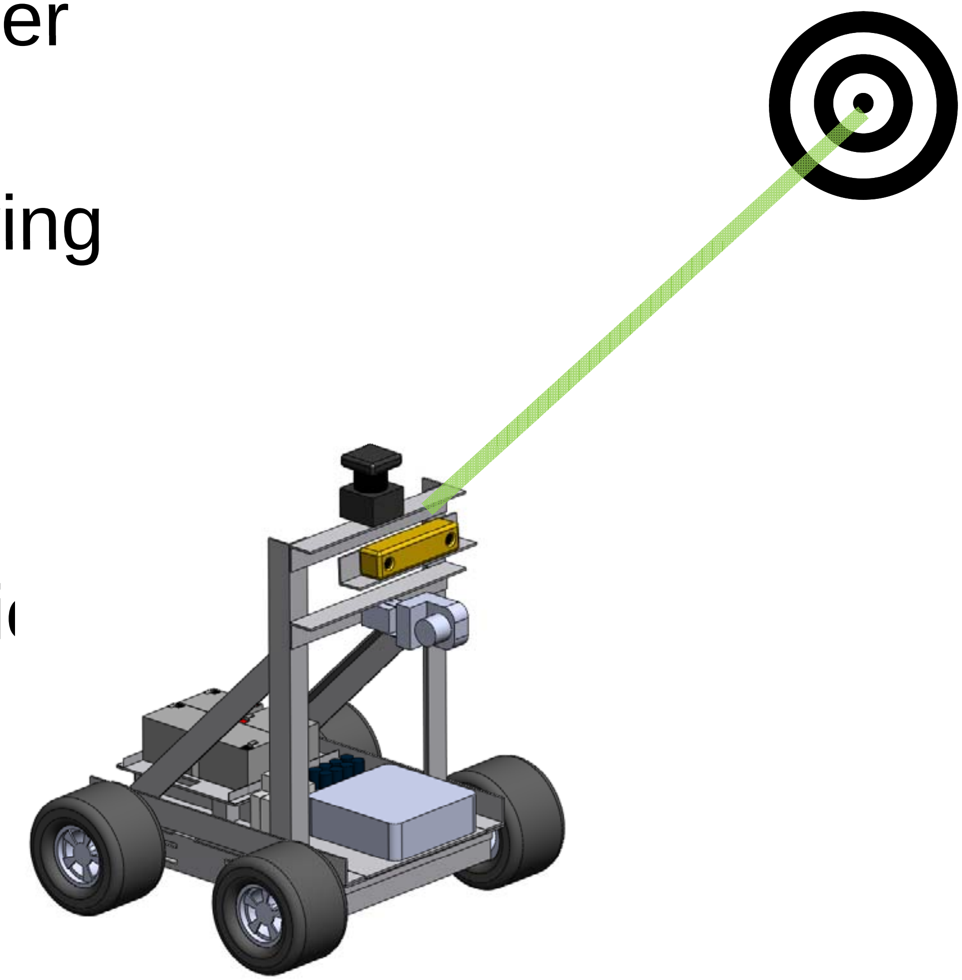
AUTONOMOUS LASER LOCKING SYSTEM

Sunfest 2010
NSF REU program

By Brett Kuprel
Advised by Professor Daniel Lee and Alex Kushleyev

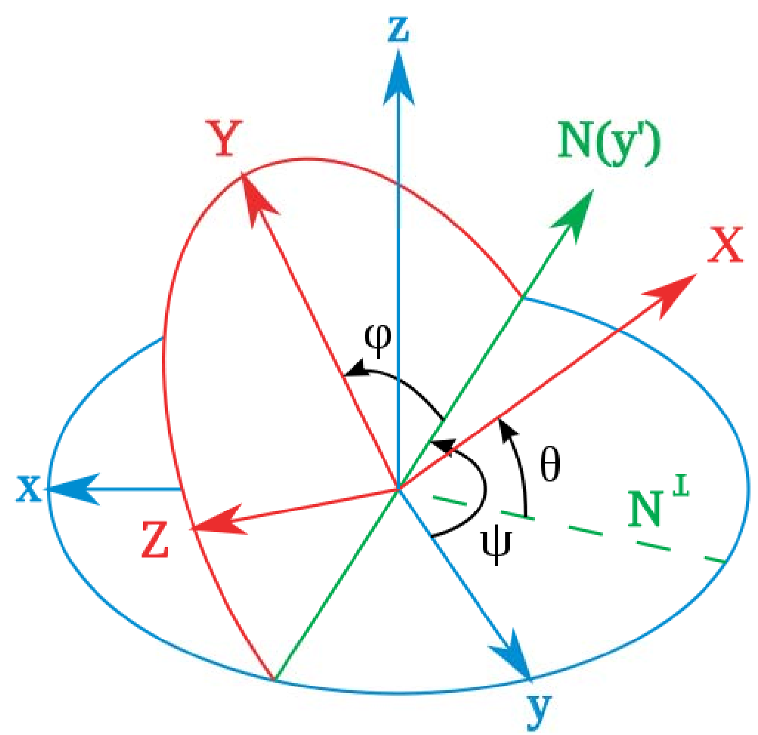
Overview

- Goal is to keep laser pointed at target
- Target can be moving relative to robot
- To be used in the MAGIC 2010 autonomous roboti competition



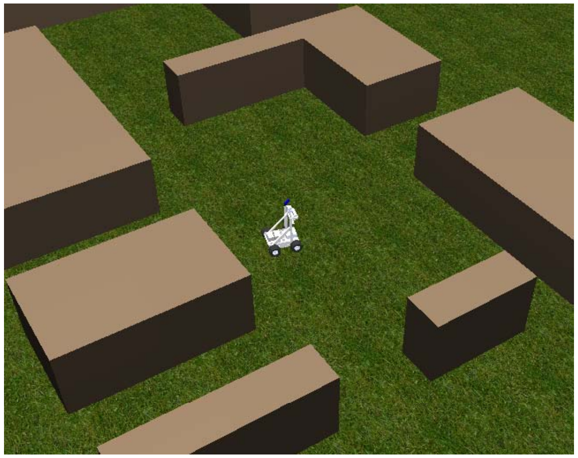
Position and Rotation Estimation

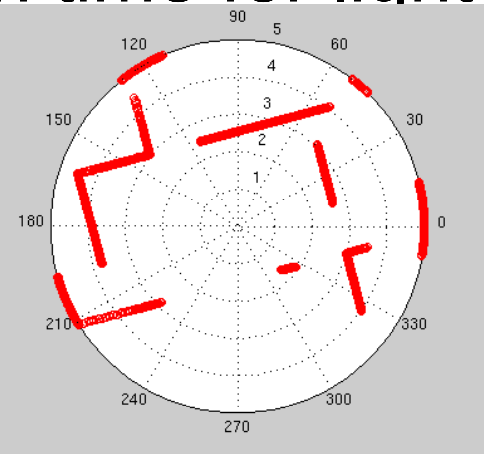
- Required for robot to know where to point laser
- GPS not precise enough, inaccurate around buildings
- Better method is to map the environment with sensors and localize the robot within the map, aka Simultaneous Localization and Mapping (SLAM)
- SLAM analyzes information from LIDAR, gyroscopes, and odometry



LIDAR

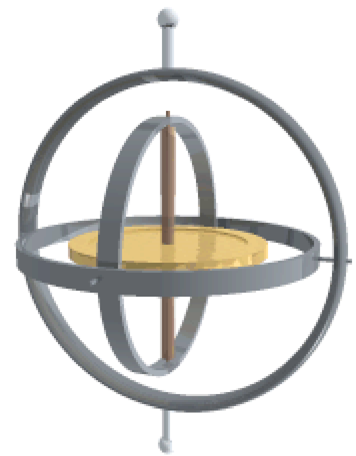
- Like RADAR, but different wavelength
- Emits and receives light at 1080 different angles spread over 270 degrees at a rate of 40 Hz
- Calculates distances based on time for light to return





