



N° d'ordre :

UNIVERSITE MOHAMED BOUDIAF-M'SILA
FACULTE DES MATHÉMATIQUES ET DE L'INFORMATIQUE
Département d'Informatique

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Systèmes d'Informations Avancés

Par : MAHDI Riyadh

SUJET

Simulation d'une foule virtuelle par la logique floue

Soutenu publiquement le : 15 / 06 / 2015 devant le jury composé de :

.....	Université de M'sila	Président
Mr. CHATRA Mohamed	Université de M'sila	Rapporteur
.....	Université de M'sila	Examineur
.....	Université de M'sila	Examineur

Promotion : 2014/2015



N° d'ordre :

UNIVERSITE MOHAMED BOUDIAF-M'SILA
FACULTE DES MATHÉMATIQUES ET DE L'INFORMATIQUE
Département d'Informatique

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Systèmes d'Informations Avancés

Par : MAHDI Riyadh

SUJET

Simulation d'une foule virtuelle par la logique floue

Soutenu publiquement le : 15 / 06 / 2015 devant le jury composé de :

.....	Université de M'sila	Président
Mr. CHATRA Mohamed	Université de M'sila	Rapporteur
.....	Université de M'sila	Examineur
.....	Université de M'sila	Examineur

Promotion : 2014/2015

Remerciements

Je remercie "Dieu" de nous avoir donné la patience, la santé, le courage d'avoir arrivé jusque-là.

Je tiens à remercier le docteur : CHATRA Mohamed qui m'a honoré par son encadrement, pour sa direction, son orientation, ses conseils et toutes ses remarques constructives pour le bon déroulement de notre travail.

Merci à mes parents de m'avoir amené jusqu'ici, merci également à toute ma grande famille, frères, sœurs.

En fin, nous remercions, avec toute la suprême sincérité, tous ceux qui nous ont aidés de près ou de loin à terminer et à réaliser ce travail.

MAHDI Riyadh.

Table des matières

INTRODUCTION GENERALE.....	1
----------------------------	---

CHAPITRE 1

PERSONNAGES AUTONOMES EN ENVIRONNEMENT VIRTUEL

Introduction	3
1.1 Les différents types d'agents	3
1.1.1. Les agents délibératifs	3
1.1.2. Les agents réactifs	4
1.1.3. Les agents hybrides	4
1.1.4. Agents intelligents et humains virtuels	5
1.2 Représentation exacte de l'environnement	6
1.2.1 Triangulation de Delaunay	6
1.2.2 Décomposition en trapèzes.....	7
1.3 Représentations approximatives de l'environnement.....	7
1.3.1 Modèles à base de grilles	7
1.3.2 Cartes de cheminement	9
1.3.3 Graphe de visibilité	9
1.3.4 Diagramme de Voronoï généralisé.....	10
1.3.5 Cartes de cheminement probabilistes	10
1.4 Le modèle de champs de potentiels.....	11
1.5 Le besoin d'humains virtuels en réalité virtuelle	12
Conclusion.....	13

CHAPITRE 2

ANIMATION COMPORTEMENTALE

Introduction	14
2.1 Observation du comportement des piétons	14
2.1.1 Environnement	14
2.1.2 Interactions avec l'environnement	15
2.1.3 Déplacements des piétons	16
2.2 L'animation comportementale	17

2.3 Le comportement d'individu	18
2.3.1 Comportement de recherche et fuite	18
2.3.2 Comportement d'arrivée	18
2.3.3 Comportement d'évitement d'obstacles.....	19
2.3.4 Comportement de suivi de chemin.....	20
2.3.5 Evitement de collision non-alignée	21
2.4 L'algorithme de parcours A étoile (A*).....	21
Conclusion.....	23

CHAPITRE 3

FOULE D'HUMAINS VIRTUELS ET LA LOGIQUE FLOUE

Introduction	24
3.1 Propriétés des humains virtuels.....	24
3.1.1 La perception.....	24
3.1.2. L'émotion	25
3.1.3. Le comportement.....	26
3.1.4. L'action	27
3.1.5. La mémoire	27
3.2 Les deux approches de foule	28
3.2.1 Modèles macroscopiques	28
3.2.2 Modèles microscopiques	29
3.3 Introduction à la logique floue	33
3.3.1 Historique et définition.....	33
3.3.2 Logique classique et logique floue.....	34
3.4 Sous-ensembles floues	34
3.4.1 Définition	35
3.4.2 Opérations de base sur les sous-ensembles floues	35
3.5 Le contrôleur à logique flou	35
3.5.1 Variables linguistiques	35
3.5.2 Fuzzification.....	36
3.5.3 Règles floues	36

3.5.4 Inférence flou	37
3.5.5 Défuzzification	38
3.5.6 Schéma d'une commande floue	38
Conclusion.....	39

CHAPITRE 4

CONCEPTION DU LA CIRCULATION DES PIETONS

Introduction	40
4.1 L'objectif du système	40
4.2 L'architecture logicielle dédiée à la simulation	40
4.2.1 Conception globale de notre système	40
4.2.2 Conception détaillée de notre système	41
4.3 Structure de notre système	43
4.3.1 Sélection de chemin	45
4.3.2 Détection des piétons dans la zone de perception	45
4.4 Modélisation de la scène	46
4.4.1 L'environnement	46
4.4.2 Les individus	47
4.5 Données observées	47
4.6 Fonctions d'appartenance de la variable 'distance'	48
4.7 Contrôleur flou	49
4.8 La prévision de collision	50
4.9 Comportement d'évitement la collision	51
Conclusion.....	54

CHAPITRE 5

IMPLEMENTATION ET RESULTATS

Introduction	55
5.1 L'environnement de développement.....	55
5.1.1 L'outil d'implémentation	55
5.1.2 Présentation Open GL	56
5.2 Méthode de simulation	56

5.3 L'interface principale de notre application	57
5.4 Structure de données et quelque algorithmes	61
5.4.1 Structure de données	61
5.4.2 Algorithme général du système	62
5.4.3 Algorithme d'évitement de collision piéton par piéton.....	62
5.5 Résultats de la simulation.....	64
5.5.1 Résultats de l'algorithme A*	64
5.5.2 Résultats de comportements d'évitement de collision	65
5.5.3 Résultats de simulation « cas de panique »	66
Conclusion.....	67
CONCLUSION GENERALE	68
Bibliographie	69

Liste des figures

Figure 1. 1 : Un agent hybride.....	4
Figure 1. 2 : Discrétisation de l'espace avec une triangulation Delaunay.....	6
Figure 1. 3 : Discrétisation en trapèzes.....	7
Figure 1. 4 : Exemple de grilles régulières.....	8
Figure 1. 5 : Utilisation de grille en animation.....	8
Figure 1. 6 : Exemple de carte de cheminement.....	9
Figure 1. 7 : Graphe de visibilité calculé sur l'environnement.....	10
Figure 1. 8 : Graphe Voronoï généralisé.....	10
Figure 1. 9 : Carte de champs de potentiels.....	11
Figure 1. 10 : Première apparition d'une présentatrice virtuelle sur internet.....	13
Figure 2. 1 : Trajectoires de piétons traversant une route, hors et sur passage piéton.....	16
Figure 2. 2 : Quelques exemples d'applications de l'animation.....	17
Figure 2. 3 : Le comportement d'arrivée.....	19
Figure 2. 4 : L'évitement d'obstacle.....	20
Figure 2. 5 : Le comportement de suivre de chemin.....	20
Figure 2. 6 : Le comportement d'évitement de collisions non-alignées.....	21
Figure 2. 7 : L'algorithme A*.....	21
Figure 3. 1 : Structure du Modèle comportemental.....	24
Figure 3. 2 : Modèles de flots.....	29
Figure 3. 3 : Automates cellulaires.....	30
Figure 3. 4 : Exemples de règles de comportement proposées par Reynolds.....	30
Figure 3. 5 : Simulations de modèles de forces.....	31
Figure 3. 6 : modèles géométriques.....	31
Figure 3. 7 : Modèles basés vision.....	32
Figure 3. 8 : Processus de résolution d'une interaction dans [Lcl07].....	33
Figure 3. 9 : Représentation graphique d'un ensemble classique et d'un ensemble flou.....	34
Figure 3. 10 : Classification des tailles d'un humain en deux ensembles.....	34
Figure 3. 11 : Schéma d'une commande floue.....	38
Figure 4. 1 : Conception globale.....	41
Figure 4. 2 : Conception détaillée.....	41
Figure 4. 3 : Structure de phase perception.....	42

Figure 4. 4 : Structure de phase raisonnement et prise de décision.....	42
Figure 4. 5 : Structure de phase action.....	43
Figure 4. 6 : Organigramme du système.....	44
Figure 4. 7 : Algorithmes de base.....	45
Figure 4. 8 : la zone de perception de piéton.....	46
Figure 4. 9 : Modélisation de la scène.....	47
Figure 4. 10 : Fonctions d'appartenance de la variable 'distance'.....	49
Figure 4. 11 : Configuration interne du notre contrôleur flou.....	50
Figure 4. 12 : types de collisions.....	51
Figure 5. 1 : Fenêtre d'exécution de CodeGear C++ Builder 2009.....	56
Figure 5. 2 : Structure de simulation.....	57
Figure 5. 3 : Fenêtre principale de notre application.....	57
Figure 5. 4 : Vue en 2D (jardine).....	59
Figure 5. 5 : Vue en 2D (Ville).....	59
Figure 5. 6 : Vue en 3D (jardine).....	60
Figure 5. 7 : Vue en 3D (ville).....	60
Figure 5. 8 : Résultat d'application de l'algorithme A* pour un individu.....	64
Figure 5. 9 : Résultat d'application de l'algorithme A* pour un groupe des individus.....	64
Figure 5. 10 : Résultat d'évitement collision face à face.....	65
Figure 5. 11 : Résultat d'évitement collision à côté.....	65
Figure 5. 12 : Résultat d'évitement collision en arrière.....	66
Figure 5. 13 : Résultats de cas panique.....	66

INTRODUCTION GENERALE

L'animation comportementale cherche à offrir des modèles et des outils permettant la création d'entités virtuelles autonomes. Ces entités perçoivent leur environnement, agissent sur ce dernier et surtout prennent d'elles-mêmes des décisions en rapport avec la situation perçue dans le but d'exhiber un comportement cohérent proche de l'organisme vivant simulé.

La simulation de foules de piétons est un sujet d'actualité dans le domaine de l'infographie. L'importance de lancer des simulations de comportement de foules dans différentes situations réside dans l'impossibilité d'accomplir, d'une manière rapide et concluante, le comportement réel avec des humains. Pour cette raison, la simulation de la foule des piétons concerne beaucoup de domaines d'applications tels que la sécurité, le génie civil, l'urbanisme...etc. En génie civil par exemple, on s'intéresse aux caractéristiques de flux de foules des piétons afin d'assurer une évacuation sécurisée dans des situations d'urgences. Dans l'urbanisme et la conception des bâtiments, la simulation des foules des piétons est utilisée pour tester la fiabilité des équipements publics et des conceptions architecturales. La simulation de foule des piétons trouve aussi d'autres applications dans l'industrie de divertissements tels que les jeux vidéo.

Notre travail est proposé un modèle de simulation des phénomènes naturels, en utilisant comme application la navigation des piétons autonomes formants une foule. Nous avons s'intéresser dans ce travail par un certain nombre d'approches et techniques pour donner l'animation de ces piétons un pragmatisme et plus logistique comme l'utilisation de la logique floue comme une technique de détection et traitement des collisions entre les individus virtuels pour donner la souplesse de l'animation. Et l'utilisation de l'algorithme A* pour la planification du plus court chemin initiale de chaque individu.

Ce mémoire est organisé en deux parties. La première partie est un développement de l'état de l'art sur tous les aspects qui entrent dans le domaine de l'animation comportementale et la simulation des humains virtuels. Cette partie s'articule sur trois chapitres en séquence comme suite :

- Le premier chapitre présente les différents types des humains virtuels, ainsi que présentons les différents modèles de représentations de leurs environnements soit exacts ou approximatifs, et nous avons fait signe à le besoin d'humains virtuels en réalité virtuelle.

- Le deuxième chapitre introduit l'animation comportementale avec l'observation du comportement des piétons, et ainsi les différents comportements d'individu, ainsi que une explication de la méthode de l'emploi de l'algorithme A étoile (A*).
- Le troisième chapitre présente les propriétés des piétons virtuels, caractérisés par la perception, l'émotion, le comportement, l'action, et la mémoire, puis nous présentons les approches de simulation de foule macroscopique et microscopique, à la fin, nous présentons la logique floue.

La deuxième partie de ce mémoire traite la conception et la réalisation de notre application avec la présentation de nos résultats. Elle est constituée de deux chapitres :

- Le quatrième chapitre présente la conception globale, ainsi les techniques de la navigation et comment modéliser notre scène pour obtenir une animation réaliste des piétons virtuels.
- Le dernier chapitre présente notre environnement de développement avec les outils utilisés pour l'implémentation de notre application, nous avons également présenté quelques images résultats pour notre simulation.

Puis, nous avons terminé ce mémoire par une conclusion générale avec quelques perspectives.

CHAPITRE 1

PERSONNAGES AUTONOMES EN ENVIRONNEMENT VIRTUEL

Introduction

La création des personnages virtuels est un thème de recherche appartenant au domaine de l'animation comportementale et l'évolution des modèles informatiques avec l'approche microscopique s'est intéressée à l'aspect individuel de la décision relative au déplacement. On voit donc apparaître avec ces modèles la nécessité d'une description topologique, permettant de représenter et d'organiser l'environnement de simulation. Cette description topologique peut être exacte ou approximative ou le modèle de champs de potentiels, et peut éventuellement contenir des informations sémantiques.

Nous allons présenter, dans le présent chapitre, les agents de l'animation comportementale mettent en œuvre différentes approches, et en cite deux types de représentation, la décomposition approximative de l'espace libre, et la décomposition exacte. Et en présente le modèle de champs de potentiels afin de choisir la meilleure représentation de notre environnement. Ensuite nous présentons le besoin d'humains virtuels en réalité virtuelle.

1.1 Les différents types d'agents

1.1.1. Les agents délibératifs

Les agents délibératifs (aussi appelés agents cognitifs) sont issus du paradigme de l'Intelligence Artificielle symbolique, et plus particulièrement de l'hypothèse du système de symbole physique formulée par Newell et Simon [Nes76]. Un système de symbole physique consiste en un ensemble d'entités appelées symboles ou formes organisées (motifs), qui peuvent être associées, par le jeu d'une ou de plusieurs relations, en de plus grandes structures et peuvent être transformées par un petit ensemble de processus de base. Ces processus peuvent créer de nouveaux symboles, créer et modifier des relations entre les symboles, stocker plus ou moins indéfiniment des chaînes de symboles, ou encore comparer des symboles par identité ou par différence. L'hypothèse du système de symbole physique dit qu'un tel système dispose des moyens nécessaires et suffisants pour exercer une action générale intelligente.

Un agent délibératif est donc une entité qui contient une représentation explicite et symbolique du monde à partir de laquelle la prise de décision (par exemple quelle action accomplir) sera faite par le biais de raisonnements logiques (ou pseudo logiques), ces derniers étant basés sur l'appariement de motifs et la manipulation symbolique. En résumé, un agent délibératif utilise une représentation interne de son environnement afin de formuler des plans d'actions pour satisfaire ses buts. Ceci est généralement réalisé à l'aide d'un moteur d'inférences fonctionnant sur le principe suivant :

État courant du monde + but $\rightarrow$ plan.

1.1.2. Les agents réactifs

Afin de pallier le manque de robustesse, de réactivité et d'adaptabilité des *agents cognitifs*, des chercheurs ont proposé, dès le milieu des années 80, des agents plus simples, n'utilisant pas de représentation interne de leur environnement et prenant leur décision de façon réflexe à partir d'un stimulus. En effet, d'après Brooks [Bro86], Steels [Ste94] ou Maes [Mae94], un comportement intelligent peut être généré sans représentation symbolique explicite ni raisonnement abstrait : il émergera des interactions de l'agent avec son environnement complexe. Ainsi, un *agent réactif* est un agent qui, pour chaque configuration de ses capteurs, sans passer par la phase de planification à partir d'une représentation interne de son environnement, déclenchera une action prédéfinie. Ce type d'agent peut être implémenté à l'aide d'un système de règles simples, ou d'un automate, selon le principe suivant :

État courant du monde $\rightarrow$ action.

1.1.3. Les agents hybrides

Les *agents hybrides* sont généralement des agents cognitifs intégrant des capacités réactives. L'association des deux méthodes permet de combiner leurs qualités complémentaires (planification, abstraction, réactivité) tout en limitant leurs défauts respectifs. (Figure 1. 1).

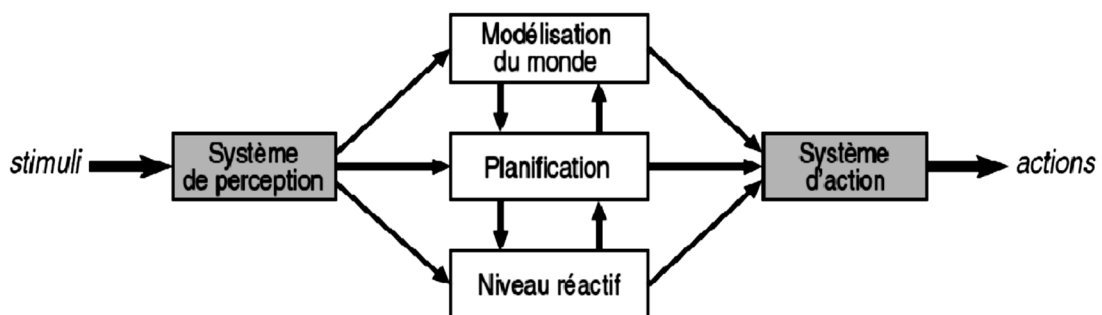


Figure 1. 1 : Un agent hybride [Che06].

1.1.4. Agents intelligents et humains virtuels

Les processus de la cognition humaine sont des plus complexes à modéliser sous la forme d'un agent intelligent. Dans l'optique de la simulation comportementale des êtres humains, Newell [New90] définit les contraintes qu'un agent idéal doit respecter pour simuler au mieux ce processus :

- flexibilité et autonomie : l'agent doit pouvoir s'adapter à des changements d'environnement.
- interactions "temps réel" avec l'environnement.
- richesse et complexité du comportement et du savoir.
- utilisation de concepts abstraits (symbolisme et généralisation).
- persistance (l'agent doit pouvoir réitérer des solutions éprouvées) et apprentissage.
- autonomie partielle ou vie en société : l'agent est régi par des lois sociales (groupe).
- conscience de soi et perception, l'agent doit pouvoir s'extérioriser à travers ses actions en fonction de sa situation.
- évolution à la fois de l'architecture le définissant (extensibilité) et de son espèce.

Newell n'a cependant pas mentionné un autre concept qui semble essentiel lors de la simulation de personnages virtuels : l'illusion de crédibilité. En effet, de nombreux chercheurs soulignent le fait que l'agent doit se comporter de manière suffisamment crédible et "intelligente" pour que son attitude soit perçue comme similaire à celle d'un être réel, ceci étant d'autant plus vrai si le personnage est humanoïde et réaliste [Rdn02].

De nos jours, les connaissances en informatique ne semblent pas encore suffisantes pour permettre la réalisation d'un système décisionnel capable de satisfaire toutes ces contraintes.

Aussi, les différentes architectures présentées dans la littérature permettent toutes la simulation d'agent intelligent mais aucune ne remplira exactement toutes les spécifications requises par Newell. L'illusion de crédibilité est, elle, plus facile à rendre en raison de sa subjectivité et d'attentes du concepteur ou de l'utilisateur d'un système spécifique.

1.2 Représentation exacte de l'environnement

Les représentations exactes de l'environnement vont chercher à organiser les données spatiales tout en conservant intégralement les informations qu'elles contiennent à l'origine.

La méthode utilisée consiste généralement à découper l'espace navigable en cellules convexes de différentes formes (triangles, polygones, trapèzes...). Il existe plusieurs modèles pour effectuer cette subdivision comme Triangulation de Delaunay et Décomposition en trapèzes.

1.2.1 Triangulation de Delaunay

La triangulation de Delaunay [Boy98] crée un ensemble de triangles en réunissant des points fournis en entrée. L'algorithme d'unification respecte pour contrainte que le cercle circonscrit à un triangle ne contienne aucun autre point que les trois sommets qui le composent. Une conséquence de cette propriété est que l'angle minimum d'un triangle produit est maximisé. La complexité de l'algorithme d'unification est en $O(n \ln n)$, avec n le nombre de points fournis en entrée.

La propriété la plus intéressante de cette triangulation est que chaque point y est relié à son plus proche voisin par l'arête d'un triangle. Ainsi, cette triangulation peut être utilisée pour représenter l'espace navigable, les points d'entrée étant issus des obstacles. Une autre utilisation possible de cette triangulation est cette fois dynamique lors de la simulation [Lad04], pour calculer un graphe de voisinage entre les entités mobiles, qui serviront cette fois-ci de points d'entrées.

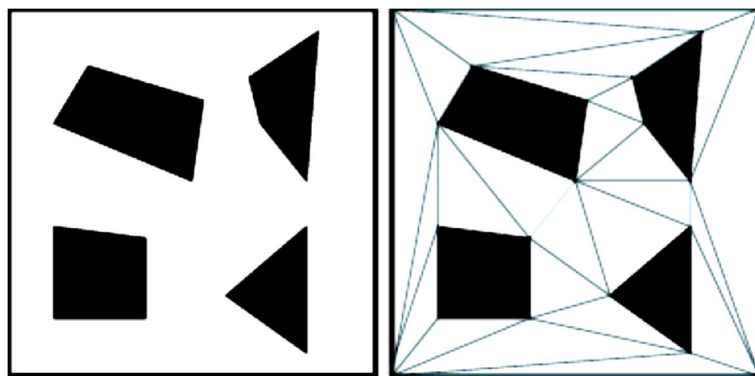


Figure 1.2 : Discretisation de l'espace avec une triangulation Delaunay [Ama12].

1.2.2 Décomposition en trapèzes

Cette décomposition permet de décomposer l'environnement sous forme de cellules trapézoïdales [Lat91]. Elle est basée sur un algorithme de balayage [Boy95]. Les points délimitant la géométrie de l'environnement sont triés suivant l'axe des ordonnées. Puis un maximum de deux segments est généré, ayant pour origine le point sélectionné et pour extrémité la prochaine intersection avec le bord d'un polygone délimitant un obstacle.

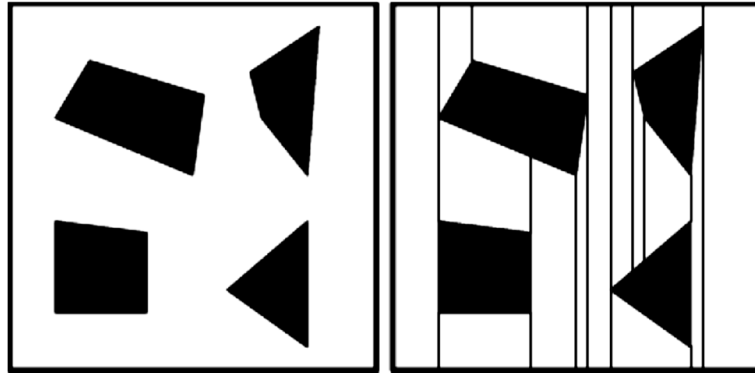


Figure 1. 3 : Discretisation en trapèzes [Ama12].

1.3 Représentations approximatives de l'environnement

Les représentations approximatives de l'environnement sont sans doute les plus utilisées en animation comportementale, du fait de leur facilité de mise en œuvre. Ces représentations vont décrire l'espace libre de navigation avec des formes géométriques simples telles que des carrés ou des segments. Deux modèles entrent dans cette catégorie : les modèles à base de grilles, et les cartes de cheminement.

1.3.1 Modèles à base de grilles

Le premier modèle de représentation approximative utilise des grilles régulières, formées de cellules carrées en deux dimensions (Figure 1.4 (a)) et cubiques en trois dimensions. L'environnement est donc pavé par ces cellules, qui peuvent avoir trois états : libre, partiellement obstruée, et obstacle.

La précision de la représentation obtenue dépend directement de la taille des cellules utilisées : plus les cellules sont grandes, moins la représentation est précise. Bien entendu, l'augmentation de la précision induit une augmentation proportionnelle de l'occupation mémoire. L'occupation mémoire de cette méthode constitue ainsi son premier point faible, se répercutant directement sur la complexité de recherche de chemin à l'intérieur de l'environnement [Stb96].

Pour réduire ce problème, une évolution de ce modèle est apparue sous la forme des grilles hiérarchiques [Wst05]. Cette méthode décrit l'espace navigable par une succession de grilles de plus en plus précises, organisées sous forme d'arbre. L'algorithme de construction va commencer par paver l'environnement avec les cases les moins précises, puis découper récursivement les cases partiellement obstruées, en quatre parties pour la 2D (formation d'un quadtree (Figure 1.4 (b)), ou en huit pour la 3D (formation d'un octree). L'algorithme arrête les subdivisions dès qu'une précision fixée est atteinte, ou si la case produite n'est plus partiellement obstruée. Cette méthode est donc d'autant plus avantageuse que l'environnement est peu dense en obstacles.

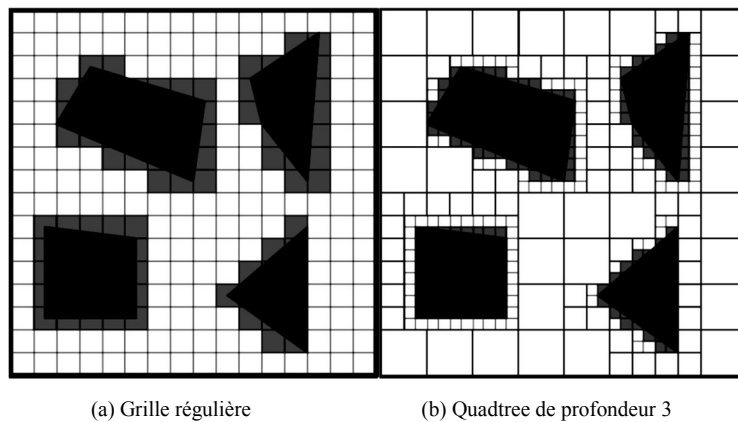


Figure 1. 4 : Exemple de grilles régulières [Ama12].

Les méthodes à base de grilles sont fortement utilisées en animation comportementale [Jjk98] du fait de leur simplicité de mise en œuvre et de leur rapidité d'exploitation. On peut ainsi voir des animations de milliers d'individus [Fcy02] basées sur ce mode de représentation (Figure 1.5).

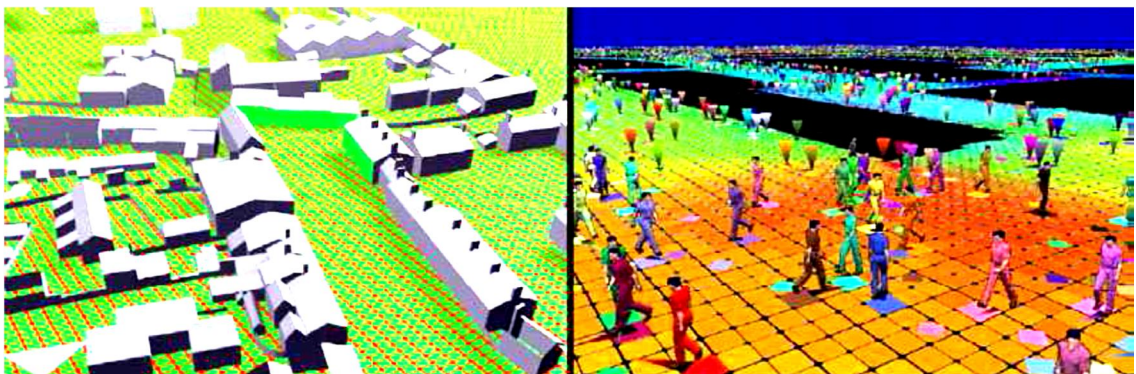


Figure 1. 5 : Utilisation de grille en animation [Séb07].

Les grilles sont très difficilement généralisables à des environnements quelconques, notamment s'ils comportent des obstacles non alignés sur les axes, et dans un cadre de simulation où l'individu devra se déplacer finement. Néanmoins, la rapidité d'accès aux informations qu'elle produit fait de cette méthode un complément envisageable à une approche plus fine de représentation de l'environnement. Pour finir, il faut remarquer que le niveau de discrétisation de la grille introduit implicitement une limite aux densités de populations que l'on peut simuler [Rja05]. Cela rend aussi le déplacement des entités discret, et les contraint généralement à avoir une taille standardisée.

1.3.2 Cartes de cheminement

Les cartes de cheminement discrétisent l'espace navigable sous la forme d'un réseau de chemins. Ce réseau est obtenu en reliant des points clefs répartis à l'intérieur de l'environnement (Figure 1.6). Plusieurs méthodes existent pour créer des cartes de cheminement, différentes dans la manière de créer et de relier ces points clefs.

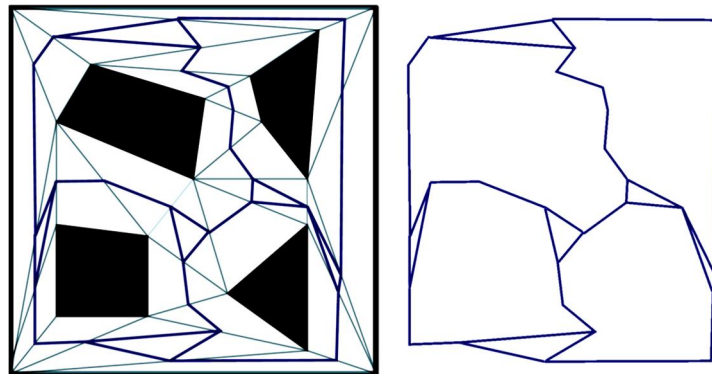


Figure 1. 6 : Exemple de carte de cheminement [Ama12].

(a) Calculs sur l'environnement d'origine

(b) Exemple de carte de cheminement

A gauche, triangulation de Delaunay (gris) de l'environnement d'origine, et carte de cheminement déduite (bleu). A droite, un exemple de carte de cheminement obtenue.

1.3.3 Graphe de visibilité

Le graphe de visibilité utilise les sommets des polygones décrivant les obstacles de l'environnement, comme points clefs de la carte de cheminement. Par la suite, deux points clefs sont reliés si et seulement s'ils sont mutuellement visibles. Autrement dit si et seulement si il existe un chemin les reliant en ligne droite et exempt d'obstacle (Figure 1.7).

Le réseau de chemins ainsi créé possède la propriété de maximiser la longueur des parcours en ligne droite, permettant ainsi de minimiser la distance parcourue [Acf01].

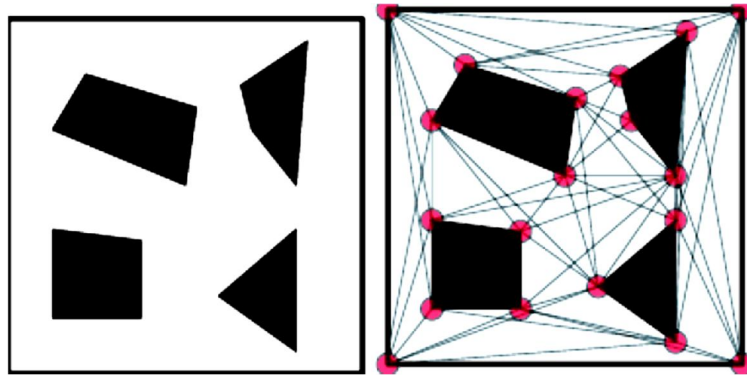


Figure 1. 7 : Graphe de visibilité calculé sur l'environnement [Ama12].

1.3.4 Diagramme de Voronoï généralisé

Cette méthode est basée sur une notion d'équidistance aux obstacles de l'environnement. Pour se faire, un ensemble de sites sont évalués au sein de l'environnement, correspondant aux obstacles, dont les intersections vont former les points clefs. Les chemins ainsi générés maximisent la distance aux obstacles. Ce calcul s'avère complexe, mais son peut être obtenu rapidement par des méthodes utilisant les cartes graphiques [Hkl99], où le calcul est obtenu directement en effectuant le rendu des sites. Une autre méthode utilise la triangulation de Delaunay pour obtenir les points clefs du diagramme de Voronoï généralisé, en se servant du centre des triangles calculés. La carte de cheminement ainsi produite peut être considérée comme un condensé des informations de la triangulation, ne contenant plus la définition géométrique des obstacles de l'environnement.

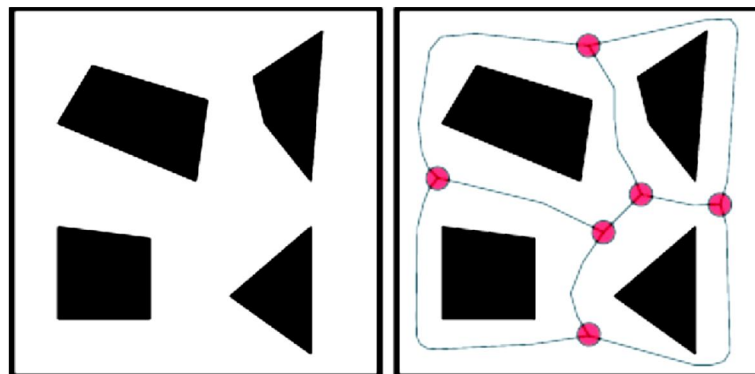


Figure 1. 8 : Graphe Voronoï généralisé [Ama12].

1.3.5 Cartes de cheminement probabilistes

Plutôt que de se baser directement sur les informations géométriques pour construire une carte de cheminement et identifier les points clefs, les cartes de cheminement probabilistes [Ove02] s'appuient sur une génération aléatoire de points clefs pour couvrir l'espace libre. Deux points peuvent ensuite être reliés s'il existe, entre eux, un chemin libre de collision. Ce

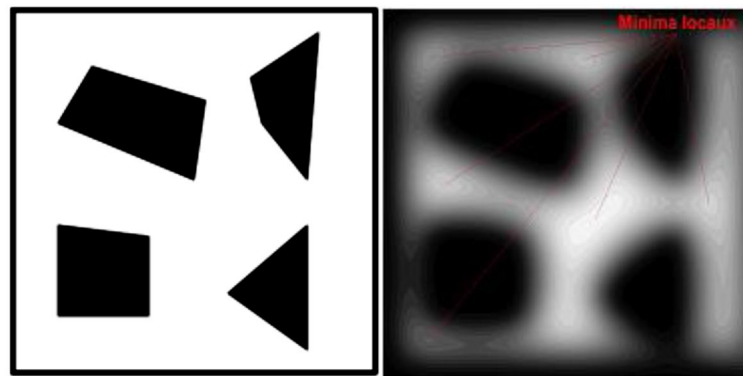
test constitue l'opération la plus coûteuse. Il exploite la géométrie de l'environnement et de l'entité en déplacement pour vérifier que le chemin généré ne provoque pas de collision.

Cette méthode est très utilisée dans le cadre de la planification de chemin pour des objets poly-articulés tels que des bras [Kuf99]. Dans le cadre de la locomotion cette technique a permis la génération de mouvements très complexes [Chl03] adaptés à la difficulté de l'environnement, tels que des sauts, marcher sur des piliers, se baisser pour éviter une branche, etc.

1.4 Le modèle de champs de potentiels

Le modèle à base de champs de potentiels consiste en une définition de l'environnement permettant directement de résoudre les déplacements de personnes.

Les champs de potentiels caractérisent les obstacles de l'environnement par des forces de répulsion et les buts par des forces d'attraction. Un gradient de forces, ou champ de potentiel, est alors déduit en chaque point de l'environnement comme étant une somme pondérée, le plus souvent par la distance, des potentiels de répulsion et du potentiel lié au but. La navigation d'une entité est alors simulée par une descente de gradient depuis sa position dans l'environnement (Figure 1.9).



(a) Environnement d'origine. (b) Exemple de champs de potentiels.

Figure 1. 9 : Carte de champs de potentiels [Par07].

Dans la Figure 1.9 (b), les obstacles (répulsion) sont décrits en noir et en blanc les zones de navigation (attraction). Le gradient de gris exprime la valeur de potentiel dans l'environnement. Cet exemple contient 6 minima locaux : zones isolées à potentiel d'attraction maximal.

1.5 Le besoin d'humains virtuels en réalité virtuelle

Avant de se focaliser sur le besoin d'humains virtuels en réalité virtuelle, il faut d'abord repréciser la finalité de la réalité virtuelle que partagent tous les acteurs du domaine. Comme expliqué en détail dans les autres volumes du traité, la finalité commune de toute application de réalité virtuelle est de :

« Permettre à une personne (ou à plusieurs) une activité sensori-motrice et cognitive dans un monde artificiel, créé numériquement, qui peut être imaginaire, symbolique ou une simulation de certains aspects du monde réel. »

Pour atteindre ce but, les techniques de réalité virtuelle offrent un environnement virtuel dans lequel l'utilisateur est immergé et avec lequel il peut interagir. Le troisième volume du traité «Les outils et les modèles informatiques des environnements virtuels» expose la conception et la réalisation des environnements virtuels interactifs, mais sans développer en détail la modélisation de l'humain virtuel dans toutes ses composantes. C'est donc l'objectif de ce cinquième volume, sachant que cette problématique est très vaste et fortement interdisciplinaire. Nous sommes bien conscients qu'un seul livre ne peut être exhaustif sur un sujet aussi étendu, ambitieux, voire utopique : modéliser l'être humain dans sa globalité. Mais bien des applications demandent des environnements virtuels peuplés d'humains virtuels. Ce volume fait donc modestement le point des possibilités actuelles et d'un futur proche. La vingtaine de chapitres sera utile à tout concepteur d'une application de réalité virtuelle qu'il souhaiterait peupler d'humains virtuels et, par extension, d'animaux virtuels (mais ce cas n'est pas traité spécifiquement dans ce livre). Schématiquement, un humain virtuel a trois usages différents :

- être un avatar, c'est-à-dire une représentation de l'utilisateur immergé dans l'environnement virtuel.
- être un personnage virtuel figuratif dans l'environnement virtuel (autonome).
- être un personnage virtuel qui va interagir avec l'utilisateur.

Dans ce troisième cas d'usage, l'interaction entre l'utilisateur réel et l'humain virtuel peut être sensorimotrice et (ou) cognitive. Au niveau sensorimoteur, l'utilisateur et l'humain virtuel peuvent participer ensemble à une tâche physique : par exemple, ils manipulent ensemble un objet virtuel, l'utilisateur pouvant ressentir les efforts physiques de l'humain virtuel via une interface à retour d'effort. Au niveau cognitif, l'utilisateur et l'humain virtuel peuvent dialoguer : par exemple, lors d'une séance de formation en environnement virtuel, un tuteur virtuel explique à l'utilisateur/apprenant comment faire une tâche.



Figure 1. 10 : Première apparition d'une présentatrice virtuelle sur internet.

[Ana00].

Conclusion

La simulation du comportement de navigation d'humanoïdes virtuels pose plusieurs problèmes et s'avère être un domaine pluridisciplinaire, ceci en plusieurs sens : d'une part en terme de domaines de recherche où les informations issues de la socio-psychologie sont importantes à prendre en compte pour la modélisation informatique du comportement, d'autre part en terme informatique où elle touche à plusieurs problèmes tels que les différents types d'agents , la représentation de l'environnement. Pour obtenir une architecture efficace alliant à la fois autonomie et rapidité.

Nous avons présenté dans ce chapitre un état de l'art sur les éléments essentiels utilisés dans l'animation comportementale et plus précisément la simulation d'humains virtuels et les différentes méthodes utilisées pour représenter son environnement virtuel.

CHAPITRE 2

ANIMATION COMPORTEMENTALE

Introduction

L'animation comportementale vise à doter des personnages artificiels autonomes de comportements leur permettant d'évoluer dans un environnement virtuel. Cette discipline de l'animation est largement employée par l'industrie du cinéma, du jeu vidéo mais aussi pour de nombreuses applications de réalité virtuelle telles que les visites virtuelles de lieux touristiques, les communautés virtuelles ou les simulations d'entraînement.

Dans ce chapitre, on s'intéresse à l'animation comportementale par la présentation de l'observation du comportement et l'animation comportementale des piétons ensuite nous présentons l'algorithme de parcours A étoile.

2.1 Observation du comportement des piétons

2.1.1 Environnement

L'environnement tient un rôle capital dans la plupart des systèmes de simulation d'agent virtuels, il n'est pas possible de construire des agents sans prendre en compte leur environnement, ni les relations qui existent entre eux. Que serait un environnement sans agent et des agents sans environnement ?

D'une manière générale, un environnement est un espace, muni d'une topologie ou non, possédant des objets passifs ou actifs, et des lois d'interactions entre lui-même et ses objets. La caractéristique principale d'un environnement est son aspect situé ou non. Un environnement est situé s'il est constitué à partir d'un espace possédant une métrique, et dans lequel les objets et les agents ont une position en termes de coordonnées. Il n'est pas situé si la notion d'espace et de positionnement dans l'espace n'est pas présentée. Les systèmes classiques à base de règles peuvent être considérés comme des environnements non-situés où les objets les constituant, les règles, ne sont pas considérés comme des entités ayant un paramètre de positionnement et évoluant dans un espace.

Les systèmes multi-agents sont développés généralement avec des environnements situés. Ceux-ci peuvent être à nouveau divisés en deux sous-ensembles. La seconde caractéristique est la continuité de l'environnement, distinguant alors les environnements continus des environnements discrets (Lattaud, 1998) :

- Un environnement est continu si, quels que soient deux points de cet environnement, il existe au moins un point entre eux. Même si d'un point de vue informatique, la notion de continuité stricte est actuellement impossible, un environnement continu peut être construit en utilisant une métrique basée sur un système vectoriel de nombres réels.
- Un environnement est discret s'il est découpé en cases, carrées ou hexagonales par exemple. La métrique est donc basée sur les nombres entiers, les agents ne peuvent alors plus se déplacer en fonction de vecteurs de nombres réels mais en fonction de valeurs discrètes entières.

2.1.2 Interactions avec l'environnement

En temps normal, lors de ses déplacements que ce soit dans un bâtiment ou à l'extérieur, le piéton fait sans cesse appel à ses sens, et en particulier à la vue pour s'orienter et adapter sa trajectoire (Patla, 1997). Un piéton surveille son environnement de façon permanente et naturelle. La zone balayée prend la forme d'une ellipse dont les dimensions s'adaptent aux conditions de trafic et aux obstacles (Hoogendoorn, 2004).

Pour se déplacer, un piéton requiert un minimum d'espace (Hoogendoorn, 2004) :

- un pas mesure environ 0,65 m.
- il avance de 2,05 pas/s pour une vitesse moyenne de 1,33 m/s.
- la distance avec le prédécesseur est d'au moins 1 m.

Le besoin d'espace se manifeste pour l'environnement longitudinal du piéton, c'est-à-dire devant lui, et dans son voisinage latéral. Les piétons qui se déplacent sur un trottoir observent une distance de confort ou de sécurité vis-à-vis :

- des murs et petits obstacles : 0,25 m.
- des maisons et obstacles de plus grandes dimensions : 0,45 m.
- et des routes : 0,35 m.

La distance entre les piétons diminue quand la densité augmente. La vitesse, l'espace vital et la longueur des pas diminuent aussi. L'espace vital d'un piéton immobile est de $0,15 \text{ m}^2$, soit une densité maximale de 6,6 piétons immobiles par m^2 . Lorsque la densité de piétons vaut 5,4 piétons par m^2 , il n'y a plus de place pour marcher.

L'évitement entre les piétons se fait le plus souvent au dernier moment en effectuant un pas latéral. Dans ce cas, les piétons ont tendance à se décaler à droite. Cependant, les piétons semblent réticents à céder le passage unilatéralement à un autre piéton (Hoogendoorn, 2004). Lorsque des piétons se retrouvent à proximité, il y a une coopération entre eux pour éviter les collisions. Cette coopération prend la forme du protocole suivant (Wakim, 2005) :

- émission de signe critique.
- reconnaissance et acquittement.
- interaction.

2.1.3 Déplacements des piétons

En moyenne, la trajectoire d'un piéton qui traverse la route sur un passage piéton sera plutôt perpendiculaire à la route, alors qu'un piéton qui traverse hors d'un passage piéton aura tendance à arrondir les angles et à prendre la diagonale (Figure 2.1).

Les différentes phases d'une traversée de route sont typiquement :

- «Démarrage» en descendant du trottoir ou au début d'une nouvelle voie, avec augmentation de la vitesse jusqu'à atteinte de la vitesse voulue.
- maintien de la vitesse de croisière.
- éventuellement, avant de traverser une nouvelle voie véhicule : accélération, ralentissement, ou changement de direction doux.
- « Fin de traversée » : léger ralentissement, changement de direction.

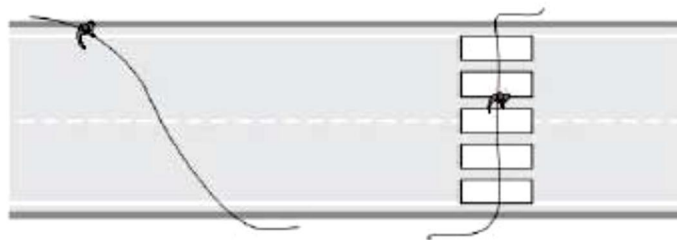


Figure 2. 1 : Trajectoires de piétons traversant une route, hors et sur passage piéton [Mou11].

Un piéton se déplace en adoptant une des trois allures suivantes : marche, course lente (jogging en anglais) ou course rapide (running en anglais).

Pour piloter son déplacement, le piéton a la possibilité d'agir sur les éléments suivants

(Wakim, 2005) :

- le piéton marche, « normalement », vers l'avant (déplacement longitudinal). Il joue sur : la longueur des pas, leur cadence et/ou la direction suivie,
- d'autres déplacements sont envisageables : les pas sur le côté, l'arrêt, ou la marche arrière.

2.2 L'animation comportementale

L'animation(ou simulation) comportementale est une branche de l'animation ayant pour objectif la production de comportements pour les acteurs de l'animation. Pour bien comprendre cette nécessité, il est impératif de cerner préalablement les enjeux actuels de l'animation.

L'animation par ordinateur est elle-même issue de l'informatique graphique, une discipline qui a pour but de produire des images grâce à des moyens informatiques. De la même manière que la photographie a donnée naissance au cinéma, l'animation est apparue d'es que les moyens techniques se sont révélés suffisants. Aujourd'hui, elle occupe une place prépondérante dans la création de films, de jeux vidéo, d'applications de réalité virtuelle (visites virtuelles de villes ou de lieux historiques, communautés virtuelles, simulations d'entraînement, etc.).



Figure 2. 2 : Quelques exemples d'applications de l'animation [Dav08].

(a) L'environnement virtuel du jeu en ligne « Second Life ». (b) Le jeu vidéo « Les Sims 2 » de Electronic Arts. (c) Les personnages du « musée virtuel » de la Cité des Sciences et de l'Industrie (Paris) utilisent l'architecture HPTS.). (d) Film « Batman Returns » de Tim Burton en 1992.

L'animation comportementale répond à ces deux besoins en proposant des méthodes et des outils pour créer des créatures ou des personnages réalistes tout en soulageant l'animateur : Les individus virtuels et Les environnements de l'animation.

2.3 Le comportement d'individu

Bien que la capacité à se mouvoir soit un comportement primordial de l'être humain, ce qui le caractérise vraiment est sa capacité à raisonner. En effet, un individu est capable de symboliser mentalement les actions qu'il veut exécuter, pour ensuite les organiser. Il est ainsi capable d'élaborer un plan comportemental lui permettant d'ordonner ses actions, et de gérer leurs liens de dépendance. Ce raisonnement va ainsi avoir un impact direct sur son déplacement.

Les tâches mentales prépondérantes concernant l'action d'un individu sur son environnement sont sans doute les tâches d'interaction. Celles-ci définissent la manière dont l'individu va pouvoir affecter d'autres entités, et ainsi modifier l'état du monde qui l'entoure.

Ces tâches d'interaction entre elles même dans un processus mental complexe, pouvant être directement désirées par l'individu, ou symboliser une étape de son raisonnement [Seb07].

Nous entendons par comportements d'individus ceux qui permettent à l'individu de se déplacer dans l'environnement. Par mais ces comportements en suite les comportements suivants :

2.3.1 Comportement de recherche et fuite

La recherche et la fuite sont deux comportements très simples qui déplacent un individu vers ou loin d'une position de cible avec une vitesse constante.

La différence entre la recherche et la fuite est que la recherche est l'acte d'orienter le caractère vers une position indiquée dans l'espace global. Ce comportement ajuste le caractère de sorte que sa vitesse soit radicalement alignée vers la cible. D'une manière simple, la fuite peut être considérée comme étant l'inverse de la recherche. Elle agit en orientant le caractère de sorte que sa vitesse soit radicalement alignée loin de la cible. La vitesse désirée se dirige dans la direction opposée [Rey99].

2.3.2 Comportement d'arrivée

Le comportement d'arrivée est une extension du comportement de recherche. De même que pour le comportement de recherche, il est utilisé pour orienter l'individu vers une cible indiquée. La différence importante se résume dans la manière selon laquelle l'individu atteint la destination.

Le comportement de recherche fait que notre individu atteint la cible à pleine vitesse. Il se déplace en fait davantage dans la direction actuelle et va ainsi générer plutôt un mouvement de danse semblable au comportement d'une mite autour d'une source lumineuse.

Cependant, le comportement d'arrivée doit être tel qu'il y ait un ralentissement commandé du véhicule conformément au cahier des charges édicté par l'utilisateur, le mouvement doit s'arrêter à la position désirée (Figure 2.3). Ceci fait provoquer le ralentissement du véhicule selon sa distance actuelle à la destination et le résultat est un véhicule qui s'arrête au niveau de la cible indiquée [Rey99].

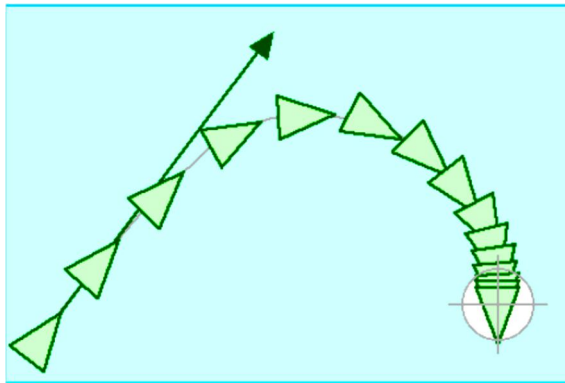


Figure 2. 3 : Le comportement d'arrivée [Rey99].

2.3.3 Comportement d'évitement d'obstacles

Le comportement d'évitement d'obstacles donne à un acteur la capacité de manœuvrer dans un environnement encombré en esquivant autour des obstacles. Il y a une distinction importante entre l'évitement d'obstacles et le comportement de fuite.

La fuite a pour conséquence d'orienter l'acteur loin d'un emplacement donné, tandis que l'action d'évitement d'obstacles agit seulement quand un obstacle se trouve directement devant cet acteur. Par exemple, si un individu se déplace sur une trajectoire parallèle à un mur, l'évitement d'obstacles ne prendrait aucune mesure corrective de direction, mais la fuite essaierait de s'éloigner du mur, en suivant la perpendiculaire à ce mur.

Le but du comportement est de garder un cylindre imaginaire de l'espace libre devant l'acteur. Le cylindre se trouve le long de l'axe vers l'avant l'acteur, à un diamètre égal à la sphère de bondissement de l'acteur, et s'étend du centre de l'acteur pour une distance basée sur la vitesse et l'agilité de l'acteur.

Le comportement d'évitement d'obstacles considère chaque obstacle alternativement et détermine si ces obstacles possèdent des intersections avec le cylindre. En localisant le centre de chaque obstacle sphérique, le test d'intersection avec le cylindre est ainsi effectué très rapidement [Rey99].

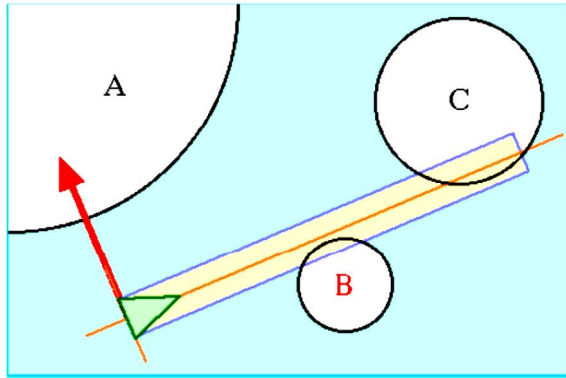


Figure 2. 4 : L'évitement d'obstacle [Rey99].

Une note finale concernant l'interaction d'évitement d'obstacles et de rechercher de but. D'une façon générale nous nous intéressons seulement aux obstacles qui sont localisés entre l'acteur et le but à atteindre.

2.3.4 Comportement de suivi de chemin

Ce type de comportement permet à un acteur de s'orienter le long d'une voie d'accès prédéterminée, telle qu'une chaussée, un couloir ou un tunnel. C'est en fait très distinct de l'action de contraindre un individu à suivre d'une manière rigide une voie d'accès tel que le roulement d'un train le long d'un rail, par exemple. Le comportement de suivi d'une voie d'accès est plutôt destiné à la production de mouvements tels que des personnes empruntant un couloir : les différentes voies d'accès restent proches et sont souvent parallèles à l'axe du couloir, elles sont néanmoins libres [Rey99].

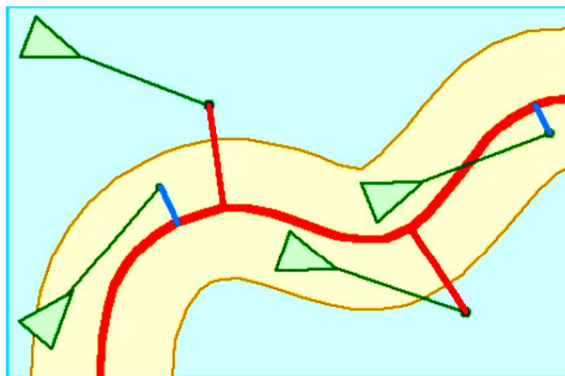


Figure 2. 5 : Le comportement de suivre de chemin [Rey99].

Autrement dit, l'acteur vire loin de la voie d'accès, ou en est trop loin de celle-ci. Pour l'orienter en arrière vers la voie d'accès, le comportement de recherche est exploité pour une orientation en arrière vers la projection de la voie d'accès de la future position prévue, de la même manière que pour l'action d'évitement [Bee90].

2.3.5 Evitement de collision non-alignée

Le comportement d'évitement de collision non alignée se doit de prévenir les collisions entre des acteurs se déplaçant dans une direction arbitraire. Considérez votre propre expérience de marche à pied à travers une place ou une entrée pleine d'autres marchants :

L'évitement de collisions implique la prévision de collisions potentielles et le changement de votre direction et de votre vitesse pour les prévenir [Rey99].

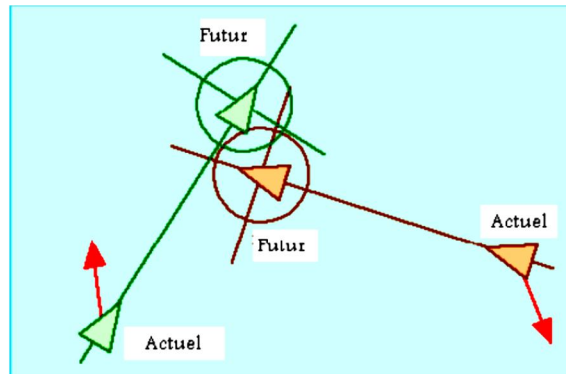


Figure 2. 6 : Le comportement d'évitement de collisions non-alignées [Rey99].

2.4 L'algorithme de parcours A étoile (A*)

Assez flexible et pouvant être employé dans un grand choix de contextes, l'algorithme A* a toujours été un choix des plus populaires.

De même que pour d'autres algorithmes de recherche graphique, l'algorithme A* permet d'effectuer des recherches virtuellement et ce dans un secteur énorme du domaine de recherche. Il est semblable à l'algorithme de Dijkstra, lequel pouvant être employé pour trouver le plus court chemin. Il est également proche de l'algorithme BFS (*Breadth First Search*), lequel peut employer une heuristique pour pouvoir se guider. Dans les cas les plus simples, il offre des vitesses de convergence proches de celles associées à l'algorithme BFS.

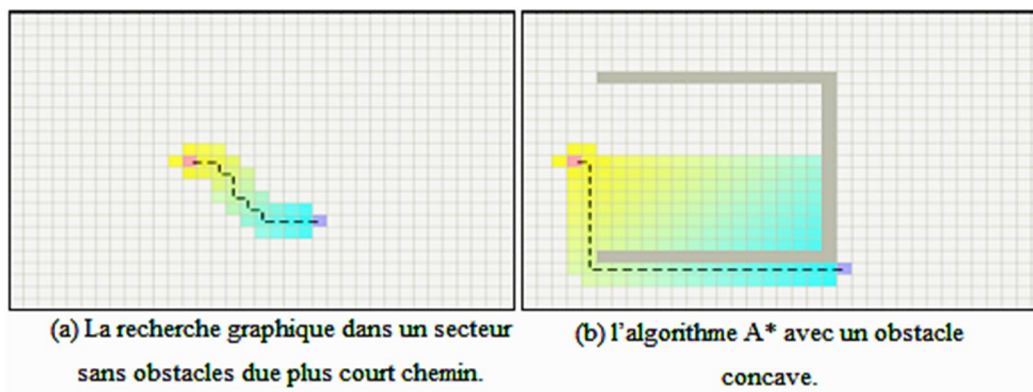


Figure 2. 7 : L'algorithme A*.

Le secret lié au succès de l'algorithme A*, consiste dans le fait qu'il combine des morceaux d'information, les même que celles utilisées par l'algorithme de Dijkstra (favorisant le sommet qui est proche du point de départ), ainsi que l'information exploitée par l'algorithme BFS (favorisant le sommet qui est proche du but).

Dans la terminologie standard employée dans un algorithme A*:

$g(n)$: représente le coût du chemin du point de départ à n'importe quel sommet n.

$h(n)$: représente le coût heuristique estimé du sommet n au but.

Dans les diagrammes suscités, h (en jaune) représente les sommets qui sont loin du but, et g (bleu) représente les sommets loin du point de départ.

L'algorithme A*équilibre les deux tout en se déplaçant du point de départ au but. A chaque pas de la boucle principale, il examine le sommet n qui possède le coût le plus bas par l'utilisation d'une fonction de la forme $f(n) = g(n) + h(n)$ selon les étapes suivantes :

1. Ajouter le point de départ à une "liste ouverte».
 2. Répéter les instructions suivantes :
 - a. Rechercher la cellule ayant le plus faible coût f dans la "liste ouverte", qui devient la cellule en cours.
 - b. Eliminer la cellule en cours de la "liste ouverte" et l'ajouter à la "liste fermée".
 - c. Pour les 8 cellules adjacentes à la cellule en cours, faire :
 - Si on ne peut pas traverser la cellule, on l'ignore.
 - Si elle n'est pas dans la "liste ouverte", on l'y ajoute. La cellule en cours devient le parent de cette cellule. On calcule les coûts f, g et h de cette cellule.
 - Si elle est déjà dans la "liste ouverte", on teste si le chemin passant par la cellule en cours est le meilleur, en comparant les coûts g.
- Un coût g inférieur : signifie un meilleur chemin.
- Si c'est le cas, on change le parent de la cellule pour devenir la cellule en cours, en on recalcule les coûts f et g. Si on conserve une "liste ouverte" triée par coût f, la liste doit être retirée à ce moment-là.
 - d. On s'arrête lorsque :
 - La cellule de destination est ajoutée à la "liste ouverte".
 - On ne trouve pas la cellule de destination et la "liste ouverte" est vide.
 3. Enregistrez le chemin. En partant de la cellule de destination, remontez d'une cellule à son parent jusqu'à atteindre la cellule de départ.

Conclusion

Nous avons essayé de définir et de décrire dans ce chapitre «l'aspect d'autonomie» des piétons et c'en termes d'individus autonomes et d'actions improvisées dont l'aspect comportemental peut être généré de différentes manières, et nous sommes penchés tout au long de ce chapitre sur la caractérisation et la reproduction de la circulation humaine.

Ces domaines d'applications variés, justifient les recherches effectuées en animation comportementale, et particulièrement le besoin d'établir un lien avec les études effectuées sur le comportement humain. Les modèles proposés, pour permettre d'obtenir un réalisme suffisant, doivent généralement prendre en compte les caractéristiques propres au comportement humain.

Nous avons également présenté, en fin de chapitre, un algorithme de recherche graphique A étoile, pour la génération de mouvements à partir de comportements simples ou élémentaires.

CHAPITRE 3

FOULE D’HUMAINS VIRTUELS ET LA LOGIQUE FLOUE

Introduction

La simulation de foule est un sujet d’actualité dans la recherche en infographie et reste encore aujourd’hui un sujet très ouvert. Ainsi, de nombreuses recherches traitent encore ce vaste sujet, et a pour but de générer facilement les mouvements, action est comportements d’un grand nombre d’individus, pour de produire une animation dont l’aspect semble réaliste.

Nous avons exposé dans la première partie de ce chapitre les propriétés des humains virtuels et les deux approches de foule, ensuite dans la seconde partie, nous avons présenté en détail la logique floue.

3.1 Propriétés des humains virtuels

L’objectif majeur de la modélisation des comportements des acteurs est de construire des individus intelligents autonomes virtuels avec adaptation, perception et mémoire, capable d’agir librement et avec émotion, d’être conscient et imprévisible (Figure 3.1).

