MIRES: Disseny i Implementació d'un Sistema de Realitat Mixta

Tomeu Canyelles Escarrer, *Enginyeria Informàtica* . Antoni Jaume-i-Capó, *Doctor en Informàtica*

***Abstract*— El present article recull tot el procés de desenvolupament d'un sistema de realitat mixta de baix cost. Es presenta l'arquitectura de tres nivells del sistema: es captura informació de l'entorn mitjançant un dispositiu RGB-D *(Kinect)*, que enregistra el núvol de punt en les estructures pertinents. Es reconstrueix un model en 3D de l'escena definint unes lleis físiques que governaran el món. Tot seguit, els usuari tendran mitjans per introduir-hi un conjunt d'objectes virtuals, que es regiran per les mateixes lleis físiques del món. El resultat és el sistema de realitat mixta desitjat: *MIRES*.**

***Index Terms*— Realitat Mixta, Kinect, OpenNI, Ogre3D, Bullet Physics, Qt.**

I. Introducció

S'ENTÉN per realitat el conjunt de tot el que existeix en el món real. Partint d'aquesta definició, i del concepte de virtualitat: tot allò que existeix només aparentment i no és real, es defineix la realitat mixta com una combinació perfecte entre la realitat i la virtualitat. Es parla de simbiosi entre ambdós mons en el sentit que és possible la interacció entre els usuaris/objectes reals i els sintètics.

La figura 1 mostra els diversos graus de compenetració entre els dos mons: real i virtual, i la nomenclatura atorgada.

Fig. 1. Distribució dels botons del teclat

Un sistema de realitat mixta és aquell que combina objectes reals i virtuals en un entorn real i permet la interacció tridimensional en els dos sentits: real-virtual i virtual-virtual. És a dir, tots aquells objectes virtuals que entrin en contacte amb algun objecte dins l'espai de col.lisions reaccionaran segons les lleis físiques que defineixen el món.

Tot projecte de realitat mixta, i en particular aquest: *MIRES*, satisfà uns principis [1]: combina objectes reals amb objectes virtuals; s'han d'eliminar de l'escena els objectes o la part d'ells *(sintètics o no)* que es trobi superposat per altres; ha d'existir una interacció en tres dimensions i en temps real; i la visualització no s'ha de veure limitada a cap tecnologia concreta.

bce892@gmail.com
anoti.jaume@uib.es

MIRES és una aplicació estructurada en tres fases: la captura de dades de l'entorn, el tractament d'aquestes i la visualització de la informació (Veure figura 2), que satisfà els principis d'un sistema de RM i un requeriment de costs. El dispositiu físic de captura i les eines emprades fan de *MIRES* un sistema de baix cost.

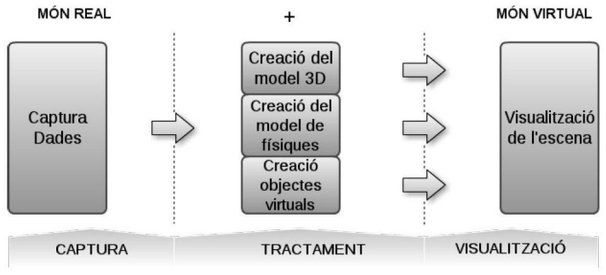


Fig. 2. Esquema de construcció del sistema de realitat mixta

El present document consta d'una breu descripció del projecte que donarà peu a descriure les eines de desenvolupament de l'aplicació. Tot seguit s'exposa el cos de l'article, les tres fases del projecte *MIRES*: captura, tractament i visualització.

Juny 22, 2012

II. Descripció general

El projecte consisteix dissenyar i implementar una aplicació de RM, és a dir, crear un espai basat en la realitat on interactuïn tant objectes i/o persones d'aquest món com els objectes virtuals.

El sistema ha de poder generar un model sintètic el més fidel possible a la realitat que s'ajusti a unes lleis físiques semblants a les del món real. Els usuaris, en qualsevol moment, han de poder ordenar la creació i la integració d'un conjunt d'objectes sintètics en el model, que actuaran de forma esperada d'acord a les lleis físiques definides. També han de poder visualitzar els resultats de la interacció entre els dos mons.

