns are permitted on a double-red light. Intersections with this configuration are quite common in Saskatoon, Calgary, and Edmonton. - https://en.wikipedia.org/wiki/Variations_in_traffic_light_operation#Double_red_lights

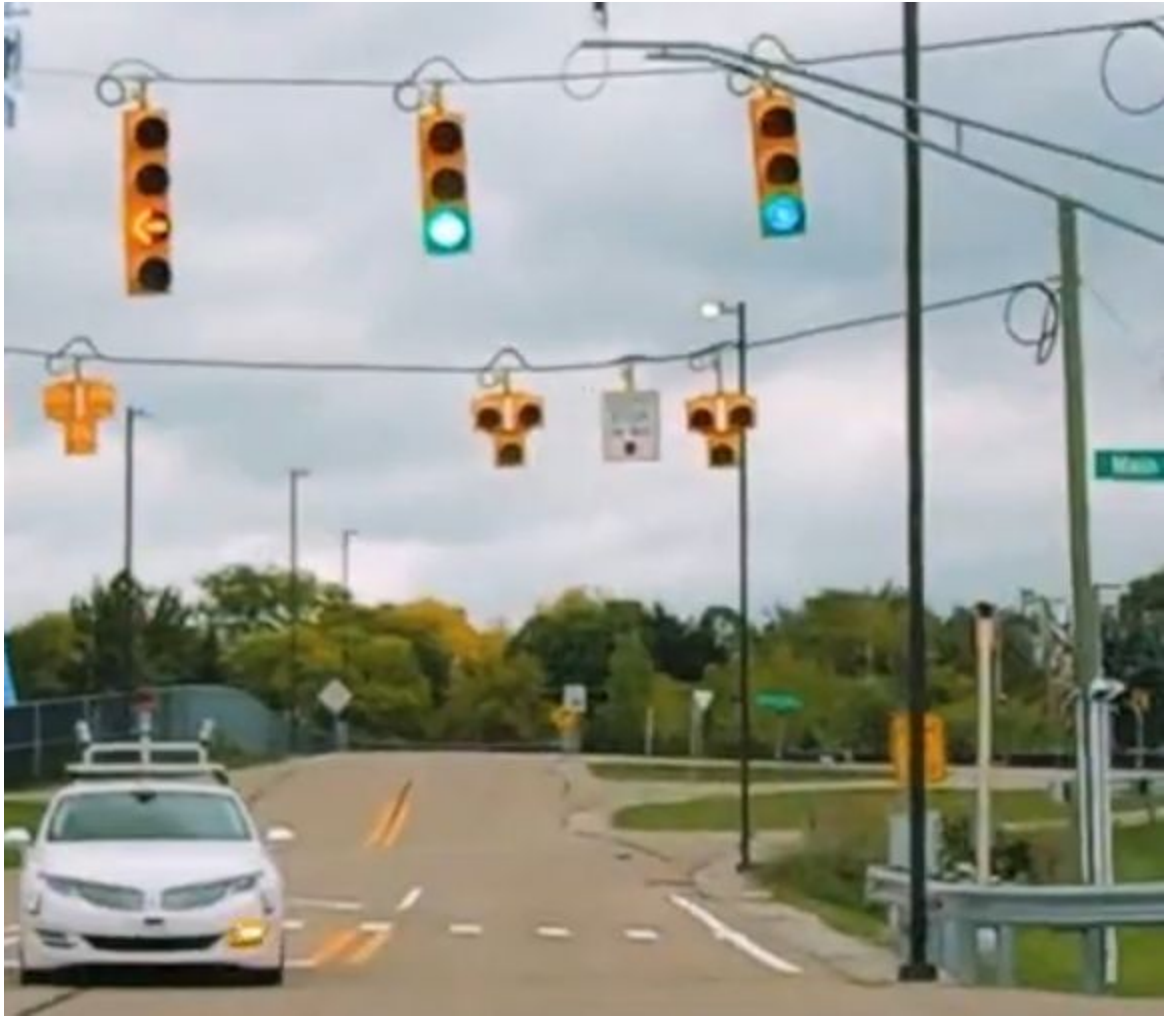
Rest self explanatory as left turn signals for protected vs not



