Relatório de Estágio

Estágio para licenciatura no tema

Técnicas de Simulação, Controlo e Optimização para Sistemas Autónomos-

Aplicações na Robótica e em Processos de Manufactura

PRODEP Proc. Nº 4.3/7.04/92

Autor

José Miguel Soares de Almeida

Junho de 1993

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Departamento de Engenharia Electrotécnica e de Computadores
Rua dos Bragas, 4099 Porto Codex, PORTUGAL
Telef. 351-2-317105/107/412/457 · Telex 27323 FEUP P · Telefax 351-2-319280

Acção de Formação Prodep 4.3/3.4/92

Formando: José Miguel Soares de Almeida
Orientador Científico: Fernando Manuel Ferreira Lobo Pereira

Este relatório corresponde à apreciação científica das actividades desenvolvidas pelo Licº José Miguel Soares de Almeida durante o segundo semestre de 1992.

O trabalho desenvolvido foi de elevado mérito e executado com grande dedicação.

Os trabalhos realizados e o relatório apresentados pelo formando reflectem, com rigor a natureza dos trabalhos executados.

Além do empenhamento e das excelentes qualidades do estagiário concorreram para o sucesso deste trabalho, os meios disponibilizados.

O Supervisor do Estágio na FEUP

Fernando M. F. Lobo Pereira
Prof. Aux. da FEUP

ISR - Instituto de Sistemas e Robótica

DEEC - R. dos Bragas, 4099 Porto Codex, Portugal

Fax: 351 - 2- 319280 Tel: 351 -2 - 2007505

REFERÊNCIA DO PROJECTO 4.3/1.61/92

Parecer Técnico

Acompanhei os trabalhos de estágio realizados no Pólo do Porto do Instituto de Sistemas e Robótica pelo Licº José Miguel Soares de Almeida, bem como apreciei os trabalhos produzidos e apresentados em Congressos Internacionais e o relatório oportunamente apresentado.

Com base no acompanhamento referido, considero que aqueles trabalhos reflectem o trabalho efectivamente realizado, revelando elevado empenhamento do estagiário.

Sou portanto de parecer que foram plenamente atingidos os objectivos inicialmente propostos no plano de trabalhos,

O Supervisor do Estágio no Instituto de Sistemas e Robótica

J. L. Martins de Carvalho

Prof. Doutor Jorge L. Martins de Carvalho

Modelização e Simulação de Veículos Submarinos Autónomos

Massas Móveis

Relatório apresentado para o estágio do PRODEP sob o tema:
Técnicas de Simulação, Controlo e Optimização para Sistemas Autónomos-
Aplicações na Robótica em Processos de Manufactura

Realizado por: José Miguel Soares de Almeida

ISR-PORTO

Indice

1 Actividade desenvolvida

1.1 Estudos orientados efectuados

Estudo de técnicas de simulação, controlo e optimização para sistemas autónomos-aplicações na robótica e em processos de manufactura.

Verificou-se num campo específico da robótica móvel, inumeras carências no que diz respeito a técnicas e à teoria base, que permitam um desenvolvimento a nível industrial deste tipo de aplicações. Este campo: Robótica móvel submarina, pela sua grande possibilidade de intervenção nas areas económicas e cientificas, é pois uma área que urge aprofundar.

Várias técnicas e paradigmas estavam dísponíveis para o estudo de veículos submarinos autónomos. Nomeadamente no que diz respeito a controlo: Controlo Robusto, Controlo Multivariável, Controlo " Sliding-Modes", Controlo Adaptativo, Redes Neuronais e Teoria clássica de Sistemas. Em identificação e modelização temos várias áreas da Física: Hidrostática, Hidrodinâmica e Mecânica, e àreas do conhecimento tais como: Métodos de estimação de parametros, construção de modelos estocásticos e ainda técnicas de simulação.

Levantamento de resultados teóricos disponíveis e análise das respectivas limitações, nomeadamente em termos de aplicações.

Para o efeito efectuaram-se pesquisas bibliográficas exaustivas em várias bibliotecas sobre os temas em consideração. Esta pesquisa foi particularmente frutuosa porque permitiu coligir e agrupar muitos documentos sobre as áreas estudadas que se encontravam dispersos e por várias instituições nacionais e internacionais.

Estudo e simulação de algumas arquitecturas propostas sob supervisão do orientador, para ultrapassar as limitações encontradas no controlo de veículos submarinos autónomo.

No âmbito dos trabalhos desta fase foi dedicada particular atenção ao desenvolvimento de um ambiente de simulação que permitirá testar a validade, num contexto de simulação, dos conceitos envolvidos no estudo.

1.2 Disciplinas frequentadas

Controlo Óptimo

Identificação e Modelização de Sistemas

Métodos de Optimização

Robótica

Análise de Sistemas Informáticos

1.3 Participações em Conferências

Participação em diversas palestras sobre modelização e controlo de veículos sobmarinos autónomos, efectuadas na FEUP pelo Prof. Anthony Healey da Naval Postgraduate School (EUA).

Participação num breve curso leccionado pelo Prof. Carlos Pedreira da Pontificia Universidade Católica do Rio de Janeiro, sobre Teoria de Redes Neuronais.

Participação no " Workshop on Autonomous Underwater Vehicles " realizado no Porto e durante o qual foram abordados temas por várias personalidades pioneiras na área, nomeadamente o Prof. James Belligham do MIT (EUA), Prof. Anthony Healey, Prof. Thor Fossen NIT (Noruega).

1.4 Estágios

O estágio no Instituto de Sistemas e Robótica permitiu, por um lado, o acesso a uma rede de computação necessária ao desenvolvimento de algumas fases do trabalho e, por outro, tornou

possível a participação em projectos de I&D nacionais e europeus que endereçam aspectos práticos dos tópicos abordados no ambito deste trabalho.

2. Apêndices

2.1 Projecto Realizado

Modelização e Simulação de Veículos Submarinos Autónomos

Massas Móveis

Relatório apresentado para o estágio do PRODEP sob o tema:
Técnicas de Simulação, Controlo e Optimização para Sistemas Autónomos-
Aplicações na Robótica em Processos de Manofactura

ISR-PORTO

1. Introdução

Os oceanos constituem uma fonte de riqueza que, para muitos países é o principal suporte da sua economia. Se a sua superfície é hoje em dia bem conhecida, o mesmo não se pode dizer acerca das suas profundezas. E se os oceanos estão ainda bastante inexplorados, isto não quer dizer que haja grande desinteresse na sua exploração. Na realidade inúmeros benefícios económicos e científicos adviriam de um maior conhecimento deste meio.

Para exploração do meio subaquático existem poucos meios. Estes incluem: o uso de mergulhadores (autónomos ou não), submersíveis tripulados, submersíveis não tripulados e controlados remotamente e veículos submarinos autónomos.

Nesta classe de veículos destacam-se os veículos submarinos autónomos (doravante designados por AUV´s da sigla inglesa "Autonomous Underwater Vehicles") que têm ainda a facilidade de trabalhar em meios onde a comunicação entre o veiculo e um operador é bastante dificultada, ou de efectuar missões de uma forma muito mais eficiente. A capacidade dos AUV´s de serem programados para uma missão específica e depois a executarem trazendo se necessário resultados, faz deles um meio necessário e promissor na exploração subaquática.

No entanto, a tecnologia destes veículos encontra-se ainda na sua infância. Não existem sistemas precisos e baratos para navegação, o conhecimento das leis hidrodinâmicas que governam o comportamento dos AUV´s é, ou ainda incompleto ou de uma complexidade excessiva., a própria tecnologia de construção está ainda a dar os seus primeiros passos e o problema genérico do controlo e gestão de missão está ainda pouco explorado.

Neste contexto, surge a necessidade de se conseguirem boas soluções para um controlo eficaz nas mais diversas situações. Estas podem incluir desconhecimento do meio onde o AUV opera (correntes, pressões, temperaturas, topologia do fundo oceânico, etc), desconhecimento de características do próprio AUV (incertezas nos modelos) e situações inesperadas tais como obstáculos moveis ou situações de avaria.

Este trabalho foca dentro do vasto tema que é o controlo de AUV´s uma pequena parte que diz respeito ao controlo a baixo nível (aquilo que geralmente é estudado na teoria de controlo). E isto para situações determinadas que são descritas no texto, sugere soluções possíveis e apresenta métodos para a avaliação destas soluções.

Mais precisamente pretende-se estudar diversas estratégias para o controlo. Propõe-se arquitecturas e meios para a simulação e avaliação destas estratégias.

Iremos muitas vezes utilizar nomenclatura em língua inglesa, dado ser esta a de uso comum.

2 Massas móveis num modelo de AUV

A introdução de massas móveis no AUV vai permitir, através da sua colocação, alterar o centro de gravidade deste. No entanto tal alteração vai afectar a dinâmica do AUV, vai ser esta influência que seguidamente vamos estudar.

De modo a termos um fácil controle da colocação do centro de gravidade, vamos considerar, para tratamento analítico a seguinte estrutura:

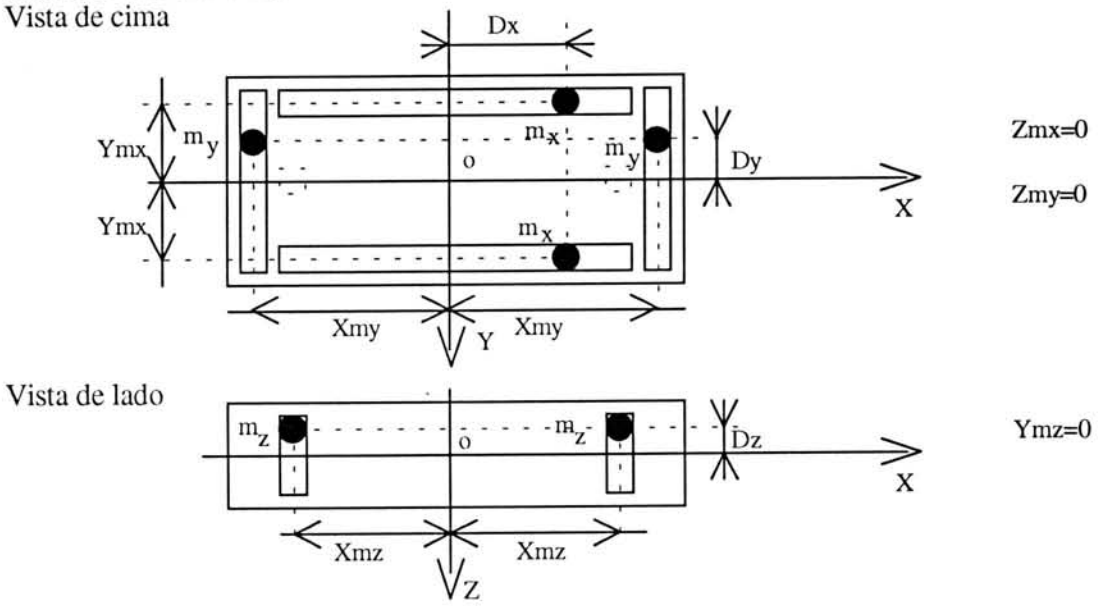


Figura 2.4

As razões fundamentais para a escolha desta estrutura são: para se poder controlar individualmente os três componentes do centro de gravidade (massas a moverem-se nos três eixos do AUV), e para os momentos resultantes das massas móveis serem não nulos apenas na perpendicular ao plano em que elas se movem.

Podemos considerar que a massa total do AUV, pode ser decomposta em duas componentes, uma fixa, (Mfixa) que corresponde á massa de toda a estrutura, e de todos os componentes do AUV, excepto as massas móveis. E outra móvel (Mmovel) que corresponde ás massas moveis. A força gravitíca aplicada ao AUV, tem então uma componente fixa Gfixa aplicada em:

$$R_{Gfixo} = R_o + \rho_{Gfixo}$$

E uma componente móvel Gmovel, aplicada em :

$$R_{Gmovel} = R_o + \rho_{Gmovel}$$

A força de gravidade resultante, com valor:

$$F_G = F_{Gfixo} + F_{Gmovel}$$

Pode ser considerada aplicada num ponto:

$$R_G = R_o + \rho_G$$

onde por definição, os momentos das diversas componentes se anulam, ou seja

$$\begin{aligned}
&F_{Gfixo} \times (R_G - R_{Gfixo}) = -F_{Gmovel} \times (R_G - R_{Gmovel}) \\
&\Leftrightarrow F_{Gfixo} \times (R_o - \rho_G - R_o + \rho_{Gfixo}) = -F_{Gmovel} \times (R_o - \rho_G - R_o + \rho_{Gmovel} \\
&\Leftrightarrow (F_{Gfixo} + F_{Gmovel}) \times \rho_G = F_{Gfixo} \times \rho_{Gfixo} + F_{Gmovel} \times \rho_{Gmovel} \\
&\Leftrightarrow \hat{g} \times (|F_{Gfixo}| + |F_{Gmovel}|) \times \rho_G = \hat{g} \times (|F_{Gfixo}| \cdot \rho_{Gfixo} + |F_{Gmovel}| \cdot \rho_{Gmovel}) \\
&\Rightarrow (|F_{Gfixo}| + |F_{Gmovel}|) \rho_G = |F_{Gfixo}| \cdot \rho_{Gfixo} + |F_{Gmovel}| \cdot \rho_{Gmovel} \\
&\Leftrightarrow \rho_G = \frac{|F_{Gfixo}| \cdot \rho_{Gfixo} + |F_{Gmovel}| \cdot \rho_{Gmovel}}{|F_{Gfixo}| + |F_{Gmovel}|}
\end{aligned}$$

Se admitirmos que :

$$\rho_{Gfixo} = 0$$

ou seja, se se considerar que o centro de gravidade das massas fixas coincide com o de impulsão, então através do posicionamento das massas moveis podemos colocar o centro de gravidade do AUV em qualquer posição em torno do centro de impulsão, sendo a distância entre elas limitada pelo valor da percentagem das massas moveis no total, e pela distância que estas podem percorrer. E então a expressão anterior toma a forma:

$$\rho_G = \frac{M_{movel}}{M_{fixa} + M_{movel}} \cdot \rho_{Gmovel}$$

Uma outra questão a ter em conta, é que, o facto de existir massas moveis vai implicar que a matriz dos momentos de inércia vai ficar dependente das posições das massas.

Deste modo temos agora que:

$$I = I_{fixo} + I_{movel} \tag{2.11}$$

onde:

$$I_{movel} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}$$

Se a dimensão das massas moveis for desprezável comparado com a sua distancia ao centro de impulsão, podemos considera-las pontuais.

E então, temos que:

$$I_{xx} = 2 \cdot m_x \left(Y_{mx}^2 + 0^2\right) + 2 \cdot m_y \cdot \left(dy^2 + 0^2\right) + 2 \cdot m_z \cdot \left(dz^2 + 0^2\right)$$
$$I_{yy} = 2 \cdot m_x \cdot \left(dx^2 + 0^2\right) + 2 \cdot m_y \cdot \left(X_{my}^2 + 0^2\right) + 2 \cdot m_z \cdot \left(X_{mz}^2 + dz^2\right)$$
$$I_{zz} = 2 \cdot m_x \cdot \left(dx^2 + Y_{mx}^2\right) + 2 \cdot m_y \cdot \left(X_{my}^2 + dy^2\right) + 2 \cdot m_z \cdot \left(X_{mz}^2 + 0^2\right)$$

$$I_{xy} = 2 \cdot m_x \cdot \left(dx \cdot Y_{mx} - dx \cdot Y_{mx}\right) + 2 \cdot m_{y\cdot} \cdot \left(dy \cdot X_{my} - dy \cdot X_{my}\right) + 2 \cdot m_z \cdot \left(X_{mz} \cdot 0 - 0 \cdot X_{mz}\right) = 0$$
$$I_{xz} = 2 \cdot m_x \cdot \left(dx \cdot 0 + dx \cdot 0\right) + 2 \cdot m_{y\cdot} \cdot \left(X_{my} \cdot 0 - X_{my} \cdot 0\right) + 2 \cdot m_z \cdot \left(X_{mz} \cdot dz - X_{mz} \cdot dz\right) = 0$$
$$I_{yz} = 2 \cdot m_x \cdot \left(Y_{mx} \cdot 0 - Y_{mx} \cdot 0\right) + 2 \cdot m_{y\cdot} \cdot \left(dy \cdot 0 + dy \cdot 0\right) + 2 \cdot m_z \cdot \left(dz \cdot 0 + dz \cdot 0\right) = 0$$

A matriz anterior toma então a forma:

$$I_{movel} = \begin{bmatrix} 2 \cdot \left(m_x \cdot Y_{mx}^2 + m_y \cdot dy^2 + m_z \cdot dz^2\right) & 0 & 0 \\ 0 & 2 \cdot \left(m_y \cdot X_{my}^2 + m_z \cdot X_{mz}^2 + m_x \cdot dx^2 + m_z \cdot dz^2\right) & 0 \\ 0 & 0 & 2 \cdot \left(m_x \cdot Y^2{}_{mx} + m_y \cdot X^2{}_{my} + m_z \cdot Y^2{}_{mz} + m_x \cdot dx^2 + m_y \cdot dy^2\right) \end{bmatrix}$$

Por último vamos analizar o efeito das forças de reacção que surgem quando se aplicam forças para movimentação das massas.

Para conseguir aplicar as acelerações pretendidas ás seis massas é necessário aplicar nestas, seis forças, que dado a simetria e estrutura do problema podem ser representadas por uma força aplicada no centro de gravidade com o valor de:

$$F_{cm} = \begin{bmatrix} m_x \cdot \ddot{d}_x \\ m_y \cdot \ddot{d}_y \\ m_z \cdot \ddot{d}_z \end{bmatrix} \tag{2.12}$$

Força esta que deve ser considerada nas forças aplicadas ao AUV.

O sistema de massas móveis pode ser considerado como um sistema àparte do AUV, assim podemos descrever o movimento das massas, por:

$$F_{massa_i} = m_i \ddot{d}_i + k(\dot{d}_i^{\,2}) + G_i(\Theta)$$

onde F_{massai} é a força aplicada à massa i, d_i é a posição da massa i e Gi é a decomposição da força da gravidade segundo a direcção de movimento da massa. Como tal Gi é dependente da posição do AUV (mais própriamente dos três ângulos de Euler).

Podemos exprimir a relação anterior de forma matricial:

$$\begin{bmatrix} F_{mx} \\ F_{my} \\ F_{mz} \end{bmatrix} = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & m_z \end{bmatrix} \cdot \begin{bmatrix} \ddot{d}_x \\ \ddot{d}_y \\ \ddot{d}_z \end{bmatrix} + k \begin{bmatrix} \dot{d}_x^{\,2} \\ \dot{d}_Y^{\,2} \\ \dot{d}_Z^{\,2} \end{bmatrix} + \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & m_z \end{bmatrix} \cdot T^{-1} \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

onde g é a aceleração da gravidade e T^{-1} é a matriz de transformação de coordenadas definida na secção 2.1. Efectuando as multiplicações da última parcela da expressão anterior, pode-se obter finalmente:

$$\begin{bmatrix} F_{mx} \\ F_{my} \\ F_{mz} \end{bmatrix} = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & m_z \end{bmatrix} \cdot \begin{bmatrix} \ddot{d}_x \\ \ddot{d}_y \\ \ddot{d}_z \end{bmatrix} + k \begin{bmatrix} \dot{d}_x^{\,2} \\ \dot{d}_Y^{\,2} \\ \dot{d}_Z^{\,2} \end{bmatrix} + \begin{bmatrix} -sen\theta \cdot m_x g \\ -sen\Phi \cdot sen\theta \cdot m_y \cdot g \\ -\cos\Phi \cdot \cos\theta \cdot m_z \cdot g \end{bmatrix}$$

Esta última relação descreve-nos o movimento das massas. Pode-se pois usa-la como um modelo para o sistema de massas que queremos controlar. Devemos notar que este modelo corresponde a uma estrutura de massas que foi descrita anteriormente. Para outros esquemas e disposições de massas móveis, o modelo do sistema será neceessáriamente diferente. No entanto a influência destas no AUV é semelhante à acima descrita. Um esquema merecedor de estudo será o de massas pêndulares. Estas poderam estar dispostas em pêndulos com 2 graus de movimento e centrados nos eixis do referêncial ou em pêndulos cónicos.

3. Resumo do modelo

Após termos estudado os efeitos da introdução de massas móveis no modelo do AUV, recordando o modelo a que tinhamos chegado na secção 2.1, vamos aqui apresentar a sua forma final já com os efeitos das massas móveis, incluídos.

As alterações introduzidas pelas massas móveis, são a inclusão de um termo variável na matriz de inércia do AUV (2.11) e pelo aparecimento de forças exteriores devidas ao movimento das massas (2.12). Além disso agora a posição do centro de gravidade é variável. Deste modo a matriz G(Θ) depende também das coordenadas do centro de gravidade. Não se vai verificar nenhuma alteração na sua expressão (2.5), apenas agora x_G, y_G e z_G deixam de ser fixos.

O modelo do AUV será dado pois, pela equação matricial:

$$M\dot{x} + Cx + D(x)x + G(\Theta, \rho_G)x + K(\ddot{\rho}_G) = B_1 u + B_2 u^2$$

onde ρ_G é a posição do centro de massa. A matriz K, inclui as forças devidas ao movimento das massas. Denotando as acelerações das massa por $\ddot{d}_x$, $\ddot{d}_y$ e $\ddot{d}_z$, teremos:

$$K(\ddot{\rho}_G) = \begin{bmatrix} m_x \ddot{d}_x \\ m_y \ddot{d}_y \\ m_z \ddot{d}_z \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

onde mx, my e mz são as massas móveis.

A matriz M, será alterada para incluir os termos variáveis da matriz de inércia do AUV. Será pois:

$$M = \begin{bmatrix} m - X_{\dot{u}} & -X_{\dot{v}} & -X_{\dot{w}} & -X_{\dot{p}} & mz_G - X_{\dot{q}} & -my_G - X_{\dot{r}} \\ -X_{\dot{v}} & m - Y_{\dot{v}} & -Y_{\dot{w}} & -mz_G - Y_{\dot{p}} & -Y_{\dot{q}} & mx_G - Y_{\dot{r}} \\ -X_{\dot{w}} & -Y_{\dot{w}} & m - Z_{\dot{w}} & my_G - Z_{\dot{p}} & -mx_G - Z_{\dot{q}} & -Z_{\dot{r}} \\ -X_{\dot{p}} & -mz_G - Y_{\dot{p}} & my_G - Z_{\dot{p}} & I_x - K_{\dot{p}} + I_{xmóvel} & -I_{xy} - K_{\dot{q}} & -I_{xz} - K_{\dot{r}} \\ mz_G - X_{\dot{q}} & -Y_{\dot{q}} & mx_G - Z_{\dot{q}} & -I_{xy} - K_{\dot{q}} & I_y - M_{\dot{q}} + I_{ymóvel} & -I_{yz} - M_{\dot{r}} \\ -my_G - X_{\dot{r}} & mx_G - Y_{\dot{r}} & -Z_{\dot{r}} & -I_{xz} - K_{\dot{r}} & -I_{yz} - M_{\dot{r}} & I_z - N_{\dot{r}} + I_{zmóvel} \end{bmatrix}$$

Onde $I_{xmóvel}$, $I_{ymóvel}$ e $I_{zmóvel}$ são dados de acordo o que foi exposto na secção anterior, por:

$$I_{xmóvel} = 2 \cdot (m_x \cdot Y^2{}_{mx} + m_y \cdot d_y{}^2 + m_z \cdot d_z{}^2)$$
$$I_{ymóvel} = 2 \cdot (m_y \cdot X^2{}_{my} + X^2{}_{mz} + m_x \cdot d_x{}^2 + m_z \cdot d_z{}^2)$$
$$I_{zmóvel} = 2 \cdot (m_x \cdot Y^2{}_{mx} + m_y \cdot X^2{}_{my} + m_z \cdot Y_{mz} + m_x \cdot d_x{}^2 + m_y \cdot d_y{}^2)$$

Onde é seguida a notação da secção anterior. As matrizes C, D, B1 e B2 mantêm-se as mesmas, sendo dadas pelas espressões apresentadas na secção 2.1.

