Milestone 2 report:

# Subsystem details:

|  |  |
| --- | --- |
| The subsystems are as follows: | |
|  | Robot simulator block, |
|  | Encoder block, |
|  | Motor block, |
|  | Line sensor block, |
|  | Ultrasonic sensor block, |
|  | *Another block* |

# Distance and Angle Control Algorithm:

This was demonstrated in the two Simulink files related to the robot moving 1 metre (with 10% error), as well as when the robot turns 90, 180, and 270 degrees (with 10% error) while remaining in the same place.

|  |
| --- |
| *Figure 1: Simulink block diagram for moving the robot 1 metre along a straight line.* |
| *Figure 2: Simulink block diagram for moving the robot turning 90 degrees in place.* |
| *Figure 3: Simulink block diagram for moving the robot turning 180 degrees in place.* |
| *Figure 4: Simulink block diagram for moving the robot turning 270 degrees in place.* |

The details of how all subsystems integrate with each other to get the desired motion is shown below. The table follows the logical order of how the system will work.

|  |  |
| --- | --- |
| For Figure 1 (Robot moving 1 metre along a straight line), | |
|  | Encoder Simulation Block,  Measures the rotation of the left and right wheels in ticks. The block outputs the tick count for the left and right motor, given as LeftEncoderTick and RightEncoderTick. If the wheel rotates clockwise, the tick counter will increment representing forward motion. If the wheel rotates counterclockwise, the tick counter will decrement representing reverse motion.  In this case both wheels will have equal tick counts.  The two outputs are fed into the system through two gain blocks. The output is also measured with a display block, to see if the correct distance is being travelled by the robot. |
|  | Gain Blocks,  These blocks perform the following math operation to convert the number of ticks from the robot into a length in metres:  The two gain blocks are fed into a summation block and the output is halved to get the distance travelled by the robot (since both wheels are inline and move the same distance). The final output is then fed into a switch block. |
|  | Switch Block,  This switch block checks if the distance travelled, in metres, exceeds 1 metre. If it does not, then the robot will be allowed to move, and the output sent to the wlwr block will be 1. As soon as the statement evaluates as true, then the output to the wlwr block will be cut off and it will stop. |
|  | To wlwr Block,  This clock converts the linear velocity of the robot given by the switch statement into left and right wheel angular velocities, wl and wr respectively. This is then fed into the left and right motor blocks. |
|  | Motor Blocks,  The angular speeds, wl and wr, are fed into the motor blocks. There is one for the left motor and another for the right motor.  Facilitates movement of the robot for the desired motion. In this specific simulation, it is an LTI system that will model the motor providing motion. There is one for the left and right motor.  The output of these is fed into directional speed control blocks to ensure the robot is always moving forward.  The output of these is both fed into the Robot simulator block |
|  | Robot simulator block,  This subsystem allows for the choice of different maps for the robot to follow depending on the simulation type. For this section, it will be the simple straight-line map for the robot to follow along until it reaches 1 metre. Here, the initial position of the robot on the map may be set, as well as the robot characteristics. The two that can be varied are the wheel radius and axle length. |

