TIPSY (Tip Inhibiting and Positioning SYstem)

A self balancing inverted pendulum robot controlled by voice commands

Dominik Martinez, Olek Peraire

6.111 Fall 2018 Final Project Presentation

Project Overview:

Motivation:

- Simplified model for sophisticated autonomous systems that retains some key challenges
- Illustrate use of FPGAs -- well suited for mobile applications given low weight and power consumption

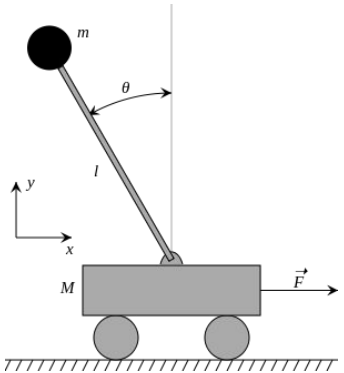
Objectives:

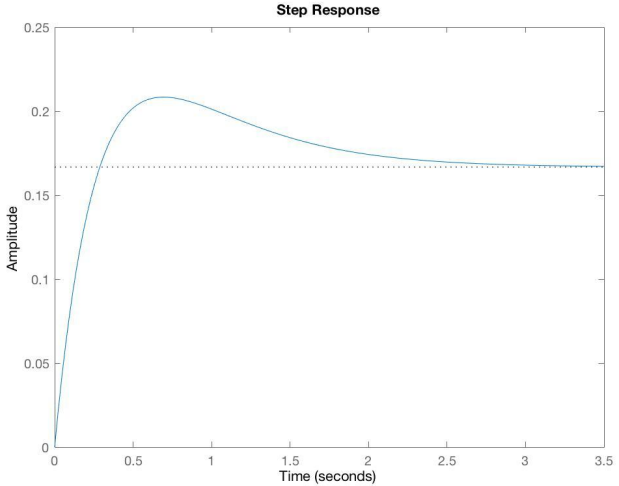
- Implement PID control algorithm on mobile FPGA to maintain attitude and enable motion
- Respond to motion voice commands -- processed on base FPGA
- Untethered -- Transmit commands between base FPGA and mobile FPGA using Bluetooth

Controls

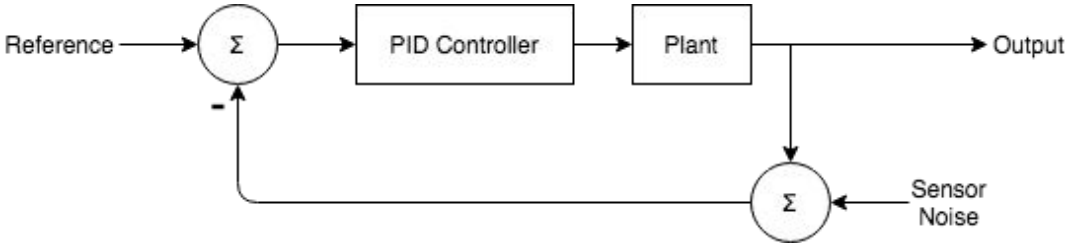
Parts of a PID Controller:

