Relatório do Projeto em WebGL da disciplina de Computação Visual

Molecule Viewer

Trabalho Realizado por:

- *Francisco Oliveira NºMec 80108*
- *Diego Hernandez NºMec 77013*

***Resumo* – Molecule Viewer é uma aplicação Web baseada na visualização em espaço tridimensional de Moléculas, composta com os seus respetivos átomos e ligações. A aplicação permite a visualização de moléculas já pré-definidas e também dá a possibilidade de o utilizador definir moléculas a partir de ficheiros, com um formato mais explicado abaixo.**

***Abstract* – Molecule viewer is an Web based application that provides a visualization in a tridimensional space of molecules, composed of their respective atoms e connections. The application allows for the visualization of already pre-defined molecules and also provides the possibility for the user to define molecules through a file, with a format specified below.**

I. Introdução

O projeto em que este relatório se baseia é referente ao primeiro projeto da Unidade Curricular Computação Visual do Curso Mestrado Integrado de Computadores e Telemáticas.

A finalidade deste projeto é a abordagem do aprendido, praticado e elaborado durante as aulas teóricas e práticas da unidade curricular relativamente a transformação e visualização 3D, utilizando WebGL como plataforma principal para a visualização e manipulação de objetos 3D e as suas propriedades.

O tema escolhido para este projeto foi a visualização de modelos de moléculas. Esta escolha foi feita não só pelo interesse dentro da área em que se baseia como também a grande utilidade que uma ferramenta do tipo tem para científicos e sujeitos interessados na área da química.

Esta plataforma permite escolher moléculas predefinidas, submeter um ficheiro com um determinado formato, mais em frente especificado, para a visualização da molécula elaborada, permite também a visualização de mais do que uma molécula ao mesmo tempo para fins de comparação e a observação mais específica do criado através do manuseamento do ângulo e posição dos objetos, neste caso, moléculas.

II. Manual Utilizador

No acesso a plataforma Web é apresentado um canvas WebGL cinzento vazio e uma série de botões com funcionalidade específicas.

A ordem pela qual cada botão irá ser abordado será pela ordem inversa, isto é, iremos falar dos últimos botões até aos primeiros.

Na plataforma Web é observável na secção de Examples, 4 imagens de modelos de Moléculas, esta é, respectivamente a sequência: Ethane (Etano), Butane (Butano), Methane (Metano) e Propane (Propano). Se o utilizador clicar numa das imagens, no canvas irá aparecer o modelo 3D da molécula escolhida pelo utilizador. Se o utilizador desejar outra molécula, basta escolher novamente uma das moléculas exemplo. Outra forma de desenhar moléculas é carregando um ficheiro com o formato pretendido (explicado mais abaixo).

O Utilizador pode movimentar, para fins de observação, as moléculas apresentadas no canvas WebGL. Para tal este pode ou selecionar, seguidamente se o desejar, os botões “Move Left” (Mexer para à esquerda), “Move Right” (Mexer à Direita), “Move Up” (Mover à Cima), “Move Down” (Mexer Abaixo). O cliente tem a capacidade de movimentar a(s) molécula(s) com as teclas ‘W’ (movimenta molécula de baixo para cima),’’D’(movimenta molécula da esquerda para à direita), ‘S’, (movimenta molécula de cima para abaixo)e ‘A’ (para movimentar a molécula da direita para a esquerda). Existem dois métodos de movimento: o que está por defeito em que a origem do eixo desloca-se com as moléculas, e o alternativo em que a origem fica sempre no centro do canvas. Estes métodos podem ser alternados através do botão “Change Moving Method”.

O utilizador pode também alterar os ângulos da molécula relativamente a um ponto origem. Isto é possível através da seleção com o botão direito do rato no canvas WebGL. Para movimentar as moléculas no sentido positivo do eixo Y o cursor tem de se movimentar para a esquerda, caso contrário o rato tem de se mexer à direita. De modo a movimentar as moléculas no sentido positivo do eixo X o cursor do rato terá de se movimentar de baixo para cima, caso contrário de cima para baixo.

O utilizador pode visualizar, das moléculas apresentadas no canvas WebGL, apenas os átomos sem as suas respetivas ligações, ou com a ligação com os átomos que lhe correspondem.

Pode também visualizar as figuras representadas no canvas em projeção Ortogonal e em Perspetiva, que se encontra inicialmente por defeito.

Finalmente, as escalas de cada molécula representada no canvas pode aumentar com o scroll up do rato, simulando o *zoom in* na molécula, como também é possível realizar scroll down para diminuir a escala da molécula, simulando um *zoom out*.

III. Desenvolvimento

De forma a estruturar melhor as moléculas criadas e representadas no canvas, como também o fácil manuseamento dos átomos e as ligações dum objeto molécula, foram criados 3 classes no ficheiro classes.js.

Classe Atom: Um objeto do tipo Atom apresenta informação relativamente às coordenadas de posição (x,y,z) desta, o elemento químico do átomo e um id único. Toda esta informação é possível ser obtida através dos métodos do tipo "get". Estes métodos auto-explicativos são "getVector()", "getAtom()" e "getId()".

Classe Connection: Um objeto do tipo "connection" tem informação relativa a posição dos dois átomos que este objeto liga, a distância absoluta entre estes, como também os ângulos de rotação para X, Y e Z. Finalmente ligações podem ser do tipo simples (um cilindro), duplas (dois cilindros) e triplas (3 cilindros).
Tal como a classe átomo, a classe Connections é possuidora de métodos do tipo get, que também são auto-explicativos. Estes são: getAngleZX(), getAngleXY(), getAngleZY(), getAtom1(), getAtom2(), getType(). De salientar que também existe o método getCenter() para determinar as coordenadas do centro de massa do cilindro através da seguinte fórmula:

$$((P2X + P1X)/2, (P2Y + P1Y)/2, (P2Z + P1Z)/2, 0)$$

.

Além disso possui também o método calculateDistance() para determinar a distância absoluta dos dois átomos que formam a ligação única, para a posterior construção do cilindro que representa esta, através da seguinte fórmula matemática:

$$\sqrt{(P2X - P1X)^2 + (P2Y - P1Y)^2 + (P2Z - P1Z)^2}$$

Classe Molecule: Sempre que é criado uma molécula é decorrida a classe Molecule. Um objeto Molecule tem: uma lista de todos os objetos ligações que lhe pertence, uma lista de todos os objetos átomos que a este lhe pertence e um id único. Esta classe é composta por métodos do tipo get auto-explicativos: getAtom(id), getConnection(id). O método setAtom(atom,position) permite associar a criação dum átomo de um tipo à molécula com as suas respetivas coordenadas. E finalmente é possível criar um tipo de ligação entre dois átomos pertencentes à molécula através do método setConnection(atom1Id, atom2Id, type).

IV. Modelos

Cada modelo ilustrado no canvas (seja uma esfera ou um cilindro) tem diversas propriedades:

- **Uma lista de vertices (vertices):** representa um array (do tipo matriz) 4 colunas, que representam, respectivamente, a coordenada x,y,z;
- **Uma lista com vetores normais:** têm os vetores normais de todos os vértices do modelo.
- **Variáveis tx, ty e tz:** representa a deslocação (vetor) dum determinado vertice definido;
- **Variáveis txMult:** de modo a definir inicialmente a distância entre ligações/cilindros do tipo 2 (duplos) e 3 (triplos)
- **originRotXX,originRotYY,originRotZZ:** variáveis necessárias para definir inicialmente os ângulos dos cilindros em relação aos eixo de x,y e z.
- **rotAngleXX, rotAngleYY, rotAngleZZ:** estas variáveis são utilizadas para a rotação interativa das moléculas. De notar que a rotação da molécula em

si não é possível com os ângulos originRotXX, originRotYY, originRotZZ, uma vez que estas só são utilizadas inicialmente para a correta visualização dos cilindros que ligam dois tipos de moleculas

- **Sx,sy,sz:** Estas variáveis representam os fatores de escalas iniciais de cada modelo representado no canvas.
- **Sxzoom, syzoom, szzoom:** representam os fatores de escalas de cada molécula em si. Para o correto aumento da escala da molécula foi necessário criar outro tipo de variáveis de escala para cada eixo.
- **rotXXOn, rotYYOn, rotZZOn:** Atributos booleanos que servem para decidir se o modelo sofre a rotação sobre um certo eixo ou não.
- **rotXXSpeed, rotYYSpeed, rotZZSpeed:** Atributos que servem para determinar a velocidade de rotação dos modelos.
- **rotXXDir, rotYYDir, rotZZDir:** Atributos que servem para determinar a direção de rotação dos modelos.
- **kAmbi, kDiff e kSpec:** Lista com os valores de iluminação ambiental, reflexão difusa e reflexão especular para cada valor de RGB.
- **cilinder:** Atributo para determinar se o modelo é um cilindro ou não (está a 1 quando é um cilindro e a 0 quando não é).

V. Atomos

Cada átomo é representado no canvas como uma esfera. Para a criação é criado inicialmente um cubo. Este irá passar por um processo recursivo de Mesh Refinement em que os triângulos que formam o cubo são divididos em mais triângulos. As nossas esferas passam por uma subdivisão de profundidade equivalente a 4, isto é os triângulos que formam o cubo são subdivididos em 4 triângulos, e cada um desses sub triângulos são divididos também 4 vezes em modo recursivo.

De seguida, os vértices de cada um destes sub triângulos gerados passam por um processo de normalização, causando que todos os vértices fiquem à mesma distância da origem, o que torna o cubo numa aproximação de esfera.

Não são atribuídos 'cor' a cada um dos vértices do triângulo que forma a esfera. Os valores de iluminação ambiental, reflexão difusa e reflexão especular são atribuídos de acordo com tipo de átomo gerado. É também de referir que o raio de cada esfera varia com o tipo de átomo.

VI. Ligações

.Cada ligação é representado no canvas como um cilindro. Para a criação do cilindro é criado um cubo ao qual se aplica o processo recursivo de Mesh Refinement com profundidade 4. De seguida aplica-se uma transformação aos vértices do modelo que o torna num cilindro. Este processo consistem em, para cada vértice do modelo, normalizar as coordenadas X e Y mas não a Z. O que isto faz é que todos os vértices do modelo fiquem à mesma distância do eixo ZZ, o que transforma o cubo num cilindro. Este processo é efetuado pela função "normalizeCylider()". De seguida são calculados os vetores normais para cada vértice.

VII. Editor de Moléculas

Uma das *features* principais pensado para este trabalho foi a criação de moléculas a partir da leitura dum ficheiro com determinado formato. A estrutura deste ficheiro consiste no seguinte:

Inicialmente é necessário recorrer a definição de cada átomo. Cada linha do ficheiro irá precisar de 4 elementos.

- O primeiro elemento corresponde ao tipo de átomo a desenhar, de modo a aplicação computar a intensidade da luz (ou visualmente a gama de cor da esfera) e o raio da esfera a criar.
- O segundo, terceiro e quarto elementos correspondem, respetivamente, às coordenadas do centro da esfera num espaço tridimensional XYZ

Para que um agregado de átomos se transforme em uma molécula é necessário que os átomos se liguem uns aos outros. Para isso é necessário definir no ficheiro as ligações destes átomos. De referir que de modo a que a aplicação distinga as linhas para a criação de átomos com as linhas de criação de ligações é necessário tipificar o caráter '\' e de seguida escrever as linhas que definem as ligações.

A estrutura da linha de ligação consiste no seguinte:

- O primeiro elemento e o segundo representam o índice da linha do átomo gerado, isto para

determinar os extremos do cilindro que representa a ligação e consequentemente o determinio de distância, posição do centro do cilindro e a rotação segundo os seus eixos para a correta ligação entre estes.

- O terceiro elemento representa o tipo de ligação, sendo: 1 no caso da ligação entre os átomos ser simples, 2 no caso de ser dupla e 3 se for tripla. De notar que o tipo de ligação pode ser entre [1,3].

VIII. Desafios

Posicionamento e rotação correta dos cilindros - Como as moléculas são geradas com informação apenas sobre o centro dos átomos e que átomos estão conectados, a determinação do ângulo correto dos cilindros para a representação correta da molécula são da responsabilidade da aplicação. Para solucionar este problema, na nossa primeira abordagem foi decidido que íamos determinar o ângulo de rotação necessário em relação a cada um dos eixos a partir das coordenadas dos centros de cada um dos átomos. Para isso usámos a função getAngle() que aceita como argumentos dois pontos e devolve o ângulo entre eles usando a fórmula :

$$\arctan((P2y - P1y)/(P2x - P1x)) * 180/\pi$$

Esta função aceita pontos de apenas duas dimensões, pois foi assumido que, por exemplo para calcular o ângulo de rotação em relação ao eixo de YY, o ponto está no plano de XOZ e por isso só passamos as coordenadas dos dois pontos em X e em Z. Esta abordagem não funcionou porque não havia uma ordem geral de aplicação das matrizes de rotação em que para qualquer combinação de ângulos a posição final dos cilindros estava correta. Numa segunda abordagem então decidimos fazer a rotação dos cilindros através apenas de dois ângulos de rotação, um em relação ao eixo de XX e outro em relação ao eixo de YY. Para calcular o ângulo em relação ao eixo de YY continuámos a usar a função getAngle(), passando apenas as coordenadas em X e em Z. No entanto para calcular o ângulo de rotação em relação ao eixo de XX foi usado outra função chamada getAngleOrigin() que recebe como argumentos dois pontos tridimensionais, que são os centros de cada esfera, e calcula o ângulo de rotação em relação ao eixo de XX fazendo uma rotação dos dois pontos para ficarem sobre o eixo de XX e depois, usando as coordenadas de X e Y de cada ponto na função getAngle(), determinar o ângulo correto de rotação. Com estes ângulos descobertos e aplicando primeiro a rotação em relação a XX e depois a rotação em relação a YY os cilindros ficam sempre na posição correta.

IX. Conclusão

Ficámos satisfeitos com o resultado final deste projeto, conseguindo ter o acordado para este, abordando maior parte da matéria da unidade curricular Computação Visual e as estratégias leccionadas nas aulas práticas para a visualização e transformação de objetos em espaço tridimensional, como também ter tido a capacidade de ultrapassar os desafios que o Molecule Viewer nos apresentou.

X. Notas

- Existe no diretório do trabalho uma pasta chamada “Molecules” onde existem já vários exemplos de moléculas já definidas em ficheiros.
- A transformação dos objetos “Molecules” para cilindros e esferas é feita pela função “moleculesToModels()” no ficheiro “sceneModels.js”.
- A distribuição de esforço relativo do trabalho foi igual para os dois membros do grupo, ficando cada um com 50%.