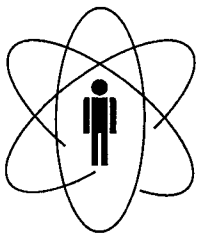
ISSN 0101-9201



CBPF - CENTRO BRASILEIRO DE PESQUISAS FÍSICAS

Rio de Janeiro

Notas Técnicas

CBPF-NT-001/96

September 1996

Newthor - Modelagem Molecular em 3D

Mário Vera

CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico

CBPF-NT-001/96

Newthor - Modelagem Molecular em 3D

Mário Vera

Centro Brasileiro de Pesquisas Físicas - CBPF/LAFEX
Rua Dr. Xavier Sigaud, 150
22290-180 - Rio de Janeiro, RJ - Brasil

Dedico este trabalho à Luiz e Léo, meus melhores professores...

Prefácio

O Projeto **Thor** consiste no primeiro programa nacional de Modelagem e Dinâmica Molecular Clássica. A primeira versão foi desenvolvida por Kleber C. Mundim e Paulo M. Bisch, idealizadores do Projeto e então Físicos pesquisadores do Centro Brasileiro de Pesquisas Físicas (CBPF). Este Projeto envolve atualmente pesquisadores de diversas instituições brasileiras, entre elas : UFRJ, USP e UFBA.

Inicialmente, o programa foi construido em *Workstations* com arquitetura *Sun SPARC*. Com a crescente difusão dos microcomputadores PCs (plataforma *Intel x86*), cuja capacidade de memória e processamento ficavam, na época, muito aquém das *Suns Workstations*, tornou-se interessante a construção de uma nova versão do programa que se adaptasse as limitações dos antigos PC's. A nova versão do programa exigia uma estruturação lógica da base de dados que otimizasse o uso de memória. À partir desta idéia surgiu o Projeto batizado de **Newthor** que envolvia também a construção de uma Interface Gráfica 3D para visualização dos resultados da Modelagem.

O **Newthor** começou a ser desenvolvido em Março de 1993. O grupo de Desenvolvimento era formado por : Roberto Valois (Analista de Sistemas) e Mário Vera (Programador).A linguagem de programação escolhida para escrever os módulos responsáveis pela Modelagem foi FORTRAN, já que era a linguagem mais difundida no meio de pesquisas em Física. A construção dos módulos em FORTRAN terminou por volta de Julho de 1995 e começaram os testes de comparação dos resultados dos cálculos da Topologia de diversas moléculas. Os resulados obtidos em computadores PC's (assim

como nas *Sun Workstations*) foram extremamente motivantes. O **Newthor** não apresentava problema algum com mémoria e sua velocidade era cerca de 5 vezes maior do que a primeira versão do **Thor** alcançava. Infelizmente, nesta mesma época, houve uma total dispersão tanto do grupo de Pesquisa como de Desenvolvimento, prejudicando assim a distribuição do **Newthor** para ser usado e testado por usuários envolvidos com Modelagem Molecular.

Apesar das dificuldades geradas pela separação dos grupos de trabalho do Projeto, tendo consciência dos exelentes resultados obtidos pelo programa, em Janeiro de 1996 comecei a construção da Interface Gráfica numa tentativa de não abandonar o Projeto inacabado. Minha intenção era fazer do **Newthor** um pacote completo e aberto à todos os grupos de Pesquisa em Modelagem Molecular do País, ou seja, eu queria que o **Newthor** fosse realmente útil ! A construção da Interface Gráfica me custou cerca de 7 meses de trabalho durante os fins de semana. Utilizei como ferramenta a biblioteca *OWL 2.0* da BORLAND. Esta ferramenta é orientada à objetos, e oferece diversas classes que facilitam bastante a comunicação com o *Windows*.

Terminada a Interface Gráfica, parti para a ligação dos módulos de Modelagem com os da Interface Gráfica. Para conseguir esta dificil tarefa construi uma DLL *(Dinamic Link Library)* contendo os módulos em FORTRAN para serem acessados pela Interface Gráfica. Com esta implementação, a parte de Modelagem ficou independente da parte de Visualização possibilitando a atualização das rotinas de cálculo à qualquer momento.

Apresento como Projeto Final de Graduação as técnicas que utilizei no desenvolvimento do Projeto **Newthor**.

Mário Vera

I - INTRODUÇÃO

Com o objetivo de abordar as principais técnicas utilizadas na construção do Projeto **Newthor** de Modelagem Molecular, o conteúdo deste texto foi divido em duas partes:

- MODELAGEM

- VISUALIZAÇÃO

A Modelagem engloba todos os módulos de cálculo da Topologia Molecular escritos em FORTRAN. São ao todo 6 módulos. Cada módulo será explicado na primeira parte do texto, assim como a estruturação lógica do programa **Newthor**.

A parte de Visualização refere-se à Interface Gráfica 3D para ambiente *Windows*. A Interface foi construída em C++, e para comunicação e acesso aos recursos do *Windows* utilizamos a ferramenta *OWL 2.0* da Bolrand que também é orientada a objeto e escrita em C++. Os detalhes de utilização das classes oferecidas pela ferramenta *OWL* foge ao objetivo deste trabalho. Esta segunda parte tem como propósito expôr os métodos utilizados para a construção da Interface Gráfica.

Os módulos escritos em FORTRAN são acessados pela Inteface Gráfica através de uma biblioteca que é *linkada* dinâmicamente aos módulos principais *(Dinamic Link Library)* , possibilitando assim a visualização do cálculo da Topologia Molecular em tempo de execução.

II - A MODELAGEM MOLECULAR

Antes de descrever os métodos computacionais e Estruturas de Dados utilizados na elaboração do programa **Newthor**, vamos a uma breve abordagem teórica de como são feitos os cálculos para se obter a Topologia Molecular :

1. Abordagem Teórica

- Num Sistema Molecular clássico aplicam-se as equações de movimento de Newton para se calcular diversas propriedades. Inicialmente é construída a superfície

de energia potencial para representar a energia do Sistema como função das coordenadas atômicas. As forças atuantes sobre cada átomo são então obtidas calculando-se o potencial gerado por uma distribuição específica de posições atômicas. A partir dessas forças, resolve-se as equações de movimento e determina-se as novas posições referentes a cada átomo. Após múltiplas iterações, teremos chegado à uma distribuição que oferece uma situação de equilibrio para o Sistema, ou seja, o potencial gerado é mínimo.