- Proportional
- Integral
- Derivative





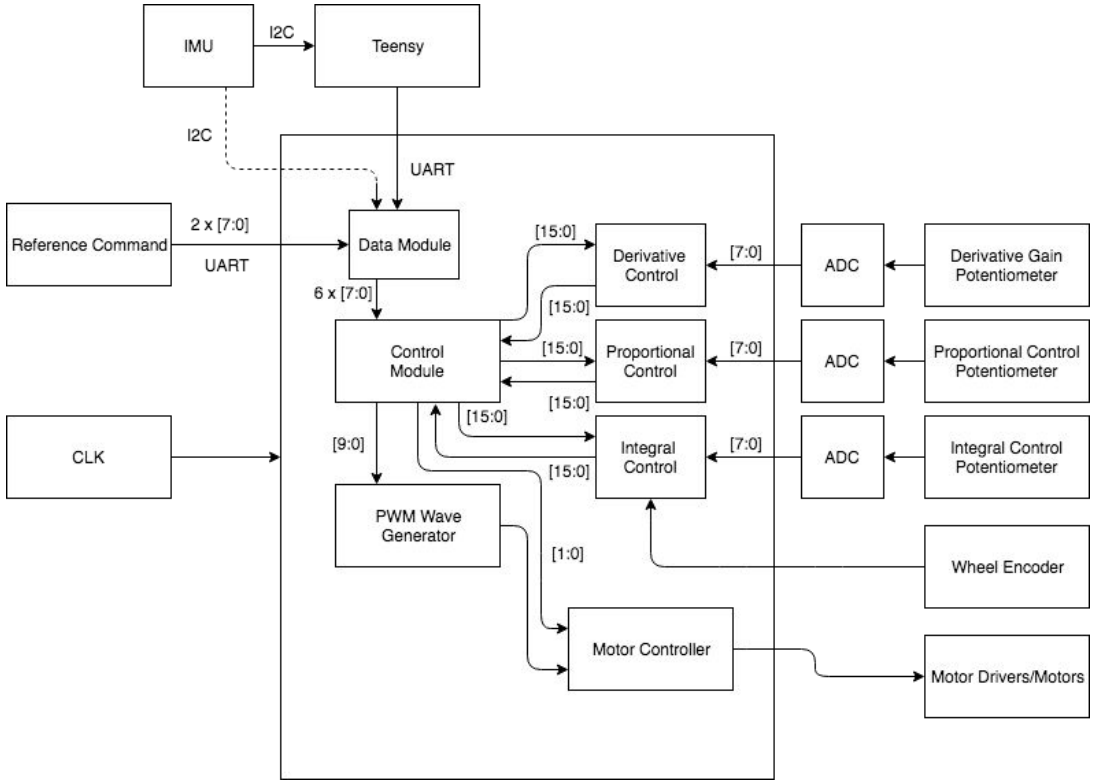
Allow us to alter response to a step input



Mobile FPGA Block Diagram

Notes:

- Commands and readings will be 16 bits
- Data module will be similar to lab 5c
- Motor controller module will be instantiated twice (one for each motor)
- CMOD A7-35T FPGA at 12 MHz
- MPU-9250 IMU
- MCP-3008 ADC (via SPI at 25kHz)



Control Modules and Details

Control Module:

- Will take in reference command as well as IMU readings from data module, pass them on to PID modules, and will then use feedback control to adjust PWM duty cycle as well as motor direction

Proportional Control Module:

- Receives current IMU readings as well as reference angle, calculates offset and multiples by gain

$$\theta_y[n] = \beta(\theta_y[n-1] + Tg_y[n-1]) + (1-\beta)\tan^{-1}\left(\frac{a_z[n-1]}{a_x[n-1]}\right)$$

- Returns proportional feedback command

Derivative Control Module:

- Receives current IMU readings but only uses gyro readings and derivative gain to return derivative feedback control

Control Modules and Details (cont.)

Integral Control Module: (stretch goal)

- Uses wheel encoder to track drift over time as well as integral gain to return integral feedback command to controller module

Motor Controller Module:

- Receives PWM wave as well as motor direction for each motor to pass onto the drivers that will run the motors

