Lecture on

Modelling and Implementing Slip Target Shaper for Traction Control in F1 Application

Ing. Ugo Pattacini

Thales Alenia Space Italia S.p.A.

(formerly Embedded SW Developer at Toyota Motorsport GmbH)

14 September 2007

Italian Institute of Technology

Table of Contents

- Introduction to Vehicle Dynamics and Traction Control
  - Tyre model and PIS controller
- Input Shaping Background
  - Posicast, Robustness, Pole-Zero Analysis
- Shaper Design and Implementation in dSPACE TargetLink
  - Model inspection and analysis of production code
- Embedding Production Code within Toyota F1 RT SW
  - TF1 HW/SW Architecture
- Jerez test reports on Slip Target Shaper
- Shaper’s automatic tuning through GA

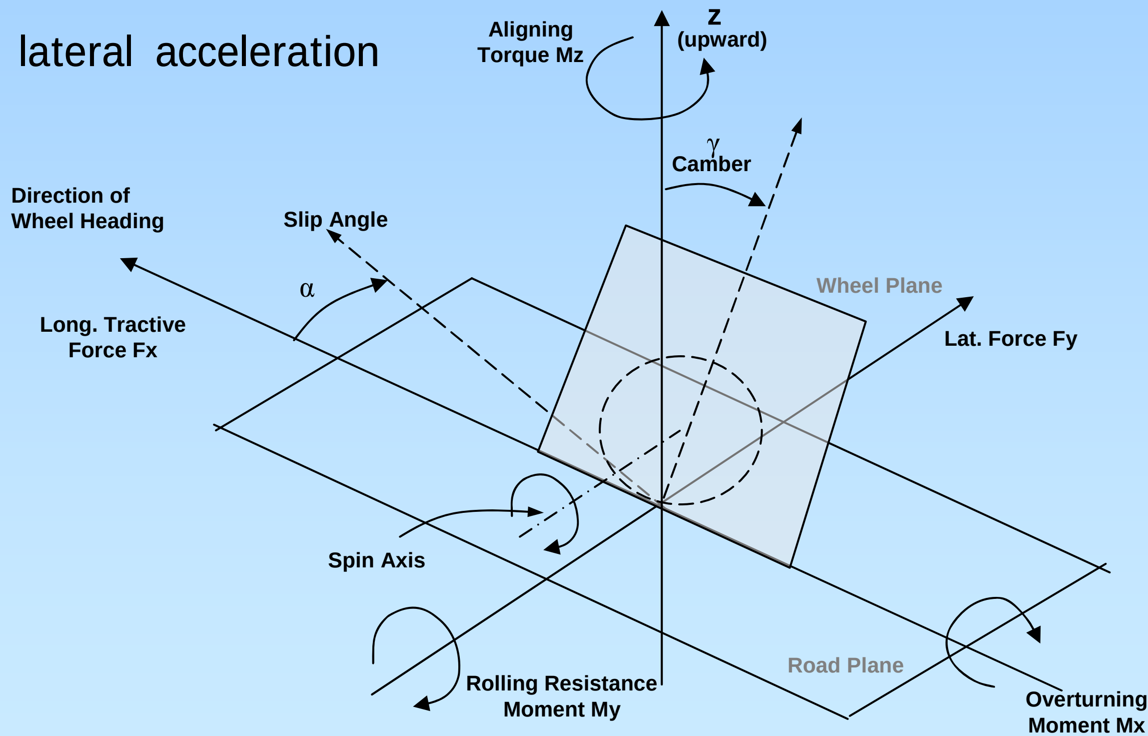
SLIP ANGLE (radiant)

The car Slip Angle is the angle between the direction of travel of the car and the longitudinal axis.

Slip angle becomes high with high lateral acceleration and low speed.

$$\alpha = \left(a_0 + a_1 A_Y + a_2 A_Y^2 + a_3 A_Y^3\right)\frac{\pi}{180}$$

Velocity (kph)	Polynomial Coeffiecients			
	a3	a2	a1	a0
64.8	0.0033	-0.0318	0.2021	-0.0519
86.4	0.0023	-0.0256	0.1827	-0.0484
108	0.0017	-0.0218	0.1732	-0.0546
129.6	0.0013	-0.0182	0.1627	-0.0577
151.2	0.0009	-0.0144	0.1481	-0.0554
172.8	0.0006	-0.0093	0.1223	-0.0425
194.4	0.0005	-0.0091	0.1206	-0.0511
216	0.0005	-0.0097	0.1231	-0.069
237.6	0.0005	-0.0108	0.1293	-0.0917
259.2	0.0002	-0.0046	0.0846	-0.0428
280.8	0.0002	-0.0066	0.1043	-0.0852
302.4	0.0002	-0.0065	0.1057	-0.0986



Linear interpolation on a_i allows the calculation of α for all velocities.

VERTICAL FORCES on the WHEELS (N)

Vertical force on a wheel is a combination of the force measured by the pushrod sensor (which are relative sensors, i.e. they see a deviation from the static load state of the car) and the wheel specific component of the dynamic mass of the car (dependent on the longitudinal and lateral acceleration and the characteristics of the suspension set-up).

$$F_{ZRR} = \frac{F_{RODRR}}{RP_{MR}} + M_{ST} * 9.81 * \left[\left(1 - \frac{M_{ST\_B}}{100} \right) \left(\frac{1}{2} + RK_{YZ} A_Y \right) - \left(\frac{1}{2} RK_{XZ\_E} A_X \right) \right]$$

$$if\ A_X > 0$$

F_{ZRR}	= Vertical load on the rear right wheel
F_{RODRR}	= Rear right push rod force (variable, available in ECU)
RP_{MR}	= Rear push to wheel force ratio (constant, available in ECU)
M_{ST}	= Static mass of car (constant, available in ECU)
M_{ST_B}	= Front to rear static mass balance (constant, available in ECU)
A_Y	= Lateral acceleration (variable, available in ECU)
A_X	= Longitudinal acceleration (variable, available in ECU)
RK_{YZ}	= Rear anti-roll coefficient (constant, available in ECU)
RKXZ_E	= Rear anti squat coefficient (constant, available in ECU)

SLIP RATIO (%)

Definition of longitudinal Slip Ratio S_X

$$S_X = \left(\frac{\dot{\theta}_w R_e}{\dot{x} * \cos(\alpha)} - 1 \right) * 100\% \quad \Rightarrow \quad \boxed{S_X = \left(\frac{V_w}{V_{car}} - 1 \right) * 100\%}$$

$\dot{\theta}_w$ = angular wheel speed
R_e = effective rolling radius
$\dot{x}$ = speed of axle relative to road
α = slip angle

As the slip angle is small (less than 5°), it has a negligible effect and can be ignored.

LONGITUDINAL TYRE MODEL

The effectiveness of the tyre model will depend upon how well it matches with the actual tyre performance. This is heavily reliant upon the constants used in the model:

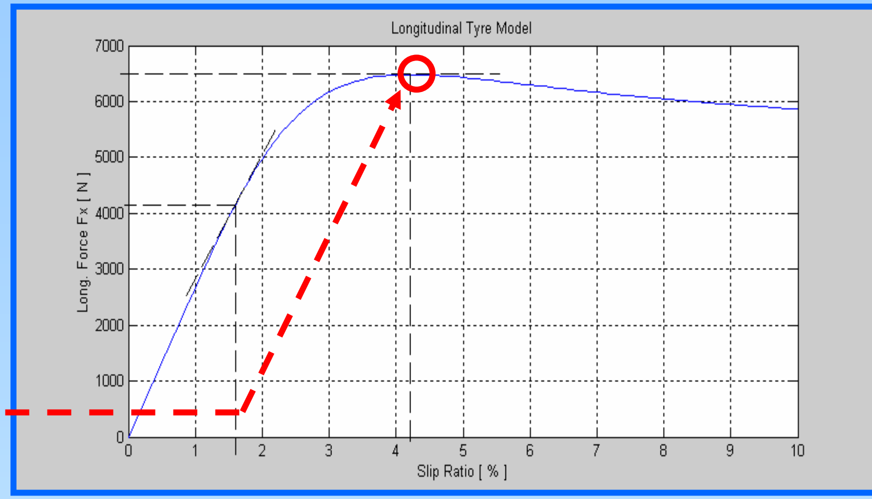
- 17 fitting constants
- 7 tuning constants
- 1 scaling constant

