University of Glasgow

DEPARTMENT OF

AEROSPACE ENGINEERING

arallelism of Rotorcraft Trimming Strategies

David H. Ewing

Internal Report No. 9513
September 1995

Parallelism of Rotorcraft Trimming Strategies

David H. Ewing

Internal Report No. 9513
September 1995

Summary

This report describes the inherent parallelism and technical merit of two trimming strategies for use in individual blade rotorcraft simulation models. The RASCAL model has been used to contrast the McVicar-Bradley trimmer with its own trimming strategy.

Nomenclature

Variables

$\underline{F}$	error vector	
F_i	element in error vector	
J	Jacobian matrix	
P	permutation matrix	
j_{ij}	element in Jacobian matrix	
nblade	number of blades	
ncontrols	number of Newton-Raphson control states	
niterations	number of Newton-Raphson iterations	
p,q,r	rotational velocities	(rad / s)
$\dot{p},\dot{q},\dot{r}$	rotational accelerations	(rad / s^2)
turn	aircraft turn rate	(rad / s)
u,v,w	translational velocities (body axes)	(m / s)
$\dot{u},\dot{v},\dot{w}$	translational accelerations (body axes)	(m / s^2)
u_e,v_e,w_e	translational velocities (inertial axes)	(m / s)
$\underline{u}$	control vector	
u_j	element in control vector	
v_{i0},v_{1s},v_{1c}	uniform and cyclic inflow components	(m / s)
$\underline{y}$	output vector	
y_i	element in output vector	
β,ζ	blade flap and lag angles	$(degrees)$
$\dot{\beta},\dot{\zeta}$	blade flap and lag rates	$(degrees / s)$
ϕ_f,θ_f	fuselage roll and pitch angles	$(degrees)$
$\theta_0,\theta_{1s},\theta_{1c}$	collective and cyclic pitch components	$(degrees)$

Subscripts

con	controls
des	desired
fs	flight state
k	Newton-Raphson iteration index
mfs	mean flight state
mr	main rotor
tr	tail rotor

Superscripts

b	beginning of forcing period
e	end of forcing period

Introduction

This paper describes the exploitation of the inherent parallelism that exists in rotorcraft simulation trimming strategies and contrasts the computational performance of two such strategies. The rotorcraft simulation model RASCAL [1] was used to compare its current trimming strategy [2] with an implementation of the McVicar-Bradley trimmer [3]. The McVicar-Bradley trimmer, which was developed for an individual blade/blade element tilt-rotor simulation [4], takes advantage of the fact that the period of the forcing from the rotors on the airframe is short. RASCAL, which is also an individual blade/blade element model, is able to simulate the full suite of rotorcraft configurations including ones such as the main and tail rotor helicopters in which the period of the forcing from the rotors on the airframe is relatively long.

An aircraft can be said to be in trim when the mean accelerations present are zero. Given dynamic stability, the aircraft will then fly to some prescribed flight condition. The pilot is able to trim the aircraft to this prescribed condition by altering the controls that are available which, in the case of a main and tail rotor helicopter are, collective, longitudinal cyclic and lateral cyclic pitch of the main rotor and collective pitch of the tail rotor.

In simulation terms the solution of trim is more complicated as it is not only the required controls that need to be calculated but also all of the other states that are present. For example, the blade states, wake states, engine states and flight states are required. The situation is further complicated in individual blade models as the states will be time-varying so either a time history or a multi-blade formulation is necessary.

The solution of the trim state of individual blade/blade element rotorcraft models, generally involve Newton-Raphson techniques and are computationally intensive. Parallel computing techniques have been employed to improve the run-time in order to ensure that such models can be utilised as design tools [5].

Parallelisation of Newton-Raphson Techniques

The solution of trim involves determining the required controls, $\underline{u}$, to achieve a desired flight condition, $\underline{y}_{des}$. An error function, $\underline{F}$, can be defined as the difference between the output $\underline{y}$ achieved using the current estimate of the controls and the desired output.

$$\underline{F} = \underline{y} - \underline{y}_{des}$$

If the error is outwith some tolerance then an updated estimate of he controls can be found using the Newton-Raphson iteration.

$$\underline{u}_{k+1} = \underline{u}_k - J^{-1}\underline{F}$$

Where, J, is the Jacobian matrix whose elements, j_{ij}, are constructed in the following manner.

$$j_{ij} = \frac{\partial F_i}{\partial u_j}$$

And are calculated using central differencing in the following manner.

$$j_{ij} = \frac{F_i(u_j + \delta u_j) - F_i(u_j - \delta u_j)}{2\delta u_j}$$

Where δu_j is a perturbation in u_j.

Each iteration requires 2 forward simulations per control plus 1 forward simulation with the unperturbed controls to evaluate the error function. New controls can then be estimated and the output with these updated controls can be evaluated and check against the tolerance. If it is outwith the tolerance then a new Jacobian matrix is constructed and the process is repeated until the converge criteria are met [6]. The sequential implementation of this method is described in figure 1.

As each of the outputs from perturbed controls that make up the Jacobian matrix are entirely independent of each other they can be carried out *in parallel*. One such parallel implementation is described in figure 2. On making the following assumptions it is possible to estimate the performance benefits that can be obtained from such a parallelisation.

1) The number of available computational nodes is greater than the number of parallel tasks.

2) Each computational node displays similar performance.

3) There are no other users on the available nodes.

4) The parallel overhead (set up and communication) is small when compared to the run-time of the forward simulations.

5) The time taken to invert the Jacobian and estimate the new controls is small when compared to the run-time of the forward simulations.

Here it is possible to estimate that the potential maximum speed up achievable is of the order of the number of controls plus one [5]. The achievable speed up can be improved by concurrently calculating the output from the perturbed new controls whilst tolerance checking those new controls. Where the new controls fall within the tolerance the perturbed output is not needed but no additional run-time is incurred and if the new controls are outwith the tolerance then the output is available straight away. This parallel implementation is described in figure 3. Given the previous assumptions the speed up is now a function of the number of controls and also the number of Newton-Raphson iterations. It can be estimated from the following expression.

$$SpeedUp = \frac{\{(ncontrols + 1) * 2\} * niterations}{niterations + 1} \qquad (1)$$

Equation (1) describes the motivation in trying to use the McVicar-Bradley trimming strategy as it uses a large number of 'control states' so the potential parallel performance improvement is vast.

Original Rascal Trimmer

The original RASCAL trimming strategy operates by solving for the mean accelerations over the forcing period to be zero. The control vector used in the Newton-Raphson iteration is of the form.

$$\underline{u} = \begin{bmatrix} \theta_{0_{mr}} & \theta_{1s_{mr}} & \theta_{1c_{mr}} & \theta_{0_{tr}} & \phi_f & \theta_f \end{bmatrix}^T$$

and the error vector is simply the resultant accelerations present after the forward simulation.

$$\underline{F} = [\dot{u} \quad \dot{v} \quad \dot{w} \quad \dot{p} \quad \dot{q} \quad \dot{r}]^T$$

In order for the iteration to converge the accelerations calculated must represent the accelerations due to the aircraft's controls being in an off-trim condition. There will be other accelerations present as the initial estimates of the blade and wake states will be decaying to their trimmed periodic values. In order to exclude these accelerations from the ones required a settling period must be allowed which will allow the transients to decay. The mean

accelerations are then calculated over the full forcing period beginning after a period of settling.

This strategy is susceptible to two types of problems. Firstly, the period of time over which the transients are allowed to decay is unclear. If too long a time is allowed the computational performance deteriorates and if too short a time is allowed then the resultant accelerations calculated will include some information that pertains to the transients decaying. Another potential problem is that if the initial estimate of the blade and wake states are near a stability boundary then the transients may fail to converge to their periodic trim values for the particular flight and control state given. The second problem associated with this method is that the overall period of time required to be simulated is sufficiently long as to allow the rigid body modes to develop. This ensures that the mean accelerations calculated contains information regarding the rigid body modes.

Houston overcomes these problems by allowing a long period of time to let the transients decay - at the expense of the computational performance (typically 6 turns of the main rotor) and by suppressing the integration of the flight states in order to ensure that the rigid body modes are not allowed to develop. This causes some discrepancy in the calculation of the forces and moments as those arising from the flight state accelerations are not calculated.

McVicar-Bradley Trimmer

The McVicar-Bradley trimmer overcomes the transient decay problems associated with the RASCAL trimmer as the wake and blade states are solved for explicitly within the Newton-Raphson iteration.

The control vector within the Newton-Raphson iteration takes the form:

$$\underline{u} = \begin{bmatrix} \underline{u}_{con} & \underline{u}_{fs} & \underline{u}_{wake_{mr}} & \underline{u}_{wake_{tr}} & \underline{u}_{blade_{mr}} & \underline{u}_{blade_{tr}} \end{bmatrix}^T$$

Where

$$\underline{u}_{con} = \begin{bmatrix} \theta_{0_{mr}} & \theta_{1s_{mr}} & \theta_{1c_{mr}} & \theta_{0_{tr}} \end{bmatrix}^T$$

$$\underline{u}_{fs} = \begin{bmatrix} u & v & w & p & q & r & \phi_f & \theta_f \end{bmatrix}^T$$

$$\underline{u}_{wake_{mr}} = \begin{bmatrix} v_{i0_{mr}} & v_{1s_{mr}} & v_{1c_{mr}} \end{bmatrix}^T$$

$$\underline{u}_{wake_{tr}} = \begin{bmatrix} v_{i0_{tr}} & v_{1s_{tr}} & v_{1c_{tr}} \end{bmatrix}^T$$

$$\underline{u}_{blade_{mr}} = \begin{bmatrix} \dot{\underline{\beta}}_{mr} & \underline{\beta}_{mr} & \dot{\underline{\zeta}}_{mr} & \underline{\zeta}_{mr} \end{bmatrix}^T$$

Where

$$\dot{\underline{\beta}}_{mr} = \begin{bmatrix} \dot{\beta}_1 & .. & \dot{\beta}_{nblade_{mr}} \end{bmatrix}^T$$

$$\underline{\beta}_{mr} = \begin{bmatrix} \beta_1 & .. & \beta_{nblade_{mr}} \end{bmatrix}^T$$

$$\dot{\underline{\zeta}}_{mr} = \begin{bmatrix} \dot{\zeta}_1 & .. & \dot{\zeta}_{nblade_{mr}} \end{bmatrix}^T$$

$$\underline{\zeta}_{mr} = \begin{bmatrix} \zeta_1 & .. & \zeta_{nblade_{mr}} \end{bmatrix}^T$$

And similarly,

$$\underline{u}_{blade_{tr}} = \begin{bmatrix} \dot{\underline{\beta}}_{tr} & \underline{\beta}_{tr} & \dot{\underline{\zeta}}_{tr} & \underline{\zeta}_{tr} \end{bmatrix}^T$$

Where

$$\dot{\underline{\beta}}_{tr} = \begin{bmatrix} \dot{\beta}_1 & .. & \dot{\beta}_{nblade_{tr}} \end{bmatrix}^T$$

$$\underline{\beta}_{tr} = \begin{bmatrix} \beta_1 & .. & \beta_{nblade_{tr}} \end{bmatrix}^T$$

$$\dot{\underline{\zeta}}_{tr} = \begin{bmatrix} \dot{\zeta}_1 & .. & \dot{\zeta}_{nblade_{tr}} \end{bmatrix}^T$$

$$\underline{\zeta}_{tr} = \begin{bmatrix} \zeta_1 & .. & \zeta_{nblade_{tr}} \end{bmatrix}^T$$

As the trimmed flight state will be periodic the problem is to solve for the controls that ensure the desired mean flight states whilst concurrently solving for the flight, blade and wake states that ensure periodicity. The error vector in the Newton-Raphson iteration takes the form:

$$\underline{F} = \begin{bmatrix} \underline{F}_{mfs} & \underline{F}_{fs} & \underline{F}_{wake_{mr}} & \underline{F}_{wake_{tr}} & \underline{F}_{blade_{mr}} & \underline{F}_{blade_{tr}} \end{bmatrix}^T$$

Where

$$\underline{F}_{mfs} = \begin{bmatrix} \bar{u}_e - \bar{u}_{e_{des}} & \bar{v}_e - \bar{v}_{e_{des}} & \bar{w}_e - \bar{w}_{e_{des}} & \overline{turn} - \overline{turn}_{des} \end{bmatrix}^T$$

$$\underline{F}_{fs} = \begin{bmatrix} u^e - u^b & v^e - v^b & w^e - w^b & p^e - p^b & q^e - q^b & r^e - r^b & \phi_f^e - \phi_f^b & \theta_f^e - \theta_f^b \end{bmatrix}^T$$

$$\underline{F}_{wake_{mr}} = \begin{bmatrix} v_{i0_{mr}}^e - v_{i0_{mr}}^b & v_{1s_{mr}}^e - v_{1s_{mr}}^b & v_{1c_{mr}}^e - v_{1c_{mr}}^b \end{bmatrix}^T$$

$$\underline{F}_{wake_{tr}} = \begin{bmatrix} v_{i0_{tr}}^e - v_{i0_{tr}}^b & v_{1s_{tr}}^e - v_{1s_{tr}}^b & v_{1c_{tr}}^e - v_{1c_{tr}}^b \end{bmatrix}^T$$

$$\underline{F}_{blade_{mr}} = \left[\underline{F}_{\dot{\beta}_{mr}} \quad \underline{F}_{\beta_{mr}} \quad \underline{F}_{\dot{\zeta}_{mr}} \quad \underline{F}_{\zeta_{mr}}\right]^T$$

$$\underline{F}_{\dot{\beta}_{mr}} = \begin{bmatrix} \dot{\beta}_1^e \\ \vdots \\ \dot{\beta}_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\beta}_1^b \\ \vdots \\ \dot{\beta}_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\beta_{mr}} = \begin{bmatrix} \beta_1^e \\ \vdots \\ \beta_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \beta_1^b \\ \vdots \\ \beta_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\dot{\zeta}_{mr}} = \begin{bmatrix} \dot{\zeta}_1^e \\ \vdots \\ \dot{\zeta}_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\zeta}_1^b \\ \vdots \\ \dot{\zeta}_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\zeta_{mr}} = \begin{bmatrix} \zeta_1^e \\ \vdots \\ \zeta_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \zeta_1^b \\ \vdots \\ \zeta_{nblade_{mr}}^b \end{bmatrix}$$

And similarly,

$$\underline{F}_{blade_{tr}} = \left[\underline{F}_{\dot{\beta}_{tr}} \quad \underline{F}_{\beta_{tr}} \quad \underline{F}_{\dot{\zeta}_{tr}} \quad \underline{F}_{\zeta_{tr}}\right]^T$$

$$\underline{F}_{\dot{\beta}_{tr}} = \begin{bmatrix} \dot{\beta}_1^e \\ \vdots \\ \dot{\beta}_{nblade_{tr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\beta}_1^b \\ \vdots \\ \dot{\beta}_{nblade_{tr}}^b \end{bmatrix}$$

$$\underline{F}_{\beta_{tr}} = \begin{bmatrix} \beta_1^e \\ \vdots \\ \beta_{nblade_{tr}}^e \end{bmatrix} - P \begin{bmatrix} \beta_1^b \\ \vdots \\ \beta_{nblade_{tr}}^b \end{bmatrix}$$

$$\underline{F}_{\dot{\zeta}_{tr}} = \begin{bmatrix} \dot{\zeta}_1^e \\ \vdots \\ \dot{\zeta}_{nblade_{tr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\zeta}_1^b \\ \vdots \\ \dot{\zeta}_{nblade_{tr}}^b \end{bmatrix}$$

$$\underline{F}_{\zeta_{tr}} = \begin{bmatrix} \zeta_1^e \\ \vdots \\ \zeta_{nblade_{tr}}^e \end{bmatrix} - P \begin{bmatrix} \zeta_1^b \\ \vdots \\ \zeta_{nblade_{tr}}^b \end{bmatrix}$$

Where the permutation matrix, P, is in this case, the identity matrix.

This has been implemented into the RASCAL model and parallelised. There were however two problems with associated with this method both due to the relatively long forcing period being considered. Firstly, the rigid body modes are able to develop significantly - this problem was encountered with the original RASCAL trimmer and was overcome by suppressing the integration of the flight states, and secondly, in constructing the rows of the Jacobian matrix that pertain to the wake and blade states. The problem here is that the values at the end of period tend to decay to the mean value for the particular flight and control states. This ensures that these elements in the Jacobian matrix do not contain the appropriate information and convergence to a solution is not achievable.

