MVP: MÓDULO VISUALIZADOR DE ESTRUTURAS 3D DE PROTEÍNAS

Daniel Moisés Gonzalez Clua[1], Vivian Dorat Betoni[2], Lílian Fernanda Castilho de Almeida[3], Luis Aníbal Diambra[4]

[1-3] Universidade do Vale do Paraíba, Av. Shishima Hifumi, 2911 – Urbanova
S. José dos Campos, SP, 12244-000,
dmgc29785@yahoo.com.br (1), vdb29586@yahoo.com.br (2), lylyanjinha@yahoo.com.br (3),
ldiambra@gmail.com (4)

Resumo – Este trabalho consiste em um visualizador tridimensional e dinâmico de estruturas de proteínas, que usa como base arquivos no formato PDB (Protein Data Bank), onde são extraídas informações correspondentes às posições dos átomos e seu tipo para a identificação gráfica dos diferentes elementos. A estrutura pode ser apresentada sob formato *backbone* (linhas conectando os carbonos-α) ou *spacefill* (esferas atômicas diferenciadas por cores, ilustrando o tipo de átomo, com raios proporcionais aos raios de van der Walls). A realização da rotação da molécula e a interpolação entre dois *frames* são feitas com suavização entre a transição de uma imagem à outra. A linguagem de programação usada para seu desenvolvimento foi a C++, juntamente com bibliotecas de OpenGL (Open Graphics Library) para a geração de gráficos das estruturas tridimensionais de forma simples e robusta.

Palavras-chave: Visualização de Proteínas, OpenGL, C++.
Área do Conhecimento: Ciências Exatas, Ciência da Computação, Bioinformática

Introdução

Tradicionalmente, a biologia molecular se dedicou ao estudo de genes ou proteínas isolados como método para determinar as funções desse único gene ou proteína. Entretanto, para determinar os princípios gerais dos complexos processos biológicos subjacentes, também é necessário examinar um grande número de elementos em paralelo. As novas tecnologias, que permitem o estudo de sistemas biológicos em escalas massivas, têm dado lugar a abordagens mais integrativas, gerando uma quantidade exponencialmente crescente de dados e transformando profundamente as ciências biológicas. Por esse motivo, considerável esforço vem sendo concentrado na interpretação desses dados. Para analisar, visualizar, armazenar e navegar grandes bancos de dados, novos métodos computacionais e estatísticos se tornam absolutamente essenciais nesta área. A biologia estrutural não tem ficado fora desta tendência e neste sentido têm sido desenvolvidos numerosos programas seja para analisar superfícies de proteínas ou para simular as interações entre as mesmas ou com outros substratos.

As proteínas constituem partes dos tecidos biológicos, muitas delas funcionando como enzimas. Juntamente com os açucares e lipídios compõem a alimentação básica dos animais. São substâncias sólidas, incolores, coloidais, geralmente insolúveis em solventes orgânicos. Podem possuir alguma solubilidade em água, ou ainda em soluções aquosas diluídas de ácidos, bases ou sais [1].

As proteínas podem ter 4 tipos de estrutura dependendo de configuração espacial da cadeia polipeptídica (ligação química entre moléculas), do tipo de aminoácidos que possui e do tamanho da cadeia: Estrutura Primária (nível estrutural mais simples e mais importante), Estrutura Secundária (nível de organização das proteínas fibrosas, mais simples estruturalmente), Estrutura Terciária (confere a atividade biológica às proteínas) e Estrutura Quaternária (guiadas e estabilizadas pelas mesmas interações da terciária) [2].

Sabendo a forma de uma molécula é possível entender como ela atua. Vários campos de pesquisa na área de biologia e medicina têm uma necessidade muito grande de uma ferramenta que diminua o esforço de se visualizar a estrutura de grandes moléculas biológicas, incluindo proteínas e ácidos nucléicos. Essas “moléculas da vida” são encontradas em vários organismos, incluindo bactérias, fungos, plantas, ratos, moscas e humanos (saudáveis ou não).

Hoje, utilizando-se de ferramentas e recursos computacionais de fácil aquisição é possível criar

um programa capaz de interpretar informações sobre estruturas tridimensionais dessas moléculas e gerar sua visualização.

Este trabalho trata-se de um programa desenvolvido capaz de visualizar tridimensionalmente e dinamicamente a evolução temporal de proteínas, assim como sua ligação entre carbonos-α, que forma seu "esqueleto" (ou "*backbone*"). Espera-se dessa forma, ajudar a melhorar a compreensão dos complexos mecanismos envolvidos nas interações proteínas-proteínas ou proteínas-substrato, seja executando suas funções fisiológicas ou para explorar as possíveis interações bioquímicas, contribuindo assim ao conjunto de ferramentas hoje em dia usado não só no desenvolvimento de novos fármacos, como também na investigação básica de certas patologias a nível molecular.

Materiais e Métodos

Codificado na linguagem C++, o programa faz a leitura de um arquivo de formato PDB, cujas informações atômicas extraídas, tipo e coordenadas, são usadas para sintetizar a visualização 3D da proteína.

O arquivo PDB segue um padrão universal e neles são listados todos os tipos e coordenadas dos átomos presentes na proteína, juntamente com algumas informações adicionais referentes ao experimento usado na resolução da estrutura. O banco de dados atual de estruturas de proteínas conta com cerca de 38 mil estruturas de proteínas já resolvidas e todas elas possuem um arquivo no formato PDB disponível no endereço eletrônico http://www.rcsb.org/pdb.

No módulo aqui apresentado, os procedimentos de leitura dos arquivos são feitos através de métodos presentes na biblioteca *fstream.h*, tais como: `ifstream` e `getline`. O primeiro funciona como declaração de uma variável para leitura de um arquivo. Por exemplo, com o comando:

```
ifstream fout;
```

seria declarado uma variável chamada *fout*, através da qual poderia ser acessado um arquivo e feitas diversas operações, como abrir, ler uma linha, fechar, etc.

A principal função de leitura utilizada neste trabalho é a `getline` mencionada anteriormente e que tem a finalidade de ler a próxima linha de um arquivo texto e armazená-la em uma variável do tipo *String* (conjunto de caracteres) se necessário.

Lembrando-se que antes de efetuar operações como a de ler uma linha, deve-se abrir o arquivo com que se vai trabalhar. Por exemplo:

```
fout.open("exemplo.pdb");
```

abriria o arquivo chamado *exemplo.pdb*. Uma vez terminado o uso do arquivo, é importante fechá-lo, para não haver desperdício de memória:

```
fout.close();
```

Tendo acesso aos dados dos arquivos *pdb*, podem-se extrair as informações necessárias e, como esses dados seguem uma formatação padrão, é possível saber a posição desses valores em cada linha.

As informações extraídas correspondem ao tipo do átomo e a sua posição espacial dentro da proteína. Estes dados são armazenados em vetores de variáveis, sendo que sua leitura é feita apenas uma vez ao abrir o arquivo e depois os valores podem ser acessados livremente quando necessário.

Com os dados da proteína já disponíveis, é possível reconstruí-la tridimensionalmente. Para isso foram usadas as bibliotecas do OpenGl.

O OpenGL é definido como "uma interface em software para hardware gráfico". Em essência, é uma biblioteca de gráficos tridimensionais e modelagem, portátil e rápida, o que permite gerar imagens interativas em tempo-real. O OpenGL não é uma linguagem, como C ou C++, mas sim uma biblioteca de funções pré-programadas. Na realidade não existe um "programa em OpenGL" mas sim um programa que o desenvolvedor escreve usando o OpenGL como um de seus APIs [3].

Na visualização os átomos são representados por esferas sendo que o tamanho (raios de van der Walls) e cor dependem do seu tipo (N , C, CA, O e H). Para desenhar as esferas deve-se primeiro posicionar a cena para o local a ser desenhada (obtido através das variáveis citadas acima). Isto é feito com o comando:

```
void glTranslatef(GLfloat x,
                  GLfloat y,
                  GLfloat z)
```

onde *x*, *y* e *z* correspondem as coordenadas da nova posição. Uma vez posicionada a cena, pode-se desenhar a esfera com o comando:

```
void gluSphere(GLUquadricObj *qobj,
               GLdouble radius,
               GLint slices,
               GLint stacks)
```

onde *qobj* se refere ao objeto quadrático a ser utilizado para desenhar a esfera, *radius* é o tamanho do raio da esfera e *slices* e *stacks* são os números de cortes verticais e horizontais

(quanto maior os valores, mais suave a superfície da esfera fica, requerendo assim mais processamento).

Ainda é necessário fazer a simulação de movimento da proteína. Para isto, as posições dos átomos são modificadas em determinado intervalo de tempo, sendo substituídas pelos valores correspondentes ao modelo seguinte (lembrando que todas as posições de cada modelo foram armazenadas inicialmente).

Para evitar mudanças bruscas, foi feita uma interpolação de movimento entre uma posição e a seguinte. É calculada a diferença desses valores e dividido pelo número de quadros entre eles. Com isto, obtém-se o quanto cada átomo se deslocará nos quadros intermediários. O resultado é um movimento contínuo e suave.

Outra possibilidade é a visualização no formato tipo *backbone*, que são linhas desenhadas conectando carbonos-α consecutivos de toda a cadeia principal. Os comandos para desenhar uma linha são:

```
glBegin(GL_LINES);
glVertex3f(px1,py1,pz1);
glVertex3f(px2,py2,pz2);
glEnd();
```

onde o primeiro comando indica que a cada coordenada consecutiva fornecida, haverá uma conexão por linhas. O segundo e terceiro comandos indicam as coordenadas dos pontos e o quarto comando finaliza a leitura de vértices.

Além dos mencionados, existem vários outros métodos e funções do OpenGl cujas finalidades variam desde a construção de uma janela até a interpretação de entradas do usuário.

Resultados e Discussão

Foi possível desenvolver um programa que recebe como entrada um arquivo em formato *PDB* capaz de visualizar a estrutura de uma proteína tridimensionalmente, permitindo verificar a partir de trajetórias de simulações seu comportamento. As Figuras 1 e 2 mostram, respectivamente, a tela inicial e a tela de visualização do módulo desenvolvido.

Figura 1 – Tela inicial do programa, onde é permitida a seleção do arquivo PDB e, em seguida, a visualização 3D da proteína é gerada.

A visualização imediata do arquivo PDB com a trajetória da molécula permite avaliar o comportamento da proteína de maneira simples e com a opção de visualização da estrutura como *backbone* ou *spacefill*. Ainda, além da rotação da molécula, o algoritmo de interpolação implementado oferece uma transição entre um frame a outro de forma suave sem comprometer consideravelmente a eficiência na amostragem da trajetória.

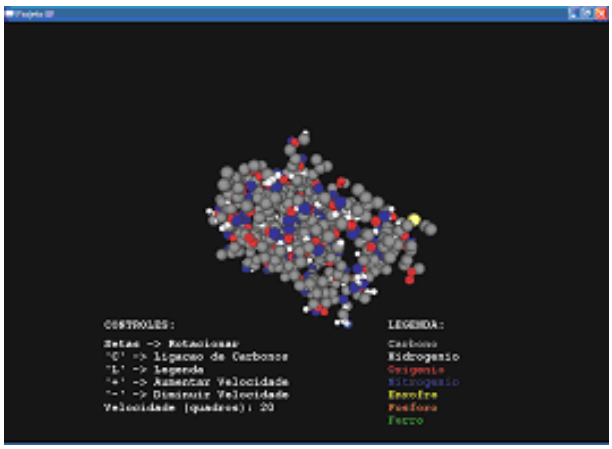

Figura 2 – Tela de visualização da proteína, do menu indicando os controles disponíveis e a legenda dos elementos que compõe a proteína.

Conclusão e Pretensões

Considerando o desenvolvido e analisando os dados obtidos a partir do mesmo, conclui-se que os requisitos funcionais iniciais foram plenamente atingidos com sucesso. As técnicas de programação foram aprimoradas, possibilitando um resultado final superior às expectativas iniciais do projeto.

Finalmente, vale ainda destacar que embora o objetivo principal deste projeto tenha sido alcançado, o módulo atual pode ainda ser aperfeiçoado se considerado, por exemplo, outros tipos de átomos, identificação de tipos de estruturas secundárias, determinação da área exposta da proteína e sua visualização, translação da proteína, etc.

Referências

[1] Proteína. Disponível em: http://www.babylon.com/definition/prote%C3%ADna/Portuguese?uil=English&uris=!!9G2CGKRAUE Acesso em: 15.jun.2006.

[2] Proteína. Disponível em: http://pt.wikipedia.org/wiki/Prote%C3%ADna Acesso em: 16.jun.2006.

[3] Daniel M. Gonzalez Clua, Valdemir Carrara; *Reconstrução Gráfica Tridimensional de Edificações Urbanas a Partir de Imagens Aéreas*. In: Seminário de Iniciação Científica do INPE, 2006, São José dos Campos, SP. Anais... São José dos Campos: INPE, 2006