Rolls-Royce

RACE YOUR CODE

INSTRUCTIONS

INSTRUCTIONS: THE COMPETITION

The Competition

- Two robots per racer – a 'leader' and a 'trailer'
- Standard code provided in advance and yours to customise as you see fit
- 45 minutes of on-track time for final adjustments
- Three timed laps for your chance to get on the leaderboard
- Time trial finals for the fastest 5 robot racers out there

RaceYourCode was conceived by Rolls-Royce as a fun, innovative way to challenge the finest digital thinkers and coders around the world. As a successful applicant, you'll be racing autonomous, Raspberry Pi-powered robots around a track as fast as you possibly can – a competition that will truly push your programming and data manipulation skills to the limits.

Each 'driver' will take control of two MonsterBorg Robots – a 'leading' and 'trailing' robot, both of which have to cross the start/finish line before your time is confirmed. These robots are controlled by a Raspberry Pi 3, with a Wi-Fi connection to the internet and to each other.

The key challenge is that your leading robot can only 'see' the track in front of it and your trailing robot can only 'see' behind it (each via a small on-board camera with limited range).

The software on each of your robots is a standard load of the series code – it provides the basic building blocks to perform the task but you'll receive the code in advance for any customisations you want to make. Whatever you decide to add, remove or leave the same is completely in your hands.

You'll be able to modify the code on these robots by connecting in to each Raspberry Pi via a single VNC connection. Each robot will have a VNC port assigned to it, and it should be possible to run two simultaneous VNC connections to both the lead and trailing robot.

On race day you'll have 45 minutes to modify and experiment with the code, as your robots run trial laps around the track. A video stream from the on-board cameras will be viewable via Twitch, with telemetry data also displayed via the stream. When your 45 minutes is up, your robots will be placed back in the starting position so that they can complete three timed laps of the course. Record one of the fastest times and you'll be invited back for the final race day, where the top 5 finest robot racers out there compete to win!

The Track

22 metres long, buffered with rubber and emblazoned with multi-coloured tracks: the PiBorg circuit is the ultimate arena for proving your robot racing credentials. The track is separated into six lanes in a particular colour sequence. Your robots will race clockwise, starting in the middle of the track and placed approximately in the correct orientation of travel. There is no requirement to stay in any particular lane during the racing.

- Approx. 22m (72ft) long circuit
- Rubber skirting along the edges of the track is approx. 100mm (4") to 150mm (6") high
- Rubber is 10mm thick to absorb impacts
- Width of track is 1.8m (6ft)
- Track is separated into 6 lanes approximately 305mm (12") wide
- Colour of the lanes from left to right (clockwise) is: red, blue, red, blue and green.

INSTRUCTIONS: THE COMPETITION

Starting Bays, Signals and Timing

Your robots will race clockwise, starting from bays that are placed on the transition between colours and placed approximately in the correct orientation of travel. You'll be placed on the red to green transition in the center of the track.

Be warned! There is a camera at the start line to flag up jump starts. It's not permitted to move your robots before the 'Go' signal. The 'Stop' and 'Go' signals are in the form of red and green LEDs and the actual race timing is done by a laser and a diode at the start-finish line.

- Lead robot's starting bay is approximately 590mm (23") in front of the trailing robot's starting bay, in line with the same transition/lane.
- Stop/Go signals are each made up of 5x20mm LEDs in red or green, placed approximately 1.8m (6ft) in front of the start line.
- Stop/Go signals are pointed directly at the front of the start bays, to enable maximum exposure on the Pi camera.
- Lighting sequence: Green (5 seconds plus); Red (random time between 2 and 30 seconds); Green again.
- Race is live at the second green and it is permitted for the robots to begin racing.

The Robots

- Each MonsterBorg robot has a Raspberry Pi 3, a V2 Raspberry Pi Camera, a ThunderBorg motor Controller, 4x300rpm 37mm motors and 4x87mm wheels.
- House batteries are 10x GP2700mAh and are charged before your allocated time.
- Fully charged batteries last for approx. 3 hours during normal robot control, with no significant discharge roll off at the 45-50 minute mark.
- A fresh set of batteries will not be used for the 3 lap run.
- Each robot has a Wi-Fi connection via a local router to the internet.
- Both robots send telemetry data to the other robot, the same data is shown on the stream
- The only data input is that of the two Raspberry Pi cameras, however any data seen by the cameras can be used in any way you see fit.
- You can also use data from the live stream in any way you see fit.
- Please note that we have seen delays of up to 15 seconds from viewing the stream, something we have no control over.