Devemos notar que aqui ao não incluirmos a posição das massas móveis como parte do estado do veículo, estamos a considerar como parâmetros da planta. Ora isto significa que consideramos o sistema massas móveis como um sistema controlado independentemente do AUV e cuja ligação com este é a expressa pelas expressões acima apresentadas. Dado que vamos apenas no controlo das massas tentar eliminar os momentos restabelecedores causados pelas forças da gravidade e de impulsão, o controlo do AUV, no que diz respeito a massas, vai pois supor que a atitude pretendida para o veículo já não é prejudicada pela força de impulsão e gravítica (já se colocaram as massas na posição apropriada).

4. Arquitectura para controlo do AUV

4.1 Descrição

Para podermos fazer um controlo eficiente do AUV, necessitamos de definir não só como vai ser o seu controlo de baixo nível, mas também a sua interacção com o controlo de alto nível.

Não é conceptível que se esteja por exemplo a optimizar qualquer parâmetro localmente sem fazer um estudo de toda a perspectiva global. Por isso surge a necessidade de definir uma arquitectura para o AUV. Esta, deve integrar os vários aspectos do controlo e execução da missão do veículo, fornecendo uma base sólida para o estudo de vários aspectos particulares do controlo.

Ao especificarmos uma arquitectura tentamos com que o AUV, tivesse o maior grau de autonomia possível e que não sofresse de limitações face a qualquer tipo de missão que se desejasse. Esta preocupação levou a que se concebesse uma arquitectura onde a tarefa de definir a trajéctoria final do veículo, não pudesse ser limitada. Com a nossa arquitectura é possível efectuar manobras de grande complexidade, caso sejam desenvolvidos controladores para isso.

Por outro lado é necessário compreender que o controlo do veículo não pode ser sempre feito tomando por base todo o detalhe do modelo deste (por dificuldades de cálculos em tempo real ou por inexistência de métodos eficientes). É pois usual transformar um modelo não linear do veículo, num linear onde as técnicas existentes para esta gama de sistemas possam ser aplicadas.

Podemos observar na figura seguinte um esquema para uma arquitectura para o AUV:

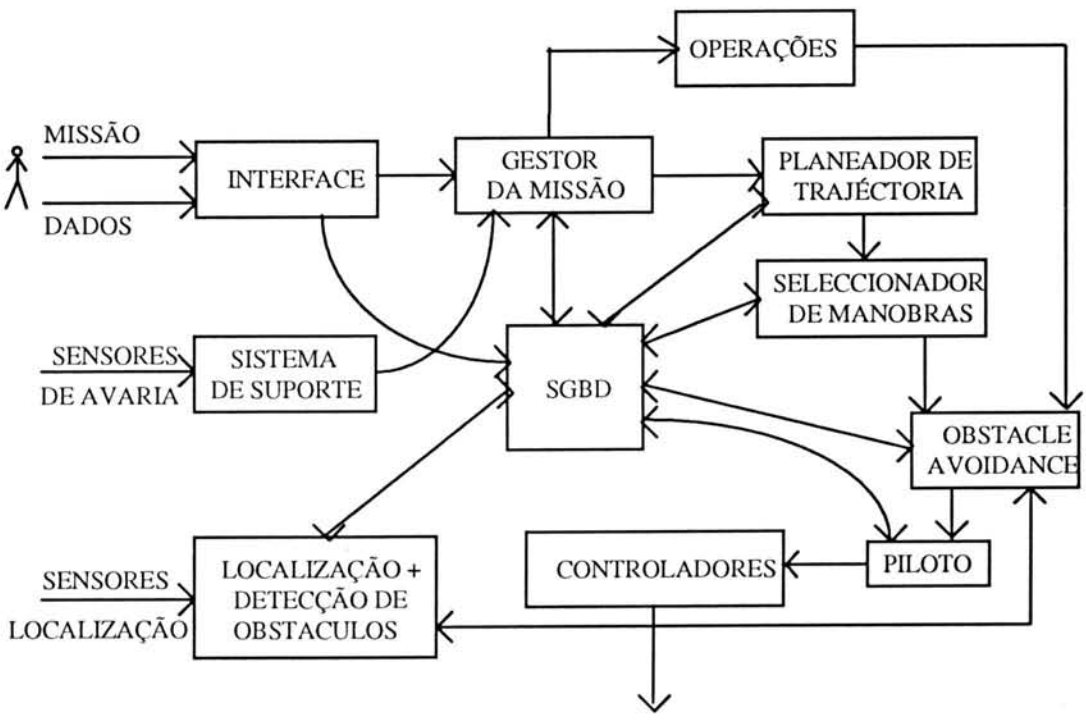


Figura 4.1

Não se encontra esquematizado o módulo de coordenação, dado ir sobrecarregar em demasia a figura. Este módulo encontra-se ligado a todos (excepto ao interface que não é residente no AUV) e faz a coordenação do funcionamento dos vários módulos.

Existem duas fases no planeamento e execução (o que obviamente envolve controlo) da missão: a sua especificação por parte do utilizador do veículo e a sua execução.

Deve existir uma interface entre o utilizador e o AUV, que permita ao primeiro definir com precisão e facilidade o conteúdo da missão a executar. Esta interface deve ser de utilização fácil e intuitiva. Como tal deve ser criada uma linguagem ou meio de comunicação que permita a especificação da missão. Esta linguagem deve não limitar as capacidades de definição da tarefa que o AUV vai cumprir. Por exemplo: deve ser possível pormenorizar o trajecto do veículo e todo o seu comportamento ao longo dele. Deve-se poder especificar a sua atitude, os valores de velocidade e aceleração instântanea, o comportamento dos sensores (aqueles destinados a fazer observações) em todos os instantes. A par de todo este detalhe, deve ser possível definir a missão em termos muito mais gerais. Pode-se por exemplo especificar para uma missão: "procura e fotografa peixes de um dado tipo". Se bem que é uma tarefa difícil, deve a interface tentar fazer uma validação da missão especificada, avaliando da sua admissibilidade.

Fazer a entrada das especificações do utilizador para o veiculo não é a única tarefa da interface. O utilizador pode também introduzir outro tipo de dados, tais como configuração do veículo ou dados de navegação e de conhecimento do meio ambiente onde o AUV vai operar.

Após o utilizador ter comunicado os dados ao sistema, a interface faz um processamento dos dados e estes são carregados no veículo. Os dados genéricos que não dizem respeito à definição da missão são processados (se for caso disso) e são armazenados na base de dados interna do veículo. A especificação da missão é processada para ter-se uma definição interna e eficiente da missão e é enviada ao gestor de missão.

A tarefa do gestor de missão é com base no conhecimento que tem sobre a situação do veículo (estado de funcionamento, localização, estratégias de execução de missão, etc.), determinar se se vai planear trajectórias ou executar uma dada manobra. É competência do gestor de missão definir o que é que o veículo vai fazer.

Dividimos o comportamento do veículo em duas possibilidades:

- condições de voo
- operações

Nas condições de voo, supõe-se que o AUV está a deslocar-se de um ponto para outro. Corrresponde a tarefas de transporte ou de deslocação para o local de trabalho. Neste tipo de condições, normalmente uma componente da velocidade do veículo é bastante maior que as outras (é frequente que seja a "surge"). Como tal a tarefa de controlo é facilitada, dadas as aproximações que se podem fazer quanto à dinâmica do veículo.

Em operações o veículo encontra-se a executar uma manobra a baixa velocidade tal como acostar ou fotografar um alvo. O caso de "hovering" (manter a mesma posição e atitude) está incluído neste tipo de situações.

O gestor de missão dado que conhece a missão deve determinar que em modo de funcionamento este vai estar e actuar em conformidade. Se o AUV deve deslocar-se para uma dado ponto, então o gestor de missão envia o destino ao planeador de trajectórias para que este determine qual a trajectória a seguir. Se o AUV tiver que executar uma manobra o gestor de missão envia a especificação dessa operação ao bloco "operações" que a transforma em manobras (mais tarde iremos definir o que entendemos por manobras).

O sistema de suporte deve com base nos dados recebidos dos sensores do veículo, determinar o seu estado de operacionalidade. Pode ser por exemplo implementado com redes neuronais, que detectam avarias e comunicam essa situação ao bloco de coordenação e à base de dados do sistema.

O bloco de navegação trata de determinar a localização do AUV e faz a detecção de obstáculos. A sua informação de entrada são os sensores do veículo e os dados armazenados na base de dados e fornece informação ao módulo de obstacle avoidance e à base de dados.

O sistema de gestão de base de dados, armazena e gere toda a informação vital ao funcionamento do veículo. Básicamente podemos dizer que é o "centro" de informação assim como o módulo de coordenação é o "centro" de controlo. Base de dados deve poder fornecer informação aos vários módulos no formato necessário e deve poder atender vários pedidos simultaneamente (aqui a simultaneidade é um conceito relativo, deve pelo menos existir um esquema de prioridades caso não se consiga garantir um fornecimento eficiente de informação).

O módulo operações recebe do gestor de missão a operação a efectuar e transforma-a num conjunto de manobras (ver definição na secção seguinte). Só actua quando o veículo deve fazer manobras delicadas ou a baixa velocidade.

O planeador de trajectórias, faz a sua tarefa usual, isto é, planeia a trajectória uma vez dados o ponto origem e o destino. Usa o conhecimento sobre o mundo, que existe na base de dados. A trajectória é fornecida ao seleccionador de manobras. Este módulo faz a decomposição da trajectória num conjunto de manobras, a ser executado pelo AUV (efectua uma tarefa similar a "gain scheduling").

O obstacle avoidance com base no conhecimento do ambiente e das informações recebidas da detecção de obstáculos determina se é necessário alterar a manobra que o AUV vai executar, e se for caso altera-a por forma a evitar o obstáculo. Se for impossível continuar sem colidir com o obstáculo informa o módulo de coordenação desse facto.

O piloto é o módulo responsável pela execução de uma dada manobra. Recebe manobras e fornece instruções e referências aos controladores para que estes possam fazer o controlo mais apropriado face ao tipo de acção que o veículo vai tomar.

Os controladores, estão directamente ligados ao veículo e determinam o valor das grandezas de controlo, tais como tensões nos motores ou nos servomecanismos das superfícies de controlo.

Podemos observar na figura seguinte um esquema mais detalhado do funcionamento dos controladores.

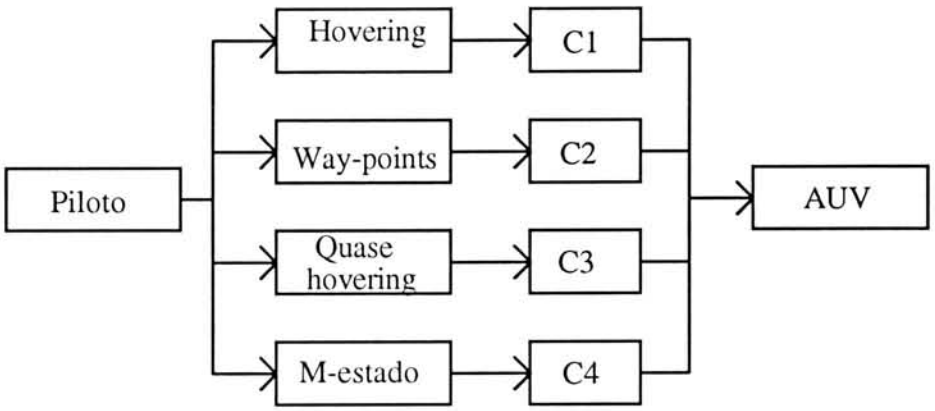


Figura 4.2

Podemos observar que o piloto encontra-se ligado a quatro tipos de controladores. A função do piloto é a de após receber uma manobra a ser efectuada, conforme as suas características seleccionar o controlador apropriado. Dividimos o controlo a baixo nível em quatro tipos, dois para cada modo de operação (condições de voo ou operações). Temos:

-Hovering

-Quase Hovering

-Way points

-Manutenção de estado

Os dois primeiros funcionam quando estamos em operações e os seguintes quando o veiculo opera em condições de voo.

Em Hovering, o controlador recebe a posição, a atitude e a tolerância e é dada ordem ao controlador de baixo nível para manter a referência. Basicamente o problema é um problema de regulação.

Em Quase Hovering estamos a efectuar uma operação a baixa velocidade, pelo que a dinâmica do modelo é diferente das condições de voo. Funcionalmente este módulo é semelhante ao de Manutenção de estado, a diferença reside nos controladores de baixo nível que cada um comanda. Esses controladores estão preparados para dois tipos de plantas diferentes (em alta ou baixa velocidade).

O bloco Way Points recebe do piloto 3 pontos: o ponto anterior, o ponto actual e o próximo ponto, recebe também a precisão com que se deve passar nesses pontos e a tolerância para a velocidade na trajectória. Este bloco calcula então qual deve ser a referência a dar aos controladores de baixo nível e os

ajustes aos seus parâmetros, por forma a que o AUV possa passar no próximo ponto, a partir do actual. Envia ao piloto a informação de que já cumpriu a tarefa quando tal acontecer.

O módulo Manutenção de estado mantém uma dada atitude e velocidade (estado) até nova ordem. Recebe também margens de tolerância para o estado e para a atitude. O Quase Hovering é semelhante a este diferindo apenas no tipo de controladores que comanda.

Por este esquema se vê que cabe ao piloto a decisão de escolher qual o bloco a ter o controlo e de lhe fornecer a informação apropriada.

Os controladores de baixo nível, são constituídos por software que processa a referência recebida dos blocos superiores (se tal for necessário) e que fornece a referência para os compensadores do sistema. Os controladores podem ser de diversos tipos tais como os descritos na secção 3 ou outros.

4.2 Manobras

4.2.1 Definição

As manobras são fases da trajectória do veículo onde o modelo pode ser considerado linear. Além disso são as diversas fases em que se poderia fazer uma decomposição lógica da trajectória. O facto de se afirmar que numa manobra o modelo é considerado linear não significa que esta não possa ser complicada. Além disso as manobras podiam não corresponder a linearizações do modelo.

Uma manobra pode ser de baixa ou de alta velocidade. É conforme o tipo da manobra que se escolhe qual vai ser o bloco de controlo, basicamente das características da manobra vai depender o modo de pilotagem do AUV

Dado que uma manobra é uma tarefa simples, podemos supor que durante esta o estado do veiculo não varia muito. Por isso podemos linearizar o modelo em torno desse estado. Este modelo linearizado será depois utilizado no controlo do veículo. Esta linearização poderá ou não ser feita pelo piloto.

4.2.2 Desacoplamento

Uma questão se põe quando temos uma trajectória complexa e a queremos subdividir em manobras: não estamos a neglegênciar o acoplamento das variáveis da dinâmica?

É uma tarefa difícil determinar a separação de uma manobra, já que se uma variável do estado pode não variar dentro de um dado período de tempo, outras não. Uma solução de compromisso consiste em tentar desacoplar (com os custos de eficácia que isso acarreta), as várias variáveis de estado e então em cada uma delas, analisar a trajectória e subdividi-la em manobras.

O problema da linearização do modelo no nosso caso não é tão grave como no caso geral, já que nós usamos massas móveis para o controlo activo das forças de restabelecimento. Um dos problemas

mais graves da linearização era que, existindo forças de restabelecimento dependentes da posição do veículo no referencial GLOBAL (ver secção 2), a linearização dependia dessa posição e logo estávamos obrigados a entrar em conta com as matrizes de transformação de coordenadas. Como nós possuímos controlo activo de forças de restabelecimento, eliminar estas forças e logo a linearização pode ser efectuada no referencial do AUV.

4.2.3 Algoritmo de separação de trajectória

Para separar a trajectória em manobras pode-se aplicar um algoritmo simples que dentro de uma dada tolerância para o valor de uma variável de estado, considera o movimento como uma manobra. Este algoritmo possui duas fases:

- Desacoplamento parcial das variáveis de estado
- Análise da evolução desejada da variável de estado ao longo do tempo. Para intervalos pré-determinados dos valores dela, sempre que a variável de estado é mantida dentro do intervalo, essa secção da trajectória é considerada uma manobra.

Este algoritmo muito simples permite determinar quais as manobras que existem numa trajectória. Posto isto, já sabemos quais são os pontos de funcionamento em torno dos quais se vai linearizar a planta. Para isto contribui o facto de nós propormos controlo activo de forças e momentos de restabelecimento.

5. Modelo de Simulação

5.1 Arquitectura do Simulador

Para podermos simular os efeitos da nossa abordagem ao problema de controlo do AUV, necessitamos de ter um modelo computacional do mesmo. De facto, dado que estamos interessados em comparar a influência do nosso controlo face aos controlos "clássicos", necessitamos de dois modelos do veículo. Um será o modelo apresentado na secção 2.1, ou seja o modelo do veículo sem controlo das forças e momentos de restabelecimento. Outro, será um modelo (descrito na secção 2.2), que inclui a alteração da posição dos centros de gravidade e de impulsão, tendo em vista o controlo das forças de restabelecimento.

No nosso simulador necessitamos de conseguir descrever, não só a planta, como também os controladores e o controlo de alto nível. Esta necessidade de incluir a simulação de controlo de alto nível, deve-se ao facto de podermos desta forma estudar a as situações ocorridas na realidade.

Se por um lado necessitamos de uma simulação que integre os aspectos fundamentais do controlo do AUV, por outro, a simulação completa do veículo, incluindo a simulação de planeamento de trajectórias (controlo de alto nível), é um problema demasiado vasto e que não permite focar a nossa atenção nas questões aqui levantadas.

A solução consiste em propor uma arquitectura para a simulação que seja ao mesmo tempo completa e modular. A modularidade permite-nos concentrar o estudo sobre determinados aspectos do comportamento do veiculo, sem nos preocuparmos imediatamente com problemas de nível mais elevado. Por outro lado, a modularidade permite mais tarde analisar outros aspectos do controlo do AUV e logo aproveitar o trabalho já feito.

Os módulos que vão existir na nosso simulador são: planta, controlo adaptativo, controlador, planeador on-line, planeador off-line e simulador.

O módulo planta, contém o modelo do veículo. Mudando este módulo, conseguimos obter simulações o para diferentes tipos de veículos. No nosso caso analisamos os modelos com e sem massas móveis. Este módulo tem por entradas o estado actual do veículo, o controlo, o tempo actual e o tempo após o qual será calculado o novo estado. Este módulo poderá apresentar ou não o próximo estado como saída. Isto corresponderá à situação de termos ou não acessível o estado do veículo. Para tornarmos o modelo de simulação mais flexível vamos admitir que a saída não é o estado da planta. E que o módulo estimador de estado obtém através da informação disponível, uma estimativa do estado da planta. No caso de querermos admitir o estado como acessível, basta fazer a saída da planta igual ao próximo estado e o estimador reduzir-se-á à identidade.

O estimador de estado transforma a saída da planta no estado dela. Isto para como foi referido, se poder simular dinâmica de sensores (incluída no módulo planta) ou inacessibilidade do estado, de uma forma geral. O estimador pode ou não possuir memória. Ou seja, pode ou não conhecer toda a história

passada das saídas da planta ou dos controlos aplicados. Como entradas recebe o controlo e a saída da planta. Como saída fornece uma estimativa do estado da planta.

O módulo controlador, implementa o controlador de baixo nível da planta (muitas vezes referido por compensador). Este controlador recebe como entradas o estado do veículo (ou uma estimativa deste), a referência (valor instantâneo) e possivelmente alguns parâmetros internos do próprio controlador (para controlo adaptativo). Como saída apresenta o próximo controlo. Deve-se referir que por troca deste módulo, pode-se mudar de diversos controladores. Por exemplo: no caso de se efectuar um estudo de estratégias de "gain scheduling" necessitamos de um módulo que em função do estado actual da planta, substitua um controlador por outro.

O controlo adaptativo procede ao ajuste dos parâmetros do controlador corrente. Este ajuste é feito de acordo com uma lei de controlo adaptativo. Por isso, esta parte necessita de saber o estado da planta (obtido a partir do estimador de estado) e a referência. Como saída fornece os parâmetros do controlador.

O planeador on-line decide em função do instante de tempo corrente e da referência recebida do planeador off-line (o conjunto de referências para todos ou uma parte dos instantes de tempo), qual vai ser a referência a aplicar à planta. Este módulo existe para que se possa efectuar algum controlo na forma como a referência é apresentada ao controlador. Podemos, por exemplo receber uma referência de alto nível a partir do planeador off-line, como uma trajectória. E o planeador on-line decide em função da missão recebida e do instante de tempo actual, qual será a referência a aplicar ao veículo. Este módulo pode também implementar um planeamento, a ser feito on-line, tal como evitar obstáculos.

O planeador off-line, produz a referência de alto nível, que vai descrever a missão pretendida pelo veículo. Corresponde usualmente ao planeador de trajectórias. Recebendo a especificação da missão por parte do utilizador, transforma esta numa referência ao longo do tempo a aplicar ao veículo. Esta referência pode ser descrita em alto nível (por exemplo, pontos onde o veículo deve passar, "waypoints" na literatura inglesa) ou em mais baixo nível (já a função ao longo do tempo, da atitude e velocidades do veículo). Competirá ao planeador on-line transformar esta referência na que vai ser apresentada ao controlador.

O módulo simulador, controla a simulação. Apenas simula a passagem do tempo. Recebe o próximo controlo e o novo estado e actualiza-os passando-os a controlo e estado corrente. Podemos não considerar o estado como acessível no que diz respeito ao controlo, mas o facto é que a planta sabe qual é o seu estado, logo ao simularmos a passagem do tempo dispomos do estado real da planta. Este bloco fornece também qual o instante de tempo actual e qual o passo de integração da planta. Isto é qual o intervalo de amostragem, no que diz respeito à integração das equações dinâmicas do veículo pela planta. O simulador recebe também do utilizador o tempo de início e fim de simulação e o passo de simulação. Recebe o estado inicial do veiculo e o controlo inicial. Regista também todos os dados necessários para um posterior estudo da simulação. Nomeadamente o controlo, a estimativa do estado, o estado, e a saída da planta, poderá ainda registar os parâmetros fornecidos pelo controlo adaptativo ou a

evolução da referência aplicada ao controlador. Como saídas apresenta o tempo actual e o passo de integração da planta, o estado actual e o controlo actual. Fornece também todas as estatísticas recolhidas na simulação.

Na Figura 5.1 podemos observar um esquema apresentando estes diversos blocos do simulador e a a sua interligação.