1.1 O Potencial de *Hooke* (HookPotencial)

À temperatura ambiente o comprimento das ligações químicas oscilam próximos ao seu valor de equilíbrio, obedecendo a função potencial aproximada à de *Hooke* para um Sistema de massas unidas por molas, na forma :

$$V_b = 1/2\, K_b (b - b0)^2 \qquad [1.1]$$

; onde K_b é a constante de *Hooke* associada à ligação química específica (valor tabelado[1]), b é o comprimento da ligação em um instante qualquer (Figura 1.1) e b0 o parâmetro que define o comprimento de equilíbrio da ligação.

1.2 O Potencial Angular (BondAngleBendPotencial)

Da mesma forma que para as ligações químicas, as oscilações dos ângulos entre as mesmas também podem ser descritas por um potencial harmônico :

$$V_\theta = 1/2\, K_\theta (\theta - \theta 0)^2 \qquad [2.1]$$

; onde θ é o ângulo entre duas ligações químicas envolvendo 3 àtomos concecutivos (Figura 1.1), e K_θ é a constante de *Hooke* (valor tabelado[2]) para a restituição do ângulo de equilíbrio $\theta 0$ entre as duas ligações químicas.

[1] Esta é a tabela TableHook

1.3 Potencial Diedral Próprio (ProDihPotencial)

Como no potencial ângular de ligação, faremos novamente associação ao modelo de *Hooke* para determinar o potencial envolvendo uma estrutura tetraédrica. Tomando como base a Figura 2.1, para manter uma estrutura tridimensional em um conjunto de 4 átomos , sendo um átomo central (i) ligado a 3 outros átomos (j,k,l), calculamos :

$$V\xi = 1/2\, K\xi\, (\xi - \xi 0)^2 \qquad [3.1]$$

; onde ξ o ângulo entre o plano formado pelos átomos i_j_k e o plano formado pelos átomos j_k_l , e $K\xi$ é a constante de *Hooke* (valor tabelado[3]) para a restituição ao ângulo de equilíbrio $\xi 0$ entre esses dois planos.

1.4 Potencial Diedral Impróprio (ImpDihPotencial)

A sequência Cα_N_C_Cβ ,como átomos i_j_k_l respectivamente, apresentada na Figura 1.1 garante que a cadeia lateral a qual faz parte o átomo Cβ esteja na posiçao levógira do carbono Cα. Caso a cadeia lateral esteja à direita do carbono Cα, ou seja na posição destrógira, é necessário inverter os átomos centrais na sequência de maneira que i_j_k_l sejam Cα_C_N_Cβ respectivamente. Isto fará com que Cβ troque de posição com H (Figura 2.1). Neste útlimo caso o potencial da Equação [3.1] é dito Potencial Diedral Impróprio.

---

[2] Referente à tabela TableBondAngleBend
[3] Tabela TableProDihed

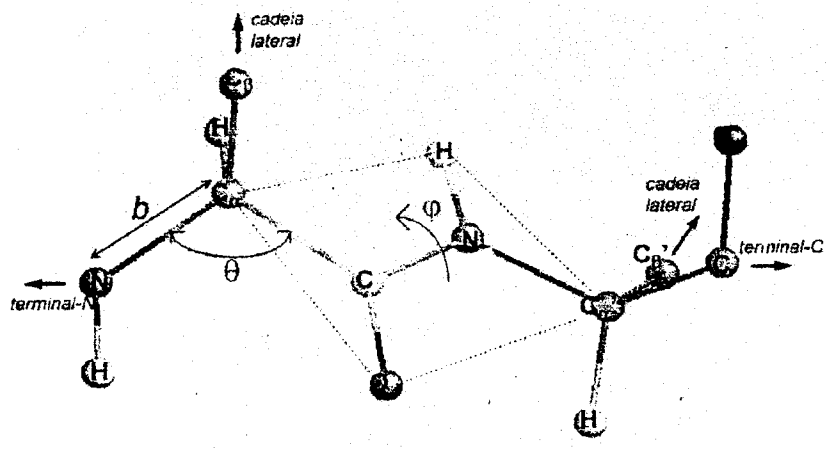


Figura-1.1 Esqueleto peptídico: b é o comprimento das ligações químicas: θ é o ângulo entre duas ligações consecutivas e φ é o ângulo torcional para ligações com liberdade de rotação (Na nomenclatura bioquímica os ângulos torcionais para as ligações do esqueleto peptídico N-C_α, C_α-C e C-N são denominados ϕ, ψ e ω respectivamente).

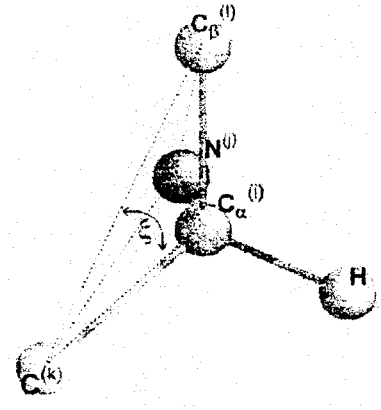


Figura-2.1 Estrutura tetraédrica do carbono-α. ξ é o ângulo entre os planos i_j_k e j_k_l.

1.5 Átomos não ligados (NoBondPotencial)

As interações entre átomos não ligados quimicamente podem ser tratadas por potenciais efetivos, compostos por termos de curto e longo alcance que incluem a rèpulsão e atração de Van der Walls e o termo eletrostático de Coulomb.Quando não há reações químicas, duas moléculas separadas reagem entre si obedecendo a potenciais deste tipo.Em uma molécula individual, com exceção dos primeiros e segundos vizinhos quimicamente ligados, cujas interações sao tratadas pelos

potenciais harmônicos já descritos, a interação de cada átomo com os outros átomos da molécula é também descrita por termos de Van der Walls e Coulomb.