S'ha definit *MIRES* com un sistema de tres fases: captura, tractament i visualització. La fase de captura es caracteritza per la generació d'un núvol de punts amb informació de profunditat i color mitjançant un dispositiu RGB-D, la *Kinect*. La fase de tractament és l'encarregada de modelar l'escena i simular la interacció dels objectes atenent-se a uns paràmetres i lleis físiques definides. La de visualització és la que mostra als usuaris els resultats mitjançant un monitor de PC o televisor.

En la següent secció es defineixen i descriuen les principals característiques dels dispositius i eines que ha possibilitat la implementació del sistema.

III. Eines

S'utilitzen un o més eines per a cadascuna de les fases escollides d'entre diverses possibilitats principalment per ser de codi obert *(software lliure)* i gratuïtes. *OpenNI*, *Ogre 3D*, *Bullet Physics Engine*, *OgreBullet* i *Qt* són els *frameworks* seł.leccionats per a satisfer els objectius imposats sobre el sistema.

OpenNI: És un *framework* multiplataforma de codi obert que possibilita la interacció natural, persona-ordinador. S'organitza en un conjunt de llibreries que proporcionen una capa d'abstracció entre el hardware, encarregat de la captura d'informació *(àudio, color i profunditat)* i l'aplicació.

Ogre3D: Són les sigles de *Object-Oriented Graphics Rendering Engine* [4]. És un motor de renderitzat de gràfics orientat a objectes i de codi obert. Està format per unes potents llibreries multiplataforma desenvolupades en C++ i basada en components, és a dir, que està constituïda per un conjunt estructurat de classes que representen conceptes reals i faciliten la comprensió i la reutilització de codi.

Bullet Pyhisics: És un motor de físiques de codi obert caracteritzat per detecció de coł.lisions en 3D, per la creació de cossos rígids o deformables i sobretot per poder modificar els paràmetres que descriuen la física del món. La funció primordial de *Bullet Physics* [5] és la detecció i resolució de múltiples coł.lisions entre objectes.

OgreBullet: És un *wrapper* per *Ogre3D* que facilita la integració dels subsistemes de renderitzat i de físiques.

Qt: És el *framework* que es fa servir perquè l'usuari pugui visualitzar els resultats de la simulació. Són un conjunt de llibreries multiplataforma composta per mòduls que proveeixen funcionalitats específiques com gestió d'esdeveniments *(signals, slots i timers)* i la creació de finestres, botons, etiquetes, etc.

IV. Disseny i Implementació

La tasca de disseny d'un sistema informàtic consisteix en definir l'arquitectura, els processos, les estructures de dades, algoritmes i les interfícies d'usuari del mateix. En general, el propòsit d'aquesta secció és tractar aquests aspectes i detallar-ne alguns de la implementació.

A. Arquitectura del sistema

L'arquitectura del sistema *MIRES* està basada en l'estructura de les fases descrites anteriorment. Així doncs, es defineix una arquitectura de tres nivells (Veure figura 3).

El primer nivell: *Captura*, defineix les estructures de dades que compartiran la resta de subsistemes: logs, nodes i matrius de color, profunditat, etc. A més, carrega la configuració i els controladors dels dispositius físics. En definitiva, s'encarrega de la comunicació amb el hardware.

El segon nivell: *Tractament*, és el més complex, es divideix en tres mòduls: el generador del model 3D, el simulador de l'entorn físic i la interfície de comunicació entre el nivell superior i aquest.

El tercer nivell: *Visualització*, és el que gestiona l'entrada-sortida d'informació des de i cap a l'usuari, és a dir, controlar els paràmetres d'entrada i projectar les imatges de sortida a l'usuari.

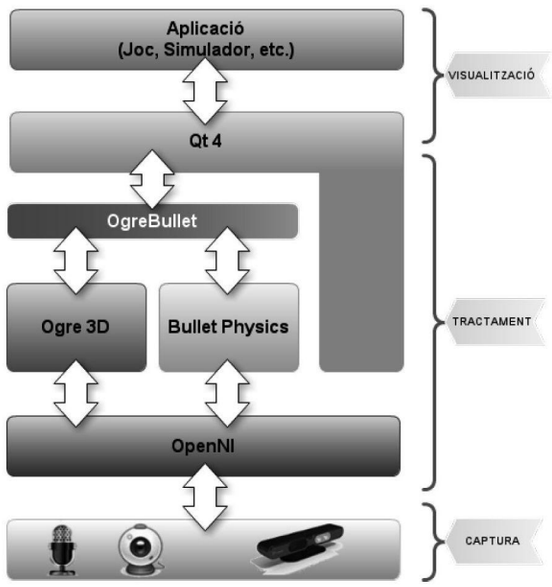


Fig. 3. Arquitectura del sistema *MIRES*

B. Processos

L'aplicació esta estructurada en tres fils d'execució concurrents però no independents *(intercanvien informació)*: un principal i dos fills. El principal és el de la GUI i s'associa unívocament al nivell superior de l'arquitectura del sistema, juntament amb el fil encarregat de mostrar a l'usuari la informació resultant de la simulació. El darrer fil d'execució està lligat als dos primers nivells, encarregats de la captació d'informació, reconstrucció del model i simulació de les físiques dels objectes.

El diagrama de HRT-HOOD [7] representat en la figura 4 defineix formalment el comportament de cada procés i la comunicació entre ells. És imprescindible detallar l'objecte *Control Simulació* ja que és l'únic objecte *actiu* (Veure figura 5).

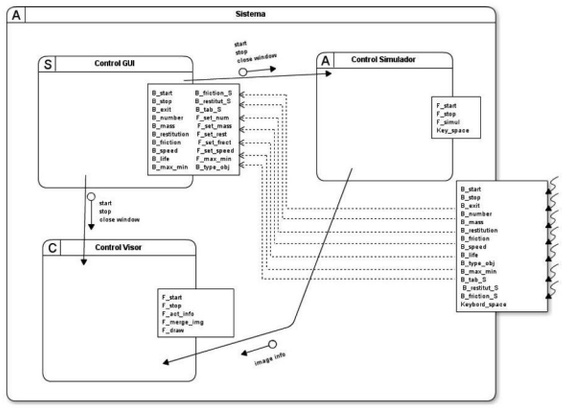


Fig. 4. Diagrama HRT-HOOD del sistema

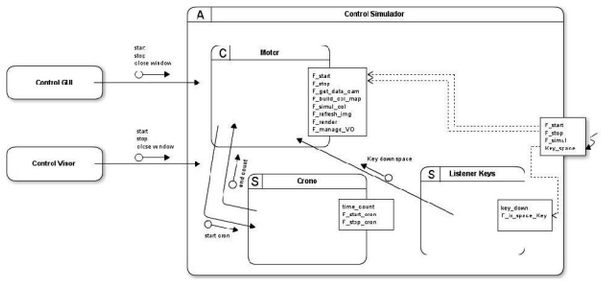


Fig. 5. Diagrama HRT-HOOD de l'objecte Control Simulació

1) Control GUI: És el mòdul destinat a la gestió de la informació d'entrada per part de l'usuari. La interfície disposa d'un seguit de variables que l'usuari pot modificar per ajustar els paràmetres físics dels objectes i/o escena. El control d'aquests paràmetres és la principal funció del modul *Control GUI*.

2) Control Visor: L'objecte visor és cíclic, això vol dir que s'activa cada cert temps per actualitzar les dades provinents de la *Kinect* i fusionar-les amb la projecció de l'escena que prové de l'objecte *Control Simulador*. Finalment mostra aquestes dades a l'usuari.

3) Control Simulador: Aquest es descompon en tres objectes més simples: *Motor*, *Crono* i *Listener keys*. L'objecte *Motor* executa una sèrie de funcions cada cert temps: recull informació de profunditat de la *Kinect*, reconstrueix un model 3D i el de col.lisions, renderitza l'escena i captura la projecció d'aquesta. L'objecte *Listener Keys* detecta la pulsació de la tecla "*space*", llança els objectes virtuals i envia un missatge a l'objecte *Crono* perquè comenci la seva tasca: comptabilitzar temps.

C. Algoritmes

Aquesta secció es centre en l'estudi de com es realitzen les tres tasques bàsiques del sistema de RM: *Captura*, *Tractament* i *Visualització*.

El següent algoritme recull les dades de color i profunditat que la *Kinect* ha capturat i les enregistra en unes estructures de dades (Veure secció IV-D) perquè puguin ser processades.

```
obte_dades(kinect_prof, kinect_color);
PER j=0 FINS kinect_prof.yRes FER
  PER i=0 FINS kinect_prof.XRes FER
    convert_projec_a_3d(XYZ);
    desa_valors([matC,RGB], [matP,XYZ]);
  FI PER;
FI PER;
```

L'algoritme de *Tractament* és un algoritme de generació del model 3D i de la malla de col.lisions dinàmic. Itera sobre les estructures adequades per tal de formar una malla uniforme, composta per un nombre variable de triangles en funció dels objectes virtual que l'usuari desitja integrar en el sistema.

```
PER j=0 FINS yRes-gap FER
  PER i=0 FINS xRes-gap FER
  /* Triangle superor */
    obtenir_punts(val1, val2, val3);
    SI (val1().z AND val2().z
    AND val3().z) /= 0 LLAVORS
      definir_color_model(cromakey);
      definir_punt3D_models(val1, val2, val3);
    FI SI;
    /* Triangle inferior */
    obtenir_punts(val1, val2, val3);
    SI (val1().z AND val2().z
    AND val3().z) /= 0 LLAVORS
      definir_color_model(cromakey);
      definir_punt3D_models(val1, val2, val3);
    FI SI;
  FI PER;
FI PER;
```

El darrer algoritme és el de fusionar els dos mons per conformar l'escena resultant, mescla informació de color de la *Kinect* i la projecció resultant de la simulació.

```
refreca_mapa_color();
SI NO acabar_proces() LLAVORS
  PER i=0 FINS xRes*yRes*3 FER
    SI color(i) /= cromakey LLAVORS
      projecta_valor_simulacio(img);
    FI SI;
  FI PER;
FI SI;
mostra (img);
```

D. Estructures de dades

S'utilitzen dos arrays, un per a informació de posició i l'altre de color, amb una dimensió de $m \cdot n \cdot 3$ posicions, que es corresponen als valors de RGB en el cas de l'array de color i XYZ pel que fa a l'array de posicions. La figura 6 detalla com es desa aquesta informació.

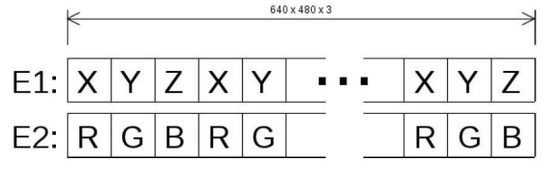
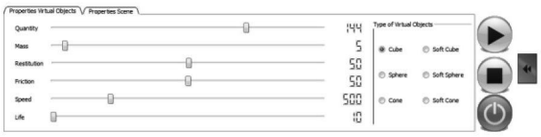


Fig. 6. Estructura de dades 1D amb informació pel model 3D

E. Interfícies d'usuari

El propòsit d'aquesta secció és descriure com els usuaris poden col.laborar amb el sistema, que i com fer-ho.

S'ha disposat d'una interfície d'usuari basada en *Qt* que permet als usuaris la manipulació del sistema mitjançant dispositius d'entrada com el ratolí i el teclat. La GUI consta de dues pipelles amb variables corresponents als paràmetres físics dels objectes virtuals i als de l'escena respectivament. Alhora s'ha dotat d'un botó per començar, un per sortir, un per pausar la simulació i un darrer per ocultar parcialment la interfície a fi de no destorbar la visualització de la simulació (Veure la figura 7).



(a) GUI: Propietats físiques dels objectes virtuals

(b) GUI: Propietats físiques del model de l'escena

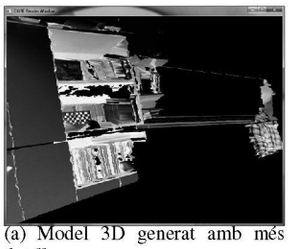
Fig. 7. Interfície d'usuari del sistema

V. Resultats

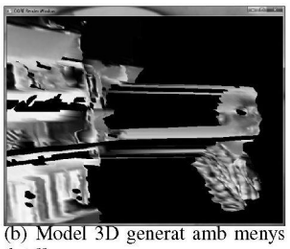
MIRES està basat en un desenvolupament modular, això suposa desglossar l'estructura en blocs més simples que satisfacin uns objectius parcials. Els mòduls d'aquest sistema són:

1) Mòdul de captura.
2) Mòdul de digitalització de l'escena.
3) Mòdul de físiques.
4) Mòdul d'interfície.
5) Integració del tots el mòduls anteriors.