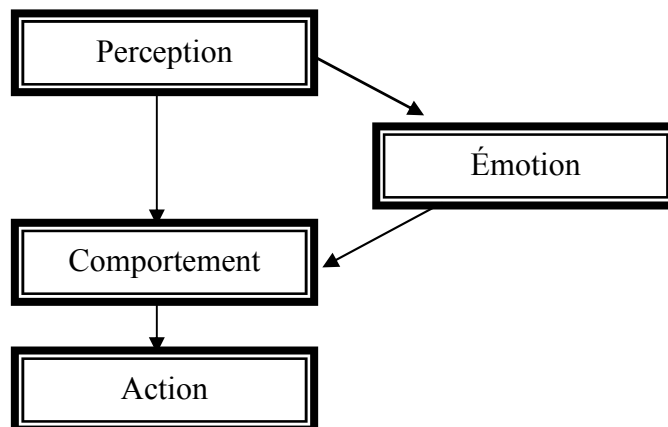


Figure 3. 1 : Structure du Modèle comportemental.

3.1.1 La perception

La perception est définie comme la conscience des éléments dans l’environnement à travers des sensations physiques. Il est réalisé en équipant les agents avec des détecteurs visuels, tactiles et auditifs ainsi ils simulent le comportement quotidien humain (aspect visuel, mouvement, réaction, ...). Le sous-système perceptuel le plus important est le système visuel.

Une approche basée sur la vision est idéale pour le modelage d'une animation comportementale. Elle offre une approche universelle pour le passage d'information de l'environnement à l'acteur dans le contexte de recherche de chemin. A un niveau plus haut, nous pouvons décomposer la perception comme suggérée par. La perception d'un acteur peut être limitée aux objets et à d'autres acteurs dans le voisinage. Mais cela limite le nombre de comportements possibles, parce que seules la présence et les caractéristiques d'un objet ou d'un acteur sont impliquées dans la sélection d'un comportement. Les actions des autres acteurs ne sont pas prises en considération. Le module de perception produit trois types de perception : la perception de la présence d'objets et d'acteurs, la perception des actions d'acteurs et la perception d'acteurs exécutant des actions sur des objets [Don04].

3.1.2. L'émotion

L'émotion peut être définie comme l'aspect affectif de la conscience : Un état de sentiment, une réaction psychique et physique (comme la colère ou la crainte), subjectivement expérimenté comme un sentiment fort et physiologiquement qui augmente les changements préparant le corps pour une action vigoureuse immédiate.

Les acteurs doivent être capables de répondre, avec émotion à leur situation et agissant physiquement à cela.

Les émotions visibles fournissent des designers avec un moyen direct pour affecter à l'utilisateur un état émotionnel propre à lui. Les acteurs sont donc équipés d'un modèle informatique simple de comportement émotionnel, qui est lié au comportement comme les expressions de visage qui peuvent être employées pour influencer leurs actions.

Une émotion est la réaction d'une personne à une perception. Celle-ci est amenée à répondre par une expression du visage, un geste, ou à choisir un comportement spécifique.

Une émotion arrive entre une perception et la réaction suivante. Deux personnes différentes peuvent avoir des réactions différentes à la même perception, selon la façon dont ils sont affectés par cette perception.

Les émotions sont causées par la réaction aux objets, les actions d'individus et les événements.

Les émotions causées par des événements peuvent être classées selon trois types d'émotions :

- Les émotions causées par des événements potentiels.
- Les événements affectant le destin d'autres.
- Les événements affectant le bien-être de l'acteur.

Chaque classe est caractérisée par des conditions d'apparition pour chacune de ses émotions et variables affectant son intensité. Les émotions auxquelles est soumis un acteur sont dues à sa perception [Don04].

3.1.3. Le comportement

Le comportement est souvent défini comme la voie dans lequel les animaux et les humains agissent. Il est usuellement décrit en termes de langage naturel qui a la signification sociale, psychologique ou physiologique, mais qui n'est pas nécessairement facilement réductible au mouvement d'un ou des muscles.

Le comportement est aussi la réponse d'un individu, groupe, ou espèce à son environnement.

Le comportement ne réagit pas seulement à l'environnement, mais inclut aussi le flux d'information par lequel l'environnement agit sur la créature vivante aussi bien sur la façon dont la créature code et emploie cette information.

Le comportement peut être décrit d'une façon hiérarchique. Le modèle comportemental décompose un comportement en comportements plus simple qui peuvent être décomposés plus loin.

Chaque niveau de cette décomposition hiérarchique contient un ou plusieurs comportements exécutés séquentiellement, ou bien concurremment. Un niveau de la hiérarchie contenant plusieurs comportements pour être exécuté séquentiellement, appelé comportement. Chaque comportement d'un ordre de comportement est appelé, cellule comportementale.

La cellule comportementale contient les comportements qui sont exécutés chacun concurremment, ou exclusivement quand les règles d'inhibition sont spécifiées. Les comportements contenus dans une cellule comportementale sont de nature composée ou élémentaire.

Un comportement permet une décomposition récursive dans la hiérarchie. Un comportement élémentaire est placé au fond de la décomposition hiérarchique et encapsule un comportement spécialisé qui contrôle directement une ou plusieurs actions. Un comportement est exécuté récursivement : en fait la hiérarchie du comportement permet d'exécuter les comportements élémentaires des entités à chaque niveau de la structure de la hiérarchie.

Pour contrôler le comportement global d'un acteur, on exploite une pile de comportements. Au début de l'animation, l'utilisateur pousse un ordre de comportements dans la pile de l'acteur. À la fin du comportement actuel le système d'animation passe le comportement suivant de la pile et l'exécute. Ce processus est répété jusqu'à ce que la pile de comportement de l'acteur se vide. Avec Ce contrôle de comportement employant une pile, un acteur devient plus autonome et crée ses propres sous buts en exécutant le scénario original [Don04].

3.1.4. L'action

Basée sur l'information perceptuelle, le mécanisme comportemental d'un acteur détermine les actions à exécuter. Les actions peuvent avoir plusieurs degrés de complexité.

Un acteur peut se développer dans son environnement ou bien agir réciproquement avec l'environnement ou encore communiquer avec d'autres acteurs.

Les actions sont exécutées en employant une architecture de mouvement commune. Le module d'action gère l'exécution des actions employées par un comportement en animant un modèle d'homme générique basé sur une hiérarchie de nœud. Il permet l'exécution simultanée ou séquentielle d'actions en gérant des transitions lisses entre des actions finales et des actions amorçant. Une boucle comportementale conduit l'animation, son rôle est de mettre à jour l'état du monde virtuel.

A chaque itération, le temps est incrémenté, le monde virtuel est mis à jour avec en particulier une mise à jour de l'état de chaque objet et acteur. Dans le cas d'un acteur, la perception est d'abord exécutée, après ses émotions sont produites avant que son comportement et ses actions ne soient exécutés [Don04].

3.1.5. La mémoire

La mémoire est d'habitude définie comme le pouvoir ou le processus de reproduction ou de rappel de ce qui a été appris et conservé, particulièrement par les mécanismes associatifs.

La mémoire est aussi le dépôt pour des informations apprises et à conserver, générées à partir de l'activité d'un organisme ou d'une expérience.

La mise en œuvre de la mémoire pour un acteur n'est pas très complexe, comme la mémoire est déjà un concept clef en informatique. Par exemple, dans, les auteurs proposent une troisième mémoire visuelle globale qui permet à un acteur de retenir l'environnement, de le percevoir et de s'adapter à ses changements [Don04].

3.2 Les deux approches de foule

3.2.1 Modèles macroscopiques

Les modèles macroscopiques sont les premiers modèles de simulation de foule à être apparus. Ceci est notamment dû à leur faible consommation en termes de ressources de calcul.

Ainsi, ils simulent un grand nombre de piétons sans s’intéresser à leur comportement individuel. Ces modèles sont généralement dédiés aux simulations d’évacuation de bâtiments et à la mise en place des conditions de sécurité qui sont liées à l’évacuation. Ils ont aussi pour but de simuler de manière réaliste les phénomènes macroscopiques sans que l’individu ne soit représenté dans le modèle.

3.2.1.1 Modèles statistiques

Les modèles statistiques s’intéressent directement aux débits de piétons. Certains sont basés sur la dynamique des fluides. Henderson [Hen71] propose un modèle gazeux permettant de simuler des foules de piétons de faible densité et assimile les états de déplacement (arrêt, marche, course) aux différents niveaux d’énergie des gaz. Archea [Arc79] propose quant à lui un modèle hydraulique pour simuler des foules de forte densité.

Plusieurs modèles s’intéressent aux débits de piétons quant à l’évacuation de bâtiments, et notamment le débit de piétons que peuvent permettre les portes et les escaliers. Ceux-ci proposent des formules calculant le temps d’évacuation en fonction de divers paramètres comme le nombre de piétons, la largeur des zones de circulation, la pente ou encore l’espace occupé par une personne.

Plus récemment, Colombo et Rosini [Rcr05] proposent un modèle qui s’inspire de modèles de trafic routier. Il s’appuie sur deux suppositions : le nombre total de piétons reste constant au cours de la simulation et la loi de vitesse est fonction de la densité. Leur modèle introduit la notion de débit de piétons à très haute densité, dans des situations exceptionnelles de type panique où le débit serait quasi-nul en situation normale.

De par leur nature, ces modèles macroscopiques sont capables de simuler des foules de piétons avec un très faible coût calculatoire et sont capables d’intégrer facilement des observations réelles. Ils ont en revanche besoin d’informations issues d’observations réelles pour fonctionner correctement et sont peu adaptables à de nouvelles situations. De plus, ils ne simulent pas les piétons à l’échelle individuelle.

3.2.1.2 Modèles de flots

Les modèles de flots sont inspirés de la dynamique des fluides. Des champs de potentiel agissent sur les piétons qui sont considérés comme des particules soumises à des champs.

Dans le cadre de la simulation de fluides, Chenney [Che04] représente des champs de vitesse à travers un carrelage de flots. Chaque carreau contient son propre champ et leur assemblage produit des flots plus grands. Les arêtes et les coins des carreaux sont élaborés de manière à assurer une continuité entre les carreaux. Non seulement des fluides peuvent être dirigés ainsi mais cette méthode peut également être appliquée aux foules. Les carreaux sont stationnaires (leur champ n’est pas modifié au cours du temps) mais peuvent générer des flots dynamiques en combinant le carrelage en fonction du temps.

Inspiré des travaux de Hughes [Hug02] et Chenney [Che04], A. Treuille [Tep06] ont simulé des foules de piétons à partir d’un modèle qui s’appuie sur les quatre hypothèses suivantes :

- chaque personne tente d’atteindre un objectif géographique.
- les personnes cherchent à marcher à la vitesse la plus élevée possible.
- il existe des endroits dits inconfortables que les piétons vont chercher à éviter.
- le piéton cherchera à emprunter le chemin le moins “coûteux”, en fonction de sa longueur, de son temps de parcours et de son niveau d’inconfort.

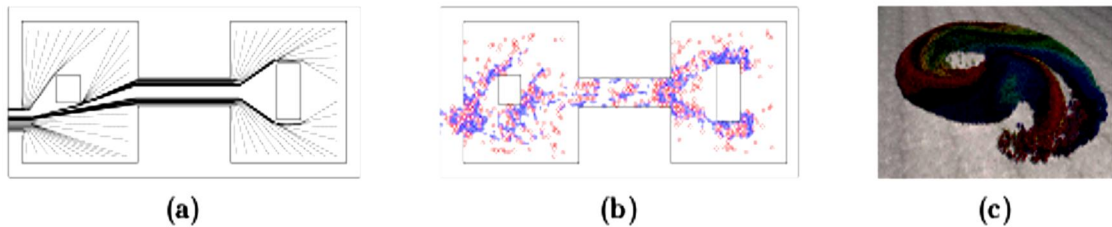


Figure 3. 2 : Modèles de flots.

a). Représentation du flot des champs de vitesse dans [MV07] (b). Exemple d’évacuation (c). Exemple de simulation dans [NGCL09] : 10000 individus placés en cercle et devant.

3.2.2 Modèles microscopiques

L’approche microscopique consiste à identifier les interactions entre chaque individu et son voisinage pour ensuite influencer le comportement de l’individu. Ces modèles se différencient par leur manière de prendre en compte les informations provenant du voisinage d’un piéton et par la réaction de l’individu compte tenu de ces informations.

3.2.2.1. Automates cellulaires

Certains modèles, appelés automates cellulaires, sont basés sur une discrétisation de l'espace en cellules, souvent carrées mais parfois hexagonales ou octogonales. Leur principe repose sur le fait que chaque piéton occupe une cellule et ne peut se déplacer vers une autre cellule que si celle-ci est libre.

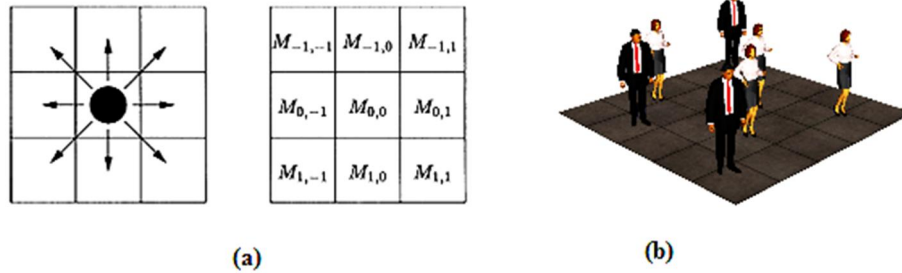


Figure 3.3 : Automates cellulaires.

(a). Matrice 3*3 des probabilités de la prochaine position d'un piéton. (b). Décomposition de l'espace en grille régulière à 2 dimensions.

3.2.2.2 Modèles à base de règles

Les modèles à base de règles ont été introduits par Reynolds [Rey87] qui, en travaillant sur des animations cinématographiques, s'est intéressé à la simulation de foules d'animaux comme les bancs de poissons, les troupes de mammifères ou les nuées d'oiseaux. Les membres d'un groupe sont considérés comme des individus et sont ainsi traités un à un. Chacun de ces individus est soumis à des règles : l'évitement de collisions, l'asservissement à la vitesse de ses voisins, et la tendance à rester au sein du groupe. Il propose ensuite [Rey99] un ensemble de règles qui décrivent différents comportements humains de base tels que l'atteinte d'une cible, l'évitement de collision, la poursuite, la cohésion de groupe ou le suivi de leader. Une combinaison de ces comportements de base permet de créer des comportements plus complexes.

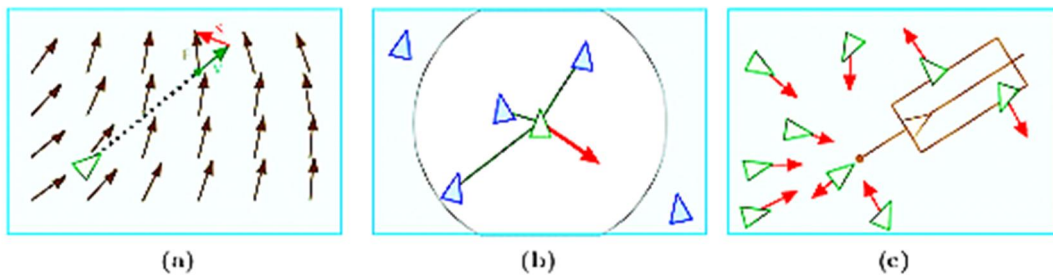


Figure 3.4 : Exemples de règles de comportement proposées par Reynolds [Rey99].

(a)Alignement de flots. (b) Séparation. (c) Suivi de chef.

3.2.2.3 Modèles à base de forces

En faisant une analogie avec la physique newtonienne, les modèles de forces considèrent qu'un piéton est soumis à des forces d'attraction et de répulsion qui agissent sur son accélération suivant la deuxième loi de Newton

$$\sum F = m \cdot a. \quad (1)$$

Où : $\sum F$ est la somme des forces appliquées à un corps de masse m et a son accélération.

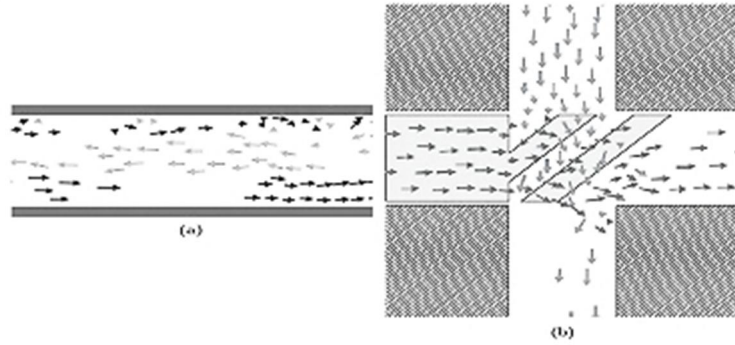


Figure 3. 5 : Simulations de modèles de forces.

(a) Formation de files dans un couloir. (b) Formation de lignes de flots diagonales à un croisement.

3.2.2.4 Modèles géométriques

Alors que les modèles basés forces ne prennent en compte que la position des différents obstacles, les modèles géométriques prennent également en compte la vitesse relative de ces obstacles. Ces modèles sont ainsi considérés comme prédictifs car ils sont capables de déterminer une future collision si aucun changement n'est apporté à la trajectoire d'un piéton.

Ils déterminent ainsi quelles vitesses futures permettront à un piéton d'éviter une collision avec les obstacles qui l'entourent.

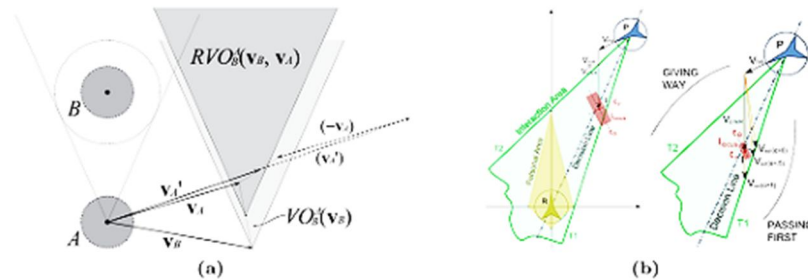


Figure 3. 6 : modèles géométriques.

Adaptation de la vitesse en fonction d'un obstacle mobile dans des modèles géométriques.

(a). Modèle RVO. (b) Modèle des tangentes.

Les modèles géométriques offrent ainsi un haut niveau de réalisme à l’échelle microscopique et sont capables de reproduire plusieurs phénomènes macroscopiques. Des questions restent posées sur la combinaison des différentes interactions et sur la manière d’incorporer différents facteurs socioculturels. La prédiction effectuée par ces modèles est principalement linéaire et ne considère pas les intentions des autres piétons, qui sont pourtant souvent prises en compte dans les cas réels.

3.2.2.5 Modèles basés vision

Turner et Penn [Tur07], [Tup02] montrent l’importance de la vision et de la perception dans la simulation de foule de piétons, affirmant que des modèles à champs de potentiel par exemple, pourraient fonctionner dans l’obscurité. En s’appuyant sur la théorie de la perception de Gibson [Gib86], ils proposent un modèle basé sur la vision où l’humain choisit une destination à atteindre dans son champ de vision en fonction de son objectif et des différents obstacles et réactualise cette destination au fur et à mesure.

Les modèles basés vision proposent en général un haut niveau de réalisme et introduisent la boucle perception/action dans le comportement de l’individu. Ils souffrent en revanche de problèmes de performance, le lancer de rayon étant un outil très coûteux en termes de calcul. De plus, ces modèles ont encore besoin d’être validés à partir d’observations réelles.

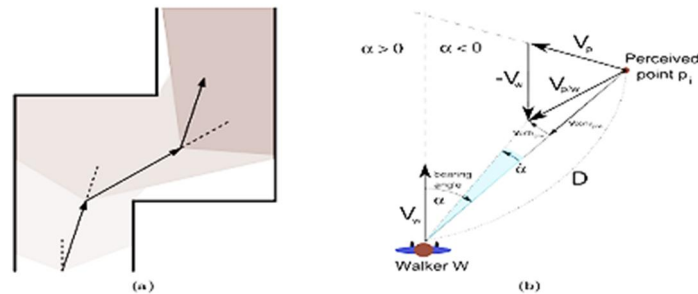


Figure 3. 7 : Modèles basés vision.

3.2.2.6 Modèles basés données

Plusieurs méthodes utilisent des enregistrements vidéos ou de capture de données pour créer des bases de données d’interactions entre piétons [Lcl07], [Coc07]. Une interaction entre piétons est modélisée en recherchant dans la base un exemple similaire à la situation. La principale difficulté réside dans la quantification de la similitude entre une situation et un exemple.