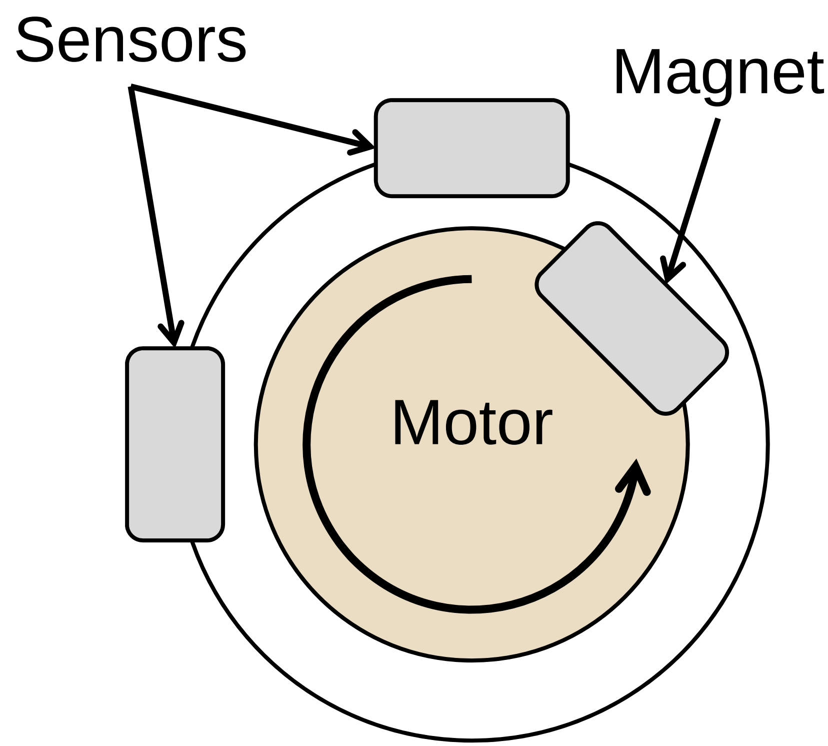
Gyroscopes

- Measures angular velocity
- Integrate to get angular displacement
- $d\theta = \int w \, dt$
- 3 required to measure rotation around any axis in space



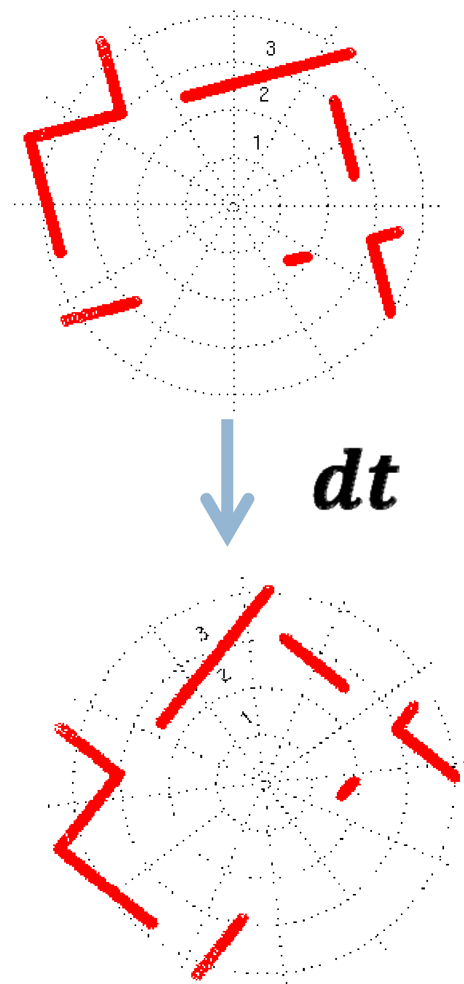
Odometry

- As motor spins, magnets from motor and cage interact causing voltage spikes
- 2 magnets to determine direction (CW or CCW)
- Resulting measurement is angle per time
- When used with wheel radius, distance traveled can be calculated



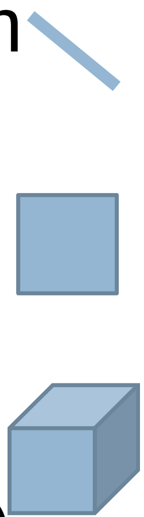
Sensor Synthesis - SLAM

- Together the sensors can be used to create a map of the environment
- Main idea: consecutive lidar maps are pieced together
- Piecing together requires translating and rotating depending on how the robot moved between lidar maps
- Robot motion can be determined by odometry and gyroscopes