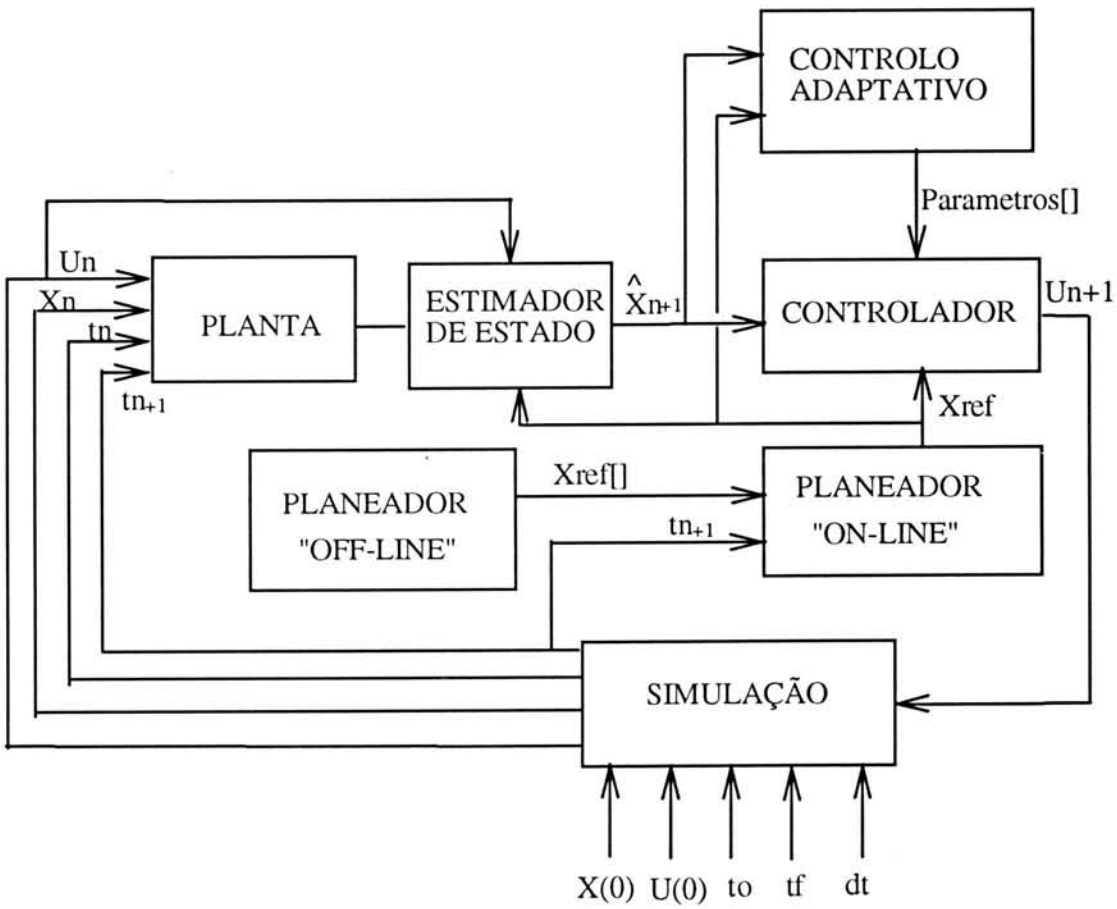


Figura 5.1

Este esquema faz a simulação de "baixo nível" do AUV, seria necessário implementar um sistema complexo de simulação para toda a arquitectura proposta no capítulo anterior.

5.2 Implementação do Simulador

Para implementar este modelo de simulação, o uso de uma linguagem de alto nível como o Matlab, proporciona facilidades ao nível do tratamento matemático. Certos módulos como o controlador podem ser implementados em C, para aumentar a rapidez de cálculo. Já que o Matlab permite aceder a rotinas em C, pode-se implementar o programa em Matlab e fazer chamadas a C sempre que necessário.

Para visualisar resultados em PC, usamos Excel dado a versão existente para PC do Matlab não ser muito poderosa em termos gráficos.

Efectuamos simulações do AUV com massas móveis e com um controlador de "feedback" simples. O objectivo do controlo foi o de anular os momentos restabelecedores devidos à gravidade e à impulsão. Para uma dada atitude colocou-se as massas na posição conveniente.

Simulamos 3 manobras para experimentar as massas móveis. Na manobra 1 o AUV arranca só ângulo no leme de "rudder". Deste modo o veículo descreve um circunferência no plano xy. Observando-se os resultados da simulação, vemos que a inclusão de massas móveis não afecta as prestações do veículo de forma perceptível, já que este à semelhança do que acontece sem massas, efectua uma circunferência como trajectória.

Na manobra 2 só o leme de "stern" tem ângulo não nula, estando o veículo já posicionado inicialmente de forma a fazer uma circunferência num plano obliquo. Aqui a ideia é a de aproveitar a superfície do AUV para o ajudar a efectuar a curva. O veículo vai fazer uma curva num plano obliquo, semelhante a um "looping" inclinado. Podemos observar em apêndice a trajectória obtida e é de salientar que o AUV, efectua a manobra sempre no mesmo plano (como se pode concluir pelas figuras). Deste modo ao introduzirmos massas móveis e com elas anularmos os momentos devidos à gravidade e impulsão, estamos a permitir que o AUV efectue manobras difíceis como esta (note-se o acoplamento entre os graus de liberdade do veículo) em muito melhores condições do que sem o nosso sistema de controlo de massas. De facto para o controlo convencional de veículos deste tipo não é usual permitir-se manobras como esta.

Na manobra 3 os lemes de "rudder" e "stern" têm o mesmo ângulo. Assim o veículo vai simultaneamente descrever uma circunferência no plano xy, e subir descrevendo outra no plano xz. Dado que esta manobra é altamente acoplada e que os efeitos do "rudder" são mais proeminentes que os de "stern", o veículo descreve uma circunferência em xy e não completa a curva em xz. De qualquer modo pode-se considerar que atendendo à dificuldade da manobra (manobra de "acrobacia" que não é admitida no controlo usual de AUV's) o nosso modelo com massas móveis apresenta prestações satisfatórias.

Em todas as manobras a força de propulsão nos motores "surge", é a mesma, estando os outros desactivados. Não se simulou os efeitos de "spin".

Em apêndice pode-se observar as simulações aqui descritas, assim como o código em C que implementa o modelo do AUV com massas móveis.

6. Conclusões e Perspectivas Futuras

A introdução do controlo através de massas móveis vem possibilitar uma melhoria de performance dos AUV´s face ao controlo só por superfícies e motores.

Ao incluirmos o controlo de atitude desta forma, estamos a poupar energia já que o efeito de "drag" e´ minimizado nas superfícies de controlo e não é necessário gastar tanta energia nos motores de posicionamento, para manter uma dada atitude. Podemos dizer que fazemos com que a força da gravidade "nos dê uma ajuda" para o movimento que queremos efectuar. É verdade que ao incluirmos massas móveis estamos a aumentar a complexidade do modelo, no entanto podemos colher os benefícios mais tarde. Podemos anular os efeitos da impulsão e da gravidade e ficamos assim em muito melhores condições para fazer a linearização da planta.

Ao subdividirmos o problema de controlo de uma trajectória em manobras estamos a aproveitarmo-nos do facto de o controlo activo de forças de restabelecimento nos facilitar a linearização do modelo e logo o controlo.

Pensamos que a arquitectura proposta serve os propósitos de permitir um bom controlo do AUV e simultaneamente é adequada a um veículo onde podemos usar massas móveis no tipo de controlo referido.

Há que estudar esquemas para efectuar um controlo eficiente das massas móveis e avaliar a influência negativa no comportamento do veículo, que esse controlo terá.

Pensamos no entanto que este trabalho constitui um bom ponto de partida para estudos futuros no campo aliciante do controlo de AUV´s e que algumas ideias aqui avançadas podem ser promissoras no futuro.

7. Bibliografia

[Fossen 91] Thor Inge Fossen, "*Nonlinear modelling and control of underwater vehicles*", Ph. D. Disert., Norwegian Institute of Technology, June 1991.

[Fossen 93] Thor Inge Fossen, "*An introduction to nonlinear modelling, stability and control of underwater vehicles*",Tech. Report. , Norwegian Institute of Technology, 1993.

[Slotine 91] Jean-Jacques E. Slotine, Weiping Li, "*Aplied Nonlinear Control*", Prentice Hall, 1991

[Maciej 89] J. M Maciejowski, "*Multivariable feedback design*", Addison Wesley, 1989

[Cristi 90] Roberto Cristi, Fotis Papoulias, Anthony Healey, "*Adaptive sliding mode control of autonomous vehicles in the dive plane*", IEEE Journ. Oceanic Eng., vol 15, no3, Jul. 1990

[Almeida 93] Alfredo Martins, Jose Almeida, "*Redes neuronais em controlo - um estado da arte*", 1993

[Silvestre 91] Carlos Silvestre, "*Modelação e controlo de veículos submarinos autónomos*", Tese de Mestrado, IST, Junho de 1991

[Ljung 87] Lennart Ljung, "*System identification*", Prentice Hall, 1987

Porto Setembro de 1993

José Miguel Soares de Almeida

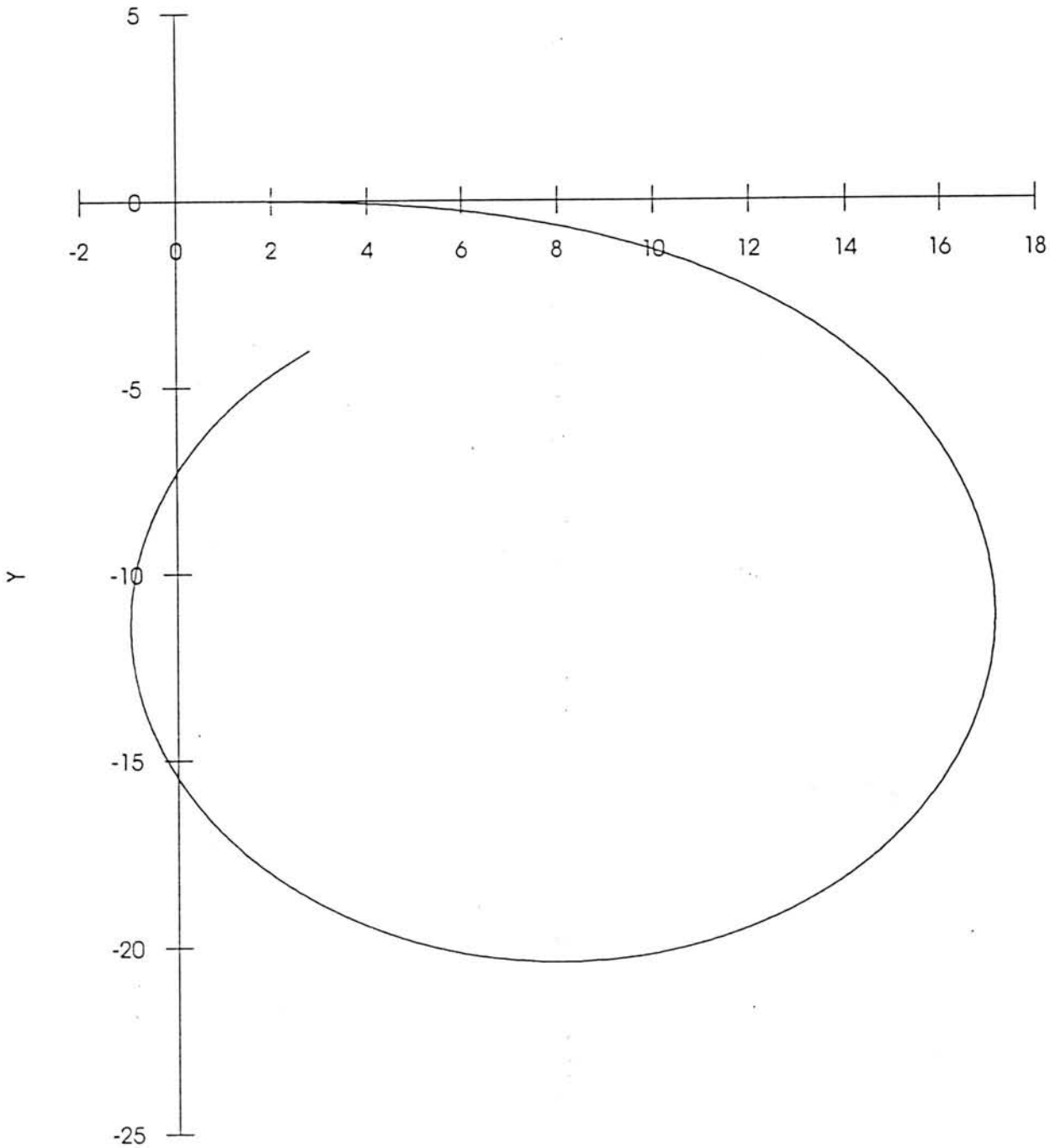
ANEXO I

Resultados de sinulação

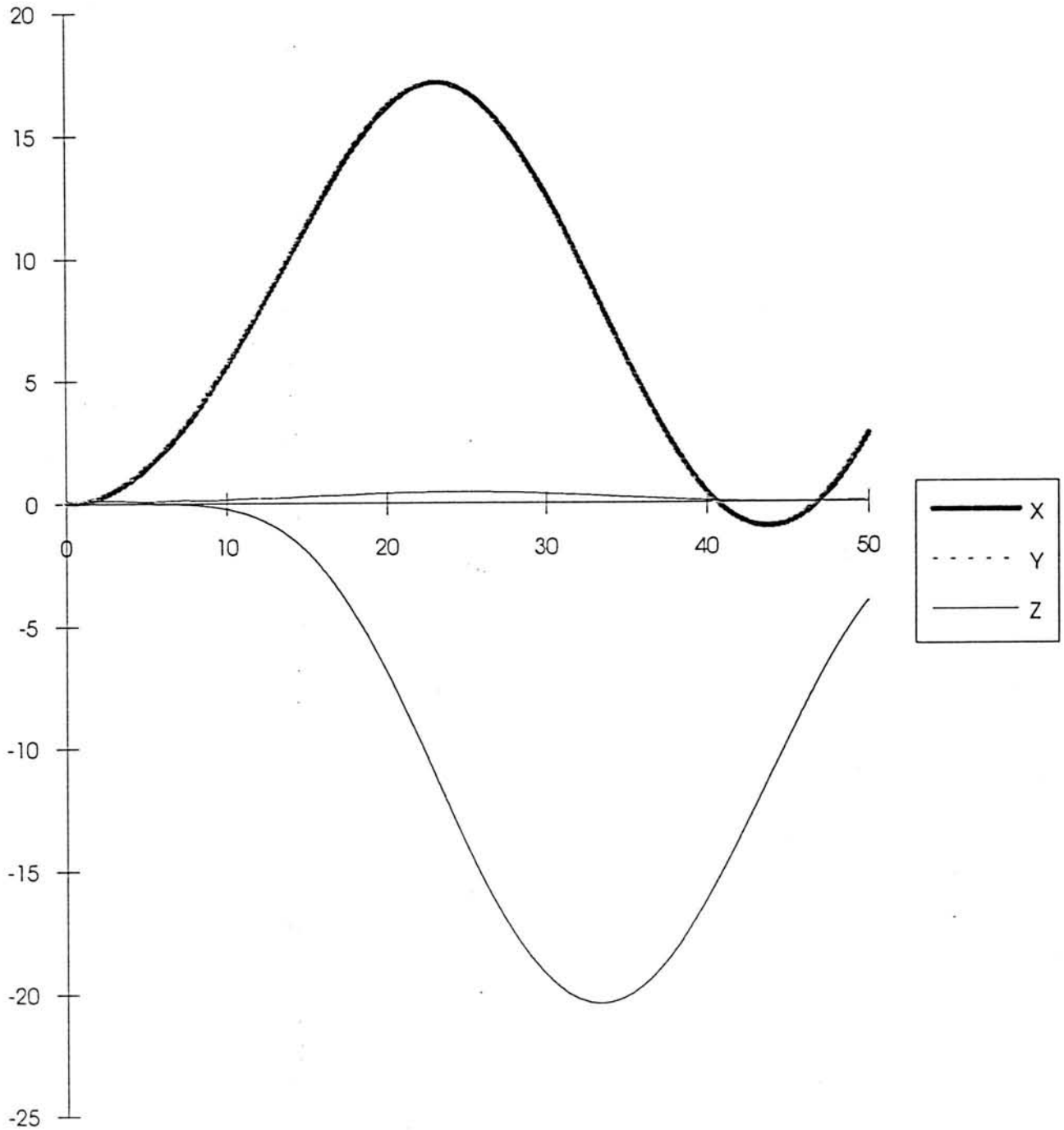
MANOBRA 1

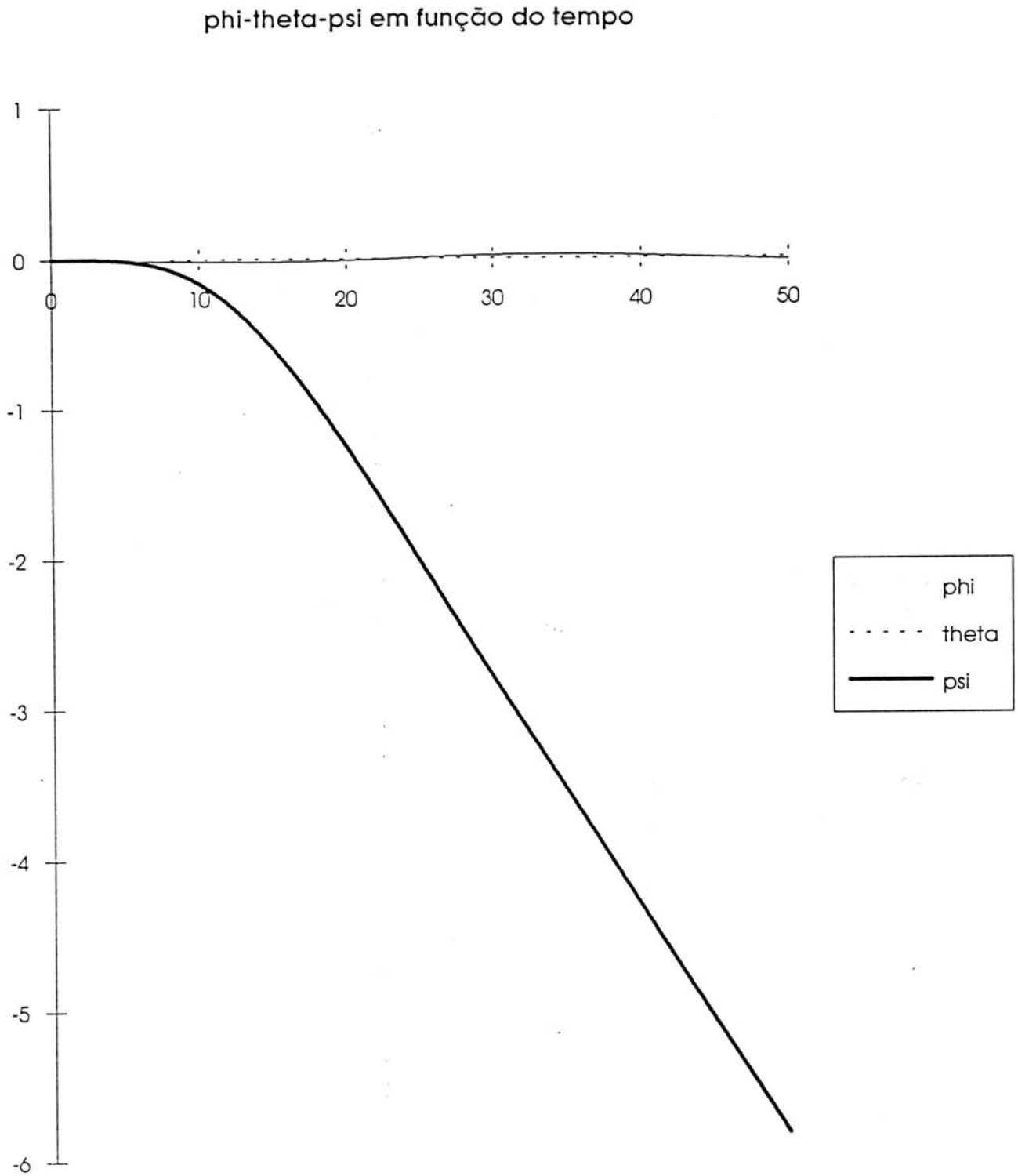
(COM MASSAS MOVEIS)

X-Y



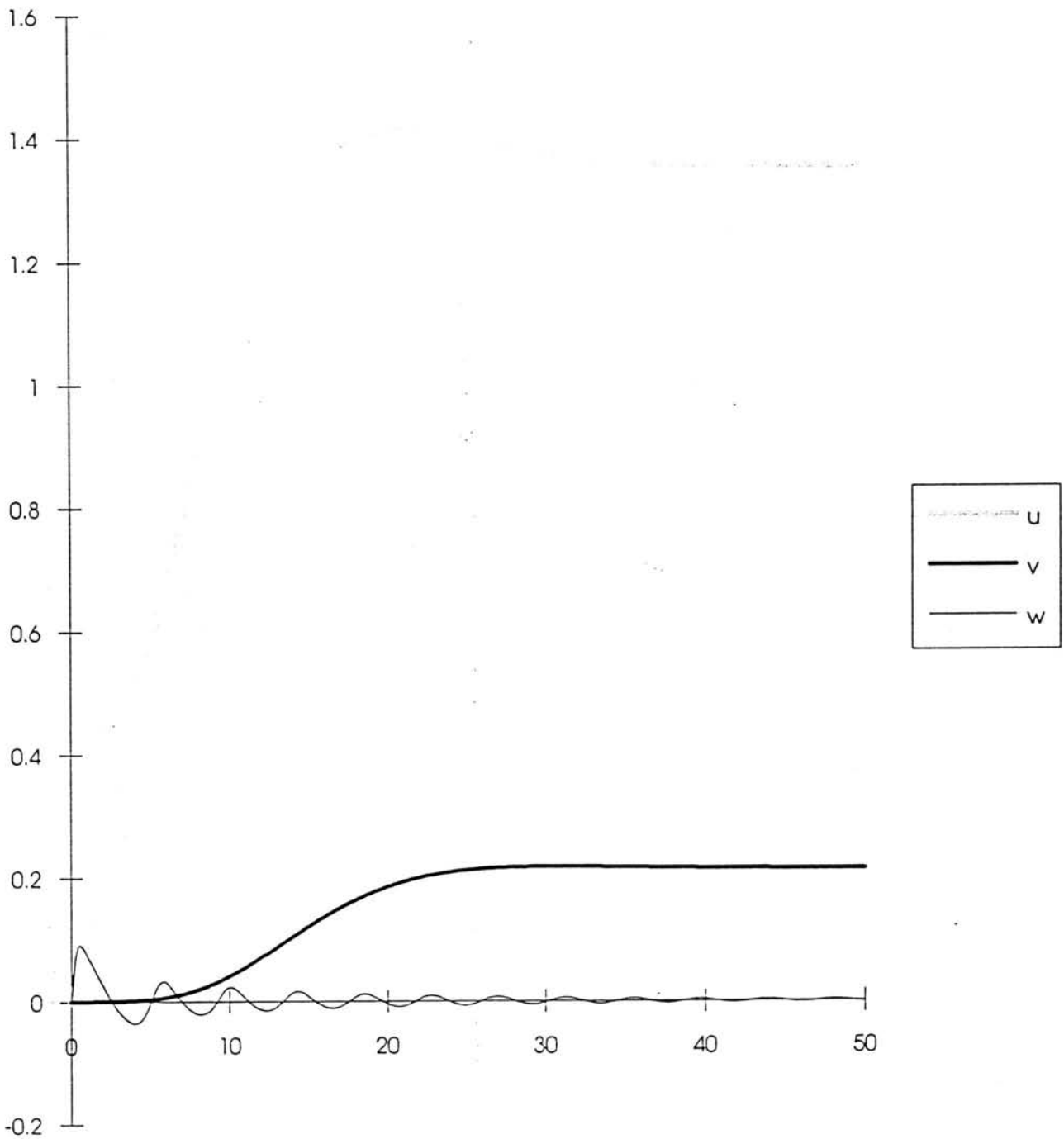
XYZ em função do tempo




phi-theta-psi em função do tempo
1
0
-1
-2
-3
-4
-5
-6
0
10
20
30
40
50
phi
theta
psi