This can be described mathematically by considering the construction of an element of the Jacobian matrix.

$$j_{ij} = \frac{\partial F_i}{\partial u_j}$$

Using central differencing techniques the element is calculated numerically from:

$$j_{ij} = \frac{F_i(u_j + \delta u_j) - F_i(u_j - \delta u_j)}{2\delta u_j}$$

Consider the case for i not equal to j

$$F_i(u_j + \delta u_j) = y_i^e(u_j + \delta u_j) - y_i^b$$

$$F_i(u_j - \delta u_j) = y_i^e(u_j - \delta u_j) - y_i^b$$

But as the transients will have decayed almost entirely over the relatively long forcing period.

$$y_i^e(u_j + \delta u_j) \cong y_i^e(u_j - \delta u_j) \cong y_i^e(u_j)$$

So

$$j_{ij} \cong 0$$

Consider the case for i=j

$$F_i(u_i + \delta u_i) = y_i^e(u_i + \delta u_i) - (y_i^b + \delta u_i)$$

$$F_i(u_i - \delta u_i) = y_i^e(u_i - \delta u_i) - (y_i^b - \delta u_i)$$

So

$$j_{ij} \cong -1$$

Therefore, when the forcing period is sufficiently long as to allow the transient wake and blade states to decay substantially as shown in figure 4 for a range of perturbation sizes, this implementation is not appropriate. In order for the method to work a shorter forcing period is required. A shorter forcing period would also potentially solve the problem of the development of the rigid body modes.

For many rotorcraft configurations, and in simulations in which only one rotor is modelled as a finite number of individual blades, a true shorter forcing period is present. In the RASCAL simulation in which high fidelity individual blade/blade element models are used in both rotor systems, configurations with non-identical rotor systems will have no true shorter period. In the case of the main and tail rotor helicopter in which the magnitude of the forcing of the tail rotor is typically small when compared to that of the main rotor and that the amplitude of the forcing of the tail rotor is relatively small, particularly at low speed, it would seem reasonable that the periodic forcing of the tail rotor could be neglected and that the method could be implemented purely considering the periodic forcing from the main rotor. This period may be described as 1/(number of main rotor blades) as the blade flap and lag states will map on to each other via an appropriate permutation matrix [3]. The control vector of this implementation is of the form:

$$\underline{u} = \begin{bmatrix} \underline{u}_{con} & \underline{u}_{fs} & \underline{u}_{wake_{mr}} & \underline{u}_{blade_{mr}} \end{bmatrix}^T$$

Where

$$\underline{u}_{con} = \begin{bmatrix} \theta_{0_{mr}} & \theta_{1s_{mr}} & \theta_{1c_{mr}} & \theta_{0_{tr}} \end{bmatrix}^T$$

$$\underline{u}_{fs} = \begin{bmatrix} u & v & w & p & q & r & \phi_f & \theta_f \end{bmatrix}^T$$

$$\underline{u}_{wake_{mr}} = \begin{bmatrix} v_{i0_{mr}} & v_{1s_{mr}} & v_{1c_{mr}} \end{bmatrix}^T$$

$$\underline{u}_{blade_{mr}} = \begin{bmatrix} \underline{\dot{\beta}}_{mr} & \underline{\beta}_{mr} & \underline{\dot{\zeta}}_{mr} & \underline{\zeta}_{mr} \end{bmatrix}^T$$

Where

$$\underline{\dot{\beta}}_{mr} = \begin{bmatrix} \dot{\beta}_1 & .. & \dot{\beta}_{nblade_{mr}} \end{bmatrix}^T$$

$$\underline{\beta}_{mr} = \begin{bmatrix} \beta_1 & .. & \beta_{nblade_{mr}} \end{bmatrix}^T$$

$$\underline{\dot{\zeta}}_{mr} = \begin{bmatrix} \dot{\zeta}_1 & .. & \dot{\zeta}_{nblade_{mr}} \end{bmatrix}^T$$

$$\underline{\zeta}_{mr} = \begin{bmatrix} \zeta_1 & .. & \zeta_{nblade_{mr}} \end{bmatrix}^T$$

The error vector in the Newton-Raphson scheme now takes the form.

$$\underline{F} = \begin{bmatrix} \underline{F}_{mfs} & \underline{F}_{fs} & \underline{F}_{wake_{mr}} & \underline{F}_{blade_{mr}} \end{bmatrix}^T$$

Where

$$\underline{F}_{mfs} = \begin{bmatrix} \overline{u}_e - \overline{u}_{e_{des}} & \overline{v}_e - \overline{v}_{e_{des}} & \overline{w}_e - \overline{w}_{e_{des}} & \overline{turn} - \overline{turn}_{des} \end{bmatrix}^T$$

$$\underline{F}_{fs} = \begin{bmatrix} u^e - u^b & v^e - v^b & w^e - w^b & p^e - p^b & q^e - q^b & r^e - r^b & \phi_f^e - \phi_f^b & \theta_f^e - \theta_f^b \end{bmatrix}^T$$

$$\underline{F}_{wake_{mr}} = \begin{bmatrix} v_{i0_{mr}}^e - v_{i0_{mr}}^b & v_{1s_{mr}}^e - v_{1s_{mr}}^b & v_{1c_{mr}}^e - v_{1c_{mr}}^b \end{bmatrix}^T$$

$$\underline{F}_{blade_{mr}} = \begin{bmatrix} \underline{F}_{\dot{\beta}_{mr}} & \underline{F}_{\beta_{mr}} & \underline{F}_{\dot{\zeta}_{mr}} & \underline{F}_{\zeta_{mr}} \end{bmatrix}^T$$

$$\underline{F}_{\dot{\beta}_{mr}} = \begin{bmatrix} \dot{\beta}_1^e \\ : \\ \dot{\beta}_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\beta}_1^b \\ : \\ \dot{\beta}_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\beta_{mr}} = \begin{bmatrix} \beta_1^e \\ : \\ \beta_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \beta_1^b \\ : \\ \beta_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\dot{\zeta}_{mr}} = \begin{bmatrix} \dot{\zeta}_1^e \\ : \\ \dot{\zeta}_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \dot{\zeta}_1^b \\ : \\ \dot{\zeta}_{nblade_{mr}}^b \end{bmatrix}$$

$$\underline{F}_{\zeta_{mr}} = \begin{bmatrix} \zeta_1^e \\ : \\ \zeta_{nblade_{mr}}^e \end{bmatrix} - P \begin{bmatrix} \zeta_1^b \\ : \\ \zeta_{nblade_{mr}}^b \end{bmatrix}$$

Where

$$P = \begin{bmatrix} 0 & 1 & : & 0 & 0 \\ 0 & 0 & : & 0 & 0 \\ : & : & : & : & : \\ 0 & 0 & : & 0 & 1 \\ 1 & 0 & : & 0 & 0 \end{bmatrix}$$

The blade states require to be mapped appropriately as only 1/(number of main rotor blades) turns are being used. This is

done via the permutation matrix which is simply the identity matrix shifted one column to the right.

This implementation solves the transient and rigid body mode problems encountered using a long forcing period but introduces a discrepancy as a period shorter than the true forcing period is used. This means that the mean flight states calculated will not true mean flight states over the forcing period and also the tail rotor states are not calculated explicitly - they are only able to decay to their steady-state values.

Results

The trim solutions for an Puma helicopter were found using both the original RASCAL trimmer and the McVicar-Bradley trimmer with a reduced forcing period for a range of forward speeds. The results are presented in Table 1. The McVicar-Bradley trimmer implemented for the full forcing period, which for the PUMA is 6 main rotor turns, failed to converge for each of these cases.

Speed (knots)	Hover A	Hover B	40 A	40 B	80 A	80 B	120 A	120 B
$\theta_{0_{mr}}$	13.998	13.438	11.860	11.658	10.531	10.592	11.207	10.755
$\theta_{1s_{mr}}$	-0.3022	-0.3095	-0.9551	-1.1470	-2.0635	-2.1076	-3.2837	-3.3654
$\theta_{1c_{mr}}$	-0.1577	-0.0818	-0.6135	-0.6033	-1.0956	-1.1188	-1.5318	-1.5291
$\theta_{0_{tr}}$	12.004	8.9352	7.8226	7.0566	3.9424	4.7664	3.3917	1.9611
ϕ_f	5.8341	4.1127	4.1514	3.7791	3.4518	3.6273	4.3838	3.0426
θ_f	5.7992	5.9378	3.8112	4.6128	3.5575	3.6308	2.2728	2.9915
ψ_f	0.0000	0.0001	0.0000	0.0001	0.0000	0.0001	0.0000	0.0000
$\beta_{0_{mr}}$	3.8373	3.6562	3.7780	3.7350	3.7207	3.7505	3.6822	3.6035
$\beta_{1s_{mr}}$	0.0794	0.1660	-0.1950	-0.0124	-0.1315	-0.1299	0.0977	0.1664
$\beta_{1c_{mr}}$	0.5196	0.4722	-0.1521	0.1127	0.2432	0.2640	0.3148	0.6136
$\zeta_{0_{mr}}$	-9.2471	-6.3547	-6.6329	-5.7414	-4.9130	-5.3481	-5.5080	-4.0446
$\zeta_{1s_{mr}}$	-0.0163	-0.0098	0.0213	0.0373	0.0457	0.0449	0.1782	0.1654
$\zeta_{1c_{mr}}$	0.1068	0.0973	-0.0649	-0.0711	0.0256	0.0292	0.0113	0.0723
$v_{i0_{mr}}$	14.300	14.241	8.7363	8.7644	4.1212	4.1225	2.5288	2.4781
$v_{1s_{mr}}$	0.0172	0.0117	0.0875	0.0871	0.0855	0.0851	0.0766	0.0796
$v_{1c_{mr}}$	-0.0530	-0.0532	-0.1158	-0.1343	-0.1317	-0.1339	-0.1392	-0.1387
$\beta_{0_{tr}}$	4.5433	2.8183	3.1657	2.7570	2.2540	2.6696	2.4014	1.4332
$\beta_{1s_{tr}}$	0.0000	0.0026	-0.1757	-0.1146	-0.3623	-0.3819	-0.6932	-0.3878
$\beta_{1c_{tr}}$	0.0000	0.0024	-1.9094	-1.6253	-2.0775	-2.5606	-0.3033	-1.6613
$v_{i0_{tr}}$	20.205	17.242	13.054	12.175	5.0512	6.0030	3.2765	1.9623
$v_{1s_{tr}}$	0.0000	0.0017	0.0754	0.0882	0.0445	0.0645	0.0170	0.0183
$v_{1c_{tr}}$	0.0000	-0.0009	-0.0549	-0.0520	-0.0368	-0.0460	-0.0172	-0.0081
Ω_{mr}	27.563	27.802	27.817	27.871	27.990	27.917	27.930	27.920

Notes: Columns headed A are the original RASCAL trimmer
Columns headed B are the McVicar-Bradley trimmer with partial period.

Table 1

In general, the McVicar-Bradley trimmer with reduced forcing period gives results which are reasonably similar to those obtained from the original RASCAL trimmer. There is however, one exception to this. The McVicar-Bradley trimmer does not make good estimates of tail rotor collective - this will be due to the fact that the period considered does not reflect the mean forcing from the tail rotor. This subsequently affects all the remaining tail rotor states, so the validity of the tail rotor results can be brought into question.

In order to assess the relative 'quality' of the trim states obtained the free response from both trimmers has been calculated and these results are shown in figures 5a-8f.

These results show that the original trimming strategy holds the trim state very well at the full range of forward speeds. The

McVicar-Bradley trimmer on the other hand does not hold the trim state nearly as well in any of the cases considered. As the same tolerance was used in both cases the assumptions used in the original trimmer must be more relevant than those in the McVicar-Bradley method.

Both of the trimming strategies were implemented on a cluster of workstations running the message passing software entitled PVM (Parallel Virtual Machine) [7] and their computational performance was measured. The times given are the CPU times of the longest process and thus reflect the overall time of execution of the job if a suitable number of workstations is available. The results, for the hover case are described for the original RASCAL trimmer and the McVicar-Bradley trimmer in tables 2 and 3 respectively.

	Sequential	Parallelisation 1	Parallelisation 2
niterations	5	5	5
ncontrols	6	6	6
Predicted SpeedUp	-	7.00	11.67
Actual SpeedUp	-	6.83	10.02
Predicted time	-	678.8s	407.3s
Actual time	4751.9s	695.3s	474.2s

Table 2 - Original RASCAL trimmer

	Sequential	Parallelisation 1	Parallelisation 2
niterations	3	3	3
ncontrols	31	31	31
Predicted SpeedUp	-	32.00	48.00
Actual SpeedUp	-	30.94	38.78
Predicted time	-	8.61s	5.74s
Actual time	275.4s	8.9s	7.1s

Table 3 - McVicar-Bradley trimmer

The results highlight the computational intensity of the original trimming strategy. This is because of the relatively large amount of conventional simulation required during each evaluation of the mean accelerations. The McVicar-Bradley trimmer on the other hand requires a substantially shorter period of simulation and this is reflected in the computational performance - even with the larger number of states that require perturbing.

They also show that trimming rotorcraft simulation models is a task that is well suited to parallel computing, particularly the McVicar-Bradley method as it has a high degree of available parallelism so the execution times can be substantially reduced. It is possible to predicted the performance improvement reasonably well by analysing the parallel strategy used. The speed-up achieved is consistently lower than that predicted - this is because the predicted value makes no allowance for the message passing overhead that is present or for the inversion of the Jacobian in order to evaluate the new controls during each iteration.

Conclusions

The following conclusions can be drawn:

1. It is not possible to implement the McVicar-Bradley trimmer in a rotorcraft simulation model in which both rotors are modelled as individual blades and are not identical, as the forcing period is too long.

2. It is possible to neglect the periodicity of the forcing of the tail rotor and to solve the trim state over a 'partial' forcing period. This implementation displays results that are reasonable when compared to the original RASCAL trimmer but is unable to hold the trim state well in free response from trim.

3. Parallel implementation displays useful computational performance improvement in both implementations. It is however, most useful in the McVicar-Bradley trimmer where a high degree of parallelism exists.