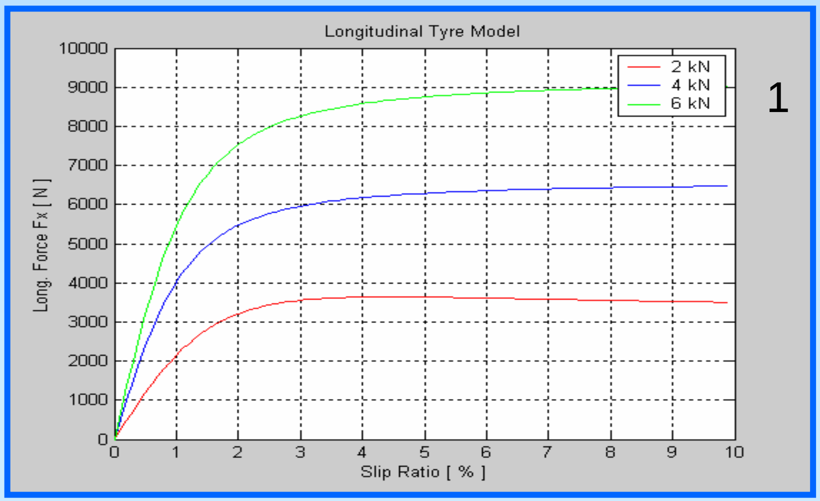
The fitting constants and scaling constant are chosen by tyre supplier to match test data and the tuning constant can be used to adapt the model to different track/weather conditions.

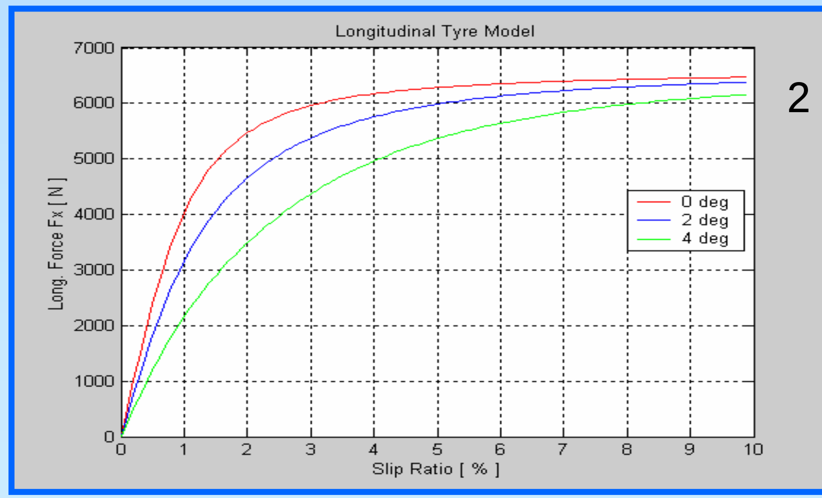
The output of the model is the longitudinal force produced by the tyre, which is a function of slip ratio, slip angle and vertical force.

$$\boxed{Fx = f\left(Sx, \alpha, Fz\right)}$$

The aim is to control the slip ratio in order to achieve the maximum torque allowed by the car and the grip conditions.





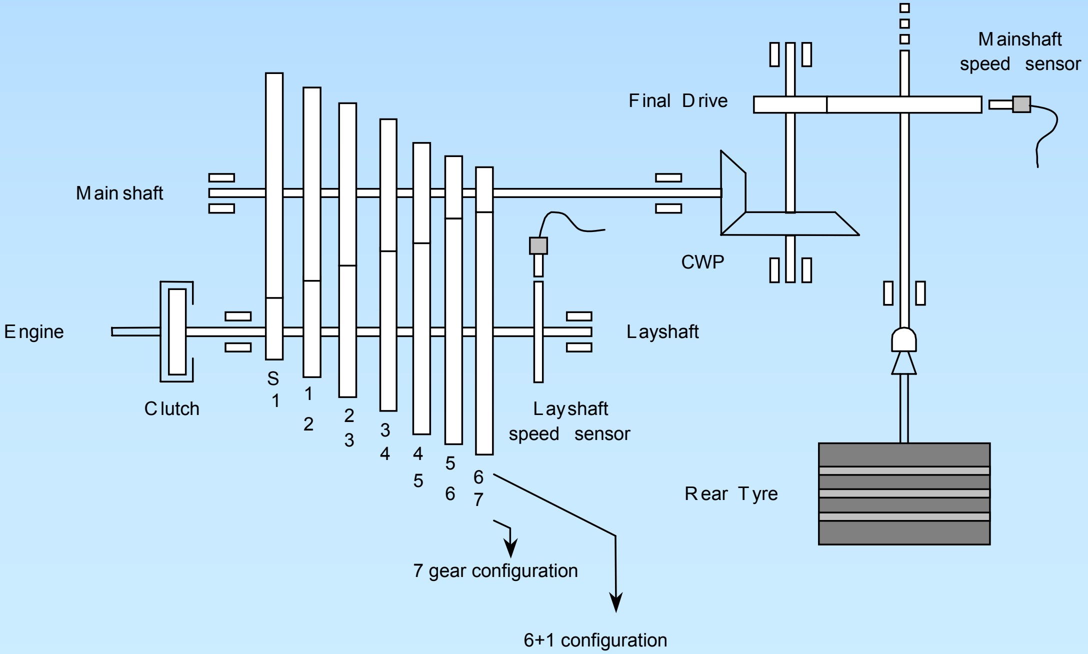


Longitudinal tyre force vs. slip ratio, for varying vertical load (1) and slip angle (2).

TRACTION CONTROL

- Traction Control tries to limit the torque transmitted to the road to an optimal value giving the best acceleration. It gets enabled typically during corner exit.
- The torque based traction control strategy sends to the torque control module a torque demand (made of a predictive term and a reactive term) and a torque reserve factor (to limit the amount of throttle reduction).
- The predictive term is based on the max transmittable torque from the tyres to the road (=f(α, F_Z)) and the power train inertial torque.
- The reactive term is the output of a slip ratio controller trying to hold a slip ratio target = f(V_{CAR}, $|A_Y|$).

From the torque transmitted to the road (Plant output) to the Engine Torque (the controlled variable)



PREDICTIVE TERM (feed-forward)

The predictive torque demand is made of the average of the optimum torque at the wheels (**T_{opt}**) related to the engine through the drive train ratios. The predictive torque is limited to the driver demand to allow better alignment to the real grip level even for bad μ settings.

$$F_{Xopt} = \mu(\alpha) F_Z$$

$$T_{RW} = \left(F_{Xopt} + F_{RollResR}\right) R_{RollR}$$

$$T_{opt} = \frac{\left(2T_{RW} + I_{PT}\right) N_{GR} N_{fd} N_{CWP}}{\eta_{PT}}$$

$$\text{with} \quad I_{PT} = \left(\frac{I_{GB}}{N_{GR}^2 N_{fd}^2 N_{CWP}^2} \eta_{PT} + I_{RW} \right) \frac{d\omega_{RW}}{dt}$$

$$\text{and} \quad \frac{d\omega_{RW}}{dt} = \left(1 + \frac{S_{Xopt}}{100} \right) \frac{A_X}{R_{RollR}}$$

The Friction coefficient μ(α) is calculated relying on baseline maps. It is the most critical parameter to be tuned, leading to several different strategies for self-adaptation (corrections on slip ratio error, Friction Ellipse)

REACTIVE TERM (feedback)

The reactive torque demand is the output of a PIS slip ratio controller, which tries to hold the slip ratio target close to the optimum value S_{Xopt} in order to ensure the best grip.