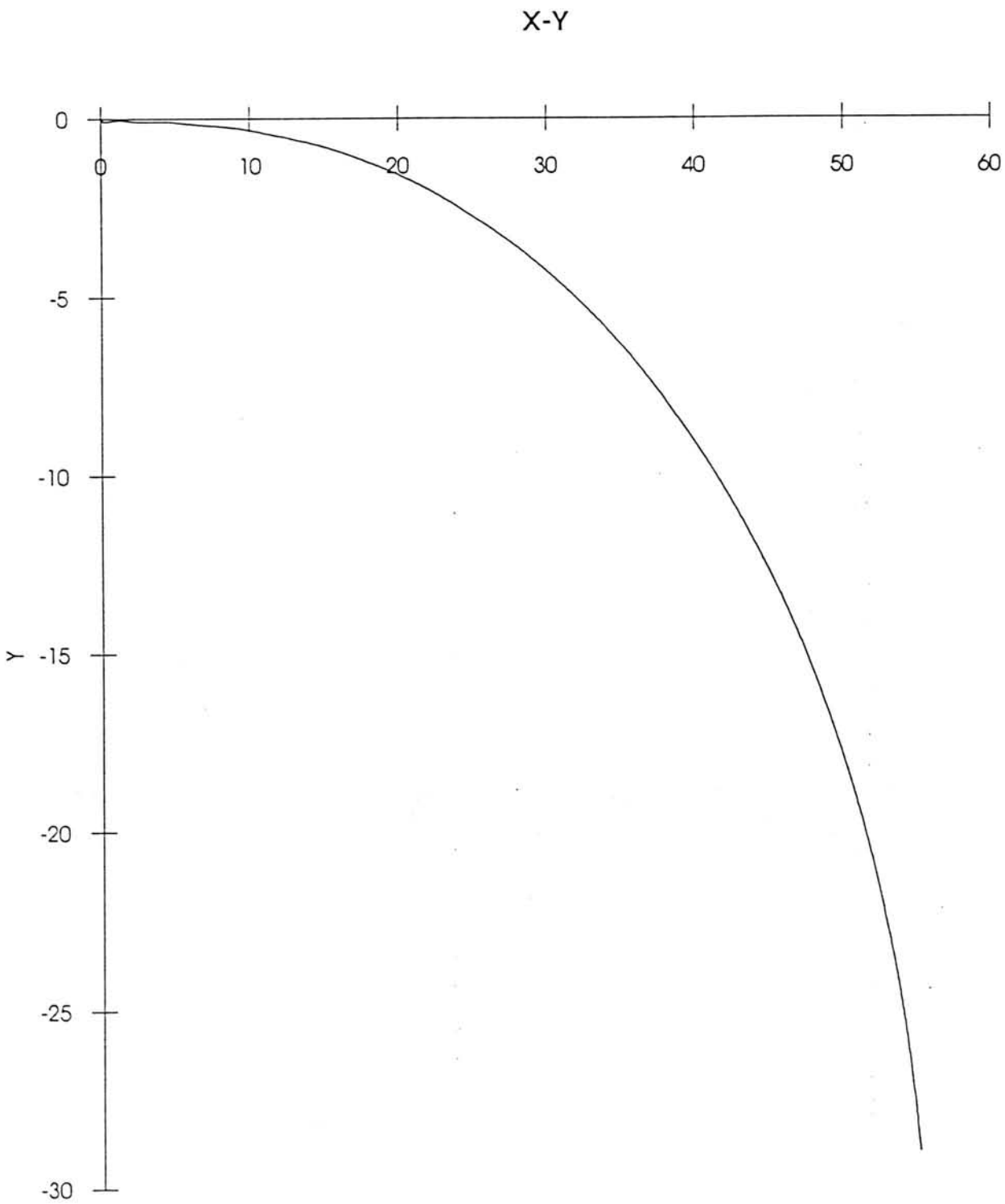

p-q-r em função do tempo
0.02
0
-0.02
-0.04
-0.06
-0.08
-0.1
-0.12
-0.14
-0.16
0
10
20
30
40
50
p
q
r

u-v-w em função do tempo

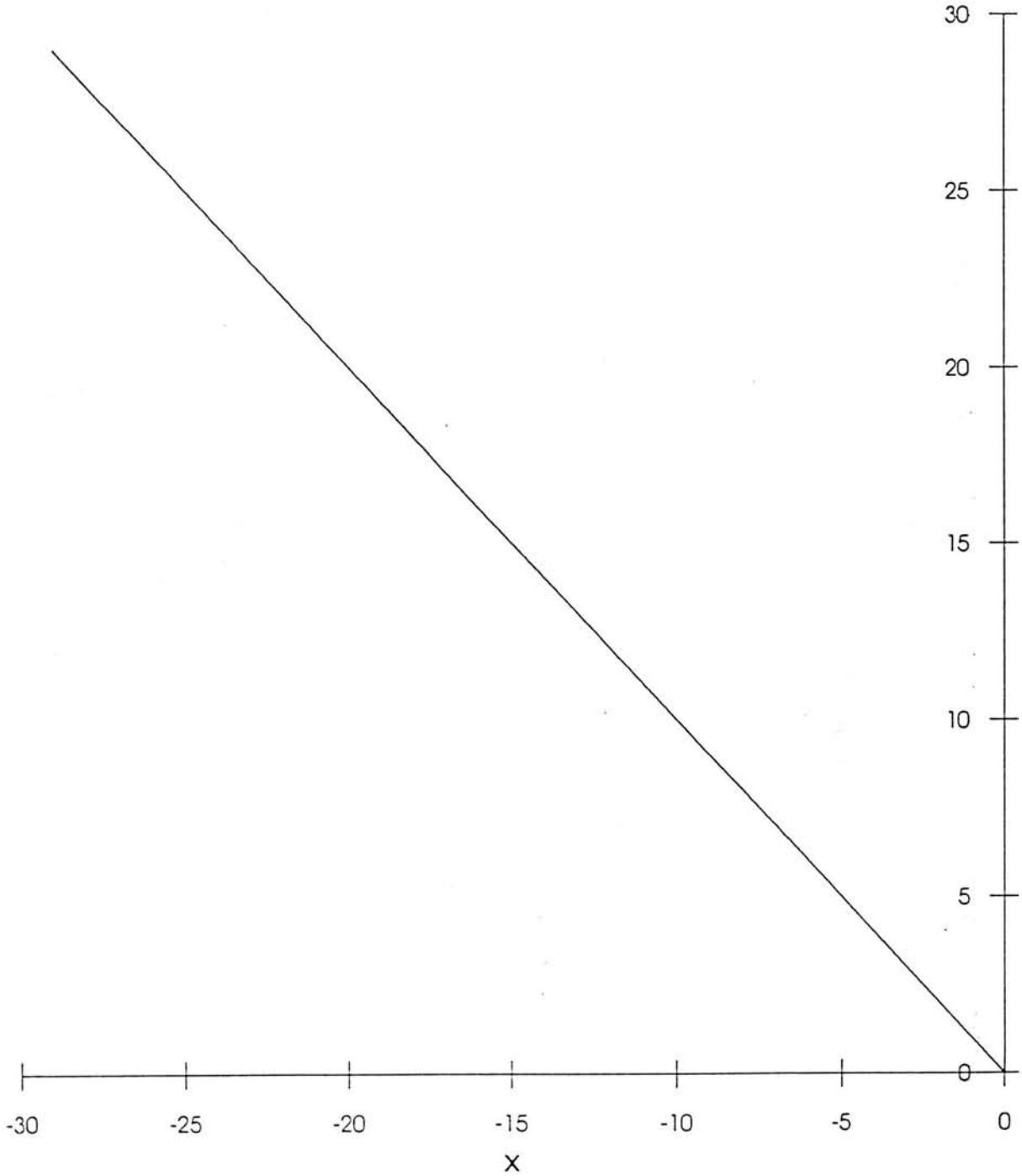


MANOBRA 2

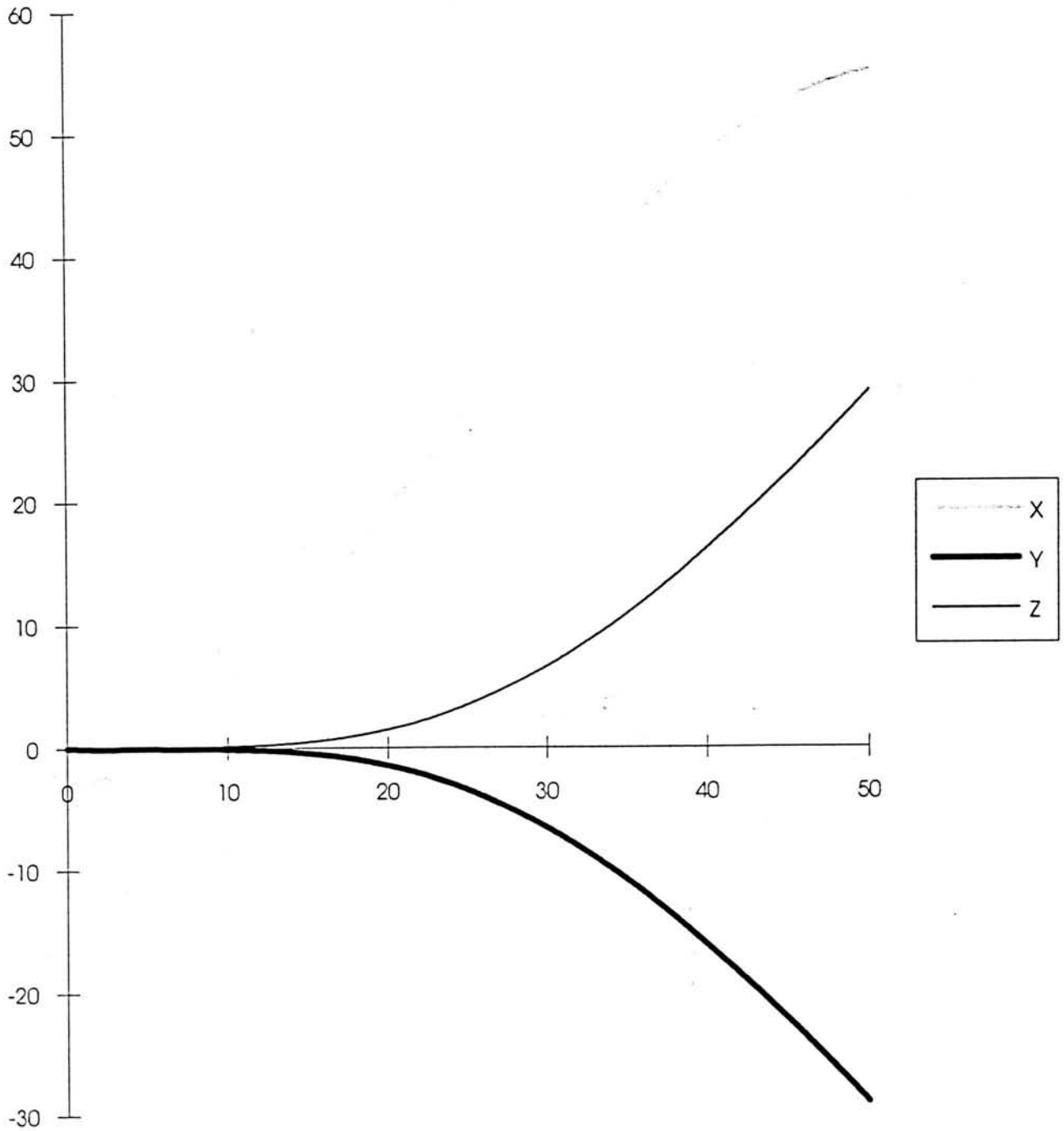
(COM MASSAS MOVEIS)


