ISIMOTOR2 AI GUIDE

By: Exefune

First to all, I want to thank the community of Rfactor, GTR2, Race 07, F1 99-02 Challenge, and Automobilista 1, thanks to all of their investigation and work, it's where I found all of the information to make this guide.

Also, I would like to thank now the people that helped me directly or indirectly, such as Shovas, The iron Wolf, TShirt, Speednut357, MJQT, jgf, Matthew Sibanyoni, Niels Heusinkveld, barryg, VonDutch for its tutorial. And the people that tried my settings and informed such like gt_gamrx22, Iker Vera and Mitzu Community by people like sigmaengine2099, xplotao and Mitzu himself

Thanks to Crimson_764 for fixing all of my English errors (I'm not native sorry)

And really, thanks to all of you, that helps make this possible.

Now we start with the tutorial and documentation:

ISIMOTOR AI is divided into a lot of files, with values spread everywhere so maybe if you want to do something, maybe only one value won`t help you, so in this documentation, its going to be divided into different sections with each file.

Also, I am going to try to cover all variables that are general in isimotor2 games, but some variables are exclusive to X games, so for that, I divided into some names:

SB: all Simbin Games
RF1: Rfactor 1
AMS: Automobilista

F1C: F1 Challenge 99-02

M/S: meter per second

Km/h Kilometer per hours

This is all the files that interact with the AI

PLR>AIW>HDV>TBC>RCD

And some other indirectly like GDB, VEH, etc.

PLAYER FILE (.PLR)

First, we start with the “most” simplest one, here its defined all user options but have some hidden AI settings that generally you can access to the in-game menu, so we're going to go here and you can change to your liking, if you wish.

Generally, its under UserData/YourProfile/YourProfile.PLR

Open it with any text editor like notepad

Now we can start (I recommend use ctrl+b to search each line we're gonna adjust)

QUICK AI Driver Strength: it’s just the AI driver strength that you can found in the game slider

AI Power Calibration: this is not known what it exactly does. There is a theory that says, this influences the factors depending on the % of the strength, I recommend leaving the value at 7 or 4 or 0.

AI to AI Collisions: This one calculates the collision of AI. At higher values, it makes the AI have less contact. I recommend a value between 18 and 40 (although Automobilista uses a value of 80, so it's unknown if it is possible to use this 80 value in other isimotor games). In some cases, a high value makes the cars too swervy because they want to avoid collisions, so it's better to have a value between 18 and 30 for those cases, but you can choose the number that you want.

From VonDutch:

Player Car Equal= 0 means the AI cars will see more variance from 1st through last. If you set it at 1 the AI will see less variance between the fastest and slowest cars.

AI Realism: this is the aggression slider basically, higher values, more aggressive AI, this is recommended to change in GTR2, VonDutch recommends a value of 1.30000. In other isimotor games you can just adjust the aggression slider and it's gonna work fine depending on your RCD files.

CrashRecovery: this one makes you flip your car when upside-down, also makes AI stay more glued to fast line especially in rain conditions where you're gonna see more cars going in dry line.

Speed Compensation: seems to make artificially more close racing, its not known how it works, only some information can be gotten from Rfactor that say have a range of 0-10

(Exclusive to RF and AMS): **Speed Comp Dist**="500.00000" // <= 2.0: fraction of track length, > 2.0: actual distance in meters for max speed comp.

Autocalibrate AI Mode (only in F1C and SIMBIN games, Rfactor and AMS have totally different purpose so I'm gonna explain those in an exclusive section): basically, makes the AI go more close to player performance, it seems to use your qualifying times to calibrate it and seems only work when Strength<100 (further tests are recommended).

Following variables can be overwritten in rfactor and automobilista in .hdv files, sadly other games requires you to adjust here in .plr so do with caution

AIFuelUsage: this affects how much AI use fuel; its better to leave at 0.9, but if you use cars that use a lot of fuel like Turbo cars, its better to use 0.3 or 0.4

AI Brake_Power Usage: higher values make the AI brake later.

AI Brake_Grip Usage: higher values make the AI brake harder.

as the ISImotor AI website says: Increase (decrease) when AI brakes too early (too late) before corners

VonDutch Recommendation: AI Brake Power Usage="1.20900" // Fraction of theoretical brake power that AI attempt to use // This one is important. By raising this number above 1 it will help to get rid of the special brakes the AI seem to have. It fools the AI into thinking that they have more brake power than they do. I have gone as high as 1.5000 but back off when I see the AI braking too deep and crashing. This will vary from track to track.
AI Brake Grip Usage="0.95550" // Fraction of theoretical brake grip that AI attempt to use // this is the amount of braking grip that the AI will attempt to use, the lower this number is the longer it will take for the AI to slowdown. Conversely the higher the number the shorter the distance for the AI to slowdown. Used in combination with the AI Brake Power Usage figure this will really allow you to tune the AI to brake more humanlike. (Should be always lower or a little lower than Brake Power Usage)

AI Corner_Grip Usage: Determines how fast an AI takes a corner. As ISImotor AI website says: Increase (decrease) when AI are slower (faster) around corners than they should be

A good practice by Niels, is to set AI brake and AI corner using Motec or a telemetry tool, let the AI drive your car on a track with a good AIW and then see how they brake, take turns and accelerate to adjust each value to be more closer to humans.

AI Max Load: Quote originated from ISI; "The default value above defines the upper bound of the range of the force in Newtons that the car will exert on the road. This value is used only to create an internal table for the physics system."

Now in practical effects, this affects how the AI calculates their braking and also how they try to use the track when overtaking other cars. I recommend to only leave vanilla values but you can try to decrease a little, say around 37000. I made the following formula with the help of an RF2 spreadsheet and ChatGPT, so this means it’s not perfect but i hope it can help modders that want a more exact value for them.

((Mass+Fuel)*9.80665)*2.5= AI MAX LOAD

Remember that with fuel, each liter weighs 0.75.

(In case of games that don't allow to put those lines in individual HDVs, use a car with a lot of weight in the game)

For example, in GTR2 the Saleen is a heavy car, and its tank capacity is 100 liters.

So we do this
((1250+ (100*0.75)*9.80665)*2.5=32484.5281

And now I round the number up to
32500.0000
In case you notice the AI start to go out of track or brake too late after adjust this, add 5000 to the result until they stop going out of track.

AI Min Radius: The tightest that AI can turn, should be left as default but if you are willing to put in a more precise value, then you can use this formula from RF2, to help improve AI Lap Times.

Wheelbase*5.5= AI Min Radius

In gtr2, I use the Viper as an example
2.880 × 5.5 = 15.84

So for GTR2, the ideal value is 15.84
(in case you can`t use hdv values (like SIMBIN games), use a car with longer wheelbase)

Following values are exclusive to RF and AMS

AI Limiter: lower values, make the race more spread based in RCDs, meanwhile higher values make more close races. I recommend using very low values like 0.01000 to 0.50000, also take in account, when its 0.0, this is gonna make them race always fast and disable the MinRacingSkill from RCD.

AI Mistakes: this make AI make more mistakes like missing braking points, i recommended very very low values like 0.1 or 0.01, due that still AI can commit some hardcore mistakes.

Auto Line Smooth: this is gonna sound illogical. But please put this line in 0, this was in reality a dev tool and should be used with caution, i don't know why ISI leave this in 1 from default, but generally make race really boring, 3 meanwhile make entertaining but with a lot of problems. I'm gonna leave Mike Z's explanation of this, although it come from a RF2 Forum.

“This parameter is really intended as a development tool. Yes, we enable it for our public gold release... but it's operation should be considered "unsupported" for reasons I will get to in the long explanation. It's purpose is to give the track designer a tool to possibly smooth out an un-optimal path....which he would then save and review. It is possible that applying this algorithm will actually make the AI drive the path slower/worse
When we load up our tracks and cars, we have EACH CAR run a "simulated" lap to gather our best guess information about what lap time he should expect and min and max speeds he should expect as he (either player or AI) travels around the track. If this parameter triggers the smoothing code, we then, for each path specified, make a list of all waypoints along with the expected speed OF THIS PARTICULAR CAR on that path. We order than list from slowest to fastest and then go through, waypoint by waypoint and adjust it's position by a couple of centimeters left and right laterally along the path to see if that raises or lowers the speed of THIS PARTICULAR CAR through this waypoint, the waypoint before, and the waypoint after the current waypoint. We move the path to the position that "optimizes" those 3 speeds and then we move on to the next slowest waypoint in our list... and on and on until each waypoint has been processed. Obviously, this is not a bulletproof algorithm (I'm sure you can already see plenty of issues with it). For one thing, if you are racing multiple car types in your race, it would "optimize" the path for each car type...one after another....I could see the final result mutating beyond recognition, so be especially wary of using it for multi-class racing.”

As he explained, this should be used with caution by track designers (I need to test this if work with the AIWCAM version of RF1), only in some tracks I found a value of 3 make races much better, I think its due how they build the left and right patch, but if that's not the case then it should be left at 0. Also maybe it can help your .ini files from Autocalibrate AI but more testing is needed to see if it affects those files.

So, this covers all PLR Sections, now we move onto:

Driver TALENT FILES (.RCD)

Now, this is going to be divided into two sections: The New RCD system (Rfactor 1 and Automobilista) and the Old RCD system (F1 99-02 Challenge and Simbin). Since they use different ranges and different variables, although it seems you can use variables that appear in F1C in Rfactor and vice versa, this is not confirmed due to a lack of debug tools

New System RCD:
(thanks AMS and RF comunity)

Aggression: As this number increases the AI will give other cars less room, both while passing and while following. It also increases the frequency at which they try to pass and increases the threshold they are willing to endure before giving up a pass. This interacts with the Aggression slider (UISlider*Aggresion) Recomended values by AMS: 60 and 80, but I feel that you can play with drivers around 80-90 range if you want more fights (useful for Stock Car Racing, or also could be used for other series, but proceed with caution) and going beyond 90 or 95-99 will result in; "peculiar drivers looking for a gap always or not caring of crashing out."

Composure: defines the frequency of "bad zones" and also frequency of mistakes (if AI Mistakes>0), lower values make worse driver (also lower value make them worse in they throttle), recommended ranges: 66-98 but AMS recommend stick around of 90 for average drivers.

Speed= the general speed of AI, very important, interact with the AIrange from AIW that from default are 0.1, so your speed values are limited with a 10% range. Unless you increase to 1.0 making speed depend 100% on rcd value. AMS range is 65-99 but remember how you set up the AI range, as if you use higher AI range or 1.0 then better to use a more close range like 80-99

Courtesy: (thanks SpeedNut) How respectful the AI drivers are when being overtaken or attempting to overtake others. Lower values make them leave less space, often resulting in contact or crashes. Higher values make them more cautious and more likely to avoid collisions. In AMS, this also interacts with the new blue flag behavior making them react more quick to blue flags. A good practice is to assign higher courtesy values to slower drivers so they yield more easily, while faster or top drivers should have lower values to encourage more aggressive overtaking attempts against slow cars.

Crash: % of times a driver may crash in the race, only used when the race is simulated.

Recovery: its seem this line is just a left-over, according to Niels this doesn’t seem to be working in Rfactor, I also tried to see if in GTR2 and F1C if it worked but I don't see any changes in those games either. Some people in AMS claim this makes the AI try to avoid any potential crash but this doesn't seem to be the case, either. Others claim that AI try to regain its position back but I really doubt this, so leave this value at 5, 15, 60 or 99 unless you found out what is the actual purpose of this line.

Reputation: Thanks to **Niels Heusinkveld**. This line along with Speed calculates an Intimidation value. It seems that higher values, make the AI try to defend their position more, while lower values make them a little more cowardly to overtake. Stick around 99 for top drivers, 90 for average and 70-90 for bottom drivers. (But you can play around with this range as you like, because there is not a lot of information available)

CompletedLaps: only used when you skip practice and qualifying, I just stick around 80 to 100%

MinRacingSkill: 0-100% When the value of the AI Limiter variable (from the PLR file) is > 0.0, the AI drivers go through cycles of optimal and sub-optimal driving where their driving skill drops to MinRacingSkill * Speed. (Set the AI Limiter variable to 0.0, and the AIs will always drive as best as they can... on every lap.) So, for example, if you have a driver with Speed = 50, an AIRange in the AIW file = 0.2, and MinRacingSkill = 90, this type of AI will generally drive at (0.8 + 0.2 * 0.5) = 90% speed (really very, very slow... but this is just an example). However, sometimes it will drop to (speed * MinRacingSkill = (0.9 * 0.9) = 81% when having a bad lap... that's extremely low, but this is just an example with easy numbers. Stick to around the 95 value and don't go below of 80 because it's gonna make the AI go very slow when they are in “bad zone”

AMS range if you have curiosity: 60-99

QualifyingAbility: This actually works in rf1 (possible in AMS also) and basically its when you skip the qualifying session, this is gonna calculate the average position of the driver, lower values put the AI in higher positions on the grid, while higher values make start more from the bottom (thanks to Whils)

AMS ONLY (thanks isiMotor2 Community AI Documentation)

TireManagement:

The lower the value, the more the AI will wear its tires.

Suggested range: 40-95 (80, 85 and 90 are the most common values)

StartSkill

The lower the value, the more chance AI driver will bog down at the start.

Suggested range: 20-90, being around 70 the most commom value. If you set this under 60, the AI driver will problably bog down very often.

Now this its the old RCD system used by F1C and SB games, I'm gonna use VonDutch tutorial as a base (Green lines) because he explained this really well but I'm gonna add some comment with // example

Henrik Roos

{

//Driver Info

Abbreviation=H. Roos // Driver Name

Nationality=Swedish // Self explanatory

NatAbbrev=SWE // Self explanatory

//Driver Stats

StartsDry=-1.0 // This is the average number of places gained or lost at the Start, positive #'s are places gained, negative numbers are places lost.

StartsWet=1.0 // keep these numbers at -1, 0 or 1. This applies to both wet and dry starts. This helps cut down on the chaos at a start. Far fewer crashes.

StartStalls=0.0 // % of time that the driver stalls his car at the start //dont go very far with this, its for simulate Stalls in the grid like what usually happened often in F1 during the 2000s era

QualifyingAbility=4.00 // This is the average qualifying ability of the driver. The number ='s position. //however it only seems to be used when you skip qualifying

RaceAbility=0.70 // The lower the number is the faster the driver is. The range is 0.0 - 6.2 //actually you can go beyond 6.2, but you are risking that the AI doesn't use full throttle, especially when you set the value higher than 15.00

Consistency=2.0 // The lower this number the less variation in lap times the higher the number you see more variation in laptimes

RainAbility=1.00 // Drivers ability to race in the rain. The range is 0.0 - 6.2//should be always the same or a little higher than RaceAbility value, it never should be lower however as it will make the AI really OP in drying conditions. Drivers that are great in the rain should have the same value as RaceAbility, normal drivers should just have a number less than 1, for example in this case, 0.70 +0.30=1.00, and bad drivers in the rain, should have a number higher than 1 (example 0.70+1.20=1.9)

Passing=97.0 // % of times driver completed a successful pass, not including pit stops or lapped traffic. Raising this number helps the AI pass more.

Crash=1.41 //% of times driver crashed //when you skip the race

Recovery=94.8 // amount of times driver carries on after a crash // not know exactly, same theory as the new rcd system or it’s just a leftover.

CompletedLaps%=99 // Self explanatory //only used when you skip practice/qualifying

Script=default.scp // refers to a script(????) unsure on this have not modified//refers to a old script that appeared back in F1 games.

TrackAggression=0.99 // this is IMPORTANT. This numbers needs to be at least in the hi .70's for even the slowest/worst drivers. Higher for the better drivers this is a important key to getting the AI to pass each other and you.//you can use higher numbers than 1.00 but be beware can break the AI, so use Von Dutch range from 0.70-1.00

// Increase attempted low-speed cornering by adding a minimum onto calculated speed.
// Reduce attempted high-speed cornering by multiplying speed by a number less than 1.0.
// <adjusted speed> = CorneringAdd + (CorneringMult * <original speed>)

CorneringAdd=2.1 // Increase attempted low-speed cornering by adding a minimum onto calculated speed. Don't go too high with this number, make small adjustments from original values until satisfied

CorneringMult=.974 // Reduce attempted high-speed cornering by multiplying speed by a number less than 1.0. Same with this, don't go to high here the higher this number and the

previous one the faster the AI will be up to a point then they have trouble staying in control of their vehicles. So small adjustments are best until you reach a point where you are satisfied.
// I always try to leave CorneringMult alone or have it almost all the same for all drivers, then you can increase a little bit the CorneringAdd if you want more slower drivers, and faster drivers should have lower CorneringAdd values but use caution with your adjusting as Von Dutch says.
//AI Throttle Control - how good they are at their own traction control upon throttle application //I have not experimented with these//me neither but here the jgf explanation and its seems its the more accurate
TCGripThreshold=0.7   // Range: 0.0-1.0
TCThrottleFract=1.2   // Range: 0.0-???
TCResponse=0.0   // Range: 0.0-???

From jgf: Threshold seems to be at what point in the tire slip range they begin to modulate the throttle; "Fract" is how much they modulate; and "Response" how quickly they modulate. But this is conjecture based on my tweaking, and is not entirely consistent across all drivers/all cars.

//AI skill mistake variables
MinRacingSkill = 0.98   // this refers to the minimum amount of skill that the driver will run at //same as I said in the new RCD system, stick to around the 0.95 value and don't go below of 0.80 unless you want drivers going really slow
Composure = 0.002   // The maximum amount of time the driver will make a mistake //it seems that in GTR2 it is the inverse; when more closer to 0, the best the AI is gonna be, this defines amount of time AI make mistakes and also the frequency of entering bad zones of driving (basically going more slow to the minracingskill point)

//AI ColdBrain variables   // "Cold Brain" refers to the beginning of a Race or Qualify session It tries to simulate the AI on cold tyres
RaceColdBrainMin=0.94   // This is the minimum amount of speed (based on a figure calculated by the game) the driver will run in a race
RaceColdBrainTime=100 // This is the amount of time the driver will run that speed in a race
QualColdBrainMin=0.94   // This is the minimum amount of speed (based on a figure calculated by the game) the driver will run in qualify
QualColdBrainTime=155 // This is the minimum amount of speed (based on a figure calculated by the game) the driver will run in

}

So remember talent files are one of the important parts to getting faster, more consistent AI and a better game experience. Use this as a guide for creating your own talent files or modifying others. Try to keep all of them fairly even, it helps to not have huge differences between the AI drivers. Time to move on.

Finally we have ended the RCD part, so now we can jump to the core of the car:

HIGH DEFINITION VEHICLE (.HDV or .HDC)

This file define the majority of physics of the car, but we're gonna focus only on the AI.

I'm gonna use some MJQT and VonDutch bases in some variables and add my comments with //example

AIMinPassesPerTick: higher number make more realistic AI suspension simulation but at the cost of more CPU Usage. Should be between 4 to 8 for most cars, I generally stick to around 6 for my preference. But never use higher values because if you use a lot of CPU, AI actually become worse (and don't use 0, as this makes the CPU work hard and use the highest calculation possible.)

AIRotationThreshold: When the Yaw of the car rotation is higher than this value, the game increases the frequency of physics updates for that car. This helps the AI detect and correct for potential loss of control (e.g., oversteer or spins) more quickly. By increasing passes per tick, the AI can make finer adjustments to steering, throttle, or braking to stabilize the car. Once the rotation rate falls below the threshold, the game reverts to standard physics updates to reduce computational load. According to **SpeedNut**.: "*Refers to the threshold for the angle of the car's rotation in the air at which point the game engine will consider the car to be spinning out of control and trigger a crash or a spin. When a car becomes airborne, either due to hitting a bump or ramp, or due to a collision with another car, it may start to rotate or spin in the air. The AIRotationThreshold parameter determines how much rotation is allowed before the game considers the car to have lost control*."

I recommend to use a value from a similar car that have good settings.

AIEvenSuspension: Averages out spring and damper rates to improve stability (0.0 - 1.0). The use of this variable would increase the disparity between AI and player physics. //always try to use 0.0 because it's gonna make AI have more separated physics than player, unless you don't have any other option.

AISpringRate: Spring rate adjustment for AI physics is used to soften or stiffen the suspension for AI vehicles. Decrease this to soften spring rates to keep the AI stable. 1 is equivalent to the setup file or default setup. //try to use 1.0 or 0.9 to be more similar to the player unless AI is really unstable, then use 0.5 instead.

AIDamperSlow: Contribution of average slow damper into simple AI damper. This simplifies dampers when used for AI. //act when AI have weight transfer like when they brake or steer

AIDamperFast: Contribution of average fast damper into simple AI damper. //act when the weight moves really fast, like when they hit a bump

Lower values make them be more soft and 1.0 make them be the same as the player.

You should use a value from a mod that work well, I try to only touch the dampers so that they are more close to human driving, but if that doesn't work well, you can touch the AISpringRate and AIEvenSuspension.

From Yoss from AMS comunity:

AIDownforceZArm: Hard-coded center-of-pressure offset from vehicle CG.
AIDownforceBias: Bias between setup and hard-coded downforce value (Range: 0-1). 0 is most realistic, 1 is least realistic. A value of 0 will ensure the AI do not have more downforce than the player, This can be useful to improve AI stability though, especially if they have problems at high speeds.

As said here, this makes the AI more stable at high speed, but the best way is to make a proper setup and leave it at 0.0.

AITorqueStab: The numbers represent how much force is needed to deflect the car from its course, longitudinally, laterally, and vertically

jgf recommends:

Run some practice sessions with the AI in the same cars as you and do a little "paint swapping"; if your car reacts more than the AI, lower the values, if the AI react more than you, raise the values. Set this too low and the slightest nudge will send the AI out of control.

// Follow jgf's recommendation, but remember this also interacts with the previous values we touched

From isiMotor2 Community AI Documentation

FeelerFlags=15: This will create the most possible accurate collision detection system, but you'll need to use the collision feelers. This part would be optional, since you'll need 3DSimEd to get the real car dimensions for each competitor. But, if you willing too, it's a worth effort.

Procedure to follow to improve collision detection of AI cars:

> Delete the // (comment marker) from the collision feelers lines and remove the line FeelerFlags=X from the top of it, since it's already informed at the beginning of HDC file.
>
> You'll notice there's a bug in most of cars I've seen so far which is the wrong signal for FeelerTopFrontLeft, FeelerTopFrontRight, FeelerTopRearLeft and FeelerTopRearRight.
>
> > Note that the coordinate system is strange (at least for me): +x = left, +y = up, +z = rear. The lines below are already fixed, so pay attention on it. Left is always positive and right is always minus!
>
> To know the real car dimensions, you'll have to open it in 3DSimEd. Take note of dimensions, subtract 10 cm from each one (helps to prevent the AI to AI crash without significant visual glitches), change the values above (double checking the signals + and -)
>
> - Remember to leave the rear-wing out of the box!
> - It's not nice to see the car turning over based in the rear-wing position, it's preferable to leave the wing going through the tarmac than the strange "rigid rear-wing" effect when turning over

If you have enough patience, the amount of AI to AI crashes and the physics reaction will improve a lot by following this procedure. If you don't use feelers, the game will create a automatic box around the car which could be pretty rough for calculations.

The section of the HDC file relevant to collision detection might look something like this:

Example:

```
//FeelerFlags=15 // or remove this second instance or simply leave the "//" infront
FeelerOffset=(0.0, 0.0, 0.0) // leave it alone, it won't be used.
FeelersAtCGHeight=0 // Set it to zero. You'll provide all real corners coordinates for the car.
FeelerFrontLeft=(1.003,0.384,-2.554) //front-left corner collision feeler
FeelerFrontRight=(-1.003,0.384,-2.554) // front-right corner collision feeler
FeelerRearLeft=(1.003,0.384,2.586) //rear-left corner collision feeler
FeelerRearRight=(-1.003,0.384,2.586) // rear-right corner collision feeler
FeelerFront=(0.064,0.384,-2.639) // front side collision feeler
FeelerRear=(0.064,0.384,2.587) // rear side collision feeler
FeelerRight=(-1.039,0.384,-0.247) // right side collision feeler
FeelerLeft=(1.039,0.384,-0.247) // left side collision feeler
FeelerTopFrontLeft=(0.478,1.540,-0.298) // top front-left collision feeler
FeelerTopFrontRight=(-0.478,1.540,-0.298) // top front-right collision feeler
FeelerTopRearLeft=(0.652,1.511,2.323) // top rear-left collision feeler
FeelerTopRearRight=(-0.652,1.511,2.323) // top rear-right collision feeler
FeelerBottom=(0.064,0.249,-0.247) // bottom feeler
```

It seems as if the AI use feelers to calculate collision avoidance, also mods should need to have good collision file to help this.

RearBrakeSetting: define the brake bias, for example a value of 36 is gonna be 64-36 brake bias, why is this important? Because AI prefer more stable braking, so this can help AI to brake easier, but its not needed to change this in the hdv because you can make AI setups as we're gonna see in another section, and from that you can use a more forward brake bias. Also in Rfactor and AMS you can make exclusive AI upgrades where you can adjust this.

RFACTOR AND AMS VARIABLES ONLY

AISlipReaction= (50.0, 30.0)   // Prediction factor for front wheel grip loss (higher numbers increase sensitivity), how quickly AI increase throttle after grip loss has occurred //nothing new needs to be added here. Test and see how AI behave, generally faster car requires more higher numbers, but those values already work well. Lower the second number if you see the AI do a quick snap when they accelerate.