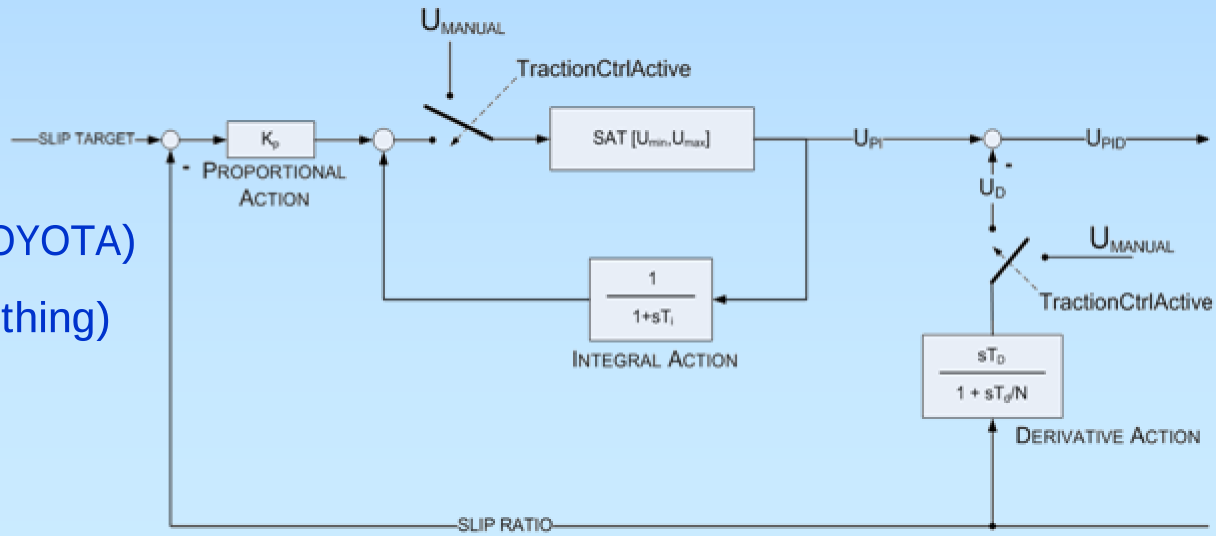
S_{Xopt} is a complex function of V_{CAR}, A_Y and slip angle:

$$\boxed{S_{Xopt} = \frac{S_0}{1 + (K + K_{\mathrm{mod}})\left|(A_Y \; or \; \alpha) A_{Ycorr}\right|^N} + C}$$

where S_0, K, N and C are functions of V_{CAR}

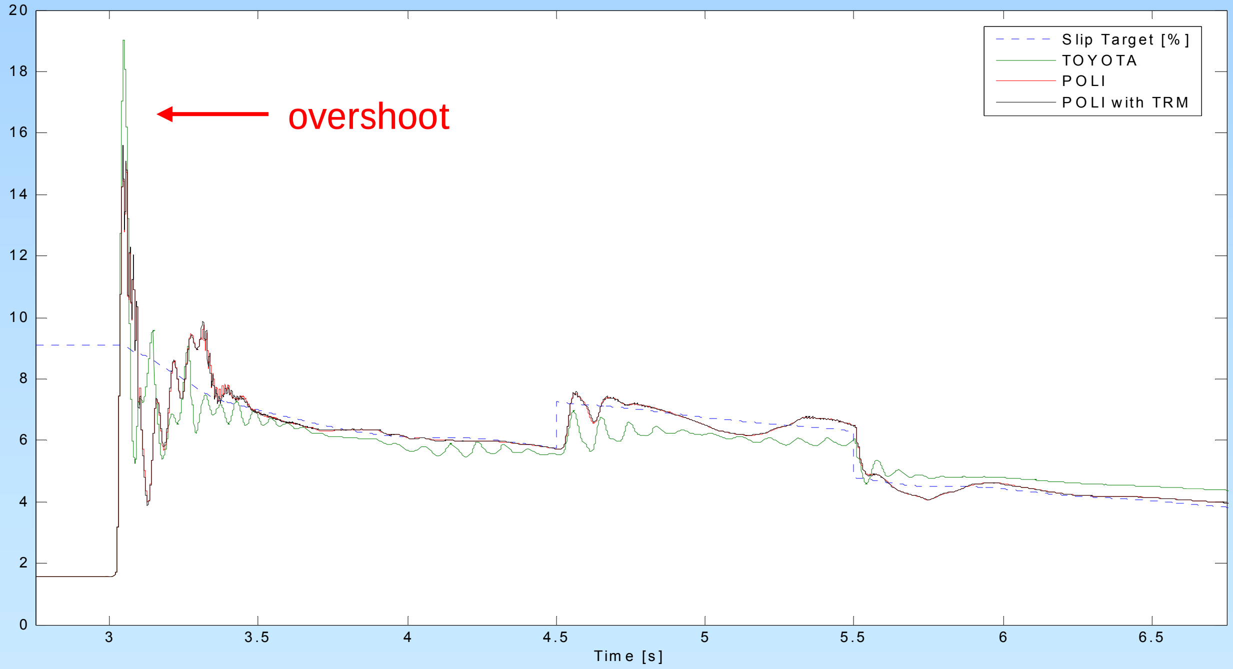
PIS implemented in F1 TC (TOYOTA)

(Proportional+Integral+Smoothing) in anti-windup configuration



PIS in action

Initial snap overshoot in the slip ratio brings about a reduction in the achievable speed during corner exit, causes also tyres degradation and a temporary loss of car stability for the driver.



Slip Target Shaper represents an attempt to cope this issue by shaping the slip command at the input stage of PIS controller

INPUT SHAPING

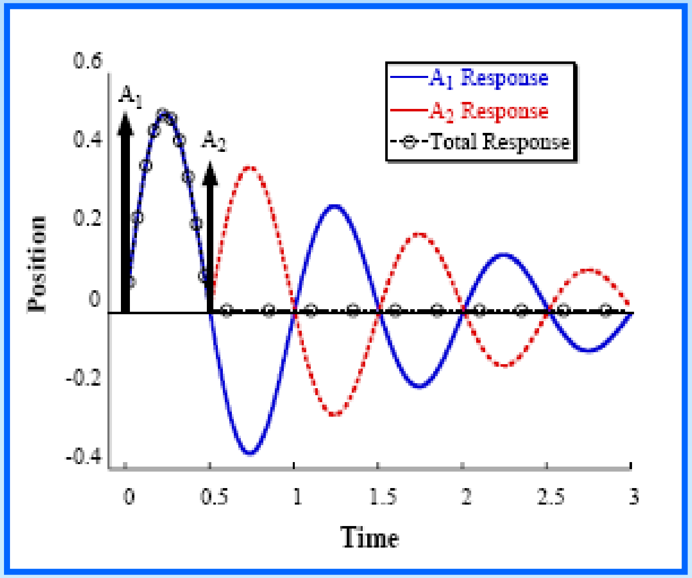
- The process of input shaping is a valuable method for controlling systems with performance limiting vibration.
- The first shaper proposed by Smith[1] was a simple technique to generate non-oscillatory response from a lightly-damped system subject to a step input. This was achieved by exciting two transient oscillations so as to result in constructive cancellation of the oscillations. (*Posicast*)
- Applications: chemical process, spacecrafts, flexible systems (cranes, robots, ...)

$$\text{Residual Vibration} \quad V(\omega,\zeta) = e^{-\zeta w t_n}\sqrt{C(\omega,\zeta)^2 + S(\omega,\zeta)^2}$$

where

$$C(\omega,\zeta) = \sum_{i=1}^{n} A_i e^{\zeta\omega t_i}\cos(\omega_d t_i), \quad S(\omega,\zeta) = \sum_{i=1}^{n} A_i e^{\zeta\omega t_i}\sin(\omega_d t_i)$$

