Desenvolupament d’un videojoc 3D basat en OpenGL

Begoña Catalina Mercadal Pons
Enginyeria Tècnica en Informàtica de Sistemes

Consultora: Eva Monclús Lahoya

18 de juny de 2.004

Pròleg

En la societat tecnològica actual són molts, per no dir gairebé tots, els àmbits en què hi trobem presència directa o indirecta de la informàtica. I lògicament, donada la diversificació dels seus caps d'aplicació, al llarg del temps, han anat sorgint nombroses ramificacions cada cop més especialitzades dins del seu context.

En el transcurs dels estudis d'Enginyeria Tècnica en Informàtica de Sistemes s'han pogut aprendre acuradament gran part dels aspectes necessaris per adquirir una visió global d'aquesta disciplina, començant per l'estructura més bàsica d'un computador, i passant per l'art de la programació, les bases de dades, les xarxes i els sistemes de comunicació, per citar-ne alguns, tot afegint una introducció a la criptografia, la informàtica gràfica i el cada dia més rellevant estudi de la interacció de l'home amb els ordinadors.

Retornant al comentat anteriorment, una de les rames més atractives i amenes dins el món de la informàtica és, sens dubte i segons el parer de la que aquí subscriu, la "Informàtica Gràfica" que, gràcies a l'auge i a la força que han anat cobrant les anomenades aplicacions gràfiques ha experimentat un ràpid creixement com a conseqüència de les investigacions realitzades en aquest camp i al desenvolupament de nous llenguatges de programació primer i a l'aparició de llibreries gràfiques més tard. Aquestes últimes han suposat un fort impuls en el camí cap a la implementació de qualsevol aplicació de caire gràfic, gràcies a l'ajuda que suposa pels programadors a l'hora de dissenyar un nou producte.

És precisament en una d'aquestes llibreries gràfiques, OpenGL, en què s'ha basat el producte desenvolupat en el present projecte.

Una de les aplicacions gràfiques que més força i protagonisme ha adquirit, entre els joves en especial i entre la societat en general, són el "Videojocs". Molts són el productes d'aquest tipus que han aparegut en el mercat i moltes són les empreses dedicades a la seva producció i distribució a nivell mundial. Tots els temes imaginables i gairebé totes les situacions les podrem trobar, ben segur, convertides en un joc virtual que podrem tenir instal·lat als nostres ordinadors personals o estar disponibles via internet per jugar un partida on-line amb algú de l'altre punta del globus.

Es presenta en aquest treball una aplicació gràfica en la que es desenvolupa un Videojoc, de temàtica lliure, amb premisses senzilles però fermes i amb la utilització d'unes eines el funcionament de les qual s'ha assolit durant la carrera i aprofundit en el temps que ha durat la confecció d'aquest projecte.

Índex

Introducció

El present document s'edita com a reforç explicatiu a la construcció d'un nou producte: un *Videojoc Tridimensional.*

Així, en els següents apartats es repassen, punt a punt, tots els aspectes relatius a l'aplicatiu dissenyat, tant pel que fa a la seva funcionalitat, com a l'estructura de dades definida i als algorismes utilitzats.

Objectius

Objectius globals:

- L'objectiu fonamental és el desenvolupament d'un Videojoc 3D basat en la llibreria gràfica OpenGL.

  De contingut lliure, el Videojoc incorpora les funcionalitats bàsiques següents:

  - Dotar a aquesta aplicació gràfica d'interactivitat home-màquina
  - Definir alguna meta incorporant algun tipus de control de la puntuació o de les accions de perdre o guanyar.

- Assolir uns coneixements amplis i sòlids en l'ús de l'esmentada llibreria gràfica, objectiu que ve directament lligat a la consecució del primer.

Objectius parcials:

- Disseny del contingut i finalitat del joc
- Visualització realista d'un entorn tridimensional emprant OpenGL
- Definició i manejament de l'entrada de l'usuari
- Inclusió d'efectes d'increment de realisme emprant OpenGL

Metodologia

La metodologia utilitzada s'ha basat en el seguiment, en la mesura del possible, de l'exposat en el *Pla de Treball*. En aquest, i tal com es reprodueix en l'apartat de *Planificació del Projecte* del present document, es distribueix la feina en distintes fases assignant a cadascuna d'elles un període de temps proporcional a la càrrega del treball a realitzar.

Les pautes seguides són les que es resumeixen a continuació:

- Definició de la temàtica del Videojoc
- Realització d'un esquema general de l'aplicació:
    - Objectius del joc
    - Definició de l'escenari o món virtual en què es situarà l'acció
    - Definició dels objectes que s'integraran en aquest escenari
    - Possibles modes de funcionament
- Creació d'un Pla de Treball per a desenvolupar l'esquema anterior, bàsic per organitzar la feina.
- Desenvolupament de cada part del projecte. Per a cada fase:
    - Esbós i recopilació d'idees
    - Estudiar i investigar les matèries necessàries per a dur a terme els objectius de l'apartat
    - Definició de l'estructura més adient
    - Implementació
    - Validació i realització de proves
- I per últim, realització d'una validació i optimització final del producte obtingut

Planificació Del Projecte

Fases	Data	Activitat
1	1-8 març 2004	Definició de la temàtica del Videojoc Elaboració del Pla de Treball
2	9-28 març 2004	Creació i disseny de l'escenari: • Objectes 2D: polígons, parets de l'habitació, finestra... • Objectes 3D: cilindres, paral.lelepípedes, octàedres... • Distribució de tots els objectes creats dins l'escenari virtual
3	29 març – 5 abril 2004	Creació i disseny del personatge: • El cos • Aplicació de textures
4	6-21abril 2004	Interactivitat amb l'usuari: • Disseny de les pautes de moviment • Implementació de la manera com l'usuari dirigirà el moviment del personatge a través de l'escenari • Proves de moviment
5	22 abril – 5 maig 2004	Creació i disseny d'un sistema de preguntes i respostes: • Recopilació de les preguntes amb les seves possibles respostes • Creació d'un sistema de visualització gràfica de les anteriors • Mètode d'interacció amb l'usuari de cara a que pugui respondre a cada pregunta.
6	6-16 maig 2004	Sistema de puntuació: • Creació del sistema i les normes de puntuació • Representació gràfica
7	22 abril – 16 maig 2004	Proves de les dues fases anteriors
8	17-23 maig 2004	Inclusió d'efectes especials per l'expressió gràfica de diferents esdeveniments del joc
9	24-31 maig 2004	Proves finals i optimitzacions
10	1-18 juny 2004	Preparació de la memòria final i la presentació del producte desenvolupat

