**Avaliação individual após Trabalho T2**

**Laboratório de Aplicações com Interface Gráfica  
MIEIC - 2013/2014**

**Avaliação individual após Trabalho T2 (duração: 2:30 horas + 0:30 de tolerância)**

O presente enunciado diz respeito a uma prova de avaliação prática da unidade curricular LAIG e contém um conjunto de três desenvolvimentos a efetuar sobre a base de trabalho desenvolvida no trabalho T2 das aulas práticas. Os computadores das salas reservadas para a prova encontram-se privados de rede. O único acesso permitido é ao software "SIGEX" para efeitos de upload dos trabalhos realizados.

**Instruções para Upload**

Siga cuidadosamente as seguintes instruções:

**Entrada/registo no sistema SIGEX:**

Use o "Código Público do Exame", de acordo com a sala e sistema operativo seguintes:

* B104 (Windows): VOA082
* B208 (Windows): WBS273
* B208 (Linux): OCW303
* B213 (Windows): YNM374

**Durante a realização da prova:**

O código que foi entregue aos docentes em resultado do trabalho **T2** da unidade curricular deve ser usado como base para a resolução **de cada um** dos três problemas enunciados (não resolver um problema proposto sobre a solução de problema anterior...).  
Para cada problema, os seguintes ficheiros deverão ser arquivados num ficheiro .zip com o nome "prob1.zip"... "prob3.zip" (esta regra de nomeação deve ser ESTRITAMENTE seguida):

* **código-fonte**
* **ficheiro de projeto** (ex: em MS-VStudio: extensão ".vcxproj"; cuidado com as extensões escondidas) para posterior recompilação;
* **executável**gerado (extensão ".exe");
* todas as **texturas**utilizadas;
* se necessário, **ficheiros YAF** para demonstrar a resolução do problema em questão

**NOTA:** todos os ficheiros resultantes da compilação, à exceção dos anteriores, devem ser apagados.

Cada ficheiro "prob\*.zip" deverá ser sujeito a *upload*utilizando as facilidades do SIGEX.

**No final da prova:**

Deve ser criado um ficheiro de texto (**ident.txt**) com:

* Identificação do estudante,
* Situação da resolução **de cada um** dos três problemas: "prob. completo", "não considera o detalhe *xpto*", "não resolvido"...
* Algumas instruções julgadas pertinentes, nomeadamente como fornecer à aplicação o nome do ficheiro YAF a processar.

O ficheiro de texto deverá também ser sujeito a *upload*utilizando as facilidades do SIGEX.

**Notas:**

* Os identificadores finais dos arquivos serão construídos pelo sistema, acrescentando-lhes um prefixo igual ao identificador do aluno respetivo.
* O sistema permite eliminar ficheiros submetidos erradamente, e/ou submeter ficheiros com o mesmo nome, que serão sobrepostos aos existentes.