All robots and batteries etc. will be supplied by us, and have approximately equal speeds, torque, turning abilities and friction. They'll be calibrated using the following methods:

- On a fresh battery, we will record the time taken from two points in the track across a laser timing rig. The ThunderBorg controllers we are using will have slightly altered PIC code, which allows us to set the maximum power level. This will be dropped down to the point where the top speed matches a pre-determined value. When the ThunderBorg controller is commanded to 100% over I2C, it will put the motors to this calibrated power level.
- Motor output power on all 4 wheels will be set to 100%. The deviation from a straight line will be recorded over 10m. If the deviation is more than a pre-determined value, the robot will be mechanically altered to rectify.
- The camera will take an image of a grid in front of it. If the image deviates by more than a pre-determined value, the camera position will be altered.
- The batteries are kept as a matched set. They will be tested from time to time. If there is a deviation in batteries by more than a pre-determined value, it will be replaced with an equivalent performing battery.

INSTRUCTIONS: THE CODE

Standard Code

Your leading robot will have the standard leading code and your trailing robot will have the standard trailing code. You'll be able to view this code via a SourceForge login and password. You don't need to do anything to transfer this standard level of code to the robots.

Code Requirements

The standard SD card image will automatically run the /home/pi/formulapi/Formula.py script upon startup. This script is responsible for starting everything up, but most of the code is actually in ImageProcessor.py. Other files you may wish to alter are Settings.py and Race.py which hold the settings used by the processing code and the code running in a separate user thread respectively.

The SD card will boot up with the VNC service active, which allows you to log-in with the username and password you're given. It's up to you to ensure that you don't cause any problems with this connection. Typically, there'll be no assistance in reconnecting after the robots have been booted.

If you don't want to use Python that's fine – you can use other programming languages.

Please note that if your software doesn't run on our standard SD card image then it'll be up to you to update libraries on the standard build within the allocated run time. This may be less than desirable given the time it will potentially take to download and install.

The internet connection is a standard FTTC VDSL and therefore susceptible to normal traffic restrictions. It will be influenced by the VNC connection, video stream and other factors. We recommend that library updates are kept to a minimum and any code transferred via an external service such as SourceForge or GitHub is kept to a small file size.

Telemetry

The video stream from around the track will include telemetry from the robots. This telemetry is transmitted with the standard code and includes:

- Track Position (offset via coloured bar)
- Track Angle
- Track curve
- Speed and steering values
- Approximated distance and lap count
- Processing states such as the lights detection sequence
- Up to three custom competitor values for competitors convenience.

Simulation

In order to test and develop the code without access to the real robots and track we have provided a Java based simulator which tracks the robots movement and generates an image stream to act as the camera input.

The simulator does have some limitations compared to the real robots:

- Everything is evenly lit, there are no shadows or bright spots
- Outside of the track is purely grey, in reality there are a lot of things in the background
- The start lights do not project light like the real ones
- It can only handle one robot at a time, not both

INSTRUCTIONS: THE CODE

- There is no collision detection, the robots can run straight though the track walls
- The robots are only roughly modelled, they do not behave exactly like the real robots

It is definitely possible to use the simulator to improve the code. We have even used it ourselves to tune updates for new robot models before testing them on the real track.

We'll give you more detail on how to get the simulator running if you're selected to race.

The Power Up Sequence

A pre-formatted SD card with the standard lead robot code will be inserted into your leading MonsterBorg robot, and a pre-formatted SD card with the standard trailing robot code will be inserted into your trailing MonsterBorg robot. The actual robots you're given will be chosen at random from the ones available.

A randomly selected charged battery pack will be put into each robot, and then they'll be placed on the center lane of the track.

Switching on is done in the order the steward decides. Each robot is to boot and wait until the lead robot can see the transition from lights off to lights green.

The lead robot then has to inform the trailing robot that lights have changed and to instruct it on how to proceed.

The stewards will wait approximately 2 minutes for the Raspberry Pi's to boot, before attempting to change light states.

Once the robot sees the transition from lights off to lights green, it should turn on the LED on the ThunderBorg to indicate to the steward that it has spotted the lights.

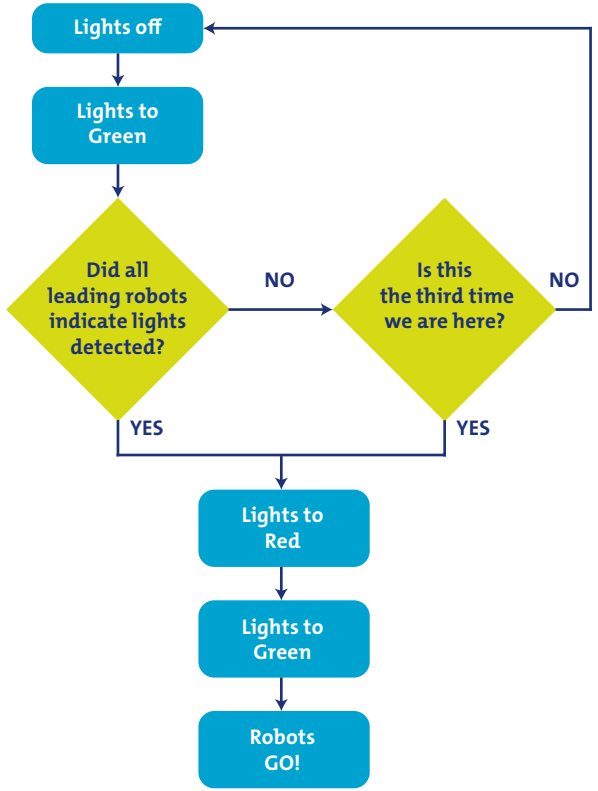
If the LED of the leading robot has not come on, the lights will go to an off state, then to green again. This process is repeated a maximum of three times, before the lights go to red, indicating the Robot should prepare to race on next transition to green. The lights will then go to green and the race is declared 'Go'.
If a robot does not go, the stewards may or may not toggle the lights in all configurations to try and get the robot moving.