$$\omega_d = \omega\sqrt{1-\zeta^2}$$

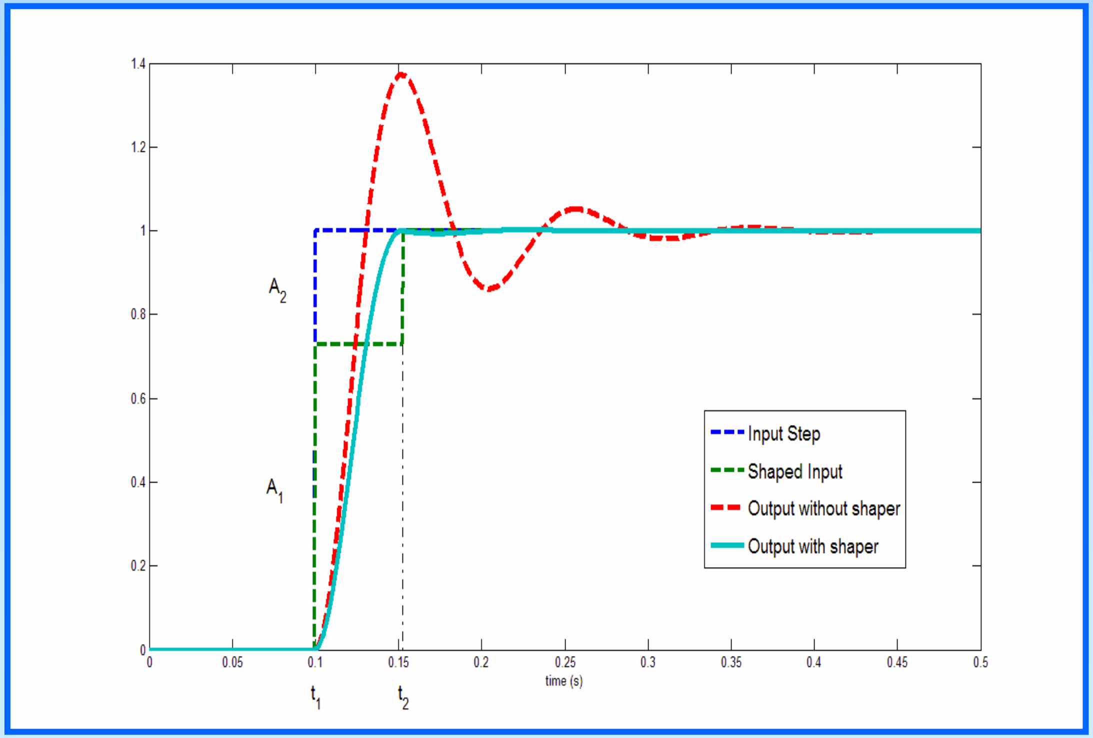


Posicast solution

Problem: find the input sequence with some constraints to obtain a bounded solution

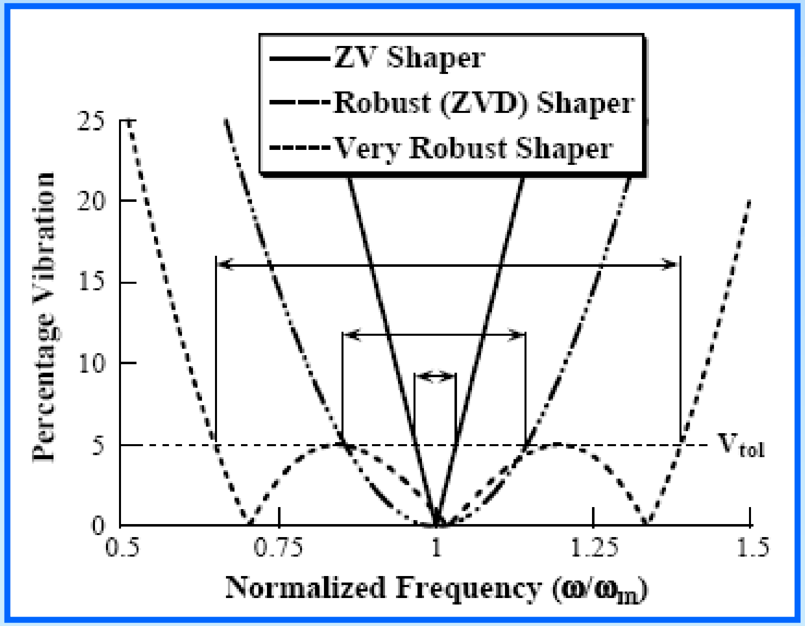
$$\begin{cases} V(\omega,\zeta)=0 \\ \sum A_i = 1 \\ A_i > 0,\ i = 1,2,\dots,n \end{cases} \quad \Rightarrow \quad \begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \frac{1}{1+K} & \frac{K}{1+K} \\ 0 & 0.5T_d \end{bmatrix},\ K = e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}},\ T_d = \frac{2\pi}{\omega_d}$$

- The impulse sequence is convolved with any desired command signal.
- If the impulse sequence causes no vibration, then the convolution product will also cause no vibration.



Robustness to Modelling Errors

- The amplitudes and time locations of the impulses depend on the system parameters (ω and ζ). If there are errors in these values (and there always are), then the impulse sequence will not result in zero vibration (ZV).
- This problem can be visualized by plotting a sensitivity curve that shows the amplitude of residual vibration as a function of the system parameters.
- Insertion of additional constraints to improve robustness: ZVD, EI.



1. ZVD (Zero Vibration Derivative)

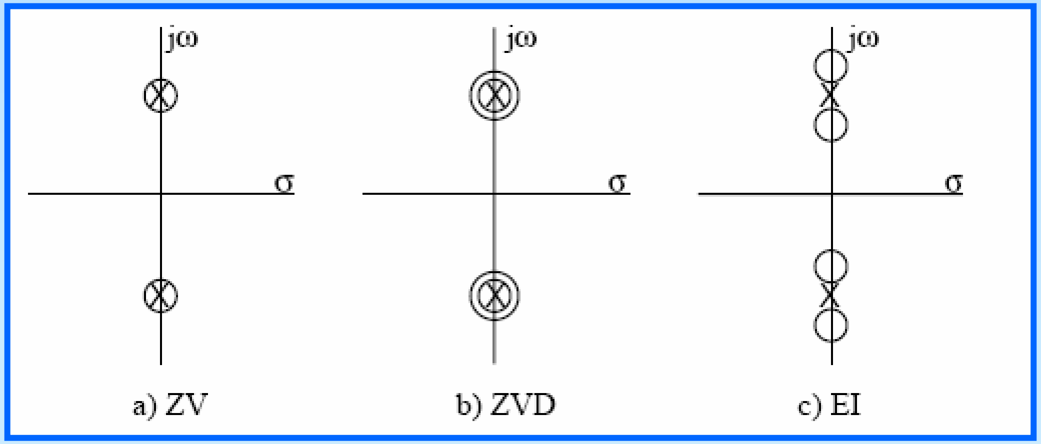
$$\frac{d}{d\omega}V(\omega,\zeta)=0$$

2. EI (Extra-Insensitive) with frequency sampling

$$V_{tol} > e^{-\zeta\omega_s t_n}\sqrt{C(\omega_s,\zeta)^2 + S(\omega_s,\zeta)^2},\;\; s=1,\ldots,m$$

Pole-Zeros Analysis

- A useful method for explaining the effectiveness of shaped command signals is to use pole-zero analysis to determine the location of the zeros and/or poles of the command shaping process.
- If a shaped command contains zeros near the poles of the system being controlled, then the vibration from those poles is attenuated by the zeros of the command.
- ZV shaper places a single zero over each of the flexible poles. If the poles deviate from their modelled locations, the attenuating effect of the zeros decreases.
- ZVD shaper places two zeros over each pole. When a pole deviates from its modelled location, the two zeros have a greater attenuating effect.
- EI shaper place zeros at nearby locations, taking into more account the uncertainties in the model.



Zero Locations of Typical Input Shapers

The zero separation of EI extends the attenuating effect to a larger area.

Adding more zeros makes the shaper more robust. We pay it with increased rising time.

Shaper Design through Pole-Zero Analysis

The shaper transfer function can be expressed as:

$$H(z) = \frac{C}{z^{2n}}(z - p_1)(z - p_1^*)\ldots(z - p_n)(z - p_n^*)$$

where p and p* represent a pair of complex poles and n is the number of flexible modes being targeted for elimination. The transfer function must have as many poles at z = 0 as zeros in the numerator so that the shaped command will be causal.

One drawback is that the shaper can contain large positive and negative valued impulses capable of saturating the actuators.

This difficulty can be avoided by using an *artificially coarse sampling rate*. This approach can be thought of as inserting zero-amplitude impulses at the true digital time step between finite amplitude impulses at some multiple of the sampling period.

Methods for selecting this surrogate sampling period are usually based on the constraint that all the impulses must have positive values. To demonstrate such a method, an example using the *six-step procedure* is reproduced here.

Shaper Design

Six-step Procedure[2]

Goal: Design a shaper for a system with modes at 1 Hz and 4 Hz and damping ratios at each mode of 0.1.

