Chapitre 5

5. Implémentation et Résultats:

5.1. Introduction

Ce chapitre a pour but de valider expérimentalement, à travers une simulation, le modèle proposé dans le chapitre précédent. Nous disposons d'un système centré sur une caméra représentant l'œil de l'observateur, tel que cette dernière puisse modifier sa position (en se rapprochant ou bien s'éloignant). Pour chaque humain virtuel de la scène, et en fonction de sa position par rapport à la caméra, le système reproduira avec plus ou moins de précision, le comportement approprié à chaque humain virtuel. Le comportement simulé au niveau de notre système, consiste en la construction d'une maison.

5.2. Le langage de programmation

Techniquement, le modèle proposé permettant de simuler le comportement des humains virtuels selon leurs niveaux de détails, ce modèle est réalisé par le biais d'une application implémentée dans le langage de programmation C++.

Nous avons choisi l'environnement de programmation Visuel C++ pour, d'abord la qualité offertes par le C++, telle que la programmation orientée objet et pour l'interface qui pourra nous fournir et surtout car il offre une compatibilité et des capacités d'interfaçage avec le moteur physique « ODE 0.11[1] ». Le moteur ODE étant destinée, dans notre travail, à la réalisation des mouvements physique et du graphisme.

[1] La dernière version du moteur ODE obtenue en 2009.

5.3. Le moteur physique de l'application

L'intégration d'un moteur physique permet à une application de reproduire des aspects physiques réels, on parle alors de simulation.

Le développement d'un « moteur physique » de l'application permet de concentrer tous les appels à ODE en un module. L'utilisation de ce dernier ne requiert donc pas la maîtrise du moteur ODE. Il s'agit, en fait, de réunir dans une classe (ODEInterface) l'ensemble des primitives nécessaires au fonctionnement du moteur ODE.

Ainsi, les objets de l'environnement peuvent être matérialisés par des primitives géométriques existantes sous ODE (GeomBox sous ODE), tous ces objets étant positionnés dans l'environnement de simulation. Il faut, par conséquent, ajouter un plan statique (GeomGround sous ODE) qui constitue le sol de l'environnement. L'espace de collision s'applique sur ces objets, ce qui signifie qu'un corps rigide (la créature où un objet quelconque) ne pourrait en aucun cas les traverser.

Un mouvement physique ne pouvant être continu numériquement, il faut échantillonner une simulation. C'est cette idée qui introduit la notion de pas de simulation : faire un pas de simulation revient à avancer dans le temps la simulation, il faut pour cela définir la taille du pas (généralement 0.05 secondes environ). Plus la taille du pas est petite, plus la simulation sera précise, mais le temps d'exécution sera plus long. Dans notre cas de simulation nous avons choisi un pas de 0,05 secondes.

5.4. L'architecture globale de l'implémentation

Le schéma ci-dessous montre les relations entre les différentes techniques faisant partie du système global implémenté afin d'accomplir le rôle recherché dans ce travail, ce rôle étant la simulation des comportements des humains virtuels, tel que ces comportements possèdent des précisions différentes, selon leurs position par rapport à l'œil de l'observateur, dans un environnement physique réaliste.

Comme nous pouvons le remarquer dans le schéma, notre système est constitué de cinq modules importants :

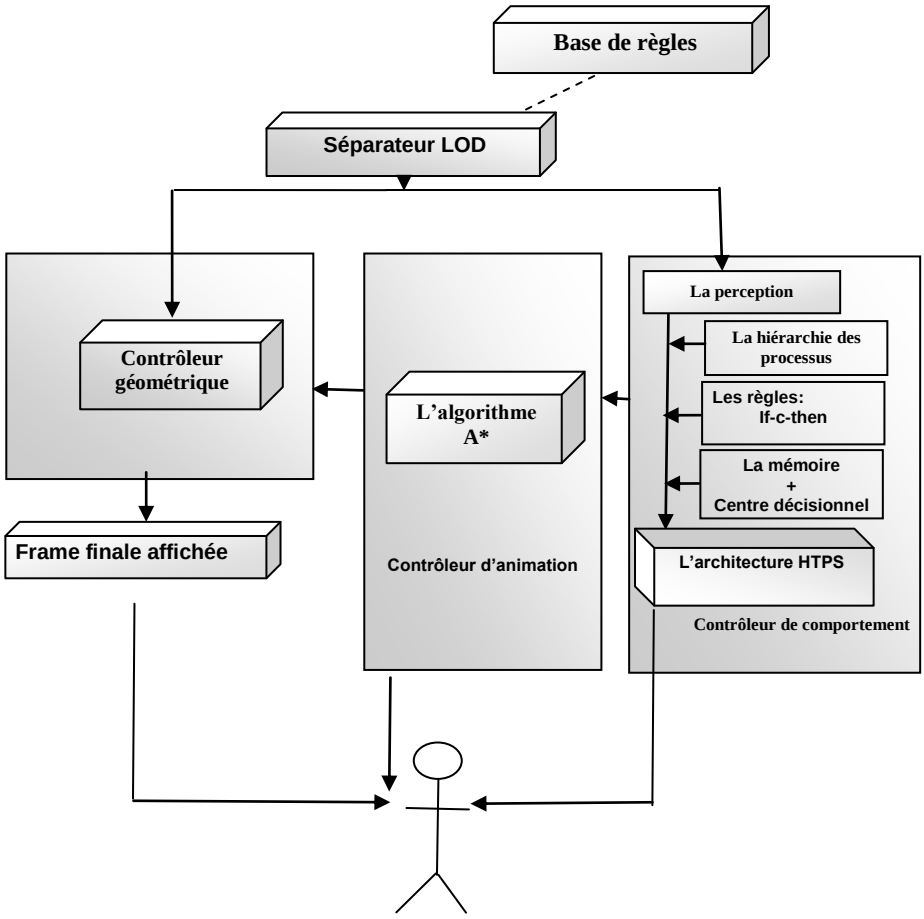


Figure.5.1 : Une vue élevée de l'architecture proposée

5.4.1. La base de règles

L'évaluation de la position d'un humain virtuel par rapport à l'œil de l'observateur peut tenir compte de plusieurs facteurs (voir section 3.2.5). Nous nous sommes limités, dans le cadre de notre travail, à la distance par rapport à l'œil de l'observateur : (Voir Figure 5.2).

- La distance par rapport à l'œil de l'observateur est calculée de manière classique – la distance euclidienne - La distance euclidienne entre le point $A(x_a,y_a,z_a)$ et le point $B(x_b,y_b,z_b)$ est calculée de la manière suivante :

$$AB=BA= \sqrt{(x_b-x_a)^2+(y_b-y_a)^2+(z_b-z_a)^2}$$

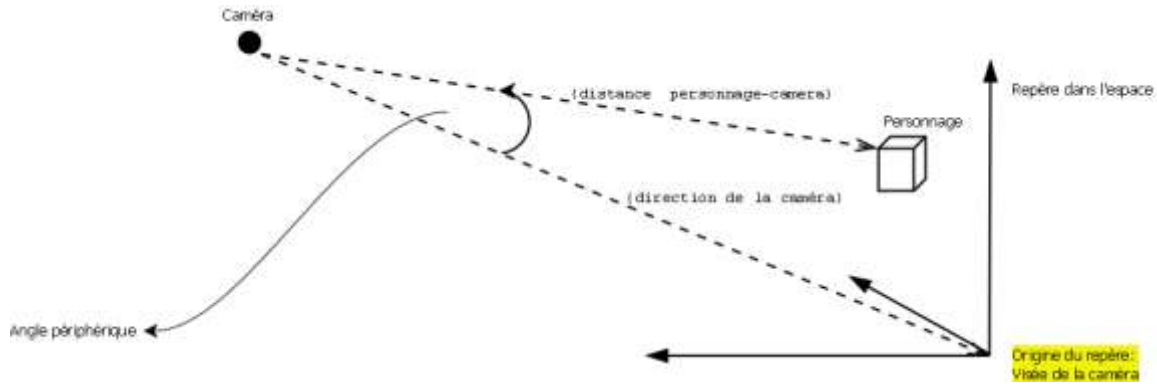


Figure 5.2 : Position par rapport à l'œil de l'observateur

5.4.2. Le séparateur LOD

A partir des données définies dans la base de règle, nous avons déduit trois niveaux de complexité liés à la position de l'œil de l'observateur :