Març								**Abril**						
Dill	Dim	Dimc	Dij	Div	Dis	Diu		Dill	Dim	Dimc	Dij	Div	Dis	Diu
1	2	3	4	5	6	7					1	2	3	4
8	9	10	11	12	13	14		5	6	7	8	9	10	11
15	16	17	18	19	20	21		12	13	14	15	16	17	18
22	23	24	25	26	27	28		19	20	21	22	23	24	25
29	30	31						26	27	28	29	30		
Maig								**Juny**						
Dill	Dim	Dimc	Dij	Div	Dis	Diu		Dill	Dim	Dimc	Dij	Div	Dis	Diu
					1	2			1	2	3	4	5	6
3	4	5	6	7	8	9		7	8	9	10	11	12	13
10	11	12	13	14	15	16		14	15	16	17	18	19	20
17	18	19	20	21	22	23		21	22	23	24	25	26	27
24	25	26	27	28	29	30		28	29	30				
31														

Aquesta és la planificació que a priori es va fer i presentar en el seu moment en el document del *Pla de Treball*.

Les deu fases en que es va dividir la feina es van distribuir en el temps d'acord amb el calendari que aquí s'exposa. Un cop feta, cal dir, que en alguns casos el temps requerit per a resoldre el treball de la fase ha estat insuficient i en d'altres, ajustat però correcte. Durant aquest mesos de treball, s'ha tingut sempre present aquesta planificació intentant ajustar-la al temps real invertit, encara que no sempre ha pogut ser així, i en determinades circumstàncies, com per exemple la fase 4 "Interactivitat amb l'usuari", ha requerit molt més temps del previst i en determinats moments s'ha hagut de treballar en paral.lel amb fases posteriors, degut als molts problemes que ha portat.

Descripció del Videojoc

Visió global

El Joc

L'usuari haurà de dirigir el Personatge per l'escenari per tal d'aconseguir guanyar un total de deu premis. Aquest és per tant l'objectiu principal d'aquest joc, amb el repte afegit d'acumular la màxima puntuació possible.

Per tal d'aconseguir cada premi es formularà una pregunta que l'usuari haurà de respondre correctament. En aquest cas el premi passarà a ser seu, en cas contrari tindrà l'oportunitat de respondre fins a dues preguntes més, després de les quals, perdrà la partida si ha respost incorrectament.

La Puntuació

La màxima puntuació possible serà de 100 punts i 10 premis.

Per a cada premi, l'usuari tindrà 3 oportunitats, o sigui, se li formularan fins a 3 preguntes diferents.

La puntuació obtinguda es regirà per les següents regles:

	Punts	Premis
Resposta correcta a la primera pregunta formulada	10	1
Resposta correcta a la segona pregunta formulada	5	1
Resposta correcta a la tercera pregunta formulada	1	1
Resposta incorrecta a la tercera pregunta formulada	Fi de la partida	Fi de la partida

El Personatge

El Personatge es pot moure per tot l'escenari, controlat per l'usuari mitjançat el teclat:

- Moviment pel terra de l'escenari en totes direccions: esquerra, dreta, avançar, retrocedir
- Moviment per l'aire (mode levitar): pot moure's per l'aire canviant d'altura i en totes direccions.
- Moviment al llarg de l'eix Z

L'escenari

L'escenari es situa en una habitació de jocs. Repartits per tota aquesta habitació es troben diferents objectes que representen obstacles pel Personatge. Es tracta d'objectes sòlids com per exemple: un joc de cubs, un joc de construcció, una taula, una prestatgeria...

Indicacions D'ús Pel Jugador

Un cop iniciada la partida, el jugador haurà de dirigir el Personatge per l'escenari evitant els obstacles que es trobarà al mig per tal d'apropar-se a cadascun dels premis l'un darrera l'altre. Per aconseguir-ho utilitzarà les combinacions de tecles següents:

→	Dreta
←	Esquerra
↑	Avançar
↓	Retrocedir
'+'	Avançar al llarg de l'eix Z
'-'	Retrocedir al llarg de l'eix Z
RePág	Mode Levitar: cap amunt
AvPág	Mode Levitar: cap avall

Un cop el Personatge es trobi situat vora un premi, per tal d'aconseguir-lo s'haurà de contestar correctament a una pregunta. Per visualitzar-la, el jugador utilitzarà la tecla de la barra espaiadora:

Espai	Nova pregunta

Cal tenir present que una pregunta només es podrà visualitzar en el moment en què el Personatge topi amb el premi.

Si l'encerta, el premi és seu, i per tant desapareixerà de l'escenari. Al marcador es podrà veure immediatament el resultat: la puntuació aconseguida sumada a l'acumulada fins el moment i el nou premi aconseguit.

Si falla la pregunta, el jugador haurà de tornar a contestar-ne una altra. Com?, doncs, topant un altre cop amb el premi i pulsant la tecla indicada més amunt. I així fins a un màxim de 3 preguntes. Si falla a la tercera, perdrà la partida. Un missatge final informarà al jugador de la puntuació obtinguda i dels premis aconseguits.