AICornerReductionBase=120.0   // (pointspeed/this number)= % deceleration we can expect through a point // Prediction of how much AI need to slow down, also according to some guys this interacts with AIAimSpeedsPerWP but it is not confirmed yet. I recommend to use this value as a default and play with it. (Generally slower cars use lower values meanwhile faster cars requires higher value)

AIAimSpeedsPerWP= (30.0, 54.0, 68.0, 82.0, 100.0, 120.0, 150.0, 185.0) // Speeds at which to look ahead X waypoints (spaced roughly 5 meters apart)

This one is tricky, basically makes the AI start to skip waypoints when they go more faster than the speed specified in each **Step**, each Step skips 5 meters of waypoint, we have 8 steps, so its stepX5, first value represent 5 meters, second 10, third 15, four 20, five 25, six 30, seven 35 and eight 40 meter, this help to calculate the throttle and braking and can improve lap times. **Its recommended to use a value from a base car, and only fine tuning it, for example if you notice AI cut too much on a turn, then increase only the first 2 values.**

The speeds are in M/S so you might be wondering, why almost all cars use exaggerated higher values (making them never use those steps), it is because it guarantees that they don't cut the track if they reach that speed, and it's more noticeable on oval tracks, generally making them use only 4 or 5 of the steps instead of the 8 that is allowed to be used. So you may want to choose the amount of step the car is gonna have allowed to use, considering the speed they reach in corner (unless you want to do a lot of testing with the 8 steps, in this case make sure that the last 5 steps are really close to the maximun speed of the car to guarantee they don't use it in corner, but have caution with this approach, as results can end badly), the last step you're gonna use should be the maximum m/s speed of car including the Draft. The rest you're gonna accommodate it, according to the speed that the AI can reach and ensuring that it doesn't cut turns (remember first two value should be configured for slow and mid-speed turns). I highly recommend to do two tests, one at a Monza or Silverstone style track (that has high speed turns and some slow turns) and another on a fast oval track like Indianapolis or Talladega (if you consider the car can race on a oval).

AIFuelMult: PLR file override for AI fuel usage - only positive value will override

AIPerfUsage: PLR file overrides for (brake power usage, brake grip usage, corner grip usage) used by AI to estimate performance - only positive values will override

AITableParams: PLR file overrides for (max load, min radius) used when computing performance estimate tables - only positive values will override.

Look in PLR section again to see how should be this adjusted, or leave it in -1.0 if you want to use default ones.

SpinInertiaAI: AI pitch inertia, used to deal with their lower physics sampling frequency, just paste the same value as SpinInertia for each suspension, this ensures AI use same Inertia as player, if this isn't found, then Game automatically *2 the Human Spin inertia, in some specific mods i see they do SpinInertia*8=SpinInertiaAI

BrakeTorqueAI: Different brake torque for AI, as they are not currently affected by cold or faded brakes, just use a Little lower value compared to player so they get a Little nerfed brakes. Example: BrakeTorque=3900 BrakeTorqueAI=3880

Tire Behavior Configuration (TBC/TYR)

I will be very brief in this part, as tires are a world of their own, so we will focus only on what is related to the AI. I must say that this is what I recommend touching the least, as it can lead to very poor results, so modify with caution.

Thanks to Yoss from AMS Comunity.

AISens: Simplified load sensitivity for the use of AI controllers. Linear instead of exponential like the player's car. It reduces the use of the CPU. Because this is linear, it would probably be prudent to establish this for a lower drop than the player's car, this is because most of the time is spent in the middle range of the loads instead of the maximum loads. This can be used to improve the stability of AI cars.

AIGripMult: Grip multiplier for AI. If the AI is too slow or spinning, these values can be modified to give them speed and / or stability. (1.0 Its same grip as player). I noticed some mods also decrease front grip and increase/decrease the rear one to alter the way the take the turns., so you maybe want to take this into consideration to adjust they grip.

AIPeakSlip: Slip of the peak AI tire, similar to LatPeak / LongPeak only for the AI. This does not change depending on the load, if the AI slips around a lot and is unstable, as a result, this value is reduced, especially in the back. Otherwise, if you do not have such problems, try to find the player's maximum average grade and use those values.you can also use more higher values to make AI go more faster or you want to force them to drift, but better leave same value as player or a little more lower (but not much because gona make them race really slower)

AITireModel: 0.0 = original AI tire model in terms of slip, 1.0 = more similar to player tire model. Try to be 1.0 or 0.9 but if you notice problems, then leave it in 0.30 or 0.40

AIPitThreshold: this defines when the tire have less grip (defined in WearGrip) than this value, then they should pit for new tyres, I recommend sticking around with 0.94 because it’s AMS recommended value. (Seems only to work only on Rfactor and AMS especially the last one because RF official content never used it, it's unknown if this works in GTR2 and Race 07).

AIWear: this defines the constant of wear that AI is gonna suffer, should be less than the player, generally I see the formula is; player wear/3 or playerwear/4 depending on the car. But some exaggerated cases use player wear/10 (not recommended). I also found (thanks to CMT) that Wear*0.29 give good results too. With the AIPitThreshold and the AMS variables of Tire wear in rcd, its better to first use the human wear rate and then test when you need to pit and when AI need to pit in those games.

Softness: this tell the game how soft the tires are, and calculates a DurabilityRating. It seems to not change anything, apart from it only works when you use a value between 0.5 and 1.0 (thanks to Niels for all this information). Anyway, softer tires should be always higher number, also 0.75 onward are reserved to wet tires I see generally.

And with this, we have now covered all physics parts of AI, now we can focus on other aspects:

AMS EXCLUSIVE PARAMETERS

Really Thanks to Matt that he passed me everything he know, the best values and the information below here.