X-Y
0
0
10
20
30
40
50
60
-5
-10
Y -15
-20
-25
-30

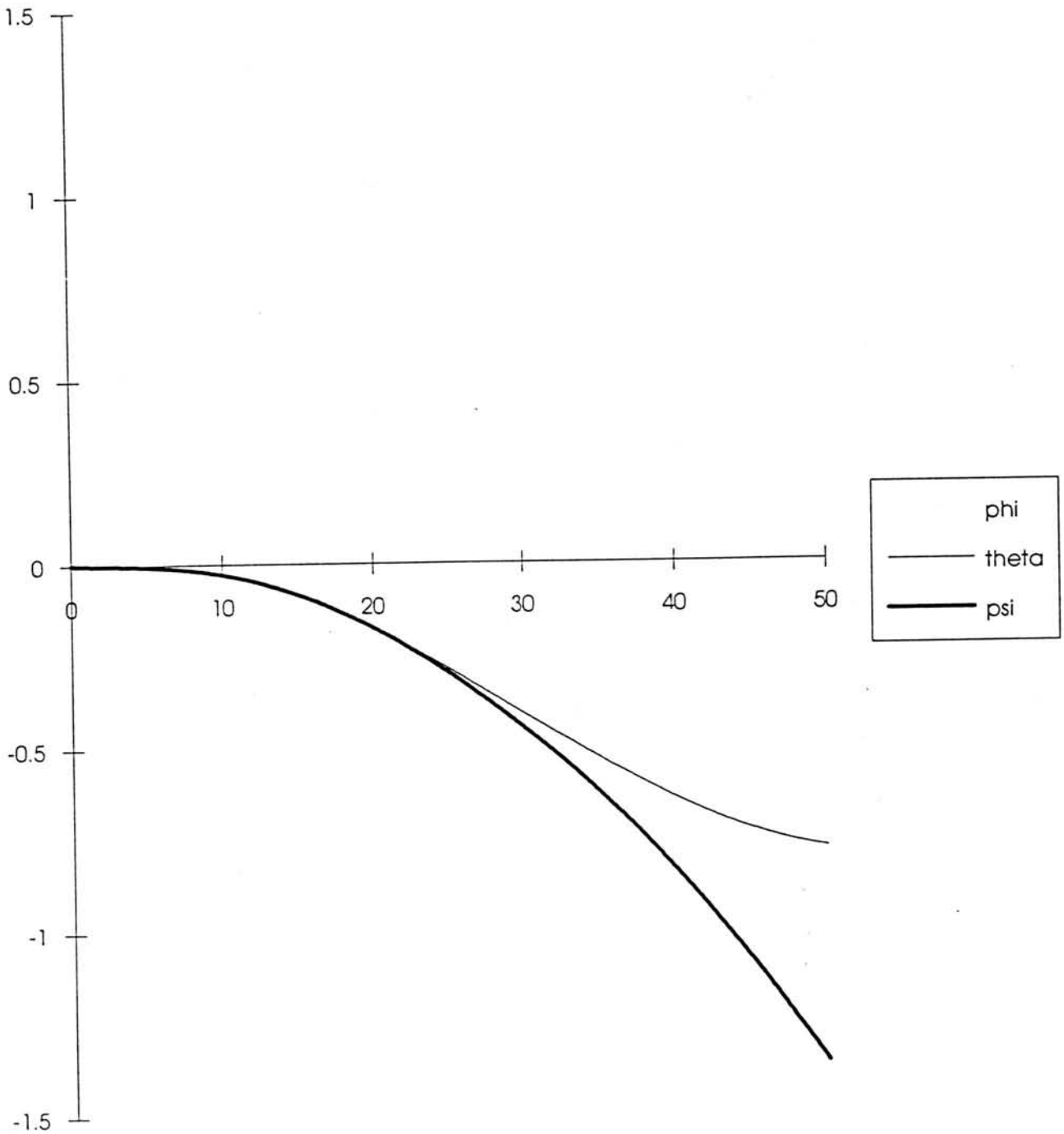
Y-Z



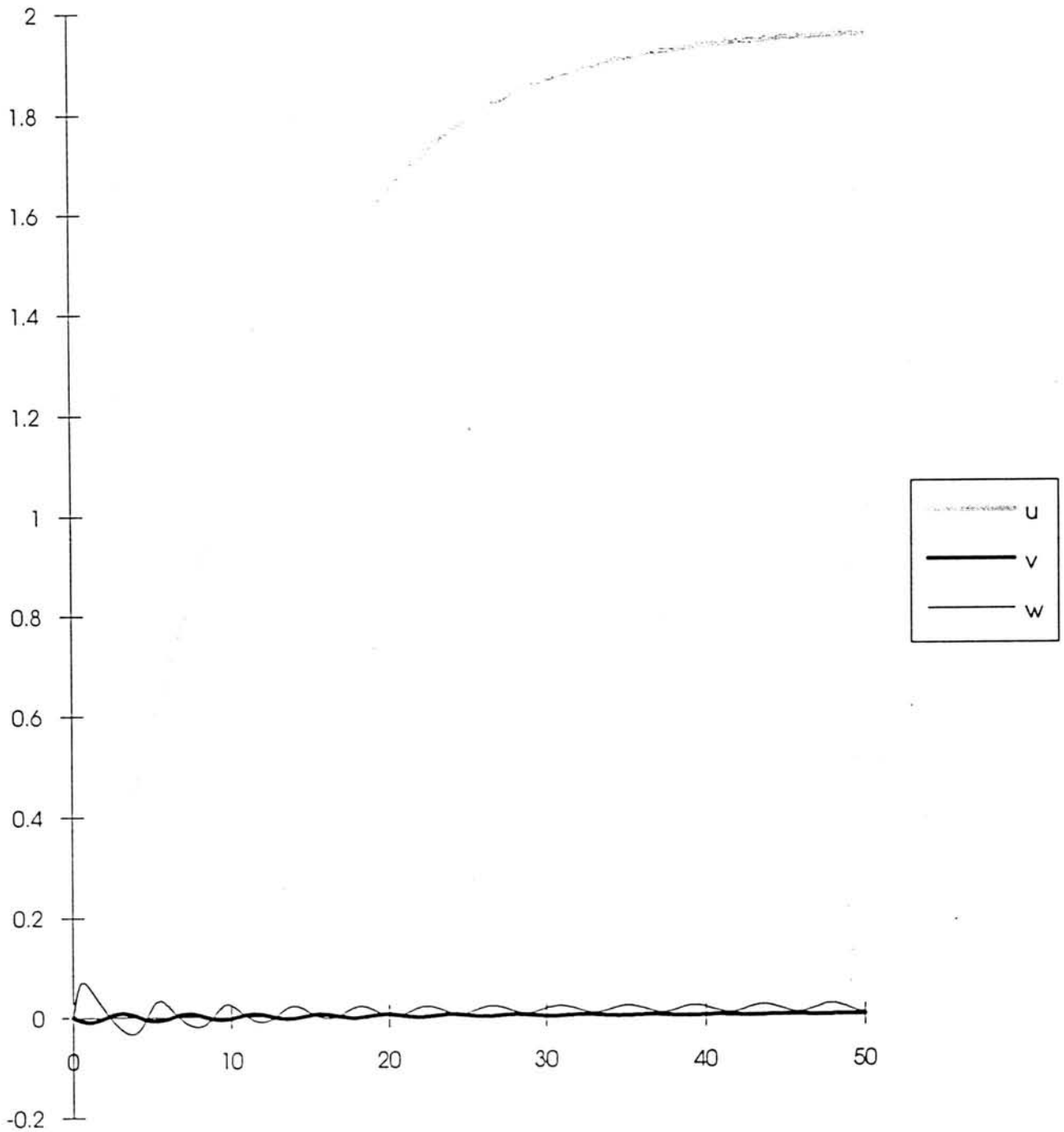
XYZ em função do tempo



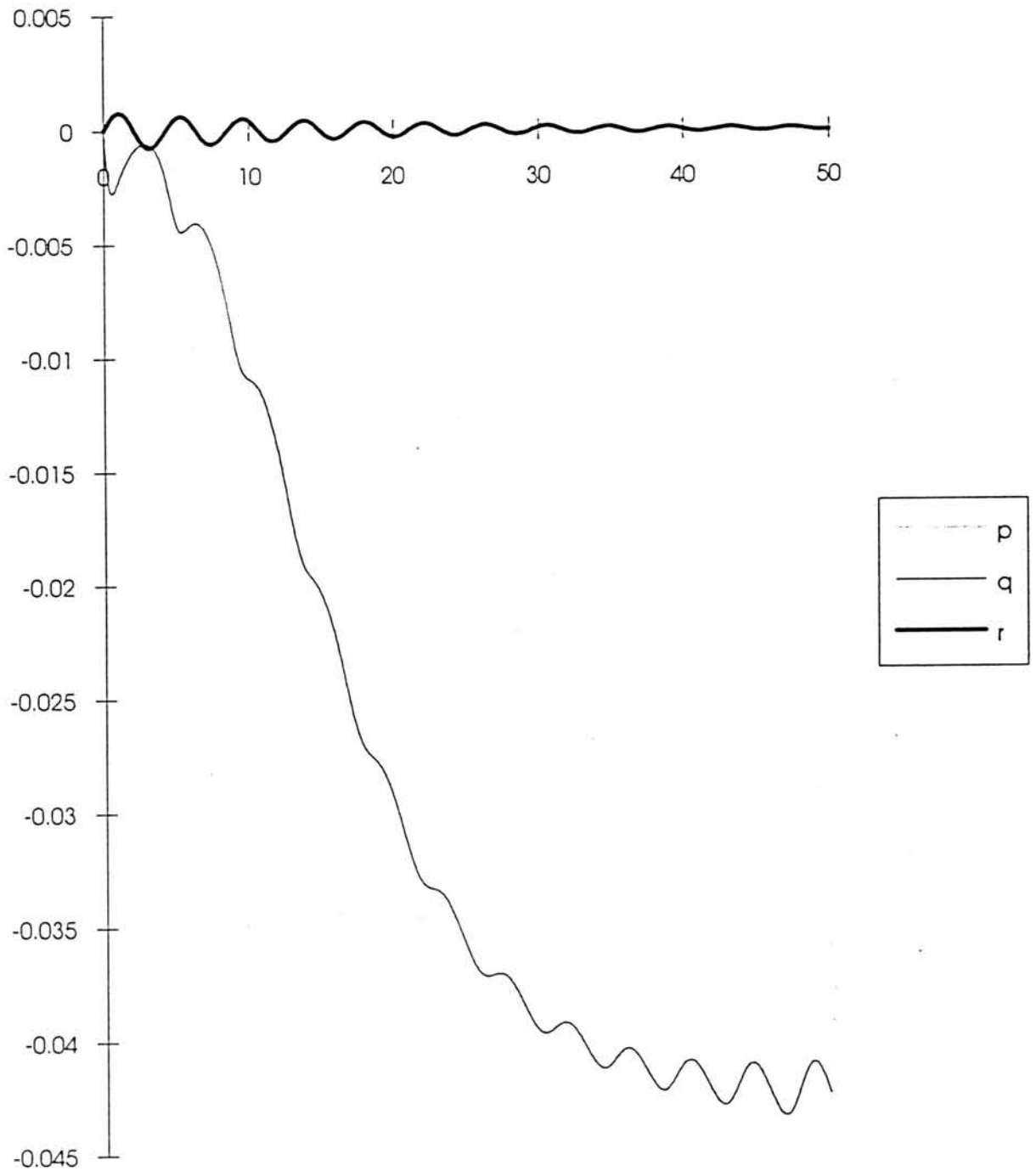
phi-theta-psi em função do tempo



u-v-w em função do tempo



p-q-r em função do tempo

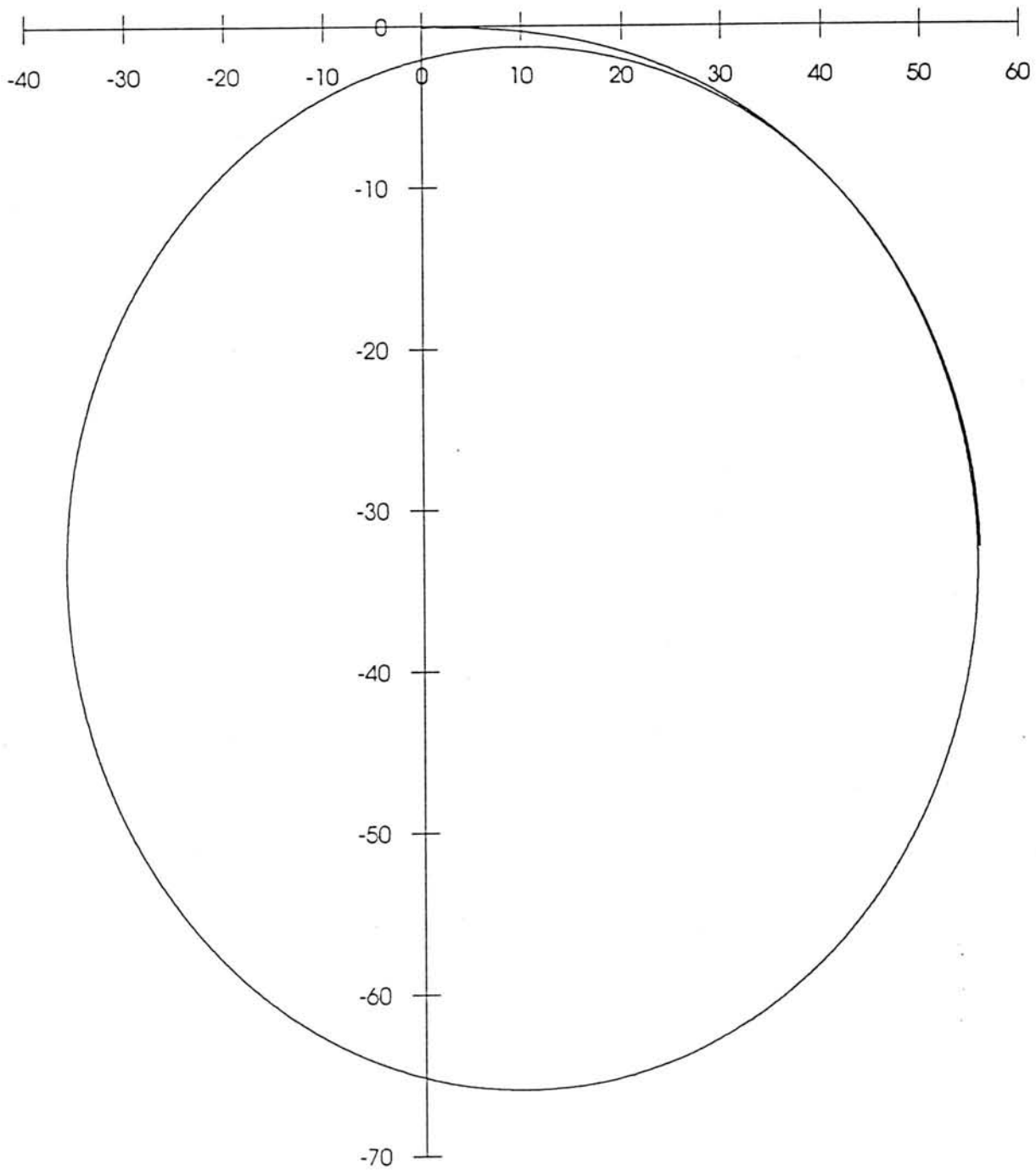


MANOBRA 2

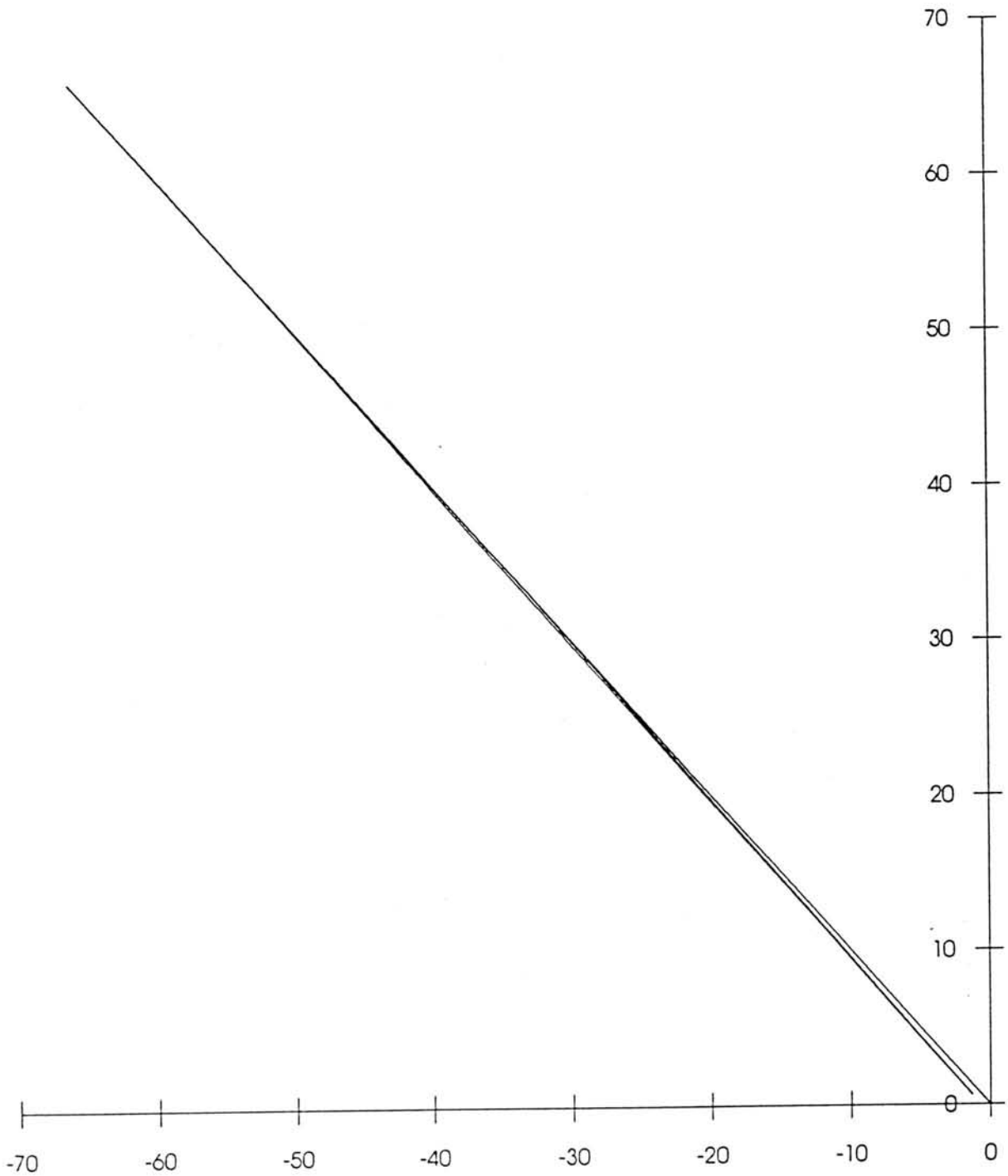
(COM MASSAS MOVEIS)

