GRAND PRIX LEGENDS ARTIFICIAL INTELLIGENCE TUTORIAL

LEE BOWDEN

REVISION 4.0
JANUARY 2006

INTRODUCTION

Welcome to the Grand Prix Legends Artificial Intelligence Tutorial!

This small booklet is my attempt to explain how GPL's artificial intelligence (AI) works and how we can adjust settings to make the AI more realistic and faithful to their real world counterparts. For those of us who do not race online, the AI are our only source of competition. I feel that the better we can make the AI, the more enjoyable will be our GPL experience.

This booklet is divided into seven parts. Part I will explore the AI in detail. We'll look at the files that we can edit to adjust the AI parameters. We'll spend a lot of time learning about how each parameter affects the AI driver speeds and lap times both during qualifying and the race. A review of how GPL models the performance of each car is included. Further, we'll explore a method that can be used to adjust the AI parameters so that the AI driver will perform as we want. Finally, we'll look at the best technique to control the speed of the entire AI field so that you as the player can compete effectively and win races regardless of good a driver your are.

Part II will show methods for quantifying a driver's real world performance based on their historical race records. We'll also take a quick look at how other researchers have handled this problem in the past.

Part III will apply the lessons learned in Part I to the performance requirements we derived in Part II to construct new AI driver parameter settings.

Part IV will test the new AI parameters settings versus the performance requirements to see how effective our settings are. Two methods for adjusting the AI so that they qualify closely with the historical race records will be presented.

Part V will apply the same methods and techniques we learned in the first four parts to the 1965 modification drivers.

Part VI will do the same for the 1969 modification drivers.

Finally, Part VII will take a look for the first time at the AI car reliability. It will examine which parameters can be adjusted to affect the car reliability and provide a method that can be used to make the AI cars have similar malfunction rates as their real world counterparts.

Before we begin, I'd like to thank David Wright of the Legends of '65/'67 website at http://fp.gplegends.plus.com and Ondrej Haderka also known as Kuratko of the GPL Seasons website at http://gplseasons.euweb.cz/index.htm. These two researchers inspired my own studies by their earlier work in exploring the GPL AI.

And of course, I want to thank Stefan Magnusson also known as Nenne of the GPL++ website at http://gplpp.com. Stefan and I collaborated in the past on two utilities, GPL Sound Player and GPL Image Viewer. Without Stefan's help, none of this would be possible.

Thanks guys!

So follow along. It's going to a challenging, but rewarding journey.

PART I

TESTS OF GPL'S AI PARAMETERS

METHODOLOGY

Throughout our discussions, reference will be made to ".ini" files which are nothing more than simple text files with a different suffix. These files may be edited with any text processing program such as Notepad. There is also the GPL AI Manager (GPLAIM) program developed by Peter O'Connor which will automatically make changes to the .ini files without having to use Notepad. I find this program to be very buggy, however, and prefer the simpler method of using Notepad to make the changes directly. Note that you must save the files as simple text files without formatting even thought they have .ini suffixes.

The method by which we can test changes to the AI parameters on their performance is simple yet time consuming...change one setting at a time, run a qualifying session and/or a race, then record the resulting time. Yes, it is simple, but also very monotonous as even on a fast computer, it takes time to modify the settings, load and run the GPL program, and record the results. I don't recommend doing this unless you have a lot of spare time!

To begin, the default gpl_ai.ini file that comes with the original GPL program is modified to use the following settings:

1. npt_override = 1.00. As will be explained in a later section, setting npt_override to 1.00 overrides the player's (that's you!) driving history so that we get consistent results.
2. disable_random_modifiers = 1.00. The GPL.exe code applies random modifiers to the driver parameters as a qualifying session or race is conducted. This is a nice feature to make the program more interesting and surprising; however, it can create havoc for testing by returning inconsistent qualifying and race results...something we don't want during our tests. Therefore, this feature is disabled. Note that the GPL.exe program code also includes some random modifiers that we cannot override. If we encounter qualifying or race times that are obviously different from previous results, we merely throw out that test and redo.
3. override_difficulty_hype = 1.00. The GPL.exe code modifies the AI hype parameter based on the simulation's novice, intermediate, or pro (F3/F2/F1) setting. Setting override_difficulty_hype to 1.00 overrides this feature so that we get consistent results.
4. n_ai_cars = 2. Once the GPL.exe program is started, you can set the number of AI drivers from a minimum of 5 to a maximum of 19. During our initial tests of qualifying and race performance, it's not necessary and even time consuming to run a full field of 19 drivers. Setting n_ai_cars to 2 overrides the program's setting and sets the number of drivers to 2. Unfortunately, the minimum setting that works is 2; otherwise we would set it to 1.

Next, the driver.ini file is modified so that Jim Clark is listed first in the file and is used as the test driver. This is done because Clark drives the Lotus which we will soon see is the overall fastest AI car. We could have used Graham Hill instead because he also drives a Lotus. Clark is set first in the list of drivers so that he is never bumped from the race as we lower the number of AI drivers. Next, we modify Clark's driving parameters of aggression, alertness, experience, hype, quickness, smoothness, qualifying, magic_grip, and global_hype_scaling so that each is set to a value of 1.00. Clark is now our baseline driver and all performance results are measured in comparison to this baseline.

Once the baseline driver's performance is recorded, we start making changes to driver parameters, one at a time, until we get a good feel for how each parameter affects qualifying and race times.

CAR PERFORMANCE

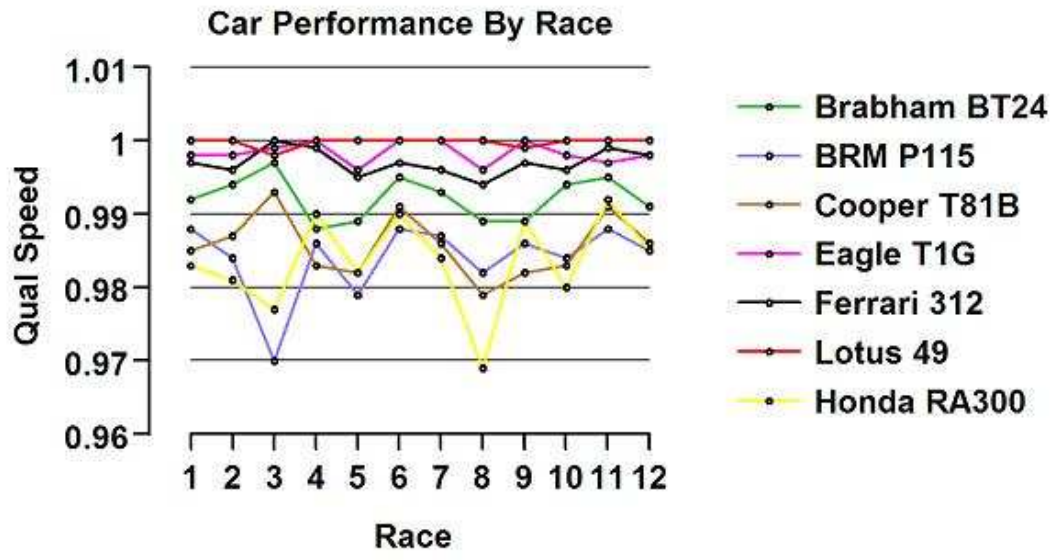
Each car as modeled by the GPL.exe program performs differently. This a great testament to the Papyrus design team's pursuit of realism and makes the simulation much more exciting and challenging for the player. After driving each, it is easily apparent how different each car really is. They certainly handle and accelerate/decelerate differently.

The AI cars also perform differently. At this time, it is unknown whether the AI car performance differences are a result of the many and varied car capabilities such as horsepower, frontal area, suspension differences, etc. or rather the performance differences are the result of a simple "fudge factor". In either case, these car differences must be taken into account when comparing actual driver historical performance and when calculating appropriate driver parameters for the driver.ini file. Otherwise, you will not get the performance you desire.

To test each AI car's performance, we simply modify the driver.ini file so that the lead driver for each car has all his driver parameters set to 1.00. In other words, we put the baseline driver into all seven cars and then measure his performance. Differences among the results then is a measure of the car's performance, not the driver's.

To complicate matters, each car performs differently at each track. Therefore, it is necessary to test each car at each track, record its relative performance with the fastest car's performance artificially set to 1.000, then average the results over all tracks. As you may know, Papyrus used the track at Rouen for the French Grand Prix; however, the actual 1967 race took place at the shortened LeMans track. Therefore, the addon LeMans Bugatti track is also used for our car performance comparison.

The following graph shows the relative performance of each car at each track including Bugatti.



The following table compares the average relative performance of each AI car at all tracks as modeled by GPL. The Lotus is the fastest car overall and is assigned an arbitrary value of 1.000. The other cars are "normalized" to the Lotus performance so that we can easily see the performance differences.

Average Car Performance

Brabham BT24	BRM P115	Cooper T81B	Eagle T1G	Ferrari 312	Lotus 49	Honda
.9925	.9844	.9857	.9987	.9974	1.000	.9838

This table compares favorably to Kuratko's tests which return almost identical results

The Lotus is the fastest car, but the Eagle is very close behind followed by the Ferrari. The Brabham is in the middle of the pack while the Cooper, BRM, and Honda bring up the rear. These results are consistent with those

obtained by actual player experience with the different cars. Remember that these are the performance differences as modeled by GPL. The actual differences are only as good as the model Papyrus used in designing the simulation. However from all accounts, it appears that they did a very good job in doing so.

AI PARAMETERS

The AI are controlled by settings in three .ini files and the GPL.exe file. The three .ini files are driver.ini, gpl_ai.ini, and track.ini.

1. The driver.ini file contains parameter settings for each driver.

    A. There are several miscellaneous settings:

        1). team_number--sets the driver's car. It is possible to have all drivers use the same car during a race. The car numbers are:
            a). 0 = Brabham
            b). 1 = BRM
            c). 2 = Cooper
            d). 3 = Eagle
            e). 4 = Ferrari
            f). 5 = Lotus
            g). 6 = Honda
        2). team_order--sets the driver's position within the team
        3). car_number--sets the driver's car number which selects the proper car graphics file (.mip) for display.
        4). first_name--self explanatory
        5). last_name--self explanatory
        6). home_town--self explanatory
        7). nationality--self explanatory
        8). helmet_color--self explanatory
        9). bump_order--unknown effect. As you reduce the number of AI drivers within the GPL program, the drivers are bumped in reverse order from their listing in the driver.ini file. The last driver is bumped first, etc.
        10). starting_grid_seed--unknown effect. Does not appear to influence the driver's qualifying position at all.
        10). photo--sets the photo that the program displays when the driver wins a race.

    B.. The most important parameters that control driver speed and behavior are:

        1). Aggression
        2). Alertness
        3). Experience
        4). Hype
        5). Quickness
        6). Smoothness
        7). Qualifying
        8). Magic_grip

Later, I will show how each of these parameter affects driver qualifying and race times in detail, but for now let's discuss the parameters in more general terms. First, no one knows exactly how changes to these parameters are modeled within the GPL.exe program. Yes, we can see the effect of changing hype, for example, on qualifying and race times; however, we don't know how these changes affect the actual AI driving style. Does increasing hype make the AI pass better? Does lowering hype decrease the AI's use of the car's cornering capability? These and many other questions can't be answered by merely changing the parameter and viewing the resulting qualifying and race times. Second, we do have some clues as to what each parameter affects. In the gpl_ai.ini file, there is the

[PARAMETER-SPECIFIC SCALING] section. This section apparently contains scaling factors that are applied to the individual driver parameters for use by the GPL.exe program. Examination of these scaling factors sheds some light on how driver parameters are handled. Third, note that some driver parameters apparently have an effect only on a driver's in-race behavior (such as passing ability) while others only effect the driver's raw speed.

The following table shows the scaling factors that are affected by each driver parameter. If there is an inverse effect, the table has an X in parenthesis.

	Aggr	Alert	Exp	Hype	Quick	Smooth	Qual
Desired_sep	(X)	(X)					
Trans_time	(X)					X	
Yaw_k1	X					X	
Lookahead		X	X				
Nominal_trac			X	X			
Adj_trac					X	X	X
Dlong_acc				X	X		X
Gearshift					(X)		(X)

Let's define each scaling factor first before discussing how each driver parameter affects them:

A. Desired_sep_scaling--modifies the longitudinal (fore and aft) and lateral (side by side) distance that the AI attempts to maintain between itself and an adjacent car.

B. Trans_time_scaling--modifies the time the AI plans to use to return to the racing line.

C. Yaw_k1_scaling--modifies the constant used for the car's yaw acceleration. This constant is not fully understood.

D. Lookahead_scaling--modifies the time that the AI looks ahead in making its forecast of the future.

E. Nom_traction_scaling--modifies the nominal traction circle which is initially set at 1.475 Gs in the [CAR CLASS PARAMETER] section of the gpl_ai.ini file. A complete discussion of traction circle theory is beyond the scope of this tutorial, but a simple explanation will suffice. Imagine a car's cornering, acceleration, and deceleration capabilities as being represented by a circle with zero G at the center. The car's cornering capability in G's is shown by the two sides of the circle, the car's acceleration capability is shown at the bottom, and its deceleration capability at the top of the circle. At any instant, the car's total cornering and acceleration/deceleration G forces must lie somewhere within the circle. Originally devised by the great CanAm champion, Mark Donohue, the traction circle is a graphical representation of how a car can corner at maximum G or accelerate/decelerate at maximum G, but cannot do both at the same time.

F. Adj_traction_scaling--apparently modifies the nominal traction circle which adjusts the car's cornering capability. Its exact effect is unknown.

G. Dlong_accel_scaling--modifies the the longitudinal (fore and aft) acceleration.

H. Gearshift_scaling--modifies the time used by the AI to shift gears.

Now that we've defined the scaling factors, let's look at how changes to each driver parameter affect them.

A. Aggression--as there is an inverse effect, increasing aggression shortens the separation the AI tries to maintain between itself and an adjacent car thus making passing capability better. Also, increasing aggression lowers the time the AI uses to return to the racing line. Finally, increasing aggression increases the yaw K1 constant whose effect is unknown. Obviously, decreasing aggression has opposite effects.

B. Alertness--as there is an inverse effect, increasing alertness shortens the separation the AI tries to maintain between itself and an adjacent car thus making passing capability better. Also, increasing alertness increases the time which the AI looks ahead in making decisions.

C. Experience--increasing experience increases the time which the AI looks ahead in making decisions and the nominal traction circle value.

D. Hype--increasing hype increases the nominal traction circle value and longitudinal (fore and aft) acceleration capability which make the car corner and accelerate better.

E Quickness--increasing quickness increases the adjusted traction circle value and increases the

longitudinal (fore and aft) acceleration capability thus making the car corner and accelerate better. Increasing quickness also decreases the gearshift time.

F. Smoothness--increasing smoothness increases the time the AI uses to return to the racing line, increases the yaw K1 constant whose effect is unknown, and increases the adjusted traction circle value thus making it corner better.

G. Qualifying--increasing qualifying increases the adjusted traction circle value, the longitudinal (fore and aft) acceleration capability, and decreases the gearshift time thus making the car corner and accelerate better.

H. Magic_grip--In the original Papyrus driver.ini file, some drivers have a magic grip setting while others do not. The purpose of this parameter is unknown. In this tutorial, magic_grip changes are ignored and all drivers have their magic_grip set to 1.00 for testing.

As mentioned before, some parameters only affect driver behavior (such as passing ability) while others only affect the driver's raw speed and time. The following table clarifies these relationships:

	Behavior	Speed/Time
Aggression	X	
Alertness	X	
Experience	X	
Hype		X
Quickness		X
Smoothness	X	
Qualifying		X

2. The gpl_ai.ini file contains settings that affect ALL drivers. In addition to the scaling factors mentioned in the previous discussion of driver parameters, the gpl_ai.ini file contains numerous settings that affect AI driver performance. As Kuratko pointed out on his website, several changes to the settings within the gpl_ai.ini file have been discussed on the GPL forum. For the most part, these changes only affect the AI's passing/following behavior; not their raw speed. My primary emphasis in this tutorial is on the parameter settings in the driver.ini file. I have not tested changes to the gpl_ai.ini file except for npt_override. Therefore, do so at your own risk!

To date, these changes are:

A. base_race_start_hiatus = 25.00 (default of 18.0)--sets the AI race start reaction time. Typically, the AI leave the starting line much too fast with the default setting. Setting base_race_hiatus to a higher setting slows the AI so that the player (that's you) can keep up. As with all timing parameters within GPL, this parameter is set in units of ticks. One tick in GPL is worth 1/36th of a second. As an aside, 36 frames per second is the maximum that GPL can display regardless of your computer or graphics card speeds. Sorry!

B. attempt_outbrake_dlong_sep = 25.00 (default of 12.4104). The AI must be within this distance in meters or they will not attempt to pass by outbraking the preceding car.

C. auto_blocker_line_speed_pct = .93 (default of .74). If I'm not mistaken, the AI looks at the speed of the preceding car compared to their own speed. If the ratio is less than the auto_blocker_line_speed setting, then the preceding car is considered a blocker and the AI are less aggressive about passing it. With the default setting, the AI tend to trail behind slower cars for long periods without passing.

D. being_squeezed_speed_coeff = .80 (default of .85). If being passed, the AI will give way to the passing car more easily with a lower setting. This makes the AI less aggressive in defending their position.

E. straightaway_pass_speed_coeff = 1.0001 (default of 0.000). I am unsure about what effect this parameter has on AI speed. I have seen posts on the GPL forum that indicate settings above 1.00 cause problems at some tracks. My limited testing confirms major problems if you set this parameter to other than 0.000. Therefore, I do NOT recommend you change the default setting.

F. braking_efficiency_coeff = 0.95 (default of .80). Some AI (particularly Surtees in the Honda) have a tendency to rear end you when braking. Raising this setting increases the braking ability of all AI cars thus making Surtees less likely to run into you.

Kuratko has done a lot of hard work in testing and reviewing the effects of various driver errors and mechanical failures on the AI's performance. I have not researched these areas yet and rely on his research.

3. The track.ini files (one per track) contain settings for each track. Within each track.ini file's [STATISTICS] section is the reference_value setting. Changing the track reference_value will not change driver qualifying or race times. There is a method to make the entire AI field faster or slower at all tracks which will be covered later. In addition, some tracks use adjustments to the track_dlong_sep_coeff, dlong_speed_adj_coeff, and dlong_speed_maximum settings to change AI driver's behavior and speed to account for track differences. These parameters only apply at that specific track.

A. track_dlong_sep_coeff--setting this parameter to values above 1.00 increases the separation that the AI attempts to maintain between itself and an adjacent car. Setting the parameter below 1.00 decreases the separation.

B. dlong_speed_adj_coeff--setting this parameter to values above 1.00 increase the AI speed. Setting the parameter below 1.00 decreases the AI speed.

C. dlong_speed_maximum--sets the maximum speed after all scaling factors are taken into account. This parameter is set in meters per tick. A setting of 1.00 results in a top speed of 80 mph; a setting of 2.00 results in a top speed of 160 mph; etc.

4. The gpl.exe file contains the code that runs the GPL program. Obviously, there are many unknown values contained within the code that affect the AI performance. Changes to the code must be done through a software patch such as the GPL65 team's 1965 modification for example or with a hexidecimal editor. Because we cannot easily change these values, code modification is not covered in this tutorial.

TESTS OF AI QUALIFICATION PERFORMANCE

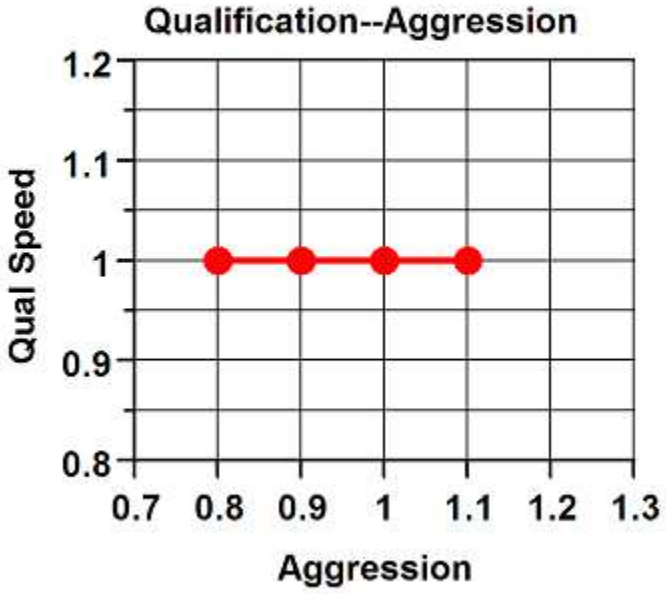
I conducted a series of tests that examined the effect of changes to driver parameters on the driver's qualifying performance. All tests were done using the baseline driver in the Lotus on the Monza track. By convention in all the tables and graphs in this tutorial, higher relative performance is defined as higher speed/lower lap time. The following table shows the results of these tests.

