Anais do XXXIV COBENGE. Passo Fundo: Ed. Universidade de Passo Fundo, Setembro de 2006.
ISBN 85-7515-371-4

SIMULAÇÃO DE SISTEMAS CAOTICOS NO MICROMEDIA FLASC

José Silvério Edmundo Germano - silverio@ita.br
ITA–Instituto Tecnológico de Aeronáutica , Departamento de Física – IEFF
Pç. Marechal Eduardo Gomes, n 50 – Campus do CTA, 12228-900, São José dos Campos – SP

Leandro Resende de Pádua - leandrorpadua@yahoo.com.br
ITA–Instituto Tecnológico de Aeronáutica , Departamento de Física – IEFF
Pç. Marechal Eduardo Gomes, n 50 – Campus do CTA, 12228-900, São José dos Campos – SP

Resumo: *O uso do microcomputador, dentro do ambiente da sala de aula, pode ser utilizado com grande eficiência, na simulação de fenômenos físicos, tanto no ensino da Física básica quanto nos estudos mais avançados. Um assunto, dos mais complexos, que será tema desse trabalho é a Teoria do Caos. Em várias áreas da física, bem como da engenharia, é necessário que sejam feitas análises de sistemas dinâmicos não-lineares. Devido a necessidade de se criar novas ferramentas de visualização para estudar esse assunto, este projeto contempla a simulação de sistemas caóticos utilizando o software Macromedia Flash®. Na solução das equações envolvidas, utilizamos técnicas de cálculo numérico, dentro da liguaguem de programação do Flash, que é o actionscript.*

Palavras-chave: *Cálculo numérico, Flash, Caos, Objetos de aprendizagem*

1. INTRODUÇÃO

Dentro da classe dos sistemas matemáticas mais conhecidos, temos aqueles em que o princípio da superposição é aplicável, os quais são ditos sistemas lineares, e aqueles aos quais tal princípio não pode ser aplicado, os sistemas não-lineares. Entretanto, esses dois tipos de sistemas estão relacionados de tal forma que se pode dizer que os sistemas lineares formam um subconjunto de um conjunto maior, o dos sistemas não-lineares. Partindo dessa linha de pensamento, fica clara a importância da análise desse último para a compreensão de fenômenos físicos.

Para sistemas não-lineares a resposta a um distúrbio não é necessariamente proporcional à intensidade do mesmo, e por esse motivo, apesar de serem governados por leis exatas, a evolução com o tempo desses sistemas é bastante sensível às condições iniciais. Esse estado de aparente desordem e irregularidade é o alvo de estudo da Teoria do Caos. Esta teoria estuda o comportamento, que à primeira vista parece ser aleatório e imprevisível, dos sistemas, mostrando uma faceta onde podem ocorrer irregularidades na uniformidade da

natureza como um todo. Fruto dessa realidade científica, essa teoria estende suas ramificações nos mais diversos campos do conhecimento científico, incluindo Física, Medicina, Economia, Matemática, etc.

Diante da importância e igual dificuldade associada à compreensão dessa classe de fenômenos, temos uma tarefa ainda mais difícil: repassar esse conhecimento a alunos de graduação em engenharia e física. Dentro desse contexto, este projeto vem apresentar uma nova ferramenta a ser utilizada no ensino da Teoria do Caos, utilizando para tanto, o *Macromedia Flash*® no desenvolvimento de algumas simulações.

As facilidades apresentadas por esse *software* para a criação de recursos visuais e multimídia em geral, são inúmeras, e simulações físicas podem se tornar muito didáticas com um bom aproveitamento de tais recursos. Contudo, a maior dificuldade aparente seria a resolução de problemas de física que apresentam uma matemática complexa, que é uma característica presente em sistemas não-lineares. Essa dificuldade deve-se ao fato do *Flash* ser primeiramente um programa de animações gráficas. Contornando tal situação, este projeto apresenta uma solução para esse problema, mostrando como utilizar o cálculo numérico para resolução de sistemas caóticos em aplicações criadas com esse *software*.