El programari Utilitzat

Per a la realització i visualització d'aquest projecte s'ha utilitzat el programari següent:

- Java
  J2sdk1.4.1_01

- Llibreria gràfica GL4Java
  gl4java2.8.2.0

- Navegador
  Internet Explorer 6.0

L’Aplicació

Classes Implementades

Per tal de desenvolupar aquest projecte s’han dissenyat i implementat les següents classes, de les quals, en les properes pàgines, se’n farà una detallada descripció:

- SolidFronteres
- Vertex
- Punt3D
- Cara
- Material
- Escena
- Caseta
- Construccio
- EscalaDoble
- Estanteria
- Llum
- Taula
- Preguntes
- Personatge
- Marcador
- FinalJoc
- projecte i dibuix

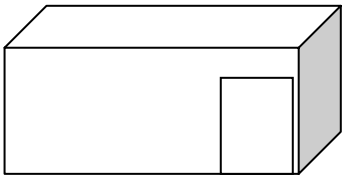
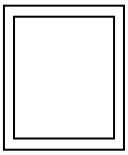
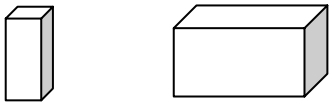
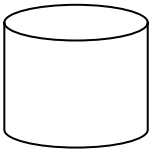
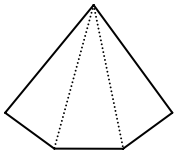
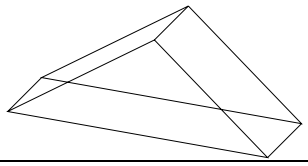
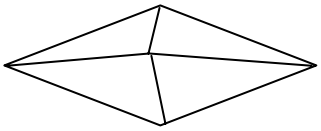
Descripció

Classe SolidFronteres

Per a la representació dels objectes sòlids que apareixen en el joc es va decidir optar per la representació per fronteres, implementant així la classe SolidFronteres, la qual emmagatzema tota la informació geomètrica i topològica de cada entitat (cares, vèrtexs, materials). Aquesta classe alhora fa ús de les classes Vèrtex, Punt3D, Cara i Material, que es comentaran més avall.

SolidFronteres
- vertexs[ ] :Vertex - cares [ ] : Cara - nc, nv: Integer - tipus: String - mat[ ]: String + orx,ory, orz, a, h, p;
+SolidFronteres() +Habitacio(x0: float, y0: float, z0: float, x1: float, y1: float, z1: float, matpareds: String, matterra: String) +Finestra(x0: float, y0: float, z0: float, x1: float, y1: float, z1: float, matmarc: String, matvidre: String) +CapsaSabates(x0: float, y0: float, z0: float, x1: float, y1: float, z1: float, material: String) +Cilindre(cx: float, cy: float, cz: float, R: float, v: float, H: float, material: String) +Lampara(cx: float, cy: float, cz: float, v: float, R: float, H: float, material: String) +PiramidePlana(x0: float, y0: float, z0: float, x1: float, y1: float, z1: float, H: float, material: String) +Octaedre(cx: float, cy: float, cz: float, R: float, material: String) + getNumVertexs(): float + getNumCares(): float + getCara(i: int): Cara + getVertex(i: int): Vertex + getTipusSolid(): String + getMaterial(i: int): String + getExtremX(): float + getExtremY(): float + getExtremZ(): float + getAmplada(): float + getAlsada(): float + getProfunditat(): float + setNumVertexs(numv: int) +setTipusSolid(tip: String) + setMaterial(material: String, i: int) + setExtremX(x: float) + getExtremY(y: float) + getExtremZ(z: float) + getAmplada(amplada: float) + getAlsada(alsada: float) + getProfunditat(produnditat: float)

Així, tal i com es veu a l'esquema anterior, aquesta classe representa els sòlids:

Habitació. Lloc on transcorre l'acció (parets, sostre i terra. Una de les parets té una obertura que representa la porta)	
Finestra de l'habitació formada per un marc de llenya i un vidre	
CapsaSabates. Paral.lelepípede usat en diversos objectes (escala, estanteria...)	
Cilindre	
Lampara. Es tracta de la figura geomètrica del con a la que se li ha tret la cara base. Servirà de mampara pel llum del sostre.	
PiramidePlana. Usat en diversos objectes (caseta, construcció)	
Octaedre. Servirà per a representar el premis.	

Classe Vèrtex

Per a representar cada vèrtex dels sòlids.

Vertex
- x, y, z: float
+Vertex(cx: float, cy: float, cz: float) +Vertex(p: Punt3D) +getX(): float

+getY(): float +getZ(): float +setX(cx: float) +setY(cy: float) +setZ(cz: float) +set(cx: float, cy: float, cz: float) +set(v: Vertex) +rotX(angle: double) +rotY(angle: double) +rotZ(angle: double)

Classe Punt3D

Per a representar un punt en un espai tridimensional.

Punt3D
- x, y, z: float
+Punt3D(cx: float, cy: float, cz: float) +Punt3D(cx: int, cy: int, cz: int) +getX(): float +getY(): float +getZ(): float +setX(cx: float) +setY(cy: float) +setZ(cz: float)

Clase Cara

Per a representar cada cara dels sòlids.

Cara
-nv: int -taulav[ ]: int -normal: Punt3D
+Cara() +getNormal(): Punt3D +getNumVertexs(): float +getIndexVertex(i:int): int +setNormal(n: Punt3D) +setNumVertex(v: int) +setIndexVertex(i: int, v: int) +addVertex(v: int)

Classe Material

Per a definir les components del material de què estarà fet cada sòlid.