The trailing robot has no way of seeing the lights itself, instead it is expected to work this out from the telemetry data supplied by the leading robot.

The time duration between lights off to lights on, lights green to lights red, and lights red to lights green is random. If your robot moves past the start position before the final transition from Red to Green, the robot is declared to have jumped the start and will not be awarded a time.

If the robot does not move based on the lights alone, it will not be aided and will become a blockage for the other robot. No time will be awarded if either of the robots does not complete at least one lap, even if other robot finishes all three laps.

Once the race is 'Go', the lights are not of significance and may be in the Green, Red or Off state.



INSTRUCTIONS: RACE FORMATS

45 Minute Run

At the start of the allocated time, your robots will be placed on their respective start lines, and the start lights will be run through the startup cycle shortly afterwards. During these 45 minutes you'll have full control over VNC of the robots and you can make any alterations to the code that you wish.

During this time, you'll have exclusive use of the track. We will not run the lights sequence during this time, so there is no need to wait for them to start running. The NoLights_Front.py script can be used to start racing properly on the leading robot without waiting for light changes.

Once 40 minutes has passed, you'll receive a five-minute warning via the Twitch chat window. You should use this time to power off the motors on each robot, save off your code and setup the script to detect the stop/go signals.

At 45 minutes, your robots will both be powered off and placed on their respective start lines. They will then be powered on and at least two minutes will elapse before the start of the stop/go signals. This sequence of lights (see Starting Bays, Signals and Timings section of this document) will indicate the

Three Lap Run

As the lights transition from red to the final green, the timer is started. If your robot moves before the lights transition to green then you may be disqualified.

Once both your leading and trailing robot have crossed the start/finish line three times, having completed three laps in normal fashion, the timer is stopped. Provided you haven't been disqualified, your total three lap time will then be added to the leaderboard.

Final

The fastest four competitors out there will go receive an additional 45 minutes and three lap run. The fastest robot pair to complete three laps without being disqualified will win.

Before the final, those who make it through will have the chance to tweak their code again. Whether you stick to your guns or make changes based on what you've seen other people do is up to you.

The final races will be on either Monday 11th December or Tuesday 12th December and the running order will be:

Monday 11th December

5th Fastest competitor goes first

4th Fastest competitor will then be run

3rd Fastest competitor is then run

Tuesday 12th December

2nd Fastest competitor runs

Finally, the fastest competitor is run

The top three winners will be the fastest three out of the Final runs and will receive a trophy.

INSTRUCTIONS: MECHANICAL AND TECHNICAL ISSUES

In the Event of a Mechanical Failure during 45 Minute Run

The stewards may choose any of the following:

- Fix the mechanical problem and allow additional lost time (approximated)
- Place the SD card into another robot and continue run
- Re allocate another time slot

In the Event of a Mechanical Failure in the Three Lap Run

The stewards may restart the entire three lap run by switching off the robot, reposition the robots and restarting the lights sequence as a new 3 lap run.

In the Event of a Mechanical Failure in a Final

The stewards may choose any of the following:

- Restart the final in which the mechanical problem occurred
- Choose as the result, the running order at the time of the mechanical as the result. This is particularly likely if there was an obvious difference

Whether there was a mechanical failure or not is determined entirely by the stewards. Stewards' decisions are final.

In the event of a power outage or internet connection failure, the stewards may choose any of the following:

- Continue the session with time in case of a competitor deliberately trying to stall for time
- Restart the session

Live Broadcast and Commentary

The races will be covered with our video cameras and you'll be able to watch the stream online. The final will also feature a live commentary to really ramp up the excitement...

Anti-Cheating Rules

RaceYourCode is all about promoting ingenuity and alternative approaches to racing robots. That said, everything you do must be in the spirit of competition. Any strategy that's deemed to be cheating by our stewards may result in your expulsion from the competition, without any compensation or reward.

Cheating could include the act of inhibiting other robots by any means other than simply trying to navigate your own robots around the track. If you have any doubts at all about your strategy, please do get in touch with our stewards in advance – you'll be provided with contact details upon successful application.

Other examples of cheating would be:

- Interfering with the data connection in any manner – for example trying to prevent the telemetry being broadcast from other robots
- Any attempt to disrupt the video stream
- Any attempt to prevent the normal running of a competitor

Please note that the above rules are all subject to change.