paths, a programmer would create a new branch in which they would add and modify the autonomous code. The programmer would then submit a pull request to the master branch in the Stryke Force repository. Pull requests undergo careful peer and mentor review before being merged into the master branch.

Team 2767
Stryke Force

Est. 2009

Kalamazoo, MI

Software Development Team 2019

Software Designed for Drivers

We strive to meld high-performance hardware with custom software to provide our drive team with the best robot possible. Our **Third Coast Swerve Drive** software has historically provided unmatched maneuverability and response.

Precision Control

StrykeForce delved deeper into motion profiling this year than ever before. Using the CTR-Electronics Talon SRX Motion Magic functionality, the programming team controlled the motion-profile by accelerating at a specified acceleration to a setpoint. This feature in the software allows the robot to make precise and repeatable actuator movements.

“Robot Safety Glasses”

At Stryke Force, safety is our primary concern so it’s only natural that safety rules apply to all, human and otherwise. Programmers are using a custom **Safety Subsystem** that interfaces with the Elevator, Biscuit, and Intake subsystems. Every 20ms, all three subsystems update their current state and then the Safety Subsystem runs through a matrix of legal and safe positions updating Talon SRX forward and reverse soft limits. Since the axes cannot pass the soft limits, the programmers have ensured the robot’s safety.

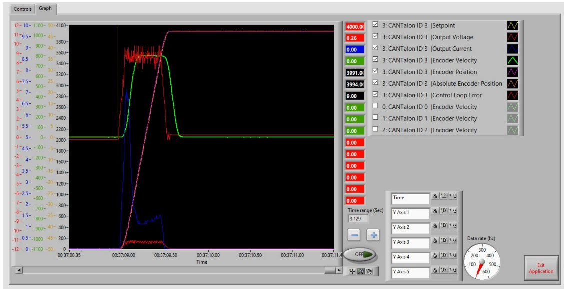
“Twisting our way through the sandstorm”

The software team focused on helping our driver perform during Sandstorm. In addition to pathfinding using Pathfinder, we’re implementing **computer vision** (CV) to guide our driver through Sandstorm. Since vision processing significantly affects CPU performance, we decided to use a Raspberry Pi co-processor. Our custom Python CV application *Pyeye* uses OpenCV to calculate the range and bearing to targets which we then publish to NetworkTables.

After transforming the range and bearing from the camera coordinate system, the robot autonomously approaches the selected target and places the Game Piece in a movement called the ”twist”. Borrowed from the Robot Operating System (ROS), the term describes the motion of a yaw transposed onto a linear movement.

Custom Development Tools

The software team also builds and maintains custom development tools called **Third Coast Telemetry** (TCT). These tools are used by the entire team to test and tune drive systems, actuators and sensors. The Stryke Force Grapher application allows the team to log and analyze the robot performance and perfect the tuning by plotting data received from the RoboRIO. Stryke Force is able to chart almost any data possible from the RoboRIO. TCT and the Grapher applications provide Stryke Force with deep insight into robot performance and are invaluable to the development process.



Similar to the Grapher, Keeper logs data recorded from the robot during the execution of a command. The data is then made available through a Dash app using PostgreSQL and Plotly.

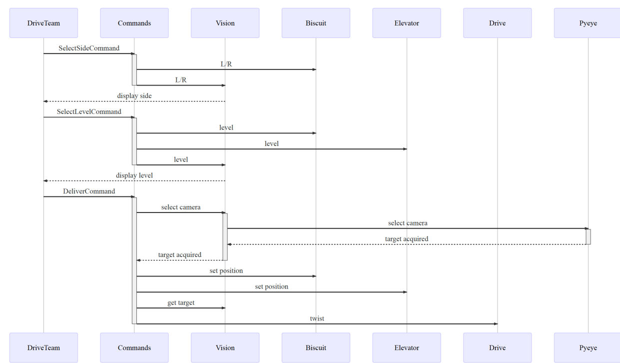
Stryke Force programmers are using a Jupyter Notebook server this year to test CV applications and algorithms, coordinate transformation math, and any other ideas that come to mind. All of our programmers have enjoyed the ease of logging in and being able to test anything anywhere.

We are pleased to make TCT and the Grapher available as open source at https://github.com/strykeforce.

Software Engineering

As members of the Stryke Force team, we learn how to work together using real-world tools and processes that will benefit us during our college years and careers ahead.

This year’s FIRST Deepspace code is written in Java, an object-oriented programming language and leverages WPILib command-based programming. Our use of system sequence diagrams speeds up development and testing since all events are carefully planned in advance.



The DEEPSPACE and Pyeye projects use GradleRIO, the open-source build automation software. Stryke Force programmers use Gradle to build projects, deploy code to the RoboRIO, and manage dependencies.

Stryke Force uses GitHub and Git for version control. The programming team adopted the OneFlow Git Branching Model and Workflow. The programmers maintain one master branch and create pull requests from feature branches. For example, when working on autonomous