Relative Qualifying Performance

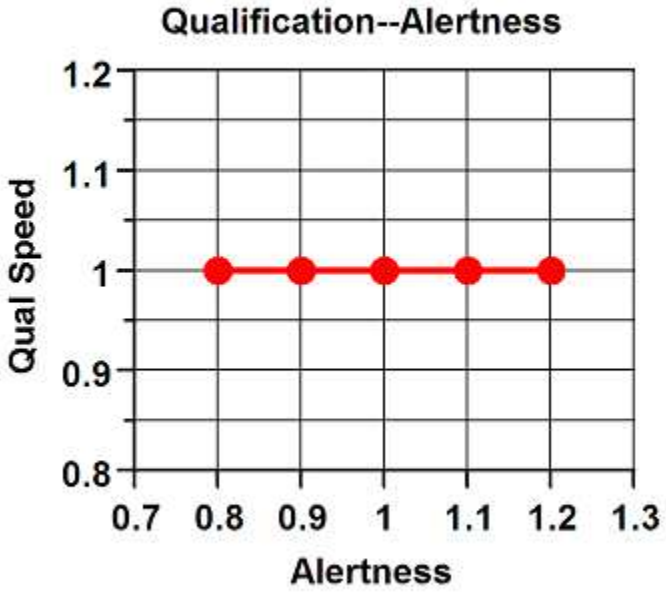
	Aggr	Alert	Exp	Hype	Quick	Smooth	Qual
0.80 |	1.000	1.000	N/A	0.807	0.949	N/A	0.950
0.90 |	1.000	1.000	0.998	0.907	0.989	0.994	0.991
1.00 |	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.10 |	1.000	1.000	1.000	1.068	1.007	1.000	1.007
1.20 |	N/A	1.000	1.000	1.122	1.008	1.001	1.009

N/A--no value obtained

1. Aggression: Has no effect on qualifying performance as seen in the following graph.



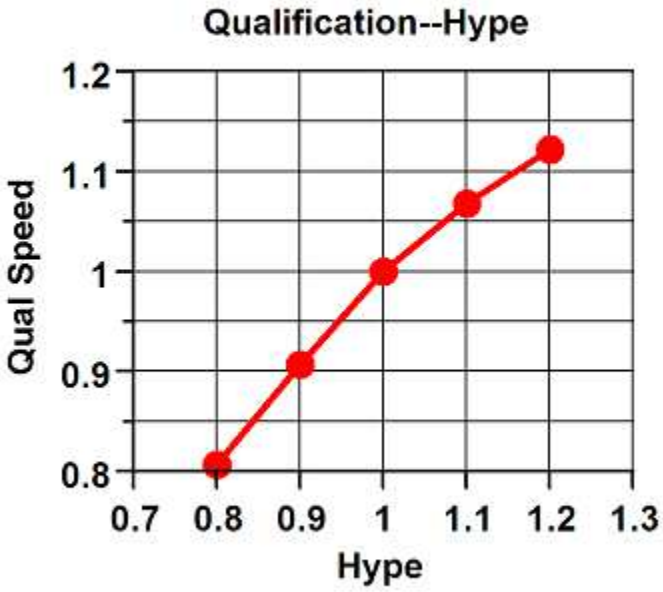
2. Alertness: Has no effect on qualifying performance as seen in the following graph.



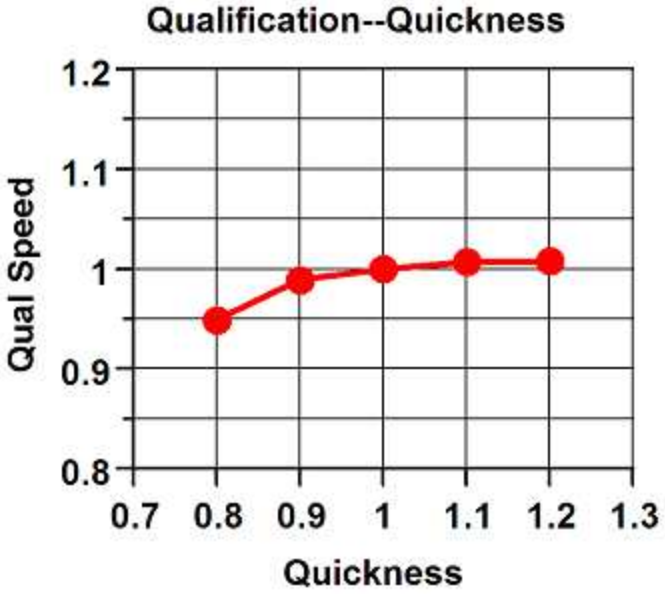
3. Experience: Has no effect on qualifying performance if set greater than .90 as seen in the following graph. All driver.ini files including the original Papyrus file use experience settings of .96 or greater. Therefore, this parameter

normally has no effect on qualifying performance.

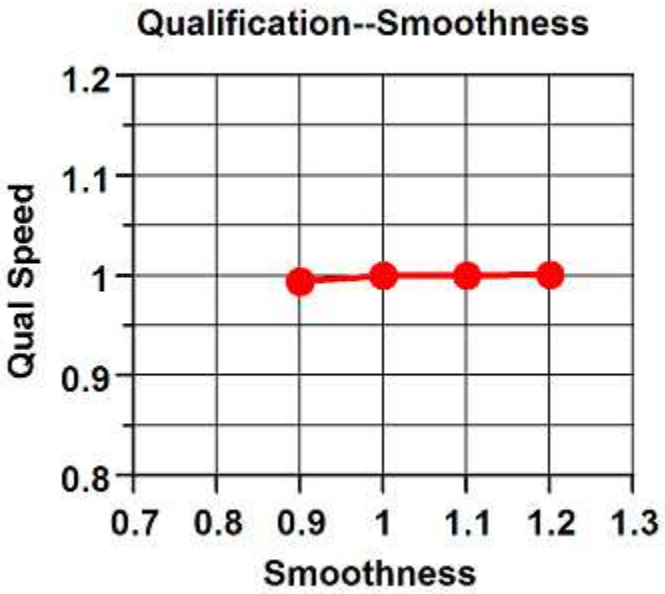
4. Hype: Has a huge effect on qualifying performance as seen in the following graph. This parameter is the major determinant of a driver's speed.



5. Quickness: Has a medium effect on qualifying performance as seen in the following graph.



6. Smoothness: Has a small effect on qualifying performance if set less than 1.00, but has no effect if set to 1.00 or greater. The original Papyrus driver.ini file uses a setting of 1.00 for all drivers. Therefore, we can assume that Papyrus didn't think that smoothness was a valuable driver characteristic. The following graph shows this effect.



7. Qualifying: Has a medium effect on qualifying performance that is identical to the effect of changing quickness. The following graph shows this effect.

In summary, the only driver parameters that affect qualifying performance are hype, quickness, and qualifying. The other parameters have no effect whatsoever.

TESTS OF AI RACE PERFORMANCE

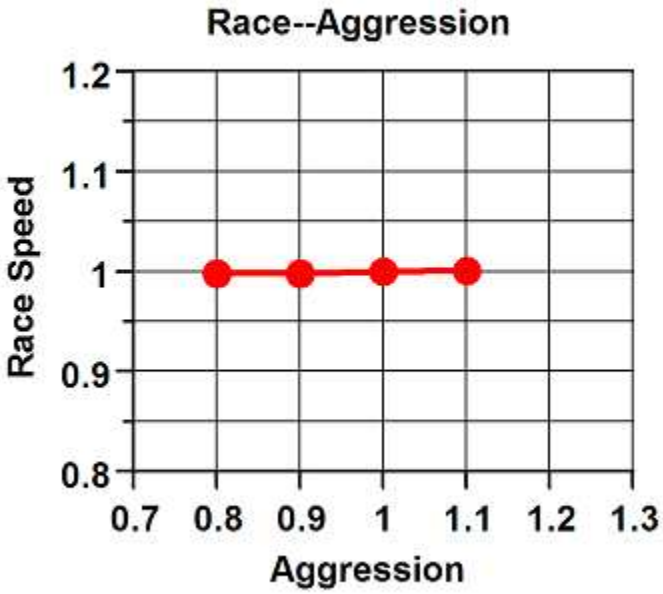
I also conducted a series of tests that examined the effect of changes to driver parameters on the driver's race performance. All tests were done using the baseline driver driving the Lotus on the Monza track. Note that these tests only show the parameter's effect on the driver's race performance without competition. It does not mean that changes to these parameters don't affect the driver's behavior and performance when racing against other drivers. The following table shows the results of these tests.

Relative Race Performance

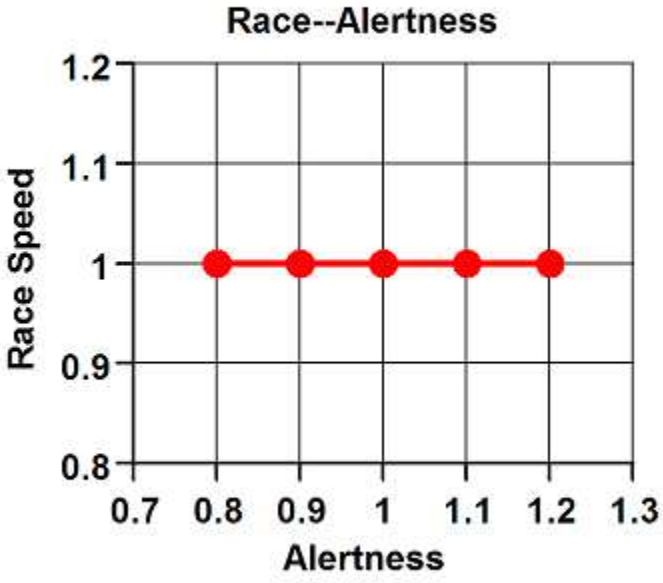
	Aggr	Alert	Exp	Hype	Quick	Smooth	Qual
0.80 |	0.998	1.000	N/A	0.809	0.948	N/A	1.000
0.90 |	0.998	1.000	0.998	0.909	0.989	0.994	1.000
1.00 |	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.10 |	1.001	1.000	1.000	1.067	1.008	1.001	1.000
1.20 |	N/A	1.000	1.000	1.121	1.010	1.000	1.000

N/A--no value obtained

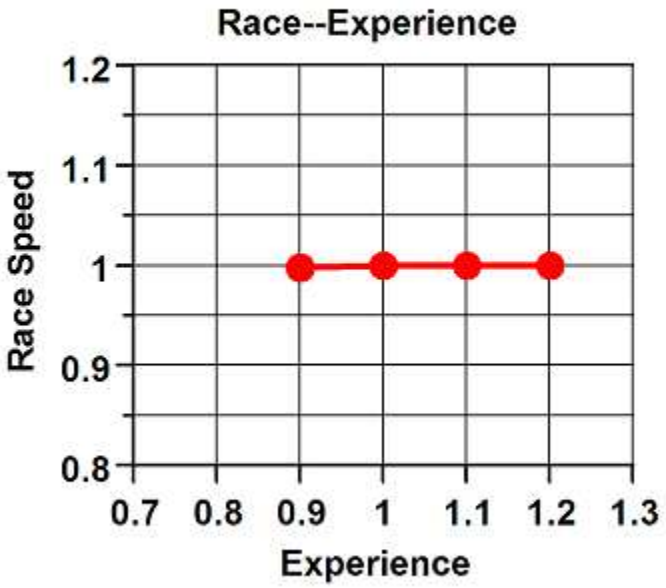
1. Aggression: Has no effect on race performance as seen in the following graph.



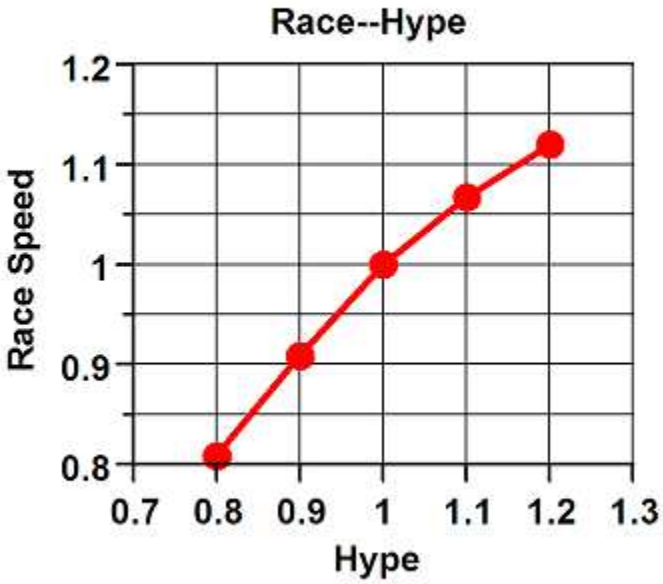
2. Alertness: Has no effect on race performance as seen in the following graph.



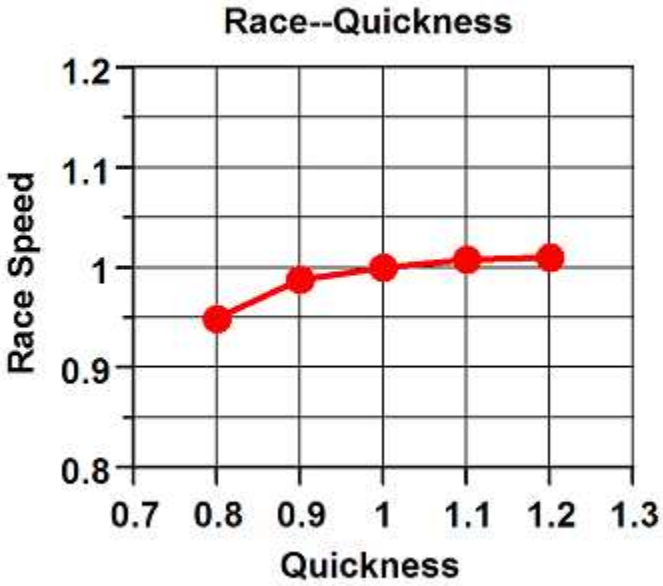
3. Experience: Has no effect on race performance if set greater than .90 as seen in the following graph. All driver.ini files including the original Papyrus file use experience settings of .96 or greater. Therefore, this parameter normally has no effect on race performance.



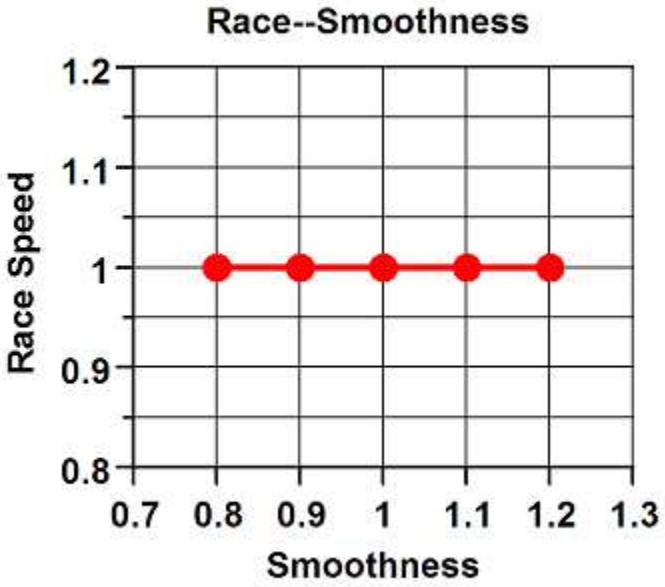
4. Hype: Has a huge effect on race performance as seen in the following graph. This parameter is the major determinant of a driver's speed.



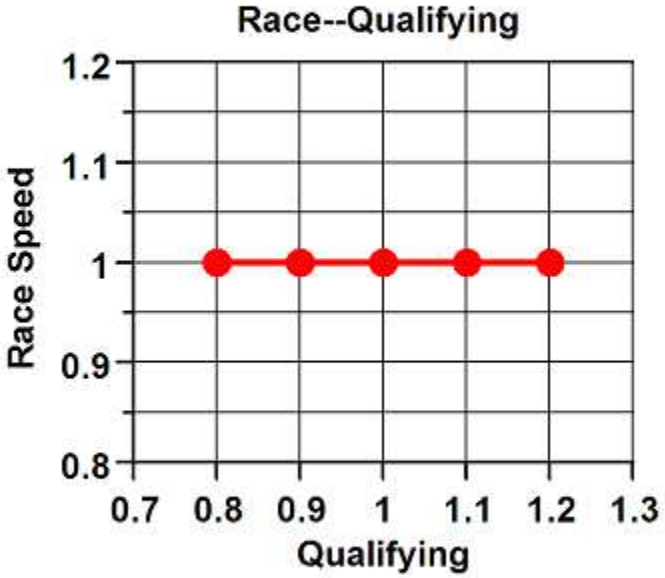
5. Quickness: Has a medium effect on race performance as seen in the following graph.



6. Smoothness: Has a small effect on race performance if set less than 1.00, but no effect if set to 1.00 or greater. The original Papyrus driver.ini file uses a setting of 1.00 for all drivers. Therefore, we can assume that Papyrus didn't think that smoothness was a valuable driver characteristic. The following graph shows this effect.



7. Qualifying: Has no effect on race performance. The following graph shows this effect.



In summary, the only driver parameters that affect race performance without competition are hype and quickness. The other parameters have no effect whatsoever. As mentioned before, just because a parameter doesn't affect a driver's race performance without competition doesn't mean the parameter doesn't affect a driver's behavior when racing against other drivers. We have seen in the AI parameters discussion that aggression, alertness, experience, and smoothness probably do affect to some extent how a driver behaves when following or passing another.

Comparing these race tests with the previous qualifying tests indicates that all parameters with the sole exception of qualifying affect qualifying and race performance exactly the same! The qualifying parameter only affects the qualifying performance, not the race performance.

TESTS OF HYPE VERSUS QUICKNESS ON RACE PERFORMANCE

Until now, we have only considered the effect of changing a single AI driver parameter on a driver's qualifying and race performance. What happens when we change two parameters at once? Obviously, it is possible to vary aggression, alertness, experience, hype, quickness, smoothness, and qualifying to many possible values, but testing the effect of all these changes is clearly impossible. Each change would require restarting the program, running a qualifying session and a race, and recording the result. This would literally take forever.

However, we already know that aggression, alertness, experience, and smoothness have no measurable effect on qualifying and race performance. Hype, quickness, and qualifying do affect qualifying performance while only hype and quickness affect race performance. Therefore, it's not too time consuming to do a test of simultaneously varying hype and quickness to see their effect on race performance. The following table shows the results.

Race Performance
Hype Versus Quickness

		Hype				
		.80	.90	1.00	1.10	1.20
	0.80 |	0.809	0.904	0.948	1.026	1.091
	0.90 |	0.809	0.908	0.989	1.055	1.112
Quickness	1.00 |	0.809	0.909	1.000	1.067	1.120
	1.10 |	0.810	0.910	1.008	1.079	1.130
	1.20 |	0.810	0.910	1.010	1.089	1.140

In summary, using both hype and quickness you can adjust race performance from .809 to 1.140 (Quickness=.80, Hype=.80 to Quickness=1.20, Hype=1.20). Thus you can slow the AI by 19.1% or speed them up by 14.0% for a range of 33.1%. This is a huge adjustment range that is far beyond what will be necessary. Using Hype alone, you can adjust the race performance from .809 to 1.120 (Quickness=1.00, Hype=.80 to 1.20). Thus you can slow the AI by 19.1% or speed them up by 12.0% for a range of 31.1%. This also is a large range of adjustment. Using Quickness alone, you can only adjust race performance from .948 to 1.010 (Quickness=.80 to 1.20, Hype=1.00). Thus you can slow the AI by 5.2% or speed them up by 1.0% for a range of 6.2%. This is a very small range of adjustment.

My conclusion is there is no point to adjusting both hype and quickness as hype alone provides an adequate range of adjustment to race performance. Why complicate matters by adjusting both as changing quickness also affects qualification performance? It is easy to set the driver's race performance by using hype alone as you can vary the race time from 90.9% to 105% of the baseline merely by setting hype between .90 and 1.05.

To make it easier to compute the needed hype that will result in a given race performance, I did a regression analysis that compared hype to race performance. Simplistically, regression analysis uses mathematical techniques to fit the best curve to the data. The formula for this curve is:

Hype = 1.056 - (1.355 * Race Performance) + (1.302 * Race Performance ^ 2)

where Race Performance is the driver's relative race performance compared to the baseline

This regression formula is extremely accurate. For you statisticians, R Squared is .999 which means the formula is 99.9% accurate in describing the relationship between hype and race performance. As an example, assume you have a driver that should perform at 96.0% of the baseline. Substituting .960 into the formula results in a hype setting of .970.