References

1. Houston, S.S., 'Rotorcraft Aeromechanics Simulation for Control Analysis - Mathematical Model Description', Internal Report No. 9123 (1991)

2. Houston, S.S., 'Validation of a Non-linear Individual Blade Rotorcraft Flight Dynamics Model using a Perturbation Method', The Aeronautical Journal of the Aeronautical Society (1994)

3.McVicar, J.S.G., Bradley, R., 'A Generic Tilt-Rotor Simulation Model with Parallel Implementation and Partial Periodic Trim Algorithm', Proceedings of the Eighteenth European Rotorcraft Forum, Avignon, France (1992)

4. McVicar, J.S.G., 'A Generic Tilt-Rotor Simulation Model with Parallel Implementation', Ph.D. Thesis (1993)

5. Ewing, D.H., 'The Viability of Parallel Processing Technology for use in Rotorcraft Simulation', Internal Report No. 9504 (1995)

6. Rutherford, S., Thomson, D.G., 'development of a Generic Inverse Simulation Algorithm', Internal Report No. 9410 (1994)

7. Geist, A., et al, 'PVM: Parallel Virtual Machine', MIT Press (1994)

Figures

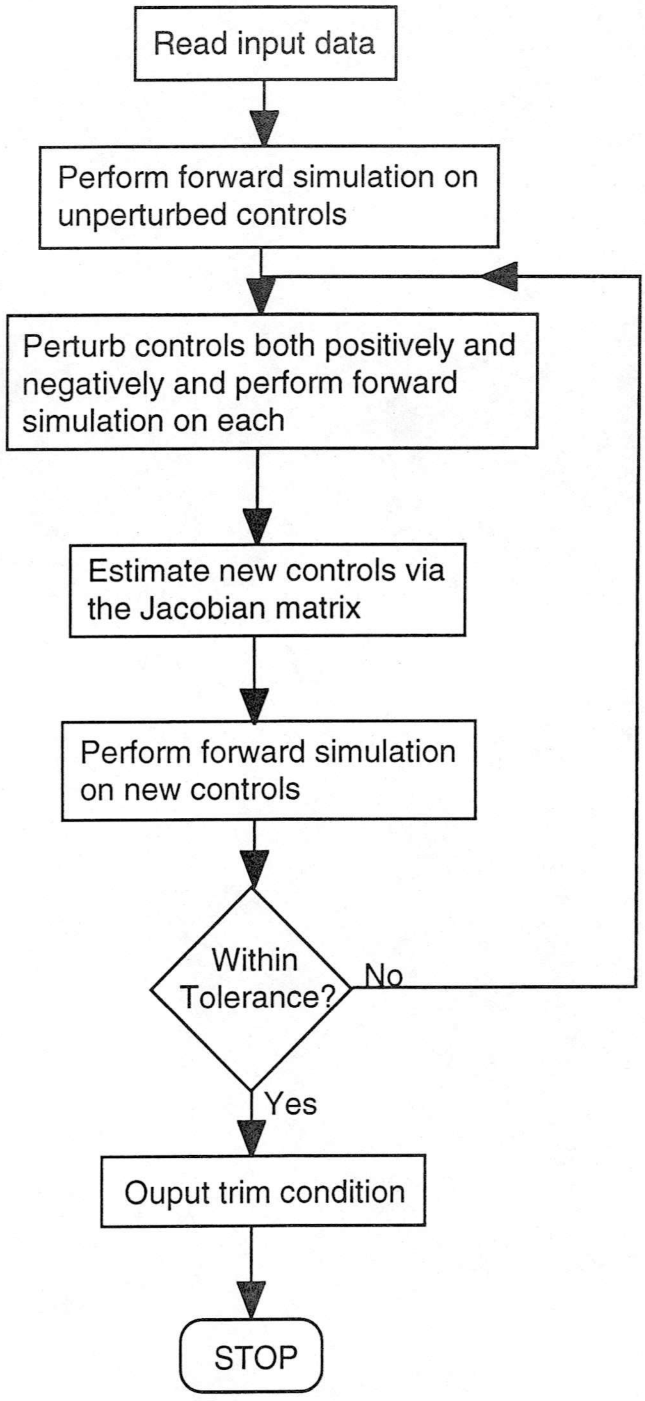

Figure 1 - Newton-Raphson Iteration Sequential

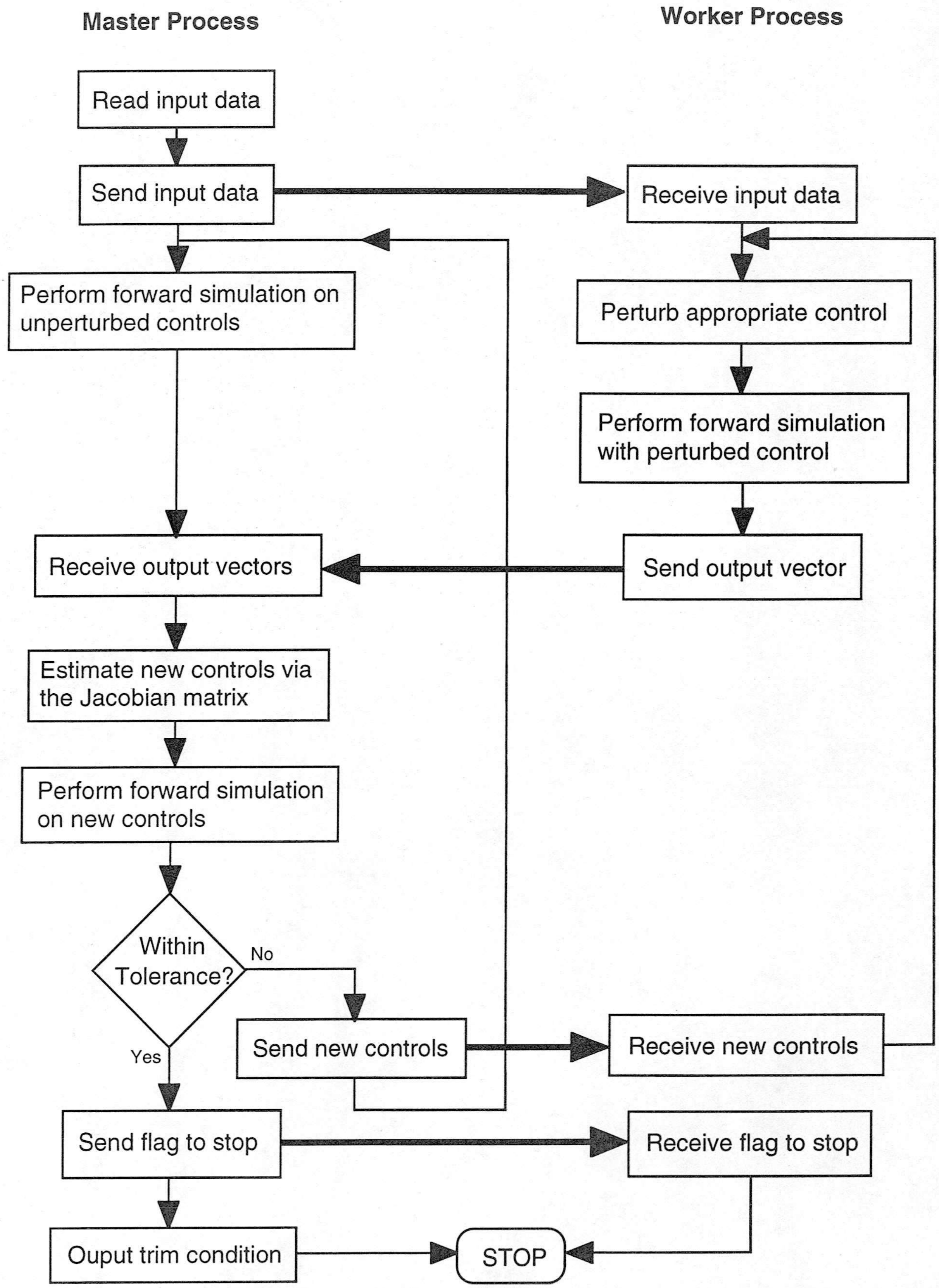

Figure 2 - Trim Algorithm Parallelisation 1

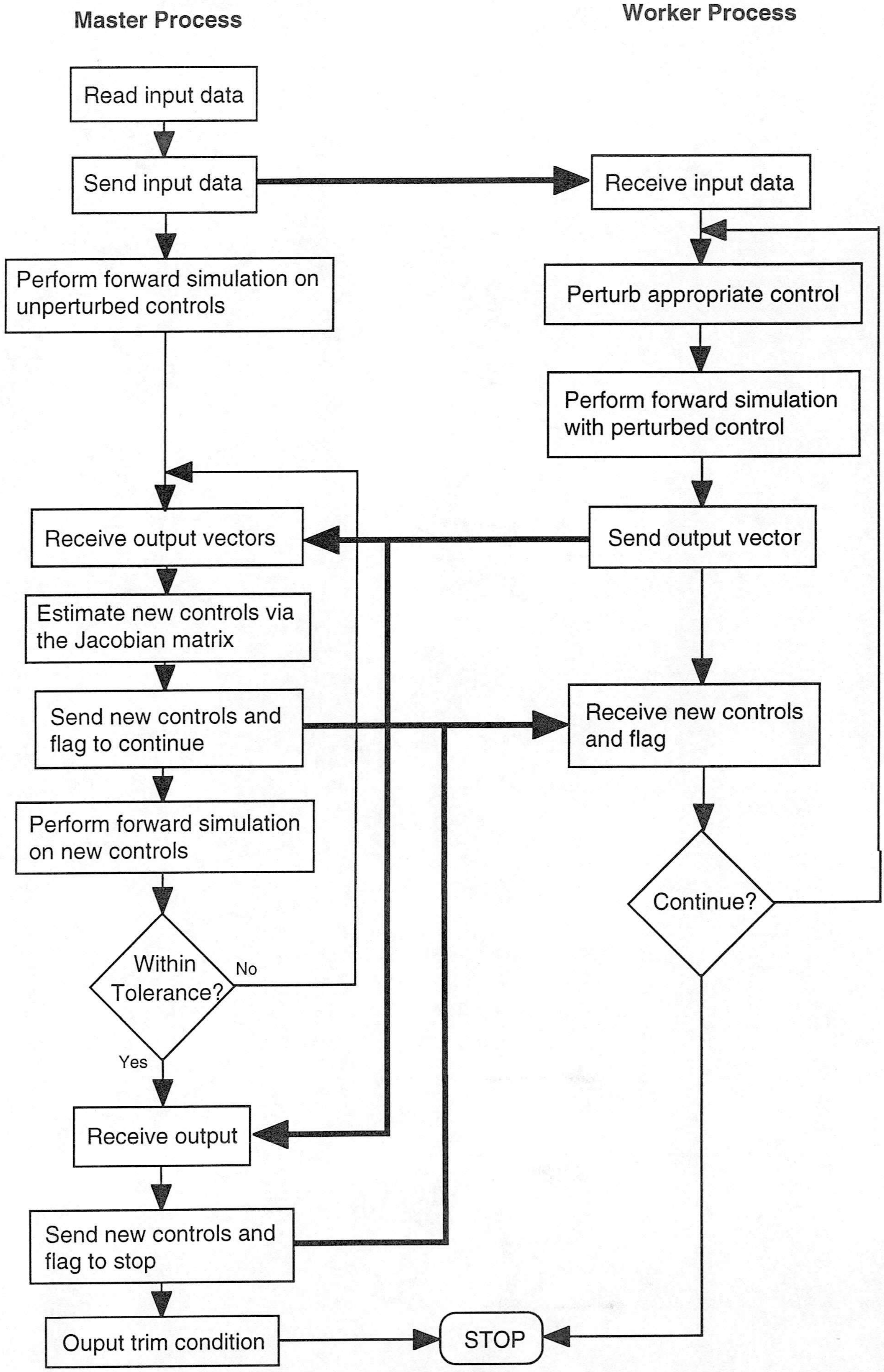

Figure 3 - Trim Algorithm Parallelisation 2

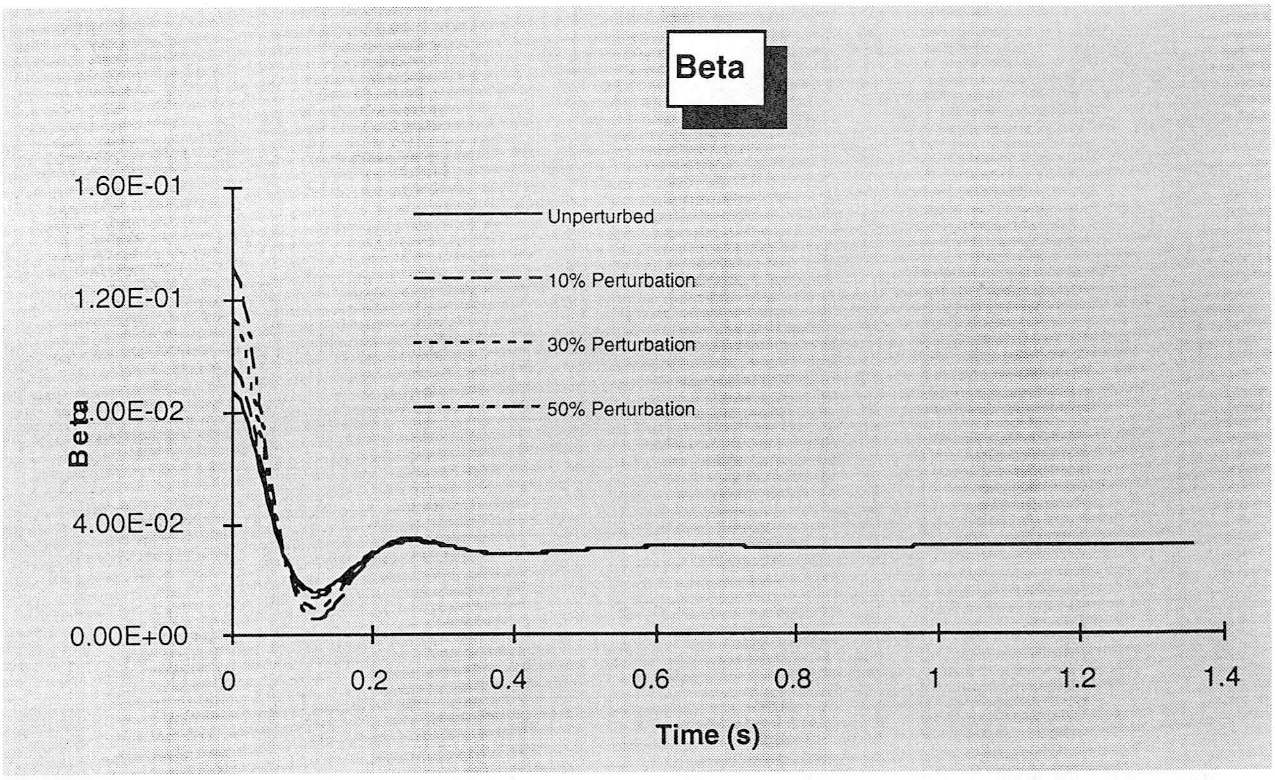

Figure 4a

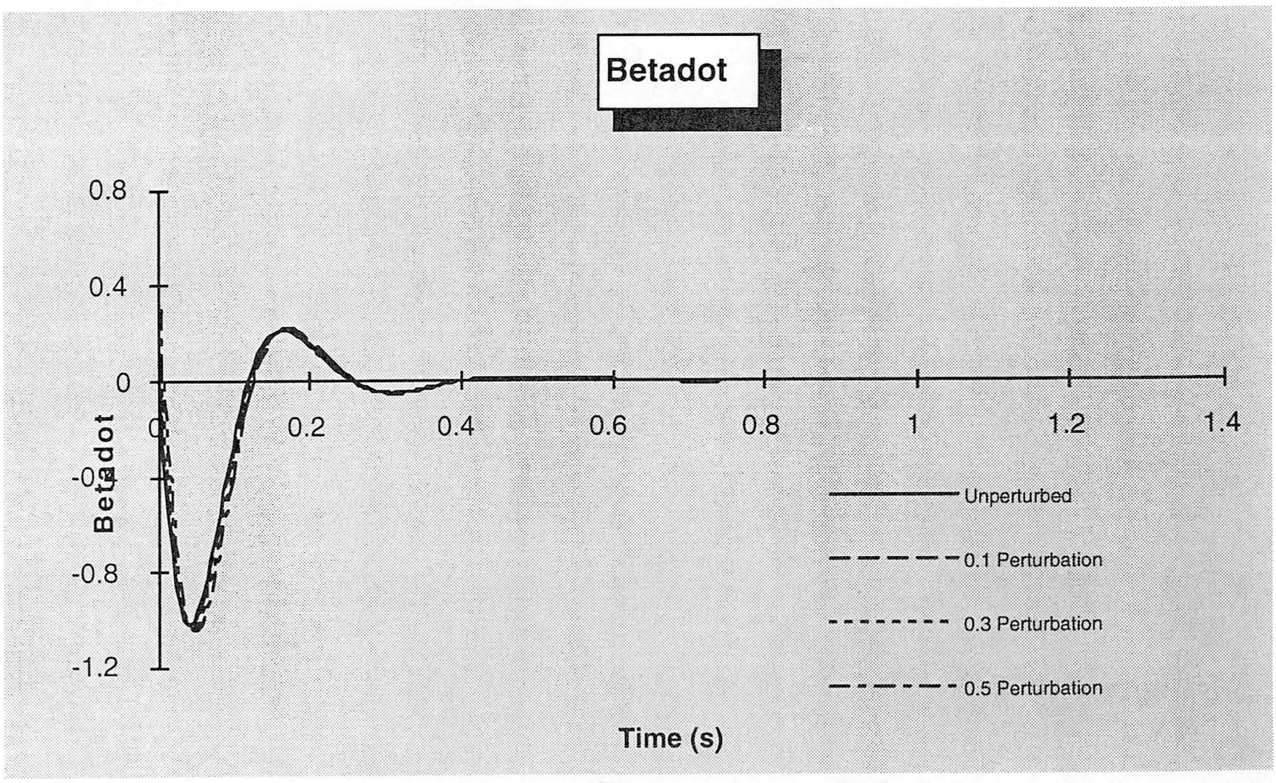

Figure 4b

