**Actions for Robot on Course A or C**

\* Same order on course B or D, except robot would TURN\_CCW in Step 5, robot would TURN\_CW in Step 9, and robot would TURN\_CCW in Step 10

1. Initialize Robot
   1. Adjust encoder thresholds for analog optosensors
   2. GPS\_Initialize function will start GPS and update initial values
      1. Enable GPS
      2. Use knob on Handy Board to initialize course region being used
   3. Kill\_Motor function to make sure motors are off
      1. Prints start status to LCD
      2. Assigns power level of 0 to both left and right wheel motors
      3. Beeps for completion and returns value indicating success
   4. Use CdS\_On\_Off function to have robot wait for light indicator for start
      1. Checks the analog port of CdS cell for value
      2. Returns ON if value is less than 60 (detects light), otherwise returns OFF
2. Begin Hill\_Climb function when CdS detects start light (rear of robot heading first up the ramp)
   1. Motor\_Kill function to ensure motors are off to begin
   2. Reset shaft encoder values to 0 for both wheels
   3. Print start status and current gps and encoder values to LCD
   4. Use selection statements to choose motor direction
      1. Motors are set to positive power levels if FORWARD is passed to function
      2. Motors are set to negative power levels if BACKWARD is passed
   5. Use while loop to monitor motor function using shaft encoder values
      1. Continue to move while the encoder counts for each wheel are less than the requested distance to travel
         1. Check if robot has crossed an finalDestinationX or finalDestinationY plane to ensure the robot has reached the correct area to leave while loop
   6. When encoder count exceeds the totalCounts and finalDestination planes have been crossed, use Motor\_Kill function
   7. Return success for a completed run
3. Hang Traffic Light from bucket located on already raised forklift
   1. Use Traffic\_Light\_Deposit function to move the robot to the appropriate traffic light
      1. Print start status and current gps and encoder values to LCD
      2. Use GPS value to determine if traffic light is needed to the LEFT or RIGHT
      3. Rotate robot toward correct traffic light post
         1. If traffic light is LEFT, set RIGHT\_WHEEL to a positive motor level and LEFT\_WHEEL to a negative motor level for a short time
         2. Else, if the traffic light is to the RIGHT, set LEFT\_WHEEL to a positive motor level and RIGHT\_WHEEL to a negative motor value for a short time
      4. Set robot drive train motors to lower power level to proceed until white line is detected by line following
      5. Implement Follow\_Line function that will lock onto the line located under the traffic light needed and have an input of a finalDestinationX or finalDestinationY
      6. Continue to drive until the microswitch located under the traffic light is deactivated, signaling the traffic light has been lifted due to magnetism
      7. Use Motor\_Kill command to stop the robot and return a value of SUCCESS
4. Move Robot Back from Traffic Light Station (FORWARD with forklift first)
   1. Use Motor\_Straight function (similar to Hill\_Climb function, but using gps\_heading to check for correct direction) going BACKWARD to center of DOWNTOWN
      1. Motor\_Kill function to ensure motors are off to begin
      2. Reset shaft encoder values to 0 for both wheels
      3. Print start status and current gps and encoder values to LCD
      4. Use selection statements to choose motor direction
         1. Motors are set to positive power levels if FORWARD is passed to function
         2. Motors are set to negative power levels if BACKWARD is passed
      5. Use while loop to monitor motor function using shaft encoder values
         1. Continue to move while the encoder counts for each wheel are less than the requested distance to travel
            1. Check if robot has crossed an finalDestinationX or finalDestinationY plane to ensure the robot has reached the correct area to leave while loop
         2. Also check that the robot is moving at the correct heading value
            1. If the direction is outside the acceptable range (found by adding or subtracting GPS\_ERROR from finalHeading )

Stop the wheels

Rotate LEFT\_WHEEL if heading needs to be decreased until within GPS range

Rotate RIGHT\_WHEEL if heading needs to be increased until within GPS range

Restart robot at the initial motorDirection and motorLevels

* + 1. When encoder count exceeds the totalCounts and finalDestination planes have been crossed, use Motor\_Kill function
    2. Return success for a completed run

