Ballbot: Omnidirectional Robot

Adam Woo '18 Graham Goodwin '18 Chloe Desjardins '18 | Advised by Prof. Huang and Prof. Mertens
Department of Engineering, Trinity College

Introduction

In 2008, a team of graduate researchers at Carnegie Mellon University developed the first dynamically stable, omnidirectional robot. The goal was to develop mobile robots that are dynamically agile, highly maneuverable, and capable of graceful motion. To accomplish this, a new approach to locomotion was required. In order to meet these design requirements, the team of researchers decided to create an intrinsically unstable robot which uses a ball as essentially a single, omnidirectional wheel. This allows the base of the robot to be much narrower relative to multi-wheeled designs. A slender robot design, in combination with the agile motion, creates a robot that is highly maneuverable in crowded environments. This new concept introduced the family of robotics to "ballbots." The purpose of this project is to explore ballbot design, and to develop our own single-contact, omnidirectional robot.

Objective

To develop a single-contact, self balancing robot that is capable of omnidirectional movement.

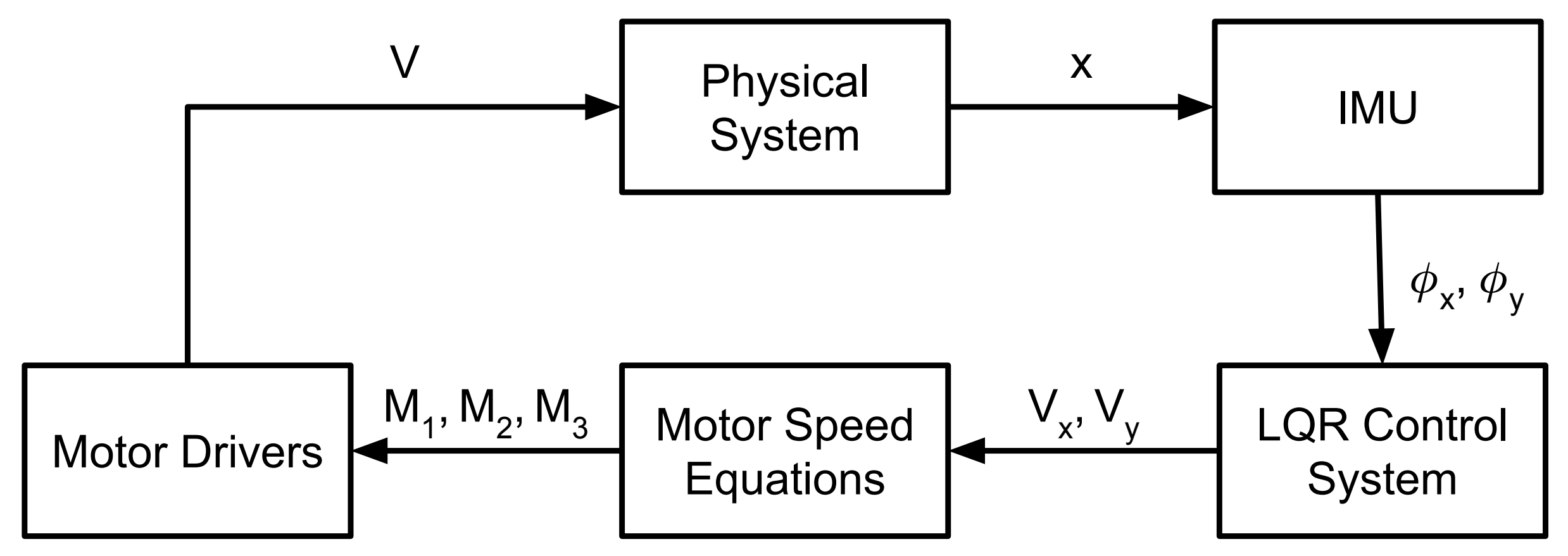
Specific Goals:

- Must be tall enough to interact with a standing human.
- Needs to be capable of performing interference corrections.

Estimating Angular Orientation

An Inertial Measurement Unit (IMU) is a combination of an accelerometer, gyroscope, and/or magnetometer which measure angular acceleration, angular velocity, and compass orientation respectively. These measurements can be used to estimate the angular position (ϕ) of our robot relative to vertical. In order to do this, the measurement signals must be passed through a filter to reduce noise and then computed to acquire the angular position. In this case, a quaternion based complementary filter was implemented, which fuses estimation in quaternion form from gyroscope data, with accelerometer data in the form of a delta quaternion to eliminate noise for pitch and roll. Yaw was ignored in this case, because rotation about the Z axis did not impact the robots stability

System Block Diagram



Acknowledgements

Thank you to…

Prof. Huang for your guidance with this project and its development.

Prof. Mertens for your advice and critiques that kept the project moving.

Andrew Muslin for your help with the practical aspects of design and implementation.

Nancy Fleming for keeping the department and students organized and aware.

References

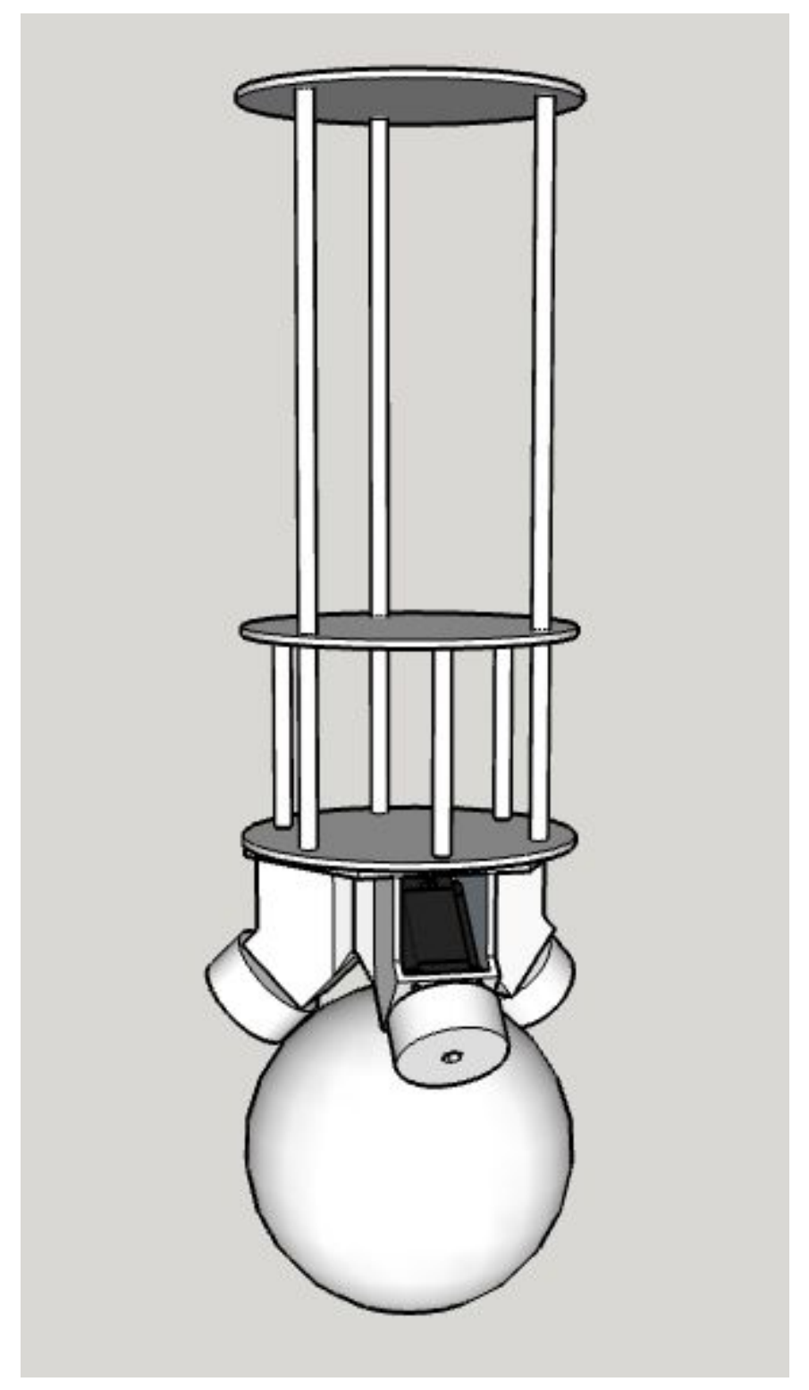
Roberto G. Valenti, Ivan Dryanovski, Jizhong Xiao. 2015. Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs. Sensors 2015, 15(8), 19302-19330.

