M.A.R.S.
Motor Algorithmic Regulation System

Aaron Jarmusch, Ethan Conway, Jacob Stafford, and Joel Huffman

CAPSTONE Senior Design 2022-2023, Department of Electrical and Computer Engineering, University of Delaware

Report GitHub

PROBLEM STATEMENT

The beginnings of space travel to Mars can be traced back to the mid-20th century with the launch of several spacecraft by the United States and the Soviet Union, the first successful Mars flyby mission being the Mariner 4 that was launched by NASA. Since then, several more missions have been carried out to Mars, most notable being the Mars rover program which began in 1997. This began with the launch of the Sojourner rover and continuing with the Spirit, Opportunity, and Curiosity rovers. The Perseverance rover is the most recent continuation of this Mars rover program, and still roams Mars today.

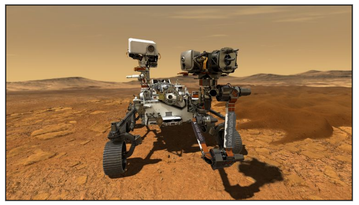


Figure 1: Perseverance Rover which utilizes Northrop Grumman-developed technology.

Northrop Grumman has always worked very closely with NASA, dating back to 1964 with their heritage company TRW contributing to placing a couple hundred satellites in orbit. Since then, Northrop Grumman have worked on several moon missions, satellites, and more recently, the Mars rover program. Northrop Grumman worked on several of the Mars rovers in the past, developing several different components and systems that would eventually be sent to space. Our senior design project is to help Northrop Grumman develop an improved noise-reducing algorithm for their motor and resolver apparatus, that will potentially be used in the systems sent to Mars in the future.

Figure 2: Picture of Mars.

PROJECT GOALS

The overall goal for our project is to work with Northrop Grumman in order to create several different algorithms for a resolver encoder, taking resolver winding sinusoids and yielding the angular position of the resolver. This will aid in the reduction of signal noise for the system that goes into the Mars rovers.

Goals:

- Create multiple unique models in Matlab Simulink to simulate the encoder's performance
- Have a working physical model of a similar system so we can better understand the application of our project
- Work on a neural network that will be trained on the different noise reduction algorithms and several different graphs of signal noise data

NORTHROP GRUMMAN

Figure 3: Northrop Grumman logo.

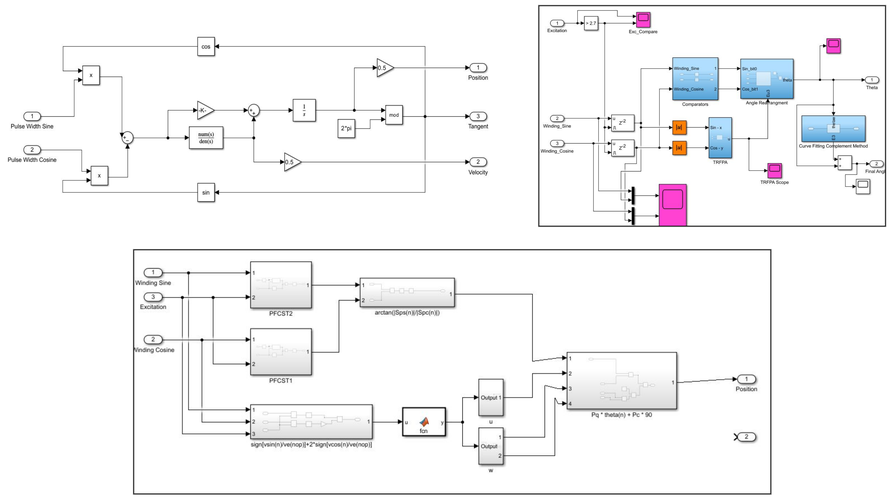
APPROACH

Due to the nature of working with a third-party the process of getting started was slow. Since we did not fully know what they were expecting of us, we initially worked on a physical model. After establishing a regular meeting routine with Northrop Grumman, we honed in on simulations resolver encoders, utilizing research papers to compare and analyze results. We persisted with building the physical model on the side, as we had already started the process earlier. We also decided after getting part way through building the simulations of resolver encoders to see if a neural network would be capable of performing as well as the simulated models. The resolver encoders we have explored:

- Physical build of resolver and resolver encoder
- Phase Locked Loop (PLL) Simulink Model
- Third-Order Ration Polynomial (3-ORP) Simulink Model
- S-Transform (S-T) Simulink Model
- Recurrent Neural Network (RNN)

DESIGN

Simulink is a matlab tool to simulate signals through a system. Below are the simulink models of the encoders.