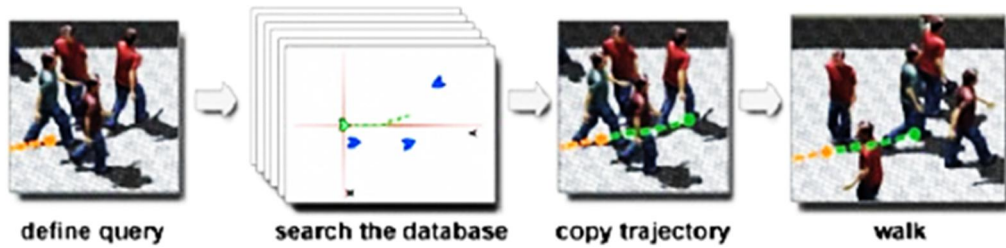


Figure 3. 8 : Processus de résolution d'une interaction dans [Lcl07].

Les modèles basés données sont implicitement réalistes puisqu'ils reproduisent des trajectoires réelles. Des problèmes se posent cependant sur l'enchaînement de ces trajectoires, le comportement des individus et la prise en compte de leur objectif. Ces modèles souffrent également d'un manque de performance et il est nécessaire de disposer d'un exemple proche de chaque situation. De plus, la complétude des données demeure une question ouverte.

3.3 Introduction à la logique floue

De nos jours, la logique floue (en anglais «fuzzy logic») est un axe de recherche important sur lequel se focalisent de nombreux scientifiques. Des retombées technologiques sont d'ores et déjà disponibles, tant dans le domaine grand public (appareils photos, machines à laver,...) que dans le domaine industriel (réglage et commande de processus complexes liés à l'énergie, aux transports, à la transformation de la matière, à la robotique, aux machines-outils) [Pay98]. Dans ce chapitre, nous présentons les éléments de base de la théorie de la logique en insistant sur les contrôleurs à logique floue.

3.3.1 Historique et définition

Depuis longtemps l'homme recherche à maîtriser les incertitudes et les imperfections inhérentes à sa nature. La première réelle manifestation de la volonté de formaliser la prise en compte des connaissances incertaines fut le développement de la théorie des probabilités à partir du XVII siècle. Mais les probabilités ne peuvent maîtriser les incertitudes psychologiques et linguistiques.

Puis la logique floue est apparue en 1965 à Berkeley dans le laboratoire de Lotfi Zadeh avec la théorie des sous-ensembles flous puis en 1978 avec la théorie des possibilités. Ces deux théories constituent aujourd'hui ce que l'on appelle Logique Floue.

La logique floue permet la formalisation des imprécisions dues à une connaissance globale d'un système très complexe et l'expression du comportement d'un système par des mots. Elle permet donc la standardisation de la description d'un système et du traitement de données aussi bien numériques qu'exprimées symboliquement par des qualifications linguistiques.

3.3.2 Logique classique et logique floue

Dans la logique classique, les variables gérées sont Booléennes. C'est à dire qu'elles ne prennent que deux valeurs 0 ou 1. La logique floue a pour but de raisonner à partir de connaissances imparfaites qui opposent résistance à la logique classique. Pour cela la logique floue se propose de remplacer les variables booléennes (Ex : 1 ou 0, vrai ou faux) par des variables flous (Ex : valeur dans l'intervalle $[0,1]$, ratios : 75% Jeune et 25% Très jeune et 0% Agé).

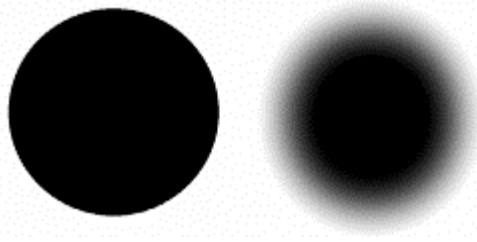


Figure 3. 9 : Représentation graphique d'un ensemble classique et d'un ensemble flou.

3.4 Sous-ensembles flous

Dans la théorie ensembliste classique, l'appartenance d'un élément à un sous-ensemble est définie par une valeur logique standard : 1 si l'élément appartient au sous-ensemble, 0 sinon.

Dans la théorie floue, un élément peut appartenir en partie à un sous-ensemble : son degré d'appartenance est décrit par une valeur comprise entre 0 et 1.

$$\text{Théorie classique : } \mu(x) = \begin{cases} 1 & \text{si } x \in A. \\ 0 & \text{sinon.} \end{cases} \quad (2)$$

$$\text{Théorie floue : } \mu(x) \in [0, 1]. \quad (3)$$

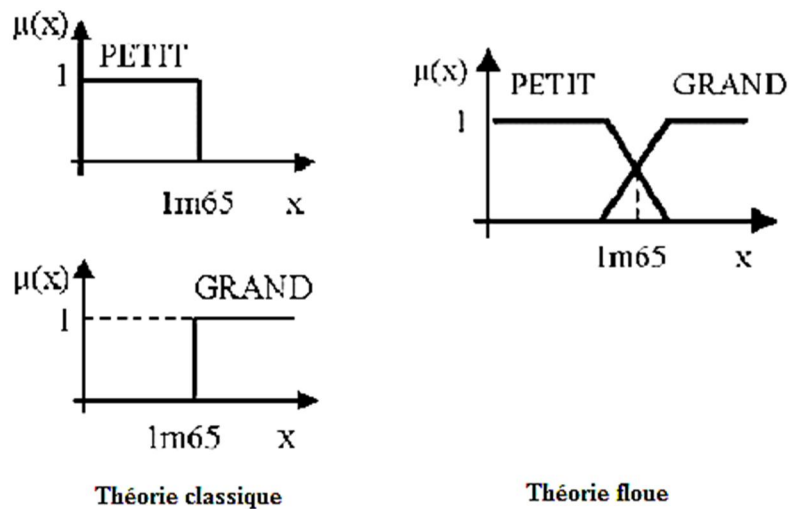


Figure 3. 10 : Classification des tailles d'un humain en deux ensembles.

3.4.1 Définition

Un sous-ensemble flou A dans un univers du discours X est caractérisé par sa fonction d'appartenance $\mu_A(x)$ qui associe à chaque élément x de X une valeur dans l'intervalle des nombres réels $[0, 1]$:

$$\mu_A(x) \in [0, 1]. \quad (4)$$

Ainsi un sous-ensemble flou A dans X peut être représenté par un ensemble de couples ordonnés

$$A = \{(x, \mu_A(x)) / x \in X\}. \quad (5)$$

3.4.2 Opérations de base sur les sous-ensembles flous

Supposons que A et B sont deux sous-ensembles flous définis dans un univers du discours X par les fonctions d'appartenance μ_A et μ_B . On peut définir des opérations ensemblistes telles que l'inclusion, l'intersection, l'union et le complément grâce à des opérations sur les fonctions d'appartenance.

Inclusion : A est dit inclus dans B , propriété que l'on note $B \subseteq A$, si tout élément x de X qui appartient à A appartient aussi à B avec un degré au moins aussi grand :

$$\forall x \in X : \mu_A(x) \leq \mu_B(x) \quad (6)$$

Intersection : L'intersection de A et B , que l'on note $A \cap B$, est le sous-ensemble flou constitué des éléments de X affectés du plus petit des deux degrés d'appartenance μ_A et μ_B :

$$\forall x \in X : \mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)) \quad (7)$$

Union : l'union de A et B , que l'on note $A \cup B$, est le sous-ensemble flou constitué des éléments de X affectés du plus grand des deux degrés d'appartenance μ_A et μ_B :

$$\forall x \in X : \mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x)) \quad (8)$$

Complément : Le complément de A , que l'on note A^C , est le sous-ensemble flou de X constitué des éléments x lui appartenant d'autant plus qu'ils appartiennent peu à A :

$$\forall x \in X : \mu_{A^C}(x) = 1 - \mu_A(x) \quad (9)$$

3.5 Le contrôleur à logique floue

3.5.1 Variables linguistiques

Une variable linguistique est caractérisée par un quintuple $(V, T(V), X, G, M)$, dans lequel :

- V est le nom de la variable définie sur l'univers du discours X .
- $T(V) = A_1, A_2, \dots, A_n$ est un ensemble des termes linguistiques qui sont des nombres flous, définissant des restrictions sur les valeurs que prend V dans X .

- G est un ensemble de règles syntaxiques qui permettent de former d’autres termes linguistiques à partir de T(V). On les appelle modificateurs linguistiques. Par exemple, pour définir la fonction d’appartenance du terme linguistique «pas A» on utilise l’expression : $\mu_{\text{pas}(A)} = 1 - \mu_A$ (10)
- M est l’ensemble des règles sémantiques qui permettent de définir les termes linguistiques.

3.5.2 Fuzzification

La fuzzification est réalisée dans l’interface d’entrée du contrôleur flou. Durant cette phase, les informations issues du système sont tout d’abord normalisées. Ensuite, les données normalisées sont transformées en qualifications linguistiques, en utilisant des règles sémantiques définies par un expert.

Durant la phase de normalisation, chaque mesure issue du système est modifiée pour fournir une valeur appartenant à un univers du discours relativement simple. On peut choisir comme univers du discours un intervalle centré sur zéro : $[-c, +c]$. Si la mesure initiale x est comprise dans un autre intervalle $[a, b]$, la normalisation est souvent réalisée par transformation linéaire, selon : $y = 2 * c(x - (a+b)/2) / (b-a)$ (11)

L’univers du discours est ensuite représenté par une variable linguistique, qui comporte un nombre assez restreint de termes (en général trois, cinq ou sept) de façon à limiter le nombre de règles.

Enfin, les valeurs normalisées déduites de chacune des entrées sont transformées en qualifications linguistiques, en utilisant les variables linguistiques correspondantes.

3.5.3 Règles floues

Les règles floues permettent de déduire des connaissances concernant l’état du système en fonction des qualifications linguistiques fournies par l’étape de fuzzification. Ces connaissances sont également des qualifications linguistiques.

Habituellement, les règles floues sont déduites des expériences acquises par les opérateurs ou les experts. Ces connaissances sont traduites en règles simples pouvant être utilisées dans un processus d’inférence floue. Par exemple, si un expert exprime la règle «si la température de l’eau est chaude, il faut ajouter de l’eau froide», le système utilisera une règle du genre «si p alors q».

3.5.4 Inférence flou

L’inférence floue est une relation floue définie entre deux sous-ensembles. La définition de la relation peut théoriquement faire intervenir n’importe quel opérateur de combinaison. Pourtant, on utilise souvent les inférences floues définies par Mamdani et Sugeno.

3.5.4.1 Inférence floue de Mamdani

Supposons que la base de connaissances est constituée de n règles d’inférence contenant chacune m prémisses et une conclusion. Le processus d’inférence peut être décrit par le schéma suivant :

Règle 1 : Si (x_1 est A_{11}) et ... et (x_m est A_{1m}) ; alors (Y est B_1)

Règle 2 : Si (x_1 est A_{21}) et ... et (x_m est A_{2m}) ; alors (Y est B_2)

⋮
↓

Règle n : Si (x_1 est A_{n1}) et ... et (x_m est A_{nm}) ; alors (Y est B_n)

Dans lequel $x_1, \dots, x_m$ sont des éléments des univers du discours $X_1, \dots, X_m$ et

$A_{ji}, (j=1, \dots, m)$, sont des quantités floues sur l’univers du discours X_i , et

$B_j, (j=1, \dots, m)$, sont également des quantités floues sur l’univers du discours Y.

Afin de définir une seule prémisse pour une règle i, les propositions « x_j est A_{ij} »,

$(j=1, \dots, m)$, sont combinées par l’opérateur minimum. La fonction d’appartenance de cette prémisse unique est donc donnée par :

$$\mu_{R_{Ai}}(x_1 \dots x_m) = \mu_{A_{i1}}(x_1) \wedge \dots \wedge \mu_{A_{im}}(x_m) \quad (12)$$

3.5.4.2 Inférence floue de Sugeno

Sugeno a proposé une méthode d’inférence floue qui garantit la continuité de la sortie.

Cette méthode d’inférence s’avère très efficace dans des applications faisant intervenir à la fois des techniques linéaires, d’optimisation et adaptatives. Dans l’inférence de Sugeno, les règles floues sont exprimées de la façon suivante :

Règle i : Si (x_1 est A_{i1}) et ... et (x_m est A_{im}) ; alors $y = f_i(x_1, \dots, x_m)$

Dans laquelle $x_1, \dots, x_m$ et y sont des éléments des univers du discours $X_1, \dots, X_m$ et

$A_{i1}, \dots, A_{im}$ sont des termes linguistiques sur ces mêmes univers du discours, y est une fonction de $x_1, \dots, x_m$.

Par rapport à l’inférence de Sugeno, celle de Mamdani est plus intuitive, plus générale et elle s’adapte particulièrement bien à l’utilisation de connaissances issues d’une expertise humaine.

3.5.5 Défuzzification

La défuzzification est le traitement qui permet de définir une correspondance entre le résultat de l'inférence et la grandeur continue fournie en sortie.

Il y'a plusieurs méthodes de défuzzification, la plus utilisé est la règle de centre de gravité :

$$y_{cg} = \left(\sum_{i=1}^m \mu_{R_i} * y_i \right) / \sum_{i=1}^m \mu_{R_i} \quad (13)$$

3.5.6 Schéma d'une commande floue

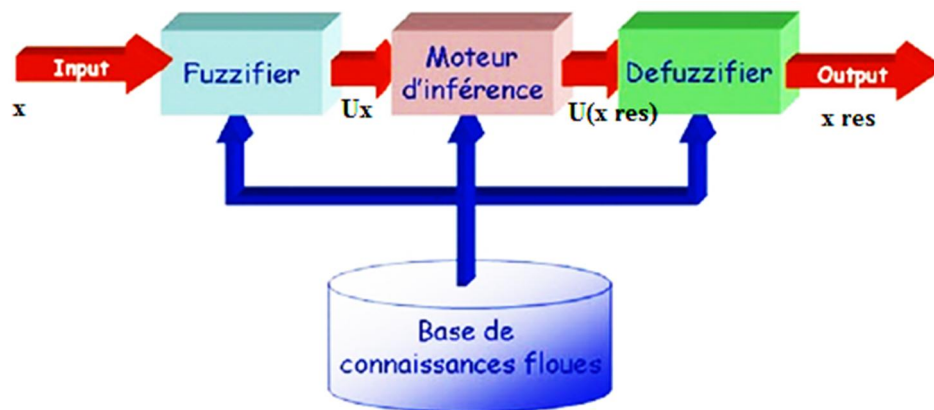


Figure 3. 11 : Schéma d'une commande floue.

La mise en œuvre d'une commande floue fait apparaître trois grands modules. Le premier module traite les entrées du système (valeurs réglant). On définit tout d'abord un univers de discours, un partitionnement de cet univers en classes pour chaque entrée, et des fonctions d'appartenance pour chacune de ces entrées (par exemple pression grande, petite, faible et changement d'écart mesure consigne de débit de matériau sortant d'une trémie très élevé, élevé, moyen, négatif, très négatif). La première étape, appelée fuzzification, consiste à attribuer à la valeur réelle de chaque entrée, au temps t , sa fonction d'appartenance à chacune des classes préalablement définies, donc à transformer l'entrée réelle en un sous ensemble floue. Le deuxième module consiste en l'application de règles de type «si l'écart de température est grand, diminué le débit du fuel». Ces règles vont, comme dans l'exemple introductif, permettre de passer d'un degré d'appartenance d'une grandeur réglant au degré d'appartenance d'une commande. Ce module est constitué d'une base de règles et d'un moteur d'inférence qui permet le calcul.

Le troisième et le dernier module décrit l'étape de défuzzification qui est la transformation inverse de la première. Il permet de passer d'un degré d'appartenance d'une commande à la détermination de la valeur à donner à cette commande.

Conclusion

Nous avons présenté dans ce chapitre l'élément principal de la foule virtuel qui est l'humain virtuel. Nous avons présenté ses propriétés pour donner un réalisme à l'animation de foule basée sur l'autonomie de comportement de chaque humain virtuel, ainsi des études sur les deux approches informatiques macroscopique et microscopique simulant ce comportement.

De manière générale, les modèles microscopiques offrent le plus de souplesse dans leur contrôle, et sont ainsi ceux qui permettent d'envisager le plus d'évolutions. Les modèles microscopiques récents sont généralement organisés sous la forme de trois étapes permettant de reproduire le phénomène de circulation pédestre. Premièrement, représenter l'environnement par une certaine forme d'abstraction pour en faciliter l'exploitation.

Dans la seconde partie nous avons présenté l'explication des principes de la logique floue pour l'utilisation ses principes afin donner plus de réalisme à l'animation d'humains virtuels.

CHAPITRE 4

CONCEPTION DU LA CIRCULATION DES PIETONS

Introduction

La simulation des circulations des personnes est une problématique nécessitant d'appréhender le comportement humain. Ceci est d'autant plus vrai dans notre domaine d'application, qui est la caractérisation de comportement des personnes n'est pas seulement un déplacement dans un environnement mais inclut aussi des interactions entre eux et avec divers équipements.

Nous allons représenter l'humains virtuels et leur environnement par de formes géométriques simples, et l'animation est une translation entre les points (ou les cellules), et utiliser l'animation comportementale pour animer les humains virtuels, nous assurons l'évitement de collisions entre les individus et les obstacles de l'environnement.

4.1 L'objectif du système

Nous allons décrire l'architecture logicielle pour effectuer la simulation d'animation de foule de piétons autonomes. Cette architecture a pour but de simule circulation des piétons dans un environnement virtuel par l'interaction continu avec les changements de l'environnement, et d'éviter la collision entre les piétons et avec les obstacles de l'environnement.

Nous avons concentré notre travail sur les méthodes d'évitement de collision entre les piétons avec l'utilisation des mécanismes de la logique floue, et l'algorithme de recherche le plus court chemin A* qui contenu l'évitement d'obstacles.

4.2 L'architecture logicielle dédiée à la simulation

4.2.1 Conception globale de notre système

Notre système est vue comme une boîte noire qui possède des entrées et qui fournit des sorties. Il prendre les informations d'extérieures et faire des opérations à l'intérieure pour donne des décisions à partir ces informations extérieures.

Le schéma suivant présent l'architecture globale de notre système :

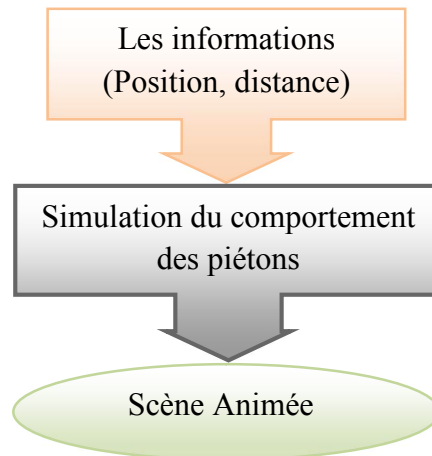


Figure 4. 1 : Conception globale.

4.2.2 Conception détaillée de notre système

Nous allons raffiner le système en un ensemble de comportements (phases) qui sont illustrées dans la figure suivante :

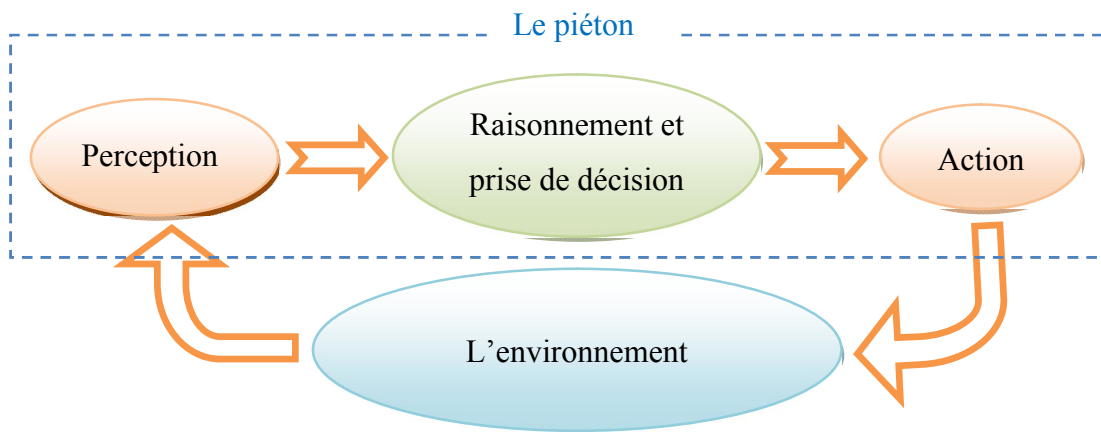


Figure 4. 2 : Conception détaillée.

a. Perception : qui permet aux piétons de percevoir l'environnement. Ce modèle qui fournit des informations sur les positions des obstacles, et les autres piétons et leurs directions, et leurs stations (destinations) sont les éléments qui composent l'environnement virtuel et ont une influence sur les décisions de piéton.