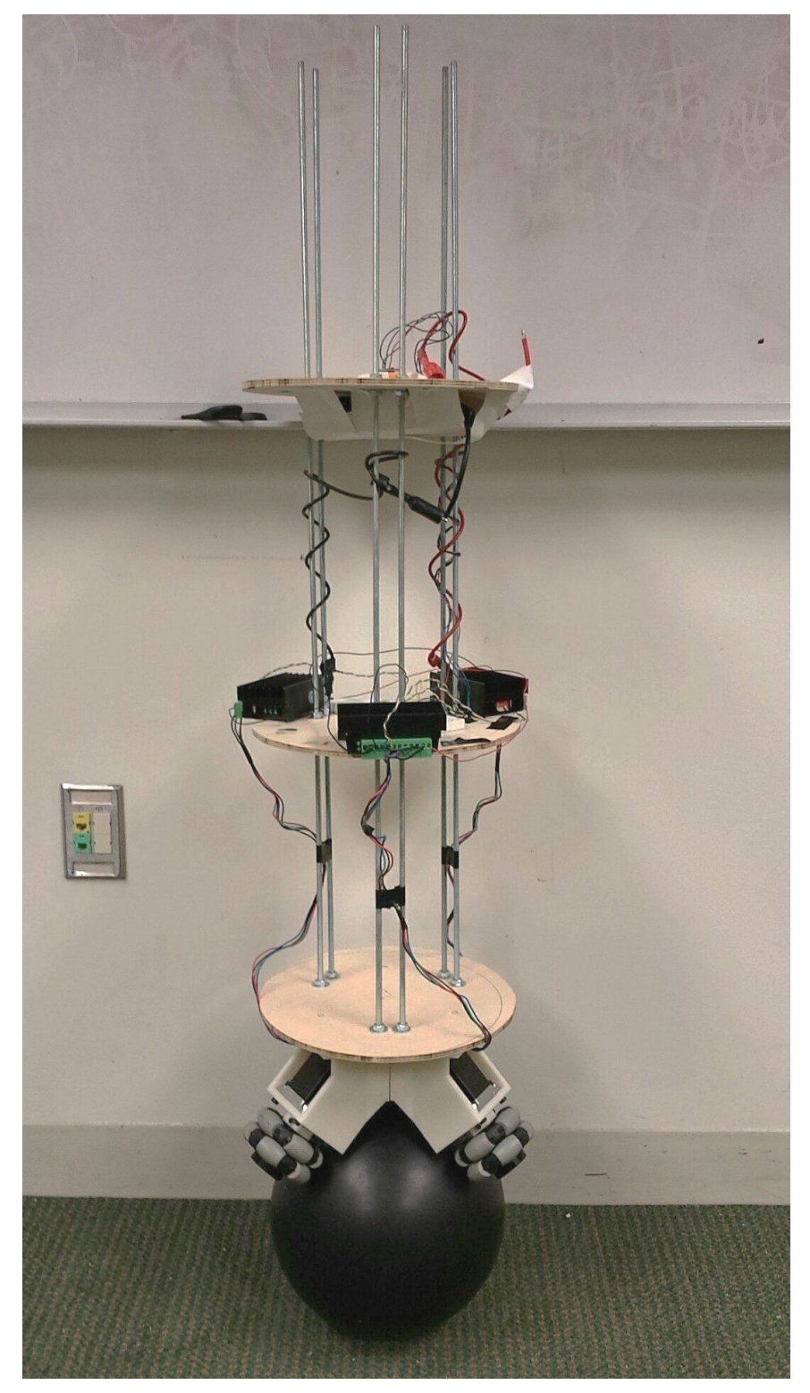
Control Tutorials for Matlab and Simulink. 2017. University of Michigan. [2017; January 3, 2018].

