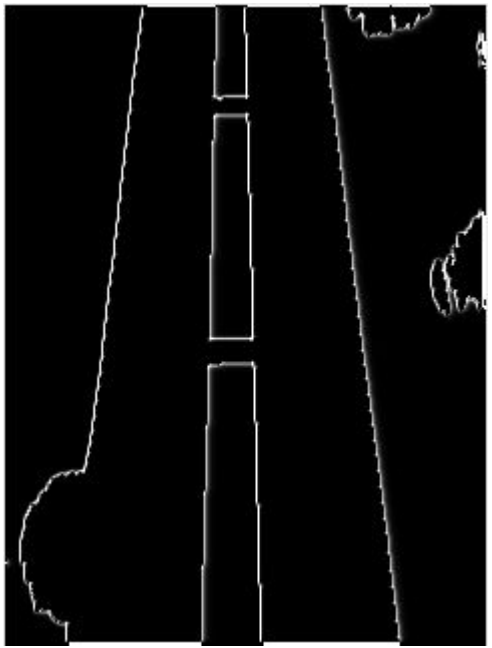
MAE 198 A

Team #3
DonkeyCar with Line Detection

Chris Yin (ECE), Noopur Khachane (ECE),
Alec Schardein (MAE), Aamir Rasheed (CSE/ECE?)

Problem Statement

- Problem:
  - Default DonkeyCar trains on entire camera image
  - Lots of unnecessary information = more resources required + time intensive processing
- Solution:
  - Extract edges of road and train on isolated part of image to reduce processing
  - Using Hough transform, edge detection, etc.
- Deliverable:
  - Modified DonkeyCar code that trains vehicle on lines extracted from camera image



https://stackoverflow.com/questions/11461910/line-detection-angle-detection-with-java

Progress + Planning

Completed/In-Process: Weeks 1-2	Planned Activities: Weeks 3-4

• Aamir
  • RPI loaded with DonkeyCar image
• Alec
  • Plate cut
  • Adaptors + camera mount printed
  • Electronics assembled on car
• Chris
  • Power supply – battery charger configured
  • Electronics assembled on car
• Noopor
  • Power supply – battery charger configured
  • Electronics assembled on car
	• All
	  • Complete Phase 1 car
	  • Help lay down track
	  • Train car using DonkeyCar software to get baseline performance metrics
	  • Implement edge detection algorithm to filter out everything but edges in camera feed
	  • Incorporate edge detection filter into DonkeyCar software + try training vehicle
	  • Research improvements to camera/imaging setup for better resolution of lines

Risks	**Needs**
• Training only on edges may reduce car's ability to respond usefully to other obstacles on the road, e.g. other cars. • Car may also respond more poorly in low-light conditions when edges are more difficult to detect but other visual information that may be available is discarded.	• Understand how to incorporate features from preprocessing the image into neural network • Clarify how lighting conditions will affect edge detection • Identify metrics to quantify and compare modified DonkeyCar performance