Team Exo

#20F09

Due: 2/12/21

Hardware Review Memo

Introduction:

Our team was tasked with the challenge of creating a testing bench to test the robot actuators. To test the actuators, the team must learn how to communicate with the actuator and its CAN bus protocol. This is done by using Serial UART and CAN 2.0 protocols in Arduino and CAN. The team uses a Teensy 3.6/4.1 Arduino board to communicate between the two protocols so the team can make a code to make the AK80-9KV100 motor rotate.

The actuator has an integrated MIT Mini Cheetah controller and the team's goal is to make it move. After that is achieved then the team will make a controller for the mini actuators that the client has in his lab. These mini actuators do not have controllers built into them so the team will have to code a current controller into them themselves. If the team meets these requirements, then additional tasks will be added on as per client requests.

Summary:

To be able to mount the motor onto the 80/20 aluminum frame a mount needed to be made. The team designed a prototype mount with SolidWorks. The part file was then exported and then 3D printed. The first iteration of the model had two main issues. The first issue was that the holes to mount to attach the motor to the mount, were too small. The second issue with the rails on the mount were too large to fit onto a spare piece of 80/20 extrusion the team had. For the second iteration of the mount, all the holes lined up and were the correct size for the motor to attach to. The only issue was that the rails were still slightly too large.

A large part of the project was programming the arduino nano 33 BLE to communicate with the motor. This program is under development and can currently control the motors position. Angular velocity and torque control must be developed/debugged. Setup() and some initialization code may be found to the left. The actuator is configured using TeraTerm and

```
//Library
#include "Serial_CAN_Nano.h"
//CAN instance
Serial_CAN can;

long unsigned int can_id = 0x1;
//Data buffer for CAN messages
unsigned char data[8] = {1,2,3,4,5,6,7,8};
int command = 0;
bool motor_on = false;

void setup() {
  Serial.begin(9600);
  can.begin(57600);
  while (!Serial) {};
  delay(500);
  //can.baudRate('4'); //115200
  //can.canRate('18'); //1Mbps
  Serial.println("Begin!");
  sendZero();
  ExitMotorMode();
}
```

this is where the CAN and Serial transmission Rates are set (1Mbps and 57600 baud respectively). Currently the program is being controlled by user input collected from the serial monitor, this will likely remain the same with some altered character bindings. The motor specific functions, which may be found in the source on Github, were modified from Skyentific's code at https://www.youtube.com/watch?v=HzY9vzgPZkA. The program uses a modified version of the Seria_CAN_Aduino library from SeeedStudios, found at https://github.com/Longan-Labs/Serial_CAN_Arduino.

Future:

Before the next Hardware Review, the team will work on refining the code so the position, angular velocity, and torque control can work more efficiently. The mounts that are being 3D printed will continue to be iterated on, and then finally mounted to the 80/20 extrusion. As of this memo, the materials to be used for the frame itself have been ordered and the team is waiting. The frame will be using two 5 inch pieces of 4040 aluminum extrusion and one piece of 20 inch long of 4040 extrusion. To attach the pieces to each other, right angle brackets and t-slot nuts will be used. The team still has to plan on how to attach the frame to the table itself.

Appendix:

Current Program: **https://github.com/ChanceCuddeback/CAN_TMotor**