Cooperative vs Non-Cooperative Autonomous Driving

...

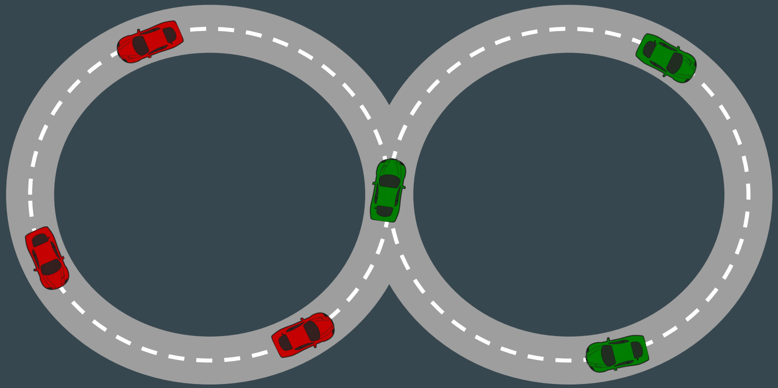
Team A1:
Tito Anammah, Serris Lew, Kylee Santos

Application Area

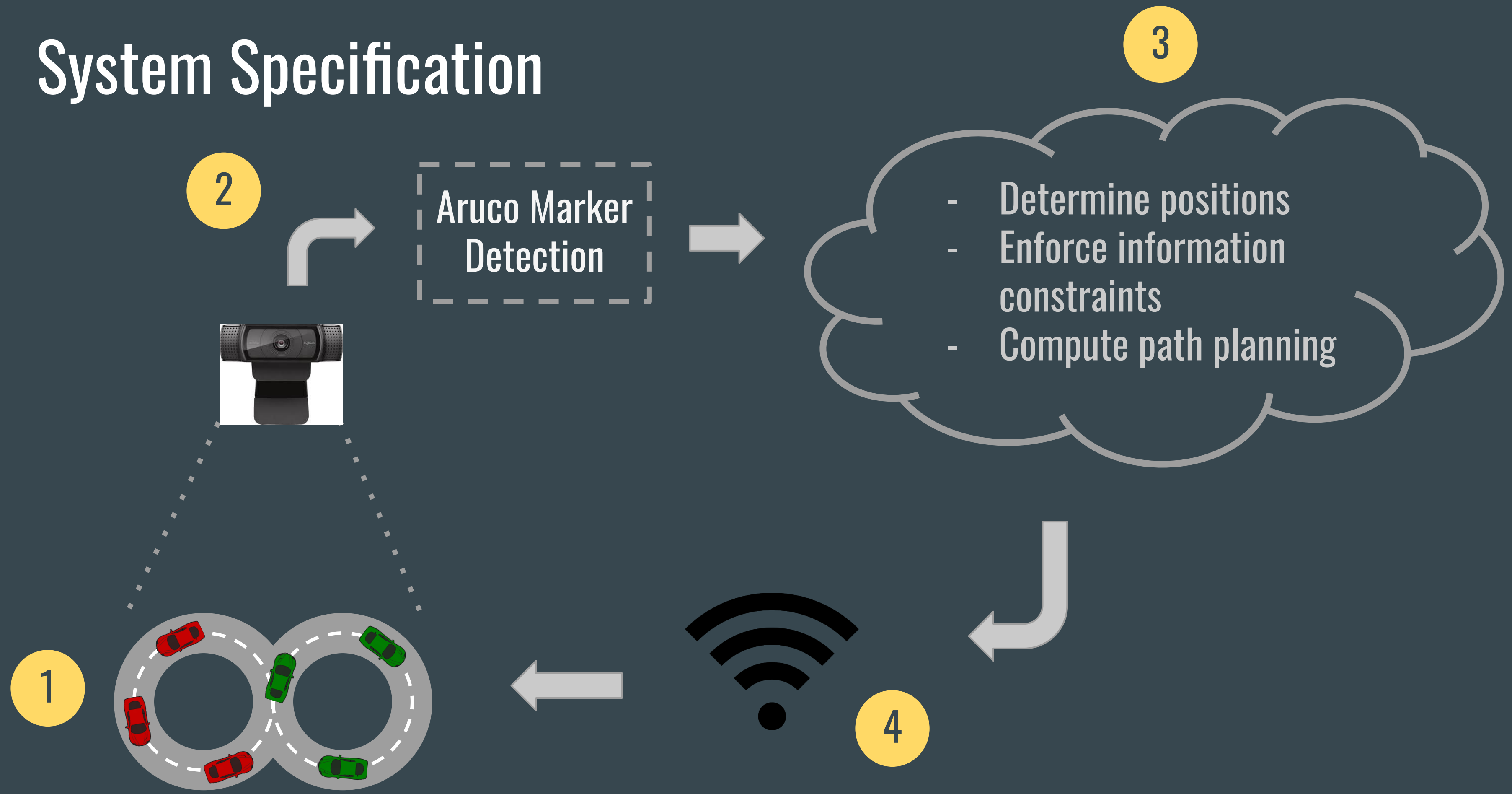
- Autonomous driving will likely be the future framework of transportation
  - But not yet safe or trusted
- Simulate and measure the effects of non-cooperative vs cooperative autonomous driving
  - **Non-cooperative**: optimizing ***individual*** goals from sensing immediate surroundings
  - **Cooperative**: optimizing overall ***system*** goals by communicating with nearby vehicles
- Experiment with 6 cars moving in circles on a figure-8 track
- Compare throughput between approaches
  - Throughput: How many cars pass the center lane of track in certain amount of time