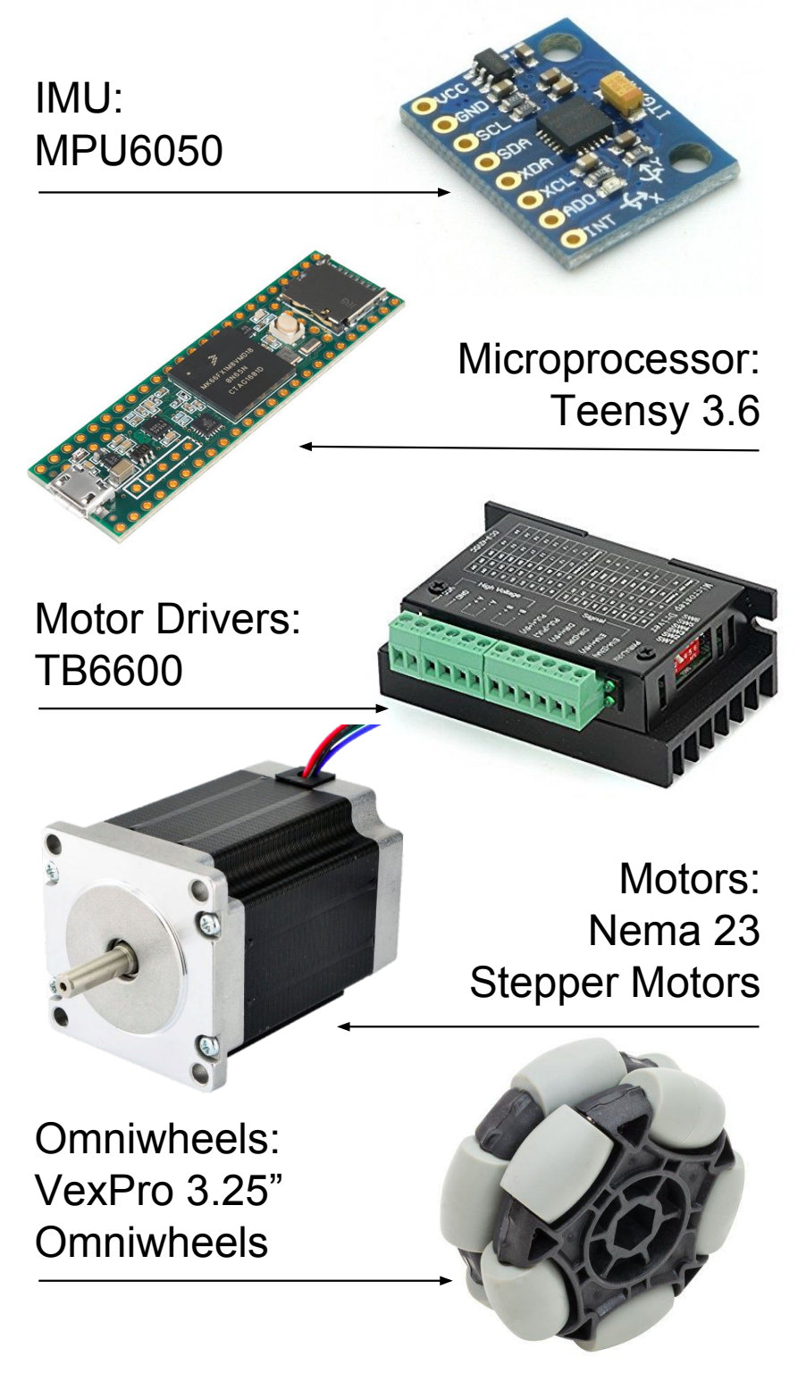
Implementation

In essence, a ballbot is an omnidirectional inverted pendulum. This means that the ballbot can be broken down into two, 2D inverted pendulums that travel along the X and Y axis respectively. The angle of the robot along these axes at any given time can be used to calculate the necessary velocity required to keep each 2D pendulum, and therefore the overall system, upright.