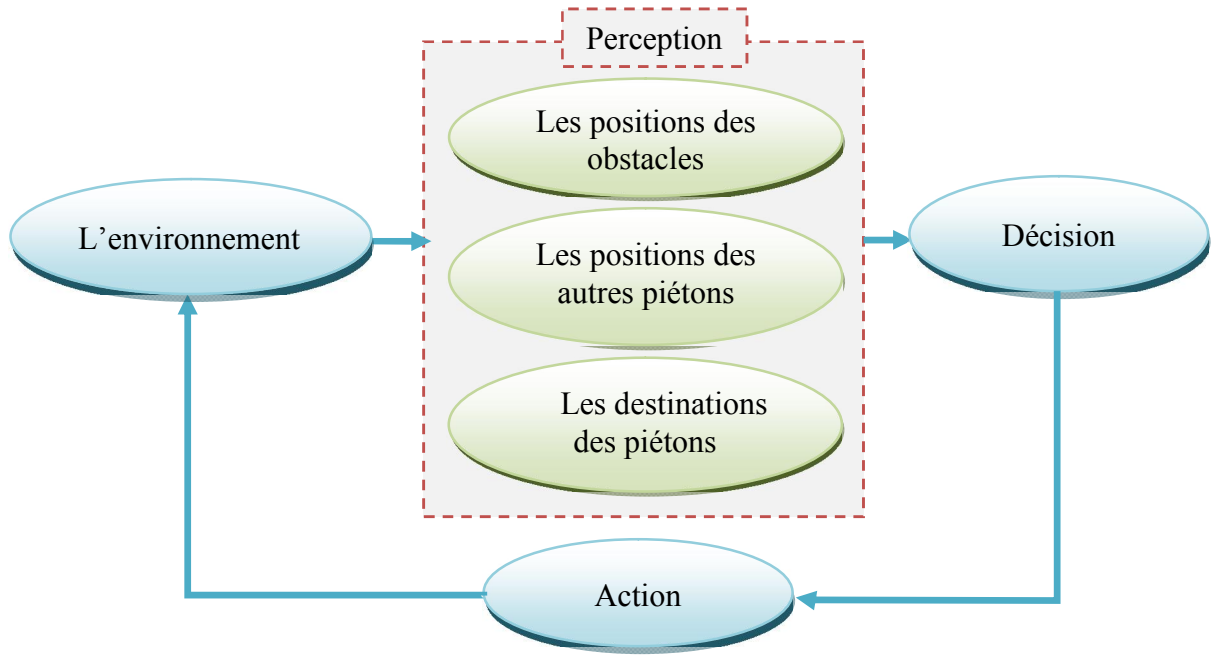


Figure 4. 3 : Structure de phase perception.

b. Raisonnement et prise de décision : le piéton dans l'environnement virtuel commence à chercher sa destination par explorer les routes à chaque pas, les facteurs de base de choix des routes destinataires sont : les positions des obstacles et les autres piétons. Ces comportements sont nécessaires pour la navigation de l'humain virtuel et qui sont :

- Sélection chemin : il permet de trouver un chemin, moins optimal pour une animation réaliste.
- Éviter obstacle : il permet d'éviter les obstacles qui se trouvent dans le chemin de piéton virtuel.
- Éviter collision : il permet d'éviter les collisions avec les autres piétons virtuels.

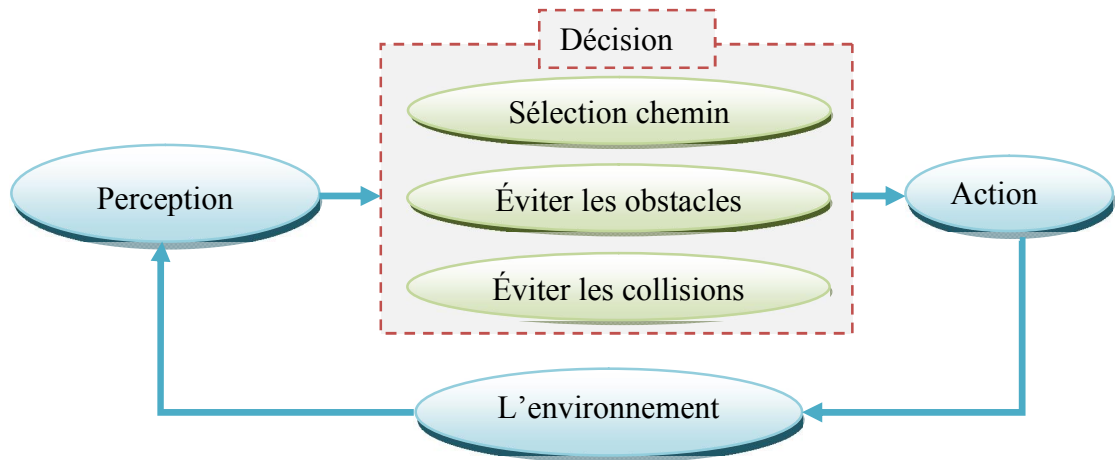


Figure 4. 4 : Structure de phase raisonnement et prise de décision.

c. Action : C'est l'exécution des résultats de phase prise de décision, l'action est présentée par le déplacement des piétons ou les groupes dans la scène. Le fait de déplacer un piéton vers son but est très simple, car on peut exprimer le mouvement de piéton par une droite vers son but.

Alors on peut dire que la phase d'action est une collection de comportements prédéfinis, tel que déplacement, arrêt, évitement de collision (changement la direction).

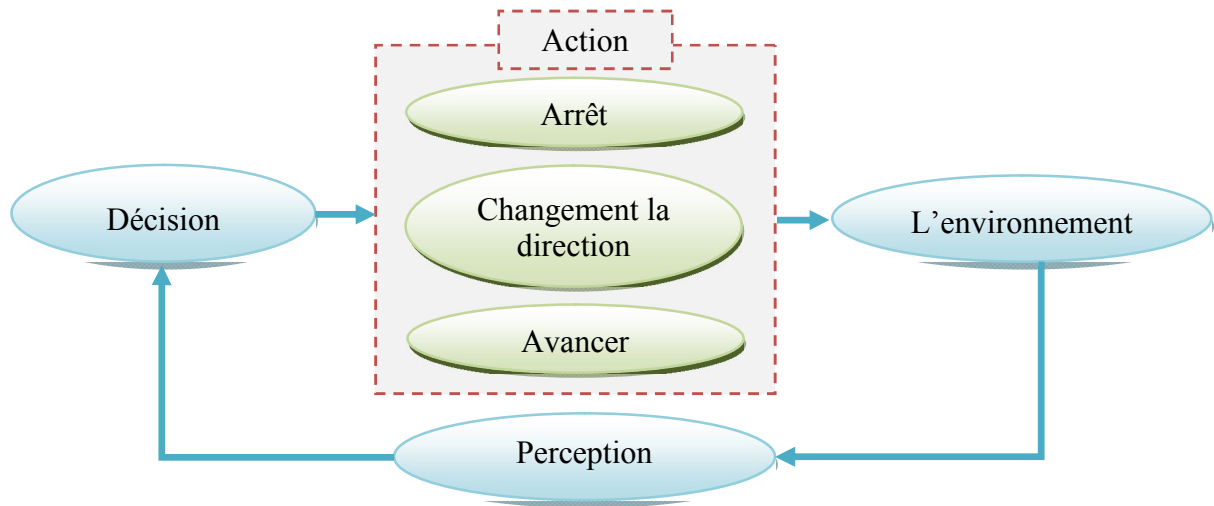


Figure 4. 5 : Structure de phase action.

4.3 Structure de notre système

Premièrement nous présentons la manière dont l'environnement virtuel est créé ainsi que la façon dont la scène est discrétisée sous la forme de cellules.

La structure générale du système comporte plusieurs modules qui permettent de simuler notre système tel que les obstacles, les individus et ses buts. En second lieu, nous présentons notre algorithme A* qui est utilisé dans l'environnement statique. Cet algorithme est employé par chaque individu pour choisir le chemin initial. Ensuite avant déplacements des piétons et pour chaque pas, faire la détection des collisions pour chaque individu et remplir dans la liste des collisions, faire traitement ces collisions à l'aide la logique floue pour aliénatrice les décisions ensuite déplacer un pas.

Le processus est répété dans la prochaine fois, et ainsi de suite, jusqu'à tous les piétons atteints leurs buts (boucle d'animation).

Le schéma suivant montre le fonctionnement de notre système :

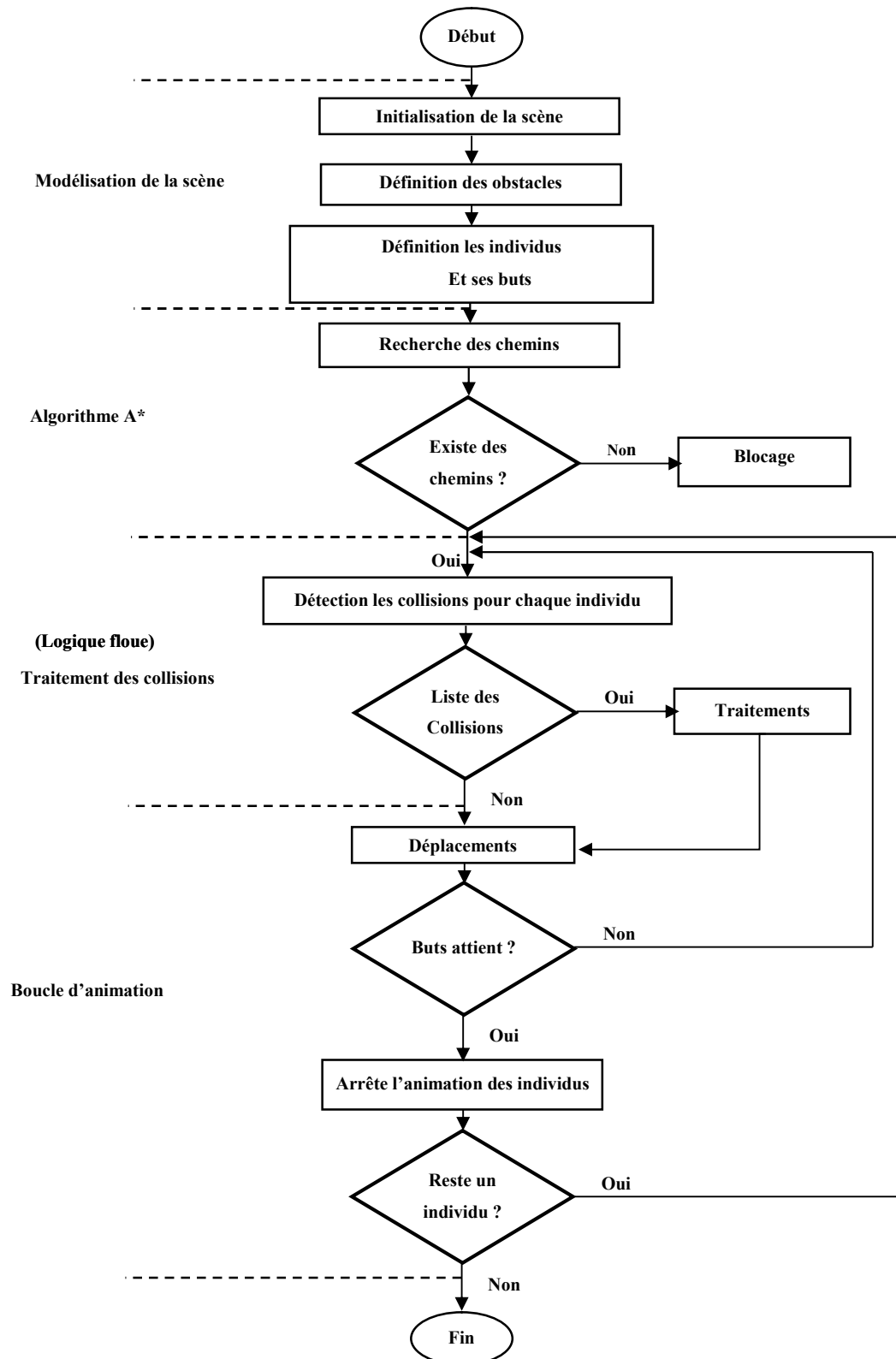


Figure 4. 6 : Organigramme du système.

Chaque piéton exécute l’algorithme de base selon le processus suivant :

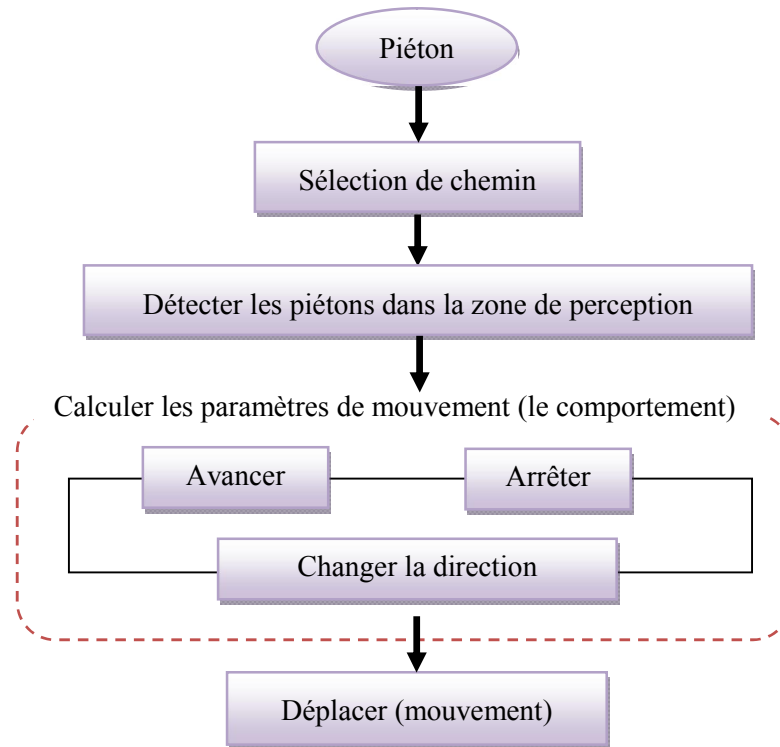


Figure 4. 7 : Algorithmes de base

4.3.1 Sélection de chemin

La planification de chemin consiste à rechercher un chemin optimal (généralement le plus court) entre un point de départ et un point de destination dans un environnement virtuel tout en évitant les différents obstacles qui peuvent se présenter. Traditionnellement, la planification de chemin a été résolue en utilisant un algorithme heuristique de recherche tel que le très célèbre algorithme A* algorithme directement couplé à l'animation des piétons de bas niveau.

4.3.2 Détection des piétons dans la zone de perception

Le piéton peut détecter seulement les piétons voisins dans la zone de perception. Dans notre modèle, la zone de perception de piéton est définie par une zone elliptique proche du champ visuel humain avec des zones proche à ce piéton.

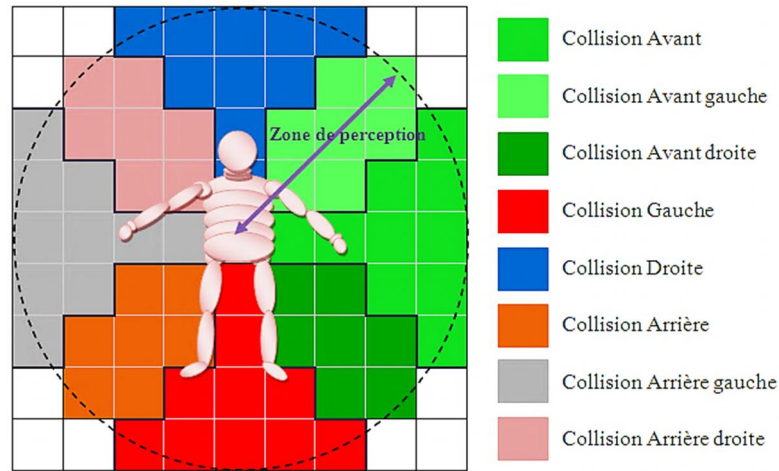


Figure 4. 8 : la zone de perception de piéton.

4.4 Modélisation de la scène

Nous nous sommes donné un environnement virtuel dans lequel les individus doivent se déplacer d'un point de départ vers une position donnée représentant le but. Les informations à donner sont :

- Le nombre ainsi que les positions d'obstacles afin de former l'environnement désiré.
- Le nombre de piétons ainsi que la position du but pour chaque piéton.

La bibliothèque OpenGL est un outil efficace pour faciliter la modélisation de l'environnement et des piétons à l'aide des objets géométriques préconfigurés comme (sphère, cube, cylindre, rectangle, tube, point, cône....).

Nous avons choisi deux types d'objets à modéliser :

4.4.1 L'environnement

L'environnement de notre simulation est représenté par des formes simples, avec une texture pour l'ajouter une splendeur à la scène.

Les obstacles : Nous avons choisi des formes simples pour représenter les obstacles ; des cylindres et des cubes, l'objectif essentiel de l'individu est d'éviter ces obstacles lors de son déplacement, la figure 4.9 illustre ce fait.

- L'arbre ou lampadaire : sont représentées par une combinaison de formes simples (cylindres et cubes et sphères) placés entre les murs.
- Les destinations : sont représentées par des cubes, elles sont considérées comme des buts à atteindre.

4.4.2 Les individus

Ce sont les éléments essentiels dans notre simulation, nous avons modélisé les individus par une combinaison de formes simples (cubes et sphères). Nous avons donné des couleurs différentes aléatoires aux individus pour différencier entre eux.

Dans notre système on a choisi le modèle RVB par exemple si on veut colorier un individu, on utilise la fonction `glColor3f(R, V, B)` avec : R (rouge), V (vert) et B (bleu) est aléatoire que représente les niveaux de gris.

$R = \text{random}()$, $V = \text{random}()$, $B = \text{random}()$.

Par exemple en cas $(R, V, B) = (0, 0, 0)$ l'individu noir, et $(R, V, B) = (1, 1, 1)$ l'individu blanc, et $(R, V, B) = (1, 0, 0)$ l'individu rouge.

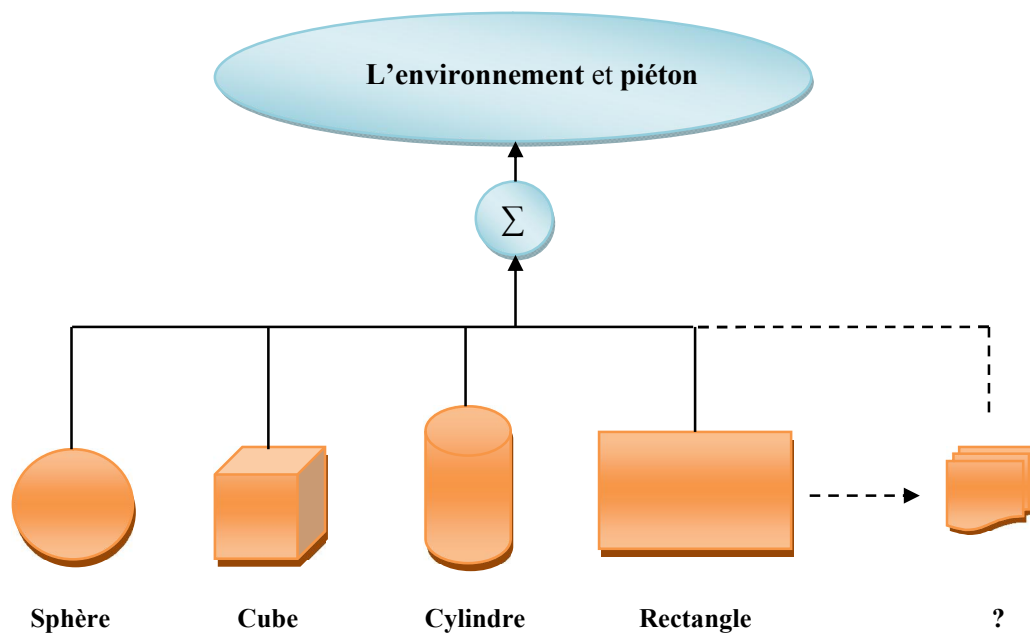


Figure 4. 9 : Modélisation de la scène.

4.5 Données observées

Les données observées sont récupérées par le processus complémentaire à notre l'algorithme, l'individu n'interprète pas des données purement géométriques, mais va directement sélectionner les informations avec leur sémantique dans la représentation de l'environnement. Cette sélection peut donc être assimilée à la mémoire à court terme d'un être humain, où sont stockées de manière éphémère l'ensemble des informations existé à l'instant présent.

Les informations récupérées concernent deux objets présents dans l'environnement :

- a. **Objets statiques obstacles** : ces objets sont directement présents sous la forme des obstacles fixes dans l'environnement, ces objets dont la localisation spatiale sont figés au cours d'une simulation comme les arbres et les lampadaires.
- b. **Objets dynamiques ou libres** : les objets dynamiques sont mobiles dans l'environnement, comme les individus virtuels (les piétons), qui avoir deux distances jouer le rôle du zone de sécurité pour la détection des collisions avec les autres individus.

Distance minimale : une distance minimale est associée à chaque piéton, exploité pour la détection une collision à proche.