TESTS OF HYPE VERSUS QUALIFYING ON QUALIFYING PERFORMANCE

We learned earlier that only hype, quickness, and qualifying have a measurable effect on qualifying performance. We also learned that using hype alone provides sufficient adjustment to vary race performance. Therefore, we now need to know how varying both hype and qualifying affect qualifying performance.

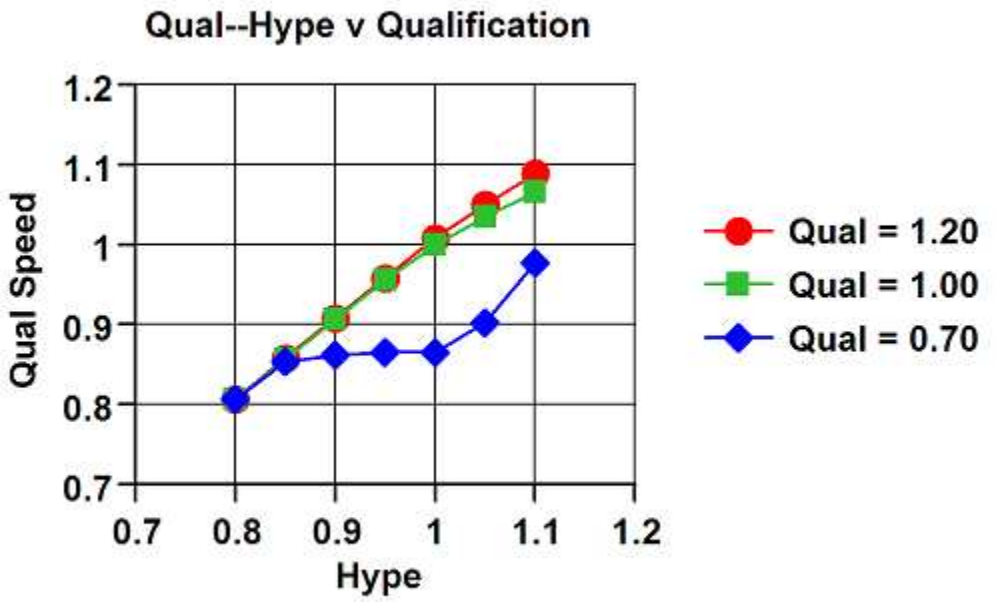
The following table shows how hype and qualifying affect qualifying performance.

Qualifying Performance
Hype Versus Qualifying

		Hype			
		.80	.90	1.00	1.10
	0.70	0.807	0.862	0.865	0.977
	0.80 |	0.807	0.908	0.950	1.030
	0.90 |	0.807	0.907	0.991	1.057
Qualifying	1.00 |	0.807	0.907	1.000	1.066
	1.10 |	0.807	0.908	1.007	1.081
	1.20 |	0.807	0.908	1.008	1.089

For example, with a hype of 1.10 you can adjust the qualification performance from 97.7 % to 108.9% of the baseline. With a hype of .80 or lower, you can't adjust the qualification time at all from the baseline. Thus at the lower hype settings, you have much less range in which to adjust qualification. For those drivers whose qualifying performance was worse than their race performance, you should have sufficient adjustment range; however, for those driver's who qualify much better than their race performance, you may not have enough adjustment.

The data is graphically depicted in this chart.



This is an interesting graph. It shows that we can vary qualification performance over a wide range by using hype and qualifying. However, there are drawbacks. The greatest range of adjustment is available when hype is close to 1.00. At lower or higher hype settings, we have a smaller range in which varying the qualifying parameter has an effect. Also, with any given hype, there isn't much capability to increase the driver's speed. We can make a driver qualify far slower than his race performance, but it is difficult to make him qualify much faster.

Because of the variability of the hype versus qualification results, multiple regression analysis does not work well at estimating qualifying performance. You can't derive a simple formula that accurately describes how a driver will qualify for given hype and qualifying settings. Instead, we have to use the hype versus qualifying table shown previously to get the qualifying parameter. For my own use, I derived an expanded hype versus qualifying lookup table (not shown) that uses interpolated values from the previous table. You enter the chart with the required qualifying performance and hype setting and read the necessary qualifying setting.

My conclusion is that the best approach to controlling the individual AI driver is to:

1. Use hype alone to set the race performance
2. Use qualifying based on hype to set the qualifying performance

TESTS OF CONTROLLING THE AI FIELD'S SPEED

To make the simulation more enjoyable, it would be nice to adjust the speed of all AI drivers so that the player (that's you!) can race competitively right from the start. As we all know, GPL is a very difficult simulation to learn to drive well. It takes time, patience, and skill. Some of us are faster drivers than others...particularly when we first start learning the simulation. Yet racing against the AI can be frustrating as they are invariably too fast for beginning players. I'll admit it. After several years, I'm not the fastest GPL driver around and still need to slow the AI field in order to compete effectively and win a race.

So how do we slow the entire AI field? Much has been written in the GPL forum about how to do this. The following discusses the easiest method to control the AI field's speed, but it is not without some drawbacks. My congratulations go to David Wright whom I believe was the first to discover this method. My findings and conclusions are a direct result of David's earlier tests.

First, let's talk about how GPL controls the AI field's speed. To their credit, Papyrus wanted the AI field to adjust in accordance with the player's speed. As the player's speed increases with practice, the AI field should also increase so that the AI field would always present a challenge to the player no matter how fast he became. Unfortunately, Papyrus made the AI too fast! To show how all this works, let's introduce the concept of Normalized Player Time (NPT).

As the player plays the simulation more, GPL remembers the last 10 lap times and takes the average as the player's NPT for that track. This value is saved in the player.sts file inside the players directory. (Don't try reviewing or editing this file with Wordpad or Notepad as it uses a proprietary format). Until the player has driven 10 laps, the default initial value for NPT is set at 1.20 by the startup_npt setting in the gpl_ai.ini file. Obviously as the player's lap times improve, the NPT will decrease. GPL then lowers the entire field's times based on the player's NPT.

This is a great idea; however, Papyrus didn't execute it very well because some drivers are less sensitive to the NPT than others. A case in point is Jimmy Clark who is always too fast (at least for me) regardless of how fast the player becomes. To paraphrase Rhett Butler in "Gone With The Wind", Clark frankly doesn't give a damn about how fast you are. He's going to proceed along his merry way; ripping off incredibly fast lap times without regard to your struggles at the wheel. Seemingly, there's nothing you can do about it.

Or is there? In addition to the player's NPT, each AI driver has a parameter called global_hype_scaling which is set in the driver.ini file. This parameter tells the GPL program how sensitive the driver is to NPT. As expected, Clark has an incredibly low global_hype_scaling setting of .05 which effectively makes him invulnerable to the player's NPT. Hence, he is always too fast for the novice player. Drivers with high global_hype_scaling values are greatly affected by the player's NPT.

Before we look at past efforts to control the AI field, let's look in more detail at how npt_override and global_hype_scaling interact to affect AI speed. This is a bit confusing so hang on.

In a nutshell:

1. If npt_override is set to 0.00 (off), then the AI are affected by changes to global_hype_scaling and the player's NPT. This is the default setting.
2. If npt_override is set > 0.00 to < 1.00, then the AI are NOT affected by changes either to global_hype_scaling or the player's NPT. The AI field is speeded up.
3. If npt_override is set to 1.00 (on), then the AI are NOT affected by changes to either global_hype_scaling or the player's NPT. The result is the baseline speed.
4. If npt_override is set > 1.00, then the AI are affected by changes to global_hype_scaling, but NOT the player's NPT. The AI field is slowed down.

The following table summarizes these effects:

Npt_override Summary

Npt_override	Global_Hype_Scaling	NPT	AI Field
0.00 (off)	Active	Active	
0.01 to .99	Inactive	Inactive	Speeded Up
1.00 (on)	Inactive	Inactive	Baseline
>1.00	Active	Inactive	Slowed Down

Armed with our new knowledge of how npt_override and global_hype_scaling affect the AI, let's examine some

methods to control the entire AI field's speed. The following table contains settings for npt_override and global_hype_scaling for Clark, as an example, that others have used in the past:

Previous Npt_override and Global_hype_scaling Settings

	Papyrus	Alison Hine	David Wright	Kuratko
Npt_override	0.00	0.00	1.00	1.00
Global_hype_scaling	0.05	0.45	1.00	1.00

As mentioned before, Papyrus used an npt_override of 0.00 which made the player's NPT active, but used a very low global_hype_scaling of only .05 which made Clark very insensitive to NPT. Good news, bad news.

Alison Hine, who was one of the beta testers for the original program, discovered the problem with the default AI speed and made a correction by setting Clark's global_hype_scaling to .45...a much more reasonable value. This made Clark more sensitive to NPT which slowed him down considerably. However, each driver's global_hype_scaling had to be individually adjusted so it was impossible to adjust the entire AI field's speed with just one setting change. This method was a step in the right direction, but needed more refinement.

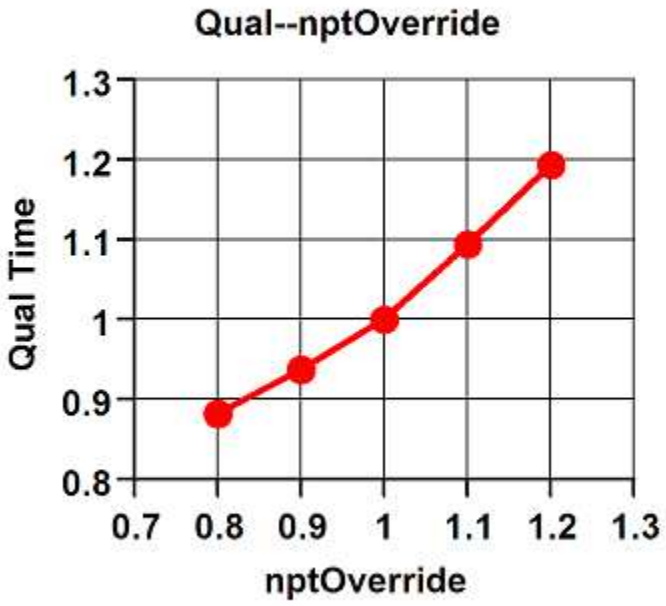
Along came David Wright who obviously investigated the relationship between npt_override and global_hype_scaling and came up with an inspired method of controlling the AI field's speed. He set all driver's global_hype_scaling to 1.00 which made them equally sensitive to changes in NPT, then set npt_override to control the entire AI field's speed! This was a great discovery and is the best method seen to date. However, even David's method has a drawback as I will soon point out. More recently, Kuratko also used David's method of adjusting npt_override to control the AI field.

So how much effect do we get by adjusting npt_override? The following table shows the relative qualification times for our baseline driver as npt_override is changed from .80 to 1.20.

Npt_override Effect On Qual Time

		Relative Qual Time
	0.80	0.883
	0.90	0.937
Npt_override	1.00	1.000
	1.10	1.094
	1.20	1.193

Here is a graph which shows the same data:



In summary, increasing npt_override above 1.00 increases qualification time by about the same percentage. For example, if you set npt_override to 1.20, the AI field will be slowed by 19.3%. As you decrease npt_override below 1.00, qualification time also decreases but at a slower rate. For example, if you set npt_override to .80, the AI field is only speeded up by 11.7%.

I believe, as do David and Kuratko, that the best method to control the AI field is to set all driver's global_hype_scaling to 1.00 and adjust npt_override. This method works extremely well, but has one glaring drawback…when we change npt_override above or below 1.00, not all drivers are equally affected during qualification. This is a complex subject, but here goes.

We have seen so far that you can control the entire AI field's speed by changing npt_override. We would hope when npt_override is changed that all AI drivers would speed up or slow down at the same rate. This is true for most drivers, but not for all.

1. Drivers who have a high hype setting (1.00 or more) with a low qualifying setting (.90 or less) do not slow down as much during qualifying as other drivers when npt_override is raised above 1.00. These drivers qualify higher than they should.
2. Drivers who have a low hype setting (less than 1.00) with a high qualifying setting (1.00 or more) speed up more during qualifying than other drivers when npt_override is lowered below 1.00. These drivers qualify higher than they should also.
3. Apparently, changing npt_override affects all drivers equally during a race. Therefore, changing npt_override doesn't materially affect the relative finishing positions of the drivers.

For example, the following table shows the effect of changing npt_override from .80 to 1.20 on the relative qualifying positions of all drivers. You can see that Jackie Stewart, in particular, does much better than he should as npt_override is raised. This is because he has a high hype setting of 1.004 and a low qualifying setting of .860. On the other hand, Chris Amon and Pedro Rodriquez do much better as npt_override is lowered. This is because they have low hype settings and high qualifying settings of 1.25. (I'll explain later how we determined these values).

Relative Qualifying Positions

npt_override=1.00		npt_override=.80		npt_override=1.20	
Clark		Amon	<===	Stewart	<===
Gurney		Clark		Gurney	
Hill		Brabham		Hulme	
Brabham		Rodriquez	<===	Surtees	
Surtees		Gurney		Brabham	
Hulme		Hill		Rindt	
Amon	<===	McLaren		Clark	
Stewart	<===	Hulme		Hill	
Parkes		Surtees		Parkes	
Rodriquez	<===	Stewart		Spence	
Rindt		Siffert		Amon	
McLaren		Parkes		Rodriquez	
Siffert		Rindt		McLaren	
Irwin		Irwin		Siffert	
Spence		Spence		Irwin	
Bonnier		Anderson		Bonnier	
Anderson		Bonnier		Anderson	
Ligier		Ligier		Ligier	

I know of no way around this problem If we are going to use npt_override to control the AI field's speed, we will just have to live with the fact that some drivers will do better than desired during qualification. Fortunately, their race times don't suffer from this same effect so the AI finish in their proper position.

PART II

QUANTIFYING HISTORICAL DRIVER PERFORMANCE

Of course, we have devoted all this time and effort in determining how the AI react to changes in driver parameters in hope that we can set these parameters to make the AI drivers perform like their real world counterparts. Wouldn't it be great if Clark's AI drove exactly as he did in 1967? In other words, these are our goals:

1. The AI driver's qualifying performance should match the real world driver's performance. His qualifying time should be the same and his relative qualifying time in relation to other drivers should be the same.

2. The AI driver's race performance should match the real world driver's performance. His finishing time should be the same and his relative finishing position in relation to other drivers should be the same.

It's doubtful that we would ever achieve these goals by simply modifying the driver.ini and gpl_ai.ini files alone. More probably, we would have to patch the gpl.exe program code to do the job perfectly. However, that step is well beyond the scope of this tutorial so we will confine ourselves to modifying the two .ini files (which is my way of saying that I haven't a clue how to patch the code!).

Before we can set the AI driver parameters, we need to know how the actual drivers did during the races in 1967. Somehow, we need to quantify their performance. Fortunately, there are several online sources for qualification times, qualifying positions, race finishing times, and race finishing positions.

My favorite Formula 1 information websites are:

1. Grand Prix Stats at http://www.f1-stats.de
2. F1 Gamers at http://www.f1gamers.com
3. F1 Database at http://www.f1db.com

Papyrus and other researchers adjusted AI parameters to control the AI drivers. The following table compares some different approaches using Jimmy Clark's parameters as an example:

	Papyrus	Alison Hine	David Wright	Kuratko
Aggression	1.030000	1.030000	1.030000	1.000000
Alertness	1.030000	1.030000	1.030000	1.020000
Experience	1.010000	1.010000	1.010000	1.070000
Hype	1.002000	1.002000	1.002000	1.016000
Qualifying	1.020000	1.020000	1.100000	0.990000
Quickness	1.050000	1.050000	0.960000	1.020000
Smoothness	1.000000	1.000000	1.000000	1.010000
global_hype_scaling	0.05	0.45	1.000000	1.000000

Alison and Nate Hine used Papy's original settings for aggression, alertness, experience, hype, and smoothness, but varied speed by adjusting the driver's global_hype_scaling parameter. David Wright used Papy's original settings for aggression, alertness, experience, and smoothness, but varied the driver's speed by adjusting hype, quickness, and qualifying. Kuratko went a step farther by adjusting all parameters.

This table begs some interesting questions. Just how do you quantify a driver's behavior? How do you set driver parameters that truly reflect his speed and driving style? We know that Clark was faster than Ligier, but do we really know that Clark was the more aggressive driver? As Kuratko so eloquently pointed out on his website, we really can't! All we can hope to do is approximate a driver's performance within the constraints of the gpl.exe program and our limited knowledge of how it works. I'm sure the GPL designers struggled with these questions too.

So let's get to it.

First, what kind of data can we can obtain to use in quantifying the driver parameters? I think we can agree that the more data we collect, the better chance we have of deriving good parameters. Typically, we can get the following data for each race:

1. Each driver's qualifying time
2. Each driver's qualifying position
3. Winner's race finishing time
4. Winner's number of laps
5. Race finishing time for drivers who finished on the same lap as the winner
6. Number of laps behind for those drivers who finished the race, but not on the same lap as the winner
7. Race finishing position for those drivers who are "classified"

Second, let's look at one method of quantifying a driver's qualification performance using this data. By convention, let's assume that the fastest qualifying driver's time is arbitrarily assigned a value of 1.00. Other drivers will be measured relative to this standard. It's easy to compute each driver's qualification time relative to the fastest qualifier. The formula for this is:

$$\text{Qual Time} = \frac{\text{Driver's Time}}{\text{Fastest Driver's Time}}$$

Knowing the driver's relative qualification time for each race, we can simply average these times for all races to get a yearly value.

Fortunately, the hard work has already been done for us as the Grand Prix Stats website has listings of "Average Gap To Pole Position" by driver for each year as a percentage of the pole winner's time. For example, in 1967 Jimmy Clark was the best qualifying driver as he had an average gap to pole position of only .638%. Therefore, Jimmy qualified on the average at (1 - .00638) or .99362 of the pole winner's time.

1967 Average Gap to Pole Position
In Percent

Driver	Gap
Hulme	2.004
Brabham	1.378
Clark	0.638
Surtees	2.562
Amon	2.064
Rodriquez	3.797
Hill	1.740
Gurney	1.453
Stewart	3.026
Spence	4.217
Rindt	2.980
Siffert	5.012
McLaren	4.495
Bonnier	6.326
Anderson	6.463
Parkes	3.461
Irwin	5.528
Ligier	11.415

We can also easily compute a driver's average qualifying position for a season. We can do this manually for each race, but again we are lucky as the Grand Prix Stats website has listings of "Average Grid Position" by driver for each year.

1967 Average Grid Position

Driver	Position
Hulme	5.455
Brabham	3.909
Clark	2.546
Surtees	7.222
Amon	6.300
Rodriquez	10.375
Hill	5.273
Gurney	4.545
Stewart	8.636
Spence	11.636
Rindt	8.200
Siffert	13.273
McLaren	7.778
Bonnier	15.000
Anderson	15.000
Parkes	9.000
Irwin	13.444
Ligier	17.857

After a bit of experimentation, I discovered a formula using average grid position that closely matched average gap to pole position. The formula is:

$$\text{Qual Position} = 1.005 - \frac{\text{Average Grid Position}}{200}$$

If a driver won the pole each race, he would receive a relative qualifying position value of 1.00. In 1967, Jimmy Clark qualified in an average position of 2.546; about halfway between second and third. Substituting 2.546 into the formula results in a value of .992 which is essentially the same as Clark's relative qualification time of .994.

We then average the relative qualification time and relative qualifying position values to get overall qualifying performance. Now someone could argue that using qualifying time alone is sufficient to rank a driver's performance, but I believe that the more data we have and use, the better the result. In Clark's case, his overall performance was .993 after rounding. In other words, Jimmy qualified at 99.3% of the mythical driver who finished on pole for every race. The formula for overall qualifying performance is:

$$\text{Qual Performance} = \frac{\text{Qual Time} + \text{Qual Position}}{2}$$

Third, let's examine a similar method to quantify a driver's race performance. Unfortunately, we don't have times for all drivers who finished a race. Only times for those drivers who finished on the same lap as the winner are available. Other drivers who were "classified", but didn't finish on the winner's lap are listed as being so many laps down. Can we still use this data? I think we can.

For the winning driver and those drivers who finished on the same lap, we have their exact finishing times. We also know the winning driver's overall time and the number of laps he completed; therefore, we can compute an average lap time for the winner. If another driver finishes one lap behind the winner, he is at least that much time behind the winner as well. So we could simply add one laps worth of time to the winner's time to get a general idea of the time

the next driver would have taken if the he had driven the full distance. In fact, we are being generous as he probably didn't finish exactly one lap behind either, but somewhere between one and two laps down. So a more appropriate amount to add would be 1.5 laps worth of time. An even more sophisticated approach would be to look at how many drivers finished one lap down. Surely, some of these actually finished closer to two laps down. So if there were two drivers who finished one lap down, the first would be assigned a time penalty of 1.33 laps while the other would receive a penalty of 1.67 laps. If three drivers finished one lap down, the first would receive a penalty of 1.25 laps, the second would be 1.50 laps down, and the third would be 1.75 down, etc.

