EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VLT PROGRAMME

Accepted by ESO Technical Archive

ESO Technical Archive 02.05.2008

VERY LARGE TELESCOPE

Adaptive RLS Vibration Cancellation algorithm

Doc. No.: VLT-SPE-ESO-15400-4558

Issue: 1

Date: 30/04/2008

Prepared:	Nicola Di Lieto Name	30/04/2008 Date	Signature
Approved:	Thanh Phan Duc Name	30/4/08 Date	Signature
Released:	Francoise Delplancke Name	30/04/08 Date	Signature

CHANGE RECORD

ISSUE	DATE	SECTION/PAGE AFFECTED	REASON/INITIATION DOCUMENTS/REMARKS
1	30/04/2008	All	First issue

TABLE OF CONTENTS

This page has been left intentionally blank.

1 Introduction

1.1 Purpose

This document describes the Adaptive Recursive Least Squares (RLS) Vibration Cancellation algorithm, also known as "New Vibration Tracking", as currently used by version 2.0 and later of the OPDC Controller Software that is part of the VLTI Fringe Tracking (FTK) Facility.

1.2 Scope

This document shortly describes the mathematical derivation of the algorithm. It then presents the implementation of this software component as a block in the Tools for Advanced Control (TAC) library ([AD#04]).

1.3 Applicable Documents

The following documents, of the issue shown if specified, form part of this manual to the extent specified herein. In the event of conflict between this document and those referenced, the content of this document shall be considered as a superseding requirement unless explicitly stated otherwise herein.

Reference	Document Number	Issue	Date	Title
[AD#01]	VLT-MAN-ESO-17210-0667	1.0	03/12/1996	Guidelines for the Development of VLT Application Software
[AD#02]	VLT-MAN-SBI-17210-0001	3.5	20/10/1999	LCU Common Software User Manual
[AD#03]	VLT-MAN-ESO-17210-2252	2	05/07/2002	LCU Server Framework User Manual
[AD#04]	VLT-MAN-ESO-17210-2970	2	21/03/2004	Tools for Advanced Control User Manual

1.4 Reference Documents

The following documents contain additional information that can be useful to the reader:

Reference	Document Number	Issue	Date	Title
[RD#01]	VLT-SPE-ESO-15420-4000	2	30/04/2008	OPD Controller (As-built) Software Detailed Design
[RD#02]	VLT-SPE-ESO-15400-4013	2	02/04/2007	VLTI-UT Vibration Monitoring System

1.5 Acronyms

This document employs several abbreviations and acronyms to refer concisely to an item, after it has been introduced. The following list is aimed to help the reader in recalling the extended meaning of each short expression:

Acronym	Meaning
AD	Applicable Document
CPU	Central Processing Unit
DL	Delay Line
FTK	Fringe Tracking
LCU	Local Control Unit
LCC	LCU Common Software
LSF	LCU Server Framework
ITF	Interferometer Task Force
OPD	Optical Path Difference
OPDC	OPD Controller
PLL	Phase Locked Loop
RD	Reference Document
RLS	Recursive Least Squares
RT	Real-Time
RTC	RT Controller
TAC	Tools for Advanced Control
UT	Unit Telescope
VLT	Very Large Telescope
VLTI	VLT Interferometer
VTK	Vibration Tracking
ZPD	Zero Path Difference

