Botons i colors

En dissenyar una aplicació en la que volem posar un conjunt de botons amb funcionalitats similars, és important utilitzar:

○ El mateix color per a tots els botons sobre un color de fons uniforme, per evitar que uns botons destaquin respecte d'altres.

○ Colors pastel per als botons, sobre un fons saturat, per a que es distingeixin bé les vores dels botons.

○ Un color diferent per a cada botó, de manera que sigui més fàcil distingir-los.

○ Sempre una paleta de colors categòrica per assegurar-se que no induïm un ordre.

○ NO CONTESTO

Càmera-1

Tenim definida una càmera ortogonal amb paràmetres OBS=(0,0,-10); VRP=(0,0,0); up=(-1,0,0); window=(-10,10,-15,15); ZNear=5; i ZFar=10. Donada una escena, quin dels següents conjunts de paràmetres definirien una càmera amb la que segur que s'aconseguiria la mateixa imatge de l'escena?

○ OBS=(0,0,-6); VRP=(0,0,2); up=(-2,0,0); window=(-10,10,-15,15); ZNear=1; i ZFar=6.

○ OBS=(0,0,-9); VRP=(0,0,0); up=(-1,0,0); window=(-10,10,-15,15); ZNear=6; i ZFar=11.

○ OBS=(10,0,0); VRP=(0,0,0); up=(1,0,0); window=(-15,15,-10,10); ZNear=5; i ZFar=10.

○

OBS=(6,0,0); VRP=(1,0,0); up=(1,0,0); window=(-15,15,-10,10); ZNear=1; i ZFar=6.

○

NO CONTESTO

Càmera-2

Tenim definida una càmera ortogonal amb paràmetres OBS=(0,10,0); VRP=(0,0,0); up=(0,0,-1); window=(-5,5,-10,10); ZNear=5; i ZFar=10. Quins vèrtexs mínim i màxim en Sistema de Coordenades de l'Aplicació (SCA) defineixen el volum de visió d'aquesta càmera?

○

Punt mínim = (-5,0,-10); Punt màxim = (5,5,10).

○

Punt mínim = (-5,-2.5,-10); Punt màxim = (5,2.5,10).

○

Punt mínim = (-10,0,-5); Punt màxim = (10,5,5).

○

Punt mínim = (-10,-2.5,-5); Punt màxim = (10,2.5,5).

○

NO CONTESTO

Càmera-3

Volem pintar una esfera de radi 5 centrada al punt (7,5,7). Un estudiant defineix una càmera perspectiva amb paràmetres de posició i orientació OBS=(7,-5,7), VRP=(7,5,7), i up=(1,0,0). Quan pinta l'esfera en un viewport quadrat, li surt la imatge següent:

Quins paràmetres de l'òptica ha usat l'estudiant per obtenir aquesta imatge?

○

FOV=60; raw=2; ZNear=5; ZFar=15.

○

FOV=60; raw=0.5; ZNear=5; ZFar=15.

○

FOV=90; raw=0.5; ZNear=10; ZFar=20.

○

FOV=90; raw=2; ZNear=10; ZFar=20.

○

NO CONTESTO

Càmera-4

Tenim definida una càmera que permet veure *correctament* una escena des d'una posició arbitrària i sense retallar en un viewport quadrat de 600x600, es realitza un resize i el nou viewport és de 600x400. El codi d'un estudiant per tractar aquest redimensionament conté només la declaració del nou viewport (correcta). Quina repercussió tindran aquests canvis en la imatge final?

○

La imatge final queda deformada.

○

La imatge final queda retallada.

○

La imatge final es continua veient sense retallar ni deformar.

○

La imatge final queda retallada i deformada.

○

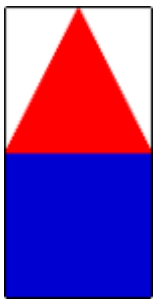
NO CONTESTO

Càmera-5

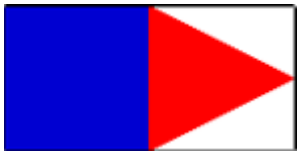
Tenim una piràmide de base quadrada de costat 5, amb la seva base centrada al punt (-2.5,0,0) i alçada de la piràmide 5 amb l'eix en direcció X-. A l'escena tenim també un cub de costat 5 centrat a l'origen. Si pintem aquesta escena amb una càmera ortogonal amb paràmetres OBS=(-2.5,10,0); VRP=(-2.5,5,0); up=(1,0,0); window=(-2.5,2.5,-5,5); ZNear=7; i ZFar=13, quina de les següents imatges

apareix en el viewport? Suposa el viewport definit adientment per a que no hi hagi deformació.