O termo repulsivo de Van der Walls surge com a correção na equação de estado dos gases ideais :

$$PV = nRT \qquad [4.1]$$

A energia potencial eletrostática é dada pela lei de Coulomb :

$$V_{col.} = qiqj / 4\pi \varepsilon 0 \varepsilon r \qquad [5.1]$$

; onde *qi* e *qj* são as cargas residuais sobre os átomos *i* e *j*, separados pela distância r. $\varepsilon 0$ é a permissividade do espaço livre e ε é a constante dielétrica que corrige $\varepsilon 0$ para considerar a polarização do meio.

A interação eletrostática varia com o inverso da distância de separação entre os átomos, sendo portanto de longo alcance.

2. Codificação

Na codificação do programa **Newthor** a atenção da equipe de desenvolvimento estava concentrada em dois fatores principais : Modularização e Estruturação Lógica.

2.1 Modularização

A primeira versão do programa **Thor** foi construída num bloco único de código. Isto dificultava a depuração de erros, assim como impossibilitava a compilação do módulo em máquinas com baixa quantidade de memória (PC's por exemplo). Para contornar este problema foram criados, na nova versão, 6 módulos independentes que são compilados em separado e *linkados* com a rotina principal.

Especificação dos Módulos

NewThor.for - módulo que contém a rotina principal da parte de Modelagem Molecular. Este módulo faz a tranferência de controle entre as rotinas que definem o ciclo principal do programa[4].

LoopThor.for - onde estão localizadas as rotinas responsáveis pelos principais ciclos do programa.

FillThor.for - contém as rotinas responsáveis pelo preenchimento das estruturas relacionadas às tabelas internas, contendo constantes de ligação, e externas que são construídas de acordo com a estrutura da molécula em estudo.

UtilThor.for - módulo que oferece diversos procedimentos e funções à todos os módulos do programa.

DoThor.for - dispara as iterações referentes ao cálculo dos potenciais harmônicos, para que atuiem sob todos os átomos pertencentes à molécula.

PotThor.for - contém as rotinas de cálculo dos potenciais harmônicos.

2.2 Estrutura Lógica da Base de Dados

A primeira versão do programa **Thor** de Modelagem Molecular apresentava sérios problemas para execução em microcomputadores devido ao consumo excessivo de memória, resultado de uma estruturação de dados que não se preocupava em otimizar este consumo.

Endereçamento Indireto com múltiplos níveis

A otimização de memória foi então o segundo fator de atenção para o grupo de desenvolvimento do **Newthor**. Para conseguir esta economia de memória simulou-se, na fase de reestruturação lógica, o modo de endereçamento indireto com

[4] Este módulo é substituído na Interface Gráfica pela rotina que dispara o cálculo da Topologia Molecular.

múltiplos níveis[5]. Este enderaçamento funciona em todas as tabelas utilizadas pelo programa.

A tabela Type & Mass

As Estruturas de Dados foram então baseadas em vetores e matrizes definindo tabelas cujo os elementos "apontam" para níveis mais baixos do endereçamento indireto até a tabela principal com a representação usual dos elementos químicos. São ao todo 37 elementos[6] reconhecidos pelo **Newthor** . Esta tabela foi batizada de Type & Mass , já que guarda também os valores de massa dos 37 átomos respectivamente :

IAC	TYPE	MASS	DESCRIPTION
1	O	15.9994	carbonyl oxygen (C=O)
2	OM	15.9994	carboxyl oxygen (CO-)
3	OA	15.9994	hydroxyl oxygen (OH)
4	OW	15.9994	water oxygen
5	N	14.0067	peptide nitrogen (N or NH)
6	NT	14.0067	terminal nitrogen (NH2)
7	NL	14.0067	terminal nitrogen (NH3)
8	NR5	14.0067	aromatic N (5-ring, 2 bonds)
9	NR5*	14.0067	aromatic N (5-ring, 3 bonds)
10	NP	14.0067	porphyrin nitrogen
11	C	12.0110	bare carbon (peptide, C=O, C-N)

[5] Para uma definição de Endereçamento Indireto ver [TANEMBAUM90] pag. 262

[6] Nem todos os elementos são átomos únicos, alguns deles formam estruturas bem definidas , por exemplo CR*6 .

12	CH1	13.0190	aliphatic CH-group
13	CH2	14.0270	aliphatic CH2-group
14	CH3	15.0350	aliphatic CH3-group
15	CR51	13.0190	aromatic CH-group (5-ring)
16	CR61	13.0190	aromatic CH-group (6-ring)
17	CB	12.0110	bare carbon (5-,6-ring)
18	H	1.0080	hydrogen bonded to nitrogen
19	HO	1.0080	hydroxyl hydrogen
20	HW	1.0080	water hydrogen
21	HS	1.0080	hydrogen bonded to sufur
22	S	32.0600	sulfur
23	FE	55.8470	iron
24	ZN	65.3700	zine
25	NZ	14.0067	Arg NH (NH2)
26	NE	14.0067	Arg NE (NH)
27	P	30.9738	phosphor
28	OS	15.9994	sugar or ester oxygen
29	CS1	13.0190	sugar CH-group
30	NR6	14.0067	aromatic N (6-ring, 2 bonds)
31	NR6*	14.0067	aromatic N (6-ring, 3 bonds)
32	CS2	14.0270	sugar CH2-group

33	SI	28.0800	silicon
34	NA	22.9898	sodium
35	CL	35.4530	clorine
36	CA	40.0800	calcium
37	MG	24.3050	magnesium

Tabela 1.2 *Type & Mass*

Todas as Estruturas presentes no programa endereçam indiretamente esta tabela.

As Tabelas Internas (constantes de ligação)

Estas tabelas armazenam os valores para as constantes referentes a cada ligação que possa eventualmente estar presente na molécula sendo estudada. São elas :

TableHook - constantes de força de ligação entre 2 átomos.

TableBondAngleBend - constantes da força Angular para ligações entre 3 átomos.

TableProDihed - cosntantes da força Diedral Própria para ligações entre 4 átomos.

TableImproDihed - constantes da força Diedral Imprópria para ligações entre 4 átomos.

TableVanderWalls - constantes da força de Van der Walls entre átomos não ligantes.

Através dos valores presentes nestas tabelas podemos calcular a resultante agindo em cada átomo, e com isso fazer uma aproximação da posição referente a cada átomo e por fim determinar a Topologia Molecular .

As Tabelas Externas

Toda molécula é formada por átomos ligantes que juntos formam a Estrutura Molecular. Ao receber a informação da estrutura referente à uma determinada molécula, uma rotina responsável pelo preenchimento das Tabelas de Ligações Moleculares, ditas tabelas externas, é acionada. Estas tabelas descrevem como estão ligados todos os átomos que formam a molécula.

Na construção das tabelas externas, novamente o endereçamento indireto foi utilizado, este agora com dois níveis. O primeiro nível de endereçamento "aponta" para um vetor denominado TypeMassIndx que ,como o nome já diz, guarda o endereço de cada átomo presente na Estrutura Molecular, com relação à tabela Type & Mass.

Por exemplo, a entrada :

```
TypeMassIndx[6] = 18
```

significa que o sexto átomo (na ordem de entrada) da molécula em estudo é um Hidrogênio !

As demais tabelas externas determinam,com endereçamento referente a tabela TypeMassIndx (3 níveis de endereçamento), a maneira como os átomos estão organizados na molécula para formarem a Estrutura Molecular. São elas :

IndxHook - lista de todas as ligações entre 2 átomos presentes na molécula estudada.

IndxBend - lista de todas as ligações entre 3 átomos presentes na molécula estudada.

IndxProdihed - lista de todas as ligações entre 4 átomos, do tipo Diedral Própria, presentes na molécula estudada.

IndxImpdihed - lista de todas as ligações entre 4 átomos, do tipo Diedral Imprópria, presentes na molécula estudada.

IndxNoBond - matriz contendo a lista de vizinhos a serem excluidos com relação a cada átomo no cálculo do potencial referente às forças de Van der Walls e Coulomb.

2.3 Algorítmos de Ordenação

Para acelerar o acessoas informações contidas nas tabelas ordenamos algumas destas. A escolha das tabelas a serem ordenadas segui o critério de prioridades :

- Tamanho da tabela.
- Frequência de acessos aos dados da tabela.

Seguindo esta estratégia não foi necessário preocupar-se com a ordenação de todas as tabelas. Por exemplo, a tabela de maior frequência de acesso é a IndxHook pois serve de base para a construção de todas as outras tabelas externas (com exceção da *IndxNoBond*) tendo sido ordenada.

A escolha do algoritmo de ordenação não foi muito difícil pois a ordenação é feita apenas uma vez no caso das tabelas internas (quando o programa é carregado). Por esta razão tratando tabelas grandes, como a de Van der Walls, o algoritmo da bolha *(BubbleSort)*[7] já se demonstrou eficiente.

2.4 Algoritmos de Busca

A escolha dos algoritmos de busca foi feita de acordo com a ordenação das tabelas. Caso ordenada a busca na tabela era feita utilizando-se o algorítmo de Busca Binária, caso contrário a busca sequencial foi a opção adotada.Ocorreu todavia uma exceção, que foi a busca feita na tabela interna de Van der Walls. Este algoritmo simulou um *Hash*[8] de chave 37 (o número de átomos existentes na tabela base *Type & Mass*). Tendo a tabela de constantes de Van der Walls 37^2 átomos (todos com todos), a rotina *FillIndxVanderWalls* que preenche a estrutura IndxVanderWalls,que por sua vez indica o endereço de uma determinada ligação Átomo_I <=> Átomo_J na tabela de constantes, segue a equação :

IndxVanderWalls [ConnectionNumber] = Atom_I*37 + Atom_J [1.2]

[7] Para uma descrição do algoritmo *BubbleSort* ver [HOROWITZ]
[8] Para uma definição de Hash [HOROWITZ]

2.5 Rotinas de cálculo das Forças Atômicas

A cada iteração, em busca do equilibrio que nos trará a Topologia correta da molécula em estudo, devemos calcular a força resultante sobre cada átomo pertencente á molécula. Esta força resultante é um somatório das forças referentes aos potenciais harmônicos tabelados. O cálculo das forças é disparado no módulo LoopThor.for especificamente na rotina *AllForce*. A rotina AllForce aciona os procedimentos respectivos à cada potencial harmônico :

DoHookForce - adiciona o valor do Potencial de *Hooke* referente a cada ligação dupla, ao vetor resultante de forças *(Force)* atuantes nos átomos que formam a ligação.

DoBendForce - adiciona o valor do Potencial Angular referente a cada ligação tripla, ao vetor resultante de forças *(Force)* atuantes nos átomos que formam a ligação.

DoProDihForce - adiciona o valor do Potencial Diedral Próprio referente a cada ligação de 4 átomos, ao vetor resultante de forças *(Force)* atuantes nos átomos que formam a ligação.

DoImpDihForce - adiciona o valor do potencial de *Hooke* referente a cada ligação de 4 átomos (que obedece à estrutura tetraédrica imprópria), ao vetor resultante de forças *(Force)* atuantes nos átomos que formam a ligação.

DoNoBondForce - adiciona o valor do Potencial de VanderWalls e de Coulomb referente a cada átomo, ao vetor resultante de forças *(Force)* atuante no átomo analizado.

2.6 Determinação do Potencial Molecular

O valor de potencial que é adicionado à estrutura *Force* (contendo os vetores resultantes referentes a cada átomo da molécula), é cálculado no módulo **PotThor.for**, com base nas tabelas internas e nas rotinas respectivas ao potencial desejado :

HookPotencial - calcula o Potencial de Hook.

BendPotencial - calcula o Potencial Angular.

ProDihPotencial - calcula o Potencial Diedral Próprio.

ImpDihPotencial - calcula o Potencial Diedral Impróprio.

NoBondPotencial - calcula os Potenciais das forças de longo alcance.

TotalPotencial = HookPotencial + BendPotencial + ProDihPotencial +
ImpDihPotencial + NoBondPotencial

[2.2]

Observações :

O valor total a cada iteração de HookPotencial, por exemplo, é o somatório dos potenciais respectivos a cada ligação entre dois átomos.

Da equação [2.2] tiramos que o potencial total (TotalPotencial) a cada iteração é, por sua vez, o somatório dos potenciais harmônicos referentes a cada força de ligação.

2.7 Otimização do Potencial Total

Como foi explicado no cálculo do Potencial Molecular, a cada iteração chega-se à um valor de potencial total (TotalPotencial). A diferença dos valores entre as iterações é comparada à precisão desejada pelo usuário. De acordo com esta diferença determinamos o ponto de parada da Otimização do Potencial Molecular. Estes cálculos estão na rotina **OtmzGeo** no módulo **LoopThor.for** . O principal trecho de código desta rotina é :

```
While (Delta_Pot > Precision)

    OtmzGeo();
```

Codificação 1.2 *Otimização do Potencial Total*

Desta forma é calculada a Topologia de uma molécula recebida como entrada[9] pelo programa **Newthor** !

[9] Entenda-se como entrada, um arquivo com dados referentes à molécula ou à molécula editada na Interface Gráfica !

III - VISUALIZAÇÃO (Técnicas Utilizadas)

As operações de Entrada e Saída (I/O) no programa **Newthor** são feitas através de arquivos, seguindo uma filosofia antiga, da época que ainda não se dispunha de ferramentas de visualização como atualmente. Esta forma de controlar "I/O" dificulta em muito o trabalho dos pesquisadores e ainda afasta certos usuários que não possuem conhecimento suficiente de recursos computacionais ditos básicos (Edição etc...), nem tão pouco estão familiarizados com os modernos e muitas vezes complexos ambientes de trabalho disponíveis hoje em dia.

Com a utilização de Interfaces Gráficas para visualização dos resultados da Modelagem Molecular, tudo ficou mais simples e agradável para o usuário. Atualmente o grupo de pesquisadores envolvidos com o Projeto **Thor** fazem uso de diversas implementações de Interfaces Gráficas disponíveis em domínio público pela *Internet*. A idéia de se construir uma Interface Gráfica própria para o Projeto **Thor** surgiu quando notamos[10] que cada ferramenta que estava sendo utilizada pecava pela falta de certos recursos disponíveis em outras Interfaces. Assim me dispus a construir uma Interface Gráfica 3D para ambiente *Windows* que pudesse servir de ferramenta para visualização dos cáculos de Topologia Molecular feitos pelo **Newthor**.

Para apresentar a Interface Gráfica serão descritos os métodos utilizados em sua construção :

1. Utilização de Bitmaps

A representação dos átomos foi feita através de imagens pré-construídas e já com os efeitos responsáveis pela noção de volume necessária para representação em 3D. As esferas foram construídas utilizando-se um editor de imagens e armazenadas em arquivos contendo *Bitmaps*.

Bitmap é o nome dado à uma estrutura de dados contendo informações referentes a representação de quadros de imagem. Ao carregar o *Bitmap* na memória temos o valor de cor endereçado por cada pixel (menor unidade de imagem) que

[10] O grupo de Desenvolvimento.

compõe o quadro mapeado na estrutura de dados do *Bitmap*. Desta forma o complexo e custoso processo de representação de esferas em 3D reduz-se a uma simples operação de transferência de bloco de memória.

Esta estratégia mostrou-se altamente eficaz e justamente adequada para nosso propósito, pois a construção das esferas em tempo dinâmico inviabilizaria o projeto.

2. Sistemas de Coordenadas

Uma condição primordial na construção de uma Interface Gráfica é torná-la indepêndente de qualquer dispositivo *(Device Independent)*. Seguindo esta filosofia deve-se tentar criar um sistema de coordenadas que possa se adaptar à qualquer dispositivo que esteja sendo utilizado para visualização da imagem. Os objetos que compõem a imagem foram criados num sistema de coordenadas que pode não estar de acordo com a condição de independência de dispositivos. Devemos então estabelecer uma transformação de coordenadas, para representar os objetos da imagem no sistema idealizado. As coordenadas dos objetos serão denominadas Coordenadas de Mundo *(World Coordinates)*, e as coordenadas do sistema ideal de Coordenadas Normalizadas *(Normalized Coordinates)*.

O Sistema de Coordenadas Normalizadas segue o modelo de coordenadas cartesianas, porém seus eixos são de comprimento finito 1, estando as coordenadas limitadas ao intervalo [0,1] .

Após a normalização das Coordenadas de Mundo devemos nos preocupar com as coordenadas do dispositivo sendo utilizado. Esta transformação se reduz à um redimensionamento de escalas, onde o intervalo [0,1] passa a ser o determinado pelas dimensões do dispositivo. A Figura 1.2 ilustra a normalização das coordenadas de mundo :

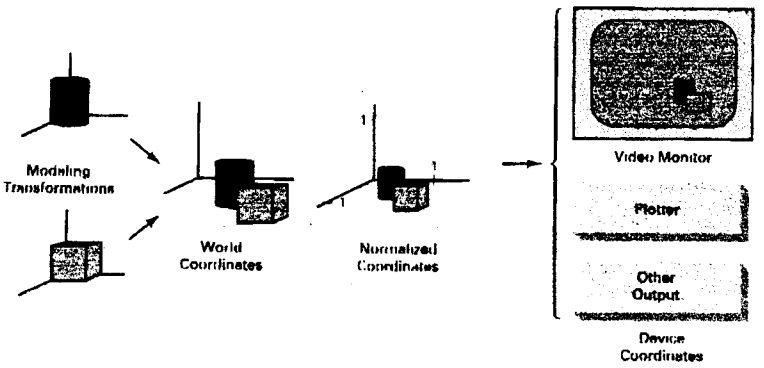


Figura 1.2 *Normalized Coordinates*

3. Transformações Matemáticas em 3D

A representação da imagem através de estruturas contendo as coordenadas dos pontos que formam a imagem torna possível a aplicação de operações matemáticas sob estes pontos. As transformações estudadas em Geometria Analítica podem ser utilizadas para obtermos uma visualização da imagem sob diferentes ângulos, assim como dar movimento à imagem. As transformações utilizadas pela Interface são descritas adiante.

3.1 A Matriz Transformação

A implementação das transformações faz uso de uma Matriz principal dita Matriz Transformação (TMatrix). A escolha desta estrutura se justifica pelas propriedades oferecidas principalmente pela operação de multiplicação. Seguidas transformações (Rotação + Translação + Rotação) que seriam feitas separadamente podem ser agrupadas numa única Matriz Transformação agilizando assim todo o processo.

3.2 Coordenadas Homogêneas

Como já foi dito todas as transformações são realizadas através da operação de multiplicação das Coordenadas do ponto pela Matriz Tranformação. Existe porém uma transformação que não poderia ser feita através da multiplicação de matrizes[11] .

A Translação é uma transformação definida sob a forma :

$$X2 = X1 + D \qquad [3.2]$$

$$Y2 = Y1 + D \qquad [4.2]$$

Este pequeno detalhe do deslocamento ser adicionado (e não multiplicado como nas outras transformações) às coordenadas do ponto (X1,Y1) nos obriga a definir uma representação para nosso sistema de coordenadas tal que introduza-se mais uma coordenada W, uniformizando assim a operação de multiplicação de matrizes para todas as transformações. Tal representação foi batizada de Coordenadas Homogêneas.

Em Coordenadas Homogêneas, as Coordenadas de um ponto passam a ser :

$$[\ XW\ ,\ YW\ ,\ ZW\ ,\ W\ ] \qquad [5.2]\ ^{12}$$

3.3 Transformação de Escalas

A operação de multiplicação por um fator K aplicada às Coordenadas de um ponto P1 é feita seguindo a equação :

$$P2 = P1 * K \qquad [6.2]$$

[11] Admitindo um vetor sendo uma matriz de uma dimensão.

[12] Exceto no caso de utilizarmos Projeção em Perspectiva, o valor de W pode ser sempre igual a 1.

A Transformação de Escala pode ser obtida através da multiplicação das coordenadas de um ponto qualquer pela Matriz Transformação:

Kx	0	0	0
0	Ky	0	0
0	0	Kz	0
0	0	0	1

Matriz 1.2 Transformação de Escala

Através da multiplicação de todos os pontos da imagem podemos alterar de diversas formas o tamanho da imagem. Podemos ainda "esticar" a imagem aplicando diferentes valores para Kx,Ky e Kz .

3.4 Rotação

Para aplicar a operação de Rotação de ϕ graus a um ponto com relação ao eixo das ordenadas por exemplo, devemos recorrer à Geometria Analítica e fazer uso da matriz rotação :

COSϕ	0	-SINϕ	0
0	1	0	0
SINϕ	0	COSϕ	0
0	0	0	1

Matriz 2.2 Rotação sob o eixo Y

A Figura 1.3 ilustra rotação de um peão de xadrez em relação ao eixo Y :

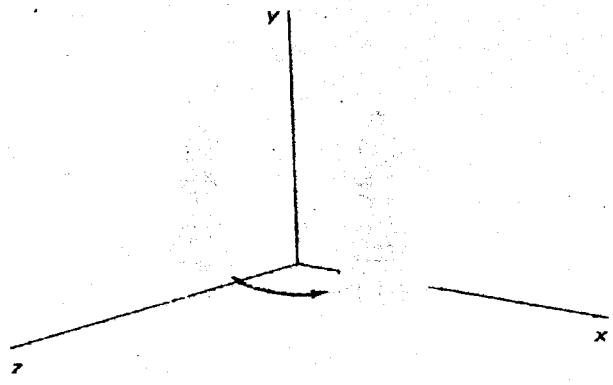


Figura 1.3 *Rotação em Y*

3.5 Translação

Para transladar a imagem podemos fazer uso da última linha da Matriz Transformação :

1	0	0	0
0	1	0	0
0	0	1	0
Dx	Dy	Dz	1

Matriz 3 Translação

Ao multiplicar um ponto qualquer pela Matriz Translação acima estaremos causando um deslocamento de Dx , Dy e Dz aos respectivos valores de coordenadas do ponto.

3.6 Rotação sob um eixo arbitrário

No caso de querermos fazer uma rotação so um eixo que não seja os cartesianos, devemos realizar duas transformações. Inicialemente uma translação para um dos eixos cartesianos que servirá de base para a rotação. Faz-se então a rotação utilizando-

se as matrizes de rotação sob o eixo escolhido como base. Finalmente realiza-se uma translação da imagem para o eixo original. A Figura 2.3 ilustra este processo que é implementado na Interface Gráfica na rotação da molécula sendo visualizada :

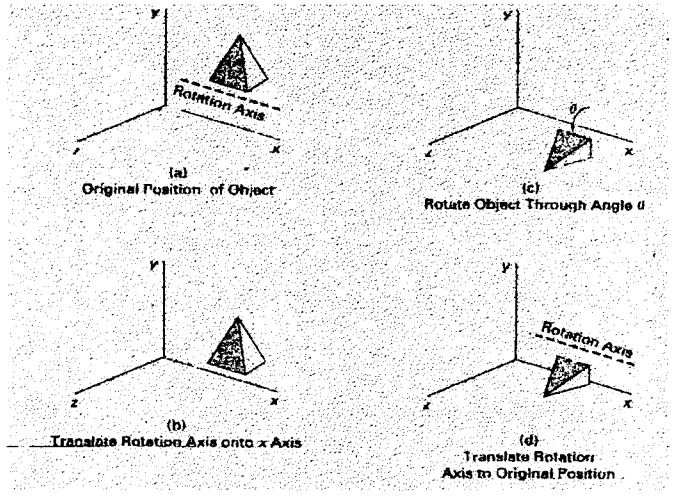


Figura 2.3 *Rotação sob um eixo arbitrário*

4. Visualização em 3D

A visualização de uma imagem em 3 dimensões envolve passos análogos aos de se tirar uma fotografia. Para fotografar uma cena, primeiramente temos de posicionar a câmera numa posição do espaço. Depois é preciso decidir a orientação da câmera (Figura 1.4), e a maneira como iremos encaixar a imagem no campo de visualização da câmera. Finalmente acionamos o disparador. A imagem é capturada e dimensionada de acordo com o tamanho da "janela" de projeção da câmera.

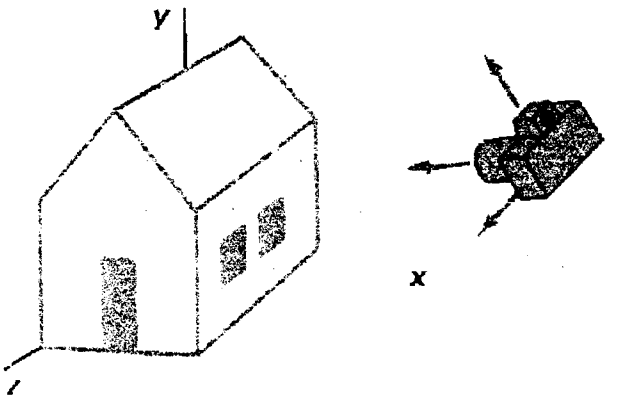


Figura 1.4 *O modelo da câmera*

A obtenção da imagem final projetada na tela de visualização sob as coordenadas de dispositivo envolve uma série de transformações. A Figura 2.4 resume os passos envolvidos na obtenção das coordenadas de dispositivo :

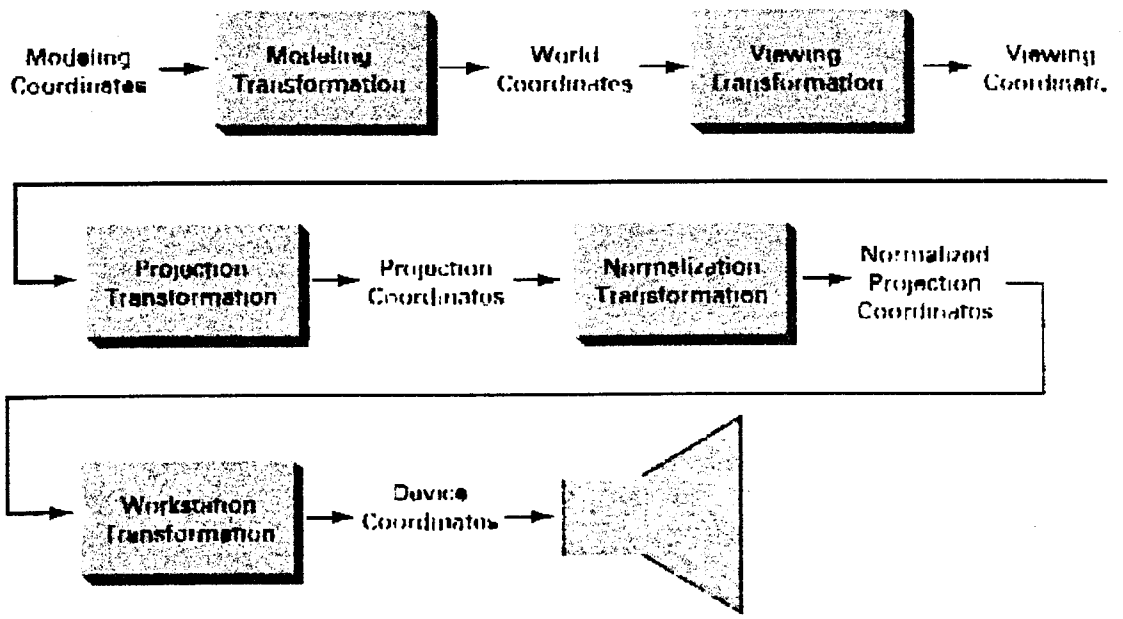


Figura 2.4 As Transformações de Coordenadas

4.1 Coordenadas de Visualização

Seguindo o modelo de câmera analógica, é necessário termos um plano onde a imagem possa ser projetada. Chamaremos este plano de *view plane*[13] .Usualmente, o *view plane* é posicionado perpendicularmente ao eixo Z, ainda nas Coordenadas de Mundo (Figura 3.2). O próximo passo é projetar a imagem no *view plane*.

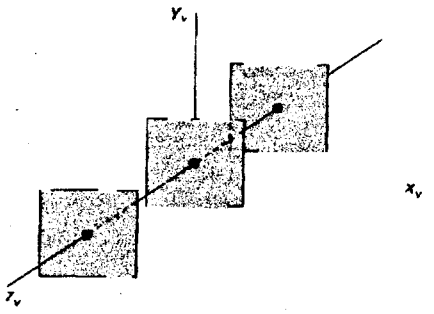


Figura 3.2 *view plane sobre o eixo Z*

4.2 Projeção

A projeção da imagem pode ser em paralelo ou em perspectiva. A projeção em perspectiva é indicada quando queremos passar a noção de profundidade à imagem projetada, o que não se aplica ao nosso caso já que a profundidade de uma molécula é desprezível comparada com suas medidas.

Projeção em Paralelo

Podemos especificar a Projeção em Paralelo com um vetor de projeção (projection vector) que define a direção das linhas de projeção.Quando a direção de projeção é perpendicular ao *view plane* temos a Projeção Ortográfica, caso contrário a

[13] Também é conhecido como *projection plane*.

Projeção é dita Oblíqua. A Projeção utilizada foi a Ortográfica :

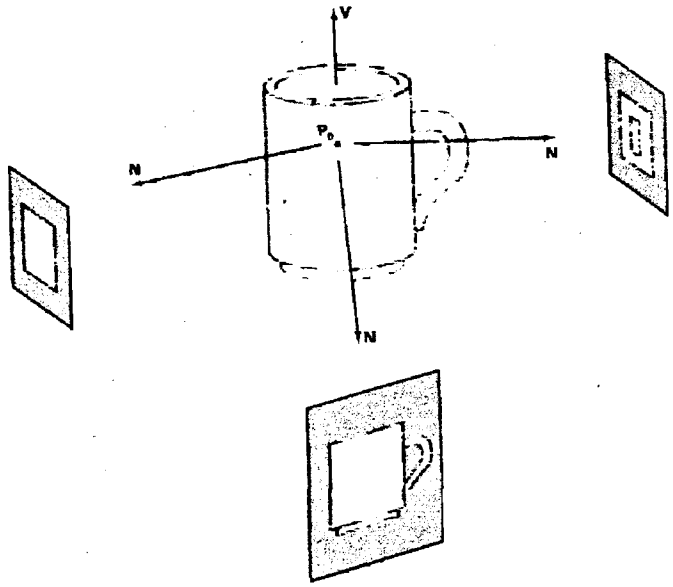


Figura 4.2 *Projeção Ortográfica*

As equações de transformação que envolvem a Projeção Ortográfica são muito simples.Para projetar por exemplo um ponto de Coordenadas (Xi,Yi,Zi) no *view plane* localizado na coordenada Zvp basta :