Step 1: Identify the unwanted system poles and cover them with zeros in the z-plane.

$$s_1, s_1^* = -\zeta_1\omega_1 \pm j\omega_1\sqrt{1-\zeta_1^2}$$

$$s_2, s_2^* = -\zeta_2\omega_2 \pm j\omega_2\sqrt{1-\zeta_2^2}$$

$z = e^{sT}$

$p1, p1^* = e^{-0.2\pi T \pm j1.99\pi T}$

$p2, p2^* = e^{-0.8\pi T \pm j7.96\pi T}$

Step 2: Insert additional zeros for robustness.
(For simplicity, it will be assumed that a single zero at each pole is adequate for this example)

Step 3: Construct the shaper transfer function.

$$H(z) = \frac{C}{z^4}(z-p_1)(z-p_1^*)(z-p_2)(z-p_2^*)$$

Step 4: Calculate the resulting impulse sequence.

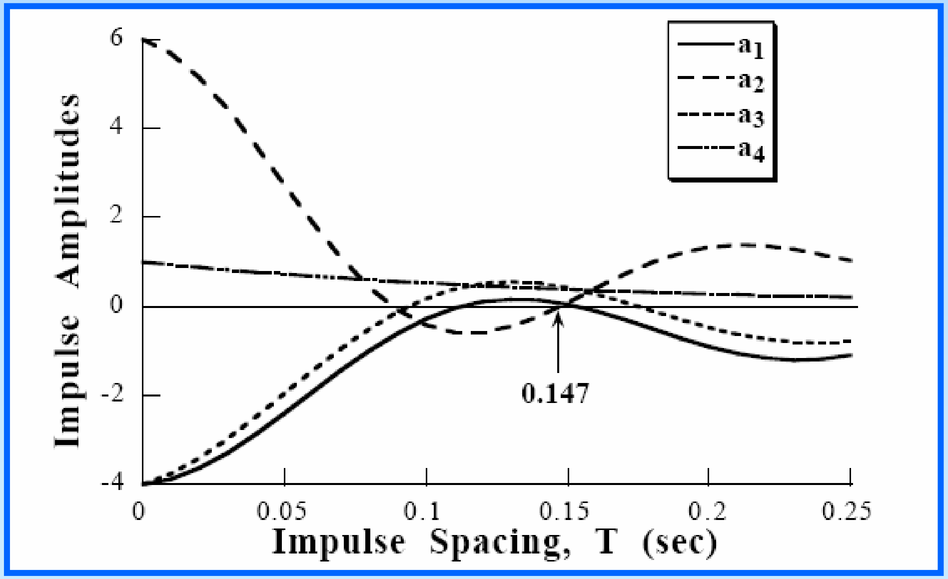
$$H(z) = C\frac{z^4 + a_1 z^3 + a_2 z^2 + a_3 z + a_4}{z^4} \Rightarrow h(t) = C\,[\,\delta(t) + a_1\,\delta(t-T) + a_2\,\delta(t-2T) + a_3\,\delta(t-3T) + a_4\,\delta(t-4T)]$$

where parameters a_i are function of sample time T.

Step 5: Plot the impulse amplitudes as a function of T and select a sequence that meets actuator constraints.

Step 6: Normalize amplitudes.
Typically C is set in order to scale the amplitudes to sum to unity.

$$h(t) = 0.516\,\delta(t) + 0.042\,\delta(t-0.147) + 0.237\,\delta(t-0.441) + 0.205\,\delta(t-0.588)$$



for T=0.147 all a_i are positive

Shaper Implementation

dSPACE TargetLink Environment

- Production code generation directly from MATLAB/Simulink/Stateflow.
- ANSI C code with the efficiency of handwritten code.
- Optimizations for individual processors.
- Built-in simulation and testing.

Key Benefits

- Connect function development and code generation easily.
- Converting graphical models directly into production code ensures perfect consistency between model and code at all times.
- TargetLink’s code generation is deterministic.
- Every step can be tested against the specification via the built-in simulation features.
- Time and costs reductions.

Real Time Workshop Embedded Coder is the MathWorks answer to dSPACE.
(It came available later on the market).

Shaper Implementation

Goal: implement the following Slip Target Shaper

$$h(t) = a_1\delta(t) + a_2\delta(t - T_1) + a_3\delta(t - T_2) + a_4\delta(t - T_3) + a_5\delta(t - T_4) + \ldots + a_{16}\delta(t - T_{15})$$

1. *f32_Traction_SlipShaperC* is a vector of 16 shaper’s coefficients as addressed six-step procedure. The sum of elements should be 1 in order to track the target perfectly.
2. *u32_Traction_SlipShaperT* is a vector of 15 shaper’s time references expressed in milliseconds. The sum of elements shall not exceed 512 ms, that is the internal buffer size.

- As the size of coefficients vector is 16, the maximum number of modes it is possible to work on is 7.

Triggering Conditions

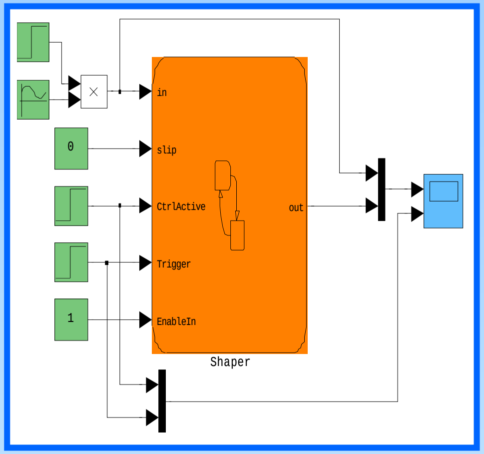
- When enabled Shaper always performs filtering. It gets armed and triggers a “shaping reset” only when the slip is greater than a specified threshold, that is, once triggered, Shaper restarts filtering the target as it would have begun from current value of slip. Therefore, the actual difference between target and slip is considered the big step to be filtered out.

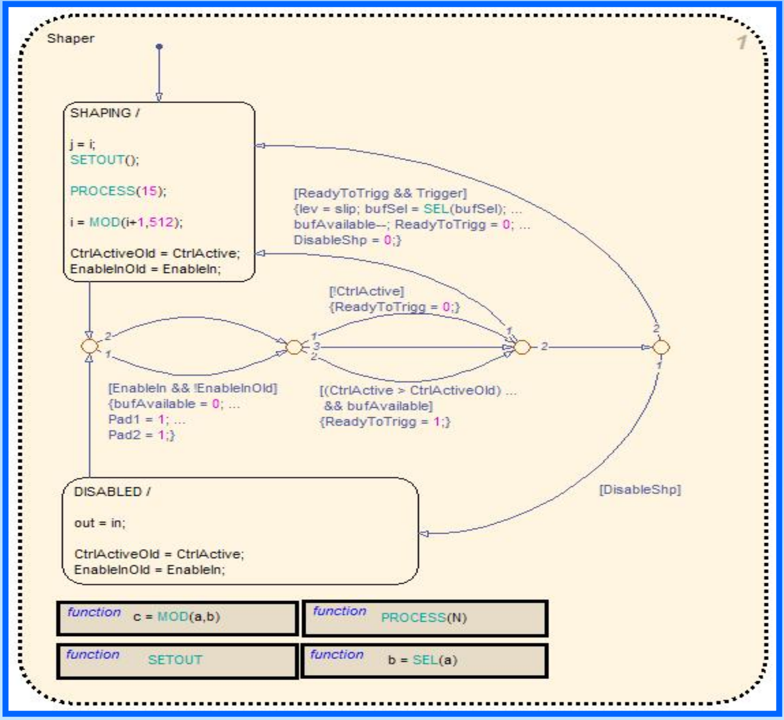
Disabling Conditions

- Shaper is disabled as soon as the shaped target gets closer to the unshaped form (threshold). This aims at avoiding introducing phase lag and offset error on target when Shaper has already accomplished its work.