2 Algorithm Description

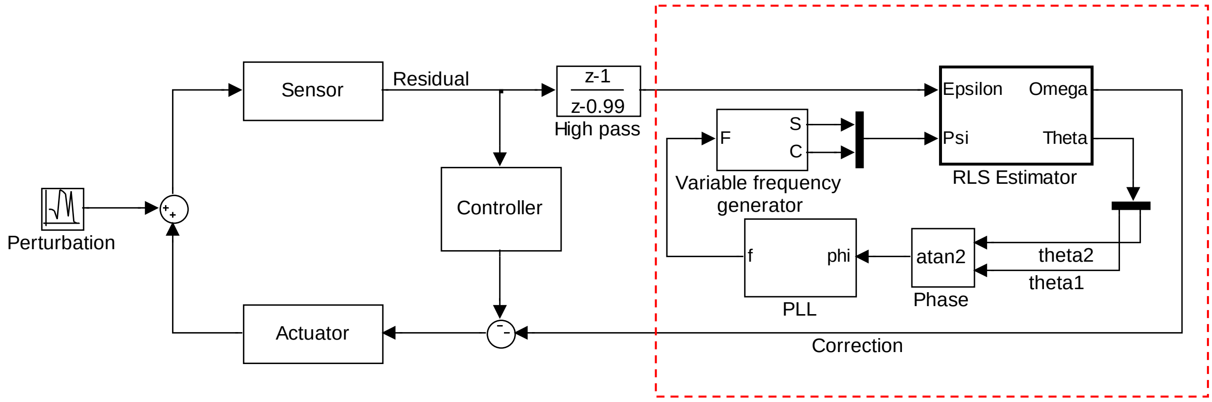
2.1 Overview

The VLTI is suffering from piston induced by vibrations of components in the optical train. Most vibrations originating on M1, M2 and M3 of the UTs are measured by accelerometers and compensated in feed-forward (RD#1). Periodic vibrations originating elsewhere or outside the frequency band (15-35 Hz) of the accelerometer-based compensation system can be modelled and compensated in real time by a suitably robust adaptive algorithm.

A former algorithm (VTK), made available by ITF in July-August 2006, failed to perform robustly enough. The necessity to tune many parameters made it operationally complex and unfortunately unusable. The VLTI team in Paranal asked the author to try and design an algorithm easier to operate, more robust and suitable for unattended operation.

The new algorithm is capable of attenuating near-periodic, slowly drifting perturbations at frequencies above the closed loop bandwidth. It is in principle applicable to any closed loop system.

2.2 Principle



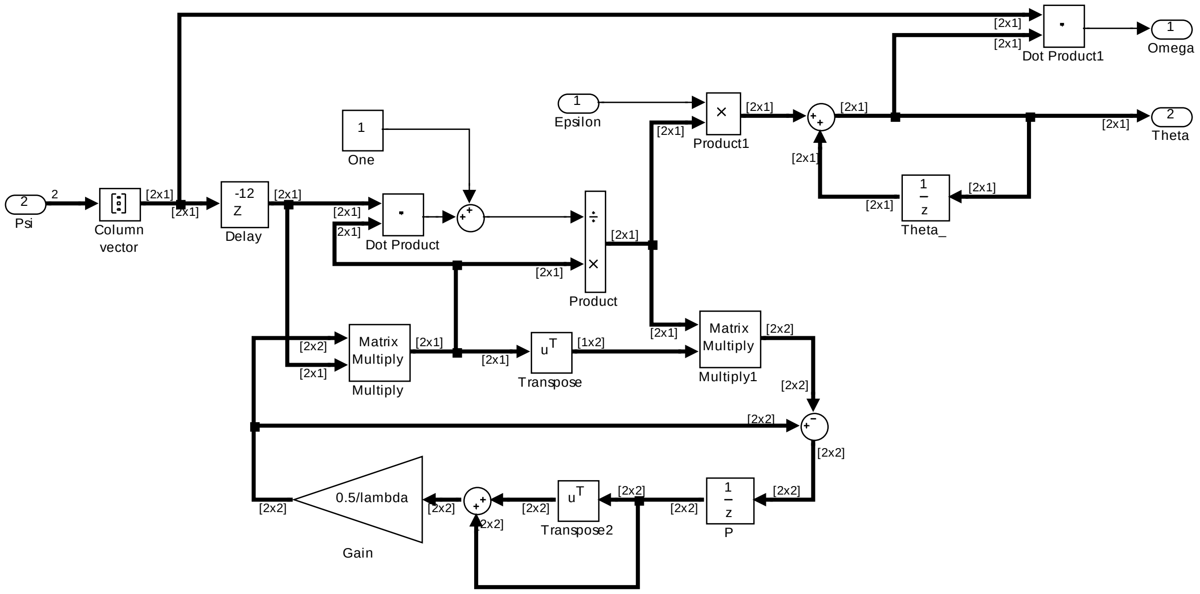
Two sine waves S_k and C_k, shifted by 90 degrees are synthesized using the recursive formulation

$$S_k = S_{k-1}\cos(2\pi\,\Delta t\, f_k) + C_{k-1}\sin(2\pi\,\Delta t\, f_k)$$
$$C_k = C_{k-1}\cos(2\pi\,\Delta t\, f_k) - S_{k-1}\sin(2\pi\,\Delta t\, f_k)$$
$$S_0 = 0$$
$$C_0 = 1$$