Material
ambient[][] : float Diffuse[][]: float especular[][]: float brillantor[][]:float nm: int nommaterial[]: String
+Material() +getAmbient(nom: String): float[] +getDiffuse(nom: String): float[] +getEspecular(nom: String): float[] +getBrillantor(nom: String): float[] +indexMaterial(nom: String): int +materials()

Classe Escena

Per emmagatzemar aquella informació relativa a l'escenari del joc referent als sòlids que hi pertanyen (afegir un sòlid a l'escena, número de sòlids de l'escena, esborrar un sòlid...), s'utilitza la classe Escena que aquí es presenta:

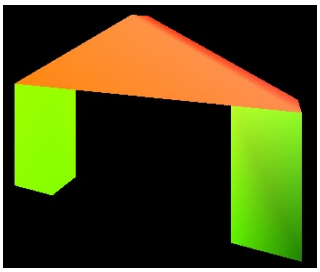
Escena
-solids[ ]: SolidFronteres -ns: int
+Escena() +getNumSolids(): int +getSolid(i:int): SolidFronteres +addSolid(s:SolidFronteres) +delSolid(i: int) +addEscena(pEsc: Escena)

A partir de la Classe SolidFronteres, o més exactament, a partir dels sòlids que en ella s'implementen, s'han creat altres objectes, formats com a resultat de la composició de diversos sòlids primitius. Parlem dels objectes: Caseta, Construcció, Escala doble, prestatgeria, llum i taula.

Classe Caseta

Hereva de la classe Escena, construeix un objecte sòlid, semblant a una caseta de joguina (feta amb peces de construcció). La formen els sòlids de la classe SolidFronteres: PiramidePlana per a simular el sostre, i CapsaSabates per a simular els pilars o parets.

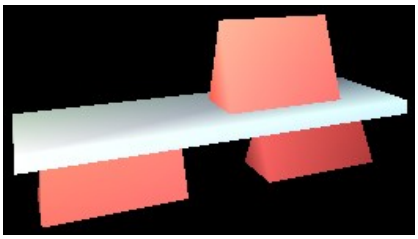
Caseta
+Caseta()



Classe Construcció

Hereva de la classe Escena, construeix un objecte format per un sòlid CapsaSabates i tres sòlids PiramidePlana, tots ells de la classe SolidFronteres. Aquest nou sòlid compost vol venir a representar un joc de construcció, on es van apilant diverses peces per formar una estructura.

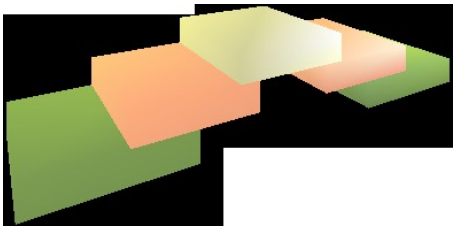
Construccio
+Construccio()



Classe EscalaDoble

Hereva de la classe Escena, implementa un objecte compost per cinc sòlids CapsaSabates de la classe SolidFronteres. Representa una escala que puja i baixa, on cada sòlid primitiu és un esglaó.

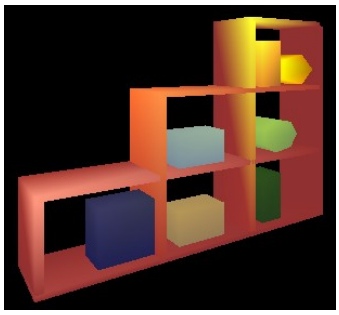
EscalaDoble
+EscalaDoble()



Classe Estanteria

Amb la composició de diversos sòlids CapsaSabates, es formen els tres nivells d'aquesta prestatgeria, i a cadascun d'ells si situen diferents sòlids també de la classe SolidFronteres, amb la finalitat de representar les joguines. Igual que en els casos anteriors, aquesta classe és hereva de Escena.

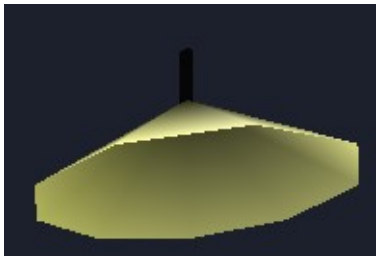
Estantería
+Estanteria()



Classe Llum

Per a representar el llum del sostre, s'usa el sòlid Lampara, més un cilindre amb el radi molt petit per a simular el fil que l'aguanta del sostre. Hereta també de la classe Escena.

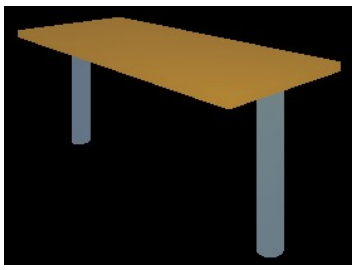
Llum
+Llum()



Classe Taula

Construeix una taula, composta per un sòlid CapsaSabates i per dos sòlids Cilindre de la classe SolidFronteres, que fan el fet de potes. És hereva de la classe Escena.

Taula
+Taula()



Classe Preguntes

Aquesta classe implementa el sistema de visualització gràfica i el mètode d'interacció amb l'usuari del "Sistema de Preguntes i Respostes" del joc.

Inclou una petita recopilació de preguntes amb les seves corresponents respostes, a mode de base de dades local.

Per a la visualització gràfica d'aquest sistema, s'ha optat per dissenyar una nova finestra utilitzant les capacitats que ofereixen els paquets AWT i SWING. Així la classe Preguntes, heretant de JPanel, construeix una finestra, i exposa la pregunta i les respostes mitjançant la utilització d'etiquetes i botons de ràdio.

Per a implementar la interactivitat amb l'usuari, es fa el corresponent tractament dels events generats pels mencionats botons de ràdio (amb la classe RadioListener). Aquest event es produirà quan el jugador pulsi l'opció que creu correcta amb el ratolí, o bé la seleccioni amb la combinació de les tecles ALT+1,2 o 3, segons correspongui.

