FRC Team 980 Code Library

**Classes**

|  |  |
| --- | --- |
| MyRobot.cpp | Handles: Main operator control loop.  Control Modes  Autonomous (Hybrid Mode)  Initializes: Jaguars  Victors  Joysticks |
| MyRobot.h |  |
| MyJoystick.cpp | Checks if joystick is “moving” or not |
| MyJoystick.h |  |
| Cyclops.cpp | Handles all camera operation:  Distance tracking  Target acquisition  Position tracking |
| Cyclops.h |  |
| jsbuttons.h | Defines all buttons on joysticks to their intended use |

**Methods**

MyRobot.cpp

|  |  |  |
| --- | --- | --- |
| void | **message(char \*fmt, …)** | |
| prints argument in printf (e.g. message(“%f”, distance); ) will print out the float distance | |

|  |  |  |
| --- | --- | --- |
| double | **limit(double val, double min = -1, double max = 1)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **Autonomous(void)** | |
| choose auton\_state in method. | |

|  |  |  |
| --- | --- | --- |
| void | **OperatorControl(void)** | |
| contains operator control while loop for all buttons and commands | |

|  |  |  |
| --- | --- | --- |
| void | **CheckStopBridge(void)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **RunBridge(bool up)** | |
| if *up* is true the bridge goes down, else the bridge goes up | |

|  |  |  |
| --- | --- | --- |
| float | **GetRPM(void)** | |
| returns the speed in RPM of the master motor | |

|  |  |  |
| --- | --- | --- |
| void | **Drive(float left, float right)** | |
| sets left and right drive motors to *Set(limit(left))* and *Set(limit(-right))* | |

|  |  |  |
| --- | --- | --- |
| void | **DriveControlPosition(float position\_right, float position\_left)** | |
| sets specific distance values to the drive motors, Set(position\_left) Set(position\_right) | |

|  |  |  |
| --- | --- | --- |
| void | **DriveControlSpeed(float speed\_right, float speed\_left)** | |
| sets specific speed values to the drive motors | |

|  |  |
| --- | --- |
| void | **SetShooterSpeed(float speed)** |
| sets speed on the master motor for shooter 🡪 gets the voltage 🡪 sets shooterSlave1 to the found voltage | |

|  |  |  |
| --- | --- | --- |
| float | **GetRightEncoder(void)** | |
| returns the value for the right encoder | |

|  |  |  |
| --- | --- | --- |
| float | **GetLeftEncoder(void)** | |
| returns the value for the left encoder | |

|  |  |  |
| --- | --- | --- |
| void | **SetBrakes(bool brakeOnStop)** | |
| If in kNeutralMode\_Coast then changes jaguars to kNeutralMode\_Brake and vice-versa | |

|  |  |  |
| --- | --- | --- |
| void | **PerformBalanceTrick(MyJoystick \*joy)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **PerformBalanceTrickSpeed(MyJoystick \*joy)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **DriveControlMode(CANJaguar::ControlMode control)** | |
| switch statement that sets the control mode to kPosition, kSpeed, or kPercentVbus | |

Cyclops.cpp

|  |  |  |
| --- | --- | --- |
| void | **UpdateDistance(int width)** | |
| sets *distance* to *m\_latestDistance* | |

|  |  |  |
| --- | --- | --- |
| unsigned int | **GetDistanceToTarget(void)** | |
| returns *m\_latestDistance* which is found in **UpdateDistance(int width)** | |

|  |  |  |
| --- | --- | --- |
| vector<vector<int>> | **GetTargetCenters(void)** | |
| sets thresholds for color processing. Checks for a rectangle target. Prints out how off we are from the center of the target in pixels. | |

|  |  |  |
| --- | --- | --- |
| void | **UpdateTargetAligned(vector<vector<int>> &points)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **SendDistance(void)** | |
| sends the distance to the LED display | |

|  |  |  |
| --- | --- | --- |
| int | **targetDiskTast(UINT32 arg)** | |
|  | |

|  |  |  |
| --- | --- | --- |
| void | **Start(void)** | |
| starts the Cyclops function calls | |

|  |  |  |
| --- | --- | --- |
| void | **Stop(void)** | |
| stops the Cyclops function call | |

MyJoystick.cpp

|  |  |  |
| --- | --- | --- |
| bool | **Dead(void)** | |
| returns false if the joystick is moving, else return true | |