Evaluation

Does the robot follow a line?

For the robot to be able to follow a line, it first must be able to detect the coloured line and distinguish it from the floor. This involved calibrating the TCS230 to incorporate the line colour values. This system worked by having the Arduino Uno code compare two variables, the colour that is currently being detected by the sensor and the colour that it is set to follow. The script compares these values to know when it is following the line and when the robot veers off the line. It was found that the highest accuracy came from a white floor as some darker surfaces could be grouped under the blue RGB values. For the robot to start following a line, a button press (1 of 3) will assign the variable and tell the Arduino Uno what colour should be followed. The line following component of the code is setup up under an if statement that contains two while loops. The if statements checks that a button is pressed and then either while loop (drive forward or turn) will be active depending on the coloured line status i.e., reading the correct colour or outside the line. The code is written with correct syntax and should work however, this is not the case. Once either while loop is initially triggered, the while loop stays locked and does not transition to the other loop as it simply ignores all code outside the active loop. A suggestion to overcome this error is to create a nested loop that has the variable updating code inside of the while loop. In the Evaluation Video 1.MOV video, based off the rotation of the DC motors, the robot is able to initially register whether the correct line is being detected or not. However, the reset button on the Arduino Uno must be pressed before switching to the other while loop as these loops are locked.

Does the robot safely operate within the vicinity of people and obstructions?

The code was successful in its ability to determine the DC motor speed based off the distance from the closest object. Relevant testing occurred in the test log record “L298N Functionality with HC-SR04”. There were no errors in the creation of this system if the robot was told to drive from the void loop() though, further code development led to issues. It was found that if the code was running a while() loop, the Arduino Uno did not update the speed variable (which is read from the HC-SR04), resulting in the speed being static. A nested loop which contains the speed changing if statements is suggested as a solution to this problem, a piece of code cannot be ignored if it is present in the loop. However, this system does not prevent all collision accidents from occurring. Due to hardware limitations (such as a limited number of ports on the Arduino Uno) only one HC-SR04 was implemented. This means that the robot can only detect what is directly in front of the robot. In a high foot traffic area, the robot could potentially be trampled over. Further improvements such as visibility could be improved. A flag that is attached to the robot and is present at average eye level is recommended as an improvement. In the Evaluation Video 2.MOV video, the speed of the DC motor wheels are dependant on the distance of the hand towards the HC-SR04. The wheels spin at a speed of 200 normally however, as the hand gets closer, the speed slows down until it is not spinning anymore. Note, the code was altered to have the DC motors spin triggered in the void loop environment, rather than in a nested loop as this allowed for the variables to be actively updated. When the hand is directly over the HC-SR04, the component will read a distance of 0 and start spinning the motors at a speed of 200 (as seen in the video). This is due to distance related hardware limitations of the HC-SR04. The component is not able to send a signal past 300 centimetres, this means that if there is simply nothing in its path, it will return a distance of 0 and not move. To fix

this, code was added that assigned a distance of 0 to a speed of 200. The robot will slow down to a stop before reaching an obstruction which means a reading of 0 should not apply to a situation where there is an obstruction as the robot will stop with at least 5 centimetres and the sensors must be purposefully covered. A recommendation to this system is to have the code check if it was previously moving as this would allow the robot to know if it is driving or should still be stopped.

Is the robot able to transport objects?

The robot can transport objects utilising its drive system however, the created robot is a prototype that focuses on the drive system and its subsystems. This means while some objects could be placed on the robot and transported, there is not a designated object holder that has been implemented in the physical construction. However, this is easily implemented in the future as this feature can be considered when designing the body (case) for the robot as a second level of prototyping (a level that would be started once the internal systems were working). As this subsystem was not focused on, there are no recorded tests regarding this feature. In the Evaluation Video 3.MOV video, a small object (Listerine packet) Is placed on a small empty area on the pinboard. As there is no security, the speed of the robot led to the object falling off. This means that the current protype version of the robot is unable to transport objects.

Is the robot able to repeat itself whilst staying switched on?

As the code is not fully completed (errors still active), the robot must continuously be reset in order to test/activate another button. The code states that once a button is pressed, it triggers a while loop which is the overall driving and line detecting system. However, once the Arduino enters the while loop, it ignores the 'stop' code (outside the while loop) which stops both motors when the button nis pressed a second time, resulting in an endless loop that must be switched off. Furthermore, there is currently no way for the robot to know that is has reached its destination (hospital room) as it is programmed to follow a line and once there is no line to follow (the end), the robot will be searching for that line. The simple user interface does not allow for a destination configuration and must purely rely on the line. An additional colour sensor could be placed in front of the first to check for 'no line' however, this will lead to problems when the line is turning. Another suggestion to improve this subsystem would be to implement a Near Field Communication (NFC) reader in the robot and NFC chips at the doors of the rooms which would feed into the reader and allow the robot to know which room it is at. However, there is not enough ports on the Arduino Uno (hardware limitation) and this system may be costly for a hospital. A simpler solution is to change the line colour and have the robot stop under the colour change conditions (rather than completely ignoring all colours) but then again, how would it be able to pass that room to move on to the next. In the Evaluation Video 1.MOV video, based off the rotation of the DC motors, the robot is able to initially register whether the correct line is being detected or not. However, the reset button on the Arduino Uno must be pressed before switching to the other while loop as these loops are locked. This means that the robot is not fully automatic and cannot operate on its own.