Found Lights

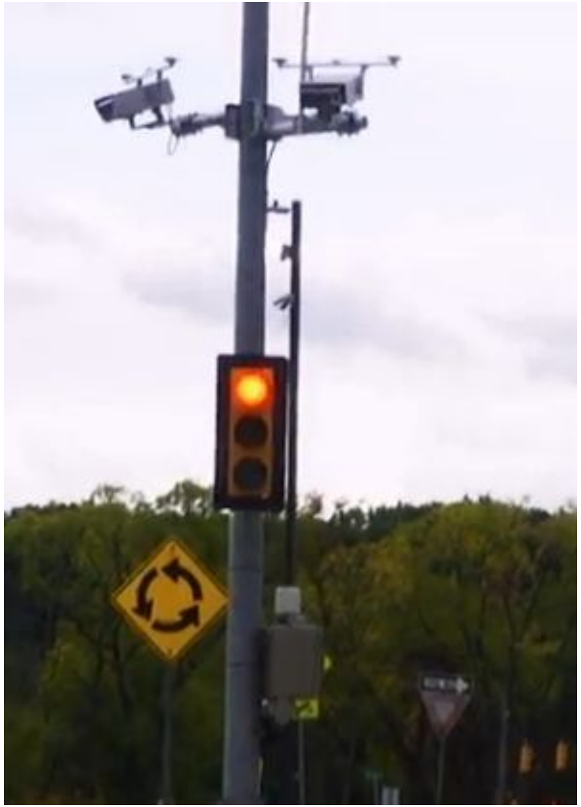


1	All red
2	Protected left turn
3	Clearance interval (end protected left-turn)
4	Permissive phase
5	Change interval (Yellow trap)
6	Opposing through phase indication still green



Triangles are apparently a thing

Flashing Red



Will need to modify state machine for this.

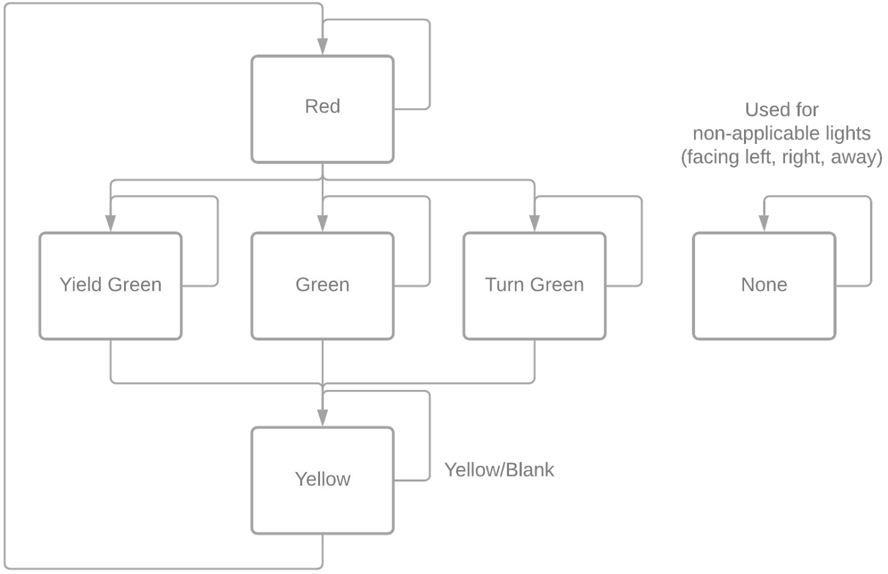
Source: 2020 Mcity Test Facility Video Tour

Todo:

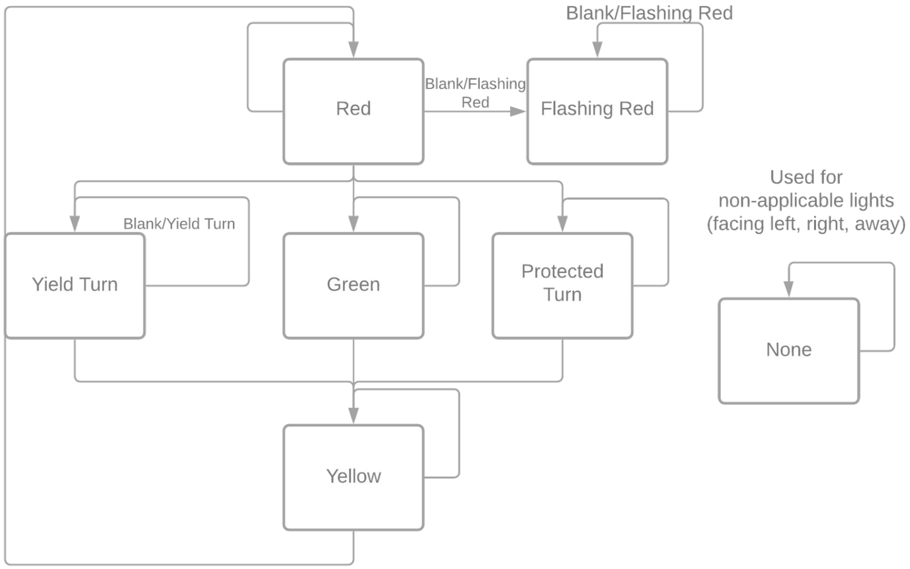
- Modify State Machine for the new states/remove unnecessary ones
- Run mock data classifier again to see accuracy change

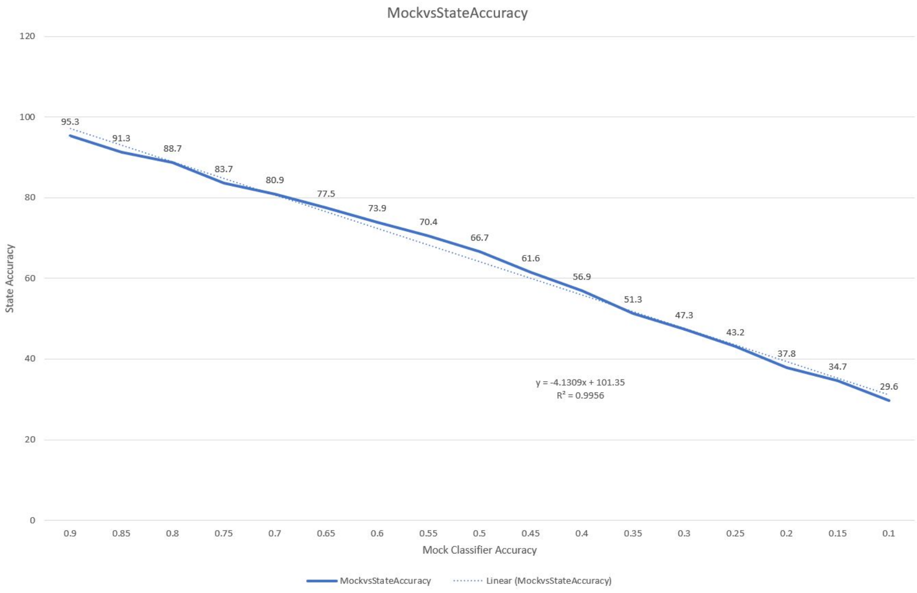
Updated Traffic Light State Machine And Analysis

Old


Red
Yield Green
Green
Turn Green
Yellow
Yellow/Blank
Used for non-applicable lights (facing left, right, away)
None