Control Challenges and Solutions

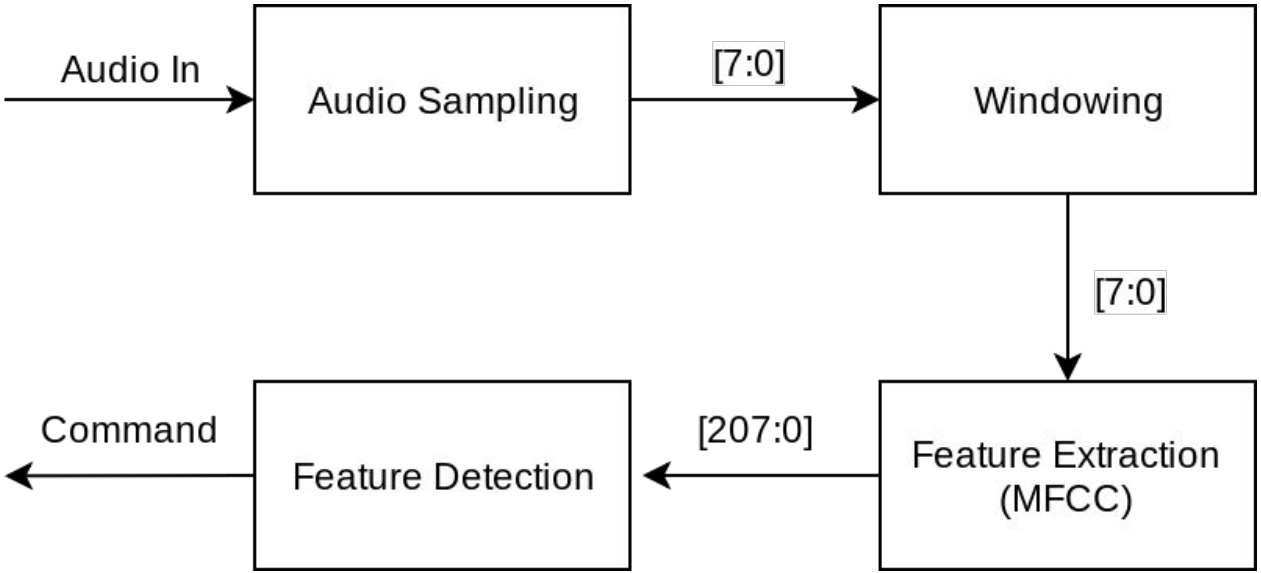
- Drift over time
  - Fixed using a wheel encoder
- Turning and balancing at the same time
  - Blending of turning and balancing commands
- Trigonometric functions in calculating angle
  - Will use small angle approximation
- Derivative control in forward loop versus using angular velocity directly from gyro

Voice Commands

Overview

- Voice controlled devices are becoming more ubiquitous
- Want to be able to give the robot voice commands such as “Go Forward”, “Turn Right”, “Stop”, etc.
- We send the command to the robot via Bluetooth from another FPGA

Base FPGA Block Diagram

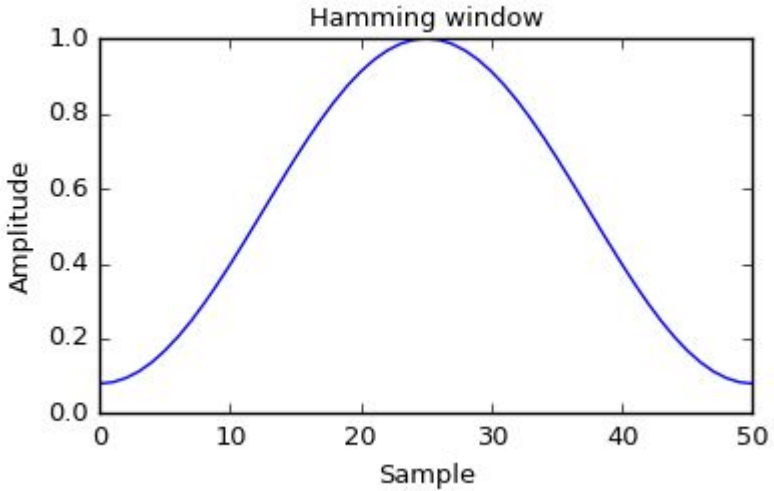


Audio Sampling

- Input audio from ADC at 48kHz
- Low-pass filter and downsample to 16kHz
- Store samples in 10ms chunks in BRAM

