racedirector

version 2

A fully automated race control tool

Created by Richard Cawthraw & Justin Makaila

1. Contents

2. About

racedirector is an automated application designed to deploy the safety car in iRacing road races, addressing the current gap in the service whereby automated full course cautions only function adequately in oval racing.

The iRacing SDK provides a stream of data to users in-sim. In hosted/league sessions, a more thorough dataset is streamed to the session admin's client. Several data points have been carefully selected and interpreted by the application with the goal of being able to determine large incidents on track in real-time.

The endurance racing experience is enhanced by the addition of cautions periods, or more accurately, the threat of cautions, which means teams must operate a continuous strategy around what their car will do in the event of a safety car (SC) being deployed. Having unbiased stewards able to handle the fair deployment of a SC is a luxurious requirement for hosted races. Also, dedicated corner marshals are unlikely and as such, being able to call a full course yellow (FCY) fairly and consistently poses a significant challenge to event administrators.

racedirector attempts to solve this problem by handling this aspect of race control without the need for any human intervention. As such, it cannot be accused of bias, it is completely objective in its decision making.

The iRacing SDK streams a basic set of real-time data to the administrator's client, containing information about all cars in the race. It also streams an enhanced set of data, but unfortunately this **only** relates to the user's car. Therefore, there is a limitation in that **racedirector** can only use the basic data to inform its decisions. On-track events, such as a car on its roof would appear, to human-eyes, to be an obvious candidate for a SC, but this cannot be 'seen' by **racedirector**. What the application does have visibility of is detailed later on in the document.

However, with **racedirector v2**, the creation of a **client** app to be used by drivers will further enhance the ability of the **racedirector admin** app (run by the session admin) to call accurate FCY's. Drivers running the **client** app will be feeding a remote server with

extra data points not previously available to the **admin** app, which can now consume this extra data, and as such, is able to make more informed decisions than before.

Drivers who do not wish to run the **client** app are welcome to opt-out from doing so. The **racedirector** platform will continue to the best of its ability with the information available regardless of how many drivers use the **client** app.

We do believe **racedirector** adds greatly to immersion, with v2 representing a huge leap forward in many areas. We hope users get a great deal of enjoyment from its addition to the base iRacing service.

3. How it works

Below lists the main data points that are gathered once per second from all cars on track directly in the **admin** app:

- Track surface material
- Lap distance (metres and percentage)
- Current gear
- Is the car in the pits?
- Number of incident points
- Repair flags (aka meatballs)

Using a combination of the above we can derive if:

- The car is in reverse
- The car is reversing
- The car is stationary
- The car is going very slow
- The car is going backwards around the racetrack
- The car is on the track
- The car is off the track
- The care is required to pit for repairs

Once an incident is picked up using a combination of the above criteria, we can assign a level of severity to the incident dependant on other key metadata values, such as:

- Are we under caution already?
- Are we on lap one?
- Are we on the first lap of a restart after a caution?
- Did the incident take place in a track hot-zone?

Hot Zones

With racedirector **v2**, users now have the ability to configure hot zones on a track by track basis. Users are then able to configure different (i.e. higher) weighted values for

incidents that take place within these hot zones, increasing the likelihood of a FCY being called in dangerous areas of the track.

A good example would be a user significantly increasing the incident value scored for a stationary car in the downhill section at Bathurst.

Below lists the main data points that are gathered from the **client** app run by a user (driver):

- Pitch rate
- Yaw rate
- Roll rate
- Roll
- Z-Velocity

Using a combination of the above we can derive if:

- The car is rotating across any axis in an unpredictable manner
- The car is upside down
- The car is airborne

All this extra data now allow **racedirector** to paint a far more accurate picture of what is happening on track. Given we know the location (lap distance) where these events are occurring we can attempt to evaluate if a cluster of incidents, that seem to represent a large incident, are taking place between multiple cars at the same point on track, or simply if the actions of single car warrant an FCY *(something not possible with v1)*.

Also, a new addition to **v2** is the use of repair flags (aka meatballs). A configurable value is now assigned to a car carrying a meatball, a value which linearly reduces as the affected car recovers from the site of the incident (both in terms of distance and time).

This all results in a combined weighted score which is compared to a user configured value that represents a threshold for triggering a FCY.

4. Configuration

When started, the application reads in various parameters from a highly customisable configuration file. This file allows users to assign weighted values to various event (incident) types, thus directly affecting the level of severity that **racedirector** applies to each type of event.

To understand how to assign values the user must understand that **racedirector** is looking for 1x incidents which, in the application, carry a weighting of 1x. Loss of control (2x) and heavy contact (4x) in-sim both carry a 2x weighting. The rationale behind effectively removing 4x's from the equation is that not only do both cars get the 4x (in result of car-to-car contact) but these can occasionally feel somewhat harsh. Having a SC triggered by a 4x spurious contact would be unsavoury, and also often does not indicate a crash and significant loss of control of a car.

Understanding that these 1x/2x values are determined as above allows us to weight the following events accordingly (example values provided in brackets):

- Stationary car off track (0.5x)
- Stationary car on track (1.7x)
- Reversing car off track (0.8x)
- Reversing car on track (3x)
- Slow car on track (1.7x)
- Car going wrong way on track (3x)
- A car with a repair flag [aka meatball] (6x)

The addition of hot zones to **racedirector** **v2** have resulted in the above values being configurable for use in hot zones..

