Departamento de Informática Mestrado Integrado em Engenharia Informática

Índice

[Introdução 3](#_Toc449914287)

[VBOs 3](#_Toc449914288)

[Superfícies de Bézier 4](#_Toc449914289)

[Superfícies de Catmull-Rom 4](#_Toc449914290)

[Atualização do Engine e Gerador 5](#_Toc449914291)

[Gerador 5](#_Toc449914292)

[Engine 5](#_Toc449914293)

[Resultados 8](#_Toc449914294)

[Teapot com 1, 5, 10, 20 de tesselation 8](#_Toc449914295)

[Ficheiro XML 9](#_Toc449914296)

[Sistema solar dinâmico 9](#_Toc449914297)

[Conclusão e trabalho futuro 10](#_Toc449914298)

## Introdução

Nesta terceira fase do projeto proposto no âmbito da unidade curricular de Computação Gráfica damos seguimento ao trabalho desenvolvido na primeira e segunda fase. Nesta fase pretende-se estender a aplicação geradora de primitivas para que seja possível criar listas de pontos correspondentes a superfícies de Bézier. A aplicação receberá como parâmetros um ficheiro (*patch*) onde se encontram definidos os vértices relativos aos pontos de controlo e o grau de tesselação pretendido. Tal como para as outras primitivas, o resultado será um ficheiro com uma lista de pontos.

No que diz respeito ao Engine as translações passam a ser definidas através da inclusão das curvas e de uma noção de tempo. Mais especificamente, uma translação é definida como um conjunto de pontos de controlo de uma curva e um tempo (em segundos, por exemplo) necessário para percorrer toda essa curva. Por isso, a animação passa a ser cíclica, isto é, quando a curva terminar regressa-se ao início da mesma, e com isto, o motor 3D passa a dispor de um relógio, ou contador de tempo (por exemplo: glutGet(GLUT\_ELAPSED\_TIME)).

Ainda sobre o motor 3D, as rotações também têm agora associado um tempo que surge no mesmo contexto que o ângulo na fase anterior.

Para além destas alterações, em vez do modo imediato, o motor 3D desenha as primitivas a partir de VBOs (*Vertex Buffer Object*).

## VBOs

Ao contrário do modo imediato usado na fase anterior para desenhar as primitivas, as VBOs (Vertex Buffer Object) proporcionam métodos de encaminhamento de dados (como por exemplo, posições, cores, etc.) para a placa de vídeo sem que estes sejam imediatamente desenhados. Desta forma é possível obter ganhos substanciais de *performance* pois os dados já estão na placa de vídeo em vez de serem alocados na memória do sistema, permitindo assim, que sejam desenhados diretamente a partir da placa.

## Superfícies de Bézier

As superfícies de Bézier são superfícies expressas como a interpolação linear entre pontos de controlo. A principal característica destas superfícies de Bézier em geral não passa pelos pontos de controlo centrais, mas é esticada na direção de cada um deles como se cada ponto de controlo fosse um ponto de atração.

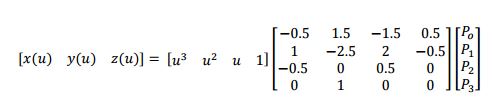
Neste sentido e no contexto do trabalho prático, a inicialização das superfícies de Bézier tem como argumentos o ficheiro *patch*, o grau de tesselação pretendido e o ficheiro de resultado. Por sua vez, é chamada uma função relativa às curvas que calcula todos os pontos pertencentes à superfície e simultaneamente os escrevendo-os em ficheiro.

Exemplo de teapots com vários graus de tesselação na secção de resultados.

## Superfícies de Catmull-Rom

As órbitas são definidas por superfícies de Catmull-Rom que em contraste com as superfícies de Bézier, os pontos de controlo fazem parte desta. Ou seja, em vez de serem apenas pontos de atração, os pontos de controlo estão contidos na própria superfície. Desta forma, as orbitas tenderam a ficar mais suaves e mais próximas daquilo que são na realidade.

Em termos matemáticos a expressão que possibilita esta funcionalidade é:



## Atualização do Engine e Gerador

### Gerador

A alteração no gerador fez com que este seja capaz de gerar, para além das primitivas abordadas, curvas e superfícies de Bézier. Para tal, criou-se novas funções que leem o patch e calculam os pontos das curvas e superfícies de Bézier.

### Engine

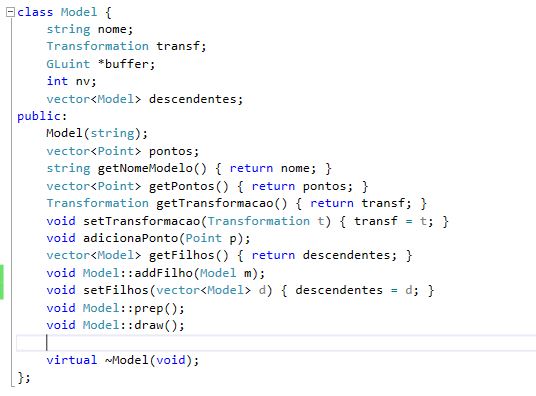
No Engine foram alteradas as classes que dizem respeito às transformações dos modelos.