1. Use Robot\_Turn function to TURN\_CW and align perpendicular to the power station crank
   1. Motor\_Kill function to ensure the motors are off before starting
   2. Reset shaft encoder values to 0 for both wheels
   3. Print start status and current gps and encoder values to LCD
   4. Use gps\_get\_data function to get new gps readings
   5. Use selection to determine what the final gps\_heading should be
      1. If the gps\_heading is outside 45 and 135, a 90 degree turn should produce a finalHeading value of 90
      2. Else, the finalHeading will be 0
   6. Check the rotationDirection passed to the function to determine motor settings
      1. If the direction is TURN\_CCW
         1. Turn LEFT\_WHEEL a negative power level and RIGHT\_WHEEL a positive power level
      2. Else, if the direction is TURN\_CW
         1. Turn RIGHT\_WHEEL a negative power level and LEFT\_WHEEL a positive power level
      3. If all else fail, beep and return a value of FAILURE
   7. Allow motors to run until the desired gps\_heading value is reached
      1. If finalHeading is 0 degrees
         1. Continue to turn until the gps range is reached ( greater than 180 – GPS\_ERROR and less than 180, or less than GPS\_ERROR and greater than 0) or the shaft encoders read a count that is close to a 90 degree turn
            1. Use Motor\_Kill function and return SUCCESS
      2. If finalHeading is 90 degrees
         1. Continue to turn until the gps range is reached (greater than 90 – GPS\_ERROR and less than 90 + GPS\_ERROR) or the shaft encoders read a count that is close to a 90 degree turn
            1. Use Motor\_Kill function and return SUCCESS
2. Drive to Crank
   1. Use Motor\_Straight function to travel the distance from the center of DOWNTOWN to the power station crank, checking the robot orientation with GPS
   2. Use Motor\_Kill function when robot is situated in front of the crank using GPS location references and shaft encoder values for distance
3. Turn Crank
   1. Use Turn\_Crank function to lift the crank multiple times to complete a full rotation
      1. Print start status and current gps and encoder values to LCD
      2. Lower the forklift motor to the lowest level
      3. Drive robot closer to the crank, positioning the fork to the left of the center
      4. Raise the forklift mechanism to the highest level
      5. Reverse drive train motors to move robot away from crank while lowering forklift motor again
      6. Repeat multiple ( 4 times ) and check GPS signal to ensure crank is turned
      7. End with forklift at highest level and backed away from the crank
4. Hit Subway Button
   1. Use Motor\_Straight function to drive across DOWNTOWN and in front of the subway station with the back first (using motorDirection BACKWARD)
   2. Use Subway\_Press function to navigate robot into the subway button
      1. Print start status and current gps and encoder values to LCD
      2. Reverse robot slowly until the back bumper triggers a microswitch, signaling the back is in contact with the subway station
      3. Use Motor\_Kill function to stop the robot
      4. Put RIGHT\_WHEEL on a negative power level, causing the robot to pivot toward the subway button and press it with the bumper
      5. Repeat process again to ensure subway button was activated
      6. Turn robot back to 0 degree heading that was used in the beginning of the function
5. Situate robot at the top of the ramp
   1. Use Motor\_Straight function (forklift forward using FORWARD motorDirection) to navigate robot from subway station to X plane of 0
   2. Use Robot\_Turn function in the TURN\_CCW direction to orient robot with forklift facing forward down the ramp
6. Read Dumpster LED
   1. Use Motor\_Straight function to get the robot down the hill and at a gps\_heading of 90 degrees (forklift forward using FORWARD motorDirection)
   2. Use Follow\_Line function to detect the curved arrow leading to the Dumpster LED indicator light and follow the line BACKWARD (bumper first)
      1. Print start status and current gps and encoder values to LCD
      2. Will use three optosensors to detect the white lines against the black course floor and steer robot
         1. If middle optosensor detects black and the right optosensor detects white
            1. Stop robot, and power left wheel to steer robot back onto track until middle optosensor detects white again
         2. If middle optosensor detects black and the left optosensor detects white
            1. Stop robot, and power right wheel to steer robot back onto track until middle optosensor detects white again
   3. Locate the LED light and use CdS\_Color\_Reading function to distinguish what color light was detected
      1. Print start status and current gps and encoder values to LCD
      2. If analog value is in a low interval between 0 – 15, return RED
      3. Else, if analog value is between 40- 150, return BLUE
      4. Otherwise, detect no light and return OFF
7. Locate Dumpsters and Pick Them Up
   1. Use Robot\_Turn function to turn the robot toward the line closest to the LED
   2. Use Motor\_Straight command to reach the line and Follow\_Line function to guide the robot to the dumpster
   3. Once end of line is reached, use Dumpster\_Lift function to hold dumpster
      1. Print start status and current gps and encoder values to LCD
      2. Drive robot pushing dumpster slowly toward the back wall of the course
      3. Motor\_Kill will be used when the microswitches determine that the robot has come in perpendicular to the wall and dumpster
      4. Robot will then back up a short distance by reversing motor power levels at both wheels and lower the forklift to the lowest position
      5. When bump switch at back of forklift frame is pressed, the dumpster has been located
      6. Forklift will be lifted slowly to the highest position
8. Bring the Dumpsters in front of the Bin
   1. Use Dumpster\_Decider function to determine which receptacle needs to be approached
      1. Print start status and current gps and encoder values to LCD
      2. Use GPS (location in negative or positive X based on the LED light) to designate receptacle locations
      3. If robot is in the same region as the bin
         1. Use Robot\_Turn function to complete two consecutive TURN\_CW processes to orient robot at 90 degrees facing the front forklift toward the receptacle
      4. Else, if the robot is not in the same region as the bin
         1. Use Robot\_Turn function to complete two consecutive TURN\_CW processes to orient robot at 90 degrees facing DOWNTOWN
         2. Use Motor\_Straight using a motorDirection FORWARD with an input distance equal to approximately the Y plane corresponding to the Dumpster LED indicator
         3. Use Robot\_Turn to make a turn (TURN\_CW if arrow pointed left, and TURN\_CCW if arrow pointed right)
         4. Use Motor\_Straight function to drive the robot to a location FORWARD and approximately center of the receptacle area
         5. Use Robot\_Turn to make a final turn toward the bin (TURN\_CCW if arrow pointed left, and TURN\_CW if arrow pointed right)
9. Deposit Dumpster in the Correct Bin
   1. Use Dumpster\_Deposit function to leave the dumpster in the correct storage area
      1. Print start status and current gps and encoder values to LCD
      2. Slowly drive robot toward bin and use Bump\_Switch function to check
      3. When robot is in contact with the depository, lower the forklift a set distance so that the forks are resting on the bin walls
      4. Continue to move robot BACKWARD slowly while slightly raising forklift to allow the dumpster to slide off into the bin
      5. Check that the dumpster is not still connected using a bump switch reading for tension, otherwise repeat the function
10. Navigate back to the Recharge Area and Press the Stop Button
    1. Use Robot\_Turn function to turn BACKWARD toward the X value center of the course (TURN\_CCW if at the Right Bin, TURN\_CW if at the Left Bin)
    2. Use Motor\_Straight function to reach the X value 0 and the distance with the rear first
    3. Use Robot\_Turn function to turn BACKWARD toward the recharge button (TURN\_CCW if at the Right Bin, TURN\_CW if at the Left Bin)
    4. Use Motor\_Straight with going BACKWARD to align with the recharge button and approach
    5. Use the Final\_Button function to guide the robot into touching the recharge button to stop the competition time
       1. Print start status and current gps and encoder values to LCD
       2. Have the robot power the wheels BACKWARD slowly
       3. While all the bump switches on the rear bumper are not activated, continue to drive backward
          1. Adjust the robot to stay in a 90 degree orientation
          2. Once the bump switches are pressed, drive FORWARD a short distance
       4. Repeat Final\_Button multiple times (3 total) to ensure the button was pressed and end the competition