Now that we can quantify an event per car, we can combine these to see if they cross a configurable threshold. If they do surpass this number, a FCY is thrown.

As an incident can unfold over a number of seconds, the **admin** app can be configured to determine the time window in which events can be used towards assessing an incident, as well as maximum allowable distance window between incidents.

E.g. Two cars spinning (and getting a 2x each) must be within 100m metres (item a) of each other on track with both events happening within 8 seconds (item b) of each other.

Both items a and b are fully configurable by the user and different values may be required at different circuits.

Two configurable options related to temporarily increasing the threshold needed for SC exist. One is a multiplier applied on the first lap of the race, the other affects the first lap of a restart. The reasoning behind these are that in real-world racing first lap incidents are usually viewed in a relaxed manner by race control. We feel users may wish to apply a similar logic for lap one of their races, and for restarts too, in the hope that a race does not end up being one of constant pacing.

Configurable items now exist relating to data retrieved from the remote server (as fed by the **client** app). They are:

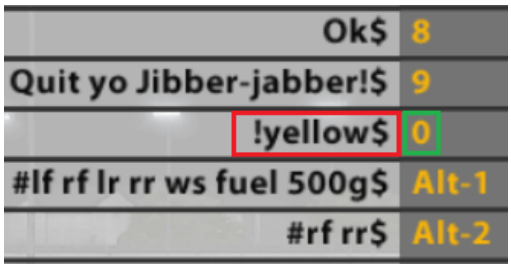
- Roll (2.5)
- Roll rate (5)
- Pitch rate (5)
- Z-Velocity (8)
- Combined (12)

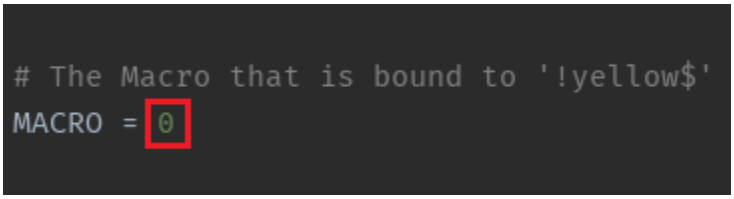
When the **client** app informs the **admin** app (via the remote server) that any of these values have been breached, an FCY will be thrown.

The Combined value aggregates certain values in order to ascertain a car being out-of-control but not crossing an individual threshold value.

5. Usage

In iRacing hosted sessions only race administrators have the ability to throw a FCY. **racedirector** executes the caution call by using an in-sim chat macro. Users must set one of the available chat macros in-sim to **!yellow$** (below left, red box) and then set the chosen value (below left, green box) in the configuration file (below right, red box).





When a FCY is required **racedirector** sends the chat macro assigned to the sim. As the sim has been set to chat "!yellow$" when that macro has been called, the caution call is executed. If this is not set properly then **racedirector** will not function.

The application can be started in advance of the race, so the user need not worry about it and can instead focus on other important actions. If the application were to terminate mid race, the program can be restarted without issue. Likewise, if race control wants to amend the configuration file values mid race, the altered values will be picked up on a restart of the application. It is **highly advised** not to change these values mid race unless the right to do so has been clearly communicated to all drivers in advance of the race.

6. Logging

racedirector runs using a CLI (Command Line Interface), all incidents detected are output to the console in real time for monitoring by race control. This output is also written to a race log text file.

When the race session in-sim ends (a state change is detected by **racedirector**), a full debrief file is written to a text file. This is a human readable summary of the race, seen through the eyes of the application. Both the debrief and the race log contain timestamps of events. Used in conjunction with a race replay (.rpy) file, all actions taken by **racedirector** can be reviewed post race by anyone in possession of the files. We highly advise that race control release these documents to the teams post race so that the maximum level of transparency is given around any actions the application took, or did not take.

Reviewing incidents post-race also allows race control to evaluate the values used in the configuration file. These can then be amended to more desirable values for future races, if race control deemed the application to be too trigger happy when calling FCY, or opposingly, not reactive enough to on track events.

7. Wave Around Advisor Integration

A separate application, created by the same developers called WaveAroundAdvisor has been in use alongside **racedirector**. This app analyses the train of cars behind the Safety Car under FCY and informs the user which cars require a wave around.

racedirector and WaveAroundAdvisor have now been amalgamated into **racedirector v2**, all running under the **admin** app.

Not only are these functions now integrated but the identification and execution of wave arounds are now fully automated and require no human (steward) intervention.

Wave arounds are currently applied inline with the FCY rules in use in the **trophi.ai** endurance series by FTR Events.

The following new features now exist natively as part of **racedirector v2**:

- Automatic extension of pacing laps where necessary
- In-sim pit open/close messages from @RACECONTROL
- Automatic wave arounds
- Detection of cars entering a closed pit entrance
- Reduction of pacing laps where there are no wave arounds
- Quick Cautions

Quick Cautions

racedirector v2 can now determine automatically the requirement for a Quick Caution (as per the rules of the **trophi.ai** endurance series by FTR Events). When required, the **admin** app will:

- Notify drivers of a Quick Caution
- Inform drivers of a closed pit lane each pass of pit entry
- Detect cars entering the (closed) pits
- Automatically apply wave arounds earlier in the FCY sequence
- Reduce pacing laps where possible to get back to green flag racing as quickly as possible.