LONGA DURAÇÃO

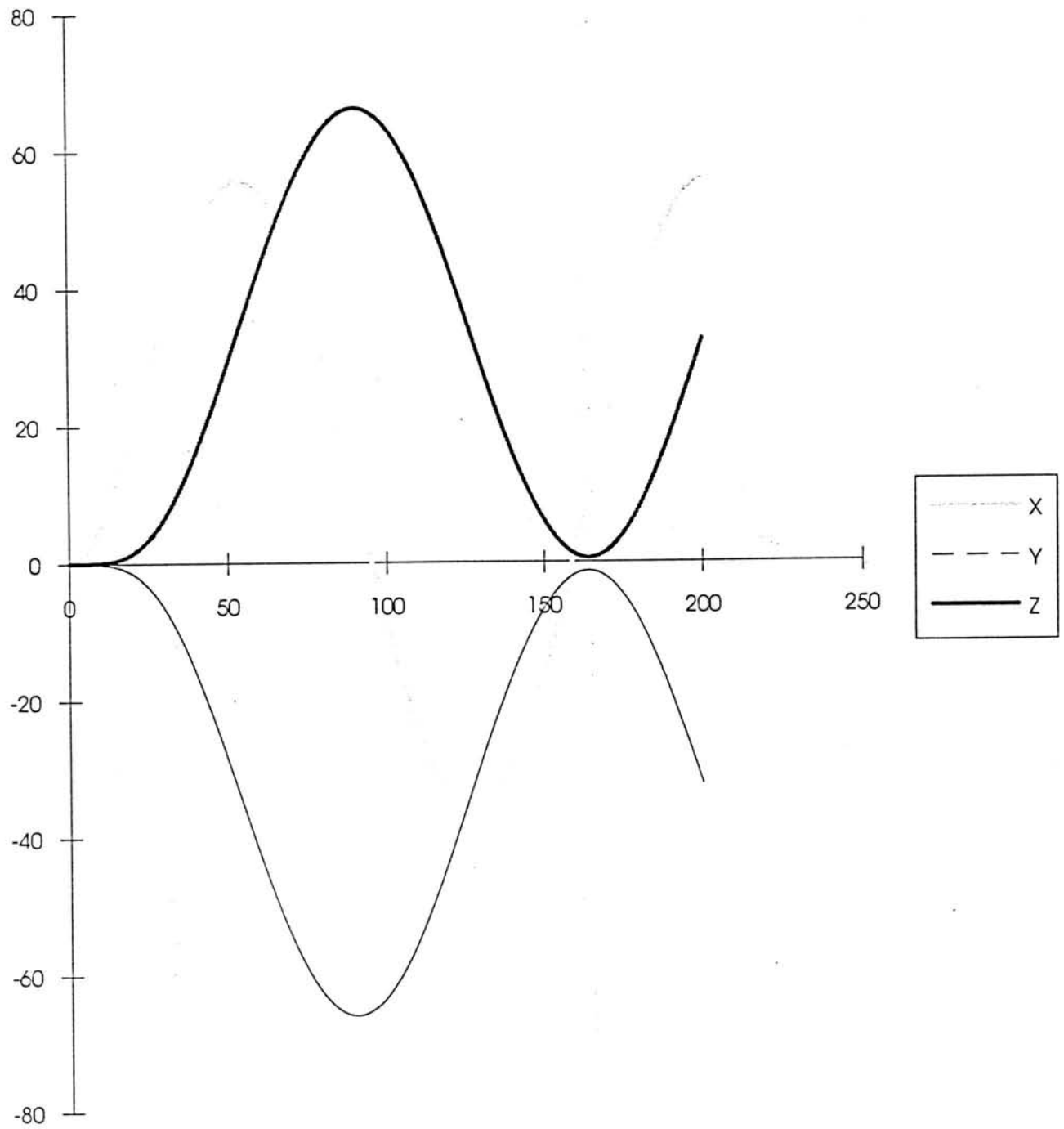
X-Y



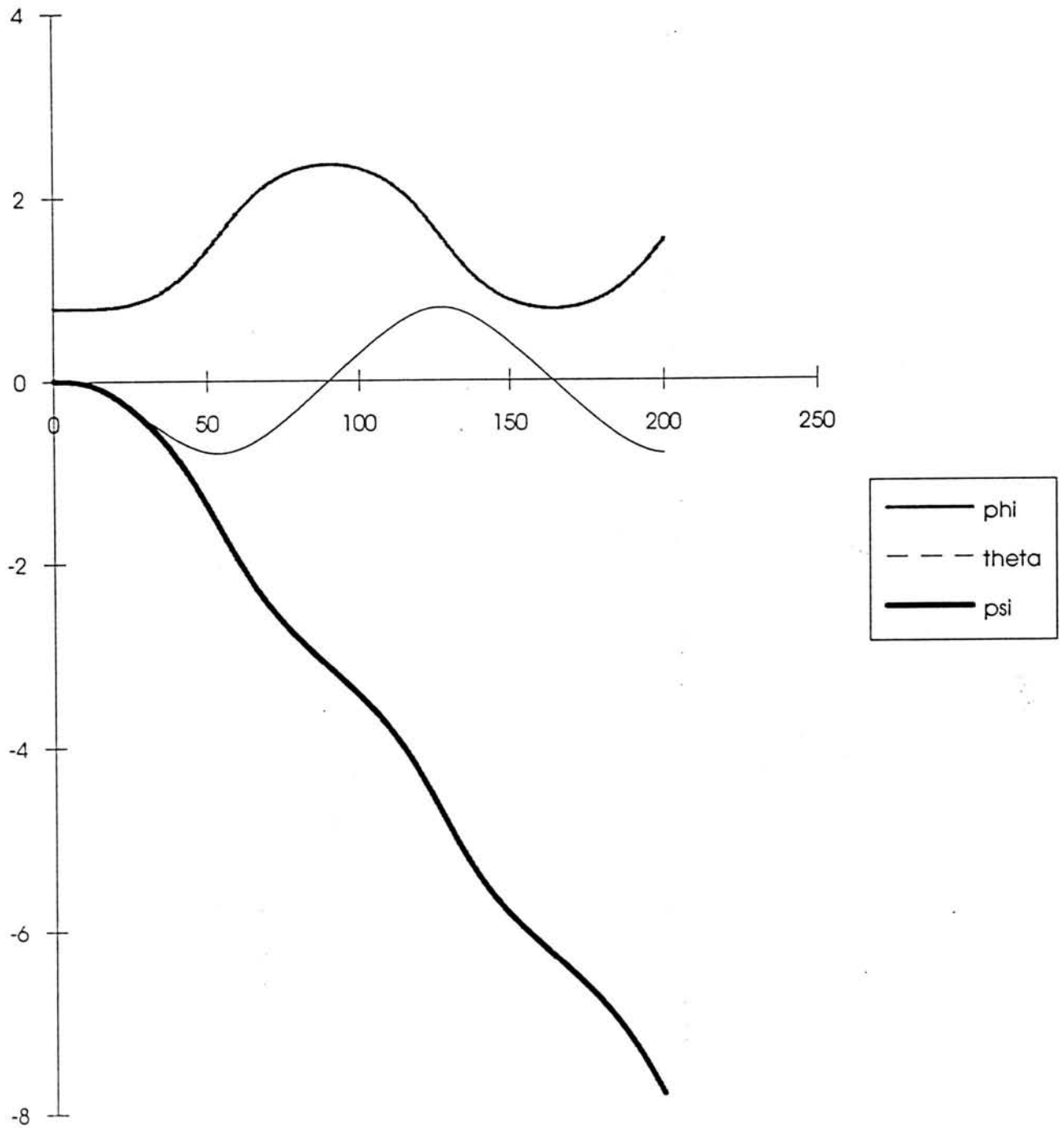
Y-Z



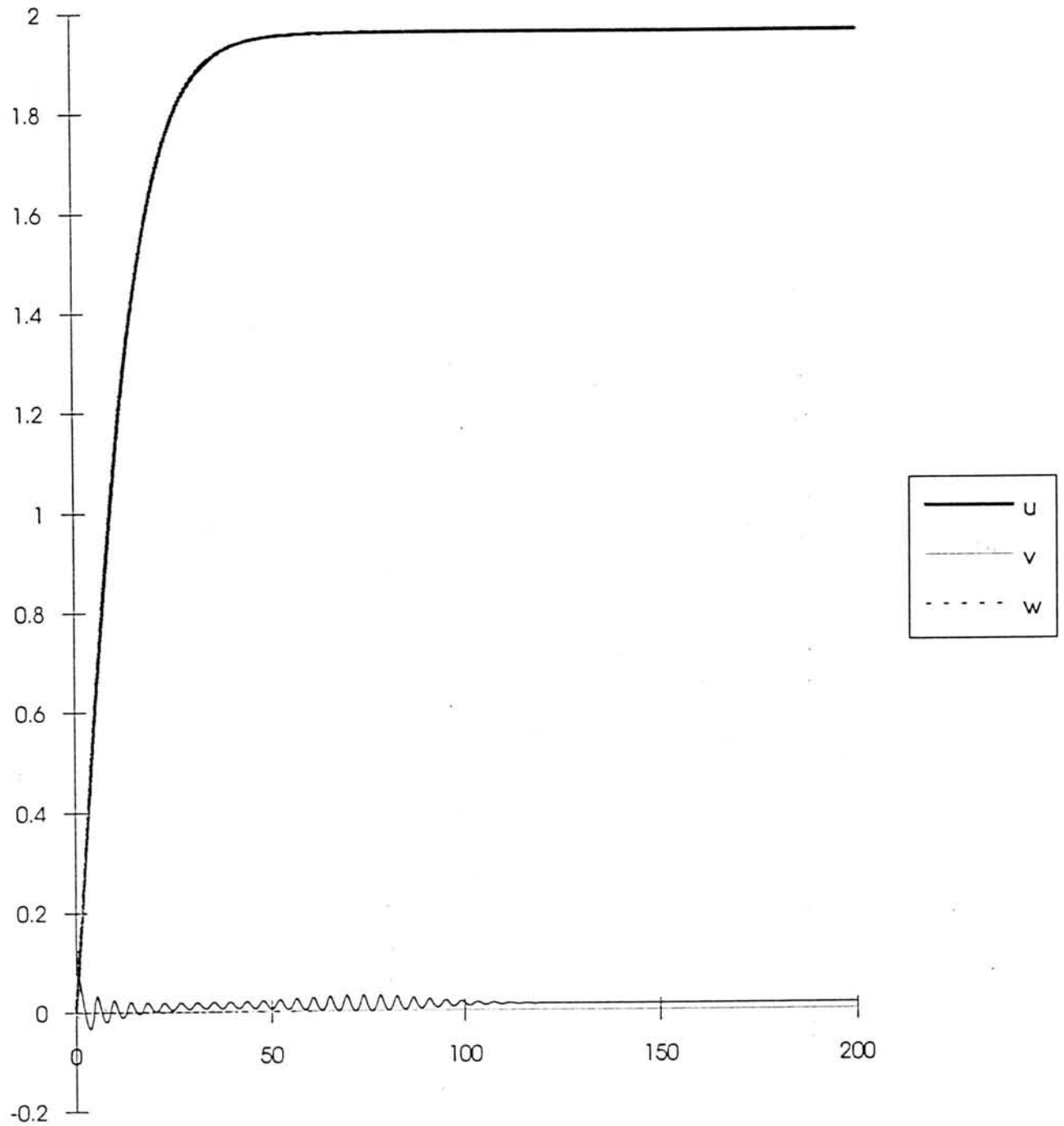
X-Y-Z em função do tempo



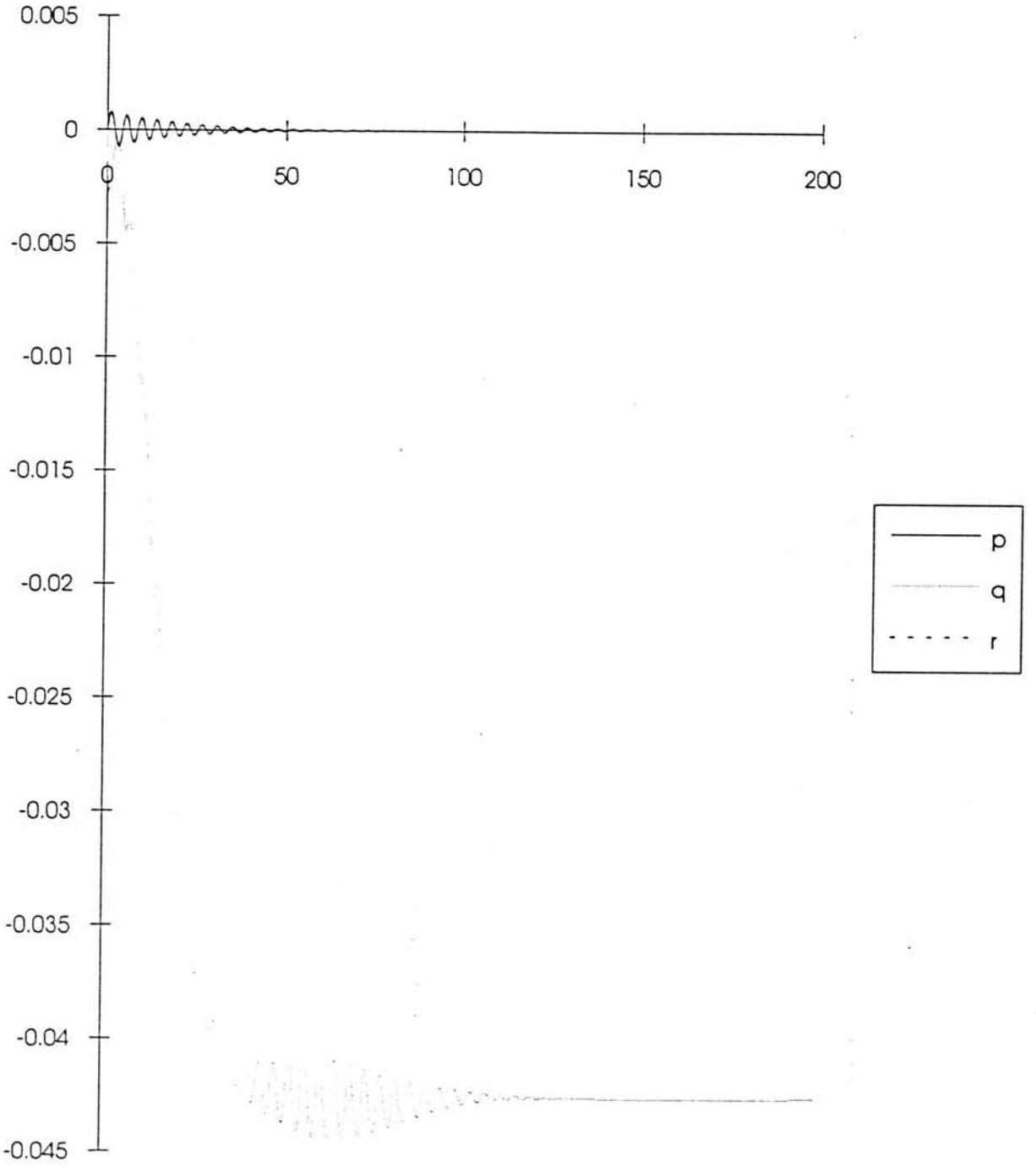
phi-theta-psi em função do tempo



u-v-w em função do tempo



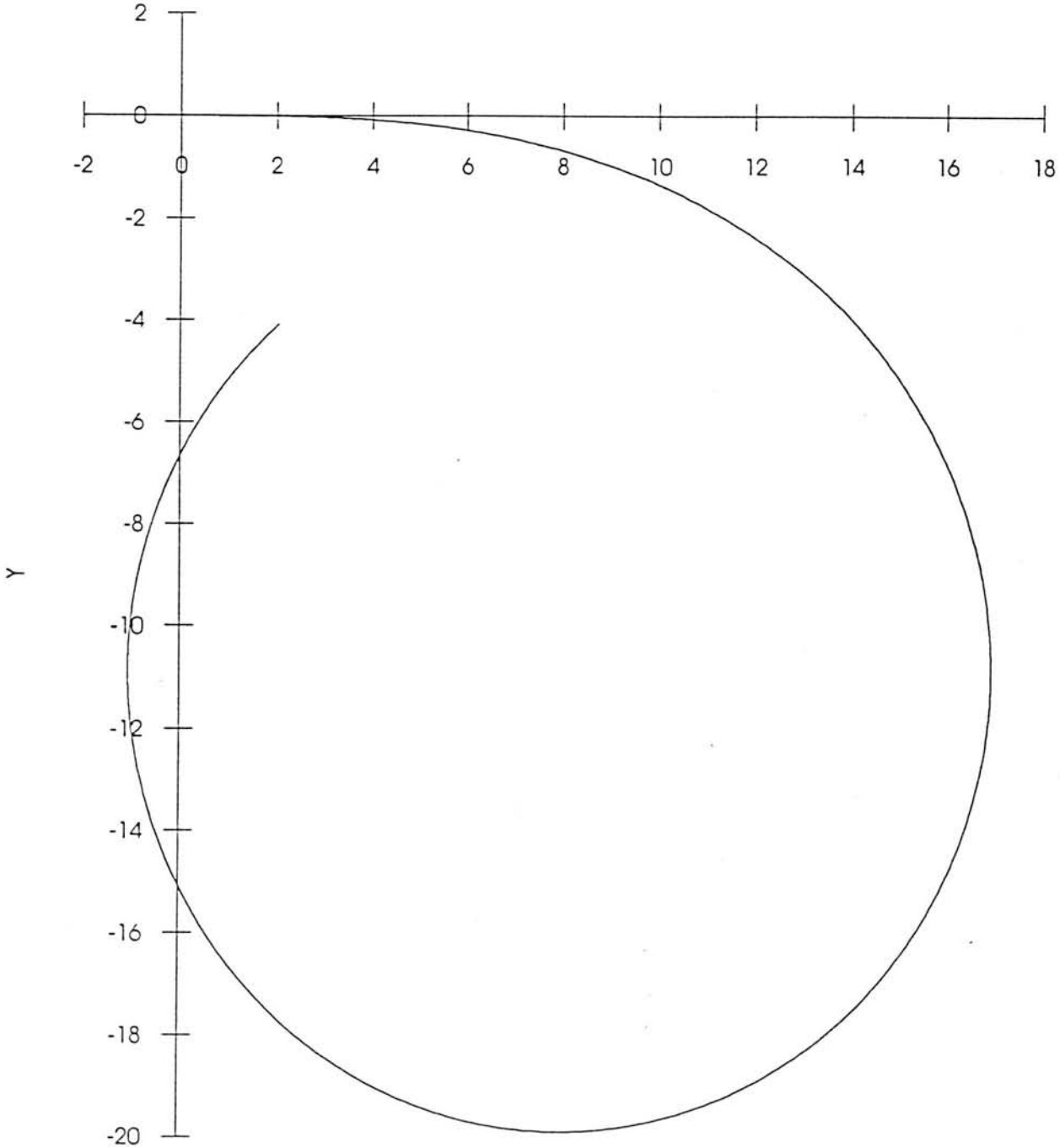
p-q-r em função do tempo



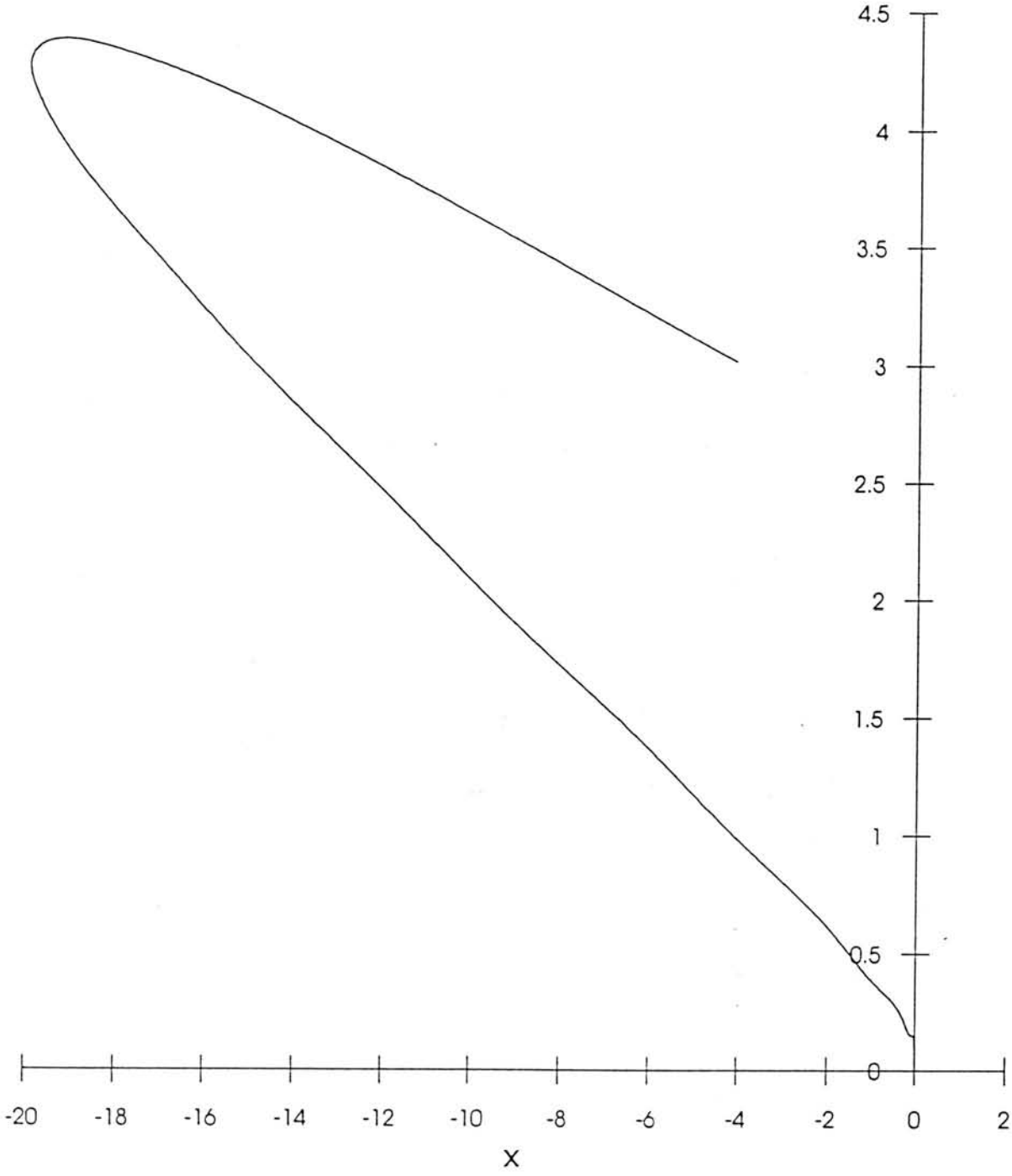
MANOBRA 3

(COM MASSAS MOVEIS)

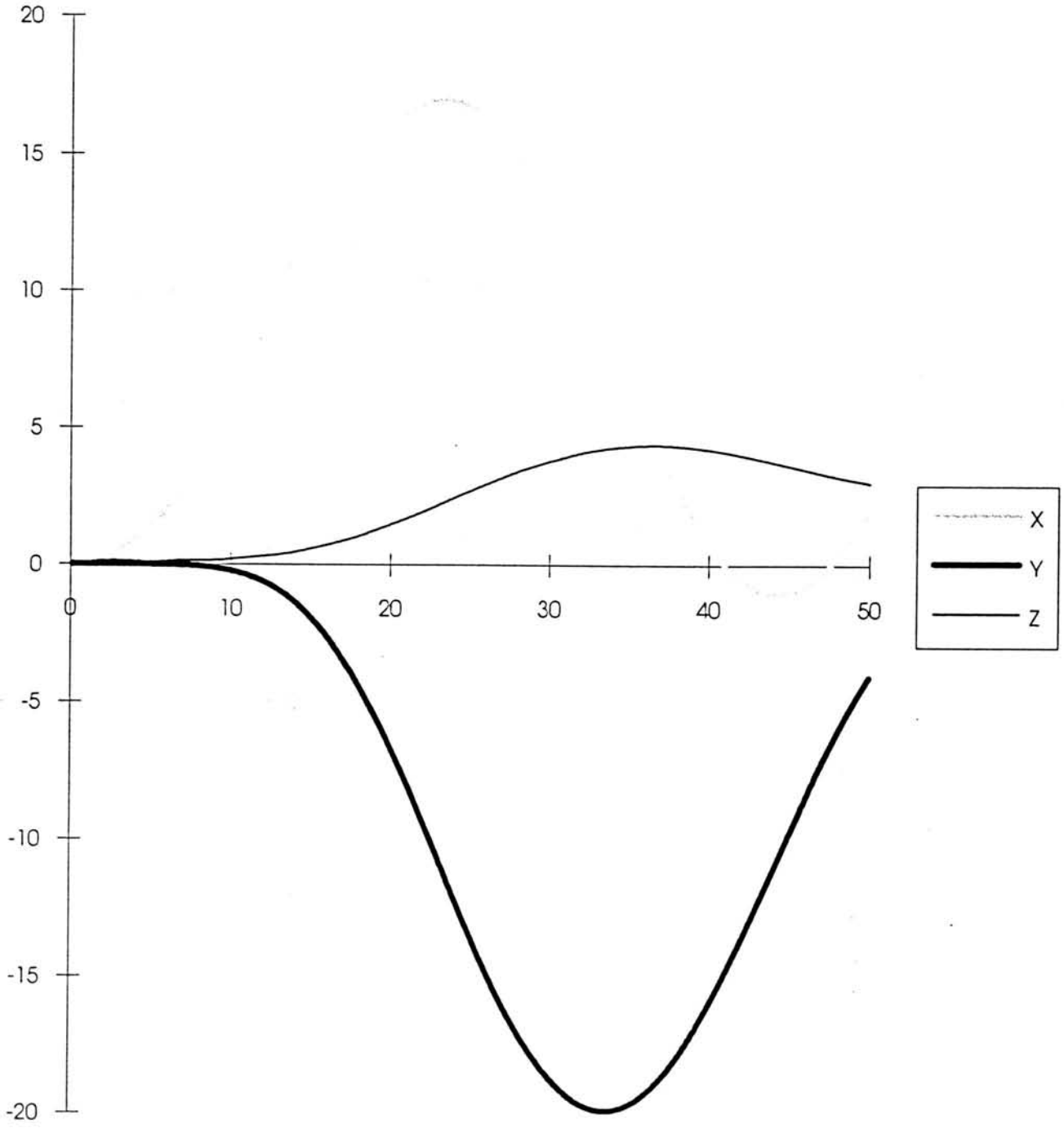
X-Y

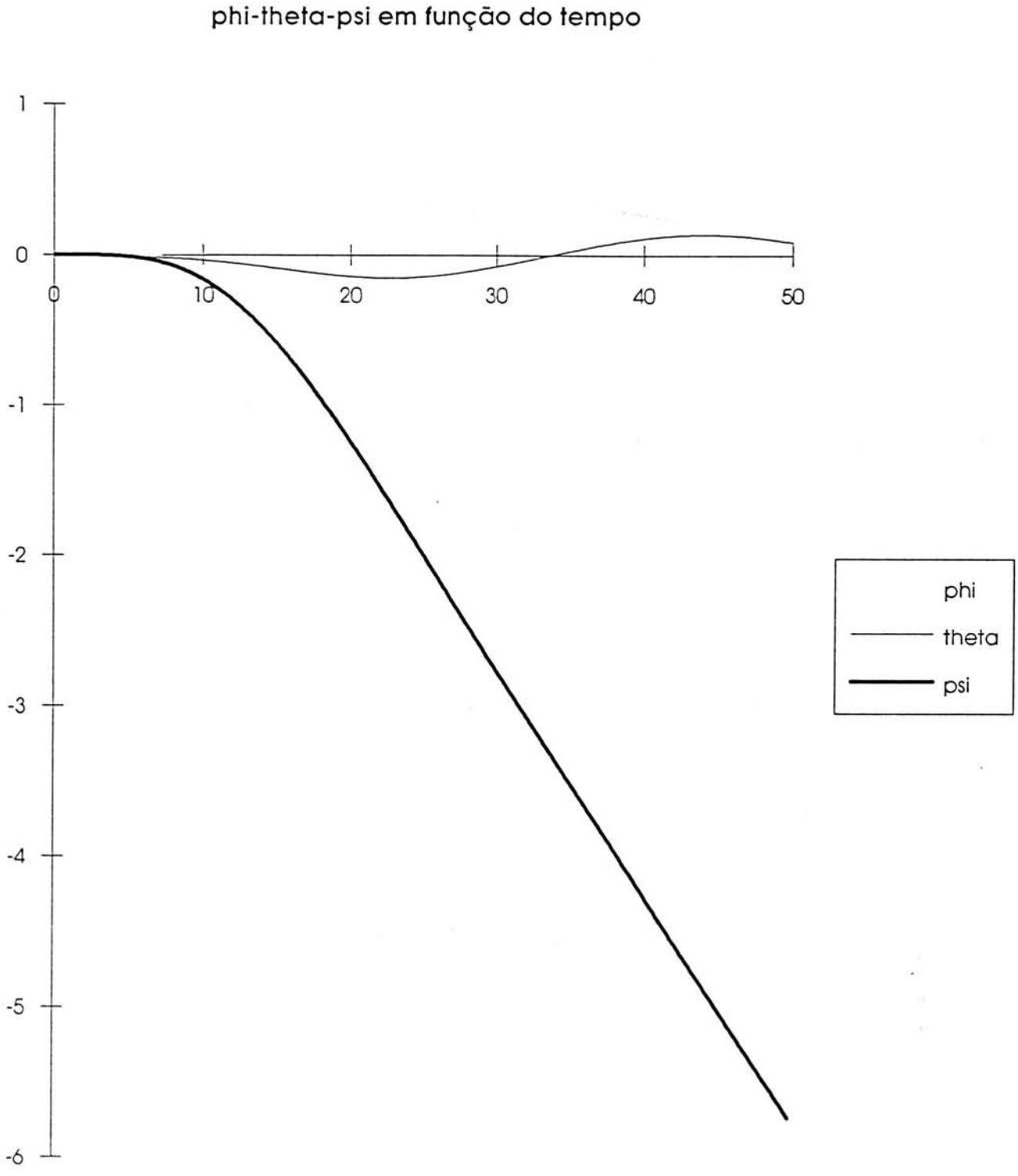


Y-Z

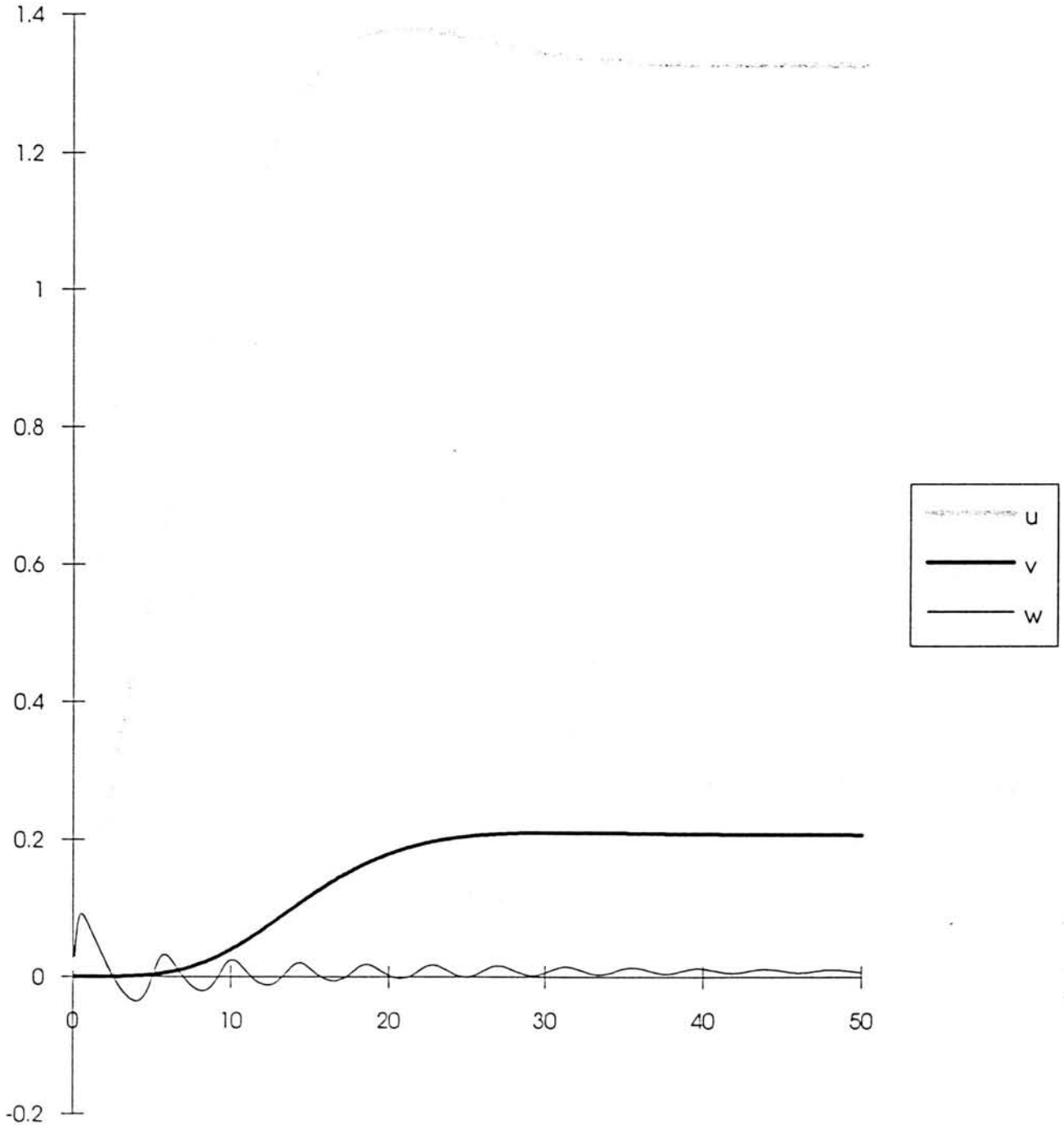


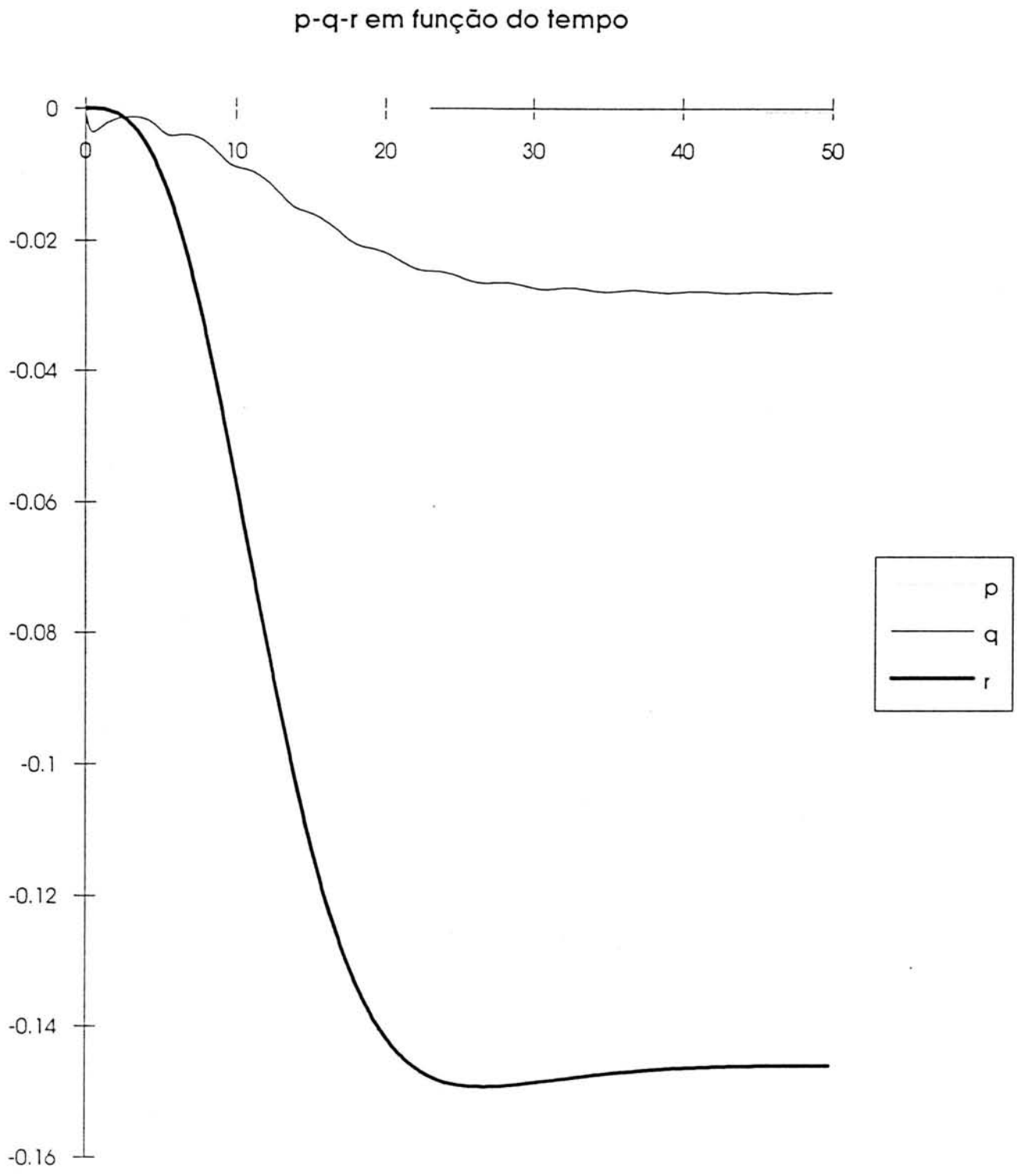
XYZ em função do tempo




phi-theta-psi em função do tempo
1
0
-1
-2
-3
-4
-5
-6
0
10
20
30
40
50
phi
theta
psi