- Niveau 1 : Le comportement de l'humain virtuel est très précis du fait que nous voyons son comportement d'une manière précise (la manière dont il construit la maison est brique par brique). (LOD_value=3)
- Niveau 2 : La visibilité du comportement de l'humain virtuel est moins précise (nous voyons la construction de la maison se faire ligne par ligne). (LOD_value=2)
- Niveau 3 : le comportement de l'humain virtuel est non visible. (LOD_value=1)

Nous définissons par la suite deux variables d_1 et d_2 de la manière suivante (voir Figure5.3) :

- d_1 : Tout humain virtuel situé dans le champ de vision à une distance inférieure ou égale à d_1 possède une position de niveau 1 (LOD_value = 3) ;
- d_2: Tout humain virtuel situé dans le champ de vision à une distance comprise entre d_1 et d_2 possède une position de niveau 2 (LOD_value = 2) ;
- Tout humain virtuel situé dans le champ de vision à une distance supérieur à d_2 possède une position de niveau 3 (LOD_value = 1) ;

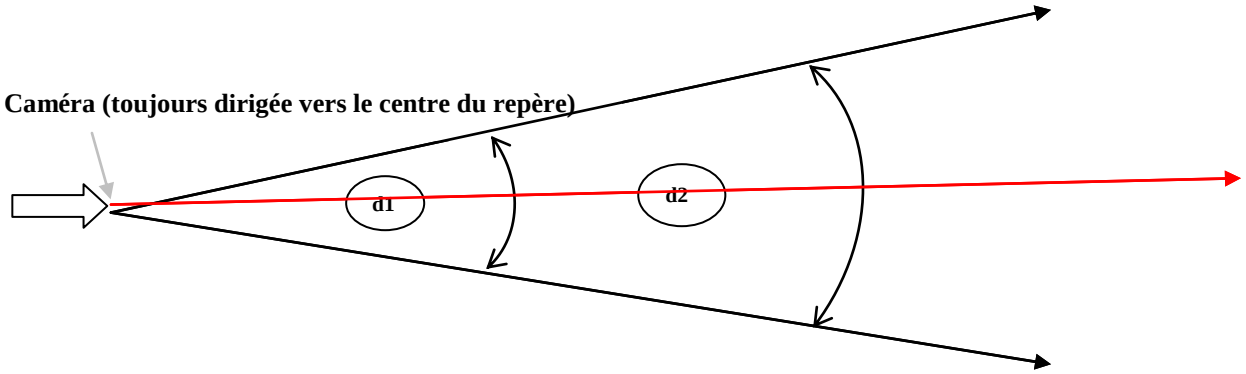


Figure.5.3 : Définition des niveaux de détails

5.4.3. Le contrôleur de comportement

Le comportement recensé est le comportement de construction d'une maison:

- A un niveau très proche de l'œil de l'observateur, la construction de la maison est effectuée brique par brique.
- A un niveau moins proche, la construction de la maison est effectuée ligne[2] par ligne.
- A un niveau très loin, le comportement est invisible

Le comportement de la construction consiste en les actions élémentaires suivantes :

- Déplacer le bras
- Eviter l'obstacle
- Prendre un objet[3]
- Déplacer l'objet
- Eviter l'obstacle
- Mettre l'objet

Nous pouvons présenter ce comportement par l'automate de déplacement d'un objet en utilisant l'architecture HPTS, présenté dans la figure suivante :

[2] La ligne est composée de trois briques et chaque mur se compose de trois lignes
[3] Dans notre simulation, un objet est représenté par une brique

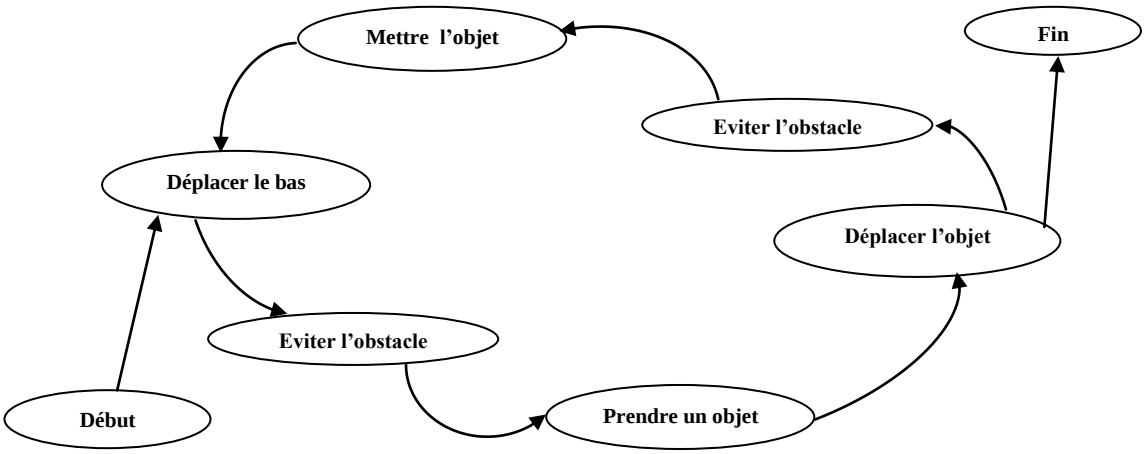


Figure 5.4 : L'architecture HPTS utilisée

5.4.4. Contrôleur de mouvement

D'après l'automate présenté, le mouvement que nous devons simuler est le déplacement d'un bras ainsi qu'un déplacement d'une brique. Le niveau de précision de la simulation de ce mouvement est limité par la position du bras par rapport à l'œil de l'observateur, nous pouvons définir les niveaux suivants :

- Niveau 1 : où le bras est très proche de l'œil de l'observateur. La simulation du mouvement est réalisée en précision.
- Niveau 2 : le mouvement est moins proche de l'œil de l'observateur et dans ce cas le mouvement sera simulé avec moins de précision.
- Niveau 3 : le comportement est invisible, par conséquent le mouvement est également invisible.

Un déplacement continu pour chaque humain virtuel, couterait beaucoup de ressources informatiques. Il faudrait alors effectuer des micro-translations répétitives (de l'ordre d'une translation tous les centièmes de seconde) pour que l'utilisateur ait l'illusion voulue. Ces opérations surchargent le processeur et ralentissent par conséquent la vitesse de la simulation. Notre idée est de réduire la fréquence et d'augmenter le pas de translation au fur et à mesure que la caméra s'éloigne de l'humain virtuel (ou vice versa) tout en maintenant la vitesse de déplacement - pour ne pas changer les données de la simulation. (Voir tableau 5.1)

Niveau	(fréquence, pas)
1	($f_1=f_{max}$, $step_1=step_{min}$)
i (i>1)	(f_i, $step_i$)

Tableau.5.1 : Tableau des paramètres de la fonction de déplacement.

$\forall$ i > 1, $f_i < f_{max}$ et $f_{i-1} < f_i$
$\forall$ i > 1, $step_i > step_{min}$ et $step_{i-1} > step_i$

f_{max} et $step_{min}$ sont les valeurs qui produisent le déplacement le plus crédible à l'œil de l'observateur.

5.4.5. Contrôleur géométrique

Comme nous avons cité précédemment, nous avons déduit 03 niveaux de complexité liés à la position de l'œil de l'observateur:

– Niveau 1 : Le personnage est entièrement visible avec précision.