A linear combination of S_k and C_k is used to produce the adaptive compensation signal Ω_k to be summed to the output of the control system:

$$\Omega_k = \theta_{k,1} S_k + \theta_{k,2} C_k$$

Ω_k is an amplified replica of S_k, phase shifted by $\varphi_{e,k} =$ atan2($\theta_{k,2}$, $\theta_{k,1}$). Since Ω_k is linear in θ_k, the latter can be estimated recursively in real time using the RLS (recursive least squares) algorithm to minimize the control system error signal ε_k, as follows:



$$\psi_k = \begin{bmatrix} S_k \\ C_k \end{bmatrix}$$

$$\theta_k = \theta_{k-1} + \varepsilon_k \frac{P_{k-1}\psi_{k-D}}{\lambda + \psi_{k-D}^T P_{k-1}\psi_{k-D}}$$

$$P_k = \frac{1}{2\lambda}\left\{ \left(P_{k-1} - \frac{P_{k-1}\psi_{k-D}\psi_{k-D}^T P_{k-1}^T}{\lambda + \psi_{k-D}^T P_{k-1}\psi_{k-D}} \right) + \left(P_{k-1} - \frac{P_{k-1}\psi_{k-D}\psi_{k-D}^T P_{k-1}^T}{\lambda + \psi_{k-D}^T P_{k-1}\psi_{k-D}} \right)^T \right\}$$

$$P_0 = \begin{bmatrix} V_0^{-1} & 0 \\ 0 & V_0^{-1} \end{bmatrix}$$

$$\theta_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Where

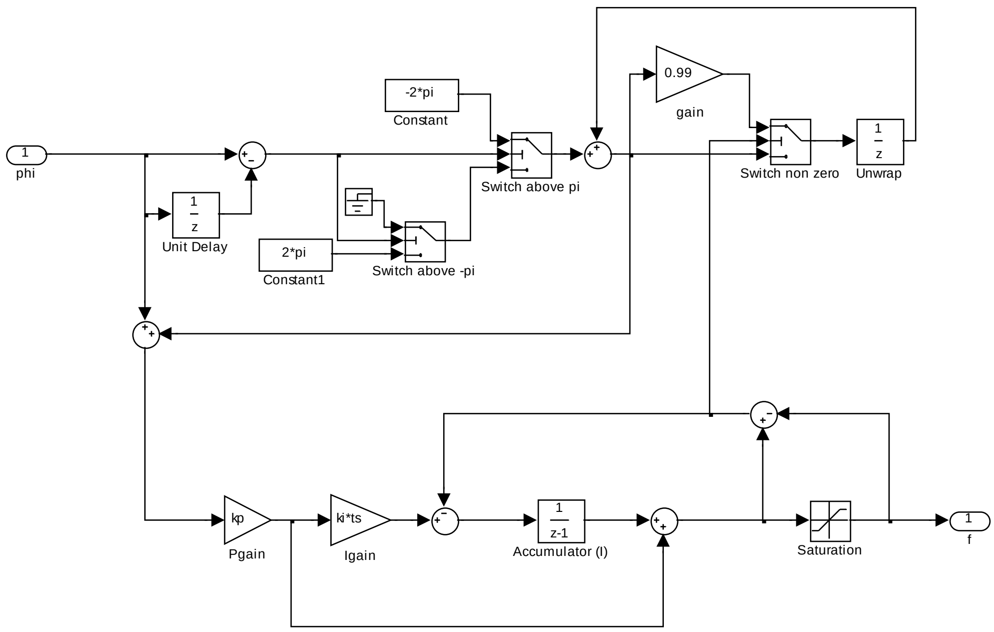
V_0	Initial parameter variance
λ	Forgetting factor
D	Plant pure delay (expressed in number of samples)

If f_k matched exactly the disturbance frequency, the RLS estimator would converge towards a constant estimated phase shift φ^*, dependent on the delay/lags in the system only.

Of course the disturbance frequency is never known accurately and it may also drift. In presence of a frequency mismatch df, the optimum phase shift (in the least squares sense) varies in time, according to $\varphi = \varphi^* + 2\pi\, \mathrm{d}f\, t$. In this situation the RLS estimator produces a time varying φ_e and in absence of a frequency adaptation mechanism the disturbance cancellation performance deteriorates.