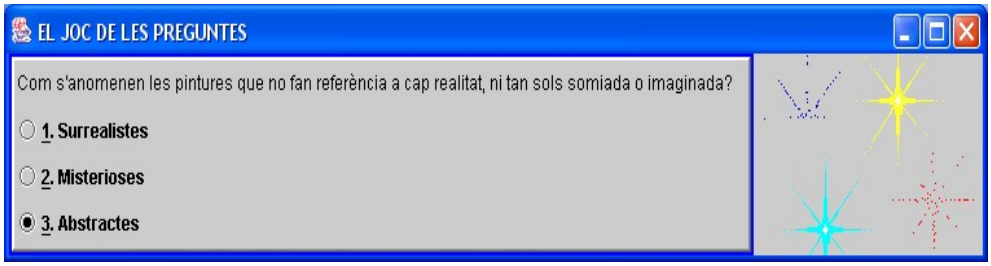
Quan es selecciona una de les possibles opcions que es mostren com a respostes, automàticament i a la mateixa finestra, apareix una animació (gif animat), que serà diferent segons sigui la resposta vertadera o falsa. Després d'això aquest finestra automàticament es tancarà. Per solucionar el problema que va sorgir quan es va comprovar que la finestra es tancava tan ràpid que no deixava veure aquest animació, es va optar per implementar una altre classe auxiliar, TimerListener, amb la qual es retarda un parell de segons la desaparició de la finestra, temps suficient perquè el jugador pugui observar la validació visual que l'aplicació li dóna, o sigui l'animació.

Resta comentar, que a aquesta finestra se li ha tret la funcionalitat del botó de tancar (x) comú en totes les finestres, donat que no es permet al jugador descartar la pregunta sense haver donat una resposta.

Preguntes
+ frame :JFrame + reaccioVisual : JLabel + resposta, solidValid: int + valid: boolean + timer: Timer
+ ferPreguntes() + nombreAleatori(): int + getResposta(): int + getPosicioValida(): boolean + getSolidValid(): int + setResposta(r: int) + setPosicioValida(v: Boolean) + setSolidValid(i: int)

RadioListener
+ actionPerformed(e: ActionEvent)

TimerListener
+ actionPerformed(evt: ActionEvent)



Classe Marcador

Amb aquesta classe s'implementa la representació gràfica del "Sistema de Puntuació".

En primera instància, es va optar per situar aquest objecte dins la habitació, com a part integrant de la decoració. El problema sorgí en el moment en què s'observà que el text no es mantenia estàtic a la posició indicada, sinó que experimentava un moviment que feia que sempre es veiés de manera frontal a la pantalla, efecte que no era el desitjat. Per aquest motiu i per a solucionar el problema, es va optar per convertir-lo en un element "immòbil", no part ja de la decoració, sinó quedant sempre al mateix lloc de la pantalla i en primer pla. D'aquesta manera, l'objecte del qual parlem està ubicat a l'extrem inferior esquerra de la pantalla, i en ell es poden veure reflectits els punts i premis aconseguits pel jugador. Aquestes dades s'actualitzen, així doncs, en temps real.

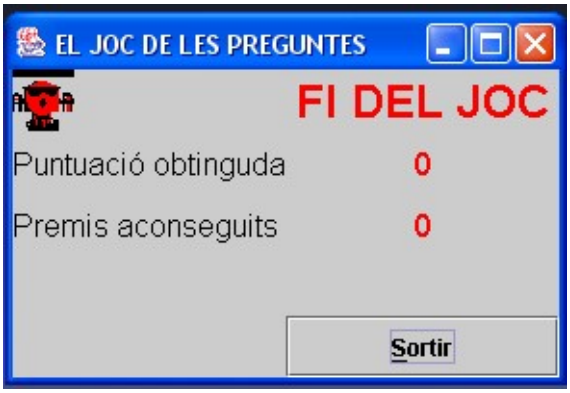
Hereta de GLJPanel i té forma quadrangular. Sobre el fons, se li ha incorporat una textura (utilitzant com a tal una imatge en format png) amb la finalitat de fer-lo destacar de la resta dels elements de l'habitació, en els que s'han utilitzat materials de colors llisos.

Per últim, destacar la utilització de la GLUT, gràcies a la qual ha estat possible, d'una manera simplificada, la presentació del text.

Marcador
+ gl: GLFunc + glut: GLUTFunc
+ Marcador() + dibuixarMarcador(puntuacio: int, numpremis: int)

Classe FinalJoc

Per a finalitzar el joc, es presenta un petit missatge:



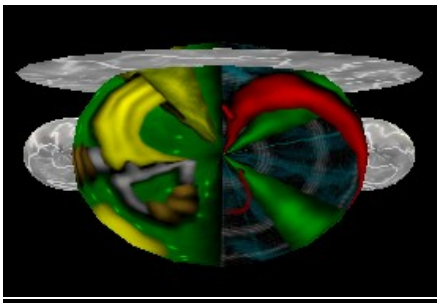
D’aquesta manera es mostren els totals aconseguits pel jugador, més el missatge informatiu de què la partida ha finalitzat (“FI DEL JOC”), ja estigui aquesta perduda o guanyada. Aquesta petita finestra aporta un element clau de cara a finalitzar l’aplicació, ja que d’altre manera l’acabament podria resultar sobtat i fred.

Aquesta classe hereta de JPanel i utilitza etiquetes per a representar el text. Incorpora una petita animació per donar-li un toc divertit i un botó per a tancar l’aplicació. Per aquest últim s’ha afegir una classe (BotoListener) dedicada íntegrament a respondre a l’event del botó (pulsat amb el ratolí o amb la combinació de les tecles ALT+S) amb l’operació de tancar el sistema, o sigui, el joc.

FinalJoc
+ frame: JFrame
+ presentarMissatge(punts: int, premi: int)

BotoListener
+ actionPerformed(e : ActionEvent)

Classe Personatge



El personatge serà el protagonista virtual del joc. Serà ell qui guiat pel jugador, es mourà per l’escenari per recollir els premis, i sobre ell recauran en tot moment les conseqüències de cada aconteixement que es produeixi (contestar malament una pregunta, xocar amb un sòlid…). Per tant, el Personatge proporcionarà la component principal d’interactivitat home-màquina d’aquest aplicació.