Shaper Implementation

Browse Model





Traction()

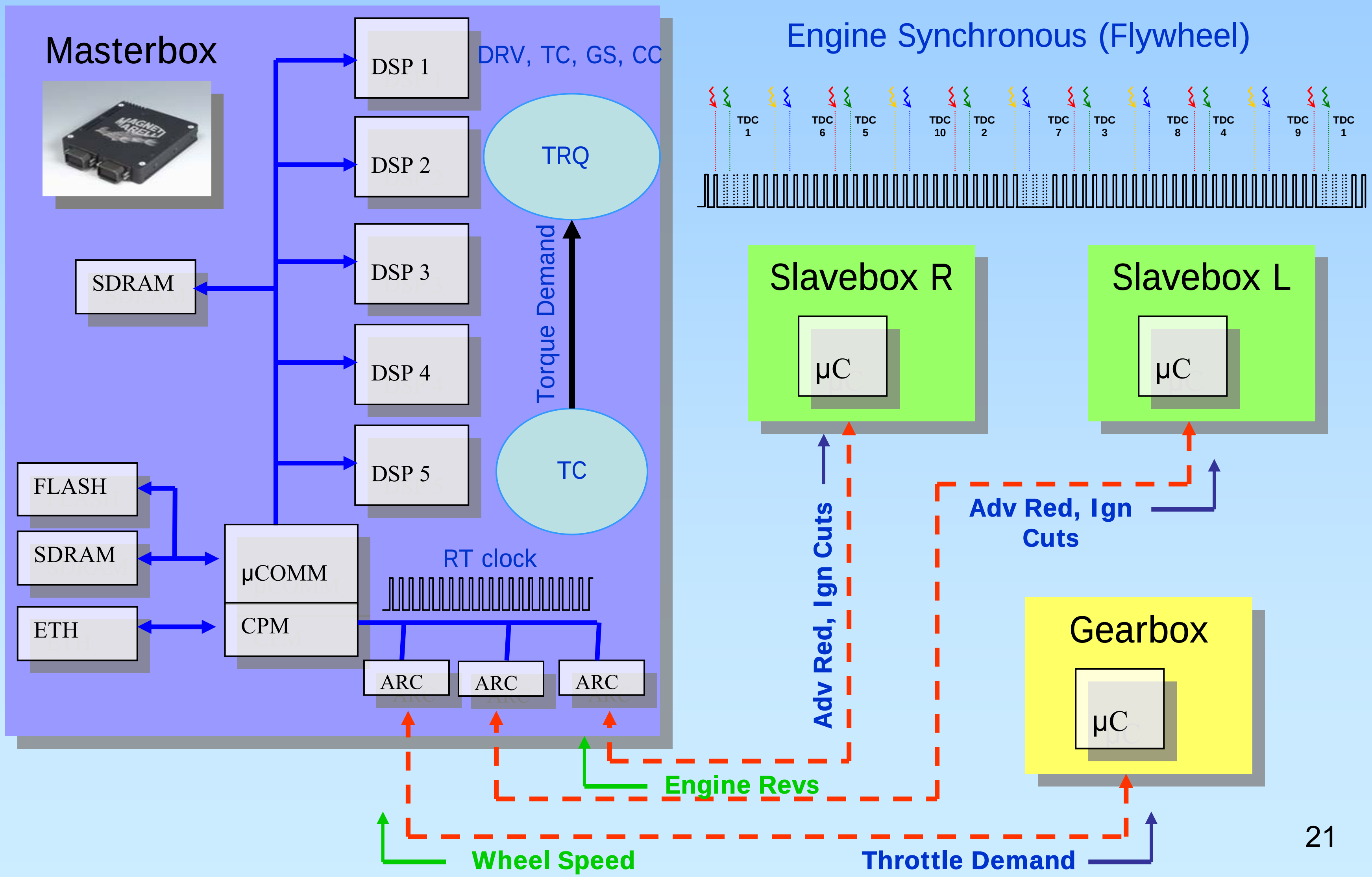
```
/* Traction/Traction/Slip Target Shaper/Shaper/Enable: Enab
   Traction/Traction/Slip Target Shaper/Shaper/Enable: Omit
if (Sc8_Relational_Operator) {
   /* update of inport for Traction/Traction/Slip Target Sh
   Cc11_Trigger = Sc32_Relational_Operator;

   /* Begin execution of chart Traction/Traction/Slip Targe
   if (SIBFS_Shaper_c.Cc12_Shaper) {
      /* Begin execution of state Traction/Traction/Slip Ta
      if (SIBFS_Shaper_c.Cc17_SHAPING) {
         /* Begin execution of state Traction/Traction/Slip
            APING */
         SIBFS_Shaper_c.Aux_Cc11_sflag1 = 2 /* 2. */;
         Cc11_Shaper_node_fcn1();

         /* End execution of state Traction/Traction/Slip T
            ING */
      }
      else {
         if (SIBFS_Shaper_c.Cc18_DISABLED) {
            /* Begin execution of state Traction/Traction/S
               .DISABLED */
            SIBFS_Shaper_c.Aux_Cc11_sflag1 = 1 /* 1. */;
            Cc11_Shaper_node_fcn1();
```

Browse C modules

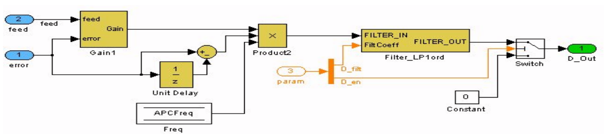
TF1 HW Architecture



TF1 RTOS Architecture

Custom RTOS developed in-house based on messaging

dSPACE TargetLink



C

Compile

DSP 5

```
void INT_1ms_slot0(void) {
     …
     Traction();
     …
     SendMsg(&msg_TS5_TS2_TrqDem);
}
```

DSP 2

```
void INT_1ms_slot1(void) {
     GetMsg(&msg_TS5_TS2_TrqDem);
     …
     Torque();
     …
}
```

Message Description File

```
#msg_TS5_TS2_TrqDem, STEP11_TS5, STEP11_TS2;

f32_Traction_TorqueDem,   IN_TS5.f32_Traction_TorqueDem,    1, 0;
f32_Traction_TorqueRes,   IN_TS5.f32_Traction_TorqueRes,    1, 0;
f32_RpmLim_TorqueDem,     IN_TS5.f32_RpmLim_TorqueDem,      1, 0;
f32_RpmLim_TorqueRes,     IN_TS5.f32_RpmLim_TorqueRes,      1, 0;
f32_CruiseCtrl_TorqueDem, IN_TS5.f32_CruiseCtrl_TorqueDem,  1, 0;
f32_CruiseCtrl_TorqueRes, IN_TS5.f32_CruiseCtrl_TorqueRes,  1, 0;
u32_Launch_State,         IN_TS5.u32_Launch_State,          1, 0;
f32_Launch_ThrDem,        IN_TS5.f32_Launch_ThrDem,         1, 0;
f32_Launch_TorqueDem,     IN_TS5.f32_Launch_TorqueDem,      1, 0;
```

file .H

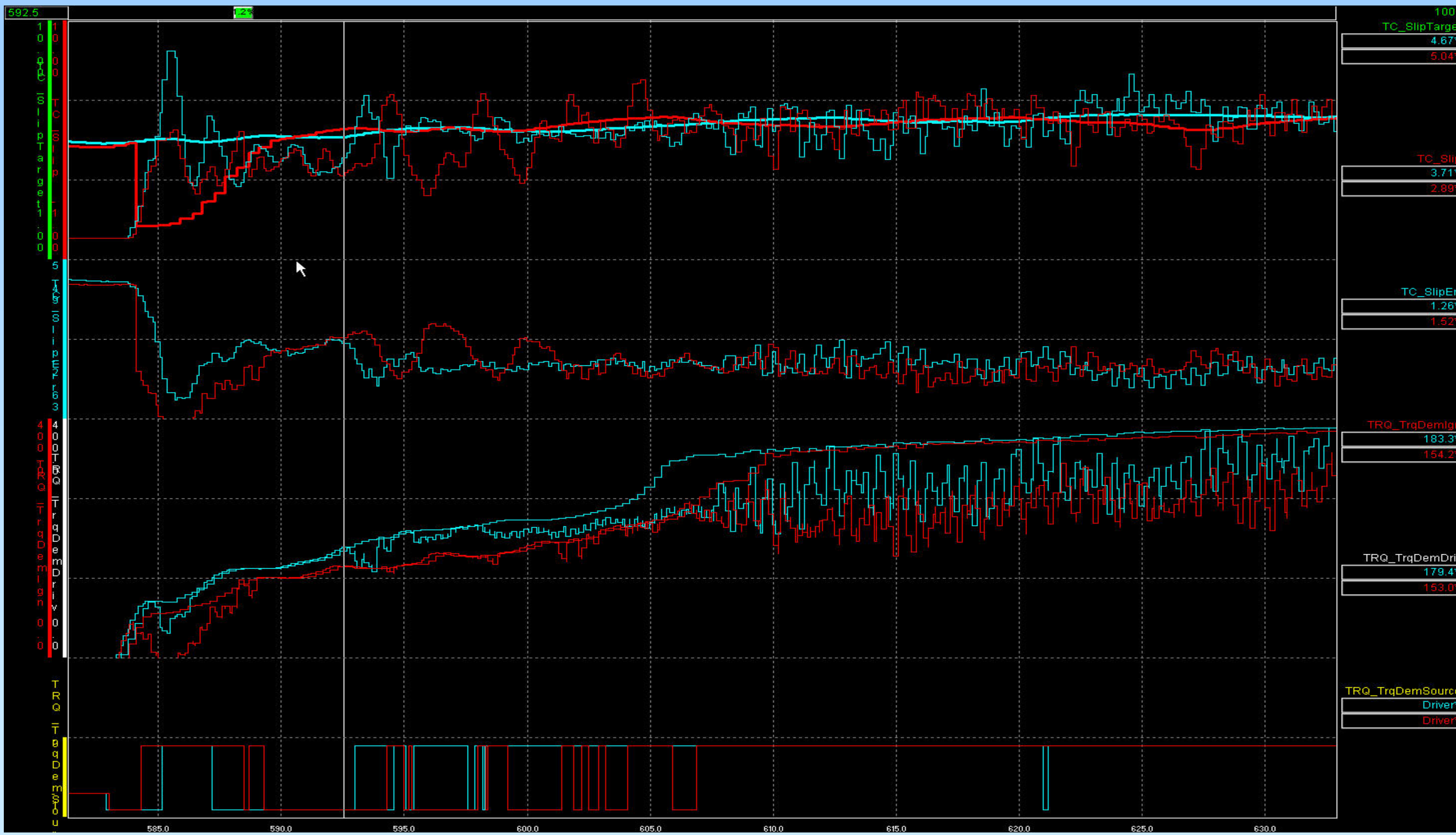
Code automatically generated to handle messaging through DPR, LinkPort, ARCNet