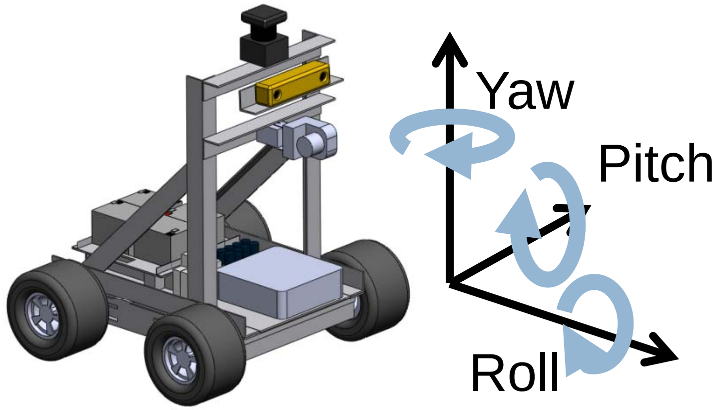
Rotational Freedom

- Position and rotation of robot can be estimated
- Need to be able to point laser at target
- Requires 2 degrees of rotational freedom
  - 0 degrees = line of sight
  - 1 degree = plane of sight (rotated line)
  - 2 degrees = volume of sight (rotated plane)



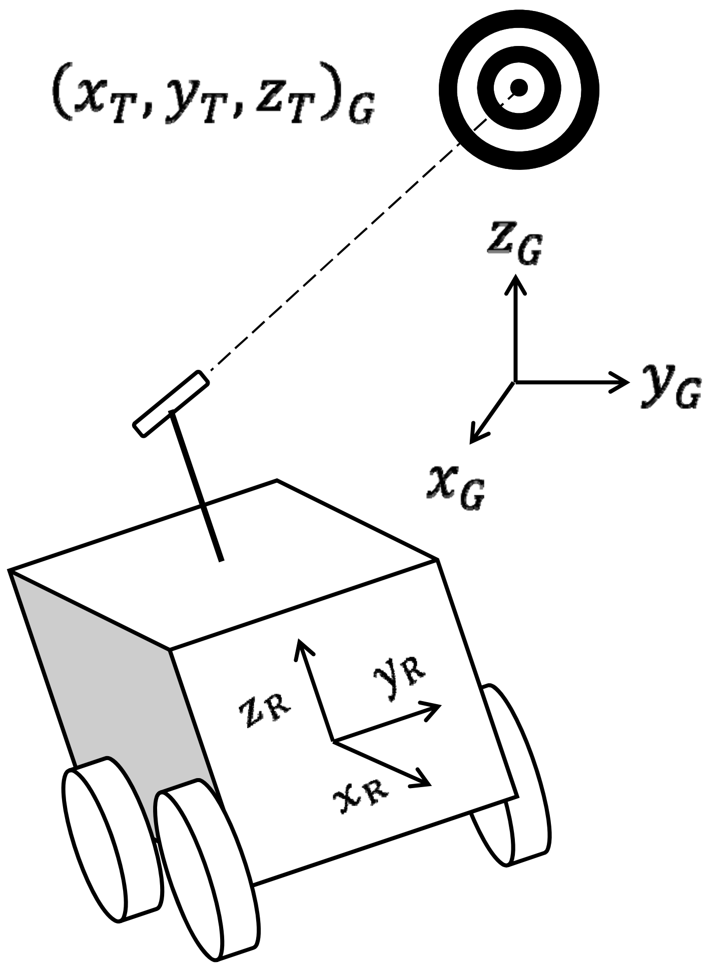
Rotational Freedom

- 2 perpendicular servos
- Each servo has 300 degrees of freedom
- Servos control pitch and yaw
- Resulting field of view is all points in space except a 60 degree pyramid