– Niveau 2 : Le personnage est partiellement visible.

– Niveau 3 : Le personnage est invisible.

Remarque : Nous avons limité l'application de la technique de LOD des données à la représentation de la géométrie des humains virtuels seulement parce que c'est l'élément géométrique existant le plus complexe et qui nécessite beaucoup de ressources informatiques, par contre les autres éléments géométriques (la table et les cubes) ont une représentation unifiée pour toutes les niveaux à cause de leurs simplicité géométrique.

5.4.5.1. Modèle proposé de l'humain virtuel à un niveau très précis

A un niveau très précis, le modèle de la représentation d'un humain virtuel utilisé est vu en tant qu'une représentation d'un modèle d'humain virtuel générique. Il a été créé à partir de primitives de corps et de joints disponibles sur ODE. Le modèle d'humain virtuel se compose de 12 primitives géométriques de corps rigide, et 13 joints : 8 Hinge joint, 2 Ball and socket joint, 2 contact joint et 1 fixed joint. Ces joints reliant les primitives du corps dans une structure articulée de corps-rigide.

Les primitives géométriques utilisées sont : 8 Capped cylinders, 3 parallélépipèdes et une sphère. La structure de l'humain virtuel est définie en utilisant les chaînes multiples, à partir de ses pieds avec chaque lien décrit en termes de liens précédents. Cette composition a comme conséquence un modèle d'humain virtuel possédant 15 degrés de liberté (DOF). La

structure du corps rigide de notre humain virtuel qui est simulé, avec l'organisation des liens entre les différentes primitives, est représentée dans la figure 5.5 et dans le tableau 5.2

Comme motionné dans le tableau 5.2, les types de jointures qui sont utilisées dans notre modèle d'humanoïde sont : hinge joint, ball & socket joint, Fixed joint et contact joint[4].

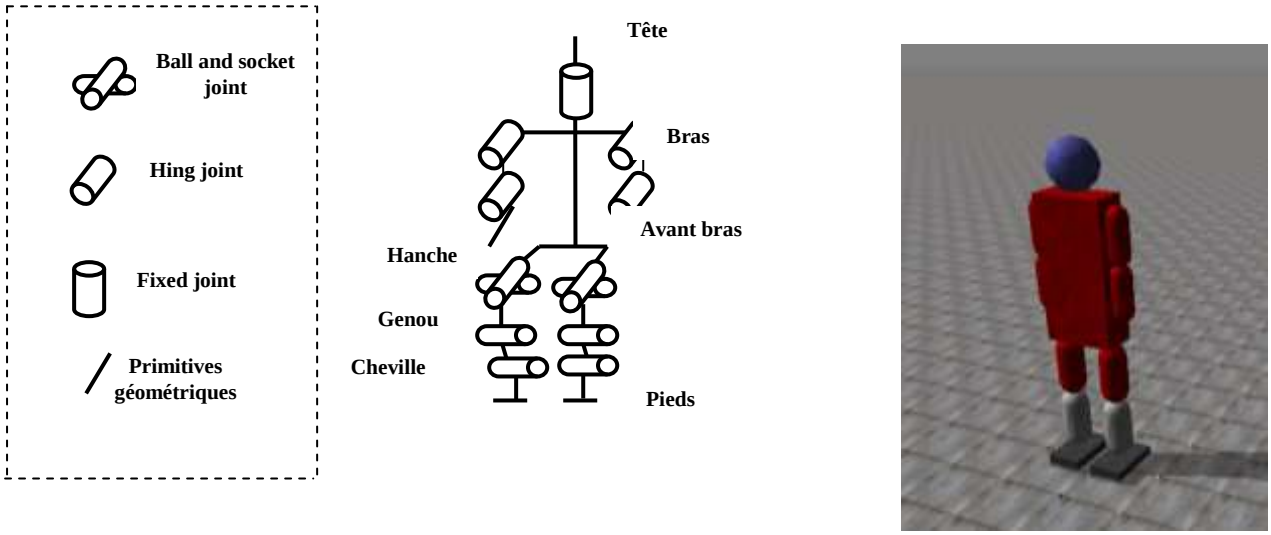


Figure 5.5: Modèle de l'humain virtuel simulé par ODE à un niveau très précis.

Parties de corps	Jointure	Primitive géométrique
Tête	Fixed joint	Sphère
Bras	Hing joint + Contact joint	Capped cylinder
Hanche	Ball and socket joint	Parallélépipède
Genou	Hing joint	Capped cylinder
Cheville	Hing joint	Capped cylinder
Pieds	/	Parallélépipèdes

Tableau 5.2: Paramètres de simulation de l'humain virtuel à un niveau très précis.

5.4.5.2. Modèle proposé de l'humain virtuel à un niveau moins précis

A un niveau moins précis, le modèle d'humain virtuel se compose de 08 primitives géométriques de corps rigide, et 09 joints : 04 Hinge joint, 2 Ball and socket joint, 2 contact joint et 1 fixed joint. Ces joints reliant les primitives du corps dans une structure articulée de corps-rigide.

[4] *Hinge*, *Ball and socket*, *contact* et *fixed* joints sont les noms des types de jointures existantes dans ODE.

Les primitives géométriques utilisées sont : 04 Capped cylinders, 3 parallélépipèdes et une sphère. La structure de l'humain virtuel est définie en utilisant les chaînes multiples, à partir de ses pieds avec chaque lien décrit en termes de liens précédents. La structure du corps rigide de notre humain virtuel qui est simulé, avec l'organisation des liens entre les différentes primitives, est représentée dans la figure 5.6 et dans le tableau 5.3

Comme motionné dans le tableau 5.3, les types de jointures qui sont utilisées dans notre modèle d'humanoïde sont : hinge joint, ball & socket joint, Fixed joint et contact joint[5].

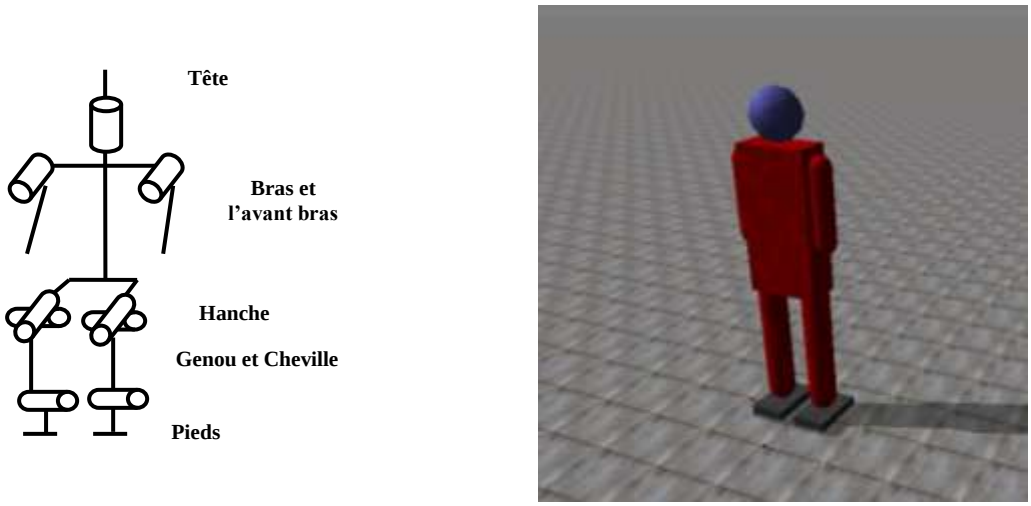


Figure 5.6: Modèle de l'humain virtuel simulé par ODE à un niveau moins précis.

Parties de corps	Jointure	Primitive géométrique
Tête	Fixed joint	Sphère
Bras et l'avant bras	Hing joint + Contact joint	Capped cylinder
Hanche	Ball and socket joint	Parallélépipède
Genou et cheville	Hing joint	Capped cylinder
Pieds	/	Parallélépipède

Tableau 5.3: Paramètres de simulation de l'humain virtuel à un niveau moins précis.