El resultat de la prova de captura i generació del model 3D de l'escena és el que es mostra en la figura 8. Es pot observar en les figures l'eficàcia de l'algoritme dinàmic de modelat.



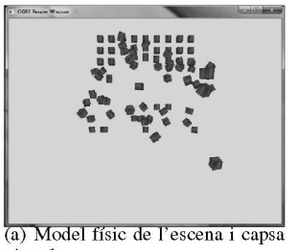
(a) Model 3D generat amb més detall



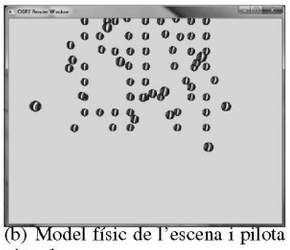
(b) Model 3D generat amb menys detall

Fig. 8. Resultats de les fases de captura i modelització

Els resultats observables de les proves amb les biblioteques de físiques, la simulació de col.lisions, són en la figura 9.



(a) Model físic de l'escena i capsa virtual



(b) Model físic de l'escena i pilota virtual

Fig. 9. Resultat de la simulació de la física

Per acabar la secció mostrar als lectors el resultat de la integració d'ambdós mons: món real i món virtual (Veure figura 10).

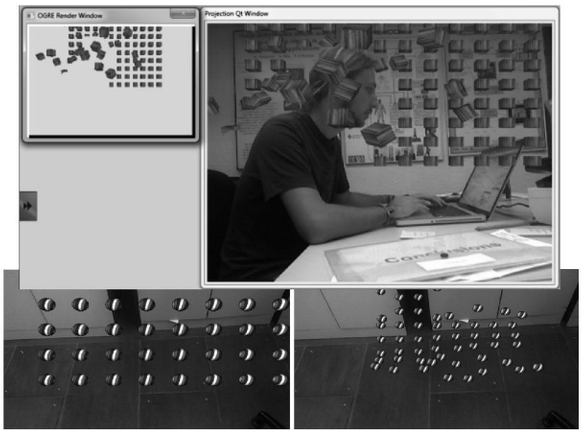


Fig. 10. Captura de l'escena de RM que l'usuari percep

VI. Conclusions

La finalitat del document és donar a conèixer una aplicació de realitat mixta de baix cost basada en eines lliures i el dispositiu de captura RGB-D *Kinect*.

Els resultats de l'aplicació són satisfactoris, per una banda dir que combina objectes reals amb objectes virtuals, existeix interacció 3D, compleix el requeriment de *clipping* dels objectes que no s'haurien de veure. Per altra banda, dir que compleix parcialment el requeriment temporal *(temps real)*, ja que la simulació de la física es veu notablement afectada en funció de la càrrega d'objectes virtuals.

En resum, s'han posat les bases per al desenvolupament d'aplicacions de RM per diversos àmbits: oci, arquitectura, medicina, etc.

Referències

[1] D.W.F. van Krevelen, R. Poelman. A Survey of Augmented Reality Technologies, Applications and Limitations. *Systems Engineering Section, Delft University of Technology, Delft, The Netherlands* (2010).

[2] Miquel Barceló, ¿Cómo funciona la tecnología Kinect?. *BYTE España*, 183 (2012).

[3] DotNetNuke Corporation. OpenNI$_T H$ (2011). http://75.98.78.94/About.aspx

[4] Ogre Team, Open Source 3D Graphics Engine (2009). http://www.ogre3d.org/about

[5] Doxygen. Bullet Collision Detection & Physics Library (2012). http://bulletphysics.com/Bullet/BulletFull/

[6] Nokia Corporation. Cross-platform applicatin and UI framework (2012). http://qt.nokia.com/

[7] R. Di Giovanni, P. L. Iachini. VDM Europe, *HOOD and Z for the development of complex software systems* (1990).

Tomeu Canyelles Escarrer
Enginyer Informàtic, especialitat en Informàtica de Sistemes per la Universitat de les Illes Balears.
bce892@gmail.com

Antoni Jaume-i-Capó
Doctor en Informàtica i professor del departament de ciències matemàtiques i informàtica de la UIB.
antoni.jaume@uib.es