However, note that φ is the integral of d*f*. Provided d*f* is slow enough compared to the RLS estimation dynamics, φ_e closely tracks φ. Under these conditions the RLS estimator can be used as the phase detector in a Phase Locked Loop, i.e. it is possible to lock f_k onto the real disturbance frequency by applying a frequency correction proportional to φ_e.

The actual PLL implementation must also contain frequency limits and be protected against windup and phase wrap-around. This can be achieved as follows:



$$\Phi_0 = 0$$

$$\Phi_k = \begin{cases} \mu_{k-1}\Phi_{k-1} & if \left|\varphi_{e,k} - \varphi_{e,k-1}\right| < \pi \\ \mu_{k-1}\left(\Phi_{k-1} - 2\pi\right) & if\, \varphi_{e,k} - \varphi_{e,k-1} \geq \pi \\ \mu_{k-1}\left(\Phi_{k-1} + 2\pi\right) & if\, \varphi_{e,k} - \varphi_{e,k-1} \leq -\pi \end{cases}$$

$$\mu_k = \begin{cases} 0.99 & if\, \tilde{f}_k \neq f_k \\ 1 & if\, \tilde{f}_k = f_k \end{cases}$$

$$\tilde{f}_k = K_p\left(\Phi_k + \varphi_{e,k}\right) + I_{k-1}$$

$$f_k = \begin{cases} f_{\min} & if\, \tilde{f}_k < f_{\min} \\ f_{\max} & if\, \tilde{f}_k > f_{\max} \\ \tilde{f}_k & otherwise \end{cases}$$

$$I_0 = 0$$

$$I_k = I_{k-1} + \Delta t K_i K_p\left(\Phi_k + \varphi_k\right) + \left(f_k - \tilde{f}_k\right)$$

where

K_i Integral gain

K_p Proportional gain

2.3 TAC implementation

A new RLSAlgo TAC block is available in taclib 2.4 and later. The block was validated on the control model to produce the same calculations illustrated in the block diagrams and formulas above. The ability to freeze the frequency/phase adaptation and smooth the amplitude down to zero was also added to enhance robustness in applications like the OPDC Fringe Tracker where residuals can be intermittently unavailable.

```
/***********************************************************************
*   NAME
*     tacRLSAlgoBlock - RLS adaptive narrow band disturbance canceller
*
*   DESCRIPTION
*     Block type: RLSAlgo
*     Parameters: 8
*                 <FINIT>  Initial Frequency guess          (Hz)
*                 <FMIN>   Minimum Frequency                (Hz)
*                 <FMAX>   Maximum Frequency                (Hz)
*                 <KP>     PLL proportional gain            (Hz/rad)
*                 <KI>     PLL integral gain                (1/s)
*                 <LAMBDA> RLS Forgetting factor            (-)
*                 <LAG>    Plant pure delay                 (s)
*                 <VINIT>  RLS Initial parameter variance   (-)
*     Inputs:     4
*                 <Residual error>                          (m)
*                 <Enable>                                  (-)
*                 <Reset>                                   (-)
*                 <Adapt>                                   (-)
*     Outputs:    4
*                 <Compensation>                            (m)
*                 <Frequency>                               (Hz)
*                 <Phase>                                   (rad)
*                 <Amplitude>                               (m)
*
*     This function block implements an adaptive narrow band disturbance
*     canceller.  It generates a synthetic sinusoidal compensation signal
*     whose phase and amplitude are adapted recursively in real time
*     to minimize (in the least square sense) the residual error.
*
*     In parallel, a Phase Locked Loop corrects the initial frequency guess
*     and makes it track the real disturbance frequency by locking onto the
*     phase estimated by the recursive least square (RLS) algorithm.
*
*     The PLL frequency is limited between FMIN and FMAX. The internal PLL
*     integrator and estimated phase unwrapper are protected against windup.
*
*     Typical parameter values:
*                 <FINIT>  18
*                 <FMIN>   15
*                 <FMAX>   20
*                 <KP>     0.1
*                 <KI>     1.0
*                 <LAMBDA> 0.998
*                 <LAG>    6e-3
*                 <VINIT>  100
*
*     This function block allows modification of all parameters at runtime.
*
*     When the <Enable> input is less than 0.5 the block freezes, i.e. internal
*     states and outputs are not updated.
*
*     When the <Reset> input is greater than 0.5 the internal states and outputs
*     are reset to their initial values, regardless of the <Enable> input.
*
*     When the <Adapt> input is less than 1.0, the block adaptation freezes,
*     i.e. phase and frequency are not updated. In addition, the amplitude is
*     smoothed down to zero by the following exponential law:
*
*       Amplitude(k) = Amplitude(k-1)*Adapt
*
*     For normal operation set <Enable> = 1, <Reset> = 0 and <Adapt> = 1
*
*   EXAMPLES
*     TAC.BLOCK9.TYPE    RLSAlgo
*     TAC.BLOCK9.NAME    RLSAlgo
*     TAC.BLOCK9.PARAM   18,15,20,0.1,1.0,0.998,6e-3,100
```