Solution Design Decisions

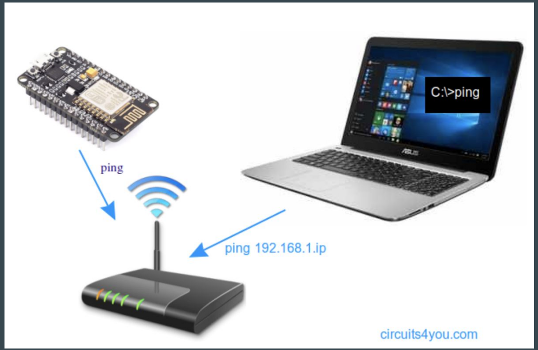
- Figure-8
  - Simplifies path planning
  - Vehicles only change speeds, not direction
- No individual sensors
  - Digresses from overall goal of project
- Global camera system
  - GPS location is imprecise on a small scale
  - Can more accurately detect pose and location of vehicles
- Communication with central server
  - Implement information constraints on server-side
  - Simulate “V2V communication”



System Specification



Robotic Vehicles / Communication 1 4

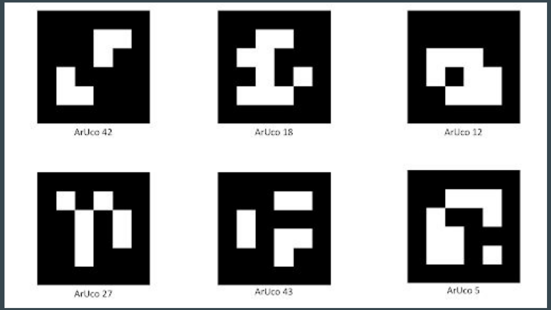


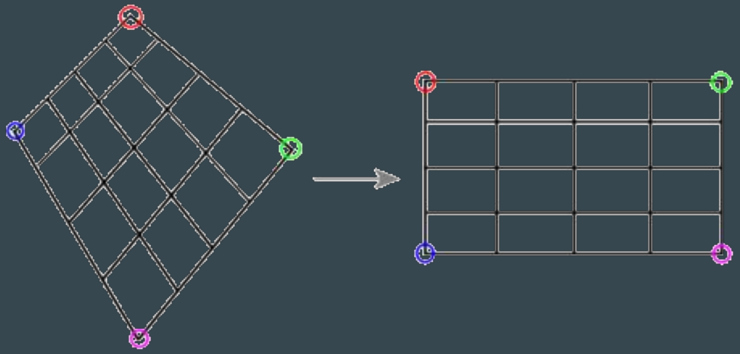
- Robot Rover Chassis Kit
  - Large, flat mount on top for identifiable tag
- L298N Driver
  - Dual H-Bridge: controls speed and direction of motors
  - Compatible with NodeMCU board
- NodeMCU ESP8266 Board
  - Unlike bluetooth, allows multiple vehicles under the same network
  - Cost-effective
- Need to implement:
  - Centralized system between all vehicles and server
  - How data will be sent while minimizing latency