In HDV

AIQualLapsperRun=5 //laps the AI does in each qualifying stint
AIWingDamageLift=0.30 //how much the AI lifts when the front wing is damaged

AIPassLatScalar= 1.5 //Amount of extra space in meters left at 0 aggression when trying to pass. I personally do not use this, it is unnecessary.
AISuddenMoveScalar= 0.6 //(estimated guess) Amount of extra space in meters left at 0 aggression when trying to defend a pass.
AIStartOffset= 2 //How long in seconds before the AI will merge into the normal racing line during a race start
AIStartMerge= 8 //How long in seconds merging should take during the first lap

In SRS file (series file)

(Not recommended to touch the values from here)

OvertakingRadiusRiskBypass = 600 - radius in meter, from this radius and above the AI will be bypassing, below it will not
OvertakingWaypointsAhead = 15 - Number of waypoints ahead of the current one to calculate the radius

BlueFlags = 1

AIBlueFlagCurveRadius = 600 ////in meters, curves with a radius under this value won't affect AI under blue flag conditions

AIBlueFlagCourtesyThreshold = 0.5 //// from 0 to 1 the amount of courtesy the AI need to lift when not obeying the blue flag is not punished. Blue flag is not punished with srs BlueFlag=1

AIBlueFlagReactionTime = 20 //// max amount of seconds for the AI to react to the blue flag

AILiftUnderBlue = 0.25 //// bonus of the throttle for the lack of courtesy of the driver to add to the final throttle of the AI

And now we can move to:

Artificial Intelligence Waypoint (AIW)

OK, we're gonna cover some basic things about AIW because everything in the AIW is related to AI in some way, especially because this defines the route AI is gonna take, the width of road, ETC.
All waypoints-related things should be covered in a separate tutorial, and also maybe some things aren't gonna work in x game, the information is very spread out and what I'm saying, may not work in your favorite game. So we can start

(Use ctrl+b, very recomended to search each line, if not, then add it).

WaypointSpan: the distance in meters of each Waypoint, modify it with caution or sometimes its better not add it. Generally I see it’s around 4 to 6, some Friends say they feel tracks with 4.x or lower are great as AI corner really smooth.

DrivingLines: when you use 2 or higher, it changes the behavior of the AI, acting more aggressive and doing bump drafting, and also making them race in 2 lanes (or the ones you specified). **Its recommended to only use 2 or higher on ovals for StockCar Racing only, because if you used it with CARTS or IndyCars; it's gonna end really badly.**

FuelUse: This one calculates the Fuel usage at that track used. This is very important as it tells you how much laps you have of fuel and for the AI.
People tested this a lot and there are multiple formulas you can use by using

the track length value.

tracklength*16 or tracklength*15.5 (someones do tracklength*21 but not tested)

AICautiousness= (0.9900): (appeared back in F1 2001 and GTR2 HQ Mod bring this in his mod): This controls how cautious the AI is on track. Lower values make them race more aggressively and fight for positions more, while higher values make them more cautious, especially noticeable in large packs of cars. The recommended range is 0.30 to 0.80. I suggest starting with a value of 0.50 — if you see the AI making too much contact, increase it; if not, you can try lowering it.

AIBrakingStiffness=(1.0000,1.00000,0.9000): This tells how aggressive AI should brake, each number is for each type of car (Rear Wheel drive,4WD,Front Wheel Drive), 1.00000 tells to brake into the last moment, you can lower a little this number for like (0.8500,0.8500,0.8000) and this is gonna tell AI to brake a little earlier, lower more this number and AI is gonna brake much earlier. It's a very important value and you can use it along with brake power/grip to find the perfect AI braking balance (better if we try to modify this value only)

Slowwhenpushed: this controls when the AI is touched/bumped, and applies the brakes. It's very recommended to ALWAYS add this line to the AIW and it should ALWAYS be set in the range of 0.05 to 0.35, but you don't want to be very low because it can end with the chaos of AI crashing and not slowing down, also some tracks like ovals its fine to put it at 0.00.

LaneSpacing: this define the distance between the left and the right lane, and is noticeable when formation lap is enabled. Also some say this define the width of the AI fast line but I cannot confirm this.

InsideAdjustment
OutsideAdjustment

Those lines are a fine tune of how wide/shallow the AI take inside/outside lines. I recommend to stick around this
InsideAdjustment=-1.5000

OutsideAdjustment=-0.5000

It's not known what the difference is if you use positive numbers, however outside always needs to be lower (in negative) number than inside as per

example here;
for Indianapolis the best value is -5.00000 for inside and -0.50000 for outside.

```
WorstAdjust=(0.8000)
MidAdjust=(1.0000)
BestAdjust=(1.1500)
```

Those lines define the speed of the AI depending of your strength setting. Worst correspond to 80% of strength, mid to 100% strength and best for 120% strength
Here is how it works. When you select a difficulty <=80, you are using the worst adjust, if you use a value <=100 you're using the mid adjust, and if you use a strength>100 you are using the best adjust.
For example, if you select 95% strength and you have in mid adjust 1.000 then you are doing 95*1.000*speed of the Waypoints and this is gonna define the actual speed of AI. Its very important to first adjust everything in 1.0000 and select 100% strenght, then you calibrate the mid adjust for being around 0.3 second of the best lap time of the series in that track and then you calibrate the best and worst adjust according to your preference. And it is important to be balanced so user doesn't need to adjust the strength for each track.(thanks jgf for all this information and recommendation)

AISpec: I think this line don't work in GTR2/Rfactor, but it seems to work in AMS. Basically this acts when you set strength<100%, AI is gonna be limited with these values. The first one limits the acceleration, the second one the speed, the third the cornering and the four the deceleration. (need further testing)

also according to **SpeedNut** in race07 do this (i never played Race 07, so i can`t test this sorry):
Refers to a set of parameters that define the behavior and performance of the game's artificial intelligence (AI) opponents.
The parameters are as follows.

- Grip level: This parameter defines the level of grip or traction available to the AI opponents. Higher grip levels will result in faster speeds and more aggressive driving behavior from the AI opponents.
- Path width: This parameter defines the width of the racing line or path that the AI opponents will follow. Narrower paths will result in more cautious driving behavior from the AI opponents, while wider paths will encourage more aggressive overtaking maneuvers.
- Speed: This parameter defines the target speed that the AI opponents will aim to achieve. Higher speeds will result in faster lap times and more aggressive driving behavior from the AI opponents.

- Braking behavior: This parameter defines the aggressiveness of the AI opponents braking when braking into corners.

AIDraftStickiness: This one is a pain because there is no actual best value, and this isn't gonna fix 'the AI don't overtake on a straight problem' but it can mitigate it, and I'm gonna talk with my preferences because some people prefer the value 3 to 5. This line defines if the AI is gonna want to change the car that he drafting or he should maintain with that car (or pull a overtake)

I prefer using a range of 0.00 to 2.50, sticking around 0.90 to 1.00, sometimes I feel better racing deleting this line (lol), but some people say that they have better racing with a value between 3 to 5. So its up to you to test this, I would take into consideration that AI maintain the draft until to just about the middle of the straight and also that AI don't wanna stay in the draft of a lapped car very much and start to overtake him.

Also it seems in AMS this was changed and most of the tracks use a value of 3, so take that also in consideration.

The way it works is, it adds this amount to the speed of the car in front (it's in M/S) and when the car that is following it surpasses this value, it then tries to overtake it, or to change draft partner.