Dentro desse contexto, é descrito nesse projeto como foi feita a implementação do método numérico de Runge-Kutta no *Flash*, para a resolução do sistema caótico do pêndulo duplo, mostrando que com essa idéia, pode-se contornar o problema da solução matemática sem perda de qualidade e interatividade para o programa, de tal forma que o *software* pode ser visto como uma nova ferramenta na criação de objetos de aprendizagem.

2. SIMULAÇÃO DO MOVIMENTO DO PÊNDULO DUPLO

Nesse ponto é importante que seja feita uma análise matemática do problema do pêndulo duplo analisado. Consideremos então o pêndulo duplo mostrado na Figura 1.

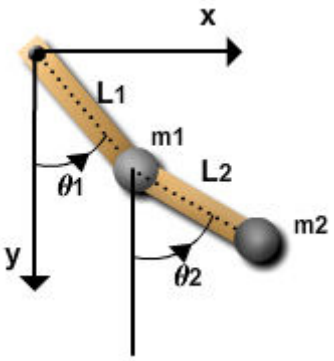

Figura 1. Pêndulo duplo e referenciais adotados.

Como pode-se observar na Figura 1, temos:

x = posição horizontal das massas do pêndulo
y = posição vertical das massas do pêndulo
θ = ângulo do pêndulo com a vertical, sendo positivo no sentido anti-horário.
L = comprimento do braço rígido e de massa desprezível.

Onde temos que x_1, y_1, θ_1 e L_1 são referentes a m_1 e x_2, y_2, θ_2 e L_2 a m_2.

Analisando a cinemática do problema, pode-se demonstrar que as equações de movimento do pêndulo são:

$$x_1'' = -\theta_1'^2 \sin\theta_1 + \theta_1'' L_1 \cos\theta_1 \quad (1)$$

$$y_1'' = \theta_1'^2 L_1 \cos\theta_1 + \theta_1'' L_1 \sin\theta_1 \quad (2)$$

$$x_2'' = x_1'' - \theta_2'^2 L_2 \sin\theta_2 + \theta_2'' L_2 \cos\theta_2 \quad (3)$$

$$y_2'' = y_1'' + \theta_2'^2 L_2 \cos\theta_2 + \theta_2'' L_2 \sin\theta_2 \quad (4)$$

2.1 Solucionando o problema

A estratégia escolhida para solucionar as equações (1), (2), (3) e (4) foi a de utilizar métodos numéricos. Foi feito dessa forma, pois a implementação do método no programa dava uma maior liberdade de escolha dos parâmetros do problema para o usuário. Suponha, por exemplo, que os programas criados para simular os sistemas caóticos devam ter uma solução analítica para cada conjunto de parâmetros e valores iniciais. Sabe-se, entretanto, que isso nem sempre é possível e quando o é, torna-se algo extremamente trabalhoso, pois obtém-se soluções totalmente diferentes em cada caso, e uma simples alteração dos parâmetros do problema poderia ocasionar soluções diferentes e, para facilitar ou até mesmo tornar possível a implementação da aplicação, o programador poderia dar menor liberdade de escolha para o usuário.

Métodos numéricos evitam esse tipo de transtorno criando uma solução única aproximada para qualquer conjunto de valores colocados ao problema. Dessa forma, o usuário do programa entra com os valores dos parâmetros (valores das massas – m_1 e m_2, posição inicial das mesmas, que são dadas por θ_1 e θ_2 e aceleração da gravidade, g) e para qualquer combinação desses valores, o mesmo bloco de código gera a solução apropriada.

Para a resolução das equações diferenciais (1), (2), (3) e (4) foi utilizado o método numérico de Runge-Kutta de ordem quatro. Sua escolha para o projeto foi devida à sua grande precisão, estabilidade e convergência do processo de solução, e pelo fato de poder ser utilizado na resolução de equações diferenciais ordinárias não-lineares.

Os métodos de Runge-Kutta são uma família de métodos numéricos para solucionar equações diferenciais ordinárias. São métodos que podem ser obtidos pela série de Taylor sem a necessidade de calcular qualquer derivada.

Esse método consiste em estimar o valor da função em vários pontos intermediários e o valor solução é encontrado pela média ponderada entre esses pontos. Por sua dedução bastante trabalhosa, limitamo-nos a enunciar sua expressão utilizada na resolução do problema do pêndulo duplo.

A partir das equações (1), (2), (3) e (4), pode-se obter a expressão de θ_1'' e θ_2'' em função de θ_1' e θ_2'' utilizando um programa como o *Mathematica.* Assim temos que:

$$\theta_1'' = \frac{-g(2m_1 + m_2)\operatorname{sen}\theta_1 - m_2 g \operatorname{sen}(\theta_1 - 2\theta_2) - 2\operatorname{sen}(\theta_1 - \theta_2) m_2 (\theta_2'^2 L_2 - \theta_1'^2 L_1 \cos(\theta_1 - \theta_2))}{L_2(2m_1 + m_2 - m_2 \cos(2\theta_1 - 2\theta_2))} \quad (5)$$

$$\theta_2'' = \frac{2\operatorname{sen}(\theta_1 - \theta_2)(\theta_1'^2 L_1 (m_1 + m_2) + g(m_1 + m_2)\cos(\theta_1) + \theta_2'^2 L_2 m_2 \cos(\theta_1 - \theta_2))}{L_1(2m_1 + m_2 - m_2 \cos(2\theta_1 - 2\theta_2))} \quad (6)$$

Fazendo que:

$$\theta_1' = w_1 = k(t^i, \theta_1^i, \theta_2^i, w_1^i, w_2^i) \quad (7)$$

$$\theta_2' = w_2 = j(t^i, \theta_1^i, \theta_2^i, w_1^i, w_2^i) \quad (8)$$

$$\theta_1'' = m(t^i, \theta_1^i, \theta_2^i, w_1^i, w_2^i) \quad (9)$$

$$\theta_2'' = n(t^i, \theta_1^i, \theta_2^i, w_1^i, w_2^i) \quad (10)$$

e sendo h o passo de indução e θ_1^i o valor de θ_1 no instante t^i, analogamente para θ_2^i, w_1^i e w_2^i, estamos então nas condições de aplicar o algoritmo de Runge-Kutta, o qual se encontra na Listagem 1.

$$\theta_1^{i+1} = \theta_1^i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$\theta_2^{i+1} = \theta_2^i + \frac{h}{6}(j_1 + 2j_2 + 2j_3 + j_4)$$

$$w_1^{i+1} = w_1^i + \frac{h}{6}(m_1 + 2m_2 + 2m_3 + m_4)$$

$$w_2^{i+1} = w_2^i + \frac{h}{6}(n_1 + 2n_2 + 2n_3 + n_4)$$

onde

$$k_1 = k(t^i, \theta_1^i, \theta_2^i, w_1^i, w_2^i);$$

$$k_2 = k(t^i + \frac{h}{2}, \theta_1^i + \frac{h}{2}k_1, \theta_2^i + \frac{h}{2}j_1, w_1^i + \frac{h}{2}m_1, w_2^i + \frac{h}{2}n_1);$$

$$k_3 = k(t^i + \frac{h}{2}, \theta_1^i + \frac{h}{2}k_2, \theta_2^i + \frac{h}{2}j_2, w_1^i + \frac{h}{2}m_2, w_2^i + \frac{h}{2}n_2);$$

$$k_4 = k(t^i + h, \theta_1^i + hk_2, \theta_2^i + hj_2, w_1^i + hm_2, w_2^i + hn_2);$$

Analogamente a k_p temos para j_p, m_p e n_p, onde p = 1, 2, 3, 4.

Listagem 1. expressão do método de Runge-Kutta de quarta ordem para resolução do sistema não-linear descrito nas equações (7), (8), (9) e (10).