Vehicle Detection

2

- Logitech C920 camera
  - Integrates well with OpenCV
  - 56.5° field of view
- ArUco Marker Detection
  - Integrates well with OpenCV
  - Fast, robust object detection
  - Yolo only detects “standard” objects and vehicles
- Homography
  - Direct overhead camera placement required
  - Preprocess a transformation matrix
  - Map pixels from warped frame for accurate positions

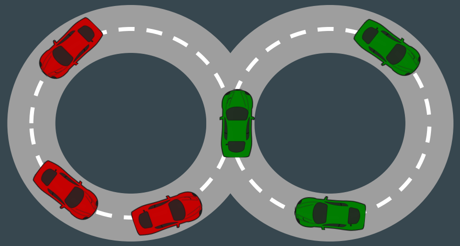




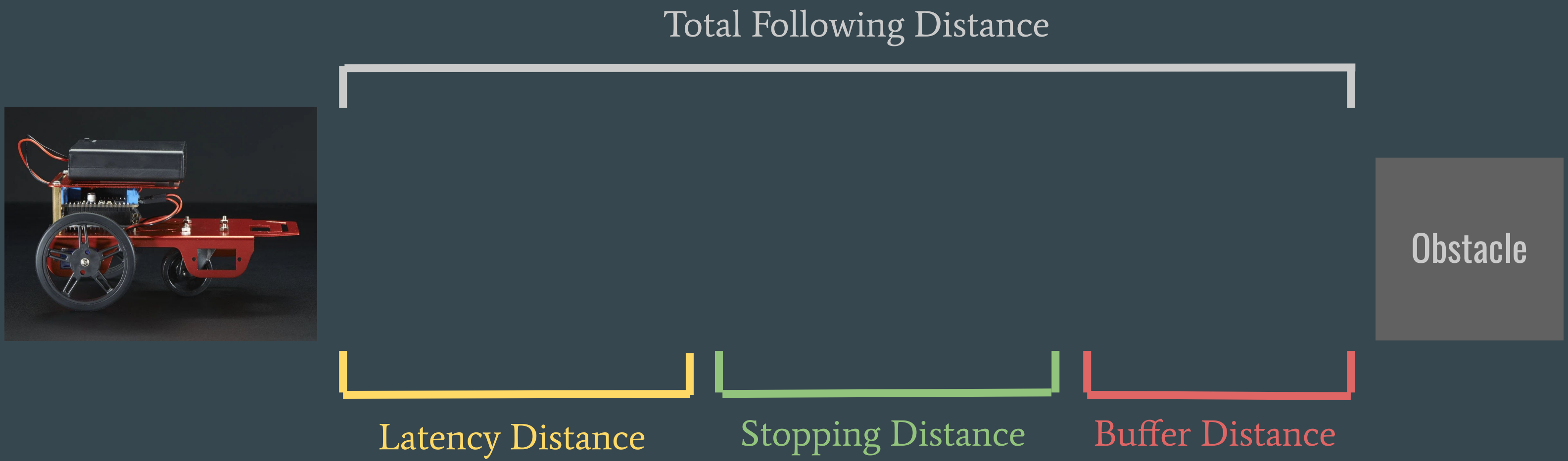
Server Computation 3

- Individual vehicle and track position
  - ArUco detection → Homography matrix → Positions
  - 3 vehicles define properties of circle
- Adjust vehicle path on circular track
- Apply Intelligent Driver Model (non-cooperative)
  - Defines equations for car-following behavior
- Detect impending path collisions (cooperative)
  - Use knowledge of nearby vehicles' paths
- Apply scheduling algorithm (cooperative)
  - Prevent starvation for vehicles waiting to cross center lane

$$a_{\mathrm{IDM}} = \alpha \left[1 - \left(\frac{v}{v_0} \right)^{\delta} - \left(\frac{s^{\star}(v, \Delta v)}{s} \right)^{2} \right]$$