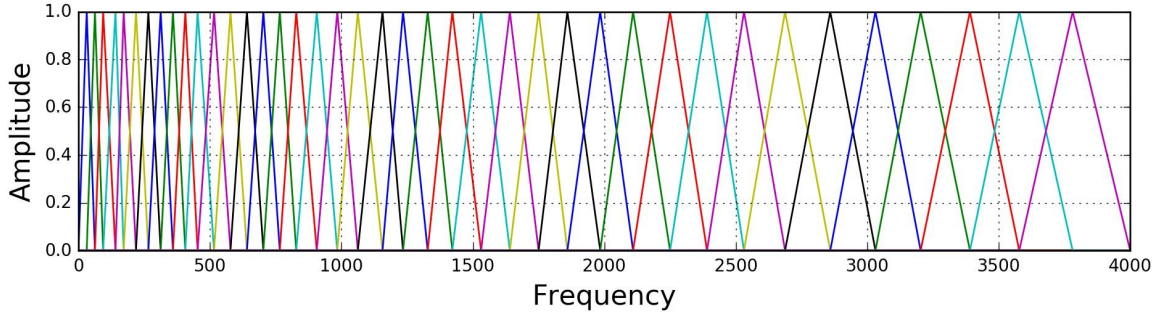
Windowing

- When we break the audio signal into 10ms chunks, we introduce discontinuities
- Since we'll be taking the Fourier transform of these samples, we need to apply a Hamming window to the samples to avoid high-frequency noise



Mel-frequency cepstral coefficients (MFCCs)

- Take the Fourier transform of the windowed audio signal and derive power spectrum
- Apply 26 triangular filters to the power spectrum
- Take the logs of each of the powers
- Take the discrete cosine transform of each of the powers to generate 26 coefficients, this is our feature vector for this sample



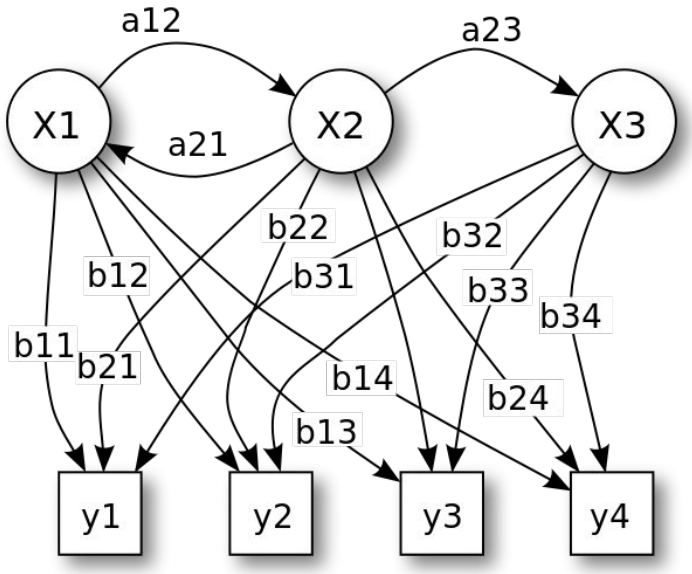
Feature Detection

Simple

- Prerecord voice commands and store their feature vectors in ROM
- Continuously compare all stored feature vectors with live calculated feature vectors via the angle between feature vectors

More Complex

- Use a Hidden Markov Model (HMM) to calculate what command is being triggered



Voice Command Challenges and Solutions

- Fourier transforms are a non-trivial thing to implement
  - Labkit's FPGA has cores that implement a FFT
- With the microphone continuously on, it may be difficult to detect the beginning of speech
  - Ideally continuously comparing the trained feature vector should allow us to properly sync the beginning of speech, alternatively, we could use a button to trigger recording on the device
- Noise could interfere with speech
  - After the audio sampling module we could insert a filter to try and remove noise, alternatively, we get a better microphone

Goals

Baseline Goals:

- Proportional feedback to help stabilize the robot
- Establish communication between both FPGAs via Bluetooth
- Respond to simple one word commands: “Forwards”, “Backwards”, etc.

Expected Goals:

- Add derivative feedback control to help stabilize the robot
- Allow blending of turns and stabilization to rotate the robot
- Respond to multi-word commands: “Go forwards”, “Turn right”, etc.

Stretch Goals:

- Add Integral feedback control and wheel encoders to track drift over time
- Read I2C from IMU directly to the FPGA to avoid using the Teensy
- Take continuous commands

Timeline

Nov. 9-16:

- Proper reading of IMU data, PWM wave generation and motor control, proportional feedback module implemented, audio processing and FFT implementation

Nov. 16-30:

- Derivative control and turning implemented, ability to communicate between FPGAs, proper voice recognition

Nov. 30 - Dec. 7:

- Ensure proper communication between FPGAs, work on stretch goals, integral control

Dec 7-10: Finishing touches before presentation

Questions?