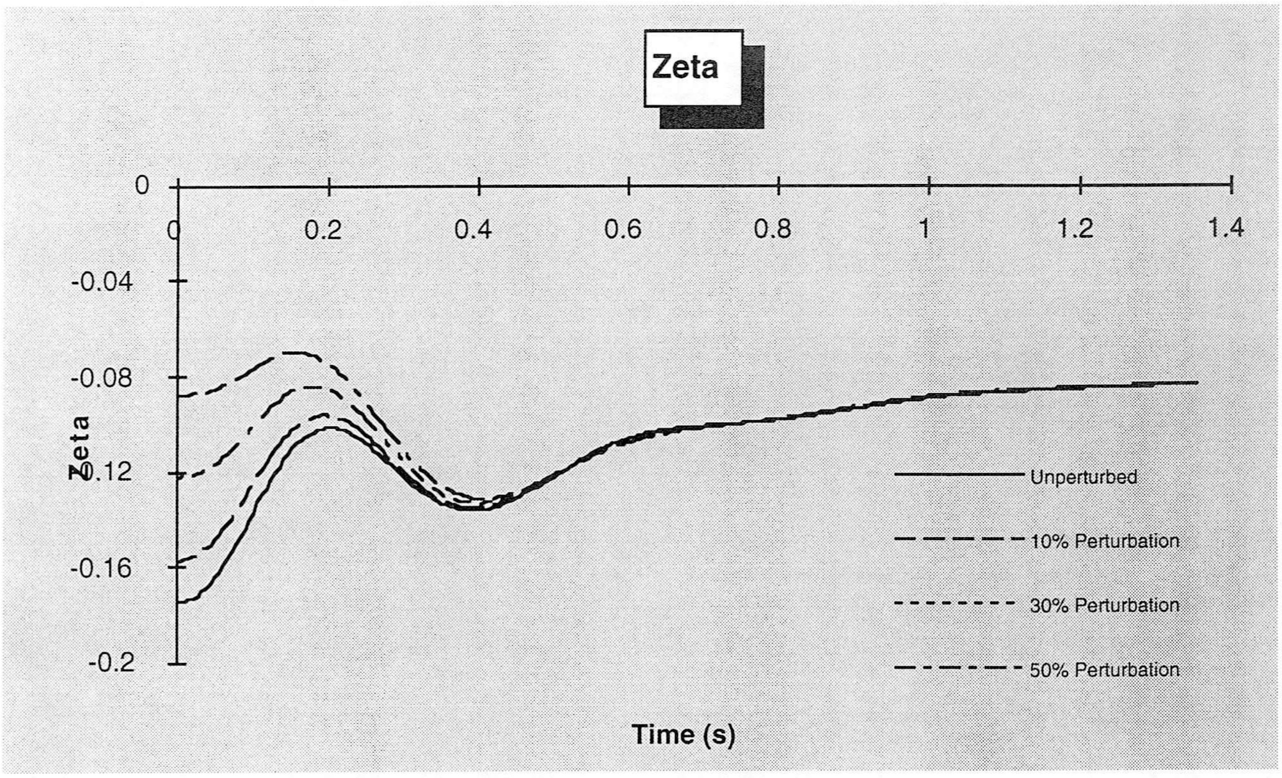

Figure 4c

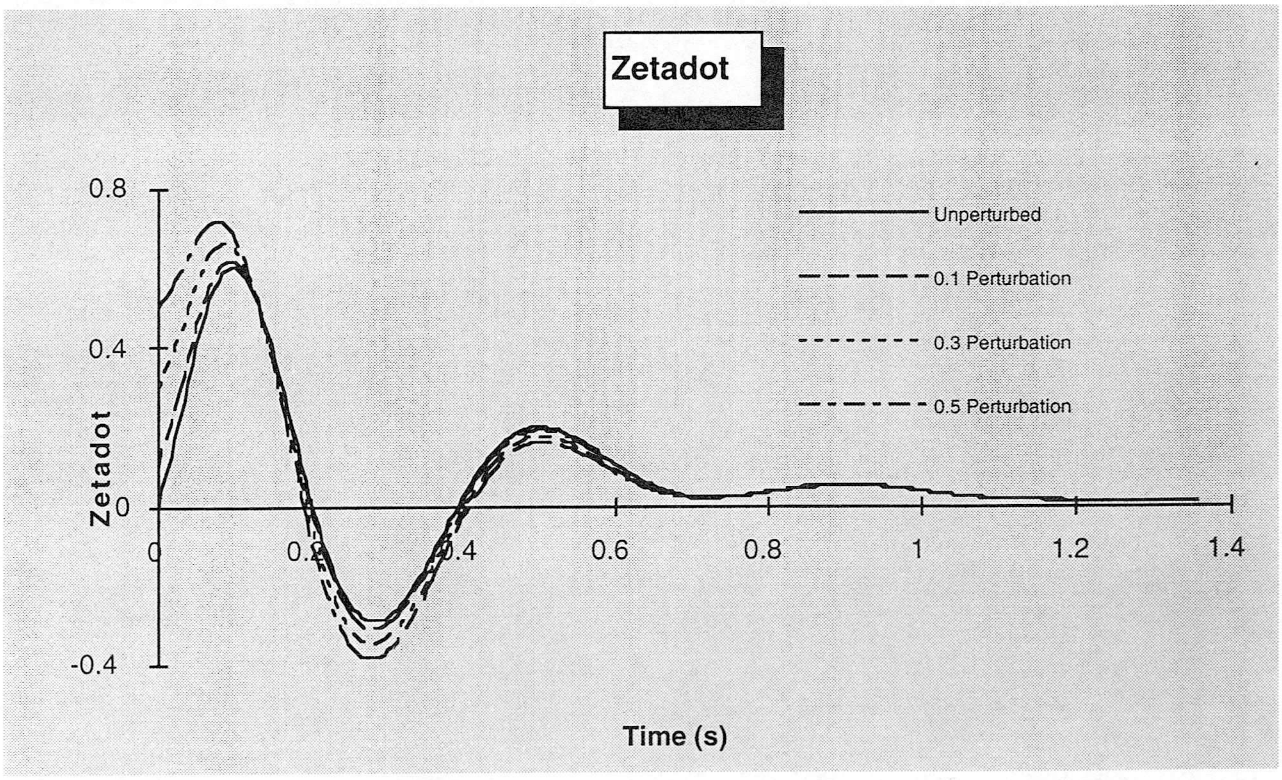

Figure 4d

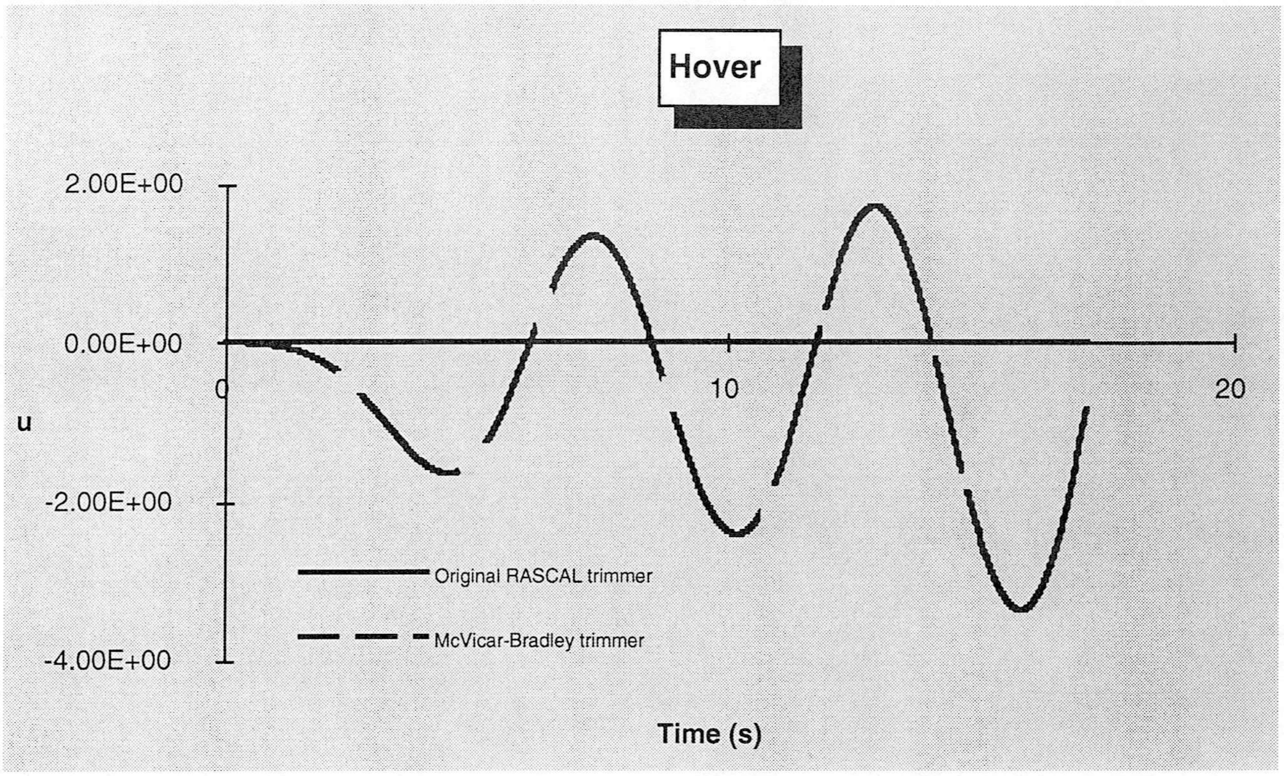

Figure 5a

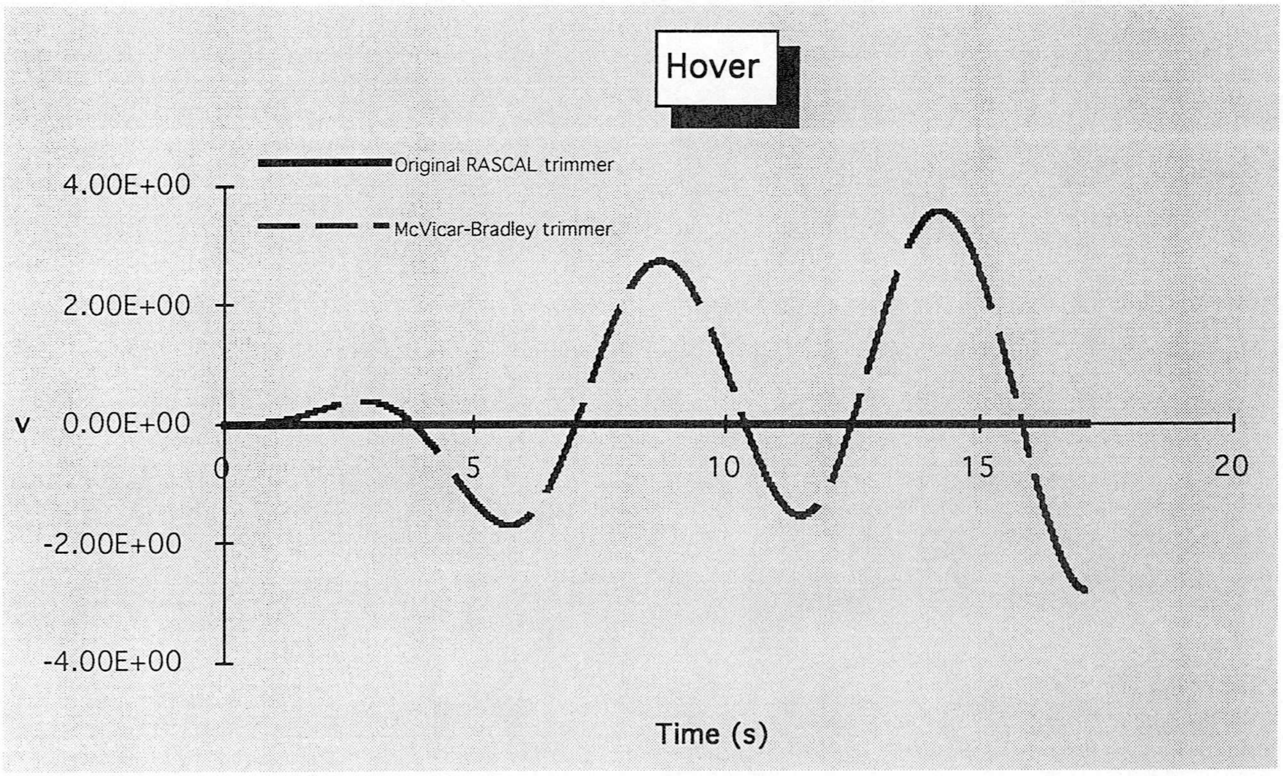

Figure 5b

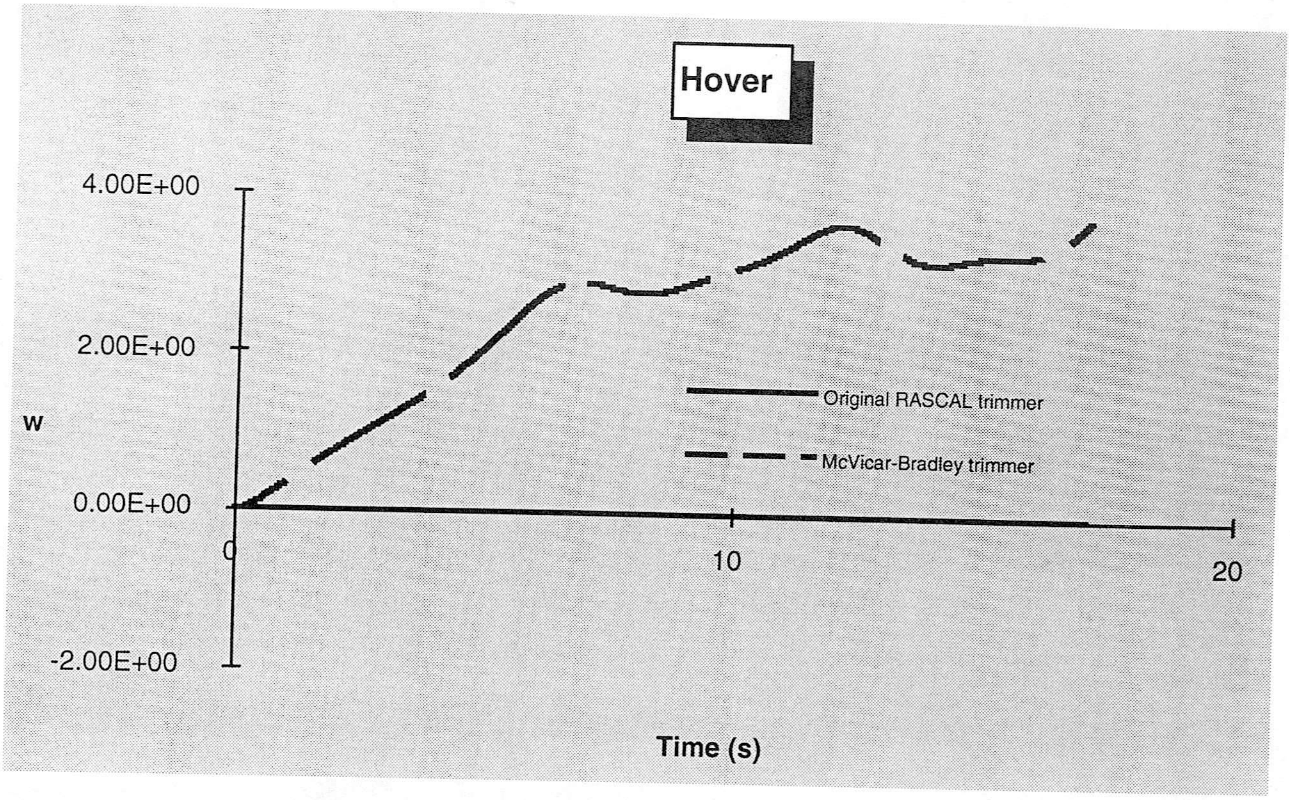

Figure 5c

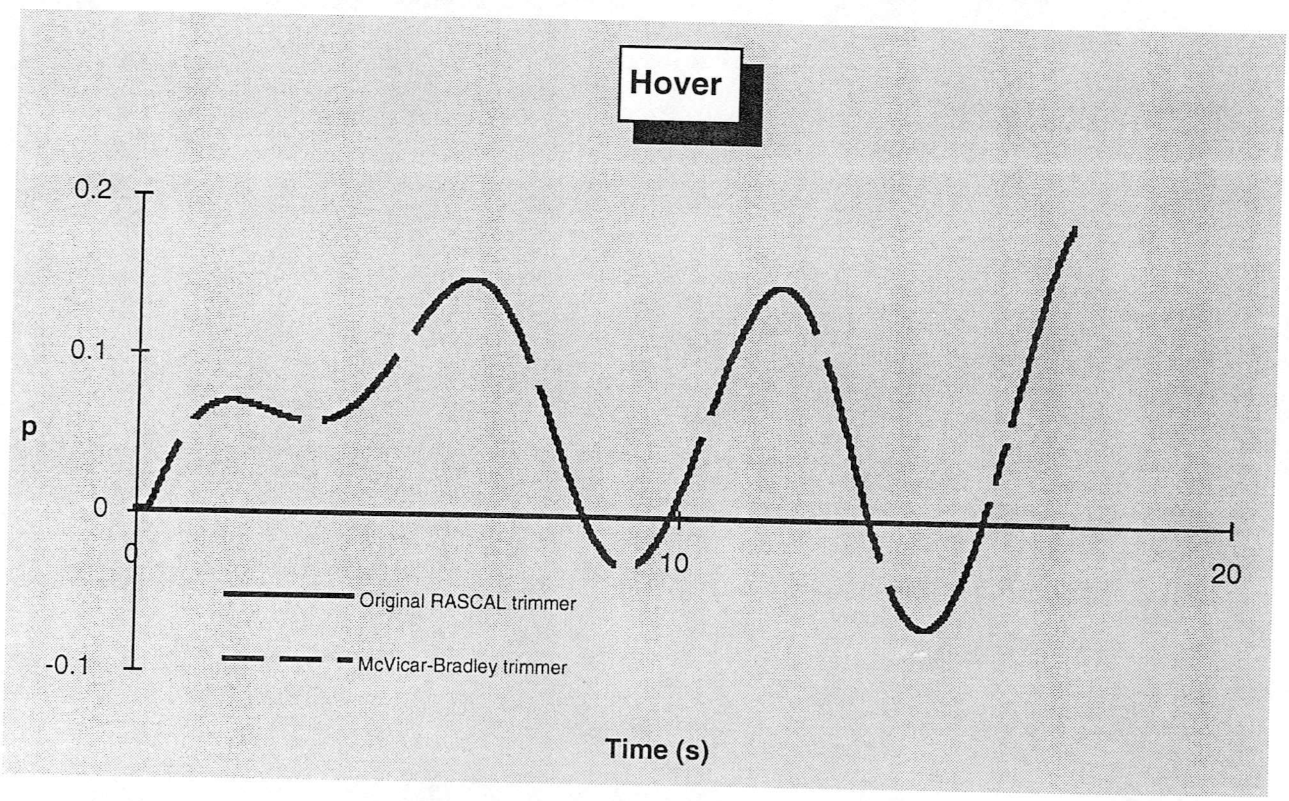

Figure 5d

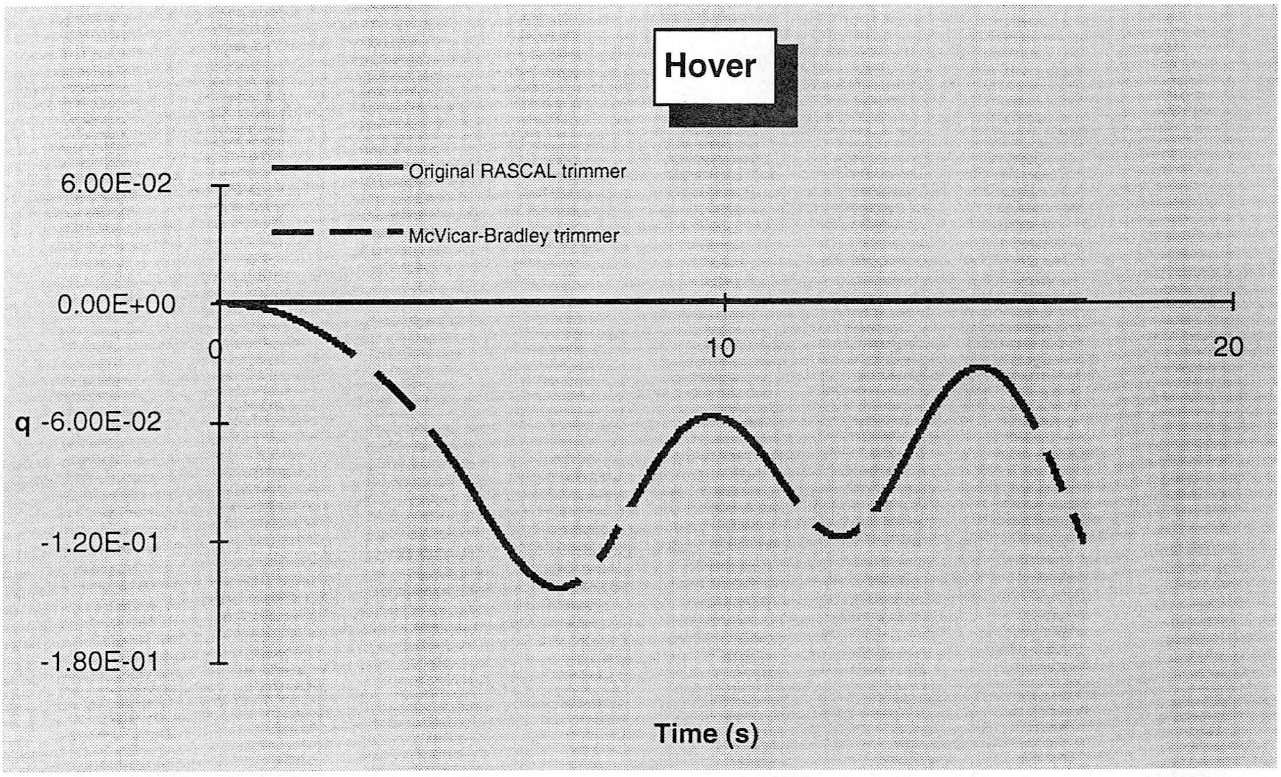

Figure 5e

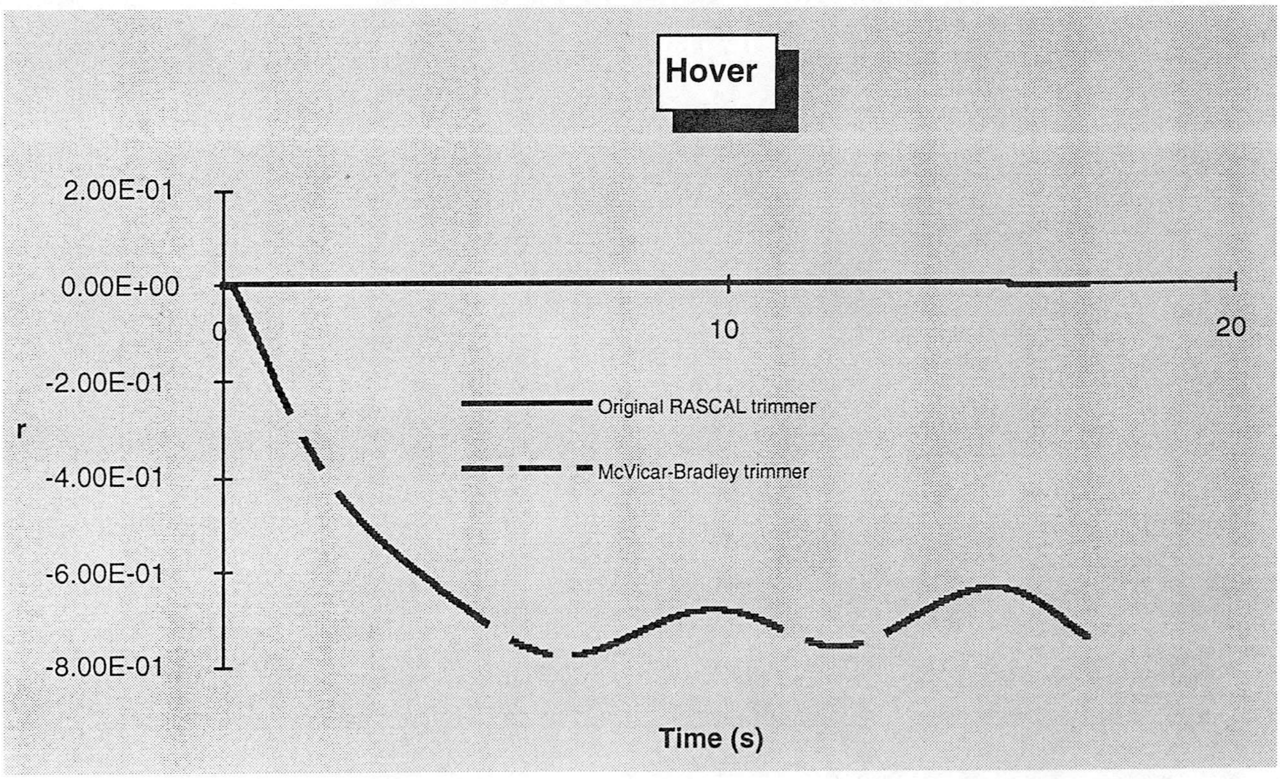

Figure 5f

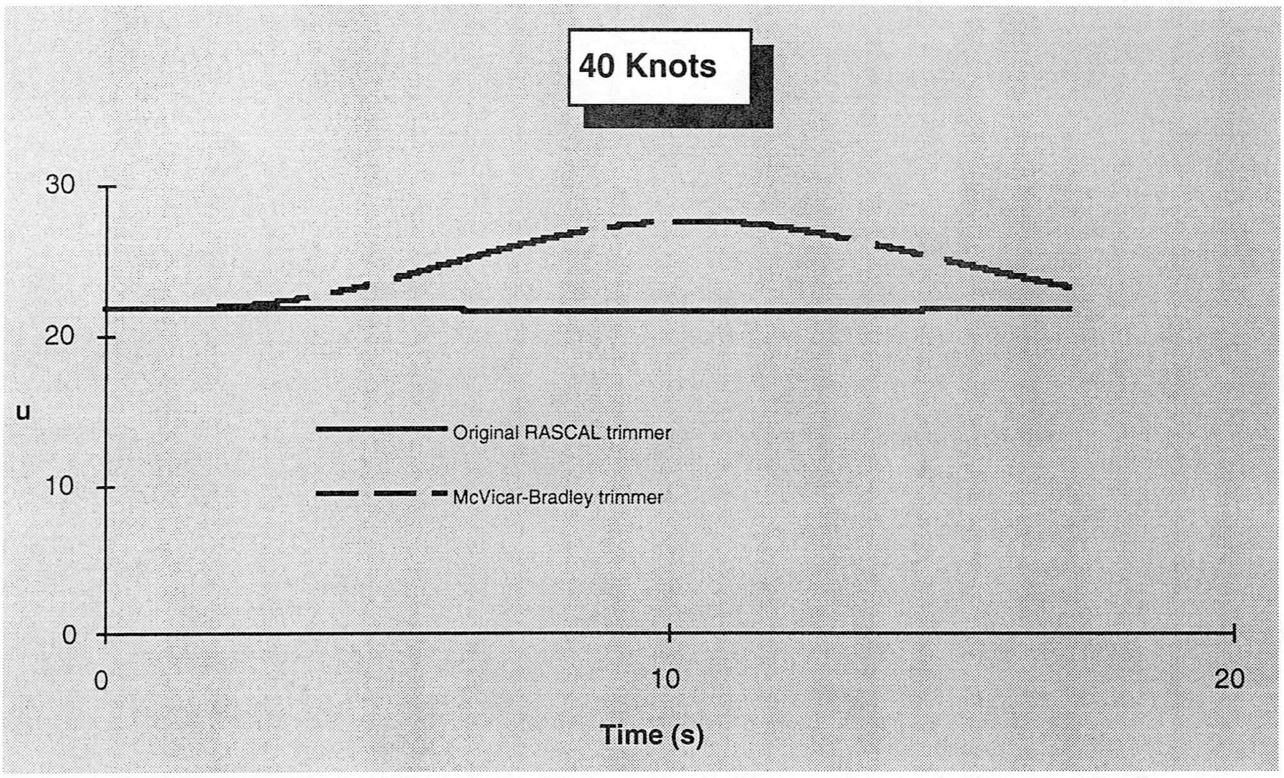