Relative race time for those drivers who didn't finish on the winner's lap is given by the following formula:

$$\text{Race Time} = \frac{\text{Winner's Time}}{\text{Winner's Time} + \frac{(\text{Winner's Time} * \text{Laps Down})}{\text{Winner's Laps}}}$$

where Laps Down is determined as explained above.

We then average the race times for all races that the driver finished in the year to get overall relative race time performance. As an example, Jimmy Clark finished six races in 1967 in which he had a relative race time of .991. In other words, Jimmy's average race time was 1 - .9908 or only .92% slower than the winning driver's time in the races he finished.

1967 Average Race Time
In Percent Of Winner

Driver	Time
Hulme	99.12
Brabham	98.71
Clark	99.08
Surtees	98.16
Amon	97.67
Rodriquez	94.34
Hill	98.57
Gurney	99.20
Stewart	98.60
Spence	96.35
Rindt	98.45
Siffert	94.98
McLaren	95.90
Bonnier	93.75
Anderson	94.60
Parkes	98.50
Irwin	93.65
Ligier	91.95

It's not hard to compute these times for all "classified" drivers at each race. Of course these are estimates, but even so, they are very good indicators of how well a driver performed in a race. If we don't do these estimates, we would only have hard times for those few drivers who finished on the winner's lap. Obviously, there were many drivers who never finished on the winner's lap; therefore we would have no performance value for them at all. One problem with this approach is that some drivers just didn't finish many races. Dan Gurney, for example, only finished two races all year. So we have a limited sample size to draw upon in his case.

One way to work around this lack of data for some drivers is to look at the known data in a different way. Not only do we have race times, but we also have race finishing positions for those races that the driver was “classified“. Let's use that data too in a similar manner that we did with qualifying positions. The rationale for using finishing position is that race car drivers don't always go flat out throughout a race. They may drive just fast enough to gain or maintain their position.

Fortunately, the F1 Gamers website lists the "Average Finish Position" for each year by driver which saves us some extra computing time.

1967 Average Finish Position

Driver	Position
Hulme	2.33
Brabham	2.78
Clark	2.17
Surtees	3.60
Amon	4.75
Rodriquez	5.71
Hill	2.67
Gurney	2.00
Stewart	2.55
Spence	5.67
Rindt	4.00
Siffert	7.40
McLaren	5.50
Bonnier	7.25
Anderson	7.33
Parkes	5.00
Irwin	6.50
Ligier	9.25

The following formula gives the relative race finishing position value:

$$\text{Race Position} = 1.01 - \frac{\text{Average Finish Position}}{100}$$

If a driver won every race, he would receive a race position value of 1.00. For example, Jimmy Clark in 1967 had an average finishing position of 2.17; therefore, his race position value was .988. This compares favorably with his race time value of .991.

I then average the relative race time and position values to get overall performance. In Clark's case, his overall race performance was .990 after rounding. In other words, Jimmy raced at 99.0% of the mythical driver who won every race. The formula for overall race performance is:

$$\text{Race Performance} = \frac{\text{Race Time} + \text{Race Position}}{2}$$

The following table shows the actual qualifying and race performances for each 1967 driver using the above formulas. The drivers are listed in the order in which they finished the championship.

1967 Actual Driver Performance

Driver	Qual Perf	Race Perf
Hulme	.979	.989
Brabham	.986	.985
Clark	.993	.990
Surtees	.972	.978
Amon	.976	.970
Rodriquez	.958	.948
Hill	.981	.984
Gurney	.984	.991
Stewart	.966	.985
Spence	.952	.958
Rindt	.967	.977
Siffert	.944	.943
McLaren	.961	.957
Bonnier	.933	.938
Anderson	.933	.941
Parkes	.963	.973
Irwin	.941	.941
Ligier	.901	.919

As an aside, Clark obviously did very well in 1967 even though he didn't win the championship. He actually outperformed Hulme, the champion, by a slight amount. The Brabham's incredible reliability was the key ingredient that won Hulme the championship that year. Also note that Dan Gurney turned in the best race performance of any driver although, as mentioned before, he only finished two races. The Eagle's reliability was terrible!

On the qualifying side, Clark was clearly the best qualifier...no one could match Jimmy in his Lotus 49.

So that's it. We've looked at a method and formulas that can be used to quantify a driver's historical performance, In this section, we haven't discussed yet how we can use this new information in GPL. That's the subject of the next part.

PART III

PUTTING IT ALL TOGETHER

In previous parts, we looked at how GPL models the AI drivers and how we can control their driving behavior and speed to some extent. We also developed some methods to quantify actual driver performance based on their historical records. In this section, we are going to put it all together and show how we can use the historical performance to derive GPL parameters that will make the AI drivers perform much like their real world counterparts.

The first thing we must consider is how GPL models each car. Remember from previous discussions that every car has a different relative performance on each track. We computed an average relative performance based on the ten original tracks plus the addon LeMans Bugatti track. Before we can assign values to our driver parameters, we've got to adjust the drivers' qualifying and race performance values for their car. We do this by "normalizing" the values to the Lotus which is arbitrarily assigned a performance value of 1.00. The reason we normalize to the Lotus is that it is the fastest car on average and all our parameter testing was done with the baseline driver in this car.

This is an important adjustment because if we didn't do so, drivers of other cars would be penalized. As an example, let's look at Chris Amon who drove for Ferrari in 1967. The Ferrari's performance was only 99.74% of the Lotus' performance. Amon's historical qualifying and race values are .976 and .970 respectively. We want him to perform at exactly the same levels within the simulation when he drives a Ferrari. To normalize Amon's performance, we must divide his qualifying and race values by .9974. This adjusts his performance upwards to indicate how well he would have done if he had driven a Lotus in 1967 instead of the Ferrari. Within the simulation however, Amon's performance will automatically lower to reflect his drive in the Ferrari. If we didn't make this adjustment, Amon's qualifying and race performances within GPL would be lower than the desired .976 and .970 respectively.

The following table shows each driver's 1967 real world performance when adjusted for his car:

1967 Driver Performance Adjusted For Car

Driver	Adj Qual Perf	Adj Race Perf	
Hulme	0.986	0.996	
Brabham	0.993	0.992	
Clark	0.993	0.990	
Surtees	0.988	0.994	
Amon	0.979	0.972	
Rodriquez	0.971	0.962	
Hill	0.981	0.984	
Gurney	0.985	0.992	
Stewart	0.981	1.001	<=== Wow!
Spence	0.967	0.974	
Rindt	0.981	0.991	
Siffert	0.958	0.956	
McLaren	0.962	0.958	
Bonnier	0.947	0.951	
Anderson	0.940	0.948	
Parkes	0.965	0.975	
Irwin	0.956	0.956	
Ligier	0.908	0.925	

This table is interesting because it shows how a driver's performance compared to his contemporaries assuming they

all drove the Lotus as modeled by GPL. Clark was the best qualifier as expected, but Brabham did just as well when his slower Brabham car is factored in. And just look at Jackie Stewart! He clearly was the best race performer when his dog-of-a-car BRM is considered. These "what if" situations are amusing and can generate a lot of discussion about who was really the best driver. I'll leave it to you to decide for yourself. One thing is certain however; Ligier was one lousy F1 driver in 1967.

For the parameters of aggression, alertness, experience, and smoothness, we will use the Papyrus default settings. For quickness, we'll use 1.00 for all drivers. Whether these values are correct or not, I don't know nor do I know how Papyrus determined them. Until we learn exactly how the GPL.exe program models the drivers, we won't know how to properly set these parameters. So for now, let's use the defaults.

Second, we compute a hype setting using the regression formula and the driver's adjusted race performance value. As a reminder, the formula is:

Hype = 1.056 - (1.355 * Race Performance) + (1.302 * Race Performance ^ 2)

Finally, we use the Hype Versus Qualification table from Part I to get a qualifying setting based on the driver's hype setting and adjusted qualification performance. I use an expanded version of this table (shown below) to make it easier to look up the qualification setting, but it is based on the Hype Versus Qualifying table previously shown. To use it, enter the table with the hype setting, read down the column until the required qualifying performance is found, then read the qualifying setting in the left column. A bit of interpolation is usually necessary. For example, if the driver's hype is computed as .95 and he has a required qualifying performance of .955, find his qualifying setting by entering the .95 hype column and reading down until you find the performance value of .955. Read left to the first column which returns a qualifying setting of .90. Usually, a little interpolation is necessary.

Qualifying Performance
Expanded Hype Versus Qualifying

		Hype						
		.80	.85	.90	**.95**	1.00	1.05	1.10
	0.70 |	0.807	0.854	0.862	0.864	0.865	0.902	0.977
	0.75 |	0.807	0.860	0.897	0.893	0.909	0.966	1.003
	0.80 |	0.807	0.858	0.908	0.934	0.950	0.995	1.030
	0.85 |	0.807	0.857	0.908	0.950	0.979	1.014	1.047
	0.90 |	0.807	0.857	0.907	**0.955**	0.991	1.026	1.057
	0.95 |	0.807	0.857	0.907	0.956	0.998	1.026	1.062
Qualifying	1.00 |	0.807	0.857	0.907	0.956	1.000	1.035	1.066
	1.05 |	0.807	0.857	0.907	0.957	1.005	1.042	1.075
	1.10 |	0.807	0.857	0.908	0.958	1.007	1.047	1.081
	1.15 |	0.807	0.858	0.908	0.958	1.008	1.049	1.085
	1.20 |	0.807	0.858	0.908	0.958	1.008	1.050	1.089

The following table shows the new AI parameter settings. It's labeled as Settings #1 because as we will see later, there is a second method for controlling the AI speed which will use a different set of values:

Driver.ini File Settings #1

Driver	Hype	Qual
Hulme	0.999	0.881
Brabham	0.993	0.997
Clark	0.990	1.027
Surtees	0.995	0.901
Amon	0.969	1.250
Rodriquez	0.957	1.250
Hill	0.984	0.925
Gurney	0.993	0.895
Stewart	1.004	0.844
Spence	0.971	0.893
Rindt	0.992	0.878
Siffert	0.951	1.012
McLaren	0.953	1.086
Bonnier	0.945	0.899
Anderson	0.942	0.848
Parkes	0.973	0.869
Irwin	0.950	0.991
Ligier	0.917	0.786

Although Amon and Rodriquez have unusually high qualifying settings, remember that these settings are relative to their hype settings.

SO WHAT ABOUT BANDINI?

Lorenzo Bandini presents a special problem for AI developers. Entering1967, he was the lead Ferrari driver, but only competed in one race. Ferrari didn't participate at Kyalami…the first race of the season. At Monaco, Bandini qualified very well in second position, but was tragically killed during the race. So how do we compute hype and qualifying parameters for him?

I obtained Bandini's qualification and race performances for the years 1964, 1965, and 1966. Averaging these performances, I got the following values:

Bandini's Historical Performance
1964 To 1966

	Qual	Race
Average	.978	.963
Adjusted for Car	.981	.965

Using these values, I computed hype and qualifying parameters for Bandini of .961 and 1.250 respectively.

WHAT ABOUT THE OTHER ORIGINAL AI DRIVERS?

The original Papyrus driver.ini file contains a few different drivers than those we have been using. Instead of Anderson, Ligier, Spence, and Stewart, Papyrus used Beltoise, Ginther, Ickx, and Scarfiotti. In this tutorial, I have

chosen to follow David Wright's driver recommendations as they are more representative of the actual historical record. However for those of you who prefer to use one or more of the original drivers, here are their settings. Their performances are adjusted for the car they drive in GPL; not necessarily the car they actually drove. For example, Beltoise drives a BRM in GPL, but he actually did the majority of his rides in a Matra which is not modeled by GPL. As with Bandini, I had to average their performances over three years to get sufficient data. For example, Ginther only tried to qualify at Monaco in 1967 and didn't even start the race. He didn't enter any other race in 1967 so I used his 1964 to 1966 performances just like Bandini.

Original Driver Performances and Settings #1

	Qual Perf	Race Perf	Hype	Qual
Beltoise	.962	.967	.963	.819
Ginther	.973	.956	.950	1.250
Ickx	.981	.985	.984	.930
Scarfiotti	.966	.980	.979	.849

Whew! This has been a long journey hasn't it? Well, we're almost done. The next part tests our new AI settings to see how well they work. We'll also find there is more than one way to skin a cat.

PART IV

TESTING THE AI VERSUS HISTORICAL PERFORMANCE

The proof is in the puddin' so how well do the new AI settings work? Do they meet our goals that the AI driver's qualifying and race performances and relative qualifying and finishing positions match his real world record?

The following table shows relative qualifying performance based on actual 1967 historical data versus the relative qualifying performance in GPL at Monza using our new hype and qualifying settings. The mythical driver who qualified first at every race would receive a value of 1.00. Remember that we are using an npt_override setting of 1.00.

Relative Qualifying Performance With Settings #1
Npt_Override = 1.00

Driver	1967 Actual	AI Driver	GPL Monza
Clark	.993	Clark	.993
Brabham	.986	Brabham	.984
Gurney	.984	Gurney	.984
Hill	.981	Hill	.984
Hulme	.979	Amon	.975
Amon	.976	Hulme	.974
Surtees	.972	Surtees	.974
Rindt	.967	Parkes	.963
Stewart	.966	Rodriquez	.962
Parkes	.963	McLaren	.958
McLaren	.961	Rindt	.955
Rodriquez	.958	Stewart	.954
Spence	.952	Irwin	.953
Siffert	.944	Spence	.950
Irwin	.941	Siffert	.949
Bonnier	.933	Bonnier	.939
Anderson	.933	Anderson	.935
Ligier	.901	Ligier	.911
		Average Difference:	0.46%

We see that the AI drivers meet the desired performance for both time and relative qualifying position. The average difference between actual and GPL performance is less than 0.5%. Therefore, our new AI parameter settings work well for qualifying.

As a reminder, the new AI parameters are derived from the baseline driver who has all parameters set to 1.00. However, even the baseline driver may not achieve the same qualifying time as the historical, real world pole winner. For example, the baseline driver qualifies at Monza at 89.78 seconds. In 1967, Jimmy Clark set the pole at 88.50 seconds. The baseline driver is over one second slower. Our GPL Clark using the new AI parameter settings is even slower at 90.40 seconds. This doesn't mean that the new AI parameter settings are bad; on the contrary they are very good at achieving the desired performance levels. It just means that we have based our AI parameter settings relative to the baseline driver and he isn't quite as quick as the fastest real world driver at Monza. It could be that GPL's Monza track length is too long or the Lotus' performance as modeled by GPL is incorrect. Later in this part, I'll show you two different methods for achieving AI qualification times that closely match their real world counterparts.

The following table shows relative race performance based on the actual 1967 historical data versus the relative race performance in GPL at Monza using our new hype settings. The mythical driver who won every race would receive a value of 1.00.

Relative Race Performance With Settings #1
Npt_override = 1.00

Driver	1967 Actual	AI Driver	GPL Monza
Gurney	.991	Gurney	.995
Clark	.990	Clark	.992
Hulme	.989	Hulme	.989
Stewart	.985	Stewart	.988
Brabham	.985	Hill	.986
Hill	.984	Brabham	.984
Surtees	.978	Surtees	.983
Rindt	.977	Rindt	.977
Parkes	.973	Parkes	.976
Amon	.970	Amon	.972
Spence	.958	Spence	.963
McLaren	.957	McLaren	.959
Rodriquez	.948	Rodriquez	.951
Siffert	.943	Siffert	.947
Anderson	.941	Anderson	.942
Irwin	.941	Bonnier	.941
Bonnier	.938	Irwin	.935
Ligier	.919	Ligier	.921
		Average Difference:	0.27%

We see that the AI drivers closely match the desired race performance and position and are even better than the qualification performance. Therefore, our new AI parameter settings work extremely well during a race as well.

Just for fun, I did a test that compared the relative qualifying performance of the AI using the three different driver.ini files created by myself, Kuratko, and David. I computed the relative qualifying performance for each driver compared to Clark who was the pole winner with all three driver.ini files. I then adjusted Kuratko's and David's results to artificially assign a performance rating of .993 for Clark and computed the relative performance of the remaining drivers in relation to Clark. This was necessary as the pole winner in each of our driver.ini files qualified at slightly different times. So to compare apples to apples, I had to make this simple adjustment. Rest assured that each AI driver's relative performance is measured the same in this table.

Relative Qualifying Performance @ Monza

Driver	1967 Actual	My AI #1		Kuratko AI		Wright AI	
Clark	.993	.993		.993		.993	
Brabham	.986	.984		.982		.973	<===
Gurney	.984	.984		.993	<===	.984	
Hill	.981	.984		.989		.992	<===
Hulme	.979	.974		.983		.972	
Amon	.976	.975		.973		.981	
Surtees	.972	.974		.981		.966	
Rindt	.967	.955	<===	.968		.961	
Stewart	.966	.954	<===	.985	<===	.963	
Parkes	.963	.963		.967		.974	<===
McLaren	.961	.958		.964		.974	<===
Rodriquez	.958	.962		.961		.953	
Spence	.952	.950		N/A		.953	
Siffert	.944	.949		.954	<===	.946	
Irwin	.941	.953	<===	.958	<===	.935	
Bonnier	.933	.939		.944	<===	.929	
Anderson	.933	.935		.943	<===	.917	<===
Ligier	.901	.911	<===	.927	<===	.903	
Average Difference		0.46%		0.78%		0.62%	

A few things stand out. It's easy to see that we three do not agree on settings for all drivers. Kuratko and I disagree on Gurney, Stewart, Siffert, Irwin, Bonnier, Anderson, and Ligier. David and I disagree on Brabham, Hill, Parkes, McLaren, and Anderson. In aggregate, my AI settings are "off" by about .46% while Kuratco's and David's are "off" by about .78% and .62% respectively.

I'm not completely happy with my settings for Rindt, Stewart, Irwin, and Ligier either. Stewart's AI performs below expected while Irwin's AI performs too well. I can't blame this on car performance at Monza either as both Stewart and Irwin drove BRMs. It would be easy to tweak the qualification settings to better match the historical requirement, but I wanted to avoid doing so. Regardless, I think this AI performs well.

Now there are a couple of reasons why each of us "experts" might develop AI settings that produce different results. One is that we may disagree on the effect of each AI parameter setting. Kuratko has explained in intricate detail how he derived his AI parameter settings, but hasn't shown his evaluation of the effect of each parameter setting within the GPL program. David merely gives us his driver.ini file without any explanation of how he determined the settings. Another possible reason for our differences (and the most probable) is that we are actually trying to make the AI perform differently. Perhaps Kuratko believes that Stewart was a better qualifier than David and I do. And perhaps David thinks that Irwin was far worse than Kuratko and I. Again, Kurtako at least explains how he derived his parameter settings while David does not.

I don't think this all bad. Of course, I believe my AI is the best of the three, but this is based on the assumption that my formulas for measuring real world driver performance are most correct. The others, particularly Kuratko, have used different formulas to measure historical performance and would naturally get different results. I'm sure Kuratko and David think their AI is the best too!

ACHIEVING HISTORICAL QUALIFYING TIMES

As mentioned before, the new AI parameters do very well both during qualification and a race; however, the pole winner's times are slower than the historical records. How can we speed up the AI so that they qualify near the real world times? There are two methods for doing so:

1. Npt_override Method

With the new AI settings, Clark qualifies at Monza at 90.40 seconds...slightly slower than we would like. To increase our AI Clark's performance so that he qualifies about 88.50 seconds where the real world Jimmy did, we can decrease npt_override to about .975. As noted before, changing npt_override from 1.00 to control the AI field causes a problem with some AI drivers who qualify better than they should; however, their race performance is NOT affected by this problem. Also, because each track is modeled differently in GPL, a good npt_override setting for Monza doesn't work perfectly for all tracks.

After a bit of experimentation, I found that an npt_override of .960 is actually the best setting for achieving the closest qualifying times for all tracks. The following table shows the effect of lowering npt_override to .960 with our new driver parameter settings on the pole winner's time:

1967 Actual Versus AI Pole Winner's Qualification Times
AI Settings #1
Npt_override = .960

Track	1967 Pole Time	AI Pole Time	Difference	
Kyalami	88.30	79.95	N/A	Track incorrectly modeled
Monaco	87.60	87.08	-0.59%	
Zandvoort	84.60	85.92	+1.56%	
Spa	208.10	197.54	-5.07%	
LeMans Bugatti	96.20	96.90	+0.73%	
Rouen	N/A	117.03	N/A	Track not used in 1967
Silverstone	85.30	89.19	+4.56%	
Nurburging	484.10	490.84	+1.39%	
Mosport	82.40	81.75	-0.79%	
Monza	88.50	87.66	-0.95%	
Watkins Glen	65.48	64.66	-1.25%	
Mexico	107.56	108.45	+0.83%	
		Average Difference:	+0.04%	

We see that using an npt_override of .960 works extremely well with the new AI parameters in achieving qualifying times that closely approximate the actual pole winner's time. It's not perfect as the times at Spa and Silverstone indicate, but the times are very close.