2.2 Implementação da solução

A solução foi implementada no *Flash* com um passo de indução h = 0.1 de forma que para cada posição do pêndulo, o algoritmo do Runge-Kutta executa um passo e imediatamente depois mostra o resultado na tela, ou seja, o valor da posição do pêndulo é plotado assim que calculado, sem guardar valores em vetores.

Foi feito dessa forma devido ao número relativamente grande de cálculos que são feitos em cada passo do Runge-Kutta tendo em vista as equações (5) e (6), ficando inviável calcular vários valores para θ_1 e θ_2 inicialmente para serem mostrados posteriormente. A simulação feita dessa forma tem também a vantagem de poder ser executada pelo tempo que o usuário desejar, dando a liberdade dele poder observar padrões apresentados pelo movimento do pêndulo após decorrido um grande espaço de tempo.

A simulação desenvolvida não apresentou atrasos perceptíveis para o usuário, indicando que o método de Runge-Kutta associado a esse modelo de implementação se adequou muito bem ao problema do pêndulo duplo.

Com isso também, foi possível testar a capacidade do *Flash* de suportar cálculos como os do Runge-Kutta, tendo sido obtido esse resultado bastante satisfatório e estimulante utilizando a seguinte configuração para o computador durante o teste: AMD Atlhon™ XP 2200+ 1,79GHz 128MB de RAM que, como pode ser notado, é uma configuração que se encaixa nos padrões dos computadores pessoais atuais. Na Listagem 2, temos um trecho simplificado do código utilizado no programa.

```
// Para a função runge θ1 = xn, θ2 = yn, w1 = wn e w2 = sn
// Essa função executa o cálculo do runge-kutta
function runge(t:Number, xn:Number, yn:Number, wn:Number, sn:Number):Void {
        k1 = k(t, xn, yn, wn, sn);
        j1 = j(t, xn, yn, wn, sn);
        m1 = m(t, xn, yn, wn, sn);
        n1 = n(t, xn, yn, wn, sn);

        k2 = k(t + h/2, xn + (h/2)*k1, yn + (h/2)*j1, wn + (h/2)*m1, sn + (h/2)*n1);
        j2 = j(t + h/2, xn + (h/2)*k1, yn + (h/2)*j1, wn + (h/2)*m1, sn + (h/2)*n1);
        m2 = m(t + h/2, xn + (h/2)*k1, yn + (h/2)*j1, wn + (h/2)*m1, sn + (h/2)*n1);
        n2 = n(t + h/2, xn + (h/2)*k1, yn + (h/2)*j1, wn + (h/2)*m1, sn + (h/2)*n1);

        k3 = k(t + h/2, xn + (h/2)*k2, yn + (h/2)*j2, wn + (h/2)*m2, sn + (h/2)*n2);
        j3 = j(t + h/2, xn + (h/2)*k2, yn + (h/2)*j2, wn + (h/2)*m2, sn + (h/2)*n2);
        m3 = m(t + h/2, xn + (h/2)*k2, yn + (h/2)*j2, wn + (h/2)*m2, sn + (h/2)*n2);
        n3 = n(t + h/2, xn + (h/2)*k2, yn + (h/2)*j2, wn + (h/2)*m2, sn + (h/2)*n2);

        k4 = k(t + h, xn + h*k3, yn + h*j3, wn + h*m3, sn + h*n3);
        j4 = j(t + h, xn + h*k3, yn + h*j3, wn + h*m3, sn + h*n3);
        m4 = m(t + h, xn + h*k3, yn + h*j3, wn + h*m3, sn + h*n3);
        n4 = n(t + h, xn + h*k3, yn + h*j3, wn + h*m3, sn + h*n3);

        teta1 = xn + (h/6)*(k1 + 2*k2 + 2*k3 + k4);
        teta2 = yn + (h/6)*(j1 + 2*j2 + 2*j3 + j4);
        omega1 = wn + (h/6)*(m1 + 2*m2 + 2*m3 + m4);
        omega2 = sn + (h/6)*(n1 + 2*n2 + 2*n3 + n4);

// A função anima é executada a cada 3milissegundos e é responsável por executar
// o algoritmo do runge-kutta e atualizar a tela para os novos valores.
var idAnima:Number = setInterval(anima, 3);
function anima():Void {
        runge(tempo, teta1, teta2, omega1, omega2);
        pendulo();  // Acerta a posição do pêndulo

        // Plotar o gráfico
        mcGrafico.lineTo(mcGraficoMaior._x+30*teta2, mcGraficoMaior._y+30*teta1);

        // Atualização da tela
        updateAfterEvent();
        tempo++;
}
```