Target Slip Shaper was successfully used during T07 Event in Jerez:

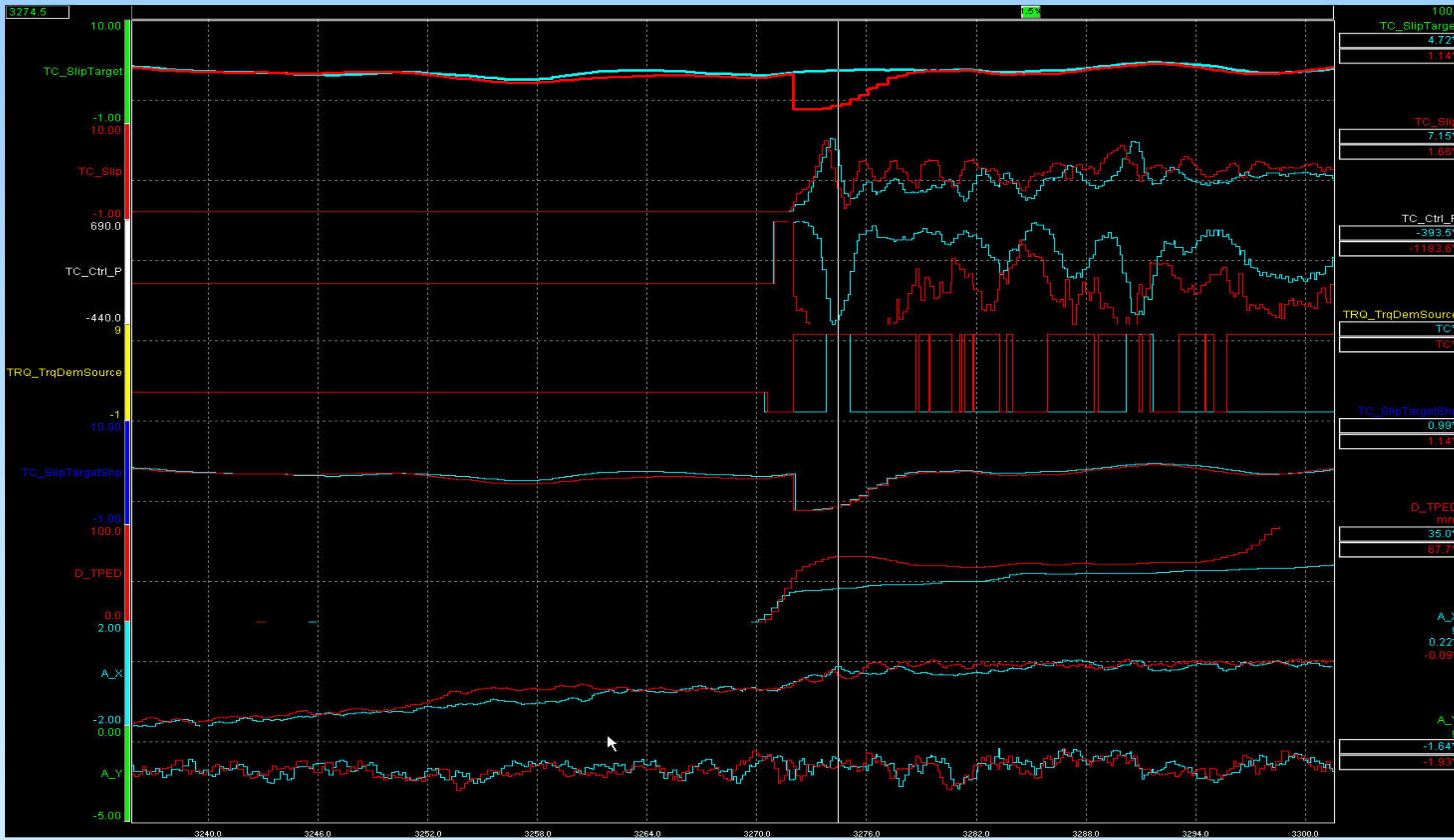
Event	T07		
Track	Jerez		
Date	07/02/2005		
Ctrl Eng.	BL/LT/FR		
Chassis	502	503	411
Drivers	RZ/RS	JT	RZ

LT commented:

- **Type used.** 7 poles at around 18Hz, 0.1 damping ratio.
- **Slip Overshoot.** significantly less slip overshoot in almost every case in the test performed. This should result in a less snappy response for drivers like **RZ** whose throttle application is quite aggressive. This could also result in less snap oversteer which is what **RZ** complained of the most today. The reduced slip overshoot is significant and to me should result in less tyre usage, if only for a short time.
- **Inside wheel oscillation.** No real difference seen in the runs we tested.
- **Performance implications.** As can be seen in the figures below the performance of the slip shaper in not noticeably different in terms of acceleration to that of no slip shaper. However as the driver is then capable of simply flooring the throttle and not having to worry about the system snapping without loss of traction, the result is clear, maybe. In several cases, significantly, the shaper achieves the target in the same amount of time as the standard method, i.e. when there is large undershoot.