The X and Y components can be summed to a desired resulting vector. This resulting vector represents how the ball must be moved to maintain the robot's verticality in 3D space.

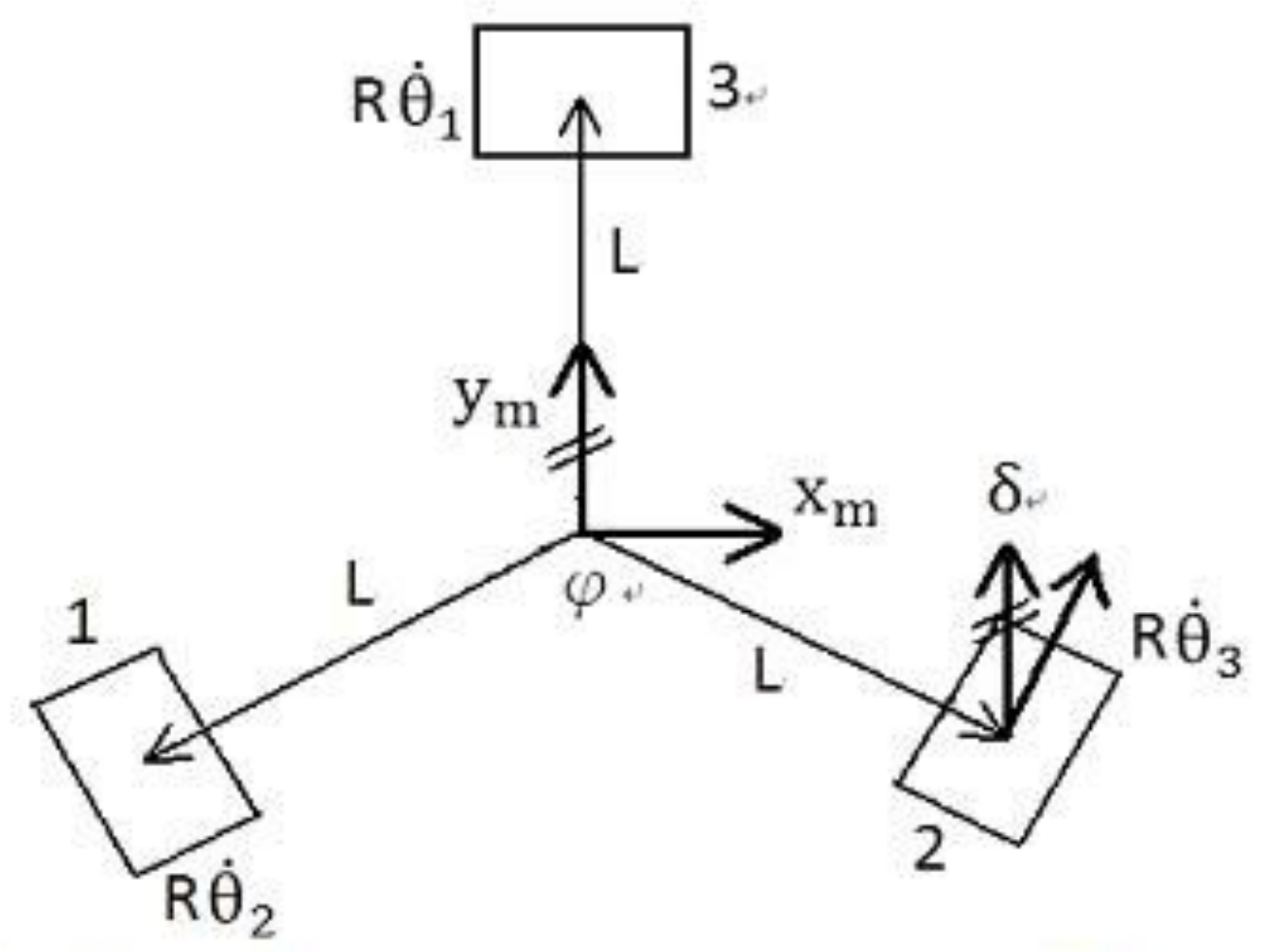


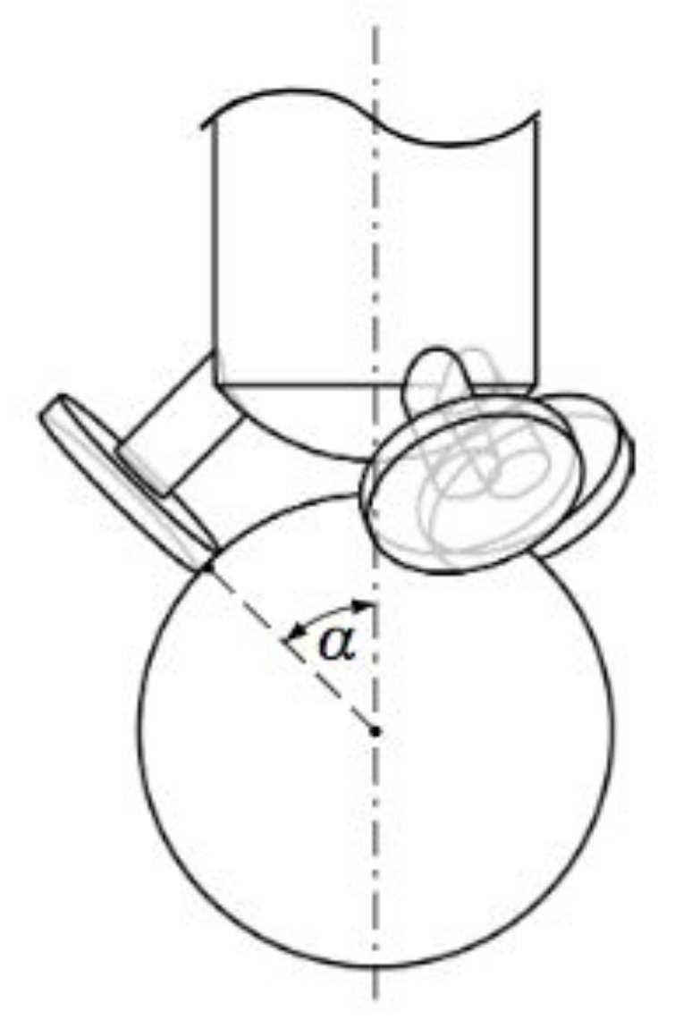



IMU: MPU6050
Microprocessor: Teensy 3.6
Motor Drivers: TB6600
Motors: Nema 23 Stepper Motors
Omniwheels: VexPro 3.25" Omniwheels

Motor Speed Kinematics

The desired velocity components need to be converted into motor speeds that will give the ball the desired motion. Having set the wheels at 120° from each other, and at a 45° angle from vertical, the required motor speeds to give the desired ball velocity is represented as the following:





$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} = \sin(45^\circ) \begin{bmatrix} -\cos(60^\circ) & \sin(60^\circ) \\ -\cos(60^\circ) & -\sin(60^\circ) \\ 1 & 0 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix}$$