Note that GPL's version of Kyalami is incorrectly modeled…the track length is too short. Rouen was not used for the French Grand Prix in 1967 so there was no qualifying time for that track. Spa and Silverstone also may not be correctly modeled.

2. Higher Hype Method

Using a lower npt_override is a perfectly acceptable method of achieving qualifying times that approximate the real world, historical times, but how can we set the AI parameters so that our AI Clark does a qualifying lap of 88.50 seconds at Monza with an npt_override of 1.00? In order to achieve that speed, we must use a higher hype setting. The method by which we achieve the higher speed is simple...increase each driver's adjusted performance in relation to the amount the AI pole winner is below the historical qualifying time.

We already know each driver's race and qualifying performances relative to the baseline driver. If we knew how well the baseline driver qualifies at each track, we can use an average of his times to calculate the necessary performance adjustments for each AI driver. The following table depicts the actual 1967 pole winner's times versus the baseline driver's qualification times using an npt_override setting of 1.00:

1967 Pole Winner Versus Baseline Driver Qualification Times
Npt_override=1.00

Track	1967 Pole Time	AI Pole Time	Difference	
Kyalami	88.30	81.99	N/A	Track incorrectly modeled
Monaco	87.60	89.55	+2.23%	
Zandvoort	84.60	88.15	+4.20%	
Spa	208.10	203.07	-2.42%	
LeMans Bugatti	96.20	99.50	+3.43%	
Rouen	N/A	120.05	N/A	Track not used in 1967
Silverstone	85.30	91.43	+7.19%	
Nurburging	484.10	504.86	+4.29%	
Mosport	82.40	84.06	+2.01%	
Monza	88.50	89.78	+1.45%	
Watkins Glen	65.48	66.37	+1.36%	
Mexico	107.56	111.34	+3.51%	
		Average Difference:	+2.72%	

On the average, the baseline driver qualifies 2.72% slower than the historical pole winner if the times at Kyalami and Rouen are thrown out for reasons previously discussed. Therefore, we need to increase each driver's qualifying performance by at least 2.72%. Because each AI driver is slower than the baseline driver, we need to make an adjustment for that as well. What we are trying to achieve is for the fastest qualifying driver (which is usually Jimmy Clark) to qualify near the historical pole winner's time. To do so, we must also increase Clark's speed by the reciprocal of his qualifying performance because he is measured relative to the baseline driver. If we adjust Clark's speed, we also must adjust every other driver by exactly the same amount for them to qualify in the same relative time and position.

The following formula is used to adjust the AI driver's qualification performance:

$$\text{Adjusted Qual Perf} = 1.0272 \times \text{Qual Perf} \times \frac{1}{\text{Fastest AI Qual Driver's Qual Perf}}$$

Which simplifies to:

$$\text{Adjusted Qual Perf} = \frac{1.0272 \times \text{Qual Perf}}{\text{Fastest AI Qual Driver Qual Perf}}$$

In Jimmy's case, his original qualification performance value was .993 and when we substitute this into the formula we get:

$$\text{Clark's Adjusted Qual Perf} = \frac{1.0272 \times .993}{.993} = 1.027$$

Another example will make this formula easier to understand so let's look at Denny Hulme. Hulme's original qualification performance when adjusted for his car is .986. We need to increase this performance before setting the hype and qualifying parameters. Remember that we still divide by Clark's original qualifying performance value of .993.

$$\text{Hulme's Adjusted Qual Perf} = \frac{1.0272 \times .986}{.993} = 1.020$$

We do essentially the same thing to find an adjusted race performance. We still must divide by Clark's qualifying performance value of .993 even though we are computing race performance. The formula is:

$$\text{Adjusted Race Perf} = \frac{1.0272 \times \text{AI Driver Race Perf}}{\text{Fastest AI Qual Driver Qual Perf}}$$

Hulmes' original race performance value when adjusted for his car is .996. The following example shows his new race performance value:

$$\text{Hulmes' Adjusted Race Perf} = \frac{1.0272 \times .996}{.993} = 1.030$$

The following table shows the new adjusted race and qualifying performance values when adjusted for the car and track:

1967 Driver Performance Adjusted For Car And Track

Driver	Adj Qual Perf	Adj Race Perf
Hulme	1.020	1.030
Brabham	1.027	1.026
Clark	1.027	1.023
Surtees	1.021	1.028
Amon	1.012	1.005
Rodriquez	1.004	0.995
Hill	1.014	1.018
Gurney	1.019	1.026
Stewart	1.015	1.035
Spence	1.000	1.007
Rindt	1.014	1.025
Siffert	0.990	0.989
McLaren	0.995	0.991
Bonnier	0.979	0.983
Anderson	0.972	0.980
Parkes	0.998	1.013
Irwin	0.989	0.988
Ligier	0.939	0.957

Using the new adjusted race and qualifying performance values adjusted for the car and track, we compute new hype and qualifying settings just as before. The following table shows the new AI parameters:

Driver.ini File Settings #2

Driver	Hype	Qual
Hulme	1.042	0.896
Brabham	1.036	1.004
Clark	1.033	1.022
Surtees	1.039	0.931
Amon	1.010	1.078
Rodriquez	0.996	1.173
Hill	1.026	0.948
Gurney	1.037	0.919
Stewart	1.048	0.859
Spence	1.012	0.911
Rindt	1.035	0.893
Siffert	0.989	1.002
McLaren	0.992	1.032
Bonnier	0.983	0.918
Anderson	0.979	0.879
Parkes	1.013	0.888
Irwin	0.989	0.988
Ligier	0.952	0.818

Bandini's new hype and qualifying settings are 1.000 and 1.250 respectively.

Overall, the second set of AI parameter settings work very well at all tracks too as shown in the following table. Just remember that these new settings use an initial npt_override of 1.00, but you can change that to control the AI field if desired.

1967 Actual Versus AI Pole Winner's Qualification Times
AI Settings #2
Npt_override = 1.00

Track	1967 Pole Time	AI Pole Time	Difference	
Kyalami	88.30	79.60	N/A	Track incorrectly modeled
Monaco	87.60	89.06	+1.67%	
Zandvoort	84.60	85.46	+1.02%	
Spa	208.10	197.11	-5.28%	
LeMans Bugatti	96.20	96.67	+0.49%	
Rouen	N/A	116.54	N/A	Track not used in 1967
Silverstone	85.30	88.80	+4.10%	
Nurburging	484.10	488.82	+0.98%	
Mosport	82.40	81.33	-1.30%	
Monza	88.50	87.24	-1.42%	
Watkins Glen	65.48	64.39	-1.66%	
Mexico	107.56	107.85	+0.27%	
		Average Difference:	-0.11%	

PART V

1965 MODIFICATION

Now that we've done our homework on the 1967 drivers, let's take a look at the drivers found in the 1965 modification. We'll apply the same techniques to derive driv65.ini settings that will make the AI perform very closely to their historical counterparts.

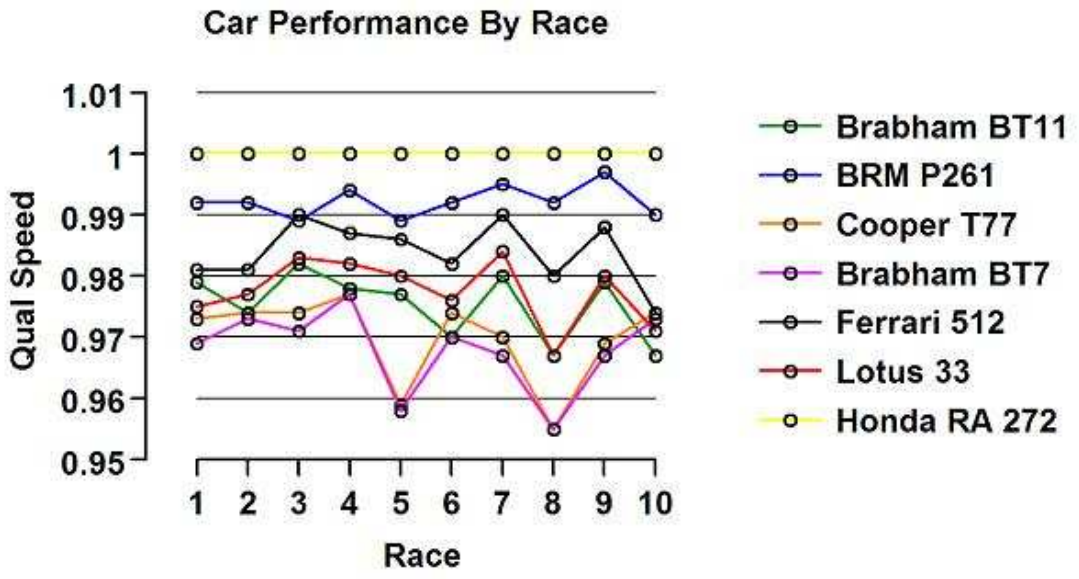
1965 CAR PERFORMANCE

As with the 1967 cars, each car as modeled by the 1965 modification performs differently. This a great testament to the modification design team's pursuit of realism and makes the simulation much more exciting and challenging for the player. After driving each, it is easily apparent how different each car really is. They certainly handle and accelerate/decelerate differently.

To test each AI car's performance, we simply modify the driv65.ini file so that the lead driver for each car has all his driver parameters set to 1.00. In other words, we put the baseline driver into all seven cars and then measure his performance. Differences among the results then is a measure of the car's performance, not the driver's.

To complicate matters, each car performs differently at each track. Therefore, it is necessary to test each car at every track, record its relative performance with the fastest car's performance artificially set to 1.000, then average the results over all tracks. In 1965, different tracks were used for the world championship than in 1967. The New London track was used at the South African Grand Prix while Clermont-Ferrand was used at the French Grand Prix. There was no Canadian Grand Prix at Mosport that year.

The following graph shows the relative performance of each car at each track:



The following table compares the average relative performance of each AI car at all tracks as modeled by the 1965 modification. The Honda is the fastest car overall and is assigned an arbitrary value of 1.0000. The other cars are "normalized" to the Honda performance so that we can easily see the performance differences.

1965 Average Car Performance

Brabham BT11	BRM P261	Cooper T77	Brabham BT7	Ferrari 512	Lotus 33	Honda
.9753	.9921	.9699	.9681	.9840	.9774	1.0000

The Honda is the fastest car, but the BRM is very close behind followed by the Ferrari. The Brabham BT11 and the Lotus are in the middle of the pack while the Cooper and Brabham BT7 bring up the rear. Remember that these are the performance differences as modeled by the 1965 modification. The actual differences are only as good as the model the 1965 modification team used in designing the simulation. However from all accounts, it appears that they did a very good job in doing so.

TESTS OF AI QUALIFICATION PERFORMANCE

I conducted a series of tests that examined the effect of changes to driver parameters on the driver's qualifying performance. All tests were done using the baseline driver in the Honda on the Monza track. By convention in all the tables and graphs in this tutorial, higher relative performance is defined as higher speed/lower lap time. The following table shows the results of these tests.

1965 Relative Qualifying Performance

	Aggr	Alert	Exp	Hype	Quick	Smooth	Qual
0.80 |	1.001	1.001	N/A	0.835	0.956	0.920	0.956
0.90 |	.999	1.001	0.996	0.918	0.989	0.990	0.990
1.00 |	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.10 |	1.000	1.000	1.001	1.082	1.005	1.003	1.007
1.20 |	N/A	1.000	1.003	1.155	1.012	1.003	1.011

N/A--no value obtained

It comes as no surprise that the 1965 qualifying performance values are very similar to the 1967 values. Once again, aggression, alertness, experience, and smoothness have little to no effect on qualifying performance. Hype has the greatest effect while quickness and qualifying have identical yet smaller effects.

TESTS OF AI RACE PERFORMANCE

Because the 1965 qualifying results are so similar to the 1967 results, a test for race performance is unnecessary. Only hype changes are necessary to adjust the AI's race performance.

TESTS OF HYPE VERSUS QUALIFYING ON 1965 QUALIFYING PERFORMANCE

We learned earlier that only hype, quickness, and qualifying have a measurable effect on qualifying performance. We also learned from the 1967 tests that using hype alone provides sufficient adjustment to vary race performance. Therefore, we now need to know how varying both hype and qualifying affect qualifying performance.

The following table shows how hype and qualifying affect qualifying performance.

1965 Qualifying Performance
Hype Versus Qualifying

		Hype			
		.80	.90	1.00	1.10
	0.70	0.825	0.857	0.906	1.036
	0.80	0.831	0.899	0.956	1.069
	0.90	0.834	0.913	0.988	1.077
Qualifying	1.00	0.835	0.916	1.000	1.081
	1.10	0.841	0.924	1.007	1.087
	1.20	0.844	0.928	1.010	1.092

For example, with a hype of 1.10 you can adjust the qualification performance from 103.6% to 109.2% of the baseline. With a hype of .80, you can adjust the qualification performance from only 82.5% to 84.4% of the baseline. Thus at the lower hype settings, you have much less range in which to adjust qualification. For those drivers whose qualifying performance was worse than their race performance, you should have sufficient adjustment range; however, for those drivers who qualify much better than their race performance, you may not have enough adjustment.

QUANTIFYING 1965 HISTORICAL DRIVER PERFORMANCE

We're going to use the same techniques to quantify the 1965 drivers' performances as we did with 1967 drivers. First, lets show the historical data found from online sources.

1965 Average Gap to Pole Position
In Percent

Driver	Gap
Clark	0.156
Hill	0.738
McLaren	2.385
Gurney	1.317
Ginther	2.020
Bonnier	2.272
Surtees	0.764
Stewart	1.059
Spence	1.671
Brabham	1.777
Hulme	2.200
Bandini	1.712
Rindt	3.070
Bucknum	2.821
Siffert	2.999
Attwood	3.728
Anderson	3.773
Ireland	4.478
Garnder	4.521

1965 Average Grid Position

Driver	Position
Clark	1.333
Hill	3.600
McLaren	9.700
Gurney	6.111
Ginther	7.000
Bonnier	11.200
Surtees	4.000
Stewart	4.800
Spence	7.111
Brabham	6.857
Hulme	9.333
Bandini	7.300
Rindt	11.667
Bucknum	11,667
Siffert	11.700
Attwood	14.125
Anderson	9.857
Ireland	13.857
Garnder	14.429

1965 Average Race Time
In Percent Of Winner

Driver	Time
Clark	100.00
Hill	99.00
McLaren	97.30
Gurney	99.60
Ginther	97.80
Bonnier	96.80
Surtees	98.80
Stewart	99.20
Spence	97.80
Brabham	96.70
Hulme	98.20
Bandini	97.00
Rindt	95.60
Bucknum	91.80
Siffert	95.90
Attwood	95.80
Anderson	86.30
Ireland	96.70
Garnder	96.60

1965 Average Finish Position

Driver	Position
Clark	2.290
Hill	2.780
McLaren	5.600
Gurney	4.140
Ginther	6.800
Bonnier	7.200
Surtees	3.800
Stewart	3.000
Spence	6.290
Brabham	5.000
Hulme	5.670
Bandini	7.220
Rindt	8.600
Bucknum	5.000
Siffert	8.000
Attwood	10.170
Anderson	9.000
Ireland	10.670
Garnder	10.330

Using the same formulas as before, I compute the actual drivers' performances as:

1965 Actual Driver Performance

Driver	Qual Perf	Race Perf
Clark	.998	.994
Hill	.990	.986
McLaren	.966	.964
Gurney	.981	.982
Ginther	.975	.960
Bonnier	.961	.953
Surtees	.989	.980
Stewart	.985	.986
Spence	.976	.963
Brabham	.976	.964
Hulme	.968	.968
Bandini	.976	.954
Rindt	.958	.940
Bucknum	.959	.939
Siffert	.958	.945
Attwood	.949	.933
Anderson	.959	.892
Ireland	.945	.935
Garnder	.944	.936

As an aside, Clark obviously was unbeatable in 1965. He completely dominated the sport.

We also must adjust the driver's qualification and race performances for his car performance so that the driver performs correctly within the simulation. This technique is fully explained in Part III for the 1967 drivers. The following table shows each driver's 1965 real world performance when adjusted for his car:

1965 Driver Performance Adjusted For Car

Driver	Adj Qual Perf	Adj Race Perf
Clark	1.022	1.017
Hill	.998	.994
McLaren	.996	.993
Gurney	1.006	1.007
Ginther	.975	.960
Bonnier	.993	.985
Surtees	1.005	.996
Stewart	.993	.994
Spence	.999	.985
Brabham	1.002	.988
Hulme	1.000	1.000
Bandini	.992	.969
Rindt	.988	.969
Bucknum	.959	.939
Siffert	.983	.969
Attwood	.971	.955
Anderson	.984	.914
Ireland	.968	.957
Garnder	.968	.960

This table is interesting because it shows how a driver's performance compared to his contemporaries assuming they all drove the Honda as modeled by GPL. Clark was the best qualifier, as expected, and also was the best race performer. Dan Gurney came closest to matching Clark which supports the story in which Clark's father told Gurney that Jimmy "feared" him the most...a great tribute from one of Formula One's greatest drivers.

If we were to compute driver hype and qualifying settings based only on their car adjusted performances, we could compare how closely the settings achieve our goal of each driver qualifying closely to their historical performance. Although these settings result in qualifying times that are slower than the historical times, they still may be used for a valid comparison of the driver's relative performance.

The following table compares how each driver qualifies at Monza with car adjusted hype and qualifying settings (not shown) versus their actual overall 1965 performance:

1965 Relative Qualifying Performance
Npt_Override = 1.00

Driver	1965 Actual	AI Driver	GPL Monza
Clark	.998	Clark	.995
Hill	.990	Surtees	.990
Surtees	.989	Hill	.989
Stewart	.985	Stewart	.984
Gurney	.981	Brabham	.980
Spence	.976	Gurney	.978
Ginther	.975	Spence	.978
Brabham	.976	Bandini	.973
Bandini	.976	Ginther	.972
Hulme	.968	Hulme	.969
McLaren	.966	McLaren	.966
Bonnier	.961	Bonnier	.962
Bucknum	.959	Siffert	.958
Anderson	.959	Bucknum	.957
Rindt	.958	Rindt	.950
Siffert	.958	Ireland	.949
Attwood	.949	Attwood	.948
Ireland	.945	Gardner	.943
Garnder	.944	Anderson	.906
		Average Difference:	0.35%

ACHIEVING HISTORICAL QUALIFYING TIMES

The new AI parameters do very well both during qualification and a race; however, the pole winner's times are slower than the historical records. We can adjust the driver's hype setting to achieve more realistic qualification times.

We already know each driver's race and qualifying performances relative to the baseline driver. If we knew how well the baseline driver qualifies at each track, we can use an average of his times to calculate the necessary performance adjustments for each AI driver. The following table depicts the actual 1965 pole winner's times versus the baseline driver's qualification times using an npt_override setting of 1.00:

1965 Pole Winner Versus Baseline Driver Qualification Times
Npt_override=1.00

Track	1965 Pole Time	AI Pole Time	Difference
East London	87.20	86.37	-0.95%
Monaco	92.50	90.79	-1.85%
Spa	225.40	223.34	-0.91%
Clermont-Ferrand	198.30	209.36	+5.58%
Silverstone	90.80	96.90	+6.72%
Zandvoort	90.70	90.44	-0.29%
Nurburging	502.70	524.39	+4.31%
Monza	95.90	98.75	+2.97%
Watkins Glen	71.25	70.77	-0.67%
Mexico	116.17	116.13	-0.33%
		Average Difference:	+1.49%

On average, the baseline driver qualifies 1.49% slower than the historical pole. Therefore, we need to increase each driver's qualifying performance by at least 1.49%. Because each AI driver is slower than the baseline driver, we need to make an adjustment for that as well. What we are trying to achieve is for the fastest qualifying driver (which is usually Jimmy Clark) to qualify near the historical pole winner's time. To do so, we must also increase Clark's speed by the reciprocal of his qualifying performance because he is measured relative to the baseline driver. If we adjust Clark's speed, we also must adjust every other driver by exactly the same amount for them to qualify in the same relative time and position. See Part IV for a complete discussion of how this is done for the 1967 drivers.