Distance maximale : sur le même principe que la distance maximale, exploité pour la prévision d'une collision à distance.

4.6 Fonctions d'appartenance de la variable 'distance'

Les fonctions d'appartenance de variable « distance » sont de forme triangulaire et forme trapèze, où la somme des degrés d'appartenance de chaque variable aux différents sous-ensembles flous est toujours égale à l'unité.

La variable distance (d) possède les valeurs qualitatives : Prés(P), Moyen (M) et Loin (L) définis par les fonctions d'appartenance comme illustré dans la figure 4.10. Ces fonctions sont définies comme suit :

Fonction d'appartenance à sous-ensemble floue Prés(P) :

$$\mu_{\text{Prés}}(d) \begin{cases} 0 & \text{si } d \geq 6 \\ 1 & \text{si } 0 \leq d \leq 2 \\ (6 - d)/4 & \text{si } 2 < d < 6 \end{cases}$$

Fonction d'appartenance à sous-ensemble floue Moyen (M) :

$$\mu_{\text{Moyen}}(d) \begin{cases} 0 & \text{si } d \leq 2 \text{ ou } d \geq 10 \\ 1 & \text{si } d = 6 \\ (d - 2)/4 & \text{si } 2 < d < 6 \\ (10 - d)/4 & \text{si } 6 < d < 10 \end{cases}$$

Fonction d'appartenance à sous-ensemble floue Loin (L) :

$$\mu_{\text{Loin}}(d) \begin{cases} 0 & \text{si } d \leq 6 \\ 1 & \text{si } d \geq 10 \\ (d - 6)/4 & \text{si } 6 < d < 10 \end{cases}$$

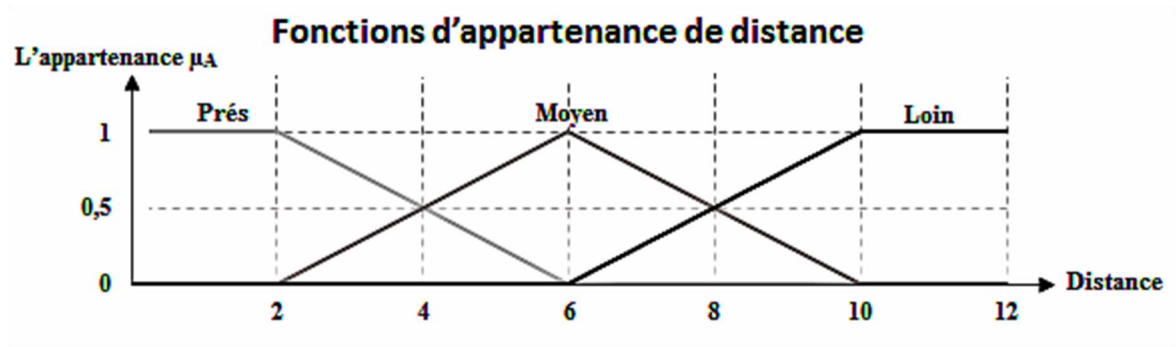


Figure 4. 10 : Fonctions d'appartenance de la variable 'distance'.

4.7 Contrôleur flou

Le contrôleur flou est utilisé pour les processus de type multi-entrées, multi-sorties, les grandeurs de sortie d'un processus à commander et éventuellement d'autres mesures déterminantes pour saisir l'évolution dynamique du processus ainsi que les consignes définissent les variables d'entrée du contrôleur flou. Les variables de sortie de ce contrôleur sont les commandes à appliquer au processus.

Le contrôleur flou est constitué de 4 blocs principaux (figure 4.11) : la base de connaissance, le système d'inférence, l'interface de fuzzification et l'interface de défuzzification. La base de connaissance est composée d'une base des données (les tables des individus, les tables des arbres, les tables des lampadaires) et d'une base de règles.

Un contrôleur flou passe généralement par les étapes suivantes :

- Choix de la stratégie de fuzzification.
- Etablissement de la base de règles.
- Choix de la méthode d'inférence.
- Choix de la stratégie de défuzzification.

La figure suivante donne le principe de conception de notre contrôleur :

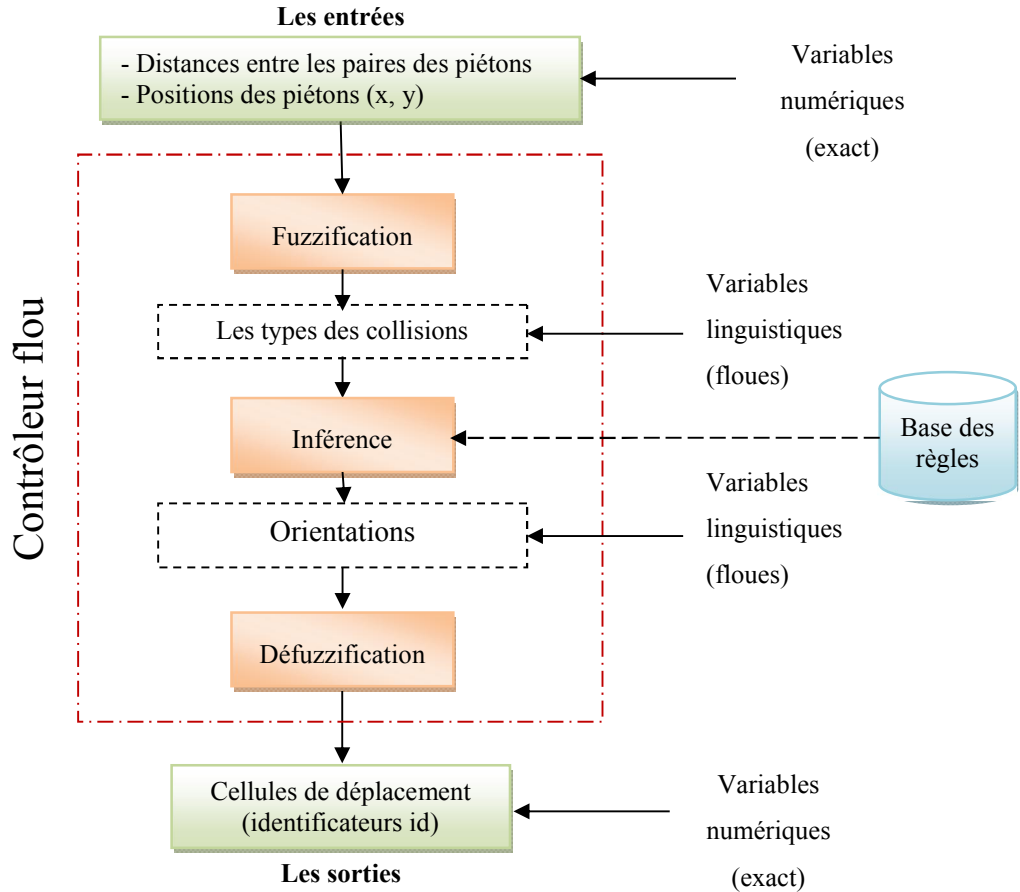


Figure 4. 11 : Configuration interne du notre contrôleur flou

Les entrées de notre contrôleur flou sont des valeurs exactes (Le sens de notre piéton et les coordonnées de position de piéton marchant vers la collision avec notre piéton x, y). Transformer ces valeurs par l'utilisation de la stratégie de fuzzification aux variables linguistiques indiqués par le type de collision, ensuite par l'utilisation de la base de règles d'inférence : *if... else...*, On transforme ces variables aux autres variables linguistiques sont des décisions afin que traitements des collisions en attendant sont transformés par l'utilisation de la stratégie de défuzzification aux valeurs exactes indiqués par les identificateurs des cellules de déplacement.

4.8 La prévision de collision

La prévision de collision est la perception des autres piétons en mouvement et qui sert par la suite à l'évitement de collisions. Le principe de cette méthode est de tester à chaque pas d'animation (un pas en avant) s'il existe un piéton qui occupe le même endroit. C'est-à-dire : est-ce que le point destination de ce piéton en mouvement est occupé par un autre piéton, dans le cas positif alors une collision est détectée.

Dans la réalité, il y a trois types de collisions possibles (Figure 4.12)

- Collision face à face : se produit si les piétons se déplacent l'un vers l'autre.
- Collision en arrière : se produit si deux piétons marchent dans la même direction et l'un derrière à l'autre.
- Collision de côte : se produit si les deux piétons marchent au la même cellule au même temps.

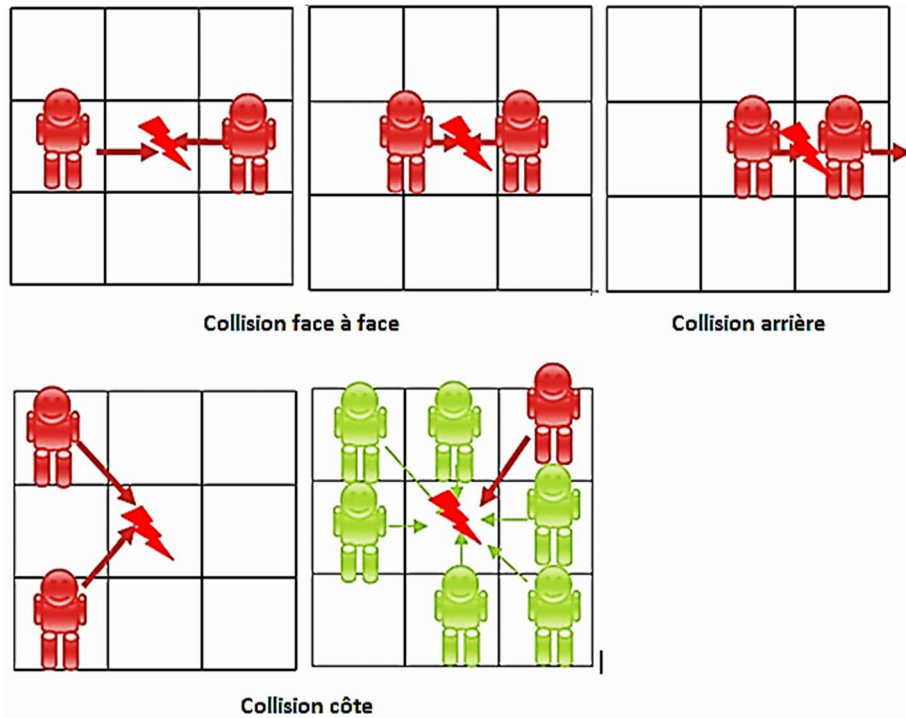


Figure 4. 12 : types de collisions.

4.9 Comportement d'évitement la collision

Pour chaque action d'évitement de collision nous besoin de différents comportements et de traitement différent. La liste de différents comportements qui peuvent être employés dans l'évitement de collisions est :

- **Avancement** : en le cas normal, c'est-à-dire n'existe pas des collisions.
- **Changement de direction** (*gauche ou droite*) : si prévision de existe une ou plusieurs collisions à proche.
- **Attente** : si prévision de existe une ou plusieurs collisions à distance.

Le processus d'évitement de collision est semblable pour toutes ces situations et pour conserver le chemin de l'algorithme A*, en appliquant l'algorithme ci-dessus pour chaque paire de piétons dans les collisions. Les tableaux suivants résument les différents comportements de traitement des collisions simples, entre deux piétons :

A. Collisions face à face

Traitement de ce type de collision est simple qui consiste à deux comportements, le changement de direction de premier piéton vers une cellule adjacente pour éviter la collision et laisser le deuxième piéton accomplir son avancement, comme suite :

Direction de piéton 1	Direction de piéton 2	Type de collision	Traitement	
			piéton 1	piéton 2
Avant	Arrière	avant ou arrière	tourner à avant gauche ou avant droite	Avancement
		avant gauche ou arrière gauche	tourner à avant droite	Avancement
		avant droite ou arrière droite	tourner à avant gauche	Avancement
		gauche ou coïncidence	tourner à droite	Avancement
		droite	tourner à gauche	Avancement
Arrière	Avant	avant ou arrière	tourner à arrière gauche ou arrière droite	Avancement
		avant gauche ou arrière gauche	tourner à arrière gauche	Avancement
		avant droite ou arrière droite	tourner à arrière droite	Avancement
		gauche	tourner à gauche	Avancement
		droite ou coïncidence	tourner à droite	Avancement
Gauche	Droite	avant ou arrière	tourner à gauche avant ou gauche arrière	Avancement
		avant gauche ou arrière gauche	tourner à gauche avant	Avancement
		avant droite ou arrière droite	tourner à gauche arrière	Avancement
		gauche	tourner à avant	Avancement
		droite ou coïncidence	tourner à arrière	Avancement
Droite	Gauche	avant ou arrière	tourner à droite avant ou droite arrière	Avancement
		avant gauche ou arrière gauche	tourner à droite arrière	Avancement
		avant droite ou arrière droite	tourner à droite avant	Avancement
		gauche ou coïncidence	tourner à arrière	Avancement
		droite	tourner à avant	Avancement
Arrière gauche	Avant droite	avant gauche ou gauche ou arrière gauche	tourner à diagonale avant	Avancement
		<i>else</i>	tourner à diagonale arrière	Avancement
Avant droite	Arrière gauche	avant droite ou arrière droite	tourner à diagonale gauche	Avancement
		<i>else</i>	tourner à diagonale droite	Avancement

Table 4.1 : Traitement des Collisions face à face.

B. Collisions à côte

Ce traitement à fin que évitement les collisions à distance, et est très simple par laisser le premier piéton accomplir son avancement et arrêter le deuxième piéton jusqu'à disparition la collision à distance :

Direction de piéton 1	Direction de piéton 2	Type de collision	Traitement	
			piéton 1	piéton 2
Avant	Gauche	droite ou avant droite	Avancement	Arrêt
	Droite	gauche ou avant gauche	Avancement	Arrêt
	Avant gauche ou Arrière gauche	droite ou avant droite	Avancement	Arrêt
	Avant droite ou Arrière droite	gauche ou avant gauche	Avancement	Arrêt
Arrière	Gauche	gauche ou avant gauche	Avancement	Arrêt
	Droite	droite ou avant droite	Avancement	Arrêt
	Avant gauche ou Arrière gauche	gauche ou avant gauche	Avancement	Arrêt
	Avant droite ou Arrière droite	droite ou avant droite	Avancement	Arrêt
Gauche	Avant	gauche ou avant gauche	Avancement	Arrêt
	Arrière	droite ou avant droite	Avancement	Arrêt
	Avant gauche ou Avant droite	gauche ou avant gauche	Avancement	Arrêt
	Arrière gauche ou Arrière droite	droite ou avant droite	Avancement	Arrêt
Droite	Avant	droite ou avant droite	Avancement	Arrêt
	Arrière	gauche ou avant gauche	Avancement	Arrêt
	Avant gauche ou Avant droite	droite ou avant droite	Avancement	Arrêt
	Arrière gauche ou Arrière droite	gauche ou avant gauche	Avancement	Arrêt
Avant gauche	Avant ou Avant droite ou Droite	gauche ou avant gauche	Avancement	Arrêt
	Arrière ou Arrière gauche ou gauche	droite ou avant droite	Avancement	Arrêt
Arrière droite	Avant ou Avant droite ou Droite	droite ou avant droite	Avancement	Arrêt
	Arrière ou Arrière gauche ou gauche	gauche ou avant gauche	Avancement	Arrêt
Avant droite	Avant ou Avant gauche ou gauche	droite ou avant droite	Avancement	Arrêt
	Arrière ou Arrière droite ou droite	gauche ou avant gauche	Avancement	Arrêt
Arrière gauche	Avant ou Avant gauche ou gauche	gauche ou avant gauche	Avancement	Arrêt
	Arrière ou Arrière droite ou droite	droite ou avant droite	Avancement	Arrêt

Table 4.2 : Traitement des Collisions à côte.

C. Collisions en arrière

Ces traitements fonctionner en cas de stationnement, par ce-que les piétons circulons avec même vitesse, qui consiste à un comportement, par laisser le premier piéton stationnement, et le deuxième piéton est que changé son direction :

Direction de piéton 1	Direction de piéton 2	Type de collision	Traitement	
			piéton 1	piéton 2
Stationnement	Avant	-	-	tourner à avant gauche ou avant droite
	Arrière	-	-	tourner à arrière gauche ou arrière droite
	Gauche	-	-	tourner à gauche avant ou gauche arrière
	Droite	-	-	tourner à droite avant ou droite arrière
	Arrière gauche	gauche ou avant gauche ou arrière gauche	-	tourner à diagonale avant
		<i>else</i>	-	tourner à diagonale arrière
	Avant droite	gauche ou avant gauche ou arrière gauche	-	tourner à diagonale arrière
		<i>else</i>	-	tourner diagonale avant
	Arrière droite	gauche ou avant gauche ou arrière gauche	-	tourner diagonale gauche
		<i>else</i>	-	tourner diagonale droite
	Avant gauche	gauche ou avant gauche ou arrière gauche	-	tourner diagonale droite
		<i>else</i>	-	tourner diagonale gauche

Table 4.3 : Traitement des Collisions en arrière.

Conclusion

Ce chapitre se focalise sur la conception détaillée de notre système par la présentation notre modèle qu'est composé de trois comportements principaux : la perception de l'environnement, le raisonnement et prise de décision, et l'action ensuite nous présentant le schéma de fonctionnement de notre système, et la conception des techniques de mouvement des piétons, et à la fin nous expliquant les concepts principaux de notre contrôleur flou, et la comportement d'évitement de collisions entre les piétons dans un environnement dynamique.

Rappelons que ce modèle s'intéresse que l'algorithme de base qui exécuté sur chaque piéton avec les données observées sont récupérées par le processus complémentaire à notre algorithme.

CHAPITRE 5

IMPLEMENTATION ET RESULTATS

Introduction

L'objectif de ce travail est de la simulation des circulations des personnes, pour se faire, nous avons proposé dans la partie précédente un ensemble de modèles qui permettent d'atteindre un certain niveau de complexité du comportement humain.

Dans ce dernier chapitre nous présentons un instantané de l'implémentation de notre modèle. Ainsi et dans un premier temps, nous présentons l'environnement de la simulation que nous avons conçu. Dans un second temps, une vue générale sur les résultats sera également présentée.

5.1 L'environnement de développement

C'est un logiciel tridimensionnel, qui fonctionne sur une machine à base d'un processeur Intel(R) Core(TM) i3-3217U, vitesse 1.8 Ghz, RAM 4,00 Go, utilisant le système d'exploitation Windows 7 professionnel de type 32 bits.

L'application est implémente en C++ Builder de CodeGear RAD Studio 2009, utilisant la bibliothèque d'OpenGL pour la modélisation de scène.

5.1.1 L'outil d'implémentation

C++ Builder est un logiciel de développement rapide d'applications (abr. RAD) conçu par Borland qui reprend les mêmes concepts, la même interface et la même bibliothèque que Delphi en utilisant le langage C++. Il permet de créer rapidement des applications Win32 ainsi qu'une interface graphique avec son éditeur de ressources.

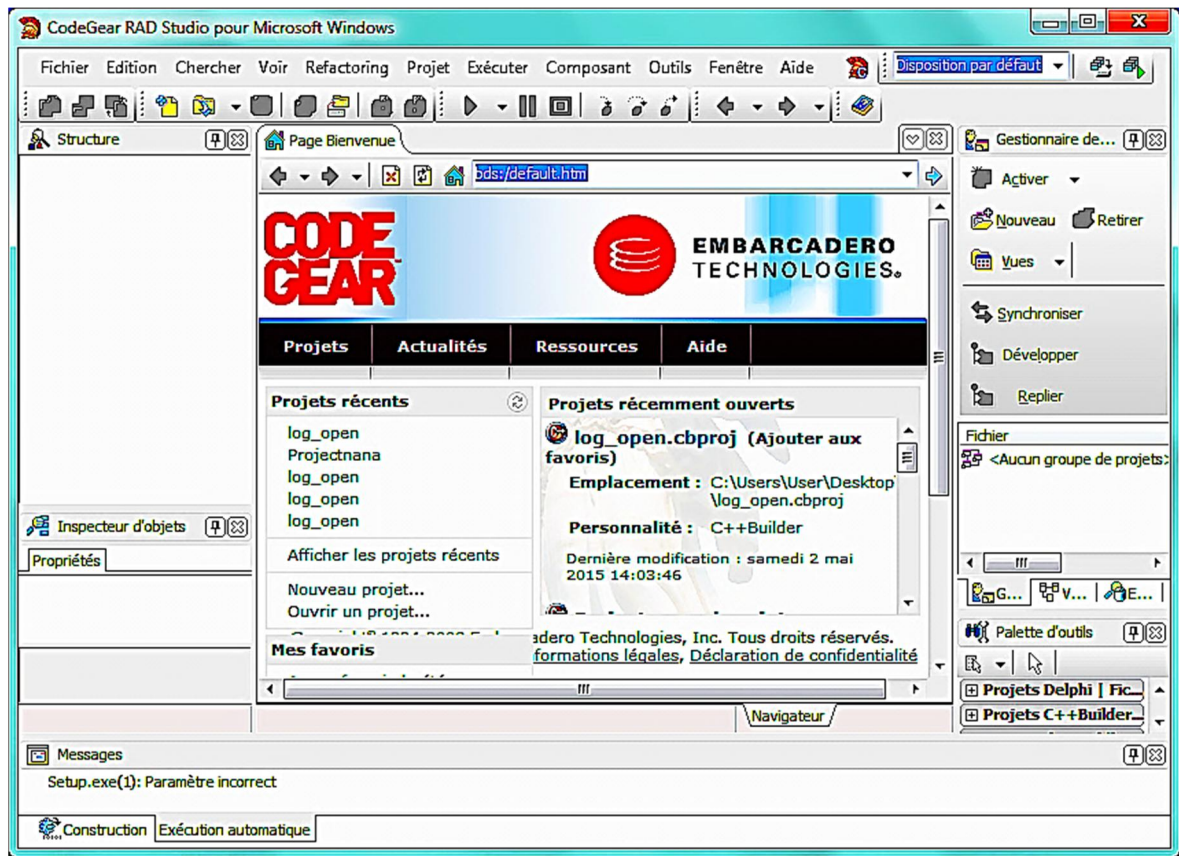


Figure 5. 1 : Fenêtre d'exécution de CodeGear C++ Builder 2009.