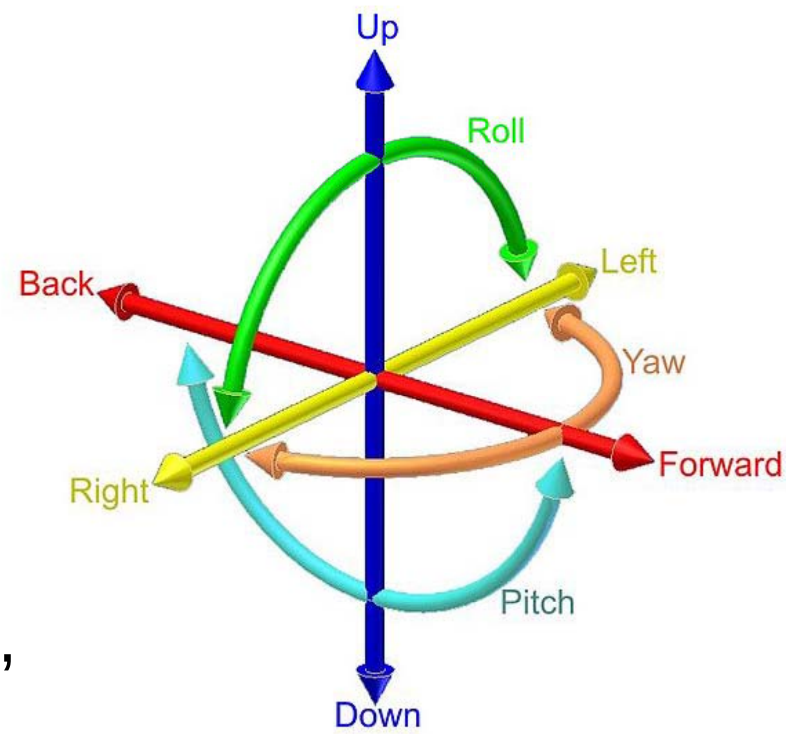
Robot Coordinate System

- Target given in global coordinates
- Use homogenous transformation matrix to convert global coordinates into robot coordinates



Required for Transformation Matrix

- Finding robot coordinates requires robot's position and rotation in 6 degrees of freedom to be known
- 6 degrees of freedom:
  - Translation in 3 perpendicular axes (eg x, y, z)
  - Rotation around 3 perpendicular axes (eg roll, pitch, yaw)



Using Transformation Matrix

- Robot coordinates can by found by simply multiplying by transformation matrix
- $(x_T, y_T, z_T)_R = T_{G\rightarrow R} * (x_T, y_T, z_T)_G$
- $T_{G\rightarrow R} = \begin{bmatrix} \text{Trans(RobotPosition)} * \cdots \\ \text{Rot}_Z(\text{yaw}) * \cdots \\ \text{Rot}_Y(\text{pitch}) * \cdots \\ \text{Rot}_X(\text{roll}) * \cdots \\ \text{Trans(LaserOffest)} \end{bmatrix}^{-1} = \cdots$

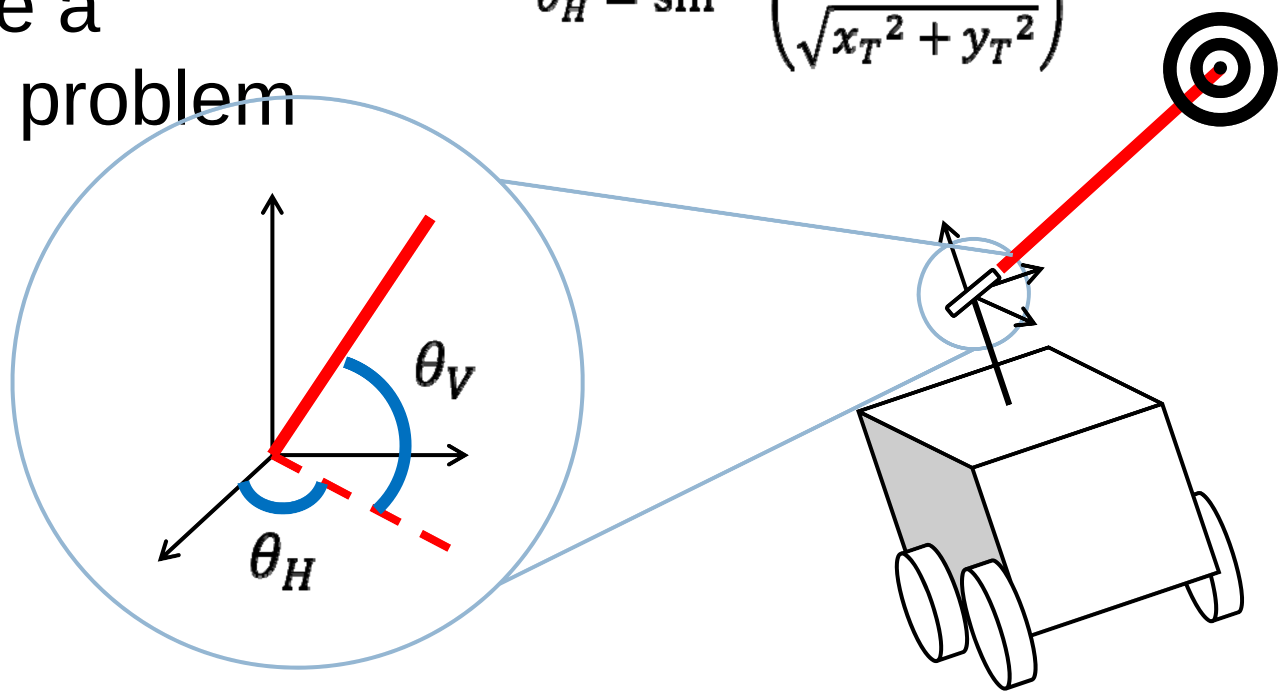
$$\left(\begin{bmatrix} 1 & 0 & 0 & x_R \\ 0 & 1 & 0 & y_R \\ 0 & 0 & 1 & z_R \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(yaw) & -\sin(yaw) & 0 & 0 \\ \sin(yaw) & \cos(yaw) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(pitch) & 0 & \sin(pitch) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(pitch) & 0 & \cos(pitch) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(roll) & -\sin(roll) & 0 \\ 0 & \sin(roll) & \cos(roll) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & x_L \\ 0 & 1 & 0 & y_L \\ 0 & 0 & 1 & z_L \\ 0 & 0 & 0 & 1 \end{bmatrix}\right)^{-1}$$