The following table shows the new race and qualifying performance values when adjusted for the car and track:

1965 Driver Performance Adjusted For Car And Track

Driver	Adj Qual Perf	Adj Race Perf
Clark	1.039	1.034
Hill	1.014	1.010
McLaren	1.013	1.010
Gurney	1.022	1.024
Ginther	.991	.976
Bonnier	1.009	1.001
Surtees	1.021	1.012
Stewart	1.010	1.010
Spence	1.016	1.002
Brabham	1.018	1.005
Hulme	1.017	1.016
Bandini	1.008	.985
Rindt	1.004	.985
Bucknum	.975	.955
Siffert	.999	.985
Attwood	.987	.971
Anderson	1.000	.929
Ireland	.984	.973
Garnder	.984	.976

Next, we compute a hype setting using a regression formula and the driver's adjusted race performance value. For the 1965 modification, the regression formula is:

Hype = (1.233 * Race Performance) - .233

Finally, we use an expanded HypeVersus Qualification table (not shown) to get a qualifying setting based on the driver's hype and adjusted qualification performance.

The following table shows the new AI parameters:

1965 driv65.ini File Settings

Driver	Hype	Qual
Clark	1.042	1.070
Hill	1.013	1.070
McLaren	1.012	1.250
Gurney	1.030	1.000
Ginther	.970	1.250
Bonnier	1.001	1.320
Surtees	1.015	1.120
Stewart	1.013	.980
Spence	1.002	1.250
Brabham	1.006	1.250
Hulme	1.020	1.200
Bandini	.982	1.250
Rindt	.982	1.400
Bucknum	.944	1.400
Siffert	.981	1.250
Attwood	.964	1.250
Anderson	.913	1.400
Ireland	.967	1.250
Garnder	.971	1.100

Interestingly, several drivers need a high qualification setting. Apparently, these drivers performed much better at qualifying in 1965 than they did during the race.

For the parameters of aggression, alertness, and experience, we will use the 1965 modification's default settings. Whether these values are correct or not, I don't know nor do I know how the modification team determined them. For quickness and smoothness, we'll use 1.00 for all drivers.

Overall, the new AI parameter settings work very well at all tracks as shown in the following table. Just remember that these settings use an initial npt_override of 1.00, but you can change that to control the AI field if desired.

1965 Actual Versus AI Pole Winner's Qualification Times
Npt_override = 1.00

Track	1965 Pole Time	AI Pole Time	Difference
East London	87.20	85.54	-1.90%
Monaco	92.50	91.77	-0.79%
Spa	225.40	220.31	-2.26%
Clermont-Ferrand	198.30	199.36	+0.53%
Silverstone	90.80	94.87	+4.48%
Zandvoort	90.70	89.19	-1.66%
Nurburging	502.70	513.11	+2.07%
Monza	95.90	96.98	+1.13%
Watkins Glen	71.25	69.87	-1.93%
Mexico	116.17	113.72	-2.11%
		Average Difference:	-0.25%

So that's it for the 1965 modification. We applied the same techniques and methods in deriving the 1965 AI settings as we did with the 1967 settings. Our test results show that the new settings work just as well.

PART VI

1969 MODIFICATION

Now that we're done with the 1965 and 1967 drivers, let's take a look at the drivers found in the 1969 modification. We'll apply the same techniques to derive drv69w.ini settings that will make the AI perform very closely to their historical counterparts.

1969 CAR PERFORMANCE

As with the 1965 and 1967 cars, each car modeled by the 1969 modification performs differently. The 1969 design team has done a great job in replicating the physics of Formula 1 cars with wing induced down force. After driving them, it is quickly apparent how different these cars are from the 1965 and 1967 versions. They certainly handle and accelerate/decelerate differently than the previous year's cars. To me, they are easier to drive than the 1965 cars while having the power of the 1967 cars. They are more controllable under braking and cornering and feel more stable.

The wing's the thing! Before we can measure each car's performance, we've got to determine what effect the wings have on the car. To do so, I did a test which compared the qualification times of the baseline driver in the Lotus 49B at Monza (a high speed track) and Monaco (a low speed track) using different wing settings. The next table shows the results of this test:

Wing Effect On Qualifying Performance

Player's Wing Setting	Monza Lap Time	Monaco Lap Time
No Wing	90.89	88.44
-5 Degrees	92.06	88.57
-10 Degrees	93.31	88.61
-15 Degrees	95.43	88.92
-20 Degrees	98.55	89.67

I was astonished to see these results even though I had been forewarned that the wing setting used by the player (that's you) had an effect on the AI as well. Until then, I assumed that GPL used a simple "fudge factor" to model the AI performance. However, the results of this test indicate that GPL actually uses the physics of each car to determine its performance. And if it uses the wing, it must also be using all the other factors that affect the car's performance such as horsepower, frontal area, suspension changes, tire slip angles, etc. We have known all along that GPL uses these factors in determining the performance of the player's car, but not about the AI. My hat is off to Papyrus for designing a truly remarkable simulation.

So back to the table. The first point to note is that a 1969 car is slower without a wing than a 1967 car even though they had similar engines and chassis (yes, the plural of chassis is chassis). The 1967 Lotus 49 which never had wings laps Monza at 89.35 seconds…over one second faster than the 1969 Lotus 49B without wings. Once you do add the wings, the lap times increase with increasing down force. According to the modification team, the wing stalls out around -15 degrees. My tests show that increasing the wing beyond -15 degrees results in even more drag and slower lap times. This result is consistent with established wing theory that drag increases up to and beyond the stall angle of attack; however, lift (down force) only increases up to the stall angle of attack and decreases above that.

The modification team believes that the best wing setting for the player is dependent on the track which is the way things work in real life. High speed tracks such as Monza need a lower wing setting; say around -5 degrees, while slow speed circuits such as Monaco need a higher wing setting; say around -15 degrees. Each track will need a different wing setting for best performance. You may also adjust the front and rear wings separately to enhance

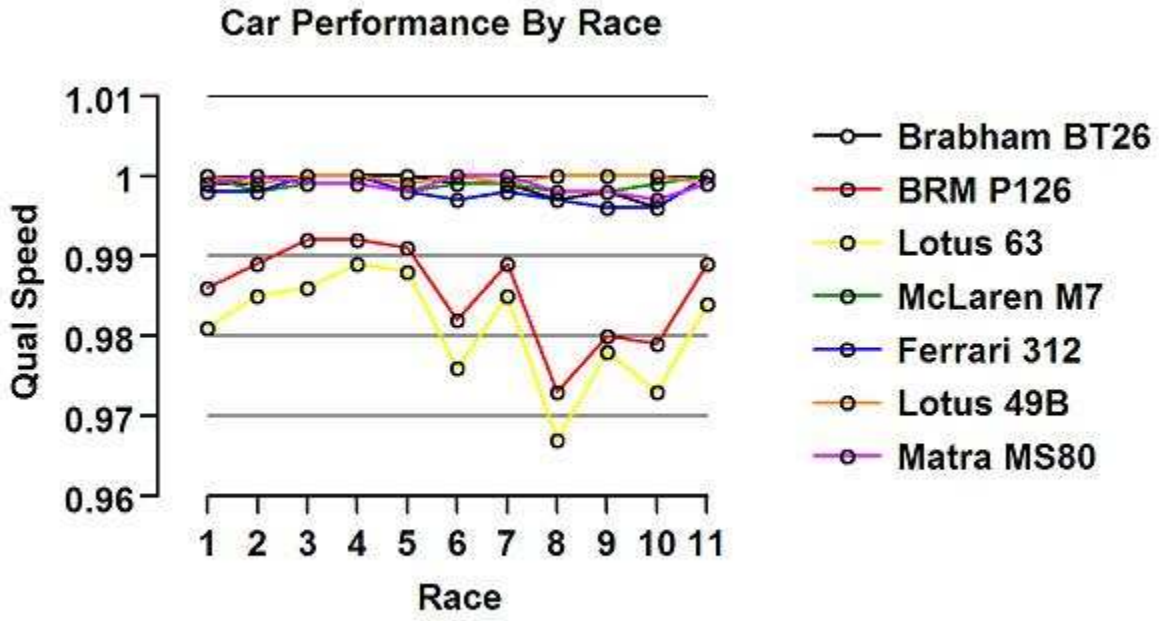
handling characteristics, but that subject is beyond the scope of the AI Tutorial.

What is important to us as AI designers is that the player's wing setting vastly affects the speed of the AI; particularly at the high speed tracks. We cannot set the AI's wings separately from the player's. In order to test each AI car performance, we've got to settle on a given wing setting; otherwise, we'd have to produce different drv69.ini files for each wing setting! After a little thought influenced by a couple of beers, I think that a wing setting of -10 degrees is a good compromise for testing the AI because it is an average of what most players will use. Yes, it's too much wing for Monza and too little for Monaco, but we'll just have to live with it.

To test each AI car's performance, we simply modify the drv69w.ini file so that the lead driver for each car has all of his driver parameters set to 1.00. In other words, we put the baseline driver into all seven cars and then measure his performance. Differences among the results then is a measure of the car's performance, not the driver's. The wing is set in the wg69i.ini file.

To complicate matters, each car performs differently at each track. Therefore, it is necessary to test each car at every track, record its relative performance with the fastest car's performance artificially set to 1.000, then average the results over all tracks. In 1969, different tracks were used for the world championship than in 1965 and 1967. The Kyalami track was used at the South African Grand Prix. Clermont-Ferrand was used at the French Grand Prix and the Montjuvich Park track was used at the Spanish Grand Prix. Although a GPL version of the Montjuich Park track is being developed, it has not been released. However with the 1969 design team's permission, I was able to obtain a beta copy of the track to complete testing at all eleven of the 1969 tracks.

The following graph shows the relative performance of each car at each track:



The following table compares the average relative performance of each AI car at all tracks as modeled by the 1969 modification. The Lotus 49B is the fastest car overall and is assigned an arbitrary value of 1.0000. The other cars are "normalized" to the Lotus 49B's performance so that we can easily see the performance differences.

1969 Average Car Performance

Brabham BT26	BRM P126	Lotus 63	McLaren M7	Ferrari 312	Lotus 49B	Matra
.9991	.9860	.9814	.9992	.9982	1.0000	.9991

The Lotus 49B is the fastest car, but just barely. In fact, all of the Ford-powered cars perform nearly identically which is what we would expect since they use the same motor. Even the Ferrari performs nearly as well. I was a bit surprised to see the Brabham perform so well as it has that huge front wing, but apparently that didn't make much difference. Just imagine being Jack Brabham or Jackie Ickx driving along at 180 mph with that huge wing wobbling around only inches from their face. No wonder the rules were changed later in the season to prohibit the high wing designs. The BRM is a step below the front runners while the Lotus 63 is clearly last. This new Lotus was a four-wheel drive car, but was too heavy to show off its technological advantage. Remember that these are the performance differences as modeled by the 1969 modification. The actual differences are only as good as the model the 1969 modification team used in designing the simulation. However, it appears that they did a very good job in doing so.

TESTS OF AI QUALIFICATION PERFORMANCE

I conducted a series of tests that examined the effect of changes to driver parameters on the driver's qualifying performance. All tests were done using the baseline driver in the Lotus 49B with -10 degrees of wing on the Monza track. By convention in all the tables and graphs in this tutorial, higher relative performance is defined as higher speed/lower lap time. The following table shows the results of these tests.

1969 Relative Qualifying Performance

	Aggr	Alert	Exp	Hype	Quick	Smooth	Qual
0.80 |	1.002	1.002	N/A	0.832	0.991	0.890	0.992
0.90 |	1.002	1.001	1.000	0.920	0.998	0.999	0.997
1.00 |	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.10 |	1.000	1.000	1.001	1.079	1.007	1.001	1.005
1.20 |	N/A	1.001	1.002	1.160	1.010	1.002	1.010

N/A--no value obtained

It comes as no surprise that the 1969 qualifying performance values are very similar to the 1965 and 1967 values. Once again, aggression, alertness, experience, and smoothness have little or no effect on qualifying performance. Hype has the greatest effect while quickness and qualifying have identical yet smaller effects.

TESTS OF AI RACE PERFORMANCE

Because the 1969 qualifying results are so similar to the 1965 and 1967 results, a test for race performance is unnecessary. Only hype changes are necessary to adjust the AI's race performance.

TESTS OF HYPE VERSUS QUALIFYING ON 1969 QUALIFYING PERFORMANCE

We learned earlier that only hype, quickness, and qualifying have a measurable effect on qualifying performance. We also learned from the 1967 tests that using hype alone provides sufficient adjustment to vary race performance. Therefore, we now need to know how varying both hype and qualifying affect qualifying performance.

The following table shows how hype and qualifying affect qualifying performance.

1969 Qualifying Performance
Hype Versus Qualifying

		Hype			
		.80	.90	1.00	1.10
	0.70	0.824	0.899	0.953	1.064
	0.80 |	0.828	0.911	0.992	1.072
	0.90 |	0.829	0.915	0.997	1.079
Qualifying	1.00 |	0.832	0.920	1.000	1.083
	1.10 |	0.832	0.922	1.005	1.083
	1.20 |	0.833	0.923	1.010	1.089

For example, with a hype of 1.10 you can adjust the qualification performance from only 106.4% to 108.9% of the baseline. With a hype of .80, you can adjust the qualification performance from only 82.4% to 83.3% of the baseline. Thus at the lower or higher hype settings, you have much less range in which to adjust qualification. For those drivers whose qualifying performance was close to their race performance, you should have sufficient adjustment range; however, for those drivers who qualify much better or worse than their race performance, you may not have enough adjustment.

QUANTIFYING 1969 HISTORICAL DRIVER PERFORMANCE

We're going to use the same techniques to quantify the 1969 drivers' performances as we did with the 1965 and 1967 drivers. First, let's show the historical data found from online sources.

1969 Average Gap to Pole Position
In Percent

Driver	Gap
Stewart	0.441
Ickx	2.625
McLaren	2.219
Rindt	0.424
Beltoise	3.310
Hulme	1.375
Hill	1.888
Courage	2.981
Siffert	2.113
Brabham	1.457
Surtees	3.513
Amon	1.219
Elford	5.811
Rodriquez	5.379
Servoz-Gavin	5.659
Moser	9.809
Oliver	4.654
Lovely	6.838
Miles	5.220
Andretti	4.250

1969 Average Grid Position

Driver	Position
Stewart	2.636
Ickx	6.091
McLaren	7.909
Rindt	2.200
Beltoise	8.364
Hulme	4.818
Hill	6.600
Courage	8.900
Siffert	7.364
Brabham	5.750
Surtees	10.700
Amon	4.000
Elford	11.600
Rodriquez	12.875
Servoz-Gavin	13.750
Moser	15.000
Oliver	12.500
Lovely	16.000
Miles	12.400
Andretti	10.333

1969 Average Race Time
In Percent Of Winner

Driver	Time	
Stewart	.998	
Ickx	.964	
McLaren	.978	
Rindt	.993	
Beltoise	.973	
Hulme	.971	
Hill	.969	
Courage	.980	
Siffert	.962	
Brabham	.987	
Surtees	.957	
Amon	.996	
Elford	.948	
Rodriquez	.955	
Servoz-Gavin	.946	
Moser	.924	
Oliver	.962	
Lovely	.927	
Miles	.898	
Andretti	N/A	Did not complete a race in 1969

1969 Average Finish Position

Driver	Position	
Stewart	1.500	
Ickx	3.750	
McLaren	3.880	
Rindt	2.500	
Beltoise	5.110	
Hulme	4.710	
Hill	5.140	
Courage	4.800	
Siffert	5.570	
Brabham	3.750	
Surtees	5.670	
Amon	3.000	
Elford	7.000	
Rodriquez	6.000	
Servoz-Gavin	7.000	
Moser	8.000	
Oliver	6.500	
Lovely	8.000	
Miles	10.000	
Andretti	4.800	Based on 1970, 1971, and 1972 performances. Did not complete a race in 1969

Using the same formulas as before, I compute the actual drivers' performances as:

1969 Actual Driver Performance

Driver	Qual Perf	Race Perf	
Stewart	.994	.997	
Ickx	.974	.968	
McLaren	.972	.975	
Rindt	.995	.989	
Beltoise	.965	.966	
Hulme	.984	.967	
Hill	.977	.964	
Courage	.965	.971	
Siffert	.974	.958	
Brabham	.981	.980	
Surtees	.958	.955	
Amon	.986	.988	
Elford	.944	.944	
Rodriquez	.943	.953	
Servoz-Gavin	.940	.943	
Moser	.916	.927	
Oliver	.948	.954	
Lovely	.928	.929	
Miles	.945	.904	
Andretti	.955	.961	Race Perf based on 1970, 1971, and 1972 finishing positions

As an aside, Jackie Stewart was the class of the field in 1969 when it came to the race. However, he and Jochen Rindt were equally good at qualifying.

Mario Andretti's race performance is an estimate based on his relative finishing positions in 1970, 1971, and 1972 as he failed to finish in any of the three races he entered in 1969.

We also must adjust the driver's qualification and race performances for his car performance so that the driver performs correctly within the simulation. This technique is fully explained in Part III for the 1967 drivers. The following table shows each driver's 1969 real world performance when adjusted for his car:

1969 Driver Performance Adjusted For Car

Driver	Adj Qual Perf	Adj Race Perf
Stewart	.995	.997
Ickx	.975	.969
McLaren	.972	.975
Rindt	.995	.989
Beltoise	.966	.967
Hulme	.984	.968
Hill	.977	.964
Courage	.966	.972
Siffert	.974	.958
Brabham	.982	.981
Surtees	.972	.969
Amon	.988	.990
Elford	.945	.945
Rodriquez	.957	.966
Servoz-Gavin	.941	.944
Moser	.917	.928
Oliver	.961	.967
Lovely	.928	.929
Miles	.963	.921
Andretti	.973	.979

This table is interesting because it shows how a driver's performance compared to his contemporaries assuming they all drove the Lotus 49B as modeled by GPL. Stewart and Rindt were the best qualifiers…locked in a dead heat for top gun. However, Stewart was the best race performer by a significant margin. In fact, Stewart's race performance was the best of any driver for the years 1965, 1967, and 1969; even overshadowing Jimmy Clark by a slight amount.

If we were to compute driver hype and qualifying settings based only on their car adjusted performances, we could compare how closely the settings achieve our goal of each driver qualifying near their historical performance. Although these settings result in qualifying times that are slower than the historical times, they still may be used for a valid comparison of the driver's relative performance.

The following table compares how each driver qualifies at Monza with car adjusted hype and qualifying settings (not shown) versus their actual overall 1969 performance:

1969 Relative Qualifying Performance
Npt_Override = 1.00

Driver	1969 Actual	AI Driver	GPL Monza
Rindt	.995	Rindt	.995
Stewart	.994	Stewart	.995
Amon	.986	Amon	.985
Hulme	.984	Hulme	.983
Brabham	.981	Brabham	.981
Hill	.977	Hill	.975
Ickx	.974	Ickx	.973
Siffert	.974	Siffert	.972
McLaren	.972	McLaren	.972
Courage	.965	Courage	.964
Beltoise	.965	Beltoise	.963
Surtees	.958	Surtees	.959
Andretti	.955	Andretti	.957
Oliver	.948	Oliver	.949
Miles	.945	Elford	.946
Elford	.944	Rodriquez	.945
Rodriquez	.943	Servoz-Gavin	.941
Servoz-Gavin	.940	Lovely	.927
Lovely	.928	Moser	.918
Moser	.916	Miles	.914
		Average Difference:	0.15%

Overall, the 1969 AI perform very well compared to their real world counterparts. The only problem is with John Miles who qualified much better in 1969 than his GPL counterpart. The reason for this lies in his 1969 race performance of .904 versus his qualification performance of .945. If we accurately set his race performance with a low hype setting, there is insufficient qualifying setting adjustment to achieve Miles' qualifying performance. As explained in Part I, it is easy to lower the AI driver's qualification performance, but sometimes difficult to raise it by much. My opinion is that it is better to set accurately the race performance and accept qualification performance differences as you, the player, will spend most of your time racing; not qualifying.

ACHIEVING HISTORICAL QUALIFYING TIMES

The new AI parameters do very well both during qualification and a race; however, the pole winner's times are slower than the historical records. We can adjust the driver's hype setting to achieve more realistic qualification times.

We already know each driver's race and qualifying performances relative to the baseline driver. If we knew how well the baseline driver qualifies at each track, we could use an average of his times to calculate the necessary performance adjustments for each AI driver. The following table depicts the actual 1969 pole winner's times versus the baseline driver's qualification times using an npt_override setting of 1.00:

1969 Pole Winner Versus Baseline Driver Qualification Times
Npt_override=1.00

Track	1969 Pole Time	AI Pole Time	Difference
Kyalami	80.00	83.12	+3.90%
Montjuich Park	85.70	94.74	+10.55%
Monaco	84.60	88.72	+4.87%
Zandvoort	80.85	88.71	+9.72%
Clermont-Ferrand	180.60	199.55	+10.49%
Silverstone	80.80	92.54	+14.53%
Nurburging	462.10	508.22	+9.98%
Monza	85.48	93.52	+9.41%
Mosport	77.40	84.55	+9.24%
Watkins Glen	63.62	67.44	+6.00%
Mexico	102.90	112.16	+9.00%
		Average Difference:	+8.88%

On average, the baseline driver qualifies 8.88% slower than the historical pole winner. Therefore, we need to increase each driver's qualifying performance by at least 8.88%. Because each AI driver is slower than the baseline driver, we need to make an adjustment for that as well. What we are trying to achieve is for the fastest qualifying driver (which is always Jackie Stewart or Jochen Rindt) to qualify near the historical pole winner's time. To do so, we must also increase Stewart's or Rindt's speed by the reciprocal of their qualifying performance because they are measured relative to the baseline driver. If we adjust their speed, we also must adjust every other driver by exactly the same amount for them to qualify in the same relative time and position. See Part IV for a complete discussion of how this is done for the 1967 drivers.