○

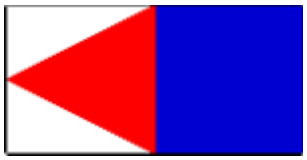


○



○

○

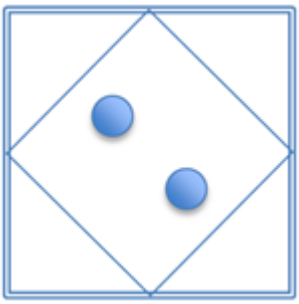


○

NO CONTESTO

Càmera-6

Tenim definida una escena formada per: un terra representat per un quadrat de 10x10 situat en el Pla Y=0, centrat en (0,0,0) i costats paral·lels als eixos de coordenades; i 2 arbres que tenen el centre de la base dels seus troncs en els punts (-2.5,0,0) i (2.5,0,0) respectivament. Indica quina de les següents inicialitzacions de la viewMatrix usant angles d'Euler permetria, suposant que l'òptica és correcta i VRP i distància (d) també, obtenir una visualització similar a l'esquematitzada en la figura:



VM = Translate(0,0,-d) * Rotate(-45,0,0,1) * Rotate(90,1,0,0) * Translate(-VRP);

VM = Translate(0,0,-d) * Rotate(45,0,0,1) * Rotate(90,1,0,0) * Translate(-VRP);

VM = Translate(0,0,-d) * Rotate(-45,0,0,1) * Rotate(90,0,1,0) * Translate(-VRP);

VM = Translate(0,0,-d) * Rotate(45,0,0,1) * Rotate(90,1,0,0) * Rotate(45,0,1,0) * Translate(-VRP);

NO CONTESTO

Color-1

Tenim un dibuix de color RGB=(0,1,1), què en podem dir respecte a la seva representació en HSB (o HSV)?

Cap de les altres respostes és correcta.

És un color amb saturació 1 i brillantor 0 (S=1 i B=0).

És un color poc saturat (S=0) i amb brillantor màxima (B=1).

És un color amb H=0, S=1 i B=1.

NO CONTESTO

Color-2

Un estudiant ha fet un dibuix pintat de color RGB=(0.5, 0.5, 0), i quan l'imprimeix en una impressora CMY en un paper de color RGB=(1, 1, 0.5), el dibuix imprès li surt de color RGB=(1, 0.5, 0.5). Quina de les següents raons que li pot passar a la impressora explicaria aquest comportament?

Li falten les tintes cian i groc.

Li falta la tinta cian (C).

Té totes les tintes carregades només al 50%.

Res, la culpa és del color del paper.

NO CONTESTO

Consistència

Quan parlem de consistència, quina de les següents afirmacions és **incorrecta**?

La inconsistència induïda no s'hauria d'utilitzar mai perquè incrementa els errors per part de l'usuari.

La consistència dins d'un conjunt de productes d'una mateixa empresa millora la usabilitat.

Hem de mantenir consistència també a nivell d'estructures invisibles.

La consistència ha de respectar les expectatives de l'usuari per millorar la usabilitat.

NO CONTESTO

Pragnanz

Els estudis demostren que percebem els objectes del nostre entorn com una composició de formes simples, encara que no ho siguin. Respecte a aquesta afirmació:

Això és el que enuncia la llei de Prägnanz, o llei de la bona figura.

L'afirmació és falsa, no hi ha estudis que demostrin això.

Precisament això és el que enuncia la llei de continuïtat, que ens permet percebre figures simples incompletes per a veure figures complexes.

L'afirmació parla de la llei de simetria que ens permet agrupar elements simples si són simètrics.

NO CONTESTO

PV-1

Respecte als processos del Procés de Visualització d'OpenGL, quina de les següents sentències és **FALSA?**

El procés de *Rasterització* projecta els punts en el pla ZNear i passa de coordenades d'observador (SCO) a coordenades de clipping (SCC).

El procés de Divisió de Perspectiva divideix per la quarta component del punt i converteix el punt de coordenades de clipping (SCC) a coordenades normalitzades (SCN).

○

El procés de *Transformació món-dispositiu* utilitza les dades del viewport per passar de coordenades normalitzades (SCN) a coordenades de dispositiu (SCD).

○

Cap de les altres respostes és correcta.

○

NO CONTESTO

PV-2

Indica quin és l'ordre correcte dels següents processos del Procés de Visualització:

1. Clipping
2. Transformació de coordenades de clipping (SCC) a coordenades normalitzades (SCN)
3. Transformació de coordenades de model (SCM) a coordenades d'aplicació (SCA)
4. Rasterització
5. Fragment Shader

○

3 – 1 – 2 – 4 – 5

○

3 – 2 – 4 – 1 – 5

○

3 – 1 – 4 – 5 – 1

○

3 – 2 – 1 – 5 – 4

○

NO CONTESTO

Shaders-1

Quan enviem a pintar un determinat triangle a OpenGL, dins del Vertex Shader:

○

Només podem accedir als atributs d’un únic vèrtex.

○

Podem accedir a les coordenades i normals dels tres vèrtex del triangle.

○

Depenent de les matrius view i proj el Vertex Shader decideix si pinta el triangle o no.

○

Cap de les altres respostes és correcta.

○

NO CONTESTO

Shaders-2

Quina de les següents afirmacions respecte al Vertex i al Fragment Shader és **FALSA**?

○

Si el Vertex Shader envia al Fragment Shader el color del vèrtex el Fragment Shader l’ha d’usar necessàriament per donar color al fragment.

○

Tant el Vertex com el Fragment Shader poden accedir als uniforms view i proj enviats des de l’aplicació amb les matrius de càmera.

○

Al Fragment Shader es pot descartar el fragment i no pintar-lo.

○

Al Fragment Shader no es poden modificar les coordenades del fragment.

○

NO CONTESTO

TG-1

Tenim una funció pinta_Patricio() que pinta un Patricio amb el centre de la base de la seva capsa contenidora a l'origen de coordenades, d'altura 2 i mirant cap a les Z positives. Volem usar aquesta funció per pintar 2 Patricios cara a cara a una distància entre ells de més de 2 unitats. Quin dels següents codis per calcular les TGs a aplicar als Patricios (TGPat1 i TGPat2) compliria aquestes condicions?

○

TGPat1 = Translate(-2,0,3) * Rotate_y(90);

TGPat2 = Translate(5,0,3) * Rotate_y(-90);

○

TGPat1 = Translate(-centreBasePatricio) * Rotate_y(90);

TGPat2 = Translate(10,0,10) * Rotate_y(180);

○

TGPat1 = Translate(-1,0,3) * Rotate_y(90) * Scale(2,2,2);

TGPat2 = Translate(1,0,3) * Rotate_y(-90);

○

TGPat1 = Rotate_y(90) * Translate(-3,0,2);

TGPat2 = Rotate_y(-90) * Translate(3,0,2);

○

NO CONTESTO

TG-2

Tenim una escena que conté dos Patricios, un damunt de l'altre que estan sobre un terra centrat a l'origen i sobre el pla XZ. El model del Patricio es pinta mitjançant la funció *pinta_Patricio()*. Les matrius de TGs (transformacions de model) dels Patricios són respectivament TG1 i TG2. Sabem que el centre de la base de la capsa contenidora del Patricio de sota està en el punt (5,0,5). Volem que els dos Patricios girin simultàniament 90 graus al voltant de l'eix paral·lel a l'eix Y que passa pel centre dels dos Patricios (és a dir, al voltant de la seva vertical). Quin tros de codi ens serviria per a pintar aquesta escena?

○

TGaux = Translate (5, 0, 5) * Rotate_y (90) * Translate (-5, 0, -5);

TG = TGaux * TG1;

modelMatrix (TG);

pinta_Patricio ();

TG = TGaux * TG2;

modelMatrix (TG);

pinta_Patricio ();

○

TGaux = Rotate_y (90);

TG = TGaux * TG1;

modelMatrix (TG);

pinta_Patricio ();

TG = TGaux * TG2;

modelMatrix (TG);

pinta_Patricio ();

○

TGaux = Translate (5, 0, 5) * Rotate_y (90) * Translate (-5, 0, -5);

TG = TG1 * TGaux;

modelMatrix (TG);

pinta_Patricio ();

TG = TG2 * TGaux;

modelMatrix (TG);

pinta_Patricio ();

○

TGaux = Translate (-5, 0,- 5) * Rotate_y (90) * Translate (5, 0, 5);

TG = TGaux * TG1;

modelMatrix (TG);

pinta_Patricio ();

TG = TGaux * TG2;

modelMatrix (TG);

pinta_Patricio ();

○

NO CONTESTO

TG-3

Tenim una funció pinta_ninot() que pinta un ninot de neu amb el centre de la base de la seva capsa contenidora al punt (-2,0,2), d'altura 2 i mirant cap a les Z positives. Volem usar aquesta funció per pintar un segon ninot de neu també d'altura 2 que tingui el centre de la base de la seva capsa contenidora al punt (2,0,-2) i mirant cap a les Z negatives. Quina és la TG a aplicar a aquest segon ninot de neu de manera que compleixi els requeriments i que utilitzi el mínim nombre de matrius a multiplicar?

○ TGNouNinot = Rotate_y(180);

○ TGNouNinot = Translate(2,0,-2) * Rotate_y(180);

○ TGNouNinot = Translate(2,0,-2) * Rotate_y(180) * Translate(2,0,-2);

○ TGNouNinot = Rotate_y(90);

○ NO CONTESTO

VAO-VBO

Disposem de dues funcions pinta_Lego() i pinta_Paret() que s'utilitzen per pintar una escena que té tres parets i dos legoman. Quants VAOs i VBOs cal utilitzar per a pintar aquesta escena? Suposem que per a cada model usem dades de posició dels vèrtexs i color.

○ 2 VAOs, 4 VBOs

○ 5 VAOs, 10 VBOs

○ 2 VAOs, 2 VBOs

○

5 VAOs, 5 VBOs

NO CONTESTO

Web-apps

Quan fem aplicacions via web apps per a mòbils, quina de les següents afirmacions es **FALSA**?

Tenim més varietat de widgets per desenvolupar GUIs que amb les apps natives.

Només hem de fer el desenvolupament una vegada.

És fàcil de desenvolupar perquè el SW és compatible amb quasi totes les plataformes.

Mantenir a tots els usuaris amb l'última actualització és molt més senzill que amb les apps natives.

NO CONTESTO

Enviar