5.5. Algorithmes utilisés

L'application réalisée, dans le cadre de notre projet, est basée sur deux concepts très importants. En premier étant la simulation graphique tridimensionnelle et physique des humains virtuels, de l'environnement et les objets statiques (i.e. utilisation d'un simulateur

[5] *Hinge*, *Ball and socket*, *contact* et *fixed* joints sont les noms des types de jointures existantes dans ODE.

physique qui est ODE) alors que le second aspect est celui de l'implémentation de l'architecture proposée, détaillée ci-dessus, qui se compose de cinq modules importants :

Le séparateur_LOD, le contrôleur de comportement, le contrôleur de mouvement et le contrôleur géométrique et la base de règles.

Tous ces modules sont des fonctions appartenant chacune à une classe spécifique. Ces fonctions sont exploitées par une fonction principale, cette fonction est : simLoop[6] .

5.5.1. Structures de données

La hiérarchie des processus est implémentée sous la forme d'une structure de données composée de trois champs (voir la structure de données1) :

- Le nom de l'action qui peut être : Construire une maison, construire une ligne ou construire une brique.
- L'état de l'action qui peut être atomique ou non atomique.
- Les sous_intentions[7] de l'action par exemple la sous_intention de l'action « construire une maison » est « construire une ligne » et la sous_intention de l'action construire une ligne est construire une brique.

Structure de données 1 : La hiérarchie des processus

```
typedef struct hierarchie_processus hierarchie_processus;
struct hierarchie_processus
{
      const char nom[100];
      const char etat[100];
      const char sous_intentions[100];
};
hierarchie_processus action[3]={
  "construire une maison","non atomique","construire une ligne",
  "construire une ligne","non atomique","construire une brique",
  "construire une brique","atomique",""};
```

Les intentions des actions sont implémentées sous la forme d'une structure de données composée d'un seul champ qui est le nom de l'intention, tel que nous avons initialisés l'ensemble de nos intentions par l'intention « construire une maison » seulement (voir la structure de données 2).

[6] La fonction de la boucle de simulation qui appartient à la classe mère, cette fonction permet de gérer la boucle en prenant en charge les différents modules de notre architecture.

[7] Une intention de l'action est le but ou l'objectif de cette action

Structure de données 2 : Les intentions des actions

```
typedef struct intention intention;
struct intention
{
        char nom[100];
};
intention Intention[3]={"construire une maison","",""};
```

5.5.2. Fonctions

Les valeurs de niveaux de détails des objets[8] (LOD_value) sont définies à l'aide de la fonction 1[9] et le comportement approprié pour chaque humain virtuel selon son niveau de détail est définie à l'aide de la fonction2.

Fonction 1 : Le séparateur LOD

```
// le séparateur LOD
static int definir_LOD_value(dGeomID table_geom)
{
 const dReal *position =dGeomGetPosition(table_geom);
double distance =
distanceeuclidienne(position[0],position[1],position[2],xyz[0],xyz[1],xyz[2]
);
if(distance >=0 && distance<d1)   return 3;
if(distance >=d1 && distance <d2) return 2;
if (distance>=d2) return 1;
}
```

Remarque 1 : d_1 et d_2 sont des valeurs définies par l'utilisateur.

Remarque 2 : la « *distanceeuclidienne* » est une fonction programmée par l'utilisateur sert à calculer la distance euclidienne entre deux points.

[8] Un objet peut être un humain virtuel ou un objet statique comme un cube ou une table
[9] La fonction 1 sert à définir la valeur de niveau de détail pour un objet

Fonction 2 : Le contrôleur de comportement

```
static void controleur_comportement(dGeomID body_geom)
{
      while(strcmp(Intention[j].nom,"") !=0)
      {
       for(i=0;i<3;i++)
        {
         if(strcmp(action[i].nom,Intention[j].nom) == 0) {

           if((strcmp(action[i].etat,"atomique")==0) ||
(definir_LOD_complex(action[i].nom)==definir_LOD_value(body_geom)))
                      {
            if (strcmp(action[i].nom,"construire une maison")==0)
      {Appel à la fonction de construction une maison à un niveau 3;}
             if (strcmp(action[i].nom,"construire une ligne")==0)
      {Appel à la fonction de construction une maison à un niveau 2 ;}


            if (strcmp(action[i].nom,"construire une brique")==0)
      {Appel à la fnction de construction une maison à un niveau 1;}
                    }
                        i=3;
                        j=j+1;
                       strcpy(Intention[j].nom,"");
            }
           else
           {
                 j=j+1;
                 strcpy(Intention[j].nom,action[i].sous_intentions);
            } }
```

Remarque 3 : Les deux fonctions « strcmp » et « strcpy » sont des fonctions prédéfinies dans le langage de programmation C++, tel que :

- La fonction « strcmp » est utilisée pour comparer deux chaines de caractères.
- La fonction « strcpy » est utilisée pour copier une chaine de caractère dans une autre chaine de caractère.

Remarque 4 : Les deux fonctions « definir_LOD_complex » et « definir_LOD_value » sont deux fonctions programmées par le concepteur, tel que :

- La fonction « definir_LOD_value » est utilisée pour définir la valeur de niveau de détail d'un objet.
- La fonction « definir_LOD_complex » est utilisée pour définir la valeur de la complexité d'un processus.

Remarque 5 : Les fonctions présentées ci-dessus ont été utilisées pour la simulation du comportement de trois humains virtuels.

5.5.3. Exemple d'une simulation sous ODE

La troisième fonction présentée dans ce chapitre est celle de la modélisation d'un objet de l'environnement qui est représenté physiquement, cet objet étant une table carrée possédant quatre supports.

Nous avons choisi de représenter cette fonction dans le but d'illustrer le fonctionnement de l'environnement ODE ainsi que la manière dont nous pouvons l'exploiter pour simuler un objet.

Fonction 3 : La construction d'un objet 3D

```
static void creertable
(dBodyID table_bodies[1000],dGeomID table_geom[1000],int TB,int TG,double d)
{
    // le haut de la table
      table_bodies[TB] = dBodyCreate (world);
      dBodySetPosition (table_bodies[TB],0,1.8 - d ,2.5);
      dMassSetBox (&m,1,6,3,0.2);
      dMassAdjust (&m,BMASS);
      dBodySetMass (table_bodies[TB],&m);
      table_geom[TG] = dCreateBox (space,6,3,0.2);
      dGeomSetBody (table_geom[TG],table_bodies[TB]);

      TB +=1;
      TG +=1;

      // pied1 de la table
      table_bodies[TB] = dBodyCreate (world);
      dBodySetPosition (table_bodies[TB],2.2,2.5 - d ,1.25);
      dMassSetBox (&m,1,0.2,0.2,2.5);
      dMassAdjust (&m,BMASS);
      dBodySetMass (table_bodies[TB],&m);
      table_geom[TG] = dCreateBox (space,0.2,0.2,2.5);
      dGeomSetBody (table_geom[TG],table_bodies[TB]);

      TB +=1;
      TG +=1;

      // pied2 de la table
      table_bodies[TB] = dBodyCreate (world);
      dBodySetPosition (table_bodies[TB],2.2,0.7 - d ,1.25);
      dMassSetBox (&m,1,0.2,0.2,2.5);
      dMassAdjust (&m,BMASS);
      dBodySetMass (table_bodies[TB],&m);
      table_geom[TG] = dCreateBox (space,0.2,0.2,2.5);
      dGeomSetBody (table_geom[TG],table_bodies[TB]);
```

```
TB +=1;
TG +=1;

// pied3 de la table
table_bodies[TB] = dBodyCreate (world);
dBodySetPosition (table_bodies[TB],-1.8,0.7 - d ,1.25);
dMassSetBox (&m,1,0.2,0.2,2.5);
dMassAdjust (&m,BMASS);
dBodySetMass (table_bodies[TB],&m);
table_geom[TG] = dCreateBox (space,0.2,0.2,2.5);
dGeomSetBody (table_geom[TG],table_bodies[TB]);

TB +=1;
TG +=1;

// pied4 de la table
table_bodies[TB] = dBodyCreate (world);
dBodySetPosition (table_bodies[TB],-1.8,2.5 - d ,1.25);
dMassSetBox (&m,1,0.2,0.2,2.5);
dMassAdjust (&m,BMASS);
dBodySetMass (table_bodies[TB],&m);
table_geom[TG] = dCreateBox (space,0.2,0.2,2.5);
dGeomSetBody (table_geom[TG],table_bodies[TB]);

TB +=1;
TG +=1;}
```

5.6. Expérimentations

Le but de notre application est d'exploiter la technique de niveaux de détails dans la simulation crédible et en temps réel du comportement des humains virtuels dans un environnement physique réaliste. Pour ce faire, nous avons utilisé la technique de LOD IA.