Driver Model



Latency: Distance traveled during ArUco detection, path planning, and communication

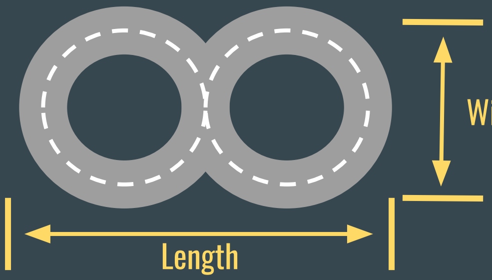
Stopping: Distance traveled between receiving a stop command and coming to a complete stop

Buffer: Distance remaining between stopped vehicle and obstacle

Metrics

Number of cars on a track	3	Latency Distance	10.25 cm
Max Vehicle Speed	50 cm/sec	Stopping Distance	2 cm
Detection Latency	150 ms	Buffer Distance	5 cm
Computation Latency	5 ms	Total Following Distance	17.25 cm
Communication Latency	50 ms	Circumference of one track	96.75 cm

Length of track: 145 cm
Width of track: 80 cm
Height of camera: 107 cm

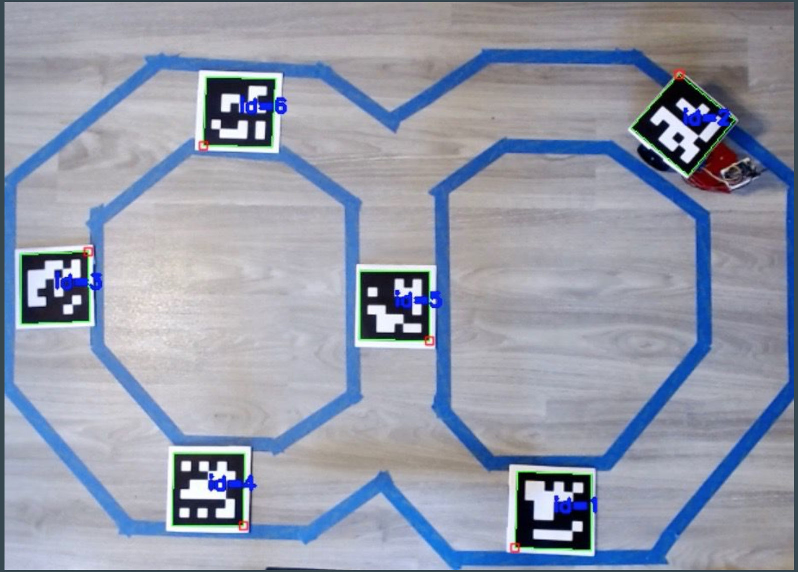


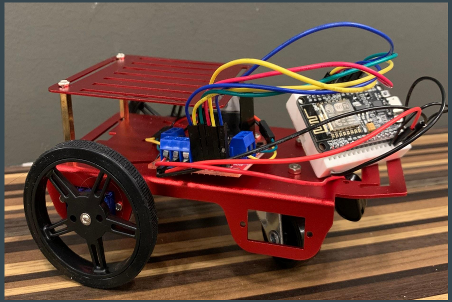
Requirements / Validation

- Video processing computation in **150 ms**
- Path planning computation in **5 ms**
- Communication latency from laptop to vehicles in **50 ms**
- **3 cm** precision in determining vehicle's position
  - Tests marker detection and homography
- **5 cm** deviation from track
- **0** collisions

30% increase in throughput in cooperative vs non-cooperative case

Final Design





Schedule

After testing our design, we made some changes to our implementation

- Discovered better alternatives
  - ArUco Object Detection
- Finalized metrics
  - Shape/size of track
  - Car interaction
- Hardware implementation
  - Car setup