Figures 4-6: Our encoder models in Simulink. PLL in the top left, 3-ORP in top right, and S-T at bottom.

Each Simulink model is working in different ways under different mathematical approximations. An arctangent-like function is more or less the goal, but arctangents are hard to realize in reality. One of models attempted is a PLL which uses a PI-controller logic with feedback being the current outputted angle of the resolver to correct the output angle to the actual angle. Another being a 3-ORP method which employs a third-order rational fraction polynomial approximation to convert sinusoidal signals into pseudo linear signals, which are then compensated for final angles using the polynomial least squares method. Lastly the S-T first extracts the envelope by PFCST, then quadrant information is used to correct the arctangent to get accurate rotor position.

The neural network is still a work in progress, but would have a similar nature to the Simulink models above. Taking in winding sinusoids and outputting angle, in rads, of the resolver. The model is taking time to be coded and data set of simulated signals collected. Once completed and trained, the recurrent neural network will use mathematical nodes to decode the signals into positional angle.

PHYSICAL MODEL

The physical model shown on the right consists of a 3D printed chassis, a threaded rod, couplers, a brushless motor, and a resolver. We will use this physical model to get readings from the resolver, then run the sine and cosine signals through our models. This represents a real way of how our models can reduce signal noise in systems and increase accuracy.

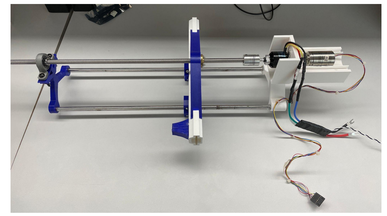


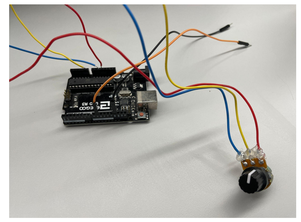
Figure 7: Physical model with motor, resolver, and ESC.

```
#include <Servo.h>

byte servoPin = 9;
byte potentiometerPin = A0;
Servo servo;

void setup() {
servo.attach(servoPin);
servo.writeMicroseconds(1500);
delay(7000);
}

void loop() {
int potVal = analogRead(potentiometerPin);
int pwmVal = map(potVal,0, 1023, 1100, 1900);
servo.writeMicroseconds(pwmVal);
}
```



Testing Video

Figure 8: Arduino code (left), Arduino with potentiometer connected (right.)

The Arduino sends signals to the electronic speed controller. This arms the motor and allows it to rotate in both directions. The potentiometer connected to the Arduino allows the user to turn the potentiometer and control the speed and direction that the motor rotates. The QR code shows a video of the physical model hooked up to an oscilloscope and reading values in real time. This allows us to extract the data from the resolver and feed it in to our various algorithms and models.

PROJECT STATUS

The physical model of our project is completely functional and allows us to control the motor with a potentiometer, and read the sine and cosine values from the resolver. We currently have three MATLAB Simulink models that encode the signal from analog resolver to digital. The neural network is still in progress and has to still be trained. We will continue to train the model with signal noise, which will improve the accuracy of our model.

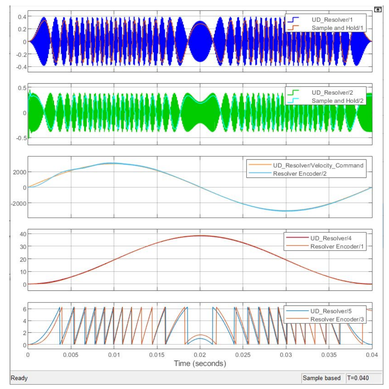


Figure 9: Output of PLL Model encoder vs. resolvers true simulated values. The UD_Resolver is the true value.