Listagem 2. Implementação do Runge-Kutta utilizada no projeto.

3. O PROGRAMA CONSTRUÍDO

O programa criado consiste em um pêndulo duplo no qual o usuário imprime as condições iniciais, escolhendo os valores das massas – m_1 e m_2, posição inicial do pêndulo, que é dada por θ_1 e θ_2 e aceleração da gravidade, g. Essa interação é feita pelo movimento do *mouse*, que pode ser arrastado pelas barras que se encontram na parte inferior do programa, como pode ser visto na Figura 2, ou inserindo os valores diretamente pelo teclado.

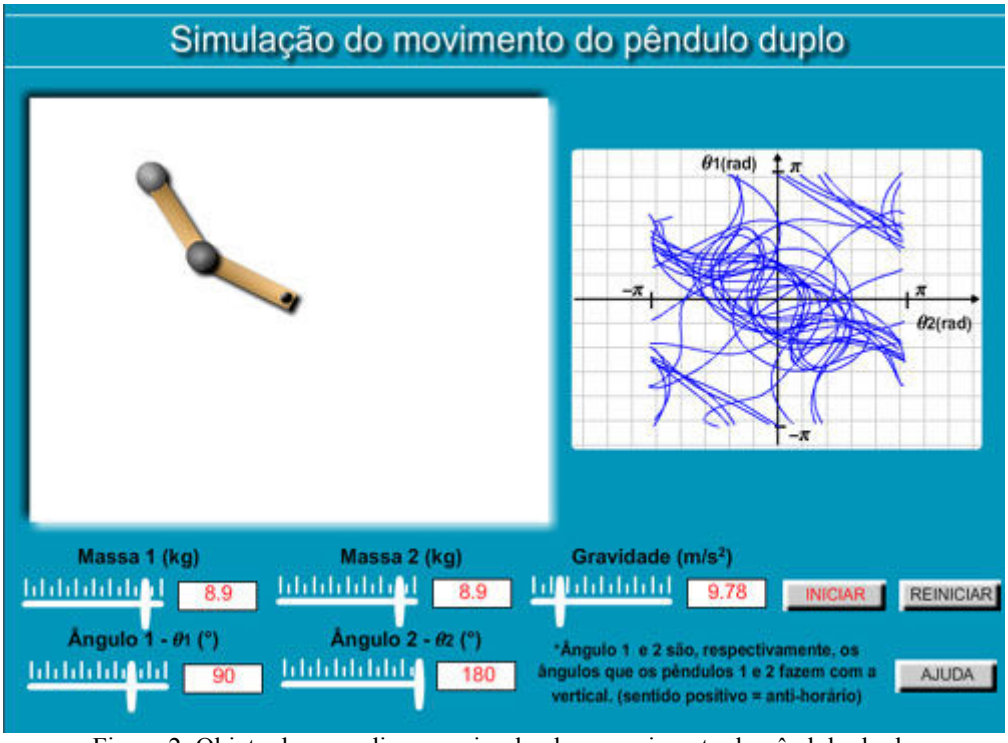

Figura 2. Objeto de aprendizagem simulando o movimento do pêndulo duplo

Depois de acertados os parâmetros supracitados, basta clicar no botão "iniciar" que o pêndulo passa a executar o seu movimento, sendo que paralelamente a esse fato é gerado um gráfico que mostra o comportamento de θ_1 em função de θ_2 durante a simulação.

Além disso, encontra-se na interface do programa um botão "Ajuda" que ao ser clicado mostra o funcionamento do programa e a física que está por trás do fenômeno representado, bem como um botão "Reiniciar" que traz a simulação de volta para o início.

4. MÉTODOS NUMÉRICOS E O ACTIONSCRIPT

O *Macromedia Flash* é uma ferramenta de desenvolvimento com muitos recursos, possibilitando inclusive a entrega de uma variedade de conteúdos dinâmicos na *web*. Uma pesquisa realizada pela Macromedia informa que o *Flash Player* está instalado em 98% dos *browsers* conectados à internet. Ao contrário de um código HTML estático, uma aplicação feita no *Macromedia Flash* pode responder rapidamente sem a necessidade de se fazer algum processamento no servidor. Essa ferramenta atende aos requisitos para os nossos estudos de caso para criação de Objetos de Aprendizagem em física.

É importante observar também que a linguagem utilizada no *Flash*, o *ActionScript*, é de fácil utilização e tornou o ambiente do *Flash* muito mais interessante de se trabalhar, pois faz com que se possa desenvolver programas bem elaborados com ambientação gráfica.

Uma novidade trazida por este projeto é a implementação de métodos numéricos em aplicações *Flash*, que só foi possível devido à evolução da linguagem *Actionscript* que agora

se encontra na sua versão 2.0. Na figura 3, temos uma comparação feita através de gráficos gerados pelo próprio *Flash*, da solução dada pelo Runge-Kutta com 500 iterações e da solução analítica para o problema do sistema massa-mola realizando um MHS que pode ser descrito pela equação (11).

$$m\frac{d^2y}{dt^2} + ky = 0 \tag{11}$$

Sendo y a coordenada vertical e os parâmetros: constante da mola k = 10 N/m e massa do bloco m = 5 kg. Pode-se notar que a diferença entre as soluções só passa a ser ligeiramente perceptível na parte final dos gráficos onde o número de iterações já executadas pelo Runge-Kutta está próximo de 500, sendo mesmo assim uma diferença muito suave.

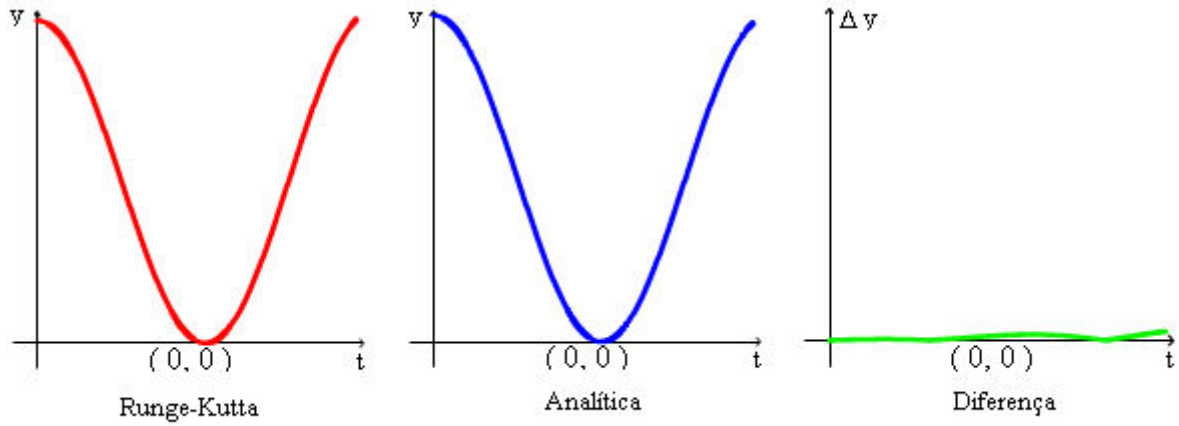

Figura 3. Comparação entre solução numérica e analítica utilizando *Actionscript*

Grandes avanços podem ser alcançados com essa técnica tanto na área de simulação quanto na utilização em tecnologias de ensino, com a construção de objetos de aprendizagem. Tendo em vista que somente com a utilização do Runge-Kutta para a resolução das equações diferenciais foi possível realizar o projeto, pode-se notar assim, a importância da implementação dos métodos numéricos em *Actionscript*. Uma conseqüência direta dessa inovação também, é que agora sem a necessidade de hipóteses simplificadoras, o *software* pode ser explorado ao máximo, de forma a aproveitá-lo na criação de objetos de aprendizagem e simuladores.

5. O USO PEDAGÓGICO

Aplicações gráficas como a do movimento pêndulo duplo podem ser utilizadas tanto dentro do ambiente da sala, com a utilização de um projetor, como fora do ambiente da sala de aula, pois outra grande vantagem do Flash é a possibilidade de executar as simulações via internet.

6. CONCLUSÃO

Como um resultado dessa aplicação, podemos apresentar o fato de que programas criados no Flash, estão sendo utilizados nas aulas de física do primeiro e segundo ano de engenharia do Instituto Tecnológico de Aeronáutica, inovando o ensino da matéria e colocando o aluno mais próximo de uma visão prática acerca do assunto. Os resultados foram muito bons,

despertando o interesse dos alunos e tirando dúvidas sobre a representação real daquilo que eles estudavam em teoria.

A principal dificuldade encontrada para a confecção dos objetos de aprendizagem foi exatamente a resolução das equações diferenciais que representam o fenômeno físico. Dessa forma, tivemos a idéia de utilizar métodos numéricos dentro do *Flash* para resolver esse problemas, cujos resultados são bastante satisfatórios, como podemos perceber diante da análise de processamento e precisão feitas e da proximidade com a realidade alcançada pelos simuladores. Estes resultados obtidos indicam que o Runge-Kutta contornou muito bem o problema encontrado, de forma que os objetos de aprendizagem criados utilizando tal método ficaram perfeitamente funcionais.

Agradecimentos

Os autores agradecem o incentivo do PIBIC no desenvolvimento de novas tecnologias de ensino, pois graças à bolsa de iniciação científica fornecida, foi possível ter início o projeto, bem como neste momento continuar a desenvolvê-lo.

REFERÊNCIAS BIBLIOGRÁFICAS

BOYCE, W.E.; DiPRIMA, R.C. **Equações Diferenciais Elementares e Problemas de Valores de Contorno**, 6a. edição, Livros Técnicos e Científicos Editora S.A., Rio de Janeiro, 1998.

FIOLHAIS, Carlos and TRINDADE, Jorge **Física no computador: o computador como uma ferramenta no ensino e na aprendizagem das ciências físicas**. Rev. Bras. Ens. Fis., Set 2003, vol.25, no.3, p.259-272. ISSN 0102-4744

LAPIDUS, L. and PINDER, G. F. **Numerical solutions of partial differential equations in science and engineering**, John Wiley & Sons Inc, Toronto, 1999;Cook, Malkus e Plesha, 1989;

PIZZI, M.. **Dominando Macromedia Flash MX**. Rio de Janeiro: Editora Ciência Moderna Ltda., 2003.

THOMPSON, J. M. T. and STEWART, H. B. **Nonlinear Dynamics and Chaos**, John Wiley & Sons, Chichester, 2002, ISBN: 0-471-87684-4

SIMULATION OF CHAOTIC SYSTEMS IN THE MACROMEDIA FLASH

Abstract: *The use of the microcomputer in the environment of the classroom can be used with great efficiency, in the simulation of physical phenomena, in the basic Physics teaching as in the most advanced studies. The Chaos Theory, that is a more complex subject, will be theme of this work. In several areas of the physics, as well as of the engineering, it is of great interest to develop analyses of no-lineal dynamic systems. With the need of the creation of new visualization tools to study this subject, this project contemplate the simulation of chaotic systems using the software Macromedia Flash®. In the solution of the equations that we will study, used techniques of numeric calculation, in the programming language actionscript of the Flash®.*

Key-words: Numeric calculation, Flash, Chaos, Learning Objects