Devido a classe Type já não ser capaz de lidar com as alterações ao xml, foi dividido em Scalate(lida com as alterações a escala), Rotate (lida com o tempo e rotação) , e a Translate (lida com as orbitas dos modelos).

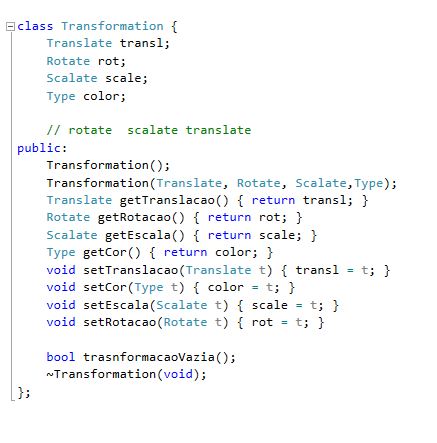
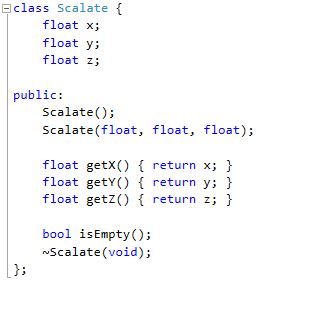
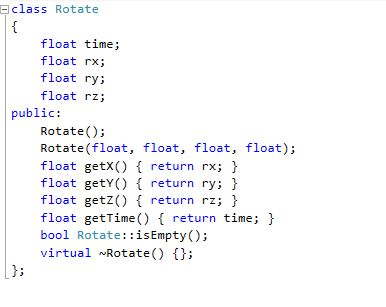
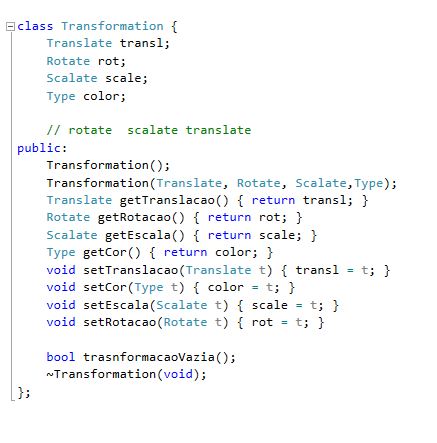
Também foram feitas alterações ao desenho dos modelos, passando de desenho imediato para invocações do desenho dos VBO.

A classe Model também foi alterado para tornar mais fácil desenhar com VBO e acomodar uma melhor transferência de Translate de pai para filho.

#### Model

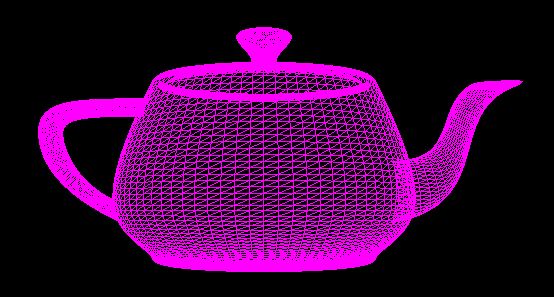
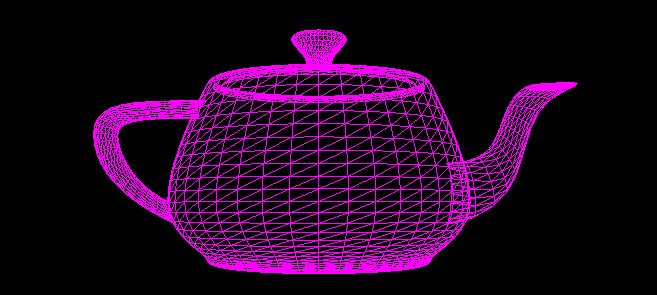
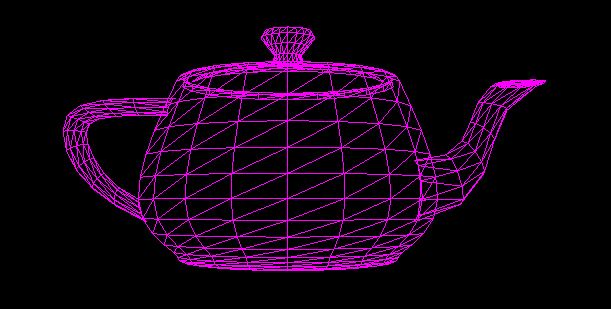
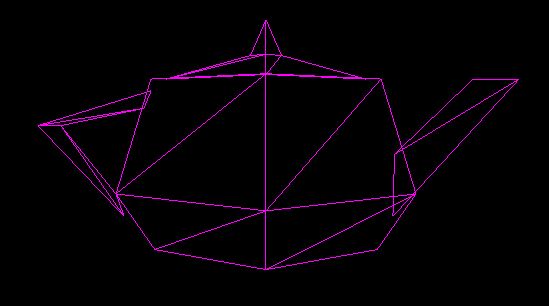