3 An example

The RLSAlgo block described above is general purpose. As many instances as necessary can be instantiated and connected in parallel, taking of course care to remain within the available CPU capabilities. Below is a snippet of how this is done in the OPDC software, starting from version 2.0.

```
VTKAlgoFunc()
{
    #  Defaults for internal parameters
    typeset DistNb=${1:-2}

    I=1; while [ $I -le $DistNb ]; do
           TAC_BLOCK _TrkEnable_$I Constant 0
           TAC_BLOCK _TrkReset_$I  Constant 0
           TAC_BLOCK _Trk_$I       RLSAlgo  0.0,0.0,0.0,0.1,0.1,0.99,0.004,100.0
           I=`expr $I + 1`
    done

    TAC_BLOCK _VibTrk  Multi $DistNb,Sum,\"++\"
    TAC_BLOCK _VibGain Gain  $RadToM
    TAC_BLOCK _LPFilter LowPass 0.1,1
    TAC_BLOCK _HPFilter Sum \"+-\"
    TAC_BLOCK _TrkAdapt Relay    6.5 0.1 1 0.9999

    # Links
    TAC_LINK Instance  INPUT     1 _VibGain 1
    TAC_LINK Instance  INPUT     2 _TrkAdapt 1
    TAC_LINK _Inputm   _VibGain  1 _LPFilter 1
    TAC_LINK _Inputm   _VibGain  1 _HPFilter 1
    TAC_LINK _LPFilter _LPFilter 1 _HPFilter 2
    I=1; while [ $I -le $DistNb ]; do
           TAC_LINK _TrkErr   _HPFilter     1 _Trk_$I 1
           TAC_LINK _Trk_En   _TrkEnable_$I 1 _Trk_$I 2
           TAC_LINK _Trk_Rst  _TrkReset_$I  1 _Trk_$I 3
           TAC_LINK _Trk_Adpt _TrkAdapt     1 _Trk_$I 4
           TAC_LINK _Trk_$I _Trk_$I 1 _VibTrk $I
           I=`expr $I + 1`
    done
    TAC_LINK Instance _VibTrk 1 OUTPUT 1
    TAC_LINK Instance _Trk_1  2 OUTPUT 2
    TAC_LINK Instance _Trk_1  3 OUTPUT 3
    TAC_LINK Instance _Trk_1  4 OUTPUT 4
}
```

The TAC code above creates `DistNb` instances of the `RLSAlgo` block, initializing each of them with dummy parameters. The actual parameters are set programmatically by the STRTVTK command implemented in `opdcSoftDev.c`; the numbers are currently read from a file but this mechanism is temporary and will be replaced by a proper database interface in the near future. The `Relay` block is connected to the `<Adapt>` input of the `RLSAlgo` block and is 1 when the OPDC state is 7 (=COPHASE_LOCK) or 0.9999 otherwise. Basically, whenever the fringe is not locked, the phase/frequency adaptation is frozen and the amplitude is smoothed slowly down to zero. The `_TrkEnable_X` and `_TrkReset_X` blocks are changed by the underlying OPDC STRTVTK/STOPVTK commands to control the algorithm.

In practice it was found that on OPDC the following set of parameters works well in all conditions independently of the frequency:

```
Init Freq, Min Freq, Max Freq, 0.1, 0.1, 0.998, 0.005, 100
```

____oOo____