u-v-w em função do tempo

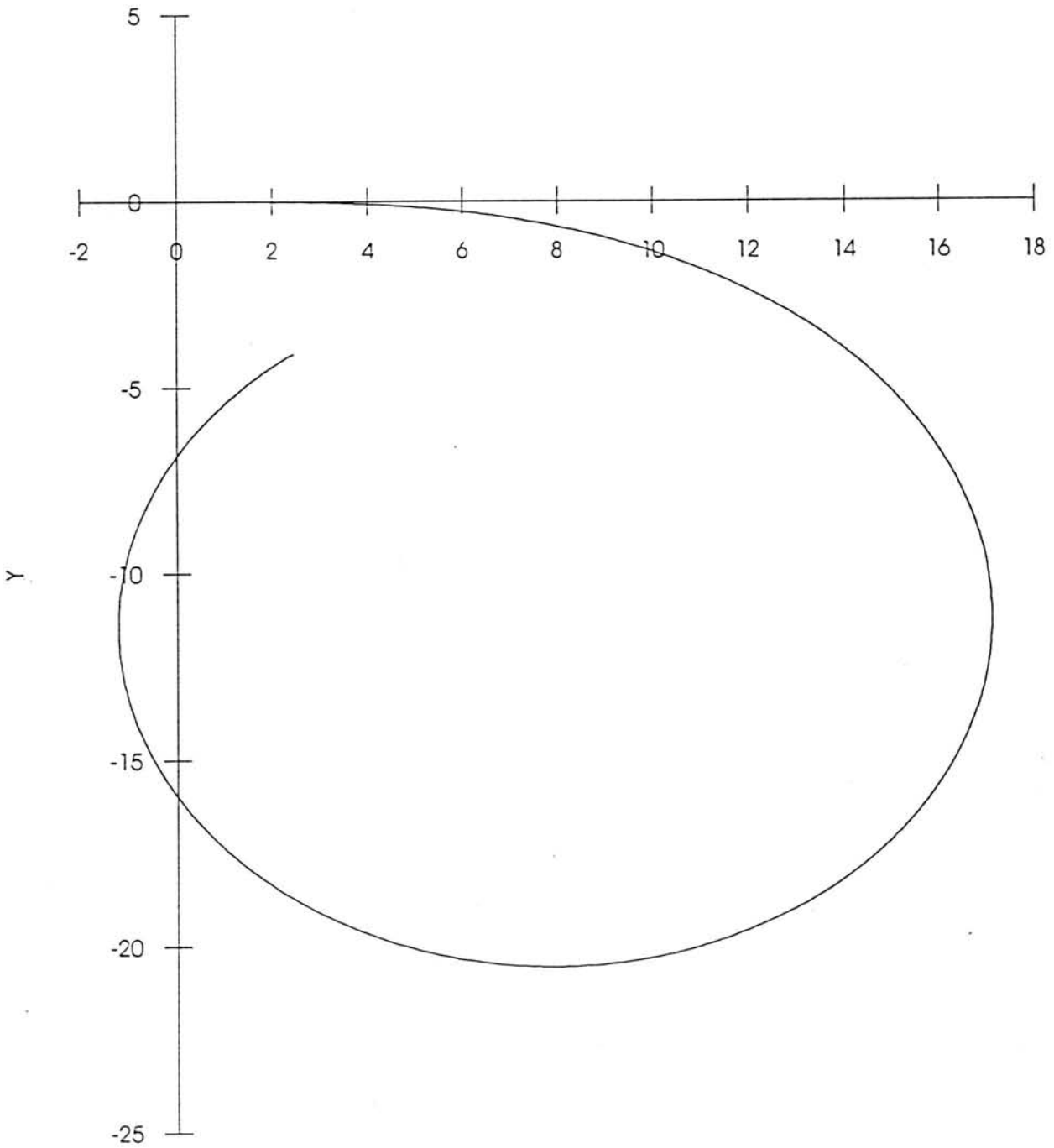


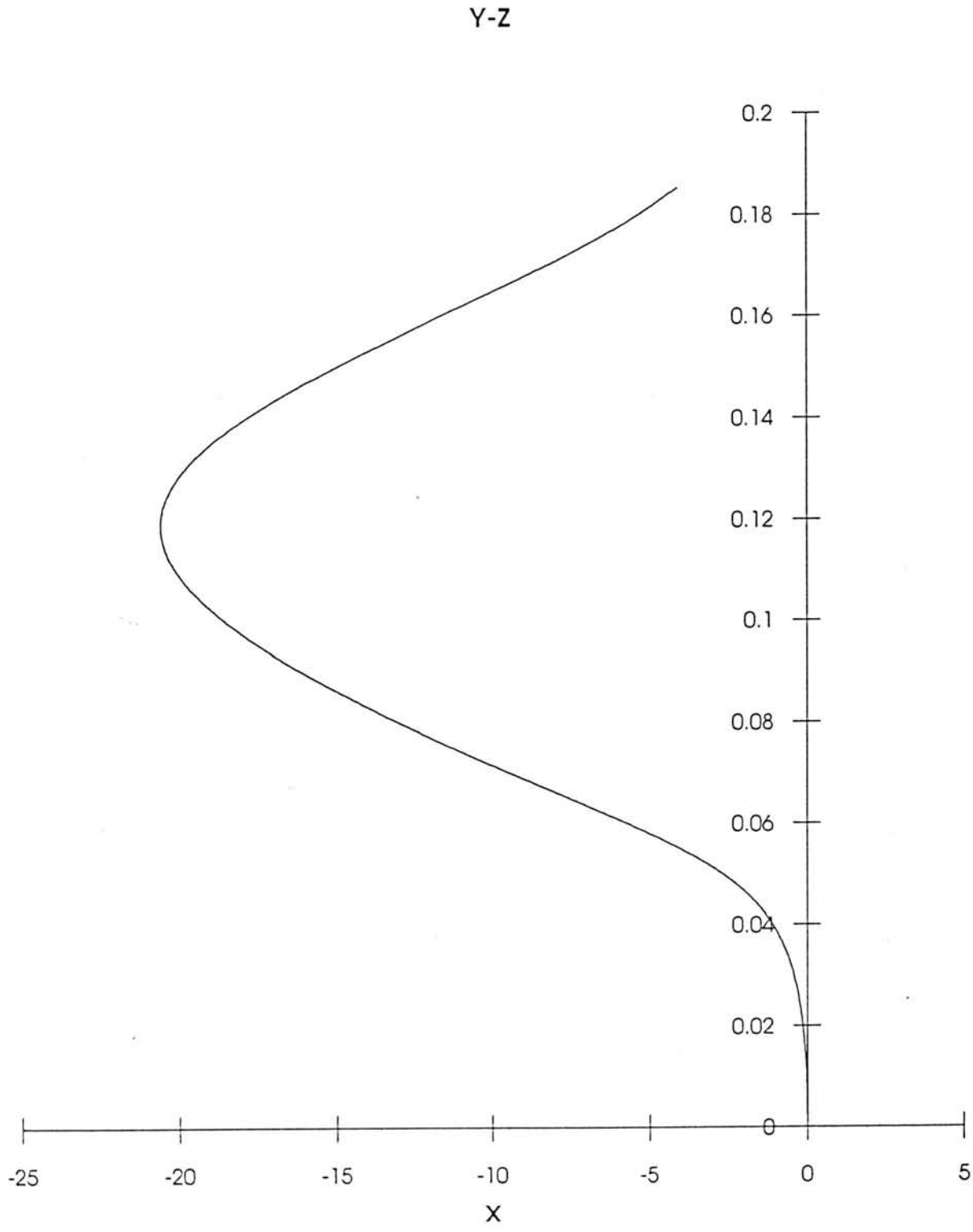

p-q-r em função do tempo
0
-0.02
-0.04
-0.06
-0.08
-0.1
-0.12
-0.14
-0.16
0
10
20
30
40
50
p
q
r

MANOBRA 1

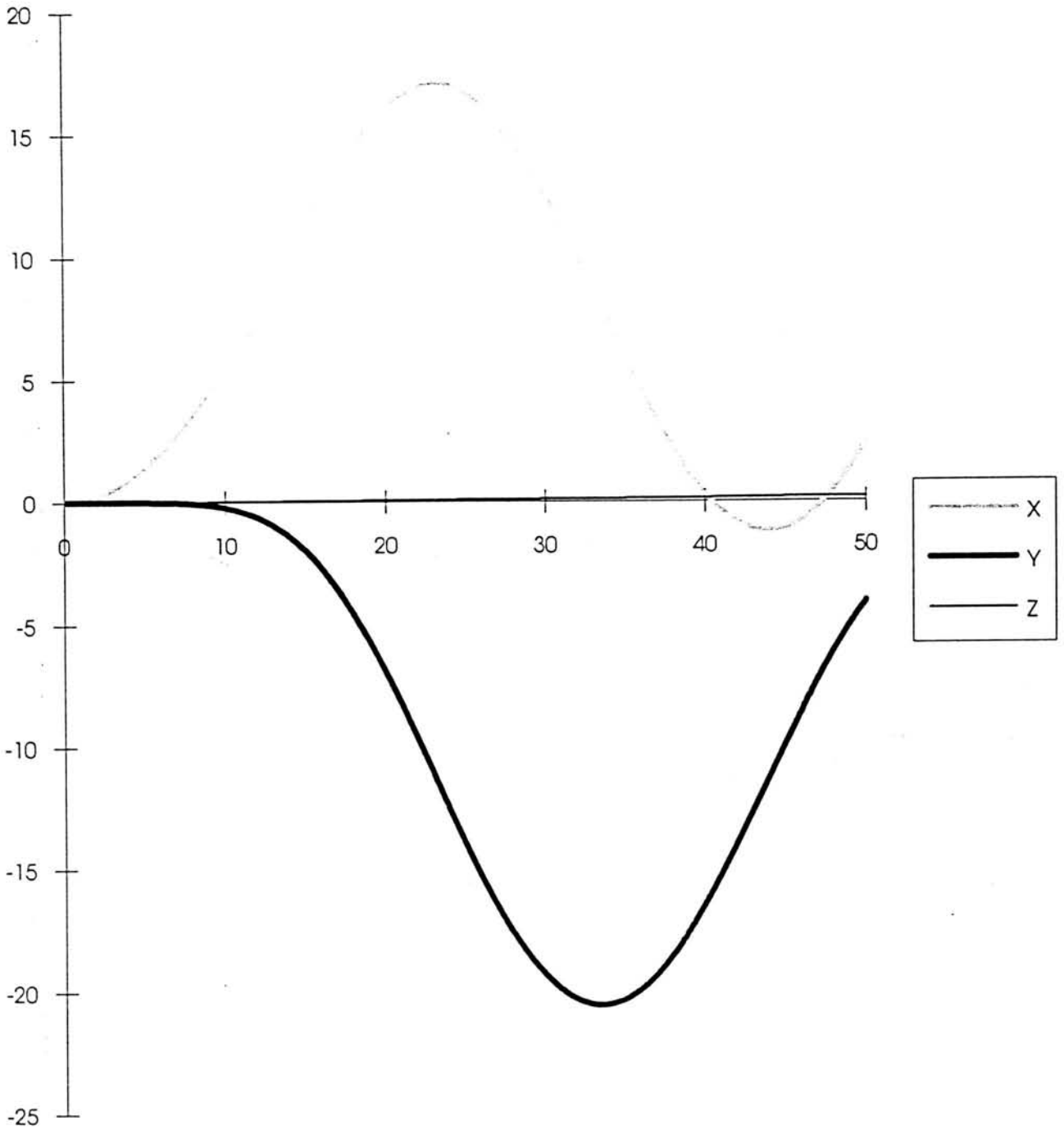
(SEM MASSAS MOVEIS)

X-Y




Y-Z
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-25
-20
-15
-10
-5
0
5
X