Control System Design

Equations of motion for the system were derived and then linearized about an upright position to acquire the state space representation seen below. Four system states were used: linear position (x), linear velocity (dx/dt), angle from vertical (ϕ), and angular velocity (dϕ/dt). The inputs to the system are the forces applied by the motors.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-b}{M} & \frac{-gm}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-b}{Ml} & \frac{-g(M+m)}{Ml} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml} \end{bmatrix} u$$

$$y = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} u$$

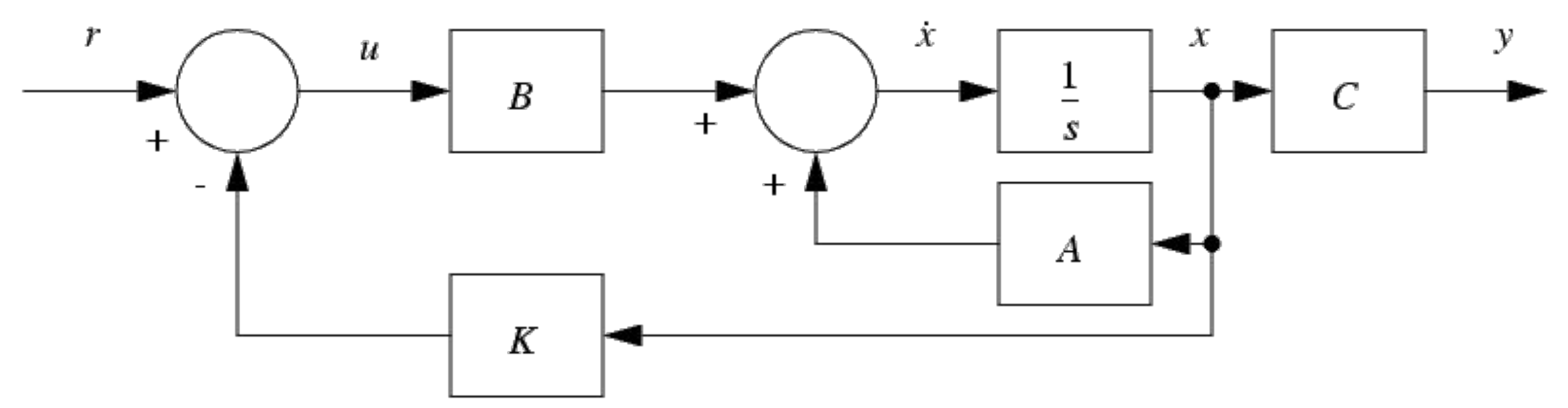
A state space controller utilizing a Linear Quadratic Regulator (LQR) was implemented to optimize the system to maintain specified states. The known states are the desired angles from the vertical (ϕ) in the z-x and z-y planes. The feedback from the IMU depends on the difference between the desired angle and the actual angles. This feedback returns the required linear x and y velocities of the ball to stabilize the system.

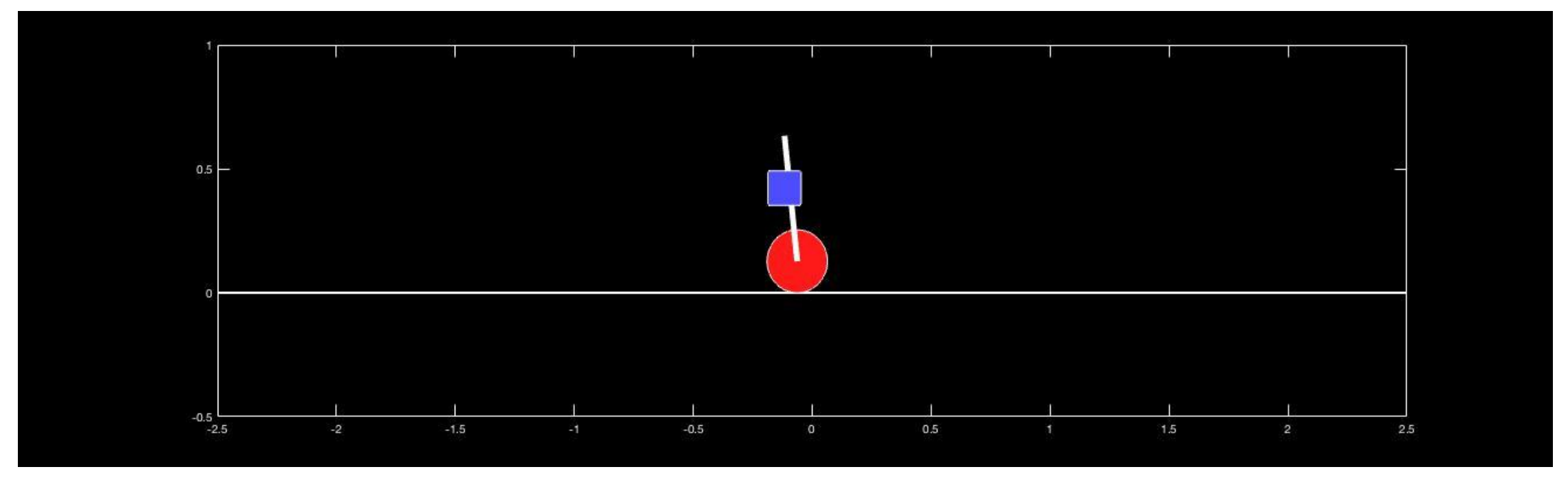
The gain K is optimized for a given Q matrix, where each state is given a certain weight, and R value, which dictates how much energy cost should be minimized compared to the other states. And example Q and R can be seen below:

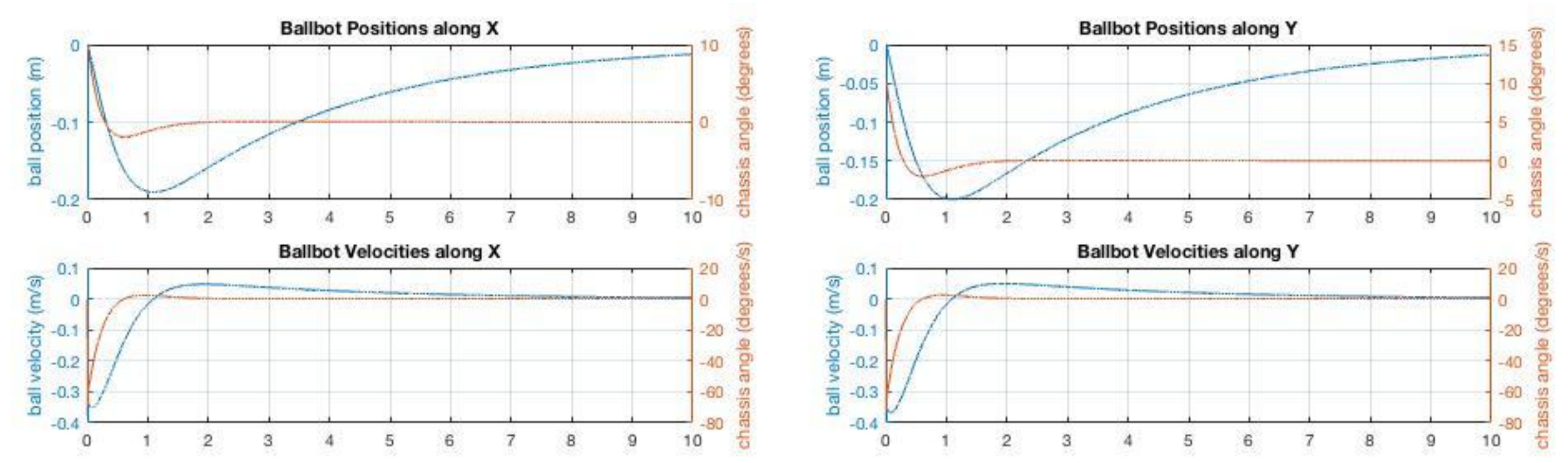
$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \qquad R = .01$$

$$K = \begin{bmatrix} -8.5354 & -33.2312 & 262.1955 & 39.7793 \end{bmatrix}$$

The system was designed and simulated in Matlab for each planar model as seen below:






Ballbot Positions along X
Ballbot Positions along Y
Ballbot Velocities along X
Ballbot Velocities along Y