Figure 6a

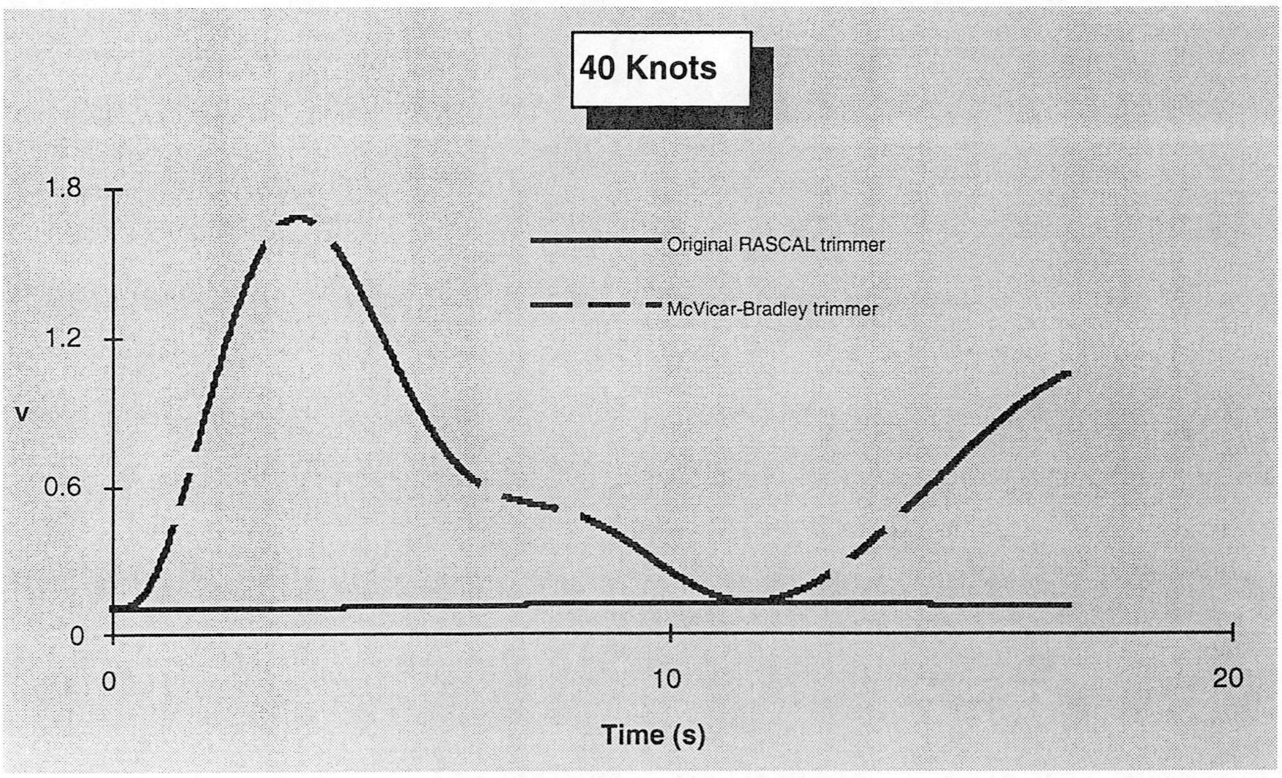

Figure 6b

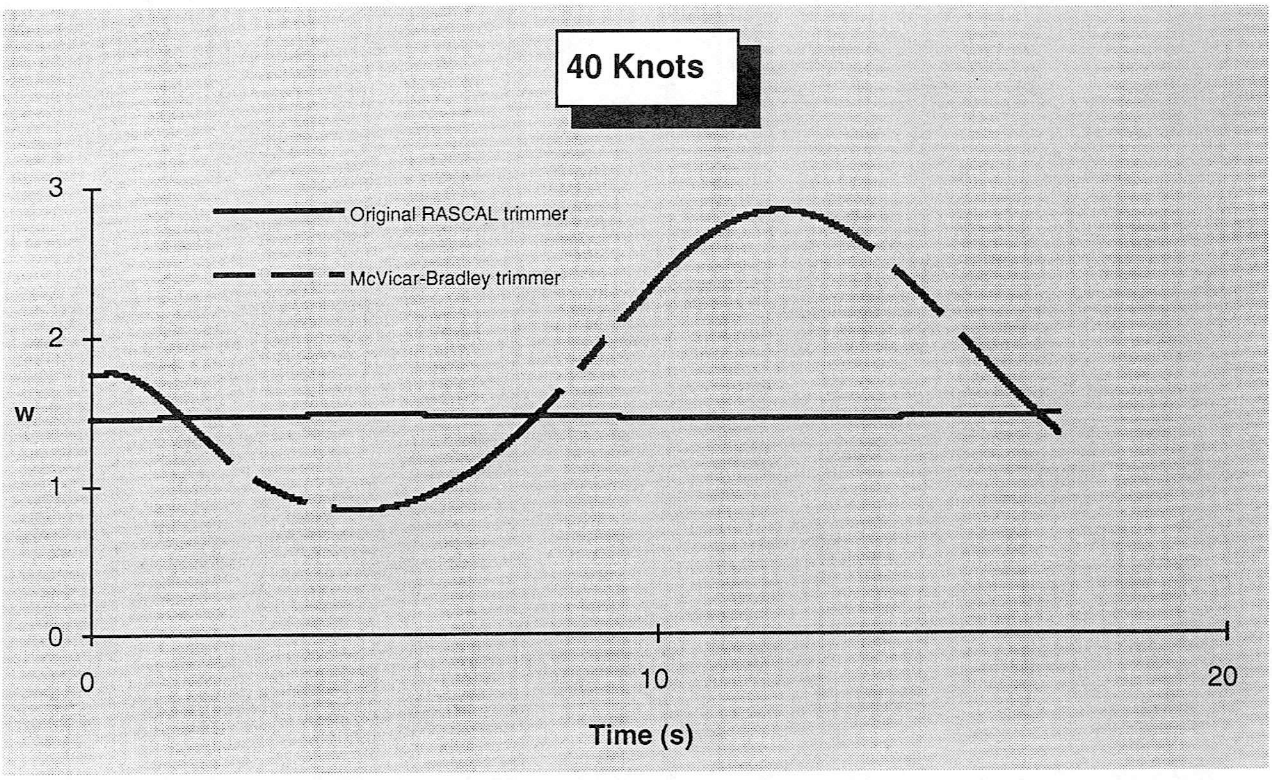

Figure 6c

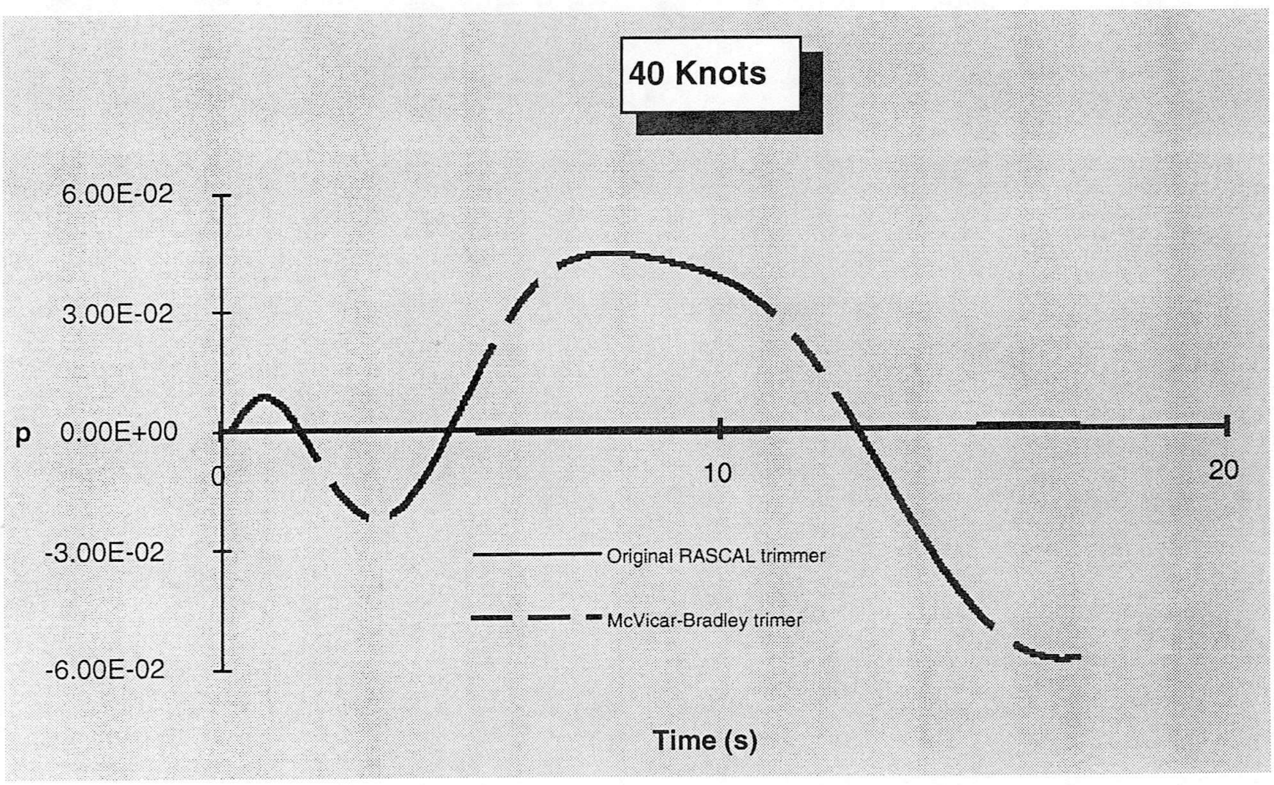

Figure 6d

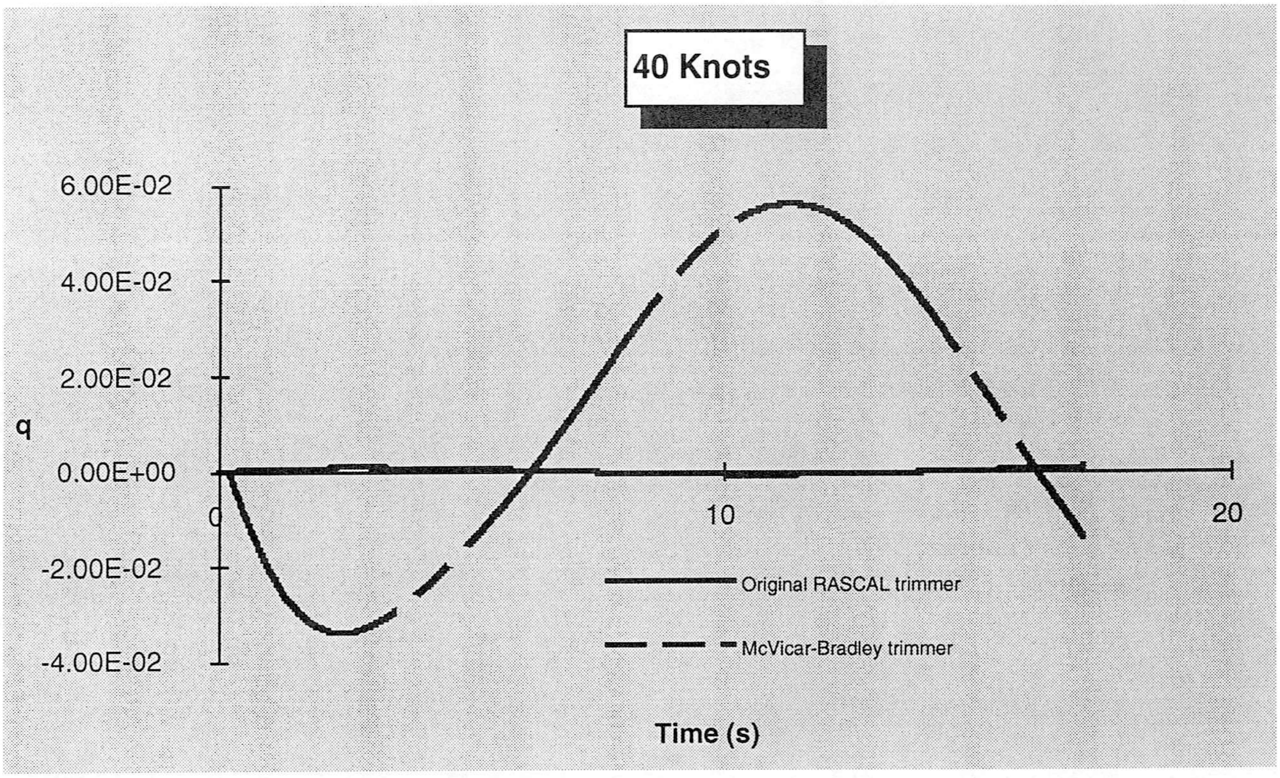

Figure 6e

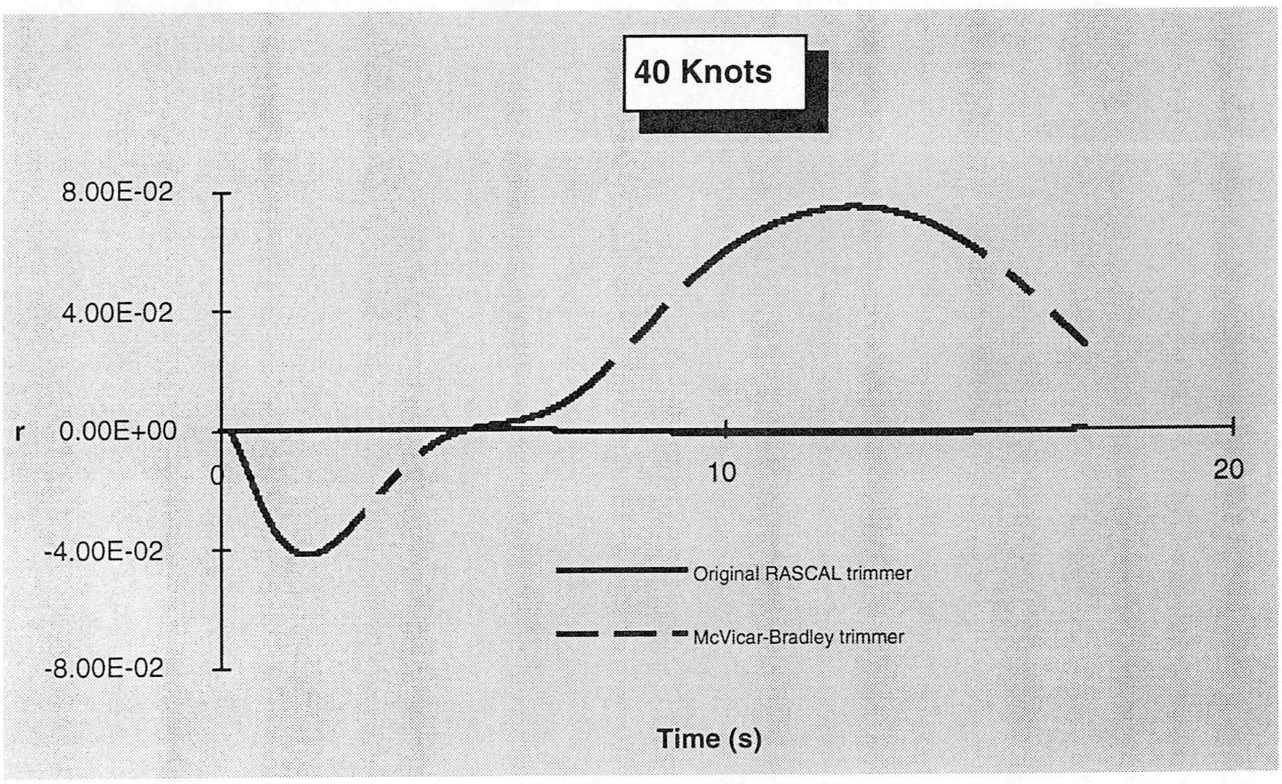

Figure 6f

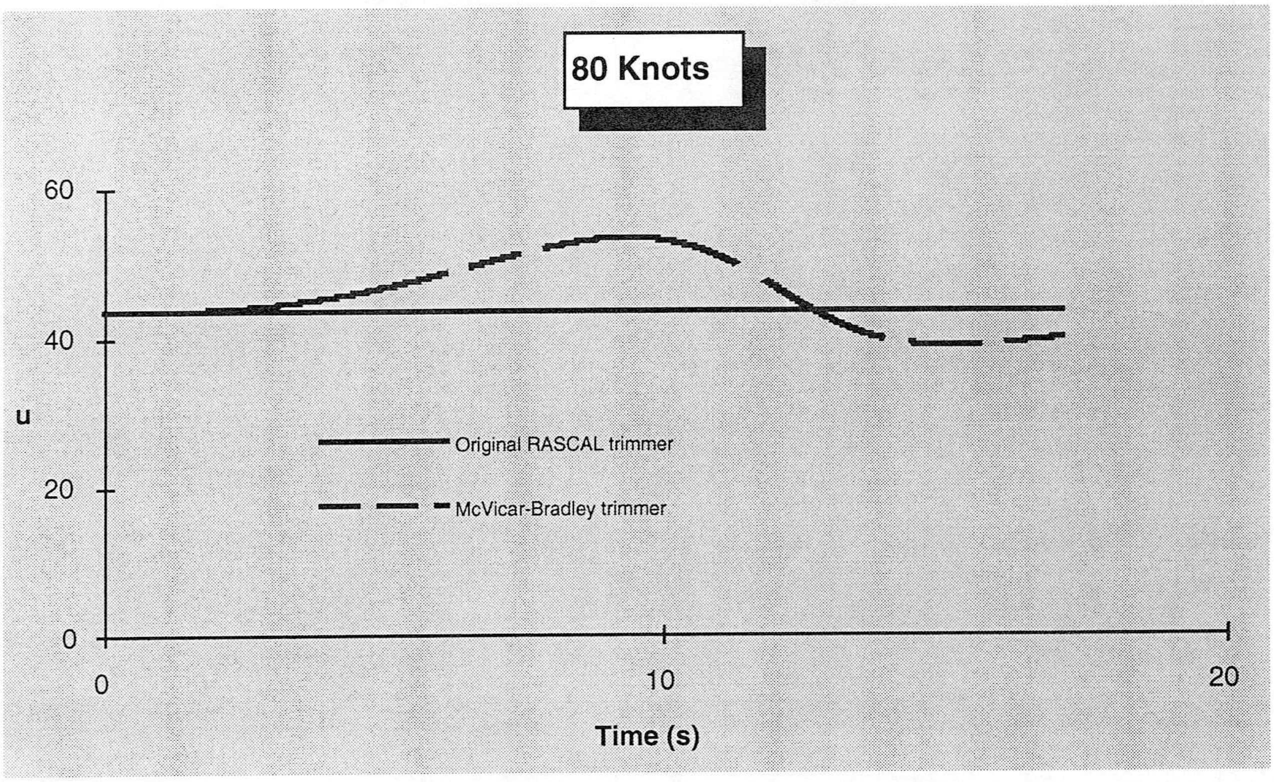

Figure 7a

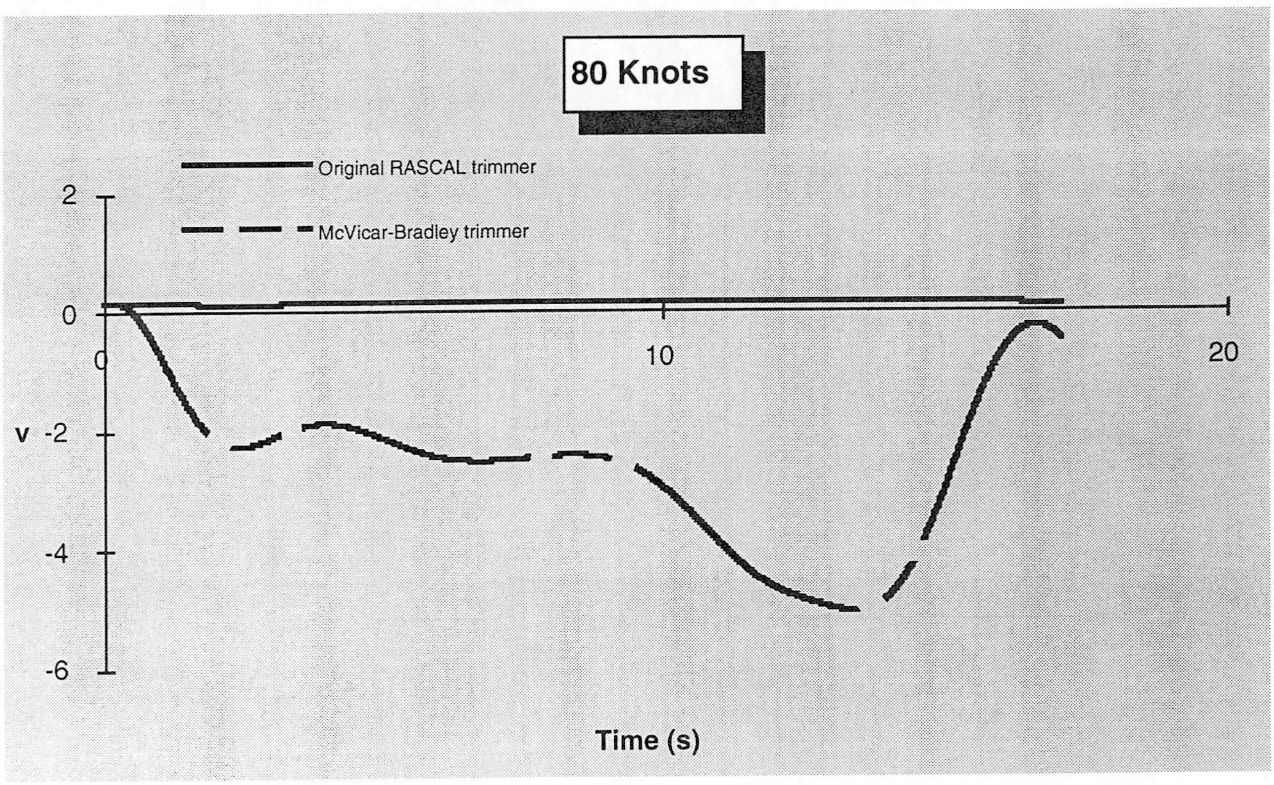

Figure 7b

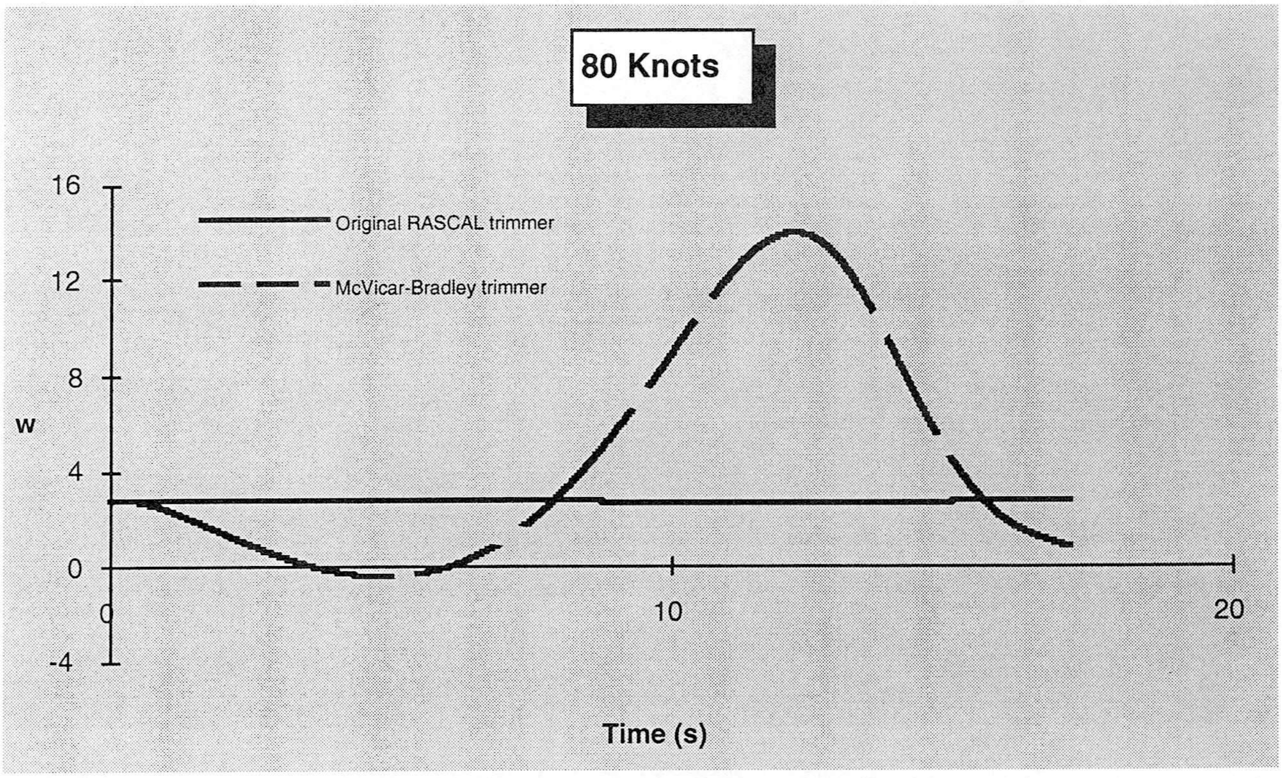

Figure 7c