A l’hora de dissenyar aquest personatge es va optar per donar-li una estructura senzilla per a fer-lo més manejable a l’hora d’implementar els moviments. Així, el seu cos està constituït per una esfera, a la que més tard se ni afegeixen dues més de radi inferior i situades a extrems oposats del cos per a simular dos braços. Un disc o donut a la part superior, o sigui al cap, representa un capell amb visera. Per a implementar aquestes figures s’ha utilitzar la GLU, degut a les facilitats que comporta l’ús d’unes figures ja definides.

Per complementar l’estructura corporal del personatge, se l’ha “vestit”, recobrint cos, braços i capell amb textures, preses d’imatges en format png. Amb això, el personatge destaca significativament de la resta de figures del nostre escenari donat el colorit de les textures escollides, afecte desitjat ja que els ulls del jugador fàcilment dirigiran la seva atenció cap a ell.

La càmara del joc estarà en tot moment enfocada cap al Personatge, seguint els seus moviments

Personatge
+ gl: GLFunc +glu: GLUFunc + orx,ory,orz: float + a,p,h: float + material[]: float
+ Personatge(gl: GLFunc, glu: GLUFunc) + dibuixarPersonatge() + getMaterial(): float[] + getExtremX(): float + getExtremY(): float + getExtremZ(): float + getAmplada(): float + getAlsada(): float + getProfunditat(): float + setExtremX(x: float) + setExtremY(y: float) + setExtremZ(z: float) + moviment(x: float, y: float, z: float, r: float)

A les properes seccions s'exposarà amb deteniment tot el que afecta al moviment i demés accions que porta a terme el personatge o bé que li afecten directament.

Classes projecte i dibuix

Totes les classe descrites fins aquest moment, són utilitzades per les classes projecte i dibuix, des de les quals, amb la implementació dels algorismes pertinents, es construeix un tot anoment Joc de les Preguntes.

La classe projecte hereta de JApplet i bàsicamanet el que fa és inicialitzar l'applet que contindrà el joc i un objecte de la classe dibuix, a més de crear l'escena, o sigui, afegir els sòlids descrits més amunt a un objecte de la classe Escena, el qual representarà l'escenari del joc.

La classe dibuix hereta de GLCanvas i a través d'ella es fa ús de la llibreria gràfica OpenGL. Incorpora els mètodes i algorismes necessaris per a pintar el objectes de l'aplicació i per a dur a terme totes les accions a realitzar, així com la captura i tractament d'events provinents del teclat, ratolí i finestres.

Interactivitat

Moviement Per L'escenari

El jugador dirigeix al Personatge amb el teclat.

La classe dibuix implementa KeyListener, i amb el mètode KeyPressed es coordina el moviment per l'escenari. Així, amb la pulsació de les fletxes podem moure el Personatge per tots els rincons de l'habitació, esquivant això sí, els obstacles que es troba pel camí, donat que com a objectes sòlids que són, no se'ls pot traspassar. Per això abans de portar a terme cap moviment, es realitza una simulació d'aquest comprovant si provoca un xoc amb un objecte sòlid. Només si té espai lliure es podrà moure el Personatge en la direcció que se li indiqui.

Amb la pulsació de les esmentades tecles dirigim el Personatge sempre pel terra de l'habitació, però si necessitam que pugi a sobre un objecte per a buscar un premi, per exemple que es trobi a dalt de la taula, llavors disposem del mode levitació (tecles RePág i AvPág). En lloc de realitzar l'acció de saltar, tal i com tots entenem el significat del terme, es va decidir que el Personatge es pogués moure lliurament per l'aire a l'alçada escollida, imaginant que el capell que porta li otorga la capacitat de volar.

Per últim, l'altre tipus de moviment consisteix en moure el Personatge al llarg de l'eix Z. Aquest idea va sorgir com a conseqüència de les proves realitzades, de les quals es va extreure la conclusió que alguns cops sorgien dificultats a l'hora de desplaçar el Personatge amb l'única ajuda de les cuatre fletxes. Aquesta opció afegida preten aportar una major facilitat d'ús.

Cal recalcar que la posició que ocupa el Personatge dins aquest món tridimentsional i virtual vé donada sempre per les coordenades del seu centre:

```
(  posx+VDirec.getX() ,  posy  ,  posz+VDirec.getZ()-r  )
```

Per implementat el moviment utilitzem:

- Tres variables per a indicar la posició del punt centre del Personatge en tot moment: *posx, posy, poz*
- Un vector per indicar la direcció: *VDirec*

Així per avançar només hem de sumar:

```
posx + VDirec.getX()*0.02          ….
```

I per retrocedir, restar:

```
posx - VDirec.getX()*0.02          ….
```

Per efectuar un gir a la dreta o a l'esquerra el que es fa es rotar les coordenades del centre: rotam un punt 3D en torn a l'eix Y, utilitzant el mètode definit a la classe *Vertex*:

```
public void rotY(double angle)
{
  float longi = (float)(java.lang.Math.sqrt((x*x)+(z*z)));
  x = (float)(java.lang.Math.cos(angle)*longi);
  z = (float)(java.lang.Math.sin(angle)*longi);
}
```

Visualitzar Una Pregunta

Cada cop que el Personatge xoca amb un objecte sòlid del tipus “premi”, es pot visualitzar una nova pregunta. El procés no és automàtic, primera perquè resulta molt fàcil fer xocar el Personatge amb el premi per error més d’un cop seguit i llavors en lloc d’una pregunta en tindríem dues o més al mateix temps, cosa que provocaria confusió en el jugador i error en l’aplicació, i segona perquè així es dóna més protagonisme al jugador, el qual pot decidir en quin moment està preparat per a visualitzar i contestar la pregunta en qüestió.

L’acció de visualitzar una pregunta es portarà a terme amb la pulsació de la barra espaidora.

El principal problema que es va trobar en aquest punt fou la captura de la resposta donada per l’usuari, és a dir, quan aquest seleccionava una de les tres respostes possibles exposades a la finestra de la pregunta. Com saber si ha encertat o ha fallat? O més ben dit, com sabrà la classe dibuix si la resposta ha estat vertadera o falsa?. Després de diverses investigacions i proves efectuades es va optar per utilitzar la classe WindowListener i en concret el seu mètode windowActivated, o sigui capturant i tractant els events de finestra. Quan s’obre la finestra de les preguntes, la principal del joc passa a estar en segon pla, perd el focus, o sigui queda desactivada. Llavors quan es respon a la qüestió formulada i la finestreta de les preguntes es tanca automàticament, la finestra principal torna a recuperar el focus i per tant s’activa. És en aquest moment, quan es captura l’event, que podem consultar la variable de la classe Preguntes que ens diu si s’ha encertat o fallat i realitzar les demés accions en conseqüència, amb el mètode:

```
public int getResposta()
{
        return resposta;
}
```

La pega que té aquest mètode, és que la partida s’ha de jugar d’una tirada, i sense intervenció d’altres aplicacions, ja que si obrim més finestres el joc es malmetrà, donat que aquestes serien interpretades com a preguntes i en realitat no tindrien res a veure amb el joc en si.

Conseqüències De Les Preguntes

Si la resposta escollida és la vertadera:

- sumar la puntuació obtinguda i augmentar en un el nombre de premis aconseguits, o sigui, actualitzar el Marcador

- reproduir una melodia, per a indicar la satisfacció del Personatge, utilitzant el mètode activarMusica() que l'única cosa que fa és recuperar un fitxer (midi) per a la seva reproducció. La melodia té una durada curta d'uns 17 segons.

- Esborrar el premi aconseguit de l'escena.

    - Amb el mètode

```
public int getSolidValid () (de la classe Preguntes)
{
        …
}
```

      sabem amb quin dels sòlids tipus "premi" s'ha topat.

    - I amb el mètode

```
public void delSolid(int i) (de la classe Escena)
{
        …
}
```

      el fem desaparèixer de l'escena.

- Finalment, si s'aconsegueix acumular els deu premis, la partida es donarà per acabada.

```
if (numpremis = = 10)
{
        fi=true;
}
```

Si la resposta es falsa:

- El Personatge es posa trist i es torna vermell. Per fer-ho, canviem el material base que s'ha usat per pintar el Personatge, de blanc a vermell.

```
public void setMaterial(float[] mat) (de la classe Personatge)
{
        …
}
```

- Aquest estat es mantindrà fins que el jugador respongui correctament a la següent pregunta.

- En el cas especial que es fallin tres preguntes seguides, llavors significarà que s'ha acabat la partida (de la mateixa manera que succeirà quan s'aconsegueixin els 10 premis) i s'activarà el missatge de Fi del Joc (de la classe FinalJoc)

Les Col.lisions

Una part molt important de la funcionalitat del videojoc, és el moviment del personatge per l'escenari. Donat que els elements incorporats a aquest s'han definit com a objectes sòlids, s'han de concebre també com a objectes corporis, en el sentit ampli de la paraula: tenen un volum i ocupen un espai. Amb aquesta reflexió volem dir que no poden ser traspassats per cap altre objecte, cosa aplicable de igual manera a la figura del Personatge.

"El Personatge no podrà passar a través de cap sòlid de la nostre escena"

Per mor d'això, s'ha definit un algorisme per a detectar les col.lisions que es produeixin entre aquest Personatge i qualsevol sòlid situat a l'habitació de jocs.

Aquest ha estat sens dubte un dels punts més conflictius dins la feina de disseny i implementació realitzada.

Així tenim que abans de moure el Personatge s'haurà de determinar si la posició a on es vol anar és una posició vàlida, és a dir que, un moviment sempre serà possible si i només si el personatge no col.lisiona amb cap dels objectes sòlids presents.

Per a realitzar el càlcul de col.lisions entre Personatge-sòlid s'ha optat per implementar el mètode de les capses contenidores, és a dir, basant-nos en la premisa que si dos objectes col.lisionen les seves capses contenidores també ho faran.

Llavors, s'ha procedit a la definició de la capsa contenidora de cadascun dels objectes sòlids i naturalment del Personatge.

La capsa contenidora té els següents paràmetres:

- *(orx, ory, orz)* representen les coordenades de l'extrem inferior esquerra de la capsa
- *a* en representa l'amplada
- *h* l'alçada i
- *p* la profunditat

A la classe SolidFronteres, s'ha incorporat el càlcul de la capsa contenidora a cadascun dels mètodes de creació dels diferents sòlids.

Com a exemple:

```
public void CapsaSabates( … )
{
```

```
        …
        /* capsa contenidora del sòlid */
        //Coordenades de l'extrem inferior esquerra de la capsa
        orx=x0;
        ory=y0;
        orz=z1;
        //amplada
        a=java.lang.Math.abs(x1-x0);
        //alçada
        h=java.lang.Math.abs(y1-y0);
        //profunditat
        p=java.lang.Math.abs(z1-z0);
}
```

S'han inclós també els pertinent mètodes consultors i modificadors per a aquestes noves variables.

Tenint en compte que des del mètode Display() es sotmet els sòlids de la classe SolidFronteres a diverses transformacions geomètriques, s'ha procedit en cada cas en particular a actualitzar les dades de la capsa contenidora segons la transformació aplicada.

Un cop fet això ja tenim els sòlids a punt, donat que són objectes immòbils i per tant les dades de la capsa contenidora seran sempres les mateixes.