AIRange: this line defines the difference between the AI drivers, remember this interacts with the speed/raceability variable from rcd, if you decide to adjust each rcd then I recommend to put this value at 1.0 or, a good practice i see from race07 smart AI is this: "0.5 for Nordschleife, 0.8 if ref time is superior to 2min, 0.9 if between 2min and 1min30".

RaceQualRatio: appears in both Rfactor and Automobilista games. However seems only the last one works. Lower values make the AI qualy more slower than race, higher does the opposite. More faster AI in qualy than in race is recommended to put at 1.005 for example, so AI is slightly more faster in qualy compared to race (remember you can do exclusive qualy setups for AI when we reach the Setup Section). Also something to note in F1C this variable also appears but does not work the same, this slows down the race pace compared to Qualy. So a lower value, would make AI in race more slower than qualy and viceversa. Should be really, really, close to 1.0 like 0.990 for example in that game.

SpeedOffset: this seems to act as an offset for the AI Cornering speed, its unknown what its original purpose is, as its seems do a lot of iterative processes (thanks Niels for that information), so I think its better leave this to 0.0000 or deleted this line, unless you know if this can improve AI in a way with this.

CheatDelta: don't seem to work in Rfactor and GTR2, Unknown if this works in automobilista.

Now according to SpeedNut in Race 07 do this (as said before I don't play Race 07 so I can`t test this):

Refers to a parameter that controls the amount of time penalty that the AI-controlled cars receive if they cut a corner or go off-track during a race.

The CheatDelta parameter specifies the amount of time penalty that the AI-controlled cars will receive if they cut a corner or go off-track. The value is specified as a number of seconds.

The three parameters used in a CheatDelta formula would be:

1. DeltaStart: This parameter specifies the start value of the time penalty for cheating or breaking the rules.
2. DeltaFactor: This parameter specifies the factor by which the time penalty should be multiplied, based on the severity of the infraction.
3. DeltaMax: This parameter specifies the maximum amount of time penalty that can be imposed on the player or AI-controlled car, regardless of the severity of the infraction.

SB EXCLUSIVE: I'm gonna copy paste what ISIMOTOR 2 COMMUNITY AI Documentation because explain this perfectly.

RaceRatio

RaceRatio is a percentage determining how much faster AI are during the race – compared to their baseline speed (e.g. in practice) as determined by WorstAdjust, MidAdjust, and BestAdjust. So, for example, RaceRatio=1.04 means that AI will be 4% faster in the race than in practice.

QualRatio

QualRatio is a percentage determining how much faster AI are during qualifying – compared to their baseline speed (e.g. in practice) as determined by WorstAdjust, MidAdjust, and BestAdjust. So, for example, QualRatio=1.04 means that AI will be 4% faster in qualifying than in practice.

GarageDepth=(3.0000)

This its the amount of meter that AI gona run when he leave its garage spot, increase this if AI steer to early or decrease it if AI crash into the pit wall due to not steering in time.

And this covers all AIW aspects, now we can go to:

Game Data Base (GDB File)

GDB files are used for defining the property and information of the track and the rules of the series you are running (SB and F1C this last one)

I'm gonna cover all related to AI plus some extra variables that seem to be not very well known (and work in almost all games)

Attrition = 30: this defines the wearing component of cars, which will suffer from retirements during a race. It's not very clear how it does work exactly and I don't know if this affects the human player, it interacts with the LifeTimeAvg from the engine file so depending on how you set that file (the general rule is to multiply the duration of the race by1.5 and LifeTimeVar generally is LifeTimeAvg/3 or /4)

Here is a good formula from F1C that really works well if you have the lifetime values set right:

(100/amount of cars)*amount of cars you want to failure

Here a example, in 1999 in Australian Grand Prix, 22 cars participate in the race, 10 cars DNF due to mechanical failure (the rest were due to accidents) so we do: (100/22)*10=45
so when you race, approximated 10 cars are gonna have a mechanical failure (not always, can vary, can be more cars or less cars but this give a good number for that GP specifically)

Also i not know if this value work with the mechanical failue in normal, only i tested it with scaled mechanical failure.

PitWindow = 42 This line appears in Nascar Thunder games, tells the AI to pit at each of the specified amount of laps (this case every 42 laps), i tested it in Rfactor and AMS and it seems to not work, unknown however if this works in GTR2 (not necessary though, so don't add this line)

GarageDepth = 2.00 Do the same as in the SB AIW files: specify the distance (in meters) that the car should travel from its garage spot before turning into the pit lane. Increase this value if the car starts steering too early, and decrease it if the car crashes into the pit lane wall.

AIDryGrip = 1.01
AIWetGrip = 1.05

This controls how much AI grip is gonna have on the track, higher values make them have more grip and race the corner more harder, only adjust it if you notice AI are slow (or fast) in turns (balance it with the adjusts from AIW), also you can use the AIWetGrip to nerf the AI in wet conditions (very useful in F1C if you don't want to adjust the wet lines from RCD)

PlayerFuelMult= 1.00
AIFuelMult= 1.00

These make player/AI to use more fuel (or less), useful if you want to adjust it for a specific track

PlayerTireWear = 1.00
AITireWear = 1.00

These lines, as the name says, control the wear of tires of the player and AI in this track, useful if you want to balance the wear for this specific track.

TireHeatMult= 1.0
RearTireHeatMult= 1.0
FrontTireHeatMult= 1.0

Control how much your tires heat, I don't recommend to use this unless you have an unrealistic mod with unrealistic tire heat.
(Something to note, these were tested in Rfactor and Gtr2, maybe might not work in F1C)

I see some mods like Race 07 Smart AI mod or GTR2 HQ mod add this line:

```
Drafting
{
  BaseDropoff=0.199 // higher number -> more drafting effect
  LeadingExponent=1.8 // higher number -> lower effect on leader
  FollowingExponent=1.9 // higher number -> lower effect on
followers
}
```

With this you can control the draft effect, you can add to each track or to the game.gdb for default value, remember Rfactor and Automobilista added more options and can appear in the .hdv of the car.

PitStopStrategies

This line defines the amount of pitstop a specific AI driver will do, and on what lap.
Here is an example from the overtake forum:

```
PitStopStrategies
{
Driver1 = 2 - 20,30
Driver2 = 2 - 25,50
Driver3 = 3 - 15,30,45
}
```

Driver names are the names that appears in the .veh (or .car) files, the number after = is the amount of pit stops they will do, and the number after – is the lap the driver will stop on.