Il serait intéressant, non seulement d'observer les humains virtuels en pleine action, mais aussi, de mesurer les performances de l'exécution de notre application avec et sans la mise en place de la technique de LOD IA. L'objectif est d'arriver à montrer au moins les trois propriétés suivantes :

- Il est possible d'observer la précision du comportement des humains virtuels selon leurs position par rapport à l'œil de l'observateur (très précis, moins précis et invisible).
- La capacité de notre système à changer la précision du comportement de nos humains virtuels une fois que la caméra change de position : à partir d'un comportement très précis, nous pouvons avoir un comportement moins précis et à partir d'un comportement invisible, nous pouvons avoir un comportement moins précis.

- Tous les comportements, quelque soient leurs niveaux de précision, doivent commencer en même temps et doivent se terminer en même temps, même si un changement au niveau de la position de la caméra est effectué.

Les tests ont été développés sur un poste muni de deux processeurs et équipé d'une carte graphique moyenne.

5.6.1. Mise en œuvre de la simulation

Les expérimentations reposent sur une simulation d'un environnement physique réaliste tridimensionnel qui contient certains objets statiques (des tables simples et des cubes) et des humains virtuels. Notre environnement simulé est centré sur une caméra représentant l'œil de l'observateur, tel que cette dernière puisse modifier sa position (en se rapprochant ou bien en s'éloignant). Pour chaque humain virtuel de la scène, et en fonction de sa position par rapport à la caméra, le système reproduira avec plus ou moins de précision, le comportement approprié à chaque humain virtuel.

Le comportement simulé au niveau de notre système, consiste en la construction d'une maison simple. Comme nous l'avons cité précédemment, nous avons trois niveaux de complexité liés à la position de l'œil de l'observateur :

- Niveau 1 : Le comportement de l'humain virtuel est très précis du fait que nous voyons son comportement d'une manière précise, le comportement de la construction de la maison est effectué brique par brique ;
- Niveau 2 : La visibilité du comportement de l'humain virtuel est moins précise. Le comportement de la construction est effectué ligne par ligne ;
- Niveau 3: le comportement de l'humain virtuel est invisible.

5.6.1.1. L'environnement virtuel

L'environnement virtuel de la simulation est un espace plat simple ne possédant aucune dénivellation. Il s'agit donc d'une surface non bornée sur laquelle nous avons positionné des objets statiques (des cubes superposés sur des tables) et des humains virtuels avec des positions fixes et ceci pour étudier le comportement de construction d'une maison selon plusieurs niveaux de détails. Cet environnement possède toutes les caractéristiques physiques nécessaires pour réaliser une simulation relativement réaliste.

5.6.1.2. Objets statiques et humains virtuels

Du moment que l'environnement simulé possède les caractéristiques physiques nécessaires à la procédure de la simulation, les humains virtuels doivent commencer le comportement de construction approprié, selon leur position par rapport à l'œil de l'observateur.

Les humains virtuels effectuent leurs constructions à l'aide des cubes qui représentent des briques, ces derniers sont superposés sur des tables (voir Figure 5.7).

Les humains virtuels, comme mentionnés précédemment, selon leurs positions par rapport à l'œil de l'observateur possèdent une représentation plus ou moins précise (Voir figure.5.8).

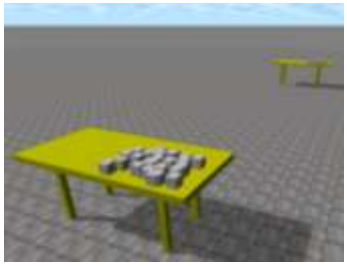


Figure 5.7 :L'environnement simulé avec des tables et des cubes en trois niveaux de détails

Figure 5.8 :L'environnement simulé avec des humains virtuels en trois niveaux de détails

Les figures ci-dessus nous montrent deux humains virtuels seulement, parce que le troisième humain virtuel est invisible .Comme il est illustré dans la figure 5.8, nous avons :

- Un humain virtuel simulé avec une précision parce qu'il est très proche de la caméra, il est situé en premier niveau.
- Le deuxième humain virtuel est simulé avec plus ou moins précision parce qu'il est situé en deuxième niveau de détail.
- Le troisième humain virtuel n'apparait pas parce qu'il est très loin de l'œil de l'observateur, il est situé en troisième niveau de détail.

Au niveau de la figure 5.7, nous avons trois tables mais il n'apparait que deux tables, parce que la troisième table est invisible, elle se situe en troisième niveau de détail.

5.6.2. Conditions initiales de l'application

L'application est réalisée sous certaines conditions qui sont nécessaires au déroulement de la simulation. Ainsi, nous avons définis tous les paramètres ainsi que les valeurs initiales avant le commencement de la simulation.

Nous avons simulé le comportement de construction d'une maison par des humains virtuels, chaque humain virtuel est muni d'un ensemble de cubes superposés sur une table.

Le nombre de cubes est fixé à 24 cubes alors que la maison simulée sera composée de quatre murs, chaque mur étant constitué de trois lignes et chaque ligne composée par trois briques[10] .

Les niveaux de détails ont été définis sur la base des distances d_1 et d_2 tels que :

- Chaque objet[11] se trouve à une distance inférieure ou égale à d_1, est au niveau 1 ;
- Chaque objet se trouve à une distance comprise entre d_1 et d_2, est au niveau 2 ;
- Chaque objet se trouve à une distance supérieure à d_2, est au niveau 3.

Les distances d_1 et d_2 ont été fixées respectivement à 30 et 60.

Le pas de déplacement des bras des humains virtuels a été fixé, selon le cas:

- Pour un niveau très précis à 1/30 pixels.
- Pour un niveau moins précis 1/10 pixels.

Par ailleurs, la caméra, peut être rapprochée ou éloignée selon le besoin.

Le comportement de construction d'une maison est simulé avec plus au moins précision, selon le niveau de détail de chaque humain virtuel :

- Si l'humain virtuel se trouve au niveau 1 : le comportement est effectué d'une manière très précise (brique par brique) ;
- Si l'humain virtuel se trouve au niveau 2 : le comportement est effectué d'une manière moins précise (ligne par ligne) ;
- Si l'humain virtuel se trouve au niveau 3 : le comportement sera invisible.

5.7. Résultats obtenus

Pour ce qui est des expérimentations, nous nous sommes attelés à évaluer notre système par rapport aux buts que nous nous sommes assignés précédemment.

Au niveau de notre système, le nombre d'humains virtuels est fixé à 3, 6 puis 10 humains. Nous avons ainsi procédé au test de l'application pour chacun des cas précédents.