XYZ em função do tempo



ANEXO II

Código em C do modelo com massas móveis

```
#include <math.h>
#include <stdio.h>
#include "model.h"

FILE *outfp1,*outfp2,*infp;
int i,j,k,n_sim,cf_flag;
float p2,p3,p4,p5,p6,p7,p8,p9,p10;
float  t0,tfinal,t,n_sim1,dt;

    /* SUPERFICIES DE CONTROLO */
double dr,ds,drs,drb,db,dsp;

    /* FORCAS DOS MOTORES  */
double Fls,Frs,Fbvt,Fsvt,Fblt,Fslt;

double cflow,ucf,vech1[16],vech2[16],vecv1[16],vecv2[16];
double s_phi,s_theta,s_psi,c_phi,c_theta,c_psi,t_theta;

double trap();

   /* TERMOS DAS FORCAS E MOMENTOS APLICADOS */
double suf1,suf2,suf3,suf4;
double swf1,swf2,swf3;
double hf1,hf2,hf3;
double rm1,rm2;
double pm1,pm2,pm3;
double ym1,ym2,ym3;

   /* FORCAS E MOMENTOS DO FLUXO  */
double cf_heave,cf_pitch,cf_sway,cf_yaw;

   /*  ESTADO & ESTADO INICIAL */
double u, v, w, p, q, r, xpos, ypos, zpos, phi, theta, psi;
double xx[13];


   /*  POSICAO DO CENTRO DE MASSA */
double xg,yg,zg;


   /*  ESTADO & ESTADO INICIAL massas moveis */
double dx,dy,dz,dxdot,dydot,dzdot,dx2dot,dy2dot,dz2dot;


   /*  TERMOS DA MATRIZ DE INERCIA DEVIDO AS MASSAS MOVEIS  */
double Imovel44,Imovel55,Imovel66;


   /*  FORCAS DOS MOTORES APLICADAS AS MASSAS */
double Fmx,Fmy,Fmz;


   /*  REFERENCIA PARA O CONTROLADOR DE MASSAS  */
double dxr,dyr,dzr;
double K=1;

   /* VELOCIDADE DE ROTACAO DOS MOTORES  */
double n_ls,n_rs,n_blt,n_slt,n_bvt,n_svt;

/*  MATRIZ DE MASSA & SUA INVERSA  */
double M[7][7],Mi[7][7];
```

```
/* Rotina de integracao numerica pelo metodo dos trapezios  */

double trap(n,a,b)

   int n;
   double a[16],b[16];
{
   int i,n1;
   double out,out1;

   n1 = n - 1;
   out = 0.0;
   for (i=1;i<=n1;++i)
   {
      out1 = 0.5*(a[i] + a[i+1])*(b[i+1] - b[i]);
      out  = out + out1;
   }
   return out;
}



main()
{
   t0   = 0.0;
/*
   n_ls = rotacao da helice esquerda      -500.0 --> +500.0 range
   n_rs = rotacao da helice direita       -500.0 --> +500.0 range
   dr   = angulo do rudder                  -0.4 --> +0.4 rad
   ds   = angulo dos                        -0.4 --> +0.4 rad
   dsp  = angulo do    deflection           -0.4 --> +0.4 rad
   Fbvt = forca do motor vertical dianteiro -2.0 --> +2.0 lbs.
   Fsvt = forca do motor vertical traseiro  -2.0 --> +2.0 lbs.
   Fblt = forca do motor lateral dianteiro  -2.0 --> +2.0 lbs.
   Fslt = forca do motor lateral traseiro   -2.0 --> +2.0 lbs.
*/
   infp = fopen("auv_inp.d","r");    /* ficheiro de entrada */
   outfp1 = fopen("auv_dat1.d","w");  /* ficheiro de saida  */
   outfp2 = fopen("auv_dat2.d","w");  /* Open file for writing */

/* FORMATO DO FICHEIRO DE ENTRADA  AUV_INP.D

   tfinal (sec)   dt (sec)        usualmente 0.1
   n_ls (rpm)     n_rs (rpm)      -500.0 < rpm < +500.0
   dr (deg)  ds (deg)  dsp (deg)  -22.5  < deg < 22.5
   Fbvt (lbs)     Fsvt (lbs)      -2.0   < lbs < 2.0
   Fblt (lbs)     Fslt (lbs)      -2.0   < lbs < 2.0
*/



   fscanf(infp,"%f %f",&tfinal,&dt);
   fscanf(infp,"%G %G",&p2,&p3);
   fscanf(infp,"%G %G %G",&p4,&p5,&p6);
   fscanf(infp,"%G %G",&p7,&p8);
   fscanf(infp,"%G %G",&p9,&p10);

   n_ls=p2;
```

```
n_rs=p3;
dr=p4;
ds=p5;
dsp=p6;
Fbvt=p7;
Fsvt=p8;
Fblt=p9;
Fslt=p10;


printf("tfinal = %f dt = %f\n",tfinal,dt);
printf("n_ls = %f n_rs = %f\n",n_ls,n_rs);
printf("dr = %f ds = %f dsp= %f\n",dr,ds,dsp);
printf("Fbvt = %f Fsvt = %f\n",Fbvt,Fsvt);
printf("Fblt = %f Fslt = %f\n",Fblt,Fslt);

fclose(infp);

/* Conversao dos angulos dr and ds para radianos */
dr = dr*0.0174532925;
ds = ds*0.0174532925;
dsp= dsp*0.0174532925;


n_sim1  = (tfinal - t0)/dt + 1.0;
n_sim = n_sim1;

u     =  u0;
v     =  v0;
w     =  w0;
p     =  p0;
q     =  q0;
r     =  r0;
xpos  =  xpos0;
ypos  =  ypos0;
zpos  =  zpos0;
phi   =  phi0;
theta =  theta0;
psi   =  psi0;


/* ATRIBUIR AS CONDICOES INICIAS AO VECTOR XX   */
xx[1]  = u;
xx[2]  = v;
xx[3]  = w;
xx[4]  = p;
xx[5]  = q;
xx[6]  = r;
xx[7]  = xpos;
xx[8]  = ypos;
xx[9]  = zpos;
xx[10] = phi;
xx[11] = theta;
xx[12] = psi;



dx=dx0;
dy=dy0;
dz=dz0;
dxdot=dxdot0;
```

```
    dydot=dydot0;
    dzdot=dzdot0;


    /* CONSTRUCAO DA MATRIZ DAS MASSAS  */

    M[1][1] =  mass - xudot;
    M[1][5] =  mass*zg;
    M[1][6] = -mass*yg;

    M[2][2] =  mass - yvdot;
    M[2][4] = -mass*zg - ypdot;
    M[2][6] =  mass*xg - yrdot;

    M[3][3] =  mass - zwdot;
    M[3][4] =  mass*yg;
    M[3][5] = -mass*xg - zqdot;

    M[4][2] = -mass*zg - kvdot;
    M[4][3] =  mass*yg;
/*  M[4][4] =  Ixx - kpdot+Imovel44;   */
    M[4][5] = -Ixy;
    M[4][6] = -Ixz - krdot;

    M[5][1] =  mass*zg;
    M[5][3] = -mass*xg - mwdot;
    M[5][4] = -Ixy;
/*  M[5][5] =  Iyy - mqdot+Imovel55;   */
    M[5][6] = -Iyz;

    M[6][1] = -mass*yg;
    M[6][2] =  mass*xg - nvdot;
    M[6][4] = -Ixz - npdot;
    M[6][5] = -Iyz;
/*  M[6][6] =  Izz - nrdot+Imovel66;

    inv(&M,&Mi,6);  */

    t = t0;


/* INICIO DO CICLO PRINCIPAL  */

    for (i=1;i<=n_sim;++i)
    {


  /* COMPONENTE DA INERCIA DEVIDO AS MASSAS MOVEIS  */

    Imovel44=2*(mx*pow(Ymx,2.0)+my*pow(dy,2.0)+mz*pow(dz,2.0));
    Imovel55=2*(my*pow(Xmy,2.0)+mz*pow(Xmz,2.0)+mx*pow(dx,2.0)+mz*pow(dz,2.0));
    Imovel66=2*(mx*pow(Ymx,2.0)+my*pow(Xmy,2.0)+mz*pow(Ymz,2.0)
             +mx*pow(dx,2.0)+my*pow(dy,2.0));


    M[4][4] =  Ixx - kpdot+Imovel44;
    M[5][5] =  Iyy - mqdot+Imovel55;
    M[6][6] =  Izz - nrdot+Imovel66;

    inv(&M,&Mi,6);
```

```
/* CALCULO DA REFERENCIA PARA O CONTROLADOR DAS MASSAS DE MODO A QUE
   PARA A ACTITUDE PRETENDIDA O Rog SEJA PRINCIPALMENTE VERTICAL */

   dxr=-mass*sin(theta)*MRo/mx/2;
   dyr=-mass*sin(theta)*sin(phi)*MRo/my/2;
   dzr=mass*cos(theta)*cos(phi)*MRo/mz/2;


/* COLOCAR AQUI O CONTROLADOR PARA AS MASSAS  */

Fmx = -sin(theta)*mx*2*g+K*(dxr-dx);
Fmy = -sin(theta)*sin(phi)*my*2*g+K*(dyr-dy);
Fmz =  cos(theta)*cos(phi)*mz*2*g+K*(dzr-dz);


/* FIM DO CONTROLADOR    */


dx2dot=(Fmx-Katrito*dxdot*dxdot+sin(theta)*g*mx*2)/mx/2;
dy2dot=(Fmy-Katrito*dydot*dydot+sin(theta)*sin(phi)*g*my*2)/my/2;
dz2dot=(Fmz-Katrito*dzdot*dzdot+cos(theta)*cos(phi)*g*mz*2)/mz/2;

dxdot=dxdot+dx2dot*dt;
dydot=dydot+dy2dot*dt;
dzdot=dzdot+dz2dot*dt;

dx=dx+dxdot*dt+dx2dot*dt*dt/2;
dy=dy+dydot*dt+dy2dot*dt*dt/2;
dz=dz+dzdot*dt+dz2dot*dt*dt/2;


   /*   INVERSAO DOS SINAIS DOS LEMES PARA A FRENTE  */
   drs = dr;
   drb = -dr;
   ds  = ds;
   db  = -ds;


   /* LIMITACAO DOS ANGULOS */

   if(fabs(drs) >= 0.4)
   {
      drs = 0.4*(fabs(drs)/drs);
   }

   if(fabs(ds) >= 0.4)
   {
      ds = 0.4*(fabs(ds)/ds);
   }

   if(fabs(drb) >= 0.4)
   {
      drb = 0.4*(fabs(drb)/drb);
   }

   if(fabs(db) >= 0.4)
   {
      db = 0.4*(fabs(db)/db);
   }
```

```
if(fabs(dsp) >= 0.4)
{
   db = 0.4*(fabs(dsp)/dsp);
}


/* LIMITACAO DAS FORCAS */

if(fabs(Fls) >= 5.0)
{
   Fls = 5.0*(fabs(Fls)/Fls);
}

if(fabs(Frs) >= 5.0)
{
   Frs = 5.0*(fabs(Frs)/Frs);
}

if(fabs(Fbvt) >= 2.0)
{
   Fbvt = 2.0*(fabs(Fbvt)/Fbvt);
}

if(fabs(Fsvt) >= 2.0)
{
   Fsvt = 2.0*(fabs(Fsvt)/Fsvt);
}

if(fabs(Fblt) >= 2.0)
{
   Fblt = 2.0*(fabs(Fblt)/Fblt);
}

if(fabs(Fslt) >= 2.0)
{
   Fslt = 2.0*(fabs(Fslt)/Fslt);
}


/* CALCULO DAS FORCAS DE 'DRAG', INTEGRANDO O 'DRAG' AO LONGO DO AUV */

cf_flag = 0;
for (k=1;k<=15;++k)
{
   ucf = pow( (v + x_cf[k]*r),2.0) + pow( (w - x_cf[k]*q),2.0);
   ucf = sqrt(ucf);
   if (ucf < 1.0e-6)
   {
      cf_flag = 1;
      break;
   }

   cflow = cdy*hh_cf[k]*pow( (v + x_cf[k]*r),2.0) +
           cdz*br_cf[k]*pow( (w - x_cf[k]*q),2.0);
   vech1[k] = cflow*(v + x_cf[k]*r)/ucf;
   vech2[k] = cflow*(v + x_cf[k]*r)*x_cf[k]/ucf;
   vecv1[k] = cflow*(w - x_cf[k]*q)/ucf;
   vecv2[k] = cflow*(w - x_cf[k]*q)*x_cf[k]/ucf;
}
```

```
if (cf_flag == 0)
{
   cf_heave = trap(15,vecv1,x_cf);
   cf_pitch = trap(15,vecv2,x_cf);
   cf_sway  = trap(15,vech1,x_cf);
   cf_yaw   = trap(15,vech2,x_cf);

   cf_heave = -0.5*rho*cf_heave;
   cf_pitch = 0.5*rho*cf_pitch;
   cf_sway  = -0.5*rho*cf_sway;
   cf_yaw   = -0.5*rho*cf_yaw;
}
   else
{
   cf_heave = 0.0;
   cf_pitch = 0.0;
   cf_sway  = 0.0;
   cf_yaw   = 0.0;
}



/* FORCA DE 'SURGE'

   Fls = Forca de propulsao da helice esquerda  (+ F) in x direct.
   Frs = Forca de propulsao da helice direita   (+ F) in x direct. */

Fls = (xprop/2.0)*(n_ls*fabs(n_ls));
Frs = (xprop/2.0)*(n_rs*fabs(n_rs));

suf1 = mass*(v*r - w*q + xg*(pow(q,2.0) + pow(r,2.0)) ) +
       mass*(-yg*p*q - zg*p*r) + xpp*pow(p,2.0) + xqq*pow(q,2.0)
       + xrr*pow(r,2.0) + xpr*p*r + xwq*w*q + xvp*v*p + xvr*v*r
       - (weight - boy)*sin(theta) - xres*u*fabs(u);
suf2 = u*q*(xqds*ds + xqdb*db) + u*r*(xrdrs*drs + xrdrb*drb);
suf3 = (xdsds*pow(ds,2.0) + xdbdb*pow(db,2.0)
       + xdrdr*(pow(drs,2.0) + pow(drb,2.0))
       +2*xdspdsp*pow(dsp,2.0))*u*fabs(u);
suf4 = Fls + Frs;
f[1] = suf1 + suf2 + suf3 + suf4 + mx*dx2dot;



/* forca de 'SWAY'

   Fblt = Forca de propulsao do motor lateral dianteiro
          (+ F) segundo y.
   Fslt = Forca de propulsao do motor lateral traseiro
          (+ F) segundo y.  */

swf1 = mass*(-u*r + w*p - xg*p*q + yg*(pow(p,2.0) + pow(r,2.0))
       - zg*q*r)
       + (weight-boy)*sin(phi)*cos(theta)
       + ypq*p*q + yqr*q*r + yp*u*p + yr*u*r + yvq*v*q + ywp*w*p
       + ywr*w*r + yv*u*v + yvw*v*w + cf_sway;
swf2 = ydrs*u*fabs(u)*drs + ydrb*u*fabs(u)*drb;
swf3 = Fblt + Fslt;
f[2] = swf1 + swf2 + swf3 + mz*dy2dot;
```

```
      /* FORCA DE 'HEAVE'

         Fbvt = Forca de propulsao do motor vertical dianteiro
                (+ F) segundo Z.
         Fsvt = Forca de propulsao do motor vertical traseiro
                (+ F) segundo Z. */

      hf1 = mass*(u*q - v*p - xg*p*r - yg*q*r + zg*(pow(p,2.0) + pow(q,2.0)))
          + (weight - boy)*cos(phi)*cos(theta) + zpp*pow(p,2.0) + zpr*p*r
          + zrr*pow(r,2.0) + zq*u*q + zvp*v*p + zvr*v*r + zw*u*w
          + zvv*pow(v,2.0) + cf_heave;
      hf2 = u*fabs(u)*(zds*ds + zdb*db);
      hf3 = Fbvt + Fsvt;
      f[3] = hf1 + hf2 + hf3 + mz*dz2dot;
   /*   printf("u,hf1,hf2,hf3 = %f %f %f %f\n",u,hf1,hf2,cf_heave); */



      /* MOMENTO DE 'ROLL'  */

      rm1 = -(Izz - Iyy)*q*r - Ixy*p*r + Iyz*(pow(q,2.0) - pow(r,2.0))+Ixz*p*q
            + mass*(yg*(u*q - v*p) - zg*(-u*r + w*p))
            + (yg*weight-yb*boy)*cos(phi)*cos(theta)
            - (zg*weight-zb*boy)*sin(phi)*cos(theta)
            + kpq*p*q + kqr*q*r + kp*u*p + kr*u*r + kvq*v*q + kwp*w*p
            + kwr*w*r + kv*u*v + kvw*v*w - 100.0*p*fabs(p);
      rm2 = u*fabs(u)*kdsp*dsp;
      f[4] = rm1+rm2;



      /* MOMENTO DE 'PITCH'

         xbvt = Posicao x do motor vertical dianteiro (+ x).
         xsvt = Posicao x do motor vertical dianteiro (+ x).*/

      pm1 = -(Ixx - Izz)*p*r + Ixy*q*r - Iyz*p*q - Ixz*(pow(p,2.0)-pow(r,2.0))
            + mass*( xg*(-u*q + v*p) + zg*(v*r - w*q) )
            - (xg*weight - xb*boy)*cos(phi)*cos(theta)
            - (zg*weight - zb*boy)*sin(theta)
            + mpp*pow(p,2.0) + mpr*p*r + mrr*pow(r,2.0) + mq*u*q + mvp*v*p
            + mvr*v*r + mw*u*w + mvv*pow(v,2.0) + cf_pitch;
      pm2 = u*fabs(u)*(mds*ds + mdb*db);
      pm3 = -xbvt*Fbvt - xsvt*Fsvt;
      f[5] = pm1 + pm2 + pm3;
/*      printf("u,pm1,pm2,pm3 = %f %f %f %f\n",u,pm1,pm2,cf_pitch); */



      /* MOMENTO DE 'YAW'

         xblt = Posicao x do motor lateral dianteiro (+ x).
         xslt = Posicao x do motor lateral traseiro  (+ x).
         yrs  = Posicao y da helice direita (+ x).
         yls  = Posicao y da helice esquerda      (- x).   */


      ym1 = -(Iyy - Ixx)*p*q + Ixy*(pow(p,2.0) - pow(q,2.0)) + Iyz*p*r - Ixz*q
*r
            + mass*(xg*(w*p - u*r) + yg*(w*q-v*r))
            + (xg*weight-xb*boy)*sin(phi)*cos(theta)
```

```
      + (yg*weight-yb*boy)*sin(theta)
      + npq*p*q + nqr*q*r + np*u*p + nr*u*r + nvq*v*q + nwp*w*p
      + nwr*w*r + nv*u*v + nvw*v*w + cf_yaw;
ym2 = ndrs*u*fabs(u)*drs + ndrb*u*fabs(u)*drb;
ym3 = xblt*Fblt + xslt*Fslt - yrs*Frs - yls*Fls;
f[6] = ym1 + ym2 + ym3;


/* VELOCIDADES INERCIAS */

s_phi = sin(phi); s_theta = sin(theta); s_psi = sin(psi);
c_phi = cos(phi); c_theta = cos(theta); c_psi = cos(psi);
t_theta = tan(theta);

f[7] = uc + u*c_psi*c_theta;
f[7] = f[7] + v*(c_psi*s_theta*s_phi - s_psi*c_phi);
f[7] = f[7] + w*(c_psi*s_theta*c_phi + s_psi*s_phi);

f[8] = vc + u*s_psi*c_theta;
f[8] = f[8] + v*(s_psi*s_theta*s_phi + c_psi*c_phi);
f[8] = f[8] + w*(s_psi*s_theta*c_phi - c_psi*s_phi);

f[9] = wc - u*s_theta + v*c_theta*s_phi;
f[9] = f[9] + w*c_theta*c_phi;


/* TAXAS DE ANGULOS DE 'EULER' */
f[10]= p + q*s_phi*t_theta + r*c_phi*t_theta;

f[11]= q*c_phi - r*s_phi;

f[12]= (1/c_theta)*(q*s_phi + r*c_phi);


/* CRIACAO DA MATRIZ DE ESTADO E CALCULO DO LADO DIREITO DE XDOT=f(X)
   XXDOT = [imm zeros(6,6);zeros(6,6) eye(6,6)]*f' */

for (j=1;j<=12;++j)
{
   for (k=1;k<=12;++k)
   {
      state_matrix[j][k] = 0.0;
   }
}

for (j=1;j<=6;++j)
{
   for (k=1;k<=6;++k)
   {
      state_matrix[j][k] = Mi[j][k];
   }
}

for (j=7;j<=12;++j)
{
   state_matrix[j][j] = 1.0;
}

for (j=1;j<=12;++j)
{
   xxdot[j] = 0.0;
```

```
            for (k=1;k<=12;++k)
            {
               xxdot[j] = xxdot[j] + state_matrix[j][k]*f[k];
            }
         }

         for (j=1;j<=12;++j)
         {
            xx[j] = xx[j] + dt*xxdot[j];
         }

         fprintf(outfp1,"%f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f\n",
         t,xpos,ypos,zpos,phi,theta,psi,u,v,w,p,q,r);


/*       fprintf(outfp2,"%f %f %f %f %f %f %f %f %f %f\n",
                        Fls,Frs,Fblt,Fslt,Fbvt,Fsvt,drs,ds,drb,db);
*/

         u     = xx[1];
         v     = xx[2];
         w     = xx[3];
         p     = xx[4];
         q     = xx[5];
         r     = xx[6];
         xpos  = xx[7];
         ypos  = xx[8];
         zpos  = xx[9];
         phi   = xx[10];
         theta = xx[11];
         psi   = xx[12];

         t = t + dt;
      }

      fclose(outfp1);
      fclose(outfp2);
}



/* function inv(a,ai,n):
   where a = n x n matrix
         ai = n x n inverse of matrix a
         n = row & column dimension of matrix a
    Usage:
         .
         .
         inv(&a,&ai,n);
         .
         .

    NOTE: Matrices in calling program must be dimensioned
          a[n+1][n+1], ai[n+1][n+1] since matrices are
          indexed as in FORTRAN!. If n > 30 array size below
          must be increased.
*/
   inv(a,ai,n)

      int n;
      double *a,*ai;
```

```
{
   int i,j,k,ki,sing_flag;
   double b,b1,b2;
   double a_local[30][30],ainv[30][30];

   sing_flag = 0;

   for(i=1;i<=n;++i)
   {
      for(j=1;j<=n;++j)
      {
         a_local[i][j] = *(a+i*(n+1)+j);
      }
   }

   for (i=1;i<=n;++i)
   {
      for (j=1;j<=n;++j)
      {
         ainv[i][j] = 0.0;
      }
   }

   for (i=1;i<=n;++i)
   {
      ainv[i][i] = 1.0;
   }

   for (k=1;k<=n-1;++k)
   {
      b = a_local[k][k];
      ki = k;

      for (i=k+1;i<=n;++i)
      {
         if( (fabs(b) - fabs(a_local[i][k])) >= 0.0 )
         {
         }
         else
         {
            b = a_local[i][k];
            ki = i;
         }
      }

      if( fabs(b) < 0.0001)
      {
         sing_flag = 1;
         break;
      }

      if( (ki-k) == 0)
      {
      }
      else
      {
         for (j=k;j<=n;++j)
         {
            b1 = a_local[k][j];
            a_local[k][j] = a_local[ki][j];
            a_local[ki][j] = b1;
```

```
        }
        for (j=1;j<=n;++j)
        {
           b2 = ainv[k][j];
           ainv[k][j] = ainv[ki][j];
           ainv[ki][j] = b2;
        }
     }

     for (j=k+1;j<=n;++j)
     {
        a_local[k][j] = a_local[k][j]/b;
     }

     for (j=1;j<=n;++j)
     {
        ainv[k][j] = ainv[k][j]/b;
     }

     for (i=k+1;i<=n;++i)
     {
        for (j=k+1;j<=n;++j)
        {
           a_local[i][j] = a_local[i][j] - a_local[i][k]*a_local[k][j];
        }

        for (j=1;j<=n;++j)
        {
           ainv[i][j] = ainv[i][j] - a_local[i][k]*ainv[k][j];
        }
     }
  }

  if(sing_flag == 0)
  {
     for (j=1;j<=n;++j)
     {
        ainv[n][j] = ainv[n][j]/a_local[n][n];
     }

     for (k=n-1;k>=1;--k)
     {
        for (j=1;j<=n;++j)
        {
           for (i=k+1;i<=n;++i)
           {
              ainv[k][j] = ainv[k][j] - a_local[k][i]*ainv[i][j];
           }
        }
     }

     for(i=1;i<=n;++i)
     {
        for(j=1;j<=n;++j)
        {
           *(ai+i*(n+1)+j) = ainv[i][j];
        }
     }
  }
  else
  {
     printf("Singular or Ill-Conditioned Matrix\n");
```

```
    }
}


/* float  dt;
double yls,yrs,xbvt,xsvt,xblt,xslt;
double xg,yg,zg,xb,yb,zb;
double mass,weight,boy;
double Ixx,Iyy,Izz,Ixy,Iyz,Ixz;
double l,rho,g;
double cd0,cdy,cdz;
double xrs,xrb;
double rl2,rl3,rl4,rl5;

  STATES & INITIAL STATES
double u0,v0,w0,p0,q0,r0,xpos0,ypos0,zpos0,phi0,theta0,psi0;

  COEFICIENTES HIDRODINAMICOS 'SURGE'
double xpp,xqq,xrr,  xpr,  xudot,xwq, xvp,  xvr,  xqds, xqdb,xrdrs,xrdrb;
double xvv,xww,xvdrs,xvdrb,xwds, xwdb,xdsds,xdbdb,xdrdr,xdspdsp,xres,xprop;

  COEFICIENTES HIDRODINAMICOS 'SWAY'
double ypdot,yrdot,ypq,yqr, yvdot,yp,yr,yvq,ywp;
double ywr,  yv,   yvw,ydrs,ydrb;

  COEFICIENTES HIDRODINAMICOS 'HEAVE'
double zqdot,zpp,zpr,zr, zrr, zwdot,zq,zvp;
double zvr,  zw, zvv,zds,zdb;

  COEFICIENTES HIDRODINAMICOS 'ROLL'
double kpdot,krdot,kpq,kqr,kvdot,kp;
double kr,   kvq,  kwp,kwr,kv,kvw,kdsp;

  COEFICIENTES HIDRODINAMICOS 'PITCH'
double mqdot,mpp,mpr,mrr,mwdot,mq;
double mvp,  mvr,mw, mvv,mds,  mdb;

  COEFICIENTES HIDRODINAMICOS 'YAW'
double npdot,nrdot,npq,nqr,nvdot,np,  nr;
double nvq,  nwp,  nwr,nv, nvw,  ndrs,ndrb; */



/* COMPRIMENTO CF_X, LARGURA CF_BR, E  ALTURA CF_HH
   PARA A INTEGRACAO DO FLUXO
double x_cf[16],br_cf[16],hh_cf[16];*/

/* VECTOR DA DERIVADA DO ESTADO E DAS FORCAS */
double xxdot[13],f[13];

/* MATRIZ DE ESTADO  */
double state_matrix[13][13];


double dx0=0;
double dy0=0;
double dz0=0;
double dxdot0=0;
double dydot0=0;
double dzdot0=0;

double mx = 13.52/60;
```

```
double my = 13.52/60;
double mz = 13.52/60;

double Ymx = 1;
double Zmx = 0;
double Xmy = 1;
double Zmy = 0;
double Xmz = 1;
double Ymz = 0;

double MRo=1*1/30/2;

double Katrito = 0.05;

/*  VELOCIDADES GLOBAIS DA CORRENTE */
double uc = 0.0;
double vc = 0.0;
double wc = 0.0;


float dt = 0.1;
double yls = -4.0/12.0;
double yrs = 4.0/12.0;
double xbvt = 17.0/12.0;
double xsvt = -17.0/12.0;
double xblt = 23.0/12.0;
double xslt = -23.0/12.0;

double xg0  = 0.0/12.0;
double yg0 = 0.0;
double zg0 = 0.0/12.0;

double xb = 0.0/12.0;
double yb = 0.0;
double zb = 0.0;

double weight = 435.0;
double g = 32.174;
double mass = 13.520;
double boy = 435.0;
double l = 87.625/12.0;
double rho = 1.94;
double cd0 = 0.00778;
double cdy = 0.5;
double cdz = 0.6;

double Ixx    = 2.7;
double Iyy    = 42.0;
double Izz    = 45.0;
double Ixy    = 0.0;
double Iyz    = 0.0;
double Ixz    = 0.0;

double xrs = -0.377*87.625/12.0;
double xrb = 0.283*87.625/12.0;

/* CONDICOES INICIAIS */
double u0 = 0.0; double v0 = 0.0; double w0 = 0.0;
double p0 = 0.0; double q0 = 0.0; double r0 = 0.0;
double xpos0 = 0.0; double ypos0 = 0.0; double zpos0 = 0.0;
double phi0 = 0.0; double theta0 = 0.0; double psi0 = 0.0;
```

```
    double r12 = 51.72;
    double r13 = 377.67;
    double r14 = 2756.81;
    double r15 = 20137.50;

    /*  COEFICIENTES HIDRODINAMICOS 'SURGE' */
    double xpp = 0.0*2756.81;
    double xqq = 0.0*2756.81;
    double xrr = -0.00753*2756.81;
    double xpr = 0.0*2756.81;
    double xudot = -0.00282*377.67;
    double xwq = 0.0*377.67;
    double xvp = 0.0*377.67;
    double xvr = 0.0*377.67;
    double xqds = 0.0*377.67;
    double xqdb = 0.0*377.67;
    double xrdrs = 0.0*377.67;
    double xrdrb = 0.0*377.67;
    double xvv = -0.01743*51.72;
    double xww = 0.0*51.72;
    double xvdrs = 0.0*51.72;
    double xvdrb = 0.0*51.72;
    double xwds = 0.0*51.72;
    double xwdb = 0.0*51.72;
    double xdsds = -0.01018*0.417*51.72;
    double xdbdb = -0.01018*0.417*51.72;
    double xdrdr = -0.01018*0.417*51.72;
    double xdspdsp = -0.01018*0.417*51.72;
    double xres = 0.4024;
    double xprop = 6.98579e-6;
/*    double xres = cd0*51.72;
    double xprop = cd0*51.72*pow((1.5/360.0),2.0); */

    /*  COEFICIENTES HIDRODINAMICOS 'SWAY' */
    double ypdot = 0.0*2756.81;
    double yrdot = -0.00178*2756.81;
    double ypq = 0.0*2756.81;
    double yqr = 0.0*2756.81;
    double yvdot = -0.03430*377.67;
    double yp = 0.0*377.67;
    double yr = 0.01187*377.67;
    double yvq = 0.0*377.67;
    double ywp = 0.0*377.67;
    double ywr = 0.0*377.67;
    double yv = -0.10700*51.72;
    double yvw = 0.0*51.72;
    double ydrs = 0.01241*51.72;
    double ydrb = 0.01241*51.72;

    /*  COEFICIENTES HIDRODINAMICOS 'HEAVE' */
    double zqdot = -0.00253*2756.81;
    double zpp = 0.0*2756.81;
    double zpr = 0.0*2756.81;
    double zrr = 0.0*2756.81;
    double zwdot = -0.09340*377.67;
    double zq = -0.07013*377.67;
    double zvp = 0.0*377.67;
    double zvr = 0.0*377.67;
    double zw = -0.78440*51.72;
    double zvv = 0.0*51.72;
    double zds = -0.02110*51.72;
    double zdb = -0.02110*51.72;
```

```
/*  COEFICIENTES HIDRODINAMICOS 'ROLL' */
double kpdot = -0.00024*20137.50;
double krdot = 0.0*20137.50;
double kpq = 0.0*20137.50;
double kqr = 0.0*20137.50;
double kvdot = 0.0*2756.81;
double kp = -0.00540*2756.81;
double kr = 0.0*2756.81;
double kvq = 0.0*2756.81;
double kwp = 0.0*2756.81;
double kwr = 0.0*2756.81;
double kv = 0.0*377.67;
double kvw = 0.0*377.67;
double kdsp = -2.9690;

/*  COEFICIENTES HIDRODINAMICOS 'PITCH' */
double mqdot = -0.00625*20137.50;
double mpp = 0.0*20137.50;
double mpr = 0.0*20137.50;
double mrr = 0.0*20137.50;
double mwdot = -0.00253*2756.81;
double mq = -0.01530*2756.81;
double mvp = 0.0*2756.81;
double mvr = 0.0*2756.81;
double mw = 0.05122*377.67;
double mvv = 0.0*377.67;
double mds = -1.7664;
double mdb =  1.3260;

/*  COEFICIENTES HIDRODINAMICOS 'YAW' */
double npdot = 0.0*20137.50;
double nrdot = -0.00047*20137.50;
double npq = 0.0*20137.50;
double nqr = 0.0*20137.50;
double nvdot = -0.00178*2756.81;
double np = 0.0*2756.81;
double nr = -0.00390*2756.81;
double nvq = 0.0*2756.81;
double nwp = 0.0*2756.81;
double nwr = 0.0*2756.81;
double nv = -0.00769*377.67;
double nvw = 0.0*377.67;
double ndrs = -1.7663;
double ndrb = 1.3259;

/* DEFINICAO DA FUNCAO DE COMPRIMENTO X, LARGURA BR, E ALTURA HH  */

double x_cf[16] = { 0.0,
-43.9/12.0,
-39.2/12.0,
-35.2/12.0,
-31.2/12.0,
-27.2/12.0,
-10.0/12.0,
 0.0/12.0,
 10.0/12.0,
 26.8/12.0,
 32.0/12.0,
 37.8/12.0,
 40.8/12.0,
 42.3/12.0,
```

```
 43.3/12.0,
 43.7/12.0};

double hh_cf[16] = { 0.0,
0.0/12.0,
2.7/12.0,
5.2/12.0,
7.6/12.0,
10.1/12.0,
10.1/12.0,
10.1/12.0,
10.1/12.0,
10.1/12.0,
9.6/12.0,
7.6/12.0,
5.6/12.0,
4.2/12.0,
2.3/12.0,
0.0/12.0};

double br_cf[16] = { 0.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
16.5/12.0,
15.5/12.0,
12.4/12.0,
9.5/12.0,
7.0/12.0,
4.0/12.0,
0.0/12.0};
```