Something to take into consideration, this will make AI fill its tank for the amount of laps for each pit stop, so if you have Michael Schumacher= 2 – 5,30. He will start the race with fuel for 5 laps, and then will refuel so he can reach to 30 laps, after that he will fill the tank for the amount of laps before the race ends. I've not tested this with time based races so this maybe might not work well in those scenarios.

Also for mods that don't allow refuelling, you should have this set to 0 so it'll depend only on tire wear. Like CMT mods
here an example

```
PitStopStrategies
{
Driver1 = 0
Driver2 = 0
Driver3 = 0
}
```

So this ensures the AI race with the tank full (or for the amount of laps the race will be) and only stop for tires.

```
  SettingsFolder = Track
  SettingsCopy = Grip.svm     // first "SettingsCopy" becomes
your default!
  SettingsCopy = track.svm
  SettingsAI = AItrack.svm
```

This tells the game what folder to create to save your setups, and also what setups to load for each specific track, the most important one is the SettingsAI attribute as this tells what setup the AI will load for this track and we're going to cover that in a specific section

```
   Qualify Laptime = 95.246
  Race Laptime = 100.926
```

These lines define the lap time the AI will do when you skip the session.

And those were the most important things you can adjust in the GDB, there are more options but they are self explanatory. We can move on finally to other things but first we're going to stop with a variable that appear in SimBin games

COMPARATIVE TIME (SB ONLY)

In each .Car File, there is a line at the end called `ComparativeTime`

In summary, it’s a percentage that multiplies the Qualifying and Race lap time when you skip the session, and adjusts it for each car group.

All top teams should have a value like 90, Mid group like 92 and lower group 94, also if this is for multi class racing, use more higher numbers like GT class 100 and NGT at 104.

SETUP FILES FOR AI (SVM)

One great advantage we have is the ability to assign specific setups for each AI. This is very important, as it allows us to control how they brake, their aero settings, tire compound, gear ratios, and other parameters.

You can assign a global setup by creating a setup file named ai.svm and placing it in the same folder as the .hdv/.hdc files or in the same .veh/car folder if you want a specific driver or team to use that setup.

It’s also possible to assign track-specific setups. To do this, check the SettingsAI line in the track’s .gdb file and name your setup file exactly as specified there. Then place it in the same folder as the .hdv/.hdc or .veh/car files, depending on whether the setup is meant for a specific car or team.

Something to note is that you can make a setup exclusive to qualifying or race sessions by adding .qual or .race before the .svm extension. For example: Brianza_Italian_RFE.qual.svm

So in summary, the format is:
trackname.qual.svm or **trackname.race.svm**

Also, when creating a setup, the svm file will also save the upgrade you selected in that moment (this applies to rFactor and AMS, where the upgrade system is available). This allows you to fix issues like AI running with a full tank during qualifying in rFactor by creating an upgrade that limits fuel capacity.

When making a set up for AI, the most important parameters to adjust are **aerodynamics** and **gear ratios**. Give the AI a slightly longer gear ratio than yours to prevent them from hitting the RPM limit constantly, which can damage their engines.

Also, remember you can now adjust **brake bias**. AI drivers generally prefer more forward brake bias. After that, feel free to test and tweak other parameters.

For qualifying setups, you might want to reduce the **radiator size** and **brake duct size**, and use a higher **turbo boost** setting. (Just remember to revert these changes for the race setup.)

Finally, you can also assign the **tire compound** the AI will use. However, keep in mind that the AI will first use the tire compound defined in the .hdv/.hdc files. After making a pit stop, they will switch to the compound specified in the .svm setup file.

CALIBRATING THE AI (AMS/RF ONLY)

An exclusive feature of rFactor/AMS is that you can train the AI on the racing line, allowing them to improve their lap times and their pace. I recommend creating a new player file with both fuel, wear, and failure turned off, and with a private session enabled.

Now, go to your PLR file and find the **Autocalibrate AI Mode** setting and set it to 1. You might also want to set the **AI Autoline Smooth** value to 1 or 3 to make their racing line smoother. Additionally, set **AI Mistakes** and **AI Limiter** to 0. After that, save the file.

Next, start a testing session and enter the track you want to train the AI on. Add an AI driver, and they will display a message like "Stay apart, human." This is when they start learning. After the AI completes 1 or 2 laps, press **Ctrl + T** to accelerate time. After a while, the AI will either return to the pits or indicate that they are finished with their training. They will then show you a meter indicating how much further they think they can improve their performance. The ideal result should be 0.0x meters, but depending on the AIW file of the track and the car's behavior, this might not always be possible.

Now, remove that AI and add another one. The new AI will reuse what the previous driver learned and continue from there. Once the second AI finishes, you can exit the game.

Go to the .veh folder of the car that completed the training, and you'll find a trackname.ini file. Copy this file and paste it into the car's .hdv folder. Now, every driver will use this .ini file and perform better on that track.

If, for any reason, you want to race with the same player profile you used to train the AI, return to the .plr file and disable **Autocalibrate AI**, as this setting makes the game not read the .rcd files.

General Considerations

You want to set the AI strength to your liking the one that makes you fight for positions and finish where you intend to. The **Aggression** setting depends on the series you're racing in and how you've configured the .rcd files. Start with higher aggression (like 100% or 90%), and then lower it by about **-20%** for each test if you feel the AI is overly aggressive. Remember to reduce it further for tracks with a poor AIW or tracks that tend to end in crashes, like Monaco.

Now, when you're in the game, don't skip qualifying. Simply press **Ctrl + T** and let the AI complete their lap times. This ensures they finish learning the track and have an appropriate starting position. I also prefer to **Ctrl + T** through the warm-up session, just to be safe. After that, during the race, I recommend not racing too aggressively with them in the start, as they tend to be in groups and “pretend to have cold tires.” Once you’ve passed that phase, race them like you would against human opponents, and you’ll notice they race well and provide a good challenge.

Conclusions

And that’s everything covered in this guide. We’ve gone through every aspect that can be adjusted. If you made it this far, thank you so much for taking the time to read it I truly hope it helps you.

Unfortunately, some things are impossible to achieve, like making the AI always overtake on straights or getting them to use more than two tire compounds. But with what we currently know, this is everything we can do. From here on, it’s all about fine-tuning each variable to make sure everything works properly.

As I mentioned at the start of this guide, thanks to everyone who helped me, to the community that has done so much deeper research, to my friends that support me, and to the developers who created these great games that we’re still enjoying today.

That’s all for now—hope you have a great race.
Goodbye! ^-^