5.7.1. Cas où le nombre d'humains virtuels est fixé à 03

A chaque niveau de détail, nous trouvons un humain virtuel équipé avec des cubes superposés sur une table (Voir figure.5.9 -5.10 -5.11 -5.12) :

[10] Une brique est représentée par un cube
[11] Un objet peut être un humain virtuel, un cube ou une table

- Le niveau1 : le premier humain virtuel se trouve à la position (x=0, y=0). Son comportement est simulé avec précision, il construit la maison brique par brique
- Le niveau 2 : le deuxième humain virtuel situé à la position (x=0, y=30). Son comportement est simulé avec moins de détail, il construit la maison ligne par ligne.
- Le niveau 3 : le troisième humain virtuel se trouve à la position (x=0, y=60). Son comportement est invisible.

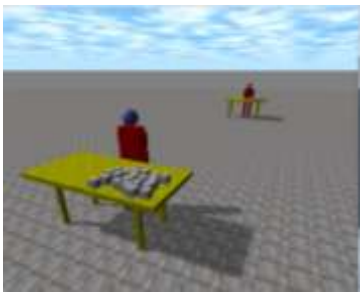


Figure 5.9. Le début de la construction

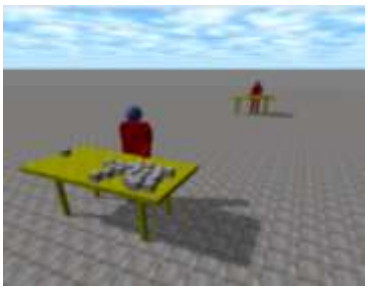


Figure 5.10 : Les humains virtuels ayant commencé la construction

Figure 5.11 : Les humains virtuels ayant construit le premier mur.

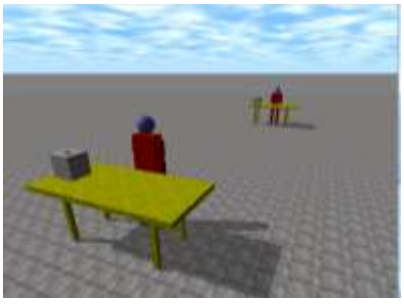
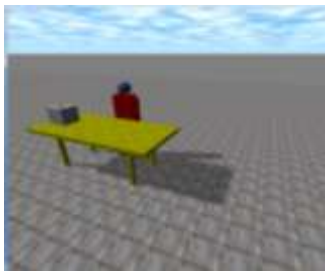


Figure 5.12 : Les humains virtuels ayant terminé la construction

Selon les figures ci-dessus, nous pouvons constater que le troisième humain virtuel n'apparait pas, parce qu'il appartient au troisième (le dernier) niveau de détail par contre le comportement des deux premiers humains virtuels apparaît avec plus ou moins de précision selon leurs niveau de LOD.

Un changement[12] au niveau de la position de la caméra, nous a permis de voir le comportement du troisième humain virtuel avec précision[13] parce qu'il est devenu très proche de la caméra (Voir figure.5.13).

[12] Nous avons effectué un rapprochement de la position de la caméra, en appuyant sur la touche A
[13] Il est situé en premier niveau de détail



Figue 5.13 : Changement de la position de la caméra est effectué, tel que l'humain virtuel, qui était invisible est devenu plus précis.

5.5.1. Cas où le nombre d'humains virtuels est fixé à 06

Dans ce cas, le nombre d'humains virtuels est fixé à 6 (Voir figure.5.14 -5.15 -5.16) :

- Pour le niveau 1, nous avons un humain virtuel situé à la position (x=0, y=5).
- Pour le niveau 2, nous avons quatre humains virtuels situés respectivement aux positions (x=2, y=30), (x=-35, y=35), (x=-20, y=40) et (x=-8, y=45).
- Pour le niveau 3, nous avons un humain virtuel situé à la position (x=-8, y=60).

Figure 5.14 : Humains virtuels ayant commencé la construction

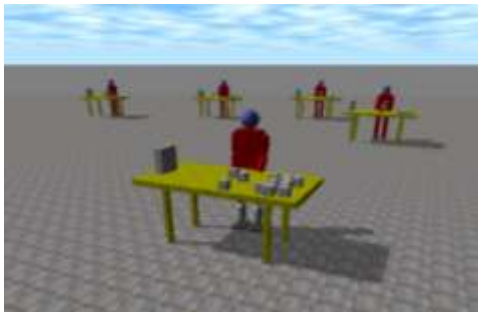


Figure 5.15 : Humains virtuels ayant construit le premier mur.

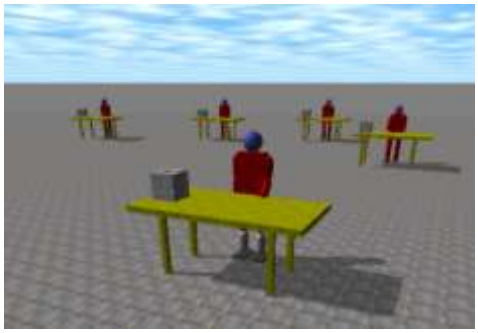
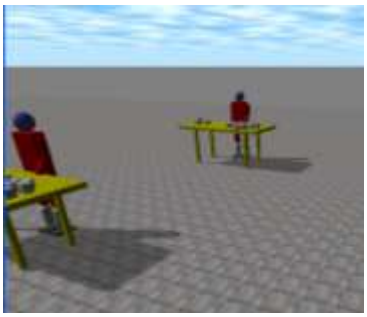


Figure 5.16 : Humains virtuels ayant terminé la construction.

Selon les figures ci-dessus, nous pouvons constater que le dernier (le sixième) humain virtuel n'apparait pas, parce qu'il appartient au troisième (le dernier) niveau de détail par contre le comportement des cinq premiers humains virtuels apparaît avec plus ou moins de précisions selon leurs niveau de détails.



Figue 5.17 : Changement de la position de la caméra est effectué, tel que l'humain virtuel, qui a été invisible est devenu très précis

Un rapprochement au niveau de la position de la caméra, nous a permis de voir le comportement du sixième humain virtuel avec précision (Voir figure.5.17).

5.5.2. Cas où le nombre d'humains virtuels est fixé à 10

Dans ce cas, le nombre d'humains virtuels est fixé à 10 (Voir figure.5.18 -5.19) :

- Dans le niveau 1, nous avons deux humains virtuels situés respectivement aux positions (x=0, y=5) et (x=-10, y=15).
- Dans le niveau 2, nous avons cinq humains virtuels situés respectivement aux positions (x=2, y=30), (x=-35, y=35), (x=-20, y=40), (x=-8, y=45) et (x=-30, y=50).
- Dans le niveau 3, nous trouverons trois humains virtuels situés respectivement aux positions (x=-8, y=60), (x=-20, y=75) et (x=-30, y=90).

Figure 5.18 : Les humains virtuels ayant commencé la construction

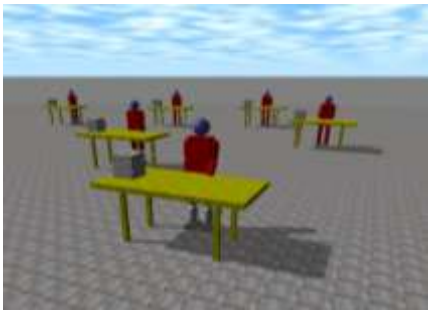
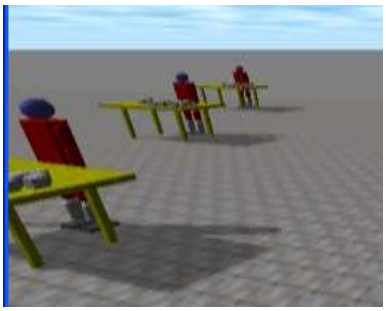


Figure 5.19 : Les humains virtuels ayant terminé la construction

Le comportement des trois derniers humains virtuels est invisible parce qu'ils se situent en troisième niveau de détail, par contre le comportement des autres humains est simulé avec plus ou moins de précisions selon leurs de LOD.