$$Xvp = Xi \qquad [1.2]$$

$$Yvp = Yi \qquad [2.2]$$

4.3 Coordenadas de Dispositivo

O passo final é a transformação das Coordenadas de Visualização, já projetadas no *view plane* definindo a "janela de visualização" ,nas coordenadas correspondentes ao dispositivo utilizado. Para isto basta fazer uma mudança de escala fazendo a janela de visualização ter as mesmas dimensões da tela do dispositivo.

5. Interação

Para podermos oferecer os recursos de edição tivemos que simular um localizador que captura o *click* do usuário num determinado ponto do espaço de visualização, assim como seleciona um átomo já editado para conectá-lo a outro.

Para isto construímos uma rotina que compara o ponto de *click* (Xc,Yc) com cada átomo pertencente à molécula, até encontrar um átomo cujo centro esteja próximo o suficiente para supormos ter sido ele o escolhido.

O cálculo segue a fórmula :

$$|X - Xc| + |Y - Yc| < \text{APERTURE} \qquad [7.2]$$

; sendo APERTURE a distância estipulada como limite.

Para agilizar a busca fazemos uso da estrutura *Z-Sort* que ordena os átomos em ordem crescente de distância em relação ao usuário. A busca começa pelos átomos mais próximos, já que estes oferecem maior chance de terem sido escolhidos !

6. Superfícies Ocultas

Para podermos obter uma visualização coerente com a imagem real em 3 dimensões devemos nos preocupar com a sobreposição de objetos. Existem diversos algorítmos para implementação de métodos que tratam este problema em específico. O algoritmo adotado aqui foi o que simula a estratégia adotada na pintura de um quadro.

6.1 O Algorítmo do Pintor

O "Algorítmo do Pintor" *(The Painter Algorithm)* constrói a imagem de forma a desenhar primeiro os objetos de fundo, como acontece na pintura de uma quadro. Objetos de coordenadas mais distantes são apresentados primeiro para que a ordem de sobreposição de objetos da imagem real seja respeitada.

Em nossa implementação supomos a ordenação em Z. Fazemos uso da estrutura *Z-Sort* e desenhamos os átomos de acordo com a ordenação do vetor *Z-Sort*.

Observação :

Um problema específico ao caso de visualização de moléculas é a representação das conexões dos átomos. A dificuldade está em achar um algorítmo que possibilite a correta ordem de construção da molécula. O algorítmo desenvolvido aqui desenha inicialmente as conexões dos átomos mais distantes, ou seja, de menor coordenada em Z para depois desenhar o átomos em si. No último passo do algorítmo o átomo mais distante e desenhado e todas as conexões referentes a este átomo já foram desenhadas !

IV - CONCLUSÃO

Gostaria inicialmente de salientar a importância deste projeto em minha graduação como Bacharel em Informática. Este projeto envolveu conceitos que estudei durante minhas duas Faculdades : Engenharia Química (imcompleto) e Informática ; tendo me acompanhado durante toda formação. Foi através dele que pude fixar os conceitos aprendidos quase que a cada cadeira cursada.

Em termos de resultados, o programa se demonstrou muito rápido em comparação com as versões que já tive acesso. A Interface Gráfica ainda apresenta alguns problemas, e lhe falta diversos recursos importantes mas que podem ser implementados rapidamente.

Espero através deste trabalho poder acrescentar benefícios ao Projeto **Thor** de Modelagem e Dinâmica Molecular Clássica.

V - PROJETOS FUTUROS

Pretendo futuramente acrescentar a parte referente a Dinâmica Molecular, assim como os diversos recursos de Modelagem presentes nas últimas versões do **Thor**. Tais mudanças poderão ser rapidamente conseguidas, formando mais uma adaptação de código.

VI - BIBLIOGRAFIA

MODELAGEM

[PASCUTTI96] - Pascutti, Pedro - "Thor - Dinâmica Molecular Clássica"- Monografia de Doutorado. Instituto de Física da Universidade de São Paulo.1996 São Paulo.

[BURK&ALL82] - Burkert & Allinger, Ulrich, Norman L.," Molecular Mechanics ACS"-Monograph 177, 1982 Washington.

[HOROWITZ] - E. Horowits e S. Sahci, "Fundamentals of Computer Algorithms", Computer Science Press.

[TANENBAUM90] - Tanenbaum, Andrew S., "Structured Computer Organization" Third Edition, 1990 Prentice Hall.

VISUALIZAÇÃO

[HARRINGTON87] - Harrington, Steven, "Computer Graphics - A Programming Approach" Second Edition, 1987 McGRAW-HILL INTERNATIONAL EDITIONS.

[HEARN&BAKER94] - Hearn, Donald & Baker, M. Pauline, "Computer Graphics" Second Edition, 1994 McGRAW-HILL INTERNATIONAL EDITIONS.

[PETZOLD92] - Petzold, Charles - "Programming Windows 3.1", Third Edition , 1992 Microsoft Press.

NOTAS TÉCNICAS é uma pré-publicação de trabalhos técnicos-científicos do CBPF.
Pedidos de cópias desta publicação devem ser enviados aos autores ou ao:

Centro Brasileiro de Pesquisas Físicas
Área de Publicações
Rua Dr. Xavier Sigaud, 150 – 4º andar
22290-180 – Rio de Janeiro, RJ
Brasil

NOTAS TÉCNICAS is a preprint of technical reports from CBPF.
Requests for copies of these reports should be addressed to:

Centro Brasileiro de Pesquisas Físicas
Área de Publicações
Rua Dr. Xavier Sigaud, 150 – 4º andar
22290-180 – Rio de Janeiro, RJ
Brazil