Calculating Angles of Servos

- Once target is known in robot coordinates, servo angles are a geometry problem

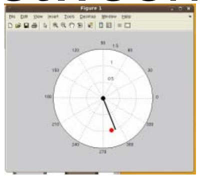
$$\theta_V = \sin^{-1}\left(\frac{-z_T}{\sqrt{x_T + {y_T}^2 + {z_T}^2}}\right)$$

$$\theta_H = \sin^{-1}\left(\frac{y_T}{\sqrt{{x_T}^2 + {y_T}^2}}\right)$$

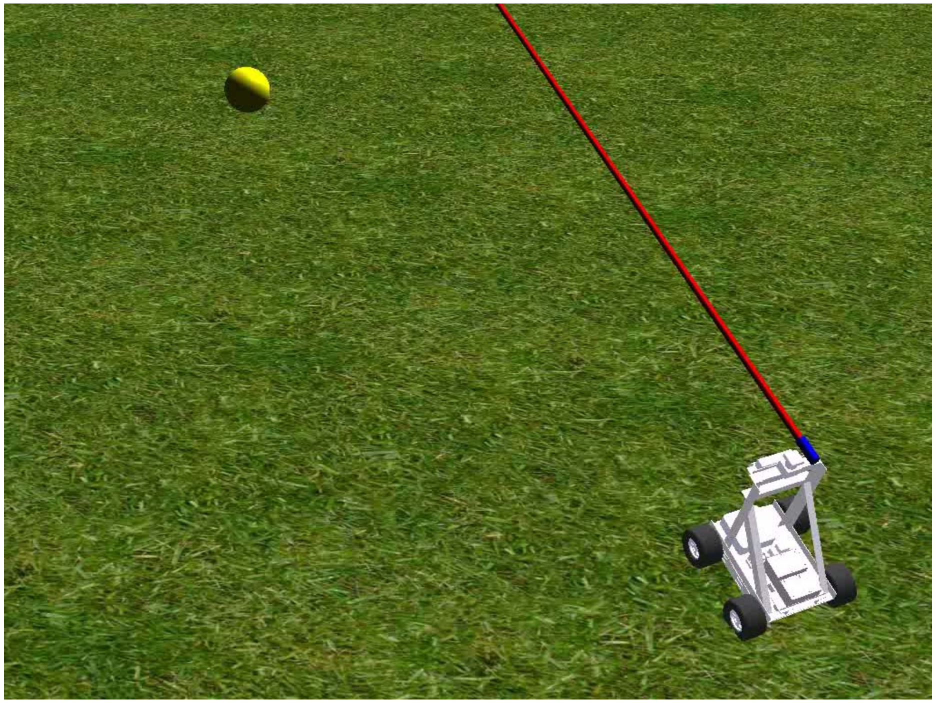


Continuous Servo Motion

- Sending target angle commands at discrete time intervals results in unsteady rotation of the servos
- Solution is to send angular velocity commands instead
- Used a PID controller to determine desired angular velocity
  - Velocity = sum of proportional, integral, and derivative of error
  - Error is difference between current angle and desired angle



Simulation



Actual Robot

Acknowledgements

- Professor Daniel Lee
- Alex Kushleyev
- Dr Jan Van der Spiegel
- National Science Foundation