204. Giochi di abilità in 3D: Mastermind e Othello

Rosa Marincola
rosamarincola@virgilio.it

Introduzione

In questo contributo presenterò delle attività realizzate nei mondi virtuali 3D OpenSim: Edmondo e Sim-On-A-Stick (cfr. [1], [2]), con le classi del secondo biennio Sistemi Informativi Aziendali dell'I.I.S. "A. Guarasci" sez. Tecnico Economico di Rogliano (Cs) nell'ambito della sperimentazione d'informatica sulla land Scriptlandia (cfr. [3], [4]).
Si propongono due giochi da tavolo astratti di abilità mentale che rientrano tra le competizioni del Mind Sports Olympiad (cfr. [5]), organizzati annualmente in Inghilterra: Mastermind e Othello.
Entrambi si collocano in un percorso didattico sulla Teoria dei Giochi riportato in parte in un articolo nel numero precedente della rivista (cfr. [6]).
Di Mastermind è stato studiato l'algoritmo risolutivo ed è stata fatta la codifica in LSL (Linden Scripting Language), il linguaggio di programmazione utilizzato nei mondi virtuali (cfr. [7]). Mastermind è un gioco in cui un giocatore, il "decodificatore", deve indovinare la sequenza segreta di cinque colori composta dal "codificatore", nel nostro caso si gioca contro il PC.
Per il secondo gioco, invece è stata riprodotta un'othelliera in 3D di 64 caselle con le rispettive pedine contenti due script ciascuna, uno per cambiare colore (bianco o nero) al click del mouse dei due avatar-giocatori; un altro script posto su un pulsante di avvio che fornisce le regole del gioco ai visitatori.

Figura 1: Prime sperimentazioni del gioco su Scriptlandia, collaborazione online con la classe

Mastermind: dal problema all'algoritmo

Nell'ambito della Teoria dei giochi, Mastermind è classificato come "gioco a informazione imperfetta" cioè gli stati del gioco sono solo parzialmente esplicitati e "di tipo deterministico" poiché gli stati sono determinati unicamente dalle azioni degli agenti.
I passi da compiere sono i seguenti:

- Il codificatore (PC) compone una lista segreta di cinque colori: giallo, verde, blu, arancio, nero.
- Il decodificatore (giocatore) fa il suo primo tentativo, cercando di indovinare il codice. Il codificatore, appena il suo avversario ha completato il tentativo, fornisce degli aiuti comunicando:
  - il numero di colori nella posizione corretta che sono stati individuati;
  - non bisogna comunicare quali colori sono nella posizione corretta.
- Se il decodificatore riesce a indovinare il codice entro il numero di tentativi stabiliti, vince, altrimenti il vincitore è il PC. Nel nostro programma abbiamo previsto sei tentativi.
- In qualsiasi momento è possibile fermare il gioco e conoscere la sequenza corretta.

Per scrivere il programma si sono volute sfruttare le potenzialità dei mondi virtuali, in particolare la chat (il canale zero) tra sfidanti e la simulazione che al click del mouse sul pulsante Start, fa apparire cinque sfere con i colori inseriti dall'utente e al termine con i colori generati in modo casuale dal PC:
1) è stata costruita una prim come tavola del gioco;
2) è stata rezzata una sfera nel cui contenuto è stato inserito il codice di seguito riportato (*), le è stato assegnato il nome "palla" ed è stata presa nell'inventario del proprietario del gioco;
3) è stato costruito un pulsante Start (come in figura 2 e 3), nel cui contenuto è stato inserito il codice di seguito riportato (**) e l'oggetto "palla", trascinato dall'inventario.

Funzionamento:
Al click del mouse sul pulsante Start, lo script genera la sequenza casuale dei numeri da 0 a 4, a cui corrispondono rispettivamente i colori: giallo, verde, blu, arancio e nero. La sequenza delle prime lettere di ciascun colore è memorizzata in una lista denominata "pc" (il linguaggio LSL utilizza essenzialmente solo le liste per i dati strutturati), poi in chat viene visualizzato un messaggio in cui s'invita il giocatore a indovinare la sequenza di colori e a scrivere la sequenza in chat. Dopo aver acquisito in input la sequenza delle iniziali dei colori, questi sono caricati in una nuova lista denominata "utente". Un'istruzione di chiamata attiva la funzione "rezzer" e vengono create dal pulsante Start cinque sferette con i colori indicati in chat dall'utente.
Le due liste sono confrontate, elemento per elemento, e si comunicano all'utente il numero di elementi indovinati e il numero di tentativi restanti.
Il dialogo tra pulsante Start e sfere rezzate avviene sul canale 800. Dopo ogni tentativo al click del mouse sul pulsante, si chiede all'utente mediante un menu a video, se intende continuare il gioco (entro il limite di sei tentativi) e in caso negativo, viene rezzata la sequenza generata dal PC all'inizio del gioco, lo script si resetta dopo 20 secondi e le sfere rezzate spariscono.
Se durante il gioco si superano i 120 secondi d'inattività, il timer fa resettare lo script.

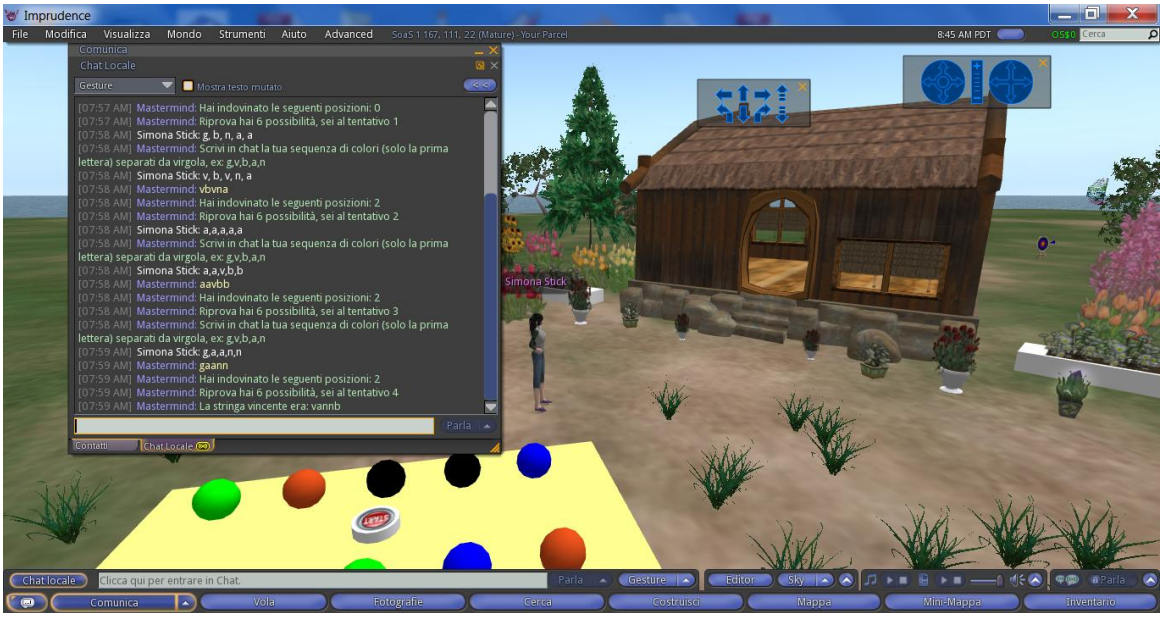

Figura 2: Il gioco avviene tra PC e avatar che dialogano tramite la chat

Il codice LSL di Mastermind

```
//codice da inserire nella sfera rezzata (*)

default
{
    state_entry()
    {
        llListen(800,"",NULL_KEY,"DELETE");
    }

    on_rez(integer param)
    {
            integer j=param;

                if (j == 0) llSetColor(<1.0,1.0,0.0>,ALL_SIDES); // se j=0 la
sfera assume il colore giallo
                if (j == 1) llSetColor(<0.0,1.0,0.0>,ALL_SIDES); // verde
                if (j == 2) llSetColor(<0.0,0,1>,ALL_SIDES); // blu
                if (j == 3) llSetColor(<0.85,0.3,0.1>,ALL_SIDES); // arancio
                if (j == 4) llSetColor(<0,0.0,0>,ALL_SIDES); // nero
    }

    listen(integer channel, string name, key id, string str)
    {
        llDie();
    }
}

////////////////////////////////////////////////////////////////////////////
```

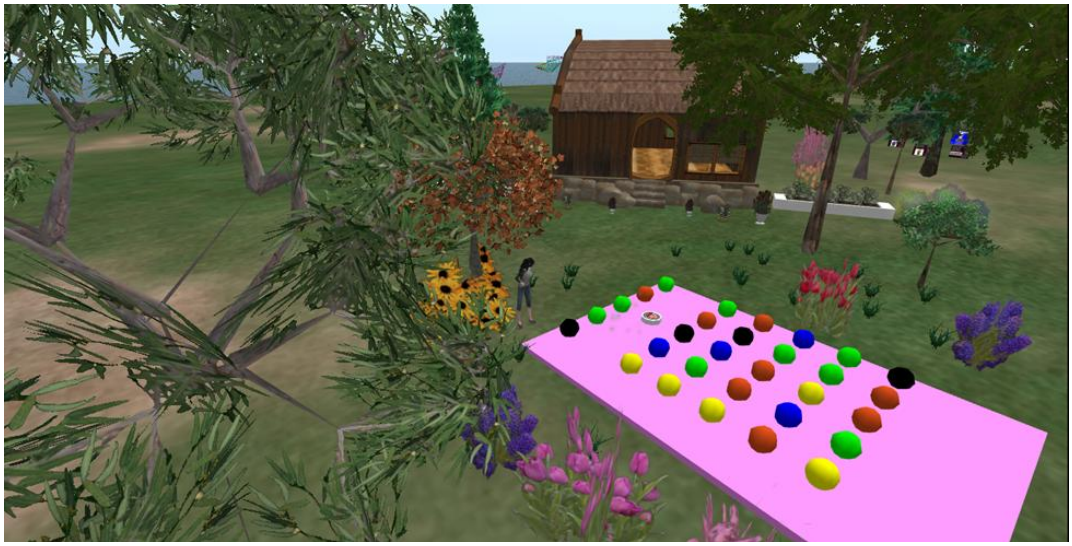

Figura 3: Mastermind su Sim On a Stick, il mondo virtuale OpenSim che s'installa sul proprio PC dove si può lavorare in locale, anche offline.

```
//codice da inserire nel pulsante Start (**)

integer handle;
key id;
integer qhandle=0;
integer punti;
integer k=1;
list pc =[
];
list utente=[
];
string stato;
integer i;
integer x;
string colore;
float TIMEOUT=150;

//la funzione rezzer genera le sfere nel numero e nella posizione indicate dai
parametri per avere la disposizione visualizzata in figura 3
rezzer (integer x, integer i, integer k)
{
    vector pos=llGetPos()+<2-i,k,0>;
    llRezAtRoot("palla",pos,ZERO_VECTOR,ZERO_ROTATION,x);
    return;
}

default
{
    state_entry()
    {
        llSay(0, "Gioca a mastermind");
        llShout(800,"DELETE");
        id=llDetectedKey(0);
        stato="iniziale";
    }

    touch_start(integer count)
    {
        llSetTimerEvent(TIMEOUT);
        if (stato=="iniziale")
            {
```

```
            for(i=0;i<5;i++)
                {
                //viene generata la sequenza casuale di cinque numeri da 0 a 4
                    x=(integer)(llFrand(5));
                  //ad ogni numero viene associato un colore e inserito nella
lista denominata pc
                     if (x==0) colore="g";
                         if (x==1) colore="v";
                         if (x==2) colore="b";
                     if (x==3) colore="a";
                     if (x==4) colore="n";
                     pc+=[colore];
                 }
                //llOwnerSay("Stringa vincente: "+(string)pc);
            //l'istruzione precedente se non commentata consente al proprietario
di leggere la sequenza vincente in chat
            llSay (0, "Clicca sul pulsante per giocare, i colori sono giallo,
verde, blu, arancio, nero");
                stato="primo";
                return;
                }
                    if (stato=="primo")
                {
                llSay(0, "Scrivi in chat la tua sequenza di colori (solo la
prima lettera) separati da virgola, ex: g,v,b,a,n");
                //viene acquisita dalla chat la sequenza scritta dal giocatore
                qhandle=llListen(0,"",llDetectedKey(0),"");
                stato="terzo";
                return;
                }
                    if (stato=="quarto")
                    {
                    handle=llListen(-8,"",id,"");
                    llSetTimerEvent(10);
                //Si apre la finestra di dialogo con il giocatore che può
proseguire il gioco o fermarsi
                    llDialog (llDetectedKey(0),"Vuoi riprovare? Se premi SI poi
clicca di nuovo sul cubo.
Se premi NO il gioco si resetta e vedrai la soluzione.",["SI","NO"],-8);
                        stato="quinto";
                        return;
                    }
    }

    listen(integer channel,string name,key id,string str)
    {
        llListenRemove(qhandle);
        qhandle=0;
        llListenRemove(handle);
        llSetTimerEvent(0);
        if (stato=="terzo")
        {
        //la stringa inserita in chat viene memorizzata in una lista denominate
utente
            utente=llCSV2List(str);
            llOwnerSay((string)utente);
            punti=0;
            for(i=0;i<llGetListLength(pc);i++)
            {
                    integer y;
                    if (llList2String(utente,i)=="g") y=0;
```

```
                    if (llList2String(utente,i)=="v") y=1;
                    if (llList2String(utente,i)=="b") y=2;
                    if (llList2String(utente,i)=="a") y=3;
                    if (llList2String(utente,i)=="n") y=4;
            //istruzione di chiamata della funzione rezzer
                rezzer (y, i, k);
            //confronto tra gli elementi delle due liste e conteggio dei punti
                if (llList2String(pc,i)==llList2String(utente,i)) punti++;}
                llSay (0, "Hai indovinato le seguenti posizioni: "
+(string)punti);
                if (punti==5)
                {
                    llSay (0, "Hai vinto!");
                    llSleep (10);
                    llResetScript();
                }
                    else
                    {
                        llSay (0, "Hai 6 possibilità, sei al tentativo " + k);
                        llSetTimerEvent(TIMEOUT);
                    }
                        stato="quarto";
                        return;
        }
        if (stato=="quinto")
        {
            string risp=str;
            if(risp=="SI" && k<6)
            {
                llSetTimerEvent(TIMEOUT);
                k=k+1;
                stato="primo";
                return;
            }
                else
                {
                    //Visualizzazione delle sfere con la sequenza corretta dei
colori a fine gioco
                    llSay (0,"La stringa vincente era: "+(string)pc);
                    for(i=0;i<llGetListLength(pc);i++)
                    {
                        integer y;
                        if (llList2String(pc,i)=="g") y=0;
                        if (llList2String(pc,i)=="v") y=1;
                        if (llList2String(pc,i)=="b") y=2;
                            if (llList2String(pc,i)=="a") y=3;
                            if (llList2String(pc,i)=="n") y=4;
                        rezzer (y, i, -1);
                    }
                        llSleep (20);
                        llResetScript();
                }
    }
}
timer()
    {
        llListenRemove(handle);
        llSetTimerEvent(0);
        llResetScript();
    }
}
```

Othello: le regole del gioco

Fu inventato verso il 1880 da Lewis Waterman con il nome di Reversi che nel 1882 ne iniziò la produzione presso la Jacques&Son. Othello è stato brevettato nel 1971 da Goro Hasegawa, pochi anni dopo la Ravensburger ne acquisì i diritti. Le regole di Othello e Reversi sono leggermente diverse, entrambi si giocano tra due sfidanti su una scacchiera (othelliera) verde 8x8, su cui si dispongono durante il gioco 64 pedine bicolore: bianco-nero (ciascun giocatore ne ha a disposizione 32), si tratta di un gioco a informazione perfetta.

Le **regole** sono molto semplici da imparare, ma per diventare campioni, occorrono strategia e molta esperienza; è molto praticato in Italia e si disputa in numerosi tornei nazionali (cfr. [8]) e internazionali (cfr. [5]).

1) Prima di iniziare a giocare a Othello si dispongono al centro della scacchiera due pedine bianche e due pedine nere come in figura 4 (per Reversi invece all'inizio del gioco la scacchiera è vuota).
2) Due giocatori scelgono rispettivamente il bianco o il nero e giocano a turno. Inizia a giocare il nero.
3) Durante lo svolgimento del gioco i partecipanti, a turno, pongono una pedina del loro colore adiacente a quelle già presenti sulla scacchiera. Nel nostro caso devono cliccare al centro di una casella apparentemente vuota pe far apparire la pedina ed eventualmente cliccare per renderla del colore desiderato.
4) Chi riesce a chiudere una pedina avversaria tra due pedine proprie (immediatamente contigue a essa sulla verticale, orizzontale o diagonale) clicca e la fa diventare del proprio colore.
5) La partita finisce quando tutte le caselle sono occupate o non ci sono più mosse possibili. Vince chi ha sulla scacchiera più pedine del proprio colore. Una variante prevede la possibilità di chiudere tra pedine proprie più pedine avversarie e di ribaltarle tutte.

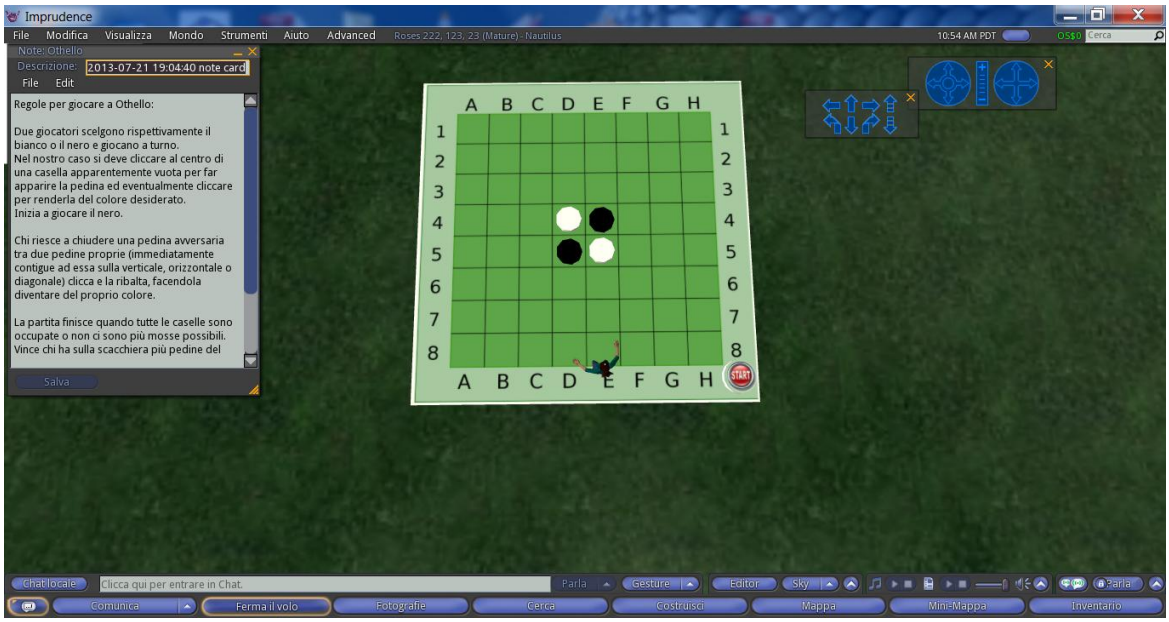

Figura 4: La configurazione iniziale di Othello

Funzionamento:

È stata costruita una othelliera da una primitive cubica a cui sono state date le dimensioni (10 m., 10 m., 0.1 m.) ad essa è stata applicata una texture realizzata con un software per la grafica (nel nostro caso il software open source Gimp).

È stato costruito un pulsante Start da una prim cilindrica, è stato posto in un angolo della scacchiera e nel suo contenuto sono stati inseriti:

- una notecard (file di testo) con le regole del gioco che appaiono a chi clicca sul pulsante;

- lo script di seguito riportato (a) che al click del mouse fa apparire la configurazione iniziale delle pedine e rilascia la notecard.

Sono state rezzate 64 pedine poste ciascuna in una casella della othelliera (riprodotte dalla prima per trascinamento tenendo premuto il tasto Shift), ciascuna contenente due script:

- il primo uguale per tutte le pedine che al click del mouse fa cambiare colore bianco-nero (b);
- il secondo script (c) che al click del mouse sul pulsante Start rende del colore della configurazione iniziale le 4 pedine poste al centro (come in figura 4) e trasparenti, dunque invisibili le restanti 60 pedine (script (d)).

Nota: inserendo in tutte le pedine i codici (b) e (d), è possibile giocare a Reversi.

Il codice LSL per giocare a Othello

```
//Script (a) da inserire nel contenuto del pulsante Start per avere la
configurazione iniziale del gioco con le pedine trasparenti tranne le 4 centrali

default
{
    state_entry()
    {
        llSay(0, "Gioca a Othello");
    }
    touch_start(integer count)
    {
        //al click sul pulsante esso dialoga con le pedine sul canale 300 e
invia la stringa fissata per convenzione "3" con questo messaggio ogni pedina
assume lo stato iniziale
        llSay(300,(string)3);
        //al click viene rilasciata la notecard con le regole, contenuta
nell'inventario dell'oggetto
        llGiveInventory(llDetectedKey(0), llGetInventoryName(INVENTORY_NOTECARD,
0));
    }
}

////////////////////////////////////////////////////////////////////////////

//Script (b) da inserire nel contenuto di ogni pedina per far cambiare colore
bianco o nero al click del mouse e renderla visibile

default
{
    touch_start(integer count)
    {
        llSetColor(<1,1,1>,ALL_SIDES); // al primo click la pedina diventa di
colore bianco su tutti i lati


        //le due istruzioni seguenti possono essere omesse nelle quattro pedine
centrali poiché esse restano sempre visibili giocando a Othello
        float alpha=1;
        llSetLinkAlpha(LINK_SET,alpha,ALL_SIDES);
        state secondo;
    }
}

state secondo
{
    touch_start(integer count)
    {
```

```
        llSetColor(<0,0,0>,ALL_SIDES); // al click successivo la pedina diventa
nera

        //le due istruzioni seguenti possono essere omesse nelle quattro pedine
centrali poiché esse restano sempre visibili
        float alpha=1;
        llSetLinkAlpha(LINK_SET,alpha,ALL_SIDES);
        state default;
    }
}

////////////////////////////////////////////////////////////////////////////

//Secondo script (c) da inserire nelle pedine poste al centro, in particolare in
D4 ed E5 che nella configurazione iniziale sono bianche (figura 4)

default
{
    state_entry()
    {
        //la pedina ascolta sul canale 300
        llListen(300,"",NULL_KEY,"");
    }

    listen(integer channel,string name,key id,string str)
    {
        integer frame=(integer)str;
        //nelle pedine poste in D5 ed E4, l'unica modifica da fare
nell'istruzione seguente è la terna del colore <0,0,0> affinché siano di colore
nero quando si inizia a giocare
        if (frame==3)  llSetColor(<1,1,1>,ALL_SIDES); // quando arriva il
messaggio "3" sul canale 300, cioè qualcuno clicca sul pulsante start, la pedina
diventa di colore bianco
    }
}
////////////////////////////////////////////////////////////////////////////
//Secondo script (d) da inserire nelle pedine che non sono nelle quattro caselle
centrali per renderle invisibili le pedine all'inizio del gioco
default
{
    state_entry()
    {
        //la pedina ascolta sul canale 300
        llListen(300,"",NULL_KEY,"");
    }

    listen(integer channel,string name,key id,string str)
    {
        integer frame=(integer)str;
        // alpha=0; l'oggetto è trasparente,
        // alpha=1; l'oggetto diventa visibile
        if (frame==3)
        {
        // quando arriva il messaggio "3" sul canale 300, cioè qualcuno clicca
sul pulsante start, la pedina diventa trasparente al 100%, quindi invisibile
finché non si clicca sulla casella che occupa.
            float alpha=0;
            llSetLinkAlpha(LINK_SET,alpha,ALL_SIDES);
        }
    }
}
```