Pel que fa al Personatge, a la seva classe s'han incorporat també mètodes consultors i modificadors de les dades de la capsa contenidora i afegit els valors pertinents :

```
public void dibuixarPersonatge()
{
        …
        /* Capsa contenidora del Personatge*/
        // Les coordenades de l'extrem inferior esquerra (orx,ory,orz) varien
        // segons la posició que ocupa el personatge dins l'escena
        a=4.f; //amplada (invariables)
        h=3.f; //alçada
        p=4.f; //profunditat
}
```

Com podem veure, l'amplada, alçada i profunditat de la capsa contenidora del personatge són valors invariables, donat que el Personatge no canvia en cap moment el seu tamany. I pel que fa a les coordenades de l'extrem inferior de la capsa, variaran a cada moviment que dugui a terme el Personatge a través de l'escena. Aquestes coordenades s'hauran de calcular, doncs, en temps real, des de la pròpia classe dibuix.

Posem com a exemple el cas en que es vulgui fer avançar al personatge amb la tecla de la fletxa cap envant, considerant que aquest ocupa la posició:

```
(  posx+VDirec.getX(),  posy,  posz+VDirec.getZ()-r  )
```

es procedira d'aquesta manera:

```
case KeyEvent.VK_UP: {  //avançar
        xprova = posx + (float)(VDirec.getX()*0.02); //provar l'avanç
        zprova = posz + (float)(VDirec.getZ()*0.02);
        personatge.setExtremX(xprova+VDirec.getX()-2);
        personatge.setExtremY(posy-1.5f);
        personatge.setExtremZ((zprova+VDirec.getZ())-r-2);
        if (!colisions())
        {
                posx=xprova;//moviment permès
                posz=zprova;
        }
}
```

Comprovam si la posició a la que anirà a parar el sòlid és una posició vàlida, i assignam les noves coordenades a l'extrem inferior de la capsa contenidora amb els mètodes setExtremX, setExtremY, setExtremZ.

L'algorisme de col.lisions queda de la següent manera:

```
if (  ( escena.getSolid(i).getExtremX()<=(personatge.getExtremX()+personatge.getAmplada()) )
        && ( personatge.getExtremX()<=(escena.getSolid(i).getExtremX()+escena.getSolid(i).getAmplada()) ))
{
        if (  ( escena.getSolid(i).getExtremY()<=(personatge.getExtremY()+personatge.getAlsada()) )
                && ( personatge.getExtremY()<=(escena.getSolid(i).getExtremY()+escena.getSolid(i).getAlsada()) )  )
        {
                if      (  ( escena.getSolid(i).getExtremZ()<=(personatge.getExtremZ()+personatge.getProfunditat()) )
                && ( personatge.getExtremZ()<=(escena.getSolid(i).getExtremZ()+escena.getSolid(i).getProfunditat()) )  )
                {
                        col=true;
                        break;
                }
        }
}
```

Si es compleixen les tres condicions de dalt, les dues capses contenidores intersectaran provocant una col·lisió.

Com a apunt final d'aquest tema, resta comentar el tractament especial que s'ha fet per al sòlid "habitació", donat que en aquest cas el Personatge es troba a dins d'ell i no n'ha de poder sortir, o sigui, no pot passar a través de les parets i la porta està tancada. Per això s'ha fet una petita modificació a l'algorisme anterior, considerant que la capsa contenidora del Personatge s'ha d'ubicar en tot moment entre les coordenades de l'habitació:

El Personatge xocarà amb les parets, terra o sostre de l'habitació quan es compleixin les condicions:

```
((personatge.getExtremY()<-10.f)||((personatge.getExtremY()+personatge.getAlsada())>10.f)
    ||(personatge.getExtremX()<-20.f)||((personatge.getExtremX()+personatge.getAmplada())>20.f)
            ||(personatge.getExtremZ()<-20.f)||((personatge.getExtremZ()+personatge.getProfunditat())>10.f))
```

Altres Elements

La Il.luminació

Per a il.luminar l'habitació s'utilitzen quatre focus de llum.

- El primer es coloca a la llum del sostre, representant la bombilla. Aquest proporciona llum a tota l'habitació des de dalt.

- Dos més s'usen per crear l'efecte de què la llum d'algun fanal del carrer entra a través de la finestra il.luminant en la direcció en què es troba la prestatgeria.

- I l'altre s'usa més que res per donar més contrast a un racó que d'altre manera quedava obscur.

Conclusions

Un cop arribat a aquest punt, i reflexionant sobre els darrers mesos en què s'ha dut a terme aquest projecte, de tot el treball realitzat se'n poden treure algunes de les conclusions que aquí s'exposen:

- L'objectiu fonamental s'ha aconseguit:
    - s'ha creat un nou producte, un Videojoc tridimensional basat en OpenGL, interactiu i funcional
    - s'han adquirit i assimilat nous coneixements en aquesta disciplina, molts més dels imaginats en un principi.
- El videojoc compleix gairebé els objectius i expectatives exposades al començament d'aquest curs.

La realització d'aquest projecte ha estat, sens dubte, una feina molt laboriosa que ha requerit moltes hores d'estudi i preparació per a assolir els objectius desitjats.

Diversos han estat els entrebancs que s'han hagut de superar, alguns d'ells ben difícils de resoldre. El moviment del personatge pel món virtual i el problema de les col·lisions han estat els més conflictius. S'han pogut comprovar les immenses possibilitats que ofereixen les llibreries gràfiques a l'hora de dissenyar aplicacions tals com la presentada aquí i al mateix temps la necessitat d'adquirir un alt nivell de coneixements de la matèria abans de començar una tasca com aquesta.

Bibliografia I Fonts D'informació

- "Programación en Java 2", Zukowski, John.
  Ediciones Anaya Multimedia, 1.999

- "Introducció a la Informàtica Gràfica" (Mòduls didàctics de l'assignatura)
  Universitat Oberta de Catalunya

- API GL4java
  http://gl4java.sourceforge.net/docs/html/index.html

- "OpenGL Programming Guide" or "The Red Book"
  http://fly.cc.fer.hr/~unreal/theredbook/

- "Fonaments de Programació II" (Mòduls didàctics i altre material subministrat al CD de l'assignatura)
  Universitat Oberta de Catalunya

.