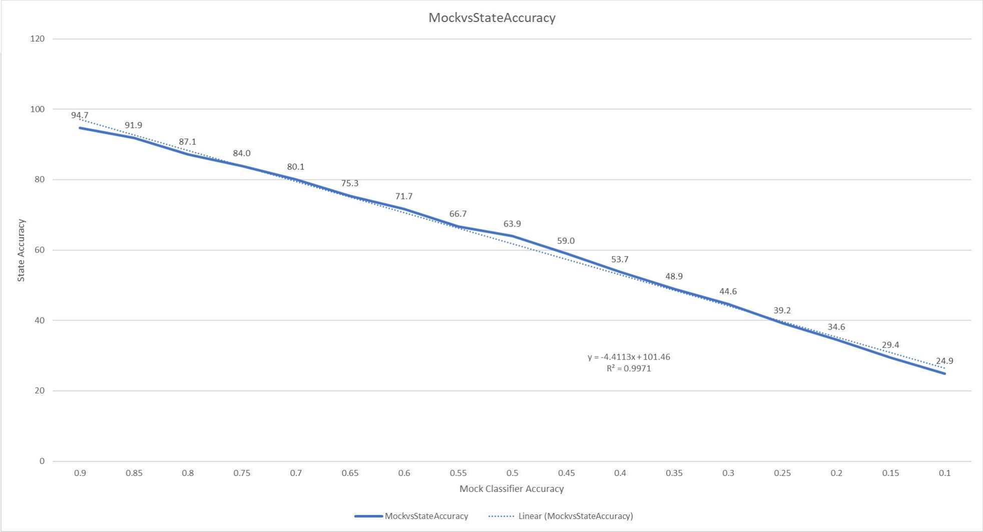
New


Blank/Flashing Red
Red
Blank/Flashing Red
Flashing Red
Blank/Yield Turn
Yield Turn
Green
Protected Turn
Used for non-applicable lights (facing left, right, away)
None
Yellow


MockvsStateAccuracy
State Accuracy
120
100
80
60
40
20
0
95.3
91.3
88.7
83.7
80.9
77.5
73.9
70.4
66.7
61.6
56.9
51.3
47.3
43.2
37.8
34.7
29.6
y = -4.1309x + 101.35
R² = 0.9956
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
Mock Classifier Accuracy
MockvsStateAccuracy
Linear (MockvsStateAccuracy)

Old

New


MockvsStateAccuracy
State Accuracy
120
100
80
60
40
20
0
94.7
91.9
87.1
84.0
80.1
75.3
71.7
66.7
63.9
59.0
53.7
48.9
44.6
39.2
34.6
29.4
24.9
y = -4.4113x + 101.46
R² = 0.9971
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
Mock Classifier Accuracy
MockvsStateAccuracy
Linear (MockvsStateAccuracy)

Takeaways

- Renamed some states to more align with the wordings used in official documentation
- More states = less reliability advantage over raw model (marginally)
- Reliability goes from 8% for each 1% classification drop to 8.8% for each 1% classification drop.
- Still 81% → 80% accuracy when base accuracy is 70%

Next Steps

- Work towards some more of the challenges listed previously (investigating nn speeds, different state of art models, etc)
- Classification model for the traffic lights.
- When classification available test state machine with an actual system as opposed to random generations.

Keeping track of previous states to classify versus implicit

Keep an array of the last x inputs to help with subclasses (arrows and flashing)

Flashing yellow classification

Yellow lights are 1 second on, 1 second off. As a result if we are matching 10fps, 20 classification array should be 50% blanks 50% yellows.

[(yellow, round), (yellow, round), (yellow, round), (yellow, round), (blank, blank), (blank, blank), (blank, blank), (yellow, round), …]

If sum(blanks) > 25% then type is goes yellow → flashing yellow

State machine breakpoint from yellow → flashing yellow is now >25% blanks during yellow. Aka 6 frames of no light @ 10fps. As a hyperparameter that can be changed quickly it is guesstimated and given full sequence videos can check.

Protected Turn

Instability in protected turn classification maintains a bound of 30% for the barrier between green light and protected turn. Same as yellow but simpler case since only dealing with arrow vs round fluctuation.

[(green, round), (green, arrow), (green, round), (green, round), (green, arrow), (green, round), (green, round), (green, arrow), (green, arrow), …]

If sum(arrows) > .3 then type goes from green → protectedturn

Issues

- Modulability
    - Since introducing these special cases the function that takes the classification has gotten more and more complicated. Should be more or less the end of special cases since there is not too much more to the traffic light systems to possibly cover
- Tuning parameters
- Hard to mock the classification inputs now, would need significant rework for the random noise generator.
- Need data
- Maybe pull a parameter that says system fps and scale list to be 2seconds of input based on that value (late in implementation)