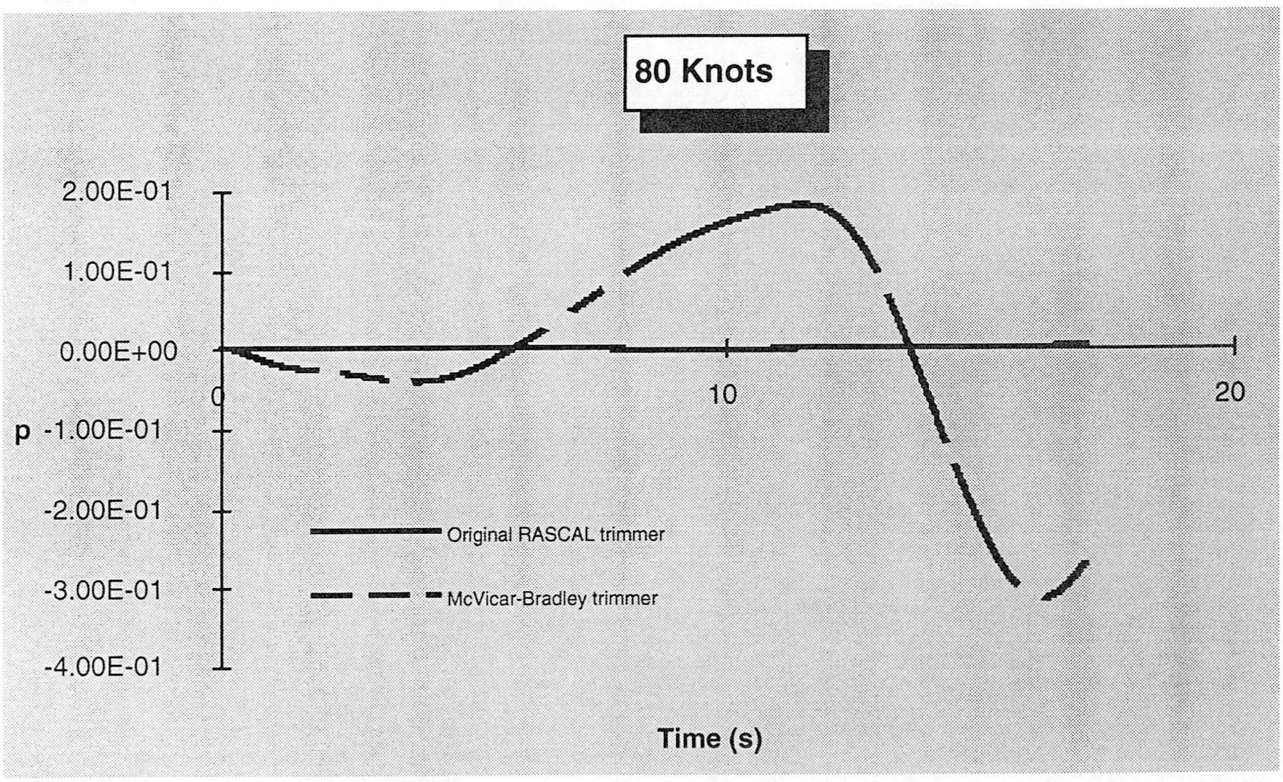

Figure 7d

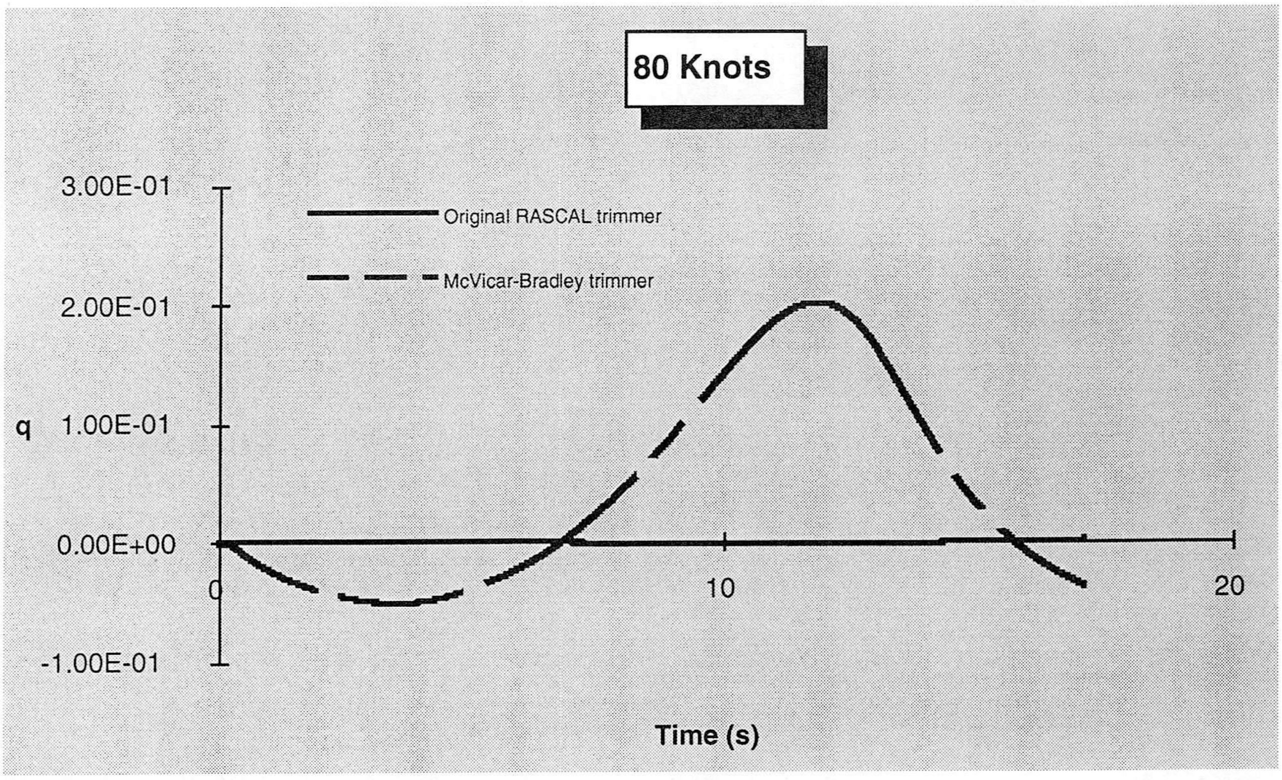

Figure 7e

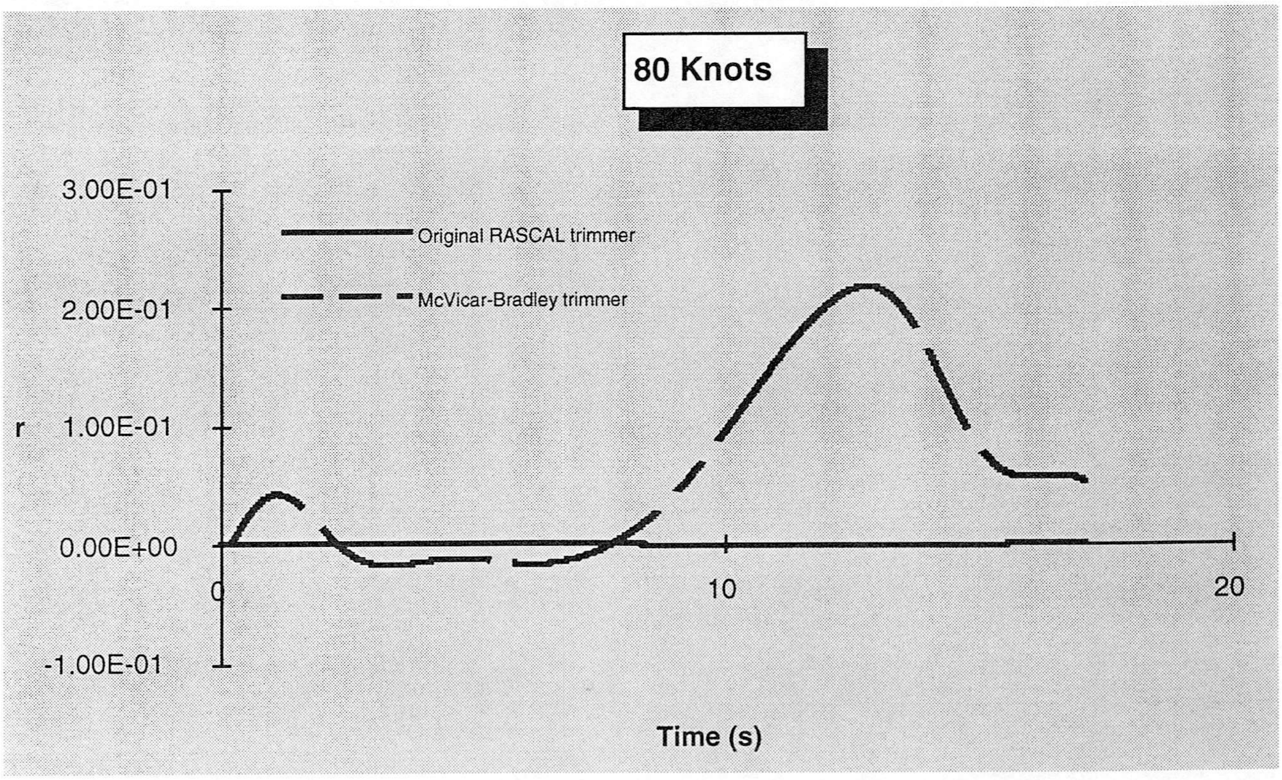

Figure 7f

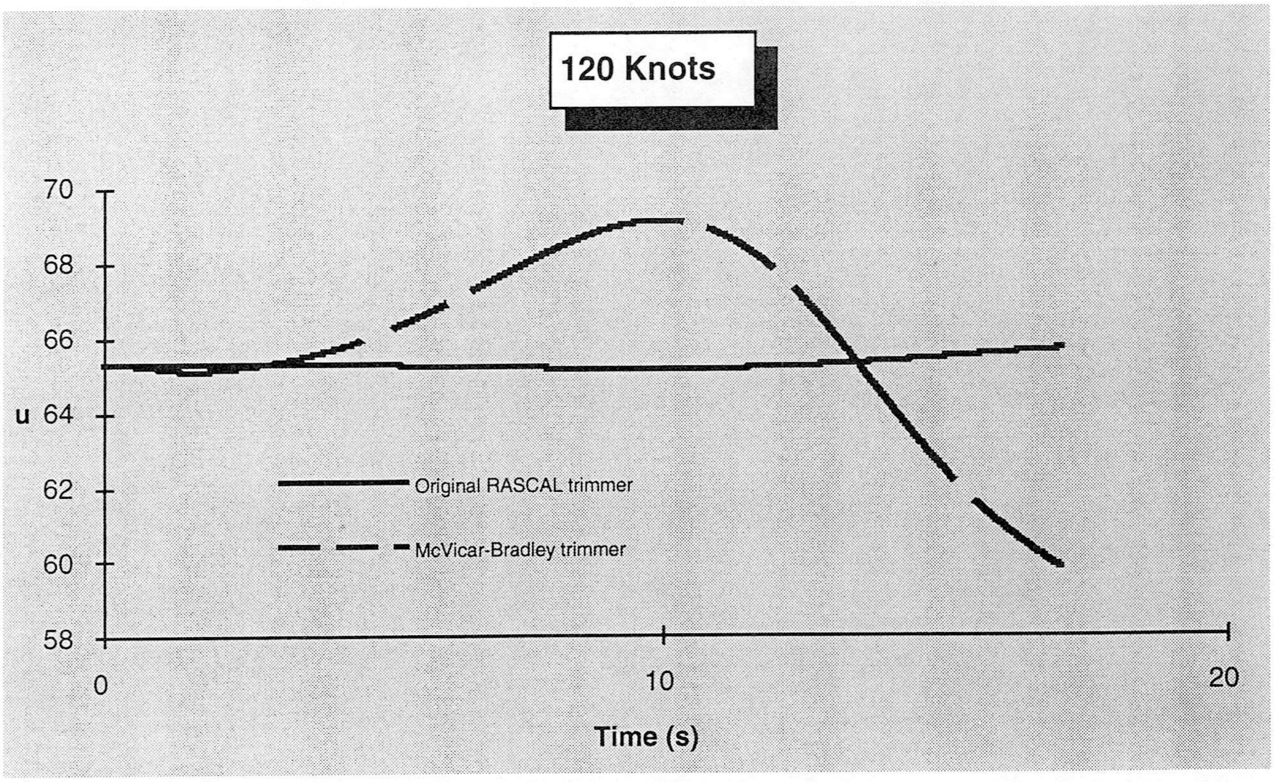

Figure 8a

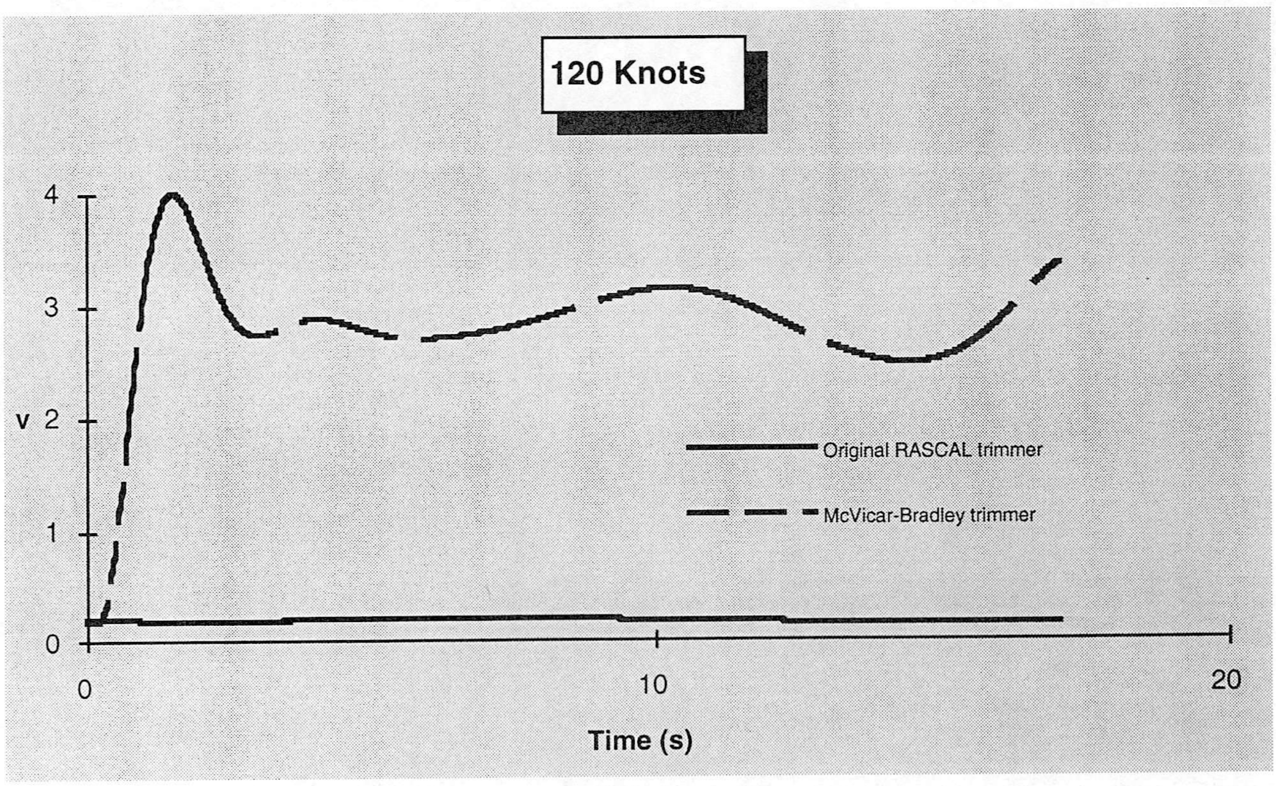

Figure 8b

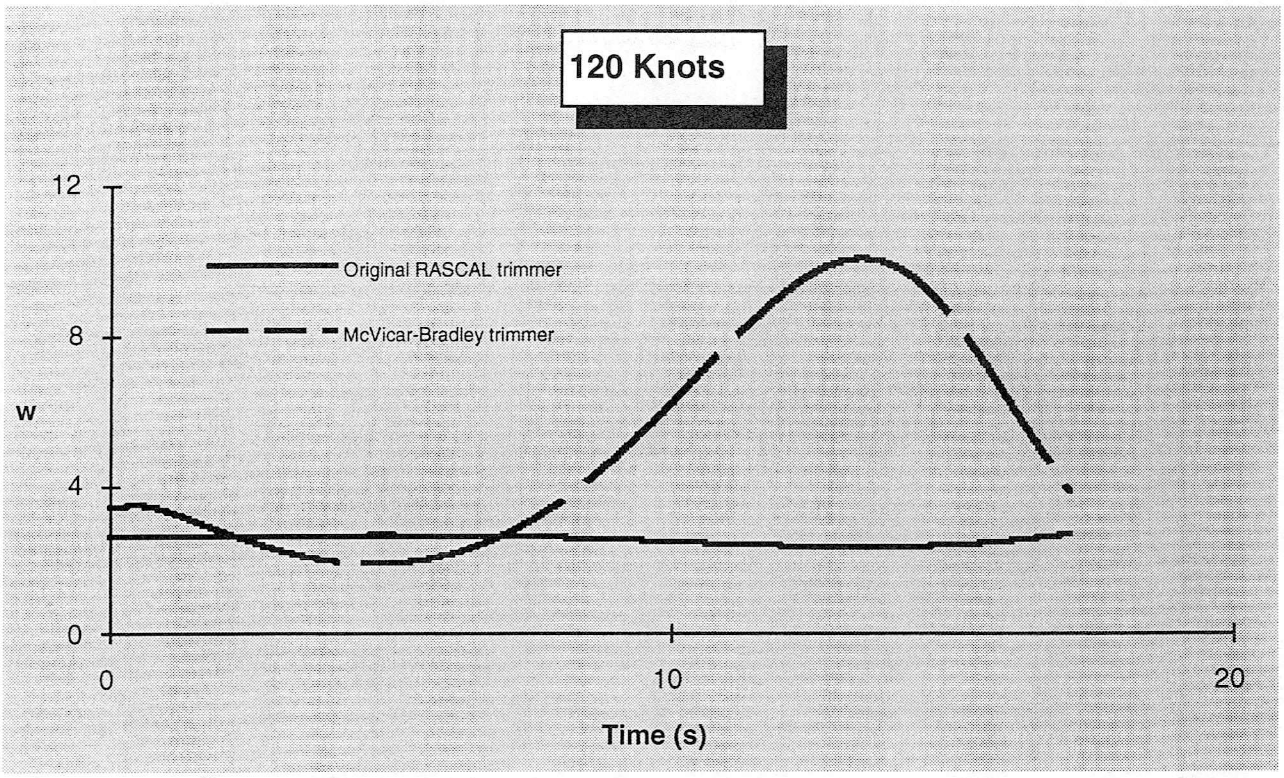

Figure 8c

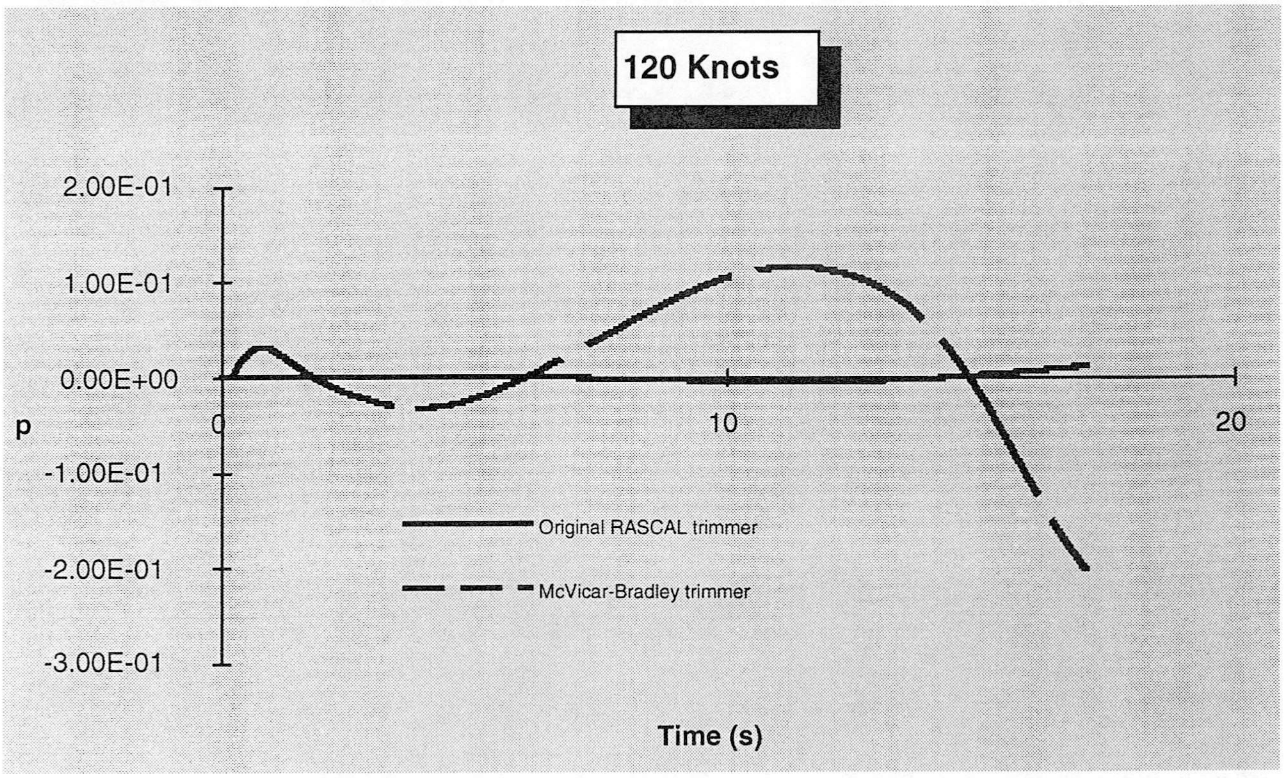

Figure 8d

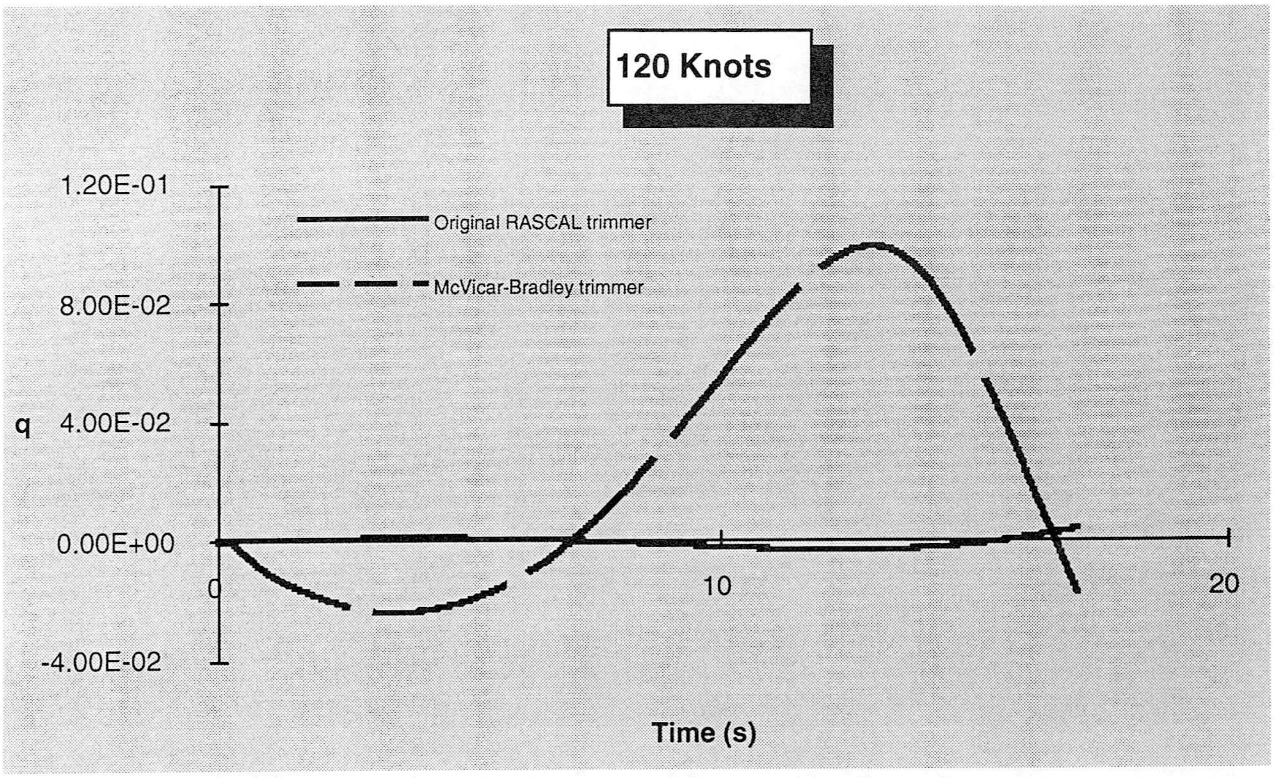

Figure 8e

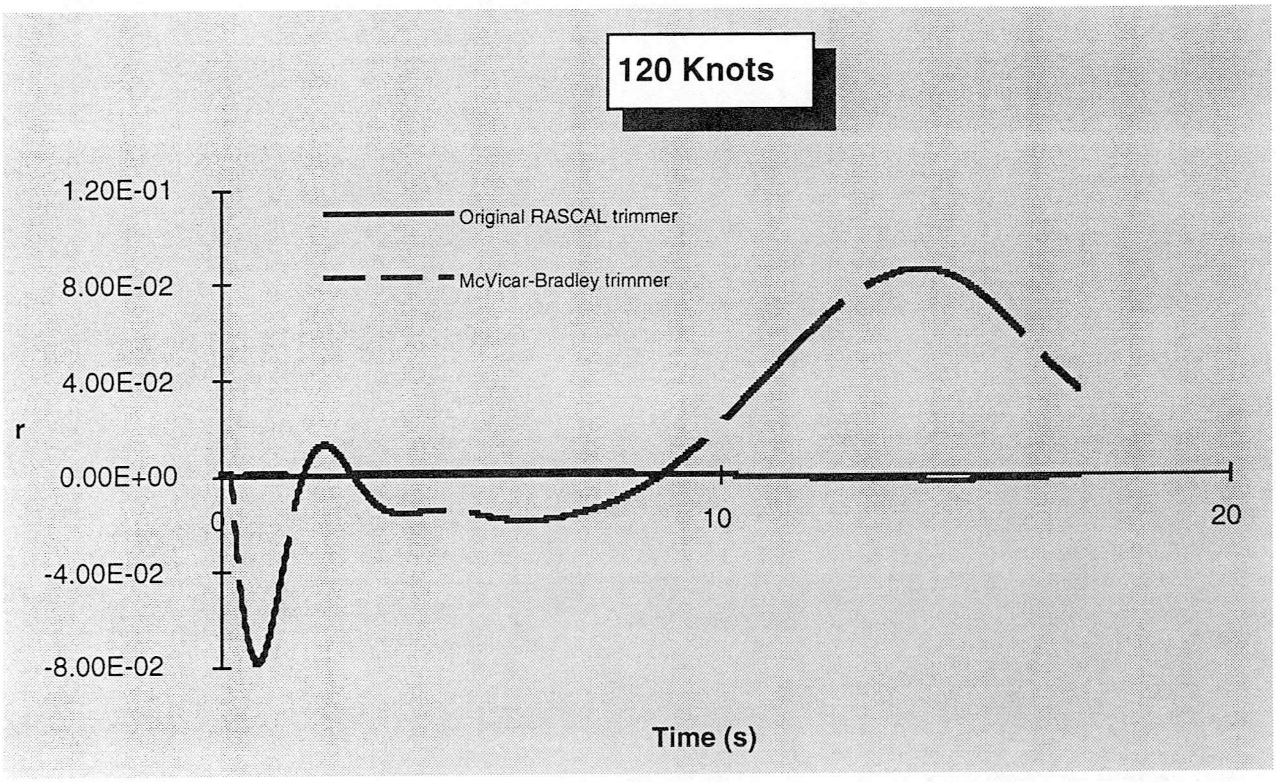

Figure 8f