As an aside, this 8.88% difference is considerably larger than the differences obtained with the 1965 and 1967 AI. The reason for this is twofold. First, the 1969 modified cars even without wings are slower than the 1967 cars without wings. Add -10 degrees of wing and the cars slow down even more due to the increased drag. This is not a criticism of the 1969 modification, rather it is result of the physics involved. I'm sure this same thing occurred in real life. If you think for a moment, the 1969 cars without wings were essentially the same as the 1967 cars in performance. The addition of wings and larger tires substantially increased the car's drag which slowed down their straight line speed even though it vastly increased their cornering capability. The net effect is that the AI are slower, but you as the driver will be faster. I knocked a full two seconds off my Watkins Glen personal best on my first day of testing the modification and only my second lap with the McLaren. So the wings definitely help. Second, a couple of the GPL tracks are suspect…they may be longer than the real world tracks (think Clermont-Ferrand and Silverstone).

The following table shows the new race and qualifying performance values when adjusted for the car and track:

1969 Driver Performance Adjusted For Car And Track

Driver	Adj Qual Perf	Adj Race Perf
Stewart	1.088	1.092
Ickx	1.067	1.061
McLaren	1.064	1.067
Rindt	1.089	1.082
Beltoise	1.057	1.058
Hulme	1.077	1.059
Hill	1.069	1.055
Courage	1.057	1.064
Siffert	1.065	1.049
Brabham	1.074	1.073
Surtees	1.064	1.060
Amon	1.081	1.083
Elford	1.034	1.034
Rodriquez	1.047	1.057
Servoz-Gavin	1.029	1.033
Moser	1.003	1.015
Oliver	1.052	1.058
Lovely	1.016	1.016
Miles	1.054	1.008
Andretti	1.065	1.072

Next, we compute a hype setting using a regression formula and the driver's adjusted race performance value. For the 1969 modification, the regression formula is:

Hype = (1.227 * Race Performance) - .224

Finally, we use an expanded HypeVersus Qualification table (not shown) to get a qualifying setting based on the driver's hype and adjusted qualification performance.

The following table shows the new AI parameters:

1969 drv69w.ini File Settings

Driver	Hype	Qual
Stewart	1.115	.925
Ickx	1.077	1.075
McLaren	1.086	.925
Rindt	1.104	1.150
Beltoise	1.074	.950
Hulme	1.075	1.400
Hill	1.070	1.250
Courage	1.081	.900
Siffert	1.063	1.250
Brabham	1.093	1.100
Surtees	1.077	1.400
Amon	1.105	1.000
Elford	1.045	1.000
Rodriquez	1.073	1.050
Servoz-Gavin	1.043	.900
Moser	1.022	.750
Oliver	1.075	1.100
Lovely	1.023	.880
Miles	1.013	1.600
Andretti	1.091	1.150

All drivers need a high hype setting to compensate for the slower speeds of the 1969 modified cars.

For the parameters of aggression, alertness, and experience, we will use the 1969 modification team's default settings. Whether these values are correct or not, I don't know nor do I know how the modification team determined them. For quickness and smoothness, we'll use 1.00 for all drivers.

Overall, the new AI parameter settings work reasonably well at all tracks as shown in the following table. Just remember that these settings use an initial npt_override of 1.00, but you can change that to control the AI field if desired.

1969 Actual Versus AI Pole Winner's Qualification Times
Npt_override = 1.00

Track	1969 Pole Time	AI Pole Time	Difference
Kyalami	80.00	75.69	-5.39%
Montjuich Park	85.70	85.76	+0.07%
Monaco	84.60	87.12	+2.98%
Zandvoort	80.85	80.11	-0.92%
Clermont-Ferrand	180.60	183.64	+1.68%
Silverstone	80.80	84.93	+5.11%
Nurburging	462.10	460.74	-0.29%
Monza	85.48	85.73	+.29%
Mosport	77.40	74.43	-3.84%
Watkins Glen	63.62	59.81	-5.99%
Mexico	102.90	101.86	-1.01%
		Average Difference:	-0.66%

So that's it for the 1969 modification. We applied the same techniques and methods in deriving the 1969 AI settings as we did with the 1965 and 1967 settings. Our test results show that the new settings work just as well.

PART VII

AI CAR RELIABILITY

AI car reliability is one area that has not had a lot of attention paid to it. Obviously, some cars were more reliable than others. For example, the Brabham of 1967 was an extremely reliable car that was the major factor in Denny Hulme's championship that year. Although Jimmy Clark in his Lotus 49 was the class of the field, the Lotus simply wasn't reliable enough to give Jimmy his third Formula One championship.

The GPL designers considered car reliability to be important enough to model it within the GPL.exe program. As we shall see, the gpl_ai.ini and driver.ini files contain settings that affect the AI car reliability. While it appears that we cannot fully control reliability, at least we can influence it to some extent. This Part is an explanation of the research I've conducted on the AI car reliability. As far as I know, this is the first time that extensive research has been done in this important area.

Unlike tests of parameters in the driver.ini file that affect a driver's speed and lap times, reliability testing is extremely time consuming. There is large randomness in the number and type of malfunctions obtained during a race. Because the results are so variable, it takes many, many test races to even approximate the effect of changing the gpl_ai.ini and driver.ini file parameters on car reliability. For example, it normally takes only three or four tests to get a very good idea of how the driver.ini file parameters, such as hype and quickness, affect a driver's qualification or race time. However with car reliability testing, twenty separate races or more at each data point are needed to begin to approximate a car's reliability. From a statistical standpoint, as many as 400 separate races for each data point would be necessary to achieve 95% confidence that the test value was accurate to within 5% of the actual reliability! Clearly, doing this many tests is impossible. So for our examination of AI car reliability, far fewer tests were done. While this causes less confidence in our test results and conclusions, we may still make some general observations and derive settings that appear to control the AI car reliability reasonably well. For most of the tests, I only did 20 races which equates to 90% confidence that the test value is accurate to within 18% of the actual reliability. Therefore, consider the following information as preliminary until someone has the time and patience to do more research.

So let's get started.

GPL_AI.INI FILE TESTS

The gpl_ai.ini file [Mechanical] section contains various parameters that apparently affect <u>all</u> AI cars. These parameters include problem/failure chances for individual malfunctions such as engine and suspension and also the effect of each failure on the adjusted traction circle, etc. These malfunction chances sum to 100%. You could assume that these values alone set the malfunction distribution for all cars, but later tests show that this isn't entirely the case.

Here is a list of the gpl_ai.ini file malfunction chance settings:

Gpl_ai.ini File Settings

Category	Setting	Category Subtotal
Brake failure	1%	
Brake problem	2%	3%
Coolant leak	2%	2%
Engine failure	16%	
Engine problem	23%	39%
Fuel leak	2%	
Fuel failure	5%	
Fuel problem	7%	14%
Oil leak	10%	10%
Suspension failure	10%	
Suspension problem	15%	25%
Tire failure	3%	
Tire problem	4%	7%
	100%	100%

For example, we see that the brake failure percentage is only 1%. In addition, the brake problem percentage is an additional 2%. So the total percentage of brake related malfunctions is 3%. For the remainder of this Part, I will refer to malfunctions as the combination of failures and problems. It appears to me that failures invariably cause a car to stop...its race is immediately over. However, problems may only degrade a car's handling or engine performance. The car may continue for several laps or even finish the race at a slower pace. If you compare the chances of failures versus problems, the percentages are just about evenly split between the two.

In addition to these settings, the gpl_ai.ini file contains two other important parameters. These settings, copied exactly from the gpl_ai.ini file, are:

1. mechanical_failure_chance = 5.000000 ; chance in 10000 (!!) per ACTIVE AI car for induced problem/failure each interval
2. mechanical_failure_interval = 540.000000 ; average interval in ticks to check for mechanical problem/failure (gets randomized)

As a race proceeds, the gpl.exe program periodically checks each car for a malfunction. The first parameter sets the chance that a malfunction occurs each time the program does a check. Note that it is measured in 10 thousandths with the default being .0005. This value is per active AI car which means that the actual malfunction chance is .0005 multiplied by the number of active AI cars. At the race start with 19 active cars, the malfunction value is 19 * .0005 = .0095 or about 95 chances in ten thousand. As the number of AI cars decreases throughout the race, the chance of malfunction goes down. For example, with only 10 cars left, the chance of failure is 10 * .0005 = .0050 or about 50 chances in ten thousand. During testing, we clearly can see this effect as most of the malfunctions occur early in a race.

The second parameter sets the interval that the program waits before checking for the next malfunction. The default value is 540 ticks which at 36 ticks per second equates to 15 seconds. In a two hour race, there would be about 480 separate malfunction checks. This value is randomized so the precise interval between failure checks varies. Apparently at every interval, each car is checked for malfunction.

The actual number of malfunctions in a race is not a straightforward calculation because the malfunction chance varies with the number of AI cars remaining and we're dealing with random intervals, but a simulation program can estimate it. I wrote a Visual Basic program that does so. With the default values in a 2 hour race, there should be

about four malfunctions. For Monza at 1 hr, 43 mins, there should be about 3.6 failures. However, my tests (shown later) indicate there are more malfunctions than this...around 8 per race...over twice as many.

The main point to remember about the gpl_ai.ini file is that is contains three main settings for controlling all of the AI cars...the malfunction distribution chances, the overall malfunction chance, and the malfunction check interval.

Before we get too carried away with testing, we need to know what specific indications does the gpl.exe program give of a car's malfunction. Fortunately when a car has a failure or problem, the reason for the malfunction is listed in the race results. I did a test where all of the driver.ini file malfunction chances were arbitrarily set to 20.00 and the gpl_ai.ini file malfunction chances were set to 0.00 except for the parameter being tested which was set abnormally high to 100.00. This test indicates which specific malfunctions occur within each problem/failure category.

Malfunction Display Test

Malfunction Tested	Malfunctions Obtained
Engine failure	header, engine, camshaft, ignition, valve, piston, clutch, gearbox
Engine problem	header, engine, camshaft, ignition, valve, piston, clutch
Fuel failure	retired?, suspension?--no fuel related failures
Fuel problem	suspension?--no fuel related problem
Fuel leak	fuel leak, no fuel, fire, fuel injection, retired?
Oil leak	oil leak, oil pump, oil pres, oil line, no fuel?, retired?
Coolant leak	coolant leak, retired?
Brake failure	brakes
Brake problem	brakes
Susp failure	suspension
Susp problem	suspension
Tire failure	tire, wheel, puncture
Tire problem	tire

My conclusions from this test are:

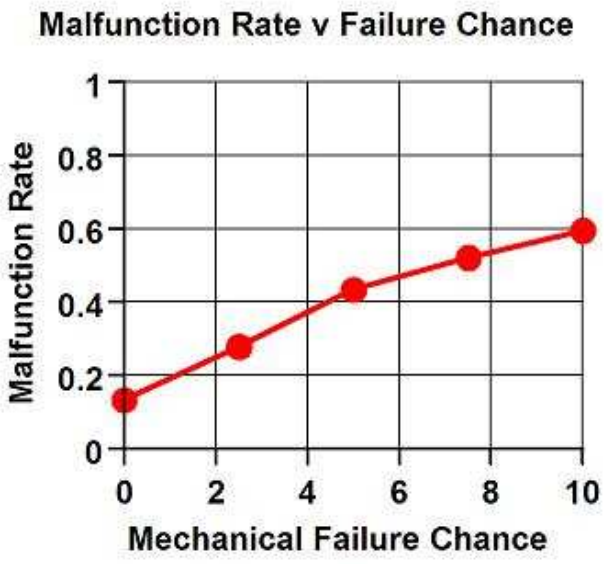
1. The gpl_ai.ini file chances affect the type of malfunction.
2. The gpl_ai.ini file chances affect the distribution of malfunctions.
2. There are few or no differences in the displayed malfunction between failures and problems.
3. Fuel failure and fuel problem don't always produce fuel malfunctions unless the "retired" malfunction is actually a misnamed fuel malfunction...I think this is so.
4. Fuel problem and coolant leak chances produce very few malfunctions of any type...certainly not as many as the other failure/problem parameters. In fact, a coolant leak is extremely rare.
5. There are some random effects as suspension malfunctions occur during fuel failure and fuel problem tests and no fuel occurs with the oil leak test even though its chances are set to 0.00.

Next, let's look at how the gpl_ai.ini file mechanical_failure_chance setting affects the number of malfunctions during a race. For this test, I set all gpl_ai.ini file settings to their defaults except for changing the mechanical_failure_chance setting. All of the driver.ini file chances were set to their defaults. This test was done with 20 trials/races at Monza.

Mechanical_Failure_Chance Setting

Mech_Failure Setting	Number of Malfunctions w/o accidents	w/ accidents	Rate of Malfunctions w/o accidents	w/ accidents
0	0.30	2.55	.016	.134
2.5	3.55	5.30	.187	.279
5 (default)	5.70	8.30	.300	.437
7.5	7.45	9.90	.392	.521
10	7.75	10.25	.461	.595

Here is a graph of this data:



My conclusions are:

1. The gpl_ai.ini files mechanical_failure_chance setting affects the overall number of malfunctions.
2. The number of malfunctions is approximately a linear result of the chance setting. The higher the setting, the more malfunctions occur. I believe if you were to continue the test with settings above 10, you'd see that the curve starts to level off.
3. Accidents occur at all settings even with a zero setting. These accidents actually may be disguised malfunctions.
4. The accident rate is fairly constant at 13% per race regardless of setting.
6. Remember that Monza is about a 1 hr 45 min race so longer races should have more malfunctions.

I did a regression analysis which shows the effect of the mechanical_failure_chance setting on the malfunction rate when accidents are included. The regression formula is:

Malfunction Rate = .160 + .047X

where X = Mechanical_failure_chance setting
R Squared =.973

This very high R Squared value indicates that there is a strong linear relationship between mechanical_failure_chance settings less than or equal to 10 and the overall number of malfunctions during a race.

A slightly better regression formula for the same data is:

Malfunction Rate = .130 + .071X - .002X^2

where X = Mechanical_failure_chance setting
R Squared =.977

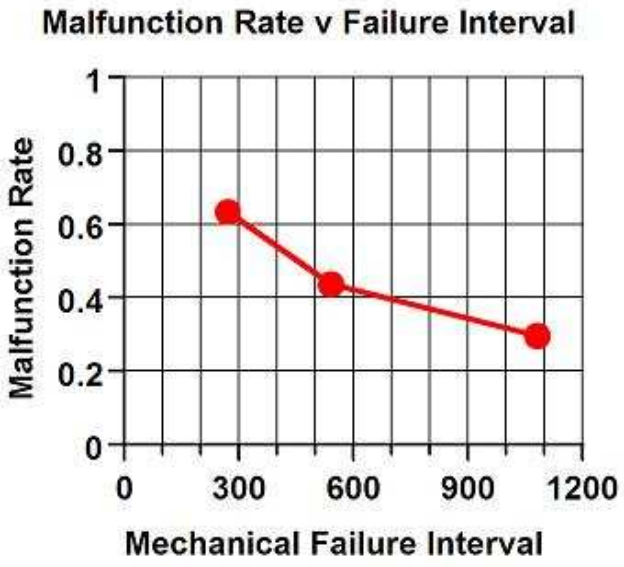
This formula clearly shows the 13% accident rate and the level off of the curve as the setting exceeds 10.

Next, I did a test where all the gpl_ai.ini file settings were set to defaults except the mechanical_ failure_interval setting was changed. All of the driver.ini files chances were set to defaults. This test checks the effect of mechanical_failure_interval changes on the number of malfunctions. The test was done with 20 trials/races at Monza.

Mechanical_Failure_Interval Setting

Mech_Interval Setting	Number of Malfunctions w/o accidents	w/ accidents	Rate of Malfunctions w/o accidents	w/ accidents
270	9.10	12.05	.479	.634
540 (default)	5.70	8.30	.300	.437
1080	3.45	5.65	.182	.297

Here is a graph of this data:



Conclusions:

1. The gpl_ai.ini file mechancial_failure_interval setting affects the number of malfunctions.

2. The number of malfunctions is a nonlinear result of the interval setting. The higher the setting, the fewer malfunctions occur.
3. Accidents occur at all settings which may be disguised malfunctions.
4. The accident rate is fairly constant at about 13% regardless of setting.
5. Remember that Monza is about a 1 hr 45 min race so longer races will have more malfunctions.

I did a regression analysis which measures the effect of the mechanical_failure_interval setting on the malfunction rate when accidents are included. The regression formula is:

Malfunction Rate = 1.985 - .243LnX

where X = Mechanical_Failure_Interval Setting
Ln = Natural Logarithm
R Squared = .991

This extremely high R Squared value indicates that the regression curve is a very good approximation of the effect of the mechanical_failure_interval setting.

What have we learned so far? The gpl_ai.ini file failure and problem chance settings apparently set the distribution of malfunctions for all cars. Also, we can adjust the overall number of malfunctions during a race by adjusting either the mechanical_failure_chance or mechanical_failure_interval settings. Thirdly, during any race there will be accidents that are probably disguised malfunctions. The accident rate is fairly constant at 13% or about two cars per race.

So that's it for the gpl_ai.ini file settings. Let's now take a look at the driver.ini file settings that affect car reliability. Hopefully, we can control the individual car reliability to the extent that they are faithful to the historical record.

DRIVER.INI FILE TESTS

Here is an example of the driver.ini file settings that affect reliability. They are taken directly from the original 1967 settings for Jimmy Clark's Lotus 49:

```
chance_brake_failure = 35.000000            ; chance_brake_failure
chance_brake_problem = 35.000000            ; chance_brake_problem
chance_coolant_leak = 35.000000             ; chance_coolant_leak
chance_engine_failure = 50.000000           ; chance_engine_failure
chance_engine_problem = 50.000000           ; chance_engine_problem
chance_fuel_leak = 35.000000                ; chance_fuel_leak
chance_fuel_system_failure = 35.000000      ; chance_fuel_system_failure
chance_fuel_system_problem = 35.000000      ; chance_fuel_system_problem
chance_oil_leak = 35.000000                 ; chance_oil_leak
chance_suspension_failure = 50.000000       ; chance_suspension_failure
chance_suspension_problem = 50.000000       ; chance_suspension_problem
chance_tire_failure = 40.000000             ; chance_tire_failure
chance_tire_problem = 35.000000             ; chance_tire_problem
```

Note that we have no idea of what unit the GPL designers used for each chance setting. Are these percentages? Or perhaps they are the number of malfunctions per 1,000 races? If you add up the individual settings, they total over 500. For now, we'll just have to let GPL keep its secret.

First, let's see how the driver.ini file and gpl_ai.ini file settings interact to affect the AI car's reliability. I did a test where all gpl_ai.ini file parameters were set to defaults including the mechanical_ failure_chance setting. I then varied the driver.ini file chances to check for their effects on the malfunction distribution...not number. All of the driver.ini file chances were set to the same value.

Driver.ini File Chance Settings Effect On Malfunction Distribution In Percent

Malfunction Category	Gpl_ai.ini Chances	Driver.ini Chances 0	5	10	20	30	40	50
Engine related	39	53	52	59	61	55	59	58
Suspension related	25	12	36	21	29	19	20	27
Tire related	7	6	3	3	4	4	7	2
Fuel related	14	30	3	13	4	11	7	11
Oil related	10	0	6	0	2	9	4	0
Brake related	3	0	0	5	0	2	4	2
Coolant related	2	0	0	0	0	0	0	0

My conclusions are:

1. The driver.ini chances have no effect on malfunction distribution.
2. The engine malfunction percentages are always too high in relation to the gpl_ai.ini file setting.

These are interesting results. Basically, we cannot control an individual car's malfunction distribution. Apparently, the distribution among the various malfunction categories is set by the gpl_ai.ini file settings (or gpl.exe program) which affect all cars the same. This is a disappointing result as I hoped that we could individually set each car's percentage of oil failures, suspension problems, etc.

Also, even the gpl_ai.ini file settings don't appear to control fully the malfunction distribution as the engine related malfunctions are always much higher than the gpl_ai.ini file settings for engine failures and problems. Apparently, the gpl.exe program has a big say in the distribution of malfunctions.

So what can we control if anything?