#### Classes das Transformações

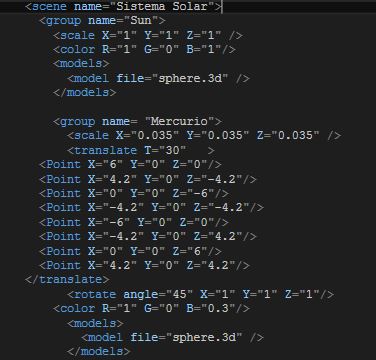


## Resultados

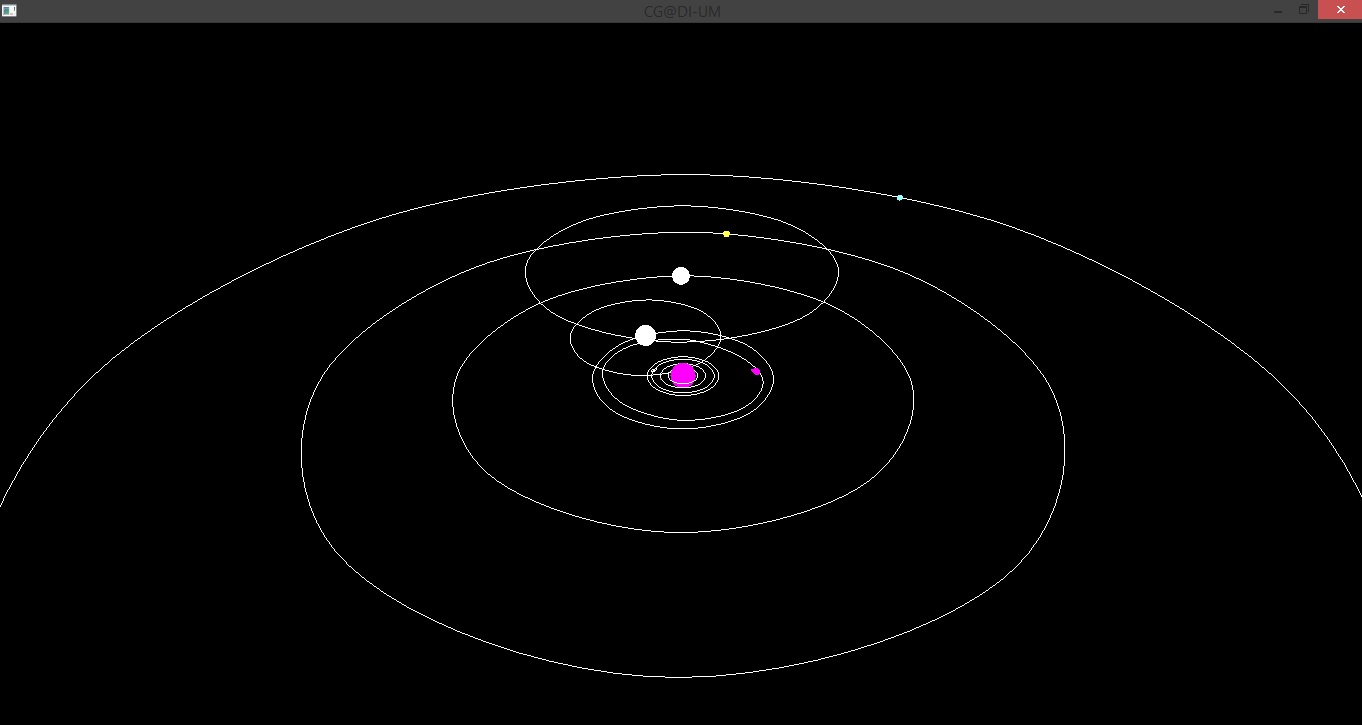
### Teapot com 1, 5, 10, 20 de tesselation



### Ficheiro XML



### Sistema solar dinâmico



## Conclusão e trabalho futuro

No final desta terceira fase do trabalho prático, consideramos que quase todos os objetivos propostos foram atingidos com sucesso. Como é bem explícito no enunciado, foram feitas alterações na leitura do ficheiro XML bem como nas transformações geométricas. Quer as translações quer as rotações passaram a ser definidas tendo em conta um tempo responsável pelos períodos de translação e rotação, respetivamente. As translações possuem ainda mais uma particularidade, que é a das curvas: a translação ocorre sobre uma curva definida à custa de pontos de controlo.

Relativamente ao sistema solar, houve um progresso bastante significativo desde a última fase. Não só passou de estático a dinâmico como também o modo de desenho é muito mais eficiente permitindo desta forma uma melhor *performance*, graças aos VBOs. A questão mais preocupante nesta fase, foi mesmo a das curvas e das superfícies, visto que mesmo nas aulas práticas já tinham surgido bastantes dúvidas e dificuldades nesta área.

Em suma, estamos satisfeitos com o trabalho desenvolvido uma vez que já é possível observar algo que já era esperado há algum tempo