Figue 5.20 : Changement de la position de la caméra est effectué, tel que les trois humains virtuels, qui ont été invisibles sont devenus très précis.

Un rapprochement au niveau de la position de la caméra, nous a permis de voir le comportement des trois derniers humains virtuels[14] (Voir figure.5.20).

5.6. Elaboration du test

Nous avons effectué un test comparatif. La technique de LOD IA n'a pas été mise en place, et l'application a été exécutée avec les paramètres suivants :

- Déplacement : fréquence maximale et pas minimal.
- Comportement : précis (la construction est effectuée brique par brique).
- Représentation graphique pour les humains virtuels (Le modèle d'humain virtuel se compose de 12 primitives géométriques de corps rigide, et 13 joints).

Niveau	Déplacement (fréquence, pas)
1	(1/56s, 0.1/3)
2	(1/56s, 0.1/3)
3	(1/56s, 0.1/3)

Tableau.5.4 : Tableau des Paramètres de déplacement, s=seconde et l'unité de mesure est le pixel

Après avoir collecté les couples de données (nombre de polygones, temps d'exécution), (voir Tableau.5.5 -5.6) associées à chaque série de test (3, 6, et 10 Humains virtuels), pour chaque série, on dessine les histogrammes correspondants(voir Figure 5.21 -5.22) et on déduit un certain nombre de conclusions.

5.6.1. Cas sans application de la technique de LOD IA

Le tableau 5.5 résume l'ensemble des résultats obtenus pour 3, 6 et 10 humains virtuels sans l'application de la technique de LOD IA.

[14] Les humains virtuels n'ont pas commencé encore la construction.

Nombre de personnage	Nombre de polygones	Temps de calcul
03	123	19.35s
06	246	38.70s
10	410	67s

Tableau. 5.5 : Tableau récapitulatif (la technique de LOD IA a été supprimée)

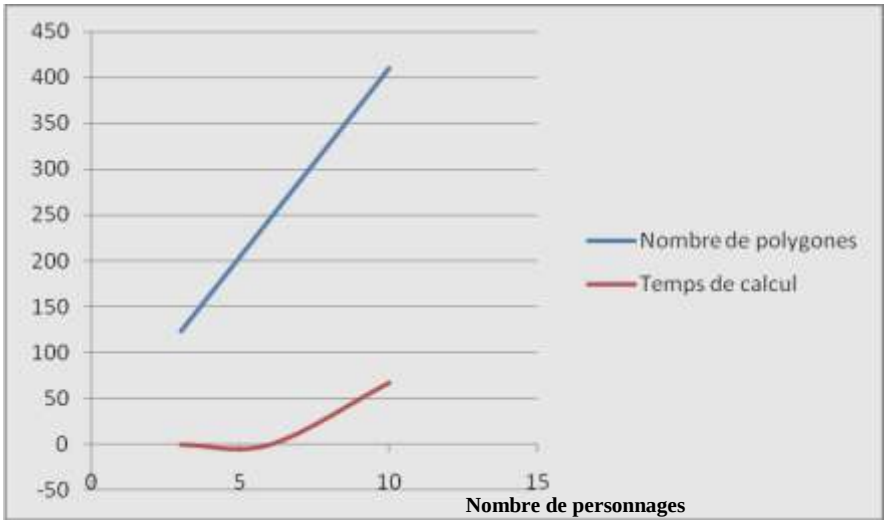


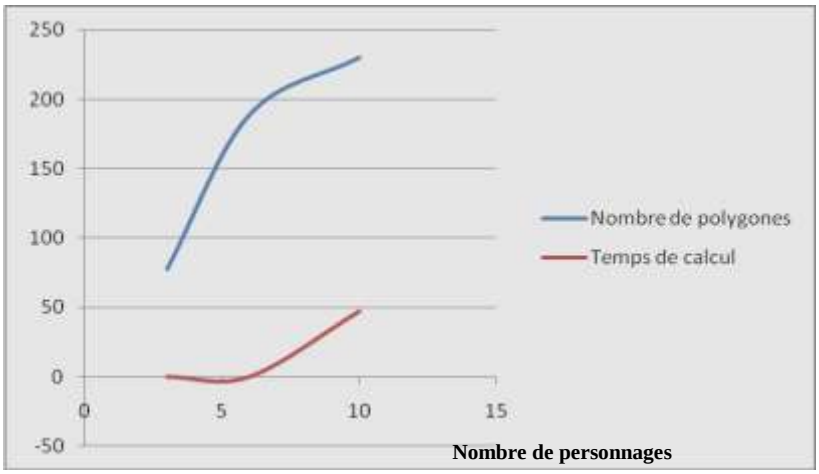
Figure 5.21 : Graphe du temps de calcul et du nombre de polygones (sans l'application de la technique de LOD IA)

5.6.2. Cas avec application de la technique de LOD IA

Le tableau 5.6 résume l'ensemble des résultats obtenus pour 3, 6 et 10 humains virtuels avec l'application de la technique de LOD IA.

Nombre de personnage	Nombre de polygones				Temps de calcul
	Niveau 1	Niveau 2	Niveau 3	Nombre total	
03	41	37	00	78	12.35s
06	41	148	00	189	24.70s
10	82	148	00	230	47s

Tableau. 5.6 : Tableau récapitulatif (la technique de LOD IA a été utilisée)



Figue 5.22 : Graphe du temps de calcul et du nombre de polygones (avec l'application de la technique de LOD IA)

5.6.3. Evaluation des résultats

Sans l'application de la technique de LOD IA (voir Figure 5.21):

- Pour le nombre de personnages 3, les performances est 123 pour le nombre de polygones et 19.35s pour le temps d'exécution.
- Pour le nombre de personnages 6, les performances est 246 pour le nombre de polygones et 38.70s pour le temps d'exécution.
- Pour le nombre de personnages 10, les performances est 410 pour le nombre de polygones et 67s pour le temps d'exécution.

Avec l'application de la technique de LOD IA (voir Figure 5.22) :

- Pour le nombre de personnages 3, les performances est 78 pour le nombre de polygones et 12.35s pour le temps d'exécution.
- Pour le nombre de personnages 6, les performances est 189 pour le nombre de polygones et 24.70s pour le temps d'exécution.
- Pour le nombre de personnages 10, les performances est 230 pour le nombre de polygones et 47s pour le temps d'exécution.

D'après les résultats, nous constatons que les performances dans le cas de l'application de la technique de LOD IA sont toujours meilleurs que les performances sans l'utilisation de la technique de LOD IA quelque soit le nombre de personnage :

- Pour trois personnes : nous avons une différence de 45 polygones et une différence de 7s en temps d'exécution.
- Pour six personnes : nous avons une différence de 57 polygones et une différence de 14s en temps d'exécution.

- Pour dix personnes : nous avons une différence de 180 polygones et une différence de 20s en temps d'exécution.

Conclusion

La mise en place de la technique de LOD IA, nous a permis de réaliser une simulation crédible ,d'un grand monde virtuel ,dans les limites des ressources informatiques disponibles.

5.7. Délimitation des niveaux : choix des distances

Le but de ce test est de déterminer la bonne séparation des niveaux, en d'autres termes, trouver les bonnes valeurs pour les variables d_1 et d_2. La meilleure façon de procéder, est de tester plusieurs valeurs, et de retenir celles qui satisfont le mieux l'utilisateur (exemple de la figure 5.23). Nous pouvons prévoir des modifications en temps réel pour donner plus de flexibilité à l'utilisateur.

Le niveau 1 : Le choix de la distance d_1 est déterminant, car si elle trop grande, on risque d'avoir beaucoup trop de personnages en niveau 1. L'application serait surchargée de calcul, et ralentirait. Si elle est trop petite, trop peu de personnages seront visibles et cela risquerait de compromettre l'observation.