|  |  |
| --- | --- |
| For Figure 2 to 4 (Robot rotating 90, 180, and 270 degrees in place), **Should it not be that instead of one robot wheel moving and the other remaining stationary, that it should be that one wheel moves forward and the other moves backwards in order for the central axis of the robot to remain stationary? Should it not pivot about its centre, instead of one of the wheels.** | |
|  | Encoder Simulation Block,  Measures the rotation of the left and right wheels in ticks. The block outputs the tick count for the left and right motor, given as LeftEncoderTick and RightEncoderTick. If the wheel rotates clockwise, the tick counter will increment representing forward motion. If the wheel rotates counterclockwise, the tick counter will decrement representing reverse motion.  In this case the outermost wheel will rotate while the innermost wheel remains stationary.  The two outputs are fed into the system through two gain blocks. |
|  | Gain Blocks,  These blocks perform the following math operation to convert the number of ticks from the robot into a length in metres:  The two gain blocks are fed into a minus block to find the distance travelled by the outermost wheel. This is then fed into a divide block where the distance is divided by the axle length to get the radial distance travelled by the robot, in radians. |
|  | Switch Block,  The output from the divide block is converted to degrees and made to always be positive (abs block) before entering the switch block. This is then fed into the switch block. If the angle is less than the required angle, then the robot will rotate, but once the condition evaluates to true (angle of the robot exceeds the required angle), the robot stops dead in its tracks. The output is then fed into the wlwr. |
|  | To wlwr Block,  This clock converts the linear velocity of the robot given by the switch statement into left and right wheel angular velocities, wl and wr respectively. This is then fed into the left and right motor blocks. |
|  | Motor Blocks,  The angular speeds, wl and wr, are fed into the motor blocks. There is one for the left motor and another for the right motor.  Facilitates movement of the robot for the desired motion. In this specific simulation, it is an LTI system that will model the motor providing motion. There is one for the left and right motor.  The output of these is both fed into the Robot simulator block |
|  | Robot simulator block,  This subsystem allows for the choice of different maps for the robot to follow depending on the simulation type. For this section, there are no maps needed since the robot moves |

# Line Following Algorithm:

|  |
| --- |
| *Figure 5: Simulink block diagram for the robot following a line of variable winding.* |

|  |  |
| --- | --- |
| For Figure 5, | |
|  | Line Sensor Simulation Block,  This block will, based on the position of the robot and offset of the sensors, will use a simulation map to compute the line sensor values.  The output of this block, LineSensorArray, is fed into a demux block as well as the lineArray block. |
|  | Demultiplexer Block,  The demux helps to see what the values of each sensor are. A “0” indicates that the sensor sees the line, and a “1” means the sensor is off the line.  The output of the Demux is fed into an AND gate. If the output of the AND gate is “0”, then the robot is on the line. If it is “1”, then the robot is completely off the line.  To make the logic easier to understand, where “1” is good and “0” is bad, a NOT gate is added. This makes the output of the Line Sensor Simulation Block that leads to the switch block more logical and better to work with. |
|  | lineArray Chart Block,  Speed control for when the robot is on the line. There are two inputs, one is the LineSensorArray, and the other is the value K. The output is w and will be fed directly into the switch block relating to w. A few calculations have to be done before being fed into the switch block for v. |
|  | Switch Blocks,  There are two switch blocks, one for v and another for w  From the demux section discussed in 2., when the output is “1”, the switch statement is evaluated as true, and the output from the lineArray chart is fed into the respective blocks. **Explain the maths done for v and w.** if the output is “0”, then all power will be cut to the robot since it has overshot its mark and gone off the line. |
|  | To wlwr Block,  This clock converts the linear velocity of the robot given by the switch statement into left and right wheel angular velocities, wl and wr respectively. This is then fed into the left and right motor blocks.  The input is from the switch blocks. The output then leads to the motor blocks, and the process follows the same route as the angle following algorithm. |
|  | Motor Blocks,  The angular speeds, wl and wr, are fed into the motor blocks. There is one for the left motor and another for the right motor.  Facilitates movement of the robot for the desired motion. In this specific simulation, it is an LTI system that will model the motor providing motion. There is one for the left and right motor.  The output of these is fed into directional speed control blocks to ensure the robot is always moving forward.  The output of these is both fed into the Robot simulator block |
|  | Robot simulator block,  This subsystem allows for the choice of different maps for the robot to follow depending on the simulation type. For this section, it will be the simple straight-line map for the robot to follow along until it reaches 1 metre. Here, the initial position of the robot on the map may be set, as well as the robot characteristics. The two that can be varied are the wheel radius and axle length. |

# Object Detection Algorithm:

# Localisation Algorithm: