Control Systems Testing Plan Document

ROVER SYSTEM

1. int GetHeartbeatCount()
   1.1. Should return zero heartbeat count
2. void IncrementHeartbeatCount()
   2.1. Should increment heartbeat count one time
3. bool IsHeartbeatSynced(int)
   3.1. Should verify rover is synced at start
   3.2. Should verify heartbeat not synced at start
   3.3. Should verify heartbeat is reset when not synced

ESP

1.

WHEEL

1. void Initialize()
   1.1. Should call the initialize function twice
   1.2. Should start with default values
2. Void SetHubSpeed()
   2.1. Should set hub speed within min and max range
   2.2. Should set hub speed beyond the min and max range
3. Void SetSteerAngle()
   3.1. Should set steer angle within min and max range
   3.2. Should set mod 360 of steer angle when beyond min and max range
   3.3.

DRIVE SYSTEM

1. void Initialize()
   1.1. Should start with default values
2. std::string GETParameters()
   2.1. Should return the starting defaults response
3. void ParseJSONResponse(std::string & response)
   3.1. Should parse expected values
   3.2. Should throw exception when given less than expected arguments
   3.3. Should not throw exception when given more than expected arguments
4. bool IsOperational()
   4.1. Should return false at start
   4.2. Should return true after setting IsOperation to 1
5. char GetCurrentMode()
   5.1. Should return starting drive mode S
6. bool IsNewMode()
   6.1. Should return false with same modes
   6.2. Should return false with different modes
7. bool IsStopped()
   7.1. Should return true since rover is stopped at start
   7.2. Should return false after all hub wheels start moving

- **7.3.** Should return false if at least one hub wheel is moving

8. void StopWheels()
    - **8.1.** Should stop all wheels after one call
9. void SetWheelSpeed(double target_speed)
    - **9.1.** Should lerp to 4 by setting speed to 2 → 3 → 4
    - **9.2.** Should lerp from 4 to zero by setting speed to 2→1→0
    - **9.3.** Should lerp to -4 by setting speed to -2 → -3 → -4
    - **9.4.** Should lerp from -4 to zero by setting speed to -2→-1→-0
10. void HomeWheels()
    - **10.1.** Should home wheels after moving
11. bool AllWheelsAreHomed()
    - **11.1.** Should return true when at start position
    - **11.2.** Should return false if all wheels are not homed
    - **11.3.** Should return false if one wheel is not homed
12. void HandleRoverMovement()
    - **12.1.** Should slow down when heartbeat is out of sync
    - **12.2.** Should stop movement when rover is not operational
    - **12.3.** Should switch into all the valid drive modes
    - **12.4.** Should stay in the current mode when passed invalid drive mode
    - **12.5.** Should stay in same mode when not operational but has new mode
    - **12.6.** Should only slow down when heartbeat not synced
    - **12.7.** ‘S’ - mode
        - 12.7.1. Should have all steer motor angles at 90
        - 12.7.2. Should have all hub motors at same non zero speed
    - **12.8.** ‘D’ - mode
        - 12.8.1. Should clamp steer angles when over 45
        - 12.8.2. Should clamp steer angles when under -45
        - 12.8.3. Should set left angle to 6, back to -15, right to 10 when passed angle 10
        - 12.8.4. Should set right angle to -6, back to 15, left to -10 when passed angle -10
        - 12.8.5. Should have correct starting angles
        - 12.8.6. TODO: Calculations for expected speed - all same speed at the moment
    - **12.9.** ‘T’ - mode
        - 12.9.1. Should have all steer motor angles at 90
        - 12.9.2. All hub motors should be at the same speed
    - **12.10.** ‘L’,’R’,’B’ - mode(s)
        - 12.10.1. Should adjust only the speed and angle of one wheel
13. void SwitchLegOrientation(int position)
    - **13.1.** should keep initial leg orientation
    - **13.2.** keep initial leg orientation with negative position"
    - **13.3.** keep initial leg orientation with position 3
    - **13.4.** should rotate wheel orientation by one leg
        - 13.4.1. should rotate wheel orientation by two leg
    - **13.5.**
14. void GetBatteryPercent() //implement these

14.1. Should return greater than 100% when voltage is greater than 4.2

14.2. Should return a warning when voltage is lower than 3.05V

14.2.1. Should return a low battery warning at voltage = 3.05V

14.3. Should return another warning when voltage is lower than 3.25V

14.3.1. Shouldn’t return a warning at voltage = 3.25V

14.4. Should return the percentage of the useable battery remains at any value

15. PrintRoverData()

WRIST JOINT

1. int GetPitchPosition()
   1.1. Should return zero angle at rover startup
   1.2. Should return non zero angle after moving wrist once
2. int GetRollPosition()
   2.1. Should return angle zero at rover startup
   2.2. Should return non zero angle at after moving wrist once
3. void SetPitchPosition(float pitch_angle)
   3.1. Should return zero when setting pitch to zero
   3.2. Should return 90 when setting pitch to 90
   3.3. Should return 180 when setting pitch to 180
   3.4. Should return 0 when setting pitch to -1
   3.5. Should return 180 when setting pitch to 181
4. void SetRollPosition(float roll_angle)
   4.1. Should return zero when setting roll to zero
   4.2. Should return 90 when setting roll to 90
   4.3. Should return 180 when setting roll to 180
   4.4. Should return 0 when setting roll to -1
   4.5. Should return 180 when setting roll to 181
5. Void setSpeed
   5.1. Should return 0 when speed is set to 0
   5.2. Should return 5 when speed is set to 5
   5.3. Should return 10 when speed is set to 10
   5.4. Should return 10 when speed is set to 11
6. void SetZeroOffsets(float left_offset, float right_offset)
   6.1. Sets offset to 0 when MPU is flat
   6.2. Sets offset to 90 when MPU is vertical
7. Acceleration GetAccelerometerData()
   7.1. Mock MPU return 0 for x,y,z
   7.2. Mock MPU return 90 for x,y,z

HAND

1. void Initialize()
   1.1. Should return default angles at hand startup
2. void HomeHand()
   2.1. Should home the hand by setting the finger to max angles

3. HandleMovement()
    3.1. Should correctly set the hand mode for every mode
    3.2. Should set hand to transport mode angles
    3.3. Should boundary test the wrist for setting roll position
    3.4. Should boundary test the wrist for setting pitch position
    3.5. Should boundary test the finger movements when in concurrent mode
        3.5.1. Should set the fingers to the maximum angle
        3.5.2. Should set the fingers to the minimum angle
        3.5.3. Should set the speed to the maximum
        3.5.4. Should set the speed to the minimum
        3.5.5. Should set the fingers to one beyond the maximum
        3.5.6. Should set fingers to one beyond the minimum
        3.5.7. Should set the speed to one beyond the maximum
        3.5.8. Should set the speed to one beyond the minimum

JOINT

1. void Initialize()
    1.1. Should start with default values
2. int GetSpeed()
    2.1. Should return zero when joint is first initialized
3. int GetPosition()
    3.1. Should return zero when joint is first initialized
4. Acceleration GetAccelerometerData()
    4.1. Mock MPU return 0 for x,y,z
    4.2. Mock MPU return 90 for x,y,z
5. void SetSpeed(float target_speed)
    5.1. Should return 0 when joint speed is set to 0
    5.2. Should return 100 when joint speed is set to 100
    5.3. Should return 100 when joint speed is set to 101
    5.4. Should return -100 when joint speed is set to -100
    5.5. Should return -100 when joint speed is set to -101
6. void SetPosition(float angle)
    6.1. Should return 0 when joint position is set to 0
    6.2. Should return 0 when joint position is set to -1
    6.3. Should return 180 when joint position is set to 180
    6.4. Should return 180 when joint position is set to 181
7. void SetZeroOffset(float offset)
    7.1. Sets offset to 0 when MPU is flat
    7.2. Sets offset to 90 when MPU is vertical

ARM SYSTEM

1. MoveRotunda()
    1.1. Should lerp to 4 degrees in steps of 2→3→4
    1.2. Should lerp to 0 from 4 degrees in steps of 2→1→0
    1.3. Should lerp to -4 degrees in steps of -2→-3→-4

1.4. Should lerp to 0 from -4 degrees in steps of -2→-1→0

2. MoveElbow()
    - 2.1. Should lerp to 4 degrees in steps of 2→3→4
    - 2.2. Should lerp to 0 from 4 degrees in steps of 2→1→0
    - 2.3. Should lerp to -4 degrees in steps of -2→-3→-4
    - 2.4. Should lerp to 0 from -4 degrees in steps of -2→-1→0
3. MoveWrist()
    - 3.1. Should lerp to 4 degrees in steps of 2→3→4
    - 3.2. Should lerp to 0 from 4 degrees in steps of 2→1→0
    - 3.3. Should lerp to -4 degrees in steps of -2→-3→-4
    - 3.4. Should lerp to 0 from -4 degrees in steps of -2→-1→0
4. IsOperational()
    - 4.1. Should return false as default because it’s set to 0
    - 4.2. Should return true when IsOperation is 1

FINGER

1. void Initialize()
    - 1.1. Should start with default values
2. void SetPosition()
    - 2.1. Should set position to 0 when set to 0
    - 2.2. Should set position to 0 when set to -1
    - 2.3. Should set position 180 when joint speed is set to 180
    - 2.4. Should set position 180 when joint speed is set to 181
3. void SetSpeed()
    - 3.1. Should set speed to 10 when set to 10