Les autres niveaux : Les autres distances doivent être choisies de telles sorte qu'en modifiant la position de la caméra, le changement de la forme des humains virtuels ne soit pas trop voyant et aussi assurer la fluidité du basculement entre les niveaux de la simulation progressive du comportement[15]. De plus, comme nous l'avons signalé dans le paragraphe précédent, il faut avoir à l'esprit, la répartition de la complexité des calculs. Il est préférable, par analogie à l'œil humain, de les choisir de telle sorte qu'il y ait un écart croissant entre elles : $d_2 - d_1 < d_3 - d_2 < d_4 - d_3$ ….. avec $d_1 < d_2 < d_3 < d_4$.

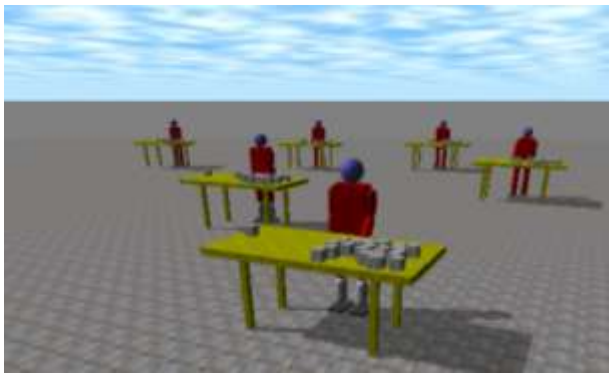


Figure 5.23 – Délimitation des niveaux : d1 = 30 et d2 = 60

[15] La fluidité du basculement entre le comportement de construction de la maison brique par brique et la construction de la maison ligne par ligne.

5.8. Comparaison avec les travaux antérieurs

Notre travail se focalise sur la simulation de grands mondes artificiels, en incorporant la technique de niveau de détail (habituellement utilisée pour réduire la complexité des données d'une scène), au niveau comportemental « LOD IA ».

Cette technique : « LOD IA » consiste à réduire les demandes de la simulation en ressources informatiques en diminuant graduellement la qualité de la simulation des comportements des humains virtuels qui se trouvent dans des places moins importantes, tout en gardant une simulation crédible. Notre travail proposé repose sur une architecture inspirée du modèle ALOHA qui offre un banc d'essai graphiquement sophistiqué pour de nouvelles technologies d'agents simulés.

La technique de LOD a été employée dans les jeux d'ordinateur, par la non-simulation des régions du monde qui ne sont pas visibles par l'utilisateur. Cependant, ceci rend souvent l'environnement incorrect et peut générer des contradictions.

L'utilisation de la technique des LOD IA consiste à effectuer un transfert du domaine des infographies des ordinateurs vers le domaine de l'intelligence artificielle [Kkp+06]. Cette dernière est basée sur une simple idée qui consiste en ce qui suit:

Il existe souvent peu de lieux, dans le monde artificiel, à un moment donné de la simulation, où le comportement des humains virtuels est important. Dans ces localités, le comportement doit être simulé avec précision, les localisations les moins importantes n'ont pas besoin de simuler le comportement des humains virtuels avec précision, leur simulation sera dégradée.

Si le comportement des humains virtuels est simulé partiellement, les exigences de la simulation en matière de ressources informatiques peuvent être réduites significativement.

Le tableau 5.7 montre une vue d'ensemble des travaux précédents utilisant la technique LOD IA, caractérisée par les critères suivants : le critère de détermination de LOD, le nombre de niveaux de détails, l'usage de LOD et les comportements simulés par l'IA.

Auteurs	Le critère de détermination de LOD	Le nombre de LOD	L'usage de LOD	Les comportements simulés par IA
Chenney et al. [Caf01]	Le potentiel de la visibilité	02	Mettre à jour le mouvement	La navigation, l'évitement de collision
O'Sullivan et al [Scv+02]	La distance	Non spécifié	La géométrie, les animations, l'évitement de collision, les gestes, les expressions faciales et la sélection d'action	La navigation, l'évitement de collision, le dialogue complexe entre les agents
Brockington [Bro02]	Distance	05	La recherche, la navigation, la sélection d'action dans un combat	La navigation, l'évitement de collision, les interactions de combat complexes
Niederberger and Gross [NG05]	La distance et la visibilité	21	La planification, l'évitement de collision, la recherche du chemin et les décisions de groupe	La navigation, l'évitement de collision et la planification du chemin
Pettré et al [Pcm+06]	La distance et la visibilité	05	La géométrie, la mise à jour de mouvement, l'évitement de collision, la navigation	La navigation, l'évitement de collision, la recherche de chemin
Brom et al. [Bsp07]	La distance simplifiée	04	La sélection d'action (avec l'arbre AND-OR), la simplification environnementale	La navigation, les interactions complexes avec les objets et les autres agents
Paris et al. [Pgo09]	La distance	03	La navigation et l'évitement de collision	La recherche de chemin, la navigation et l'évitement de collision
Lin et Pan [LP07]	distance	Non spécifié	La géométrie et les animations	La locomotion
Osborne and Dickinson [OD10]	La distance	Non spécifiée	La navigation, flockage, les décisions de groupe	La navigation
M.Wibner et al [Wka10]	La distance et la visibilité	08	La mise à jour du mouvement, l'évitement de collision, la navigation et l'exécution de l'action	La navigation, l'évitement de collision, les interactions basées sur les désires avec les agents et les objets intelligents, les dialogues
Notre approche S.Zertal et al. [Zds 11]	La distance	03	La géométrie, les animations et la mise à jour du comportement selon le niveau de LOD	L'évitement de collision, la recherche du chemin et le comportement de construction d'une maison.

Tableau 5.7 : Comparaison des différentes approches de LOD IA

Ainsi, de cela, nous pouvons dire que la nouveauté apportée par notre travail consiste à pouvoir effectuer une simulation crédible du comportement des humains virtuels, en réduisant les demandes de la simulation en ressources informatiques.

Donc les points forts de nos résultats sont :

- La crédibilité dans le processus de simulation du comportement des humains virtuels.
- La simulation rapide, en temps réel et dans les limites des ressources informatiques disponibles.
- la simulation d'un grand monde artificiel possédant un nombre important d'humains virtuels exécutant un comportement précis.
- L'extensibilité de notre système, nous pouvons ajouter d'autres humains virtuels exécutant des comportements complexes comme le comportement de préparation du diner par exemple

Les points que nous pourrons améliorer à propos des autres travaux ou comme nouvelle perspectives sont :

- L'application de la technique de LOD au niveau de la géométrie, ce qui permettrai de gagner davantage dans l'optimisation, en utilisant à titre d'exemple le maillage triangulaire.
- L'utilisation des techniques d'animation comme la dynamique, la cinématique et la cinématique inverse pour simuler les mouvements et les animations des humains virtuels avec plus ou moins de précision. Par exemple, pour simuler un mouvement très précis, nous pourrons exploiter la dynamique et pour un mouvement moins précis nous pourrons nous contenter de la cinématique.
- La prise en compte de la complexité et la diversité du comportement au niveau de notre application.
- L'utilisation d'autres critères de détermination des LODs comme la périphérie à titre d'exemple.
- Le contrôle du comportement de nos humains virtuels peut être amélioré en intégrant un algorithme d'apprentissage.
- La simulation des comportements non-hiérarchiques qui ne peuvent pas être divisés en d'autres comportements.

5.9. Conclusion

Ce chapitre a été consacré à la description de quelques détails de l'implémentation de notre architecture proposée dans le chapitre précédent qui consiste à simuler le comportement des humains virtuels selon plusieurs niveaux de détails en appliquant la technique de « LOD IA ». Nous avons essayé de spécifier les différentes étapes d'implémentation tout en décrivant la structure de données et les algorithmes utilisés. La présentation des résultats et l'élaboration des tests nous a permis de faire une évaluation globale sur les performances du modèle proposé.