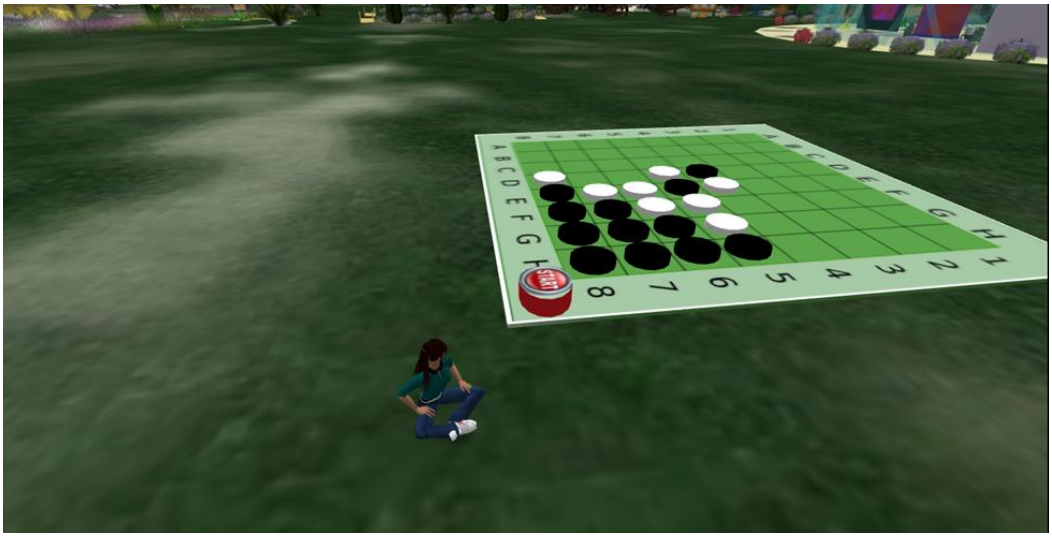

Figura 5: Il gioco è pronto, la sfida comincia!

Conclusioni e ringraziamenti

La costruzione di giochi è un modo coinvolgente per la realizzazione di applicazioni software che inducono alla riflessione, all'analisi di situazioni problematiche e alla produzione di manufatti finalizzati ad elaborare ed affinare strategie, capacità logiche, memoria e spirito di osservazione. Nel primo caso si è pervenuti alla stesura di codici con cui un giocatore può mettere alla prova le proprie capacità mentali contro il PC. Per il secondo gioco, l'obiettivo è stato la realizzazione in 3D di un gioco da tavola astratto dove coppie di studenti o di visitatori possano sfidarsi in singole gare o tornei per utilizzare i modi virtuali in modo altamente interattivo e per scopi didattici interdisciplinari.
Tutti i software, le risorse e le piattaforme utilizzate sono open source e gratuite.

Sitografia essenziale

[1] EdMondo
http://www.scuola-digitale.it/ed-mondo/progetto/perche-edmondo/
[2] OpenSimulator 0.7.5
http://simonastick.com/
[3] Lezioni di scripting in LSL a Scriptlandia
http://www.matematicamente.it/magazine/18dic2012/177marincola-scriptlandia.pdf
[4] Curve algebriche: gioielli virtuali
http://www.matematicamente.it/magazine/19aprile2013/179-Maricola-Curve.pdf
[5] Mind Sports Olympiad
http://it.wikipedia.org/wiki/Mind_Sports_Olympiad#Mind_Sports_Olympiad
[6] Rosa Marincola, 196. Programmare giochi in 3D
Matematicamente.it Magazine N. 21 Gennaio 2014,
[7] LSL Portal
http://wiki.secondlife.com/wiki/LSL_Portal
[8] Federazione Nazionale Gioco Othello
http://www.fngo.it
[9] Gimp
http://www.gimp.org/