\* A labeled course sketch is included to show the positions of tasks in their orders. \*

**List of Global Variables:**

// Universal definitions

ON 1

OFF 0

// Definitions for motor ports and motor speeds

LEFT\_WHEEL 3

RIGHT\_WHEEL 1

DISTANCE\_CONVERSION 1.75

MOTOR\_POWER\_FULL 100

MOTOR\_POWER\_CRUISE 75

MOTOR\_POWER\_HALF 50

MOTOR\_POWER\_QUARTER 25

MOTOR\_OFF 0

TURN\_CCW -1

TURN\_CW 1

FORWARD 1

BACKWARD 0

GPS\_ERROR 4

SUCCESS 1

FAILURE 0

// Definitions for sensor operations

BUMP\_PORT\_1 1

BUMP\_PORT\_2 2

BUMP\_PORT\_3 3

BUMP\_PORT\_4 4

CDSPORT 6

RED 1

BLUE 2

**List of General Local Variables:**

// Motor\_Straight () and Hill\_Climb () Function Variables:

int motorDirection

int totalCounts

int finalHeading

int motorLevelL

int motorLevelR

int finalDestinationX

int finalDestinationY

int resumeMotor

// Robot\_Turn () Function Variables:

int finalHeading

int continueTurn

int rotationDirection