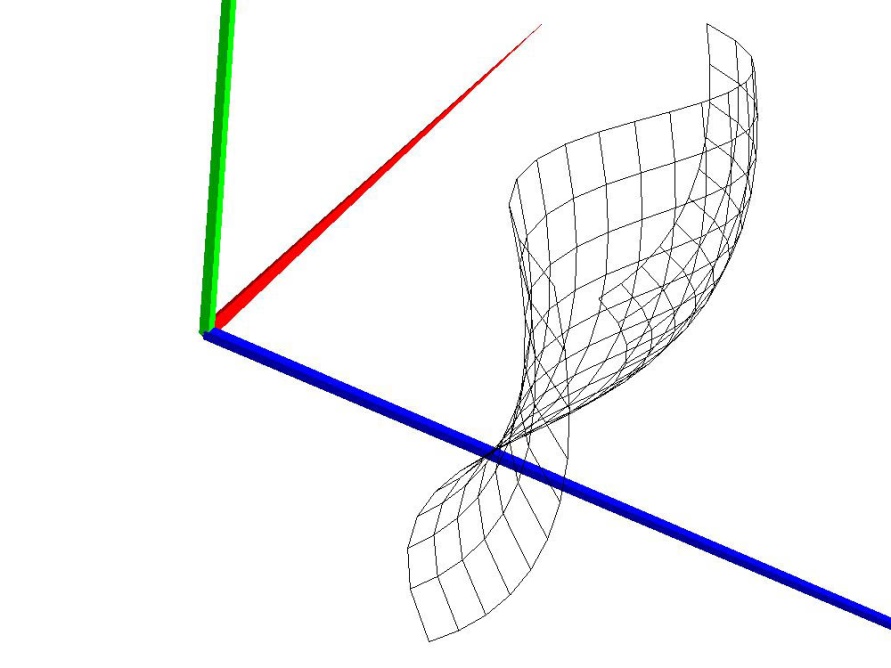
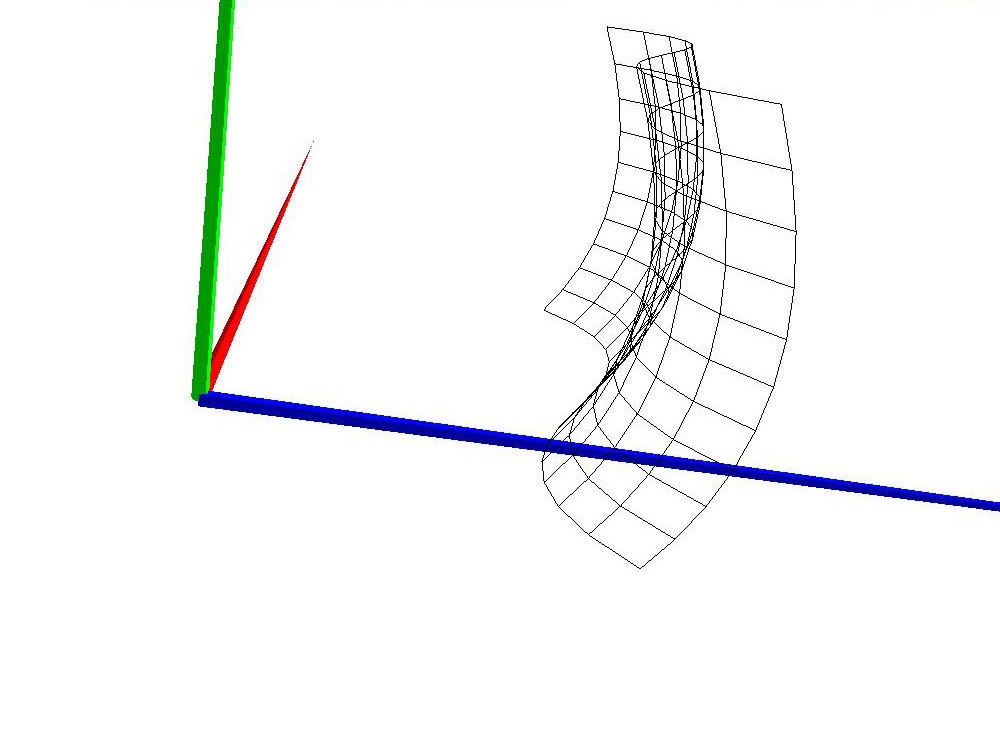
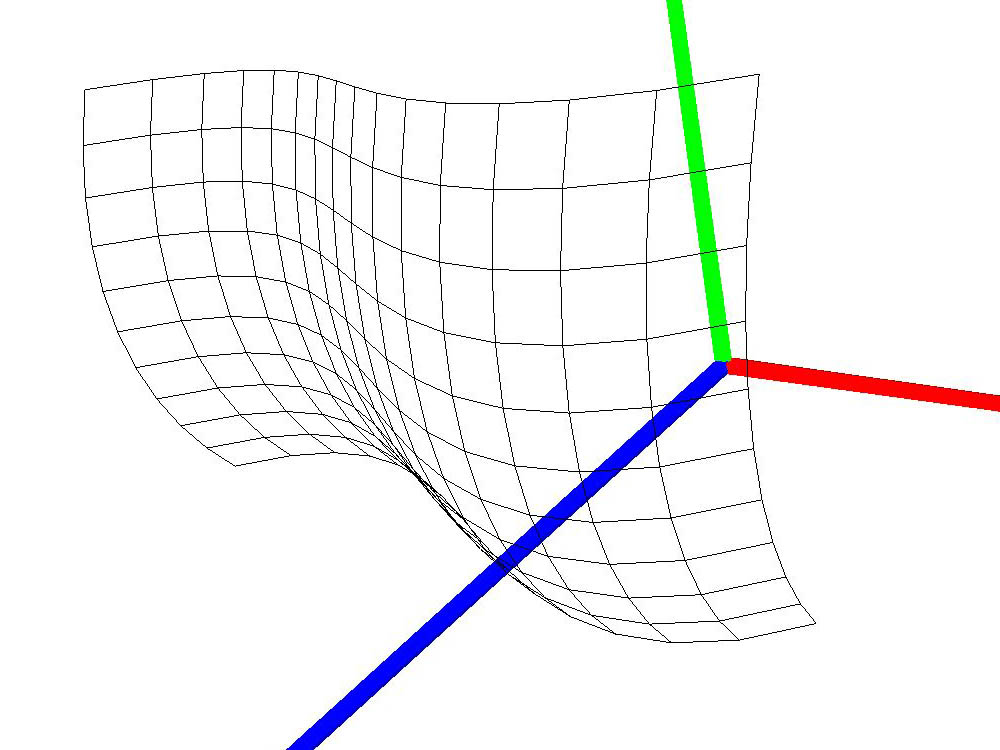
**Enunciado - Novas funcionalidades sobre o trabalho T2**

**1. Evaluators [8 valores]**

Crie uma "rede" usando *evaluators*desenhados em modo *wireframe*, de acordo com as seguintes regras:

* Numa das direções, a ordem deve ser tal que garanta uma curva com um máximo/mínimo (curva em *"U"*); na outra direção, a ordem deve ser tal que garanta uma curva com dois máximos/mínimos (curva em*"S"*).
* Em cada uma das direções, a ordem da curva respetiva deve ser a mínima que garanta a forma que é apresentada na figura 1.
* O *evaluator* deve apresentar 10 divisões na direção da curva em *"U"* e 15 divisões na direção da curva em *"S"*.
* Na linguagem .YAF, se utilizada, o *evaluator* deve ser declarado da seguinte forma:

<patch **orderU=”ii” orderV=”ii”** partsU=”ii” partsV=”ii” compute=”ss”>  
    <controlpoint x=”ff” y=”ff” z=”ff” />  
    …  
</patch>

**Fig. 1: O evaluator observado de três pontos de vista**

**2. Animação [8 valores]**

Desenvolva um processo de animação no qual:

* Um objeto, no caso uma pirâmide com bico para cima, se desloca segundo uma trajetória composta por uma sequência de segmentos de reta (à semelhança do que foi desenvolvido nas aulas).
* O objeto roda em torno de si próprio, segundo o seu eixo vertical, à medida que avança; prevê-se que a rotação total possa corresponder a um ângulo superior a 360º, ou seja, mais do que um volta completa.
* Na linguagem .YAF, se utilizada, a animação deve ser declarada da seguinte forma, em que "rot" é o ângulo total de rotação desde o início até ao final da animação:

<animation id=”ss” span=”ff” **rot="ff"** type=”linear”>  
   <controlpoint xx=”ff” yy=”ff” zz=”ff />  
   ...  
</animation>

**3. Shaders [4 valores]**

Altere os *shaders*utilizados para a superfície de água, de forma a adicionar uma "máscara" de areia. Deve adicionar suporte para uma terceira textura, "máscara", que terá uma parte com areia, e outra preenchida a preto  (ver fig. 2(a),  textura fornecida no SIGEX). Essa textura deverá ser usada nos *shaders*da seguinte forma:

* Nos pontos em que o valor da componente **vermelha**da máscara for inferior a **0.2** (zona preta), os *shaders*devem comportar-se como originalmente (superfície de água modificada, com textura e animada de acordo com o mapa de ondulação, textura de água e tempo, respetivamente).
* Nos restantes pontos, os *shaders*devem modificar a superfície para ter a altura (valor de YY) correspondente ao ponto médio dos limites teóricos de altura da ondulação (se não houver escalamentos, será 0.5), e colorir os fragmentos usando **as cores da máscara**.
* Na linguagem .YAF, se utilizada, o *shader* deve ser declarado da seguinte forma:

         <waterline heightmap=”ss” texturemap=”ss” **maskmap="ss"** fragmentshader=”ss” vertexshader =”ss”/>

A figura 2 (b), (c) mostra um exemplo do resultado pretendido.

(a)  (b)   
(c)

**Fig. 2: A textura de areia a usar como máscara (a); a superfície de água mascarada com areia: vista em perspetiva (b) e vista de lado (c)**