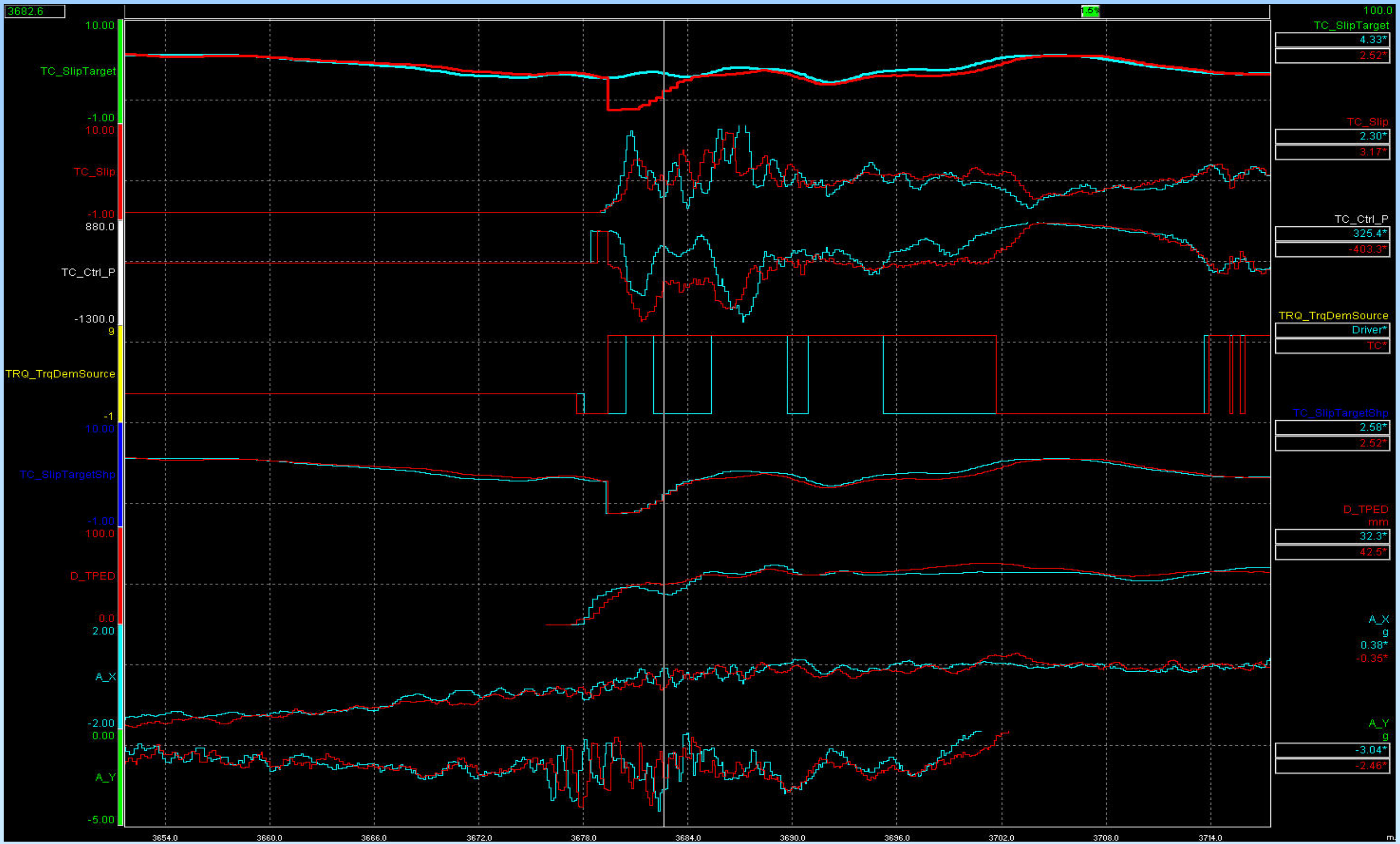


Effect of TC shaper applied (**red**) and not (**blue**). Much higher overshoot initially. Always had as good if not lower degree of overshoot.



Effect of much higher application of throttle and the corresponding shapes of the slip overshoot. In this case higher throttle is tempered by the shaper. The corresponding accelerations show no obvious difference.

Test Report



Similar throttle application resulting in lower slip overshoot.

Auto-tuning through GA

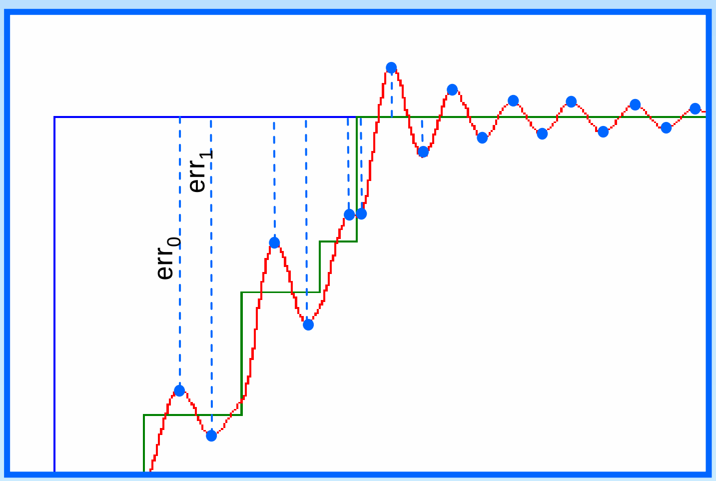
Use Genetic Algorithm to find shaper parameters which best fit model response

Chromosome definition:

C_0	T_1	C_1	T_2	C_2	…	T_n	C_n

Fitness function definition:

$$fitness = \sum_{i=0}^{N} err_i^2$$



ITERATIVE PROCESS to generate new populations:

- Calculate fitness function for each chromosome of population.
- Apply **elitism**: only two best chromosomes of each population will survive unchanged.
- Apply **crossover**.
- Apply **mutation** (with a specified probability).
- Discard worst two chromosomes.
- Apply **constraints** to chromosomes of new population: C and T need to be normalized => $\Sigma C_i=1$, $\Sigma T_i<max$.

GA is capable of finding parameters which fit response model in a better way with respect to theoretical parameters (with the same number of zeros):

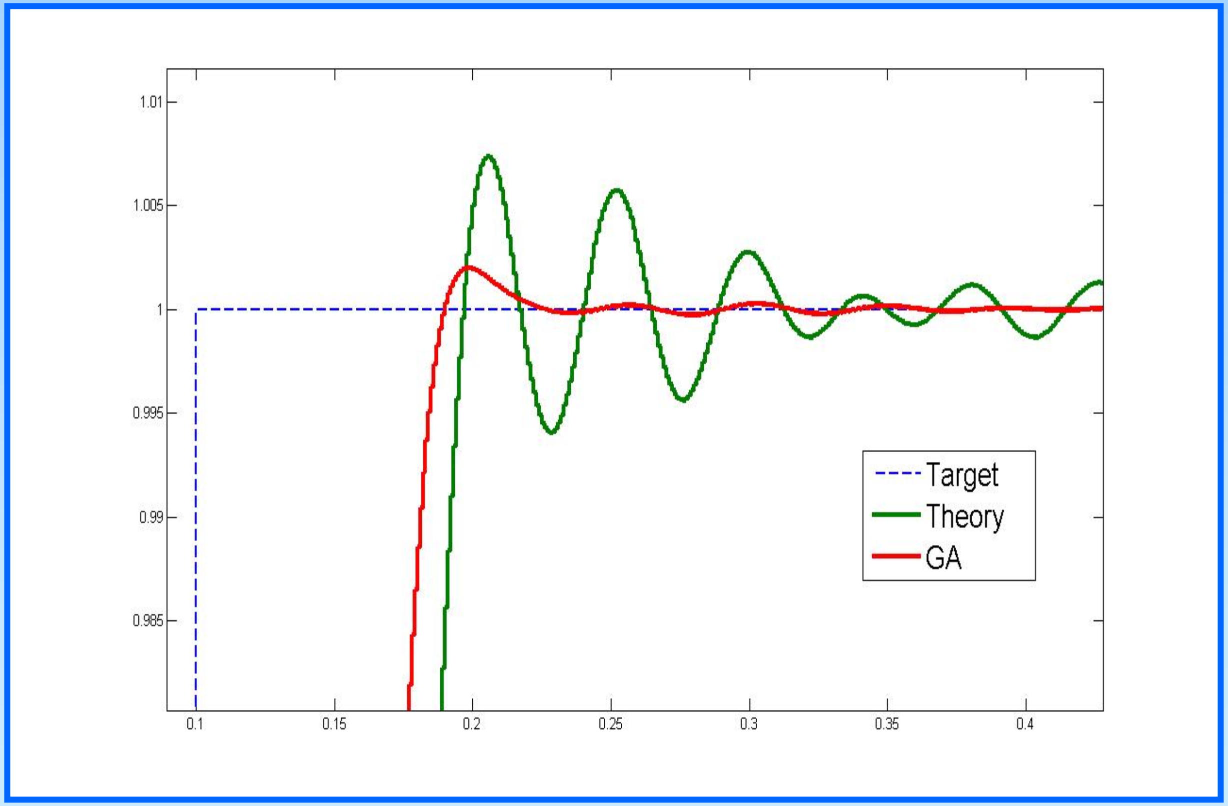
Test with 3rd order plant:

f_1=17.5 Hz, ζ_1=0.1

f_2=20 Hz, ζ_2=0.7

f_3=23 Hz, ζ_3=0.05

- Useful also when Plant dynamic is not known.



Run Shaper Scan

References

References

1) Smith, O. J. M., “Posicast Control of Damped Oscillatory Systems”, Proc. of the IRE, 1957, pp 1249-1255.
2) W. E. Singhose, “Command Generation for Flexible Systems”, S.B.M.E., Massachusetts Institute of Technology, 1990.