For the final test, I set the gpl_ai.ini file settings to their defaults and then varied the driver.ini file chances to check for their effect on malfunction number. Each chance was set to the same value. The test was done with 40 races for each setting at Monza. Altogether, it took over 300 races to produce the following table.

Driver.ini Settings Effect On Number of Malfunctions

Driver.ini Settings	Number Of Malfunctions w/o accidents	w/ accidents	Rate Of Malfunctions w/o accidents	w/ accidents
00.0	2.08	4.25	.109	.224
02.5	2.83	5.18	.149	.272
05.0	3.53	5.40	.186	.284
07.5	4.55	6.70	.240	.353
10.0	4.47	7.00	.236	.368
20.0	5.87	8.08	.309	.425
30.0	5.33	7.75	.280	.408
40.0	5.52	8.07	.291	.425

Here is a graph of this data:

My conclusion from this test are:

1. The driver.ini file chances have an effect when set less than or equal to 20. We can use this to set the number of malfunctions for each car.
2. The driver.ini file chances do not affect the number of malfunctions if set greater than 20.

In addition to these conclusions, here are a few comments on this test. Remember again that accidents may be disguised malfunctions. Also, note how random the malfunction rates are. With only 40 tests/races for each data point, the malfunction rates appear to hop all over the place. Yet, I believe a trend stands out. As the driver.ini chance settings are increased from zero, the total number of malfunctions (with or without accidents) increases and then appears to level off about 28% and 42% respectively. This effect may be only be true with the default gpl_ai.ini file chance settings. In other words, if we use a higher mechanical_failure_chance setting, then driver.ini file chance settings above 20 may well show some differences. On the low side, we might achieve lower malfunction rates down to about 11% as indicated by our previous test of the mechanical_failure_chance setting.

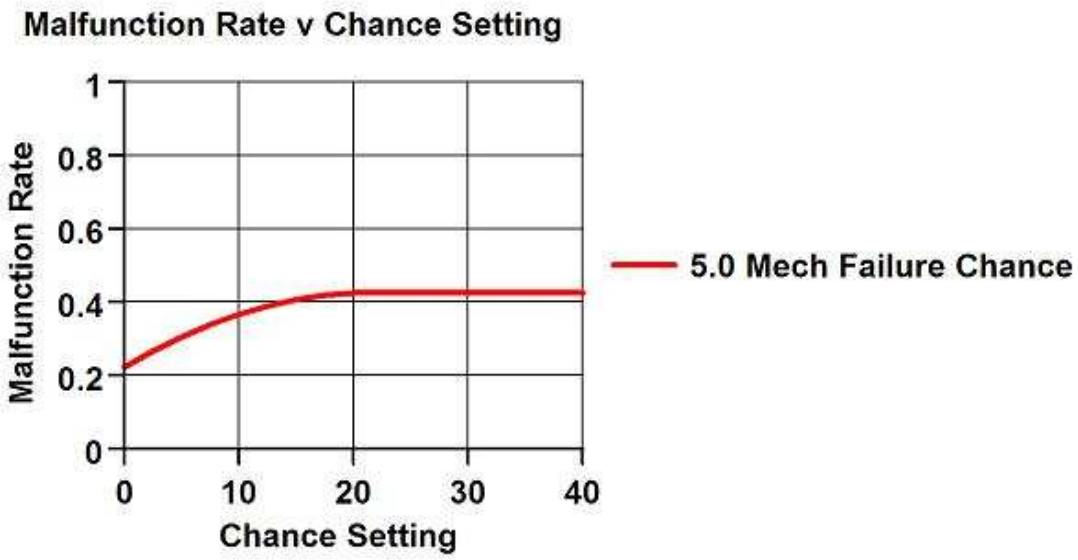
I developed a regression formula for chance settings from 0 to 20 using the malfunction rate with accidents. Here is the formula:

Malfunction Rate = .222+.019X-.0004X^2

where X = Driver.ini chance setting (0 <= X <= 20)
R Squared =.975

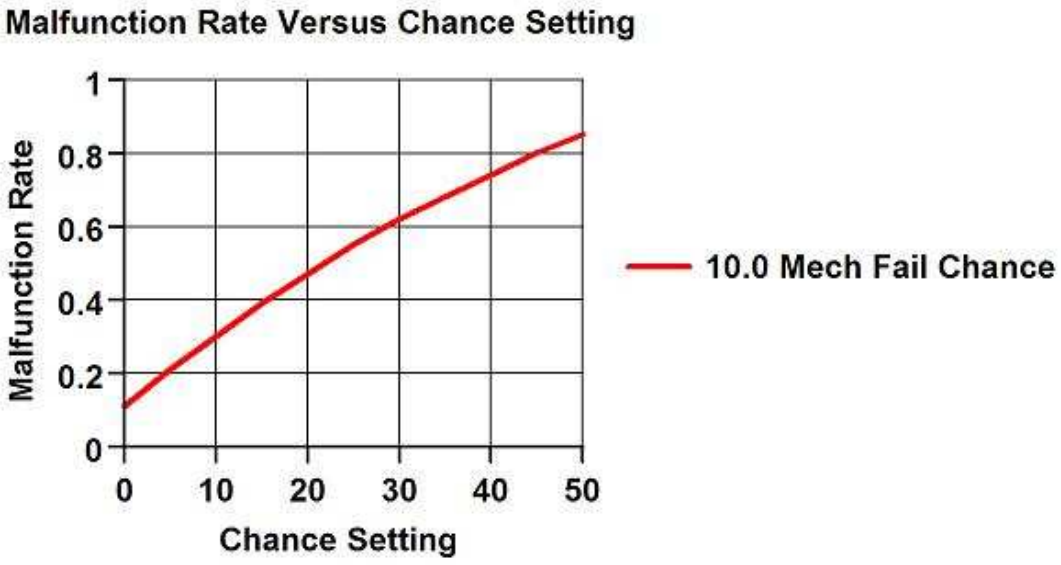
With an R Squared of .975, this formula is very accurate in describing the relationship between the driver.in file chance settings and the malfunction rate with settings in the range of 0 to 20.

Here is a graph of the regression curve that depicts how the driver.ini file settings affect individual car reliability.



We are still not out of the woods however. With a mechanical_failure_chance default setting of 5, we can only vary the malfunction rate from .22 to .42 for a range of only .20. This is clearly insufficient as we will soon see when we start looking at the historical record. Ideally, we would like to vary the malfunction rate from 0 to 1.00; however, this may prove to be impossible.

So let's up the mechanical_failure_chance setting to 10.0 to see what effect that has. Here is a graph showing the results of this test.



I derived a regression formula using the mechanical_failure_chance setting of 10.0. Here is the formula:

Malfunction Rate = 0.11065+.02022X-.00011X^2

Where X = Driver.ini Chance Setting
R Squared = .990

This incredibly high R Squared value shows that the formula is outstanding in describing the effect of the driver.ini file chance settings on the malfunction rate. As anticipated, using a higher mechanical_failure_chance setting allows the use of driver.ini chance settings up to 50 or more. Also, the range of adjustment is now higher...from about .11 to .85.

RELIABILITY SETTINGS TEST

So let's try a test of controlling the AI car reliability using our newfound knowledge. Here is an example using the 1969 historical record of the actual Formula One cars. The following table shows the "Did Not Finish" percentage for the seven cars modeled by GPL. The data was obtained from the F1 Gamers website.

1969 "Did Not Finish" Percentage

Brabham BT26	44
BRM P126	69
Lotus 63	88
McLaren M7	29
Ferrari 312	83
Lotus 49B	50
Matra MS80	22

Overall, fully 48% of the cars that started a Formula One race in 1969 did not finish! Also note the large disparity between the cars. The Matra MS80 was very reliable while the McLaren M7 was only slightly worse. However, the Ferrari 312 and Lotus 63 were horrible. Poor Chris Amon only finished one of the six races he entered in the Ferrari. There's not much chance of winning a driver's championship with that poor reliability record.

The Lotus 49B, as an example, had a malfunction rate of 50%. With the new regression formula, we obtain a drv69w.ini file chance setting of 22.0 for the Lotus 49B. We do the same for all cars. The following table shows the computed drv69w.ini file chance settings for each car:

1969 Drv69w.ini File Chance Settings
Mechanical_Failure_Chance = 10.0

Car	Setting
Brabham BT26	18.0
BRM P126	35.0
Lotus 63	54.0
McLaren M7	9.0
Ferrari 312	48.0
Lotus 49B	22.0
Matra MS80	5.0

Our desired overall malfunction rate is dependent on the number of each type of car and its associated malfunction

rate. So for our driver.ini file mix of 4 Brabhams, 3 BRMs, 1 Lotus 63, 3 McLarens, 1 Ferrari, 4 Lotus 49Bs, and 3 Matras, the desired malfunction rate is the weighted average of all 19 cars for 48%...by mere coincidence the same as the actual 1969 Did Not Finish Rate for all cars.

Remember that all our tests were done at the Monza track. A Grand Prix length race at Monza takes about 1 hour and 45 minutes to complete. This is usually the shortest race during a Formula One season. The other races typically run about 2 hours. Because the other races are longer, we should expect more malfunctions than desired at these longer tracks.

So how well do these new reliability settings work? The following table shows the results of a 20 race test at Monza.

1969 Season Malfunction Rate

Car	Desired %	Test %
Brabham BT26	44	40
BRM P126	69	70
Lotus 63	88	73
McLaren M7	29	27
Ferrari 312	83	80
Lotus 49B	50	48
Matra MS80	22	23

As you can see, our reliability settings are outstanding in controlling the AI reliability. It's not perfect nor would I expect it to be given the limited number of test races we used. The Lotus 63 has a slightly lower malfunction rate than desired, but the other cars are almost exact.

Now that we have found an effective method to control the AI reliability, let's use our new regression formula to derive settings for the original 1967 cars and the 1965 modification cars too. Here are tables that show the desired malfunction rate for each car and it's associated setting:

1965 Season Malfunction Rates And Driv65.ini File Chance Settings
Mechanical_Failure_Chance = 10.0

Car	Malf Rate	Setting
Brabham BT11	36	13
BRM P261	20	4
Cooper T77	47	20
Brabham BT7	50	22
Ferrari 512	22	5
Lotus 33	28	9
Honda RA272	50	22

1967 Season Malfunction Rates And Driver.ini File Chance Settings
Mechanical_Failure_Chance = 10.0

Car	Malf Rate	Setting
Brabham BT24	26	7
BRM P115	29	9
Cooper T81B	47	20
Eagle T1G	80	45
Ferrari 312	25	7
Lotus 49	59	28
Honda RA300	44	18

1969 Season Malfunction Rates And Drv69w.ini File Chance Settings
Mechanical_Failure_Chance = 10.0

Car	Malf Rate	Setting
Brabham BT26	44	18
BRM P126	69	35
Lotus 63	88	54
McLaren M7	29	9
Ferrari 312	83	48
Lotus 49B	50	22
Matra MS80	22	5

The driver.ini files that accompany this tutorial use these reliability settings. You should change the mechanical_failure_chance setting to 10.0 in each gpl_ai.ini file to achieve the correct malfunction rates.

SUMMARY

If you've lasted this far, you've learned a lot about how GPL uses the AI parameters in the driver.ini file and how changes to these parameters affect both qualifying and race performance. You also learned a method for controlling the speed of the entire AI field using npt_override so that you as the player can effectively compete and win races. Further, we have seen how to quantify a driver's real world historical performance. Also, we've seen two methods for adjusting the driver.ini file or gpl_ai.ini parameters so that the GPL AI perform closely to their real world counterparts. Finally, we've done some preliminary research into how GPL models AI car reliability and what we can do to control it.

The important point to remember is that there are very few parameters that actually affect the AI driver's raw speed. During a race, only hype and quickness have any effect. During qualification, only hype, quickness, and qualifying have an effect. Because we can adjust a driver's race speed easily with hype alone, quickness is not needed. Once we have set race speed with hype, then it is a simple matter to adjust qualifying to set the qualifying speed.

I hope you've found this tutorial to be educational and that it will generate more interest in how to better adjust the parameters to make the AI drivers more faithful to the real world. There remains a lot of work to be done. Just take a look at the multitude of parameters in the gpl_ai.ini file sometime. You could spend a lifetime testing changes to all those settings!

I've included driver.ini files with the new AI settings described in this tutorial. There are two 1967 files: one uses the first method where hype and qualifying settings are not modified to account for the actual pole winner's time. It uses an npt_override setting of .960 to achieve the historical time. The second file uses the second method where hype and qualifying are adjusted to account for the actual pole winner's time. It uses an npt_override of 1.00. There is only one file each for the 1965 and 1969 drivers....these files use an npt_override setting of 1.00 to achieve the actual pole winner's time.

In place of the original Papyrus drivers, my 1967 files substitute Stewart, Anderson, Ligier, and Spence. Bandini' s settings are an estimate based on his performances in 1964, 1965, and 1966 as he qualified then failed to finish at his ill-fated Monaco race...the only race he entered in 1967.

Mario Andretti's settings for 1969 are based on his race performances in the years 1970, 1971, and 1972 as he failed to finish any of the three races he entered in 1969.

You should adjust the mechanical_failure_chance setting to 10.0 in each gpl_ai.ini file to achieve the correct malfunction rate for each car.

I hope the new files make your GPL racing experience more enjoyable. Remember, you can adjust the npt_override setting in the gpl_ai.ini file to speed up or slow down the entire AI field as necessary so that you can effectively compete and have a better chance of winning. Further, you can adjust the mechanical_failure_chance setting to control the overall number of AI car malfunctions.

Feel free report any problems or comments to me at (RemoveThisSpamPreventer)Lee200@earthlink.net.

APPENDIX 1
1967 AI #1 driver.ini FILE SETTINGS
Npt_Override = .960

Driver	Aggr	Alert	Exp	Hype	Qual	Quick	Smoot
Brabham	1.020	1.030	1.050	.993	.997	1.000	1.000
Stewart	1.020	1.030	1.000	1.004	.844	1.000	1.000
Rindt	1.020	.990	.980	.992	.878	1.000	1.000
Gurney	1.025	1.020	1.020	.993	.895	1.000	1.000
Bandini	1.030	1.020	1.000	.961	1.250	1.000	1.000
Clark	1.030	1.030	1.010	.990	1.027	1.000	1.000
Surtees	1.020	1.020	1.020	.995	.901	1.000	1.000
Hulme	1.020	1.020	1.000	.999	.881	1.000	1.000
Spence	1.010	1.010	.993	.971	.893	1.000	1.000
Rodriquez	1.015	1.000	.990	.957	1.250	1.000	1.000
McLaren	1.020	1.020	1.040	.953	1.086	1.000	1.000
Amon	1.015	1.020	1.000	.969	1.250	1.000	1.000
Hill	1.020	1.030	1.030	.984	.925	1.000	1.000
Ligier	.990	1.000	.970	.917	.786	1.000	1.000
Irwin	1.000	1.000	.970	.950	.991	1.000	1.000
Bonnier	.990	1.030	1.040	.945	.899	1.000	1.000
Parkes	.990	1.000	.970	.973	.869	1.000	1.000
Anderson	.980	1.000	.993	.942	.848	1.000	1.000
Siffert	.980	1.010	1.000	.951	1.012	1.000	1.000
Beltoise	.980	.980	.960	.963	.899	1.000	1.000
Ginther	1.000	1.020	1.030	.950	1.250	1.000	1.000
Ickx	1.000	.980	.970	.984	.930	1.000	1.000
Scarfiotti	1.010	1.000	.990	.979	.849	1.000	1.000

APPENDIX 2
1967 AI #2 driver.ini FILE SETTINGS
Npt_Override = 1.000

Driver	Aggr	Alert	Exp	Hype	Qual	Quick	Smoot
Brabham	1.020	1.030	1.050	1.036	1.004	1.000	1.000
Stewart	1.020	1.030	1.000	1.048	.859	1.000	1.000
Rindt	1.020	.990	.980	1.035	.893	1.000	1.000
Gurney	1.025	1.020	1.020	1.037	.919	1.000	1.000
Bandini	1.030	1.020	1.000	1.000	1.250	1.000	1.000
Clark	1.030	1.030	1.010	1.033	1.022	1.000	1.000
Surtees	1.020	1.020	1.020	1.039	.931	1.000	1.000
Hulme	1.020	1.020	1.000	1.042	.896	1.000	1.000
Spence	1.010	1.010	.993	1.012	.911	1.000	1.000
Rodriquez	1.015	1.000	.990	.996	1.173	1.000	1.000
McLaren	1.020	1.020	1.040	.992	1.032	1.000	1.000
Amon	1.015	1.020	1.000	1.010	1.078	1.000	1.000
Hill	1.020	1.030	1.030	1.026	.948	1.000	1.000
Ligier	.990	1.000	.970	.952	.818	1.000	1.000
Irwin	1.000	1.000	.970	.989	.988	1.000	1.000
Bonnier	.990	1.030	1.040	.983	.918	1.000	1.000
Parkes	.990	1.000	.970	1.013	.888	1.000	1.000
Anderson	.980	1.000	.993	.979	.879	1.000	1.000
Siffert	.980	1.010	1.000	.989	1.002	1.000	1.000
Beltoise	.980	.980	.960	.995	.915	1.000	1.000
Ginther	1.000	1.020	1.030	.981	1.250	1.000	1.000
Ickx	1.000	.980	.970	1.018	.947	1.000	1.000
Scarfiotti	1.010	1.000	.990	1.011	.869	1.000	1.000

APPENDIX 3
1965 driv65.ini FILE SETTINGS
Npt_Override = 1.000

Driver	Aggr	Alert	Exp	Hype	Qual	Quick	Smoot
Clark	1.030	1.030	1.010	1.042	1.070	1.000	1.000
Hill	1.020	1.030	1.030	1.013	1.070	1.000	1.000
McLaren	1.020	1.020	1.040	1.012	1.250	1.000	1.000
Gurney	1.025	1.020	1.020	1.030	1.000	1.000	1.000
Ginther	1.000	1.020	1.030	.970	1.250	1.000	1.000
Bonnier	.990	1.030	1.040	1.001	1.320	1.000	1.000
Surtees	1.020	1.020	1.020	1.015	1.120	1.000	1.000
Stewart	1.010	1.010	.980	1.013	.980	1.000	1.000
Spence	1.010	1.010	.993	1.002	1.250	1.000	1.000
Brabham	1.020	1.030	1.020	1.006	1.250	1.000	1.000
Hulme	1.020	1.020	1.000	1.020	1.200	1.000	1.000
Bandini	1.020	1.020	1.000	.982	1.250	1.000	1.000
Rindt	1.020	.999	.998	.982	1.400	1.000	1.000
Bucknum	1.015	1.020	1.000	.944	1.400	1.000	1.000
Siffert	.980	1.010	1.000	.981	1.250	1.000	1.000
Attwood	.980	.980	1.000	.964	1.250	1.000	1.000
Anderson	1.010	1.000	.990	.913	1.400	1.000	1.000
Ireland	1.000	.980	1.000	.967	1.250	1.000	1.000
Gardner	1.015	1.000	1.000	.971	1.100	1.000	1.000

APPENDIX 4
1969 drv69w.ini FILE SETTINGS
Npt_Override = 1.000

Driver	Aggr	Alert	Exp	Hype	Qual	Quick	Smoot
Stewart	1.020	1.020	1.000	1.115	.925	1.000	1.000
Ickx	1.030	1.030	1.030	1.077	1.075	1.000	1.000
McLaren	1.010	1.020	1.040	1.086	.925	1.000	1.000
Rindt	1.040	1.020	1.000	1.104	1.150	1.000	1.000
Beltoise	1.000	1.000	.970	1.074	.950	1.000	1.000
Hulme	1.020	.990	.980	1.075	1.400	1.000	1.000
Hill	1.000	1.030	1.030	1.070	1.250	1.000	1.000
Courage	.990	1.000	.980	1.000	1.250	1.000	1.000
Siffert	1.020	1.010	1.000	1.063	1.250	1.000	1.000
Brabham	1.020	1.030	1.020	1.093	1.100	1.000	1.000
Surtees	1.020	1.020	1.020	1.077	1.400	1.000	1.000
Amon	1.015	1.020	1.020	1.105	1.000	1.000	1.000
Elford	.980	1.000	1.000	1.045	1.000	1.000	1.000
Rodriquez	1.015	1.000	.990	1.073	1.050	1.000	1.000
Servoz-	1.010	1.000	1.010	1.043	.900	1.000	1.000
Moser	.990	1.000	1.000	1.022	.750	1.000	1.000
Oliver	1.030	1.030	1.010	1.075	1.100	1.000	1.000
Lovely	.990	1.030	1.040	1.023	.880	1.000	1.000
Miles	.990	1.030	1.040	1.013	1.600	1.000	1.000
Andretti	1.010	1.030	1.000	1.091	1.150	1.000	1.000