5.1.2 Présentation Open GL

OpenGL est une bibliothèque graphique, c'est à dire une interface logicielle permettant d'effectuer des opérations d'affichage sur un écran graphique. Cette interface, développée par Silicon Graphics, est constituée d'environ 150 fonctions graphiques, de l'affichage de points et segments jusqu'au plaquage de textures sur des objets tridimensionnels.

5.2 Méthode de simulation

La figure 5.2 illustre les différents composants de notre méthode de simulation. Notre modèle de simulation est une scène composé de deux éléments principaux « humains virtuels et l'environnement » déterminés par le réalisateur de l'animation.

Cette structure représente les différentes unités principales de notre simulation. Elle détermine aussi les relations entre différents objets, qui permettent les échanges des données.

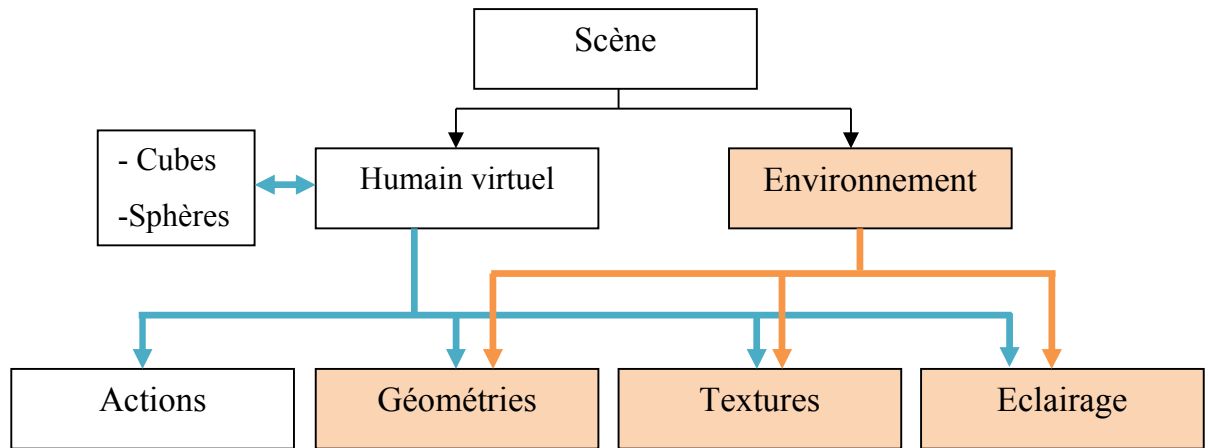


Figure 5. 2 : Structure de simulation.

5.3 L'interface principale de notre application

Notre application contient quatre fonctions aides à utilisation de notre simulateur qui illustrée dans la figure suivante :

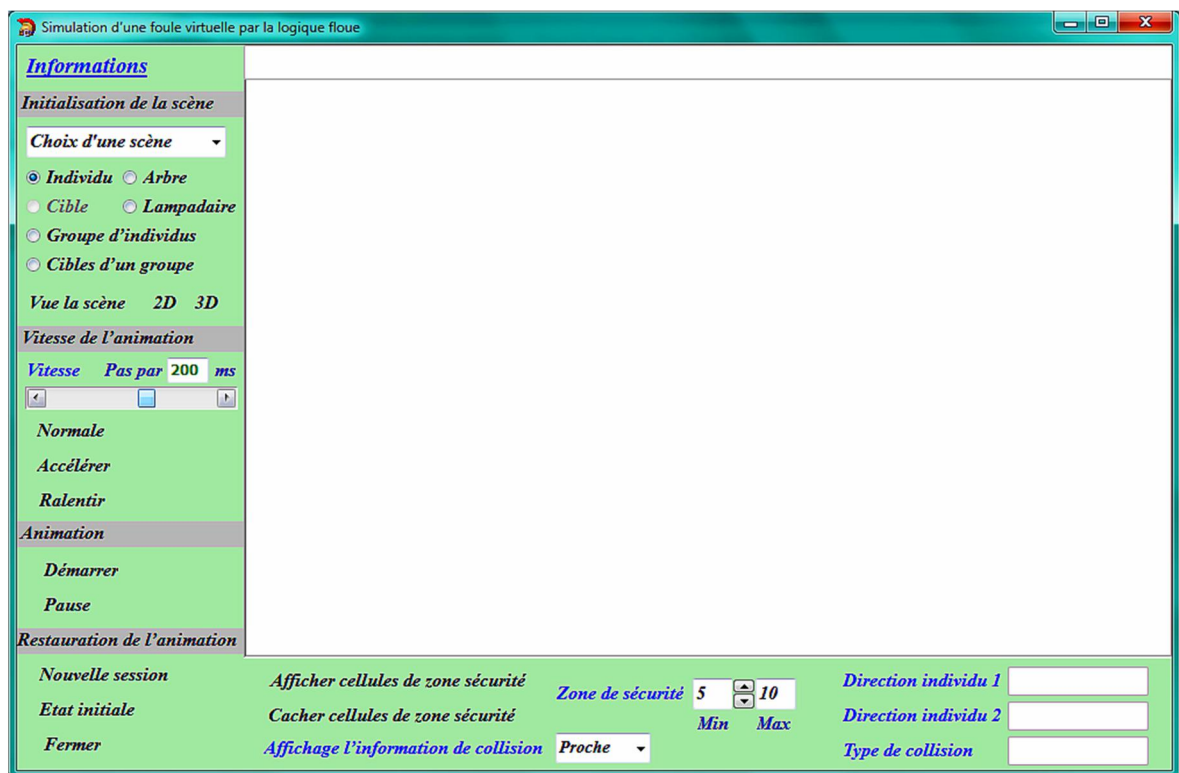


Figure 5. 3 : Fenêtre principale de notre application.

Initialisation de la scène

- **Choix d'une scène** : pour choix ou modifier la scène de l'animation.
- **Individu** : pour création un piéton.
- **Cible** : pour limitation un but d'un piéton.
- **Groupe d'individus** : pour création groupe de piétons.
- **Cibles d'un groupe** : pour création des cibles d'un groupe des piétons.
- **Arbre** : pour création un arbre.
- **Lampadaire** : pour création un lampadaire.
- **2D** : pour manifestation la scène de l'animation en deux dimensions.
- **3D** : pour manifestation la scène de l'animation en trois dimensions.

vitesse de l'animation

- **Vitesse** : pour contrôle la vitesse de l'animation.
- **Normale** : pour rendre la vitesse de l'animation moyenne.
- **Accélérer** : pour rendre la vitesse de l'animation rapide.
- **Ralentir** : pour rendre la vitesse de l'animation lente.

Animation

- **Démarrer** : pour démarrer l'animation.
- **Pause** : pour faire l'arrêt de l'animation.

Restauration de l'animation

- **Nouvelle session** : pour faire une nouvelle session d'animation.
- **Etat initiale** : pour retourner à l'état initial de dernière animation.

Autres boutons

- **Affichage l'information de collision** : pour limitation le type de collision de l'affichage.
- **Direction individu1, Direction individu1, type de collision** : pour l'affichage des informations d'une collision.
- **Zone de sécurité** : pour modifier la taille de la zone de sécurité.
- **Fermer** : pour fermer notre application.

Nous avons présenté quelques scènes avec un vu en deux dimensions sont illustré dans les figures suivantes (figure 5.4, figure 5.5) :

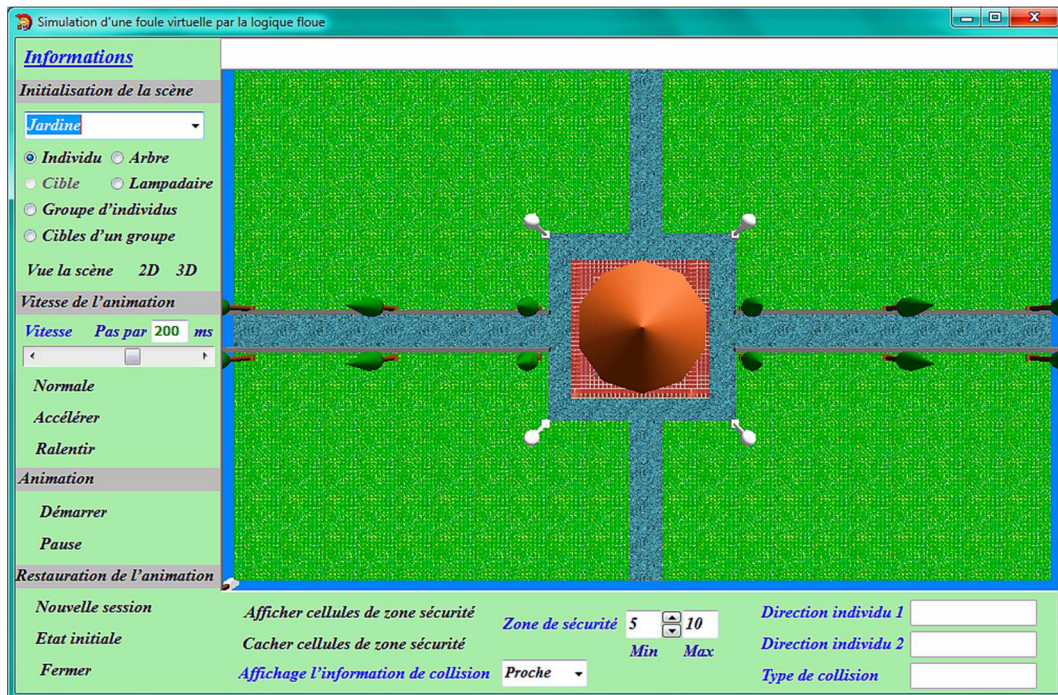


Figure 5. 4 : Vue en 2D (jardine).

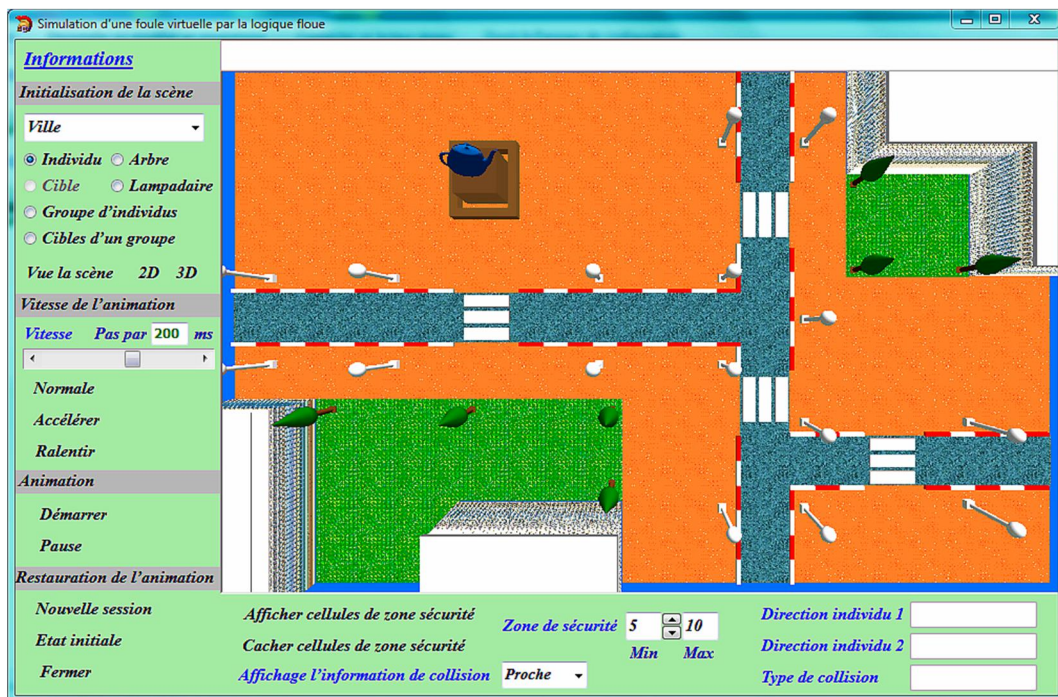


Figure 5. 5 : Vue en 2D (Ville).

Nous avons présenté quelques scènes avec un vu en trois dimensions sont illustré dans les figures suivantes (figure 5.6, figure 5.7) :

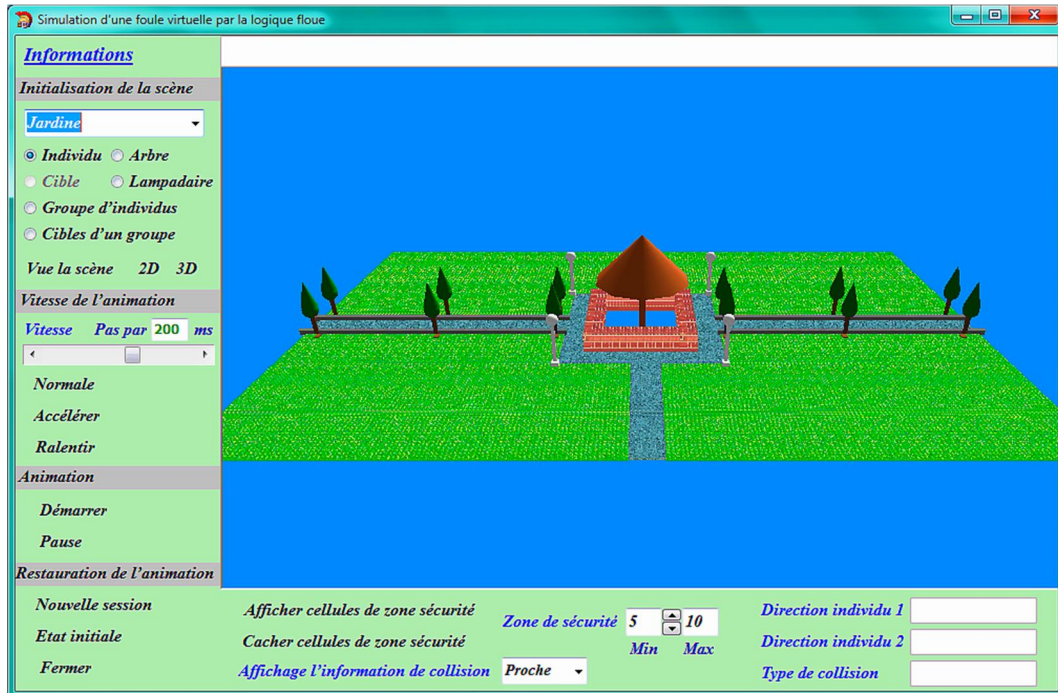


Figure 5. 6 : Vue en 3D (jardine).

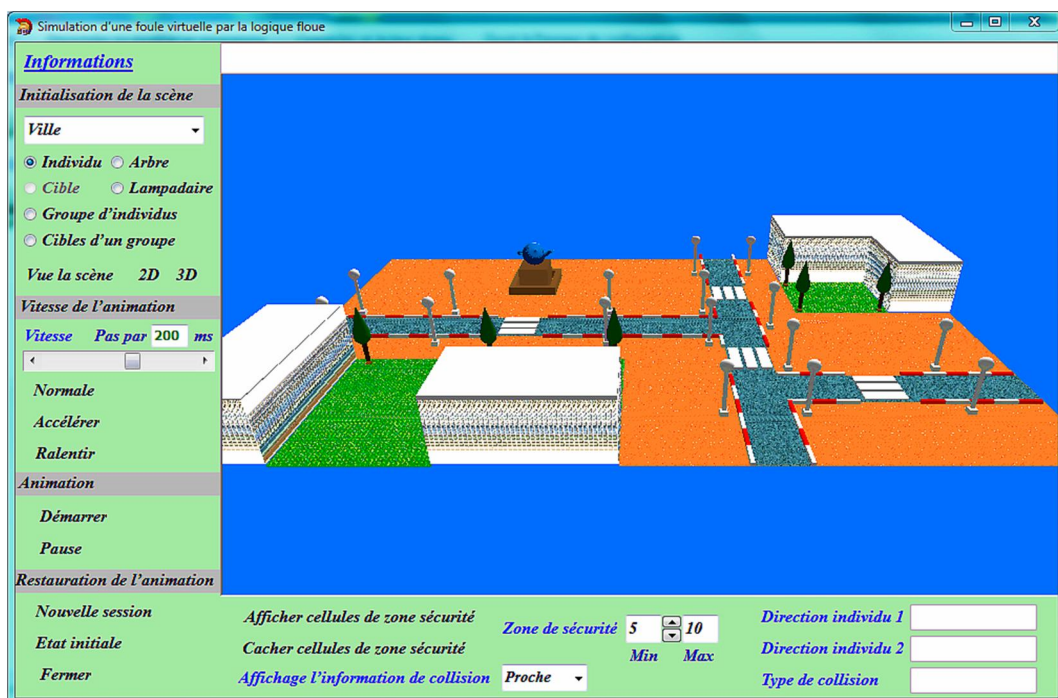


Figure 5. 7 : Vue en 3D (ville).

5.4 Structure de données et quelque algorithmes

5.4.1 Structure de données

La structure de données qui nous avons utilisée est de type table, et avant la déclaration des tables nous avons définie autres type de données nécessaire pour le système par exemple :

Le type Unit représente les coordonnées de l'animateur et leur direction utilise pour sauvegarder les chemins des individus :

```
typedef struct {int positionx ; int positiony ; float sens_ind ;  
float cout_f ;}Unit ;
```

Le type paire_collision utilise pour définie les propriétés de chaque collision entre deux piétons :

```
typedef struct {int individu1 ; int individu2 ; String type_  
collision ; String direction1 ; String direction2 ;}  
paire_collision ;
```

Quelques exemples des tableaux :

Unit liste_o [25000] : pour stocker les coordonnées des cellules voisines de cellule étudiée (les cellules adjacents).

Unit liste_f [25000] : pour stocker les coordonnées des cellules de chemin.

Unit tab_ind [10] : pour stocker les informations des individus comme les cellules des démarrages et les cibles.

Unit tab_arb [10] : pour stocker les coordonnées des arbres.

Unit tab_lamp [10] : pour stocker les coordonnées des lampadaires.

paire_collision tab_collision [50] : pour stocker les informations des collisions entre les paires d'individus.

5.4.2 Algorithme général du système

Le principe de cette méthode est faire l'application de l'algorithme A* pour chercher un chemin initial, ensuite tester à chaque pas d'animation (un pas en avant) s'il existe un piéton qui occupe le même endroit. C'est-à-dire est-ce que le point destination de le piéton en mouvement est occupé par un autre piéton, dans le cas positif alors une collision est détectée alors ajoutent les informations de cette collision dans la table de collisions. En fin nous appliquent l'algorithme d'évitement de collision piéton par piéton à la table de collisions et animer les individus un pas.

Pour chaque piéton **faire**

 Appliquer l'algorithme A* pour chercher le chemin initial

Fin pour

Pour chaque pas d'animation comportemental **faire**

Pour chaque paire de piétons **faire**

Si (Prédiction de collision) **alors** ajout dans la table de collisions

Fin pour

 Appliquer l'algorithme d'évitement de collision piéton par piéton à la table de collisions

 Déplacement des piétons

Fin pour

5.4.3 Algorithme d'évitement de collision piéton par piéton

En cas de collision, l'algorithme d'évitement de collision piéton par piéton est appliqué pour traiter les collisions contenues dans la table des collisions. Les comportements d'évitement de collision utilisés sont : l'avancement, le changement de directions et l'arrêt des individus.

Pour chaque paire de piétons **faire**

Si collisions à côte **alors**

Si la collision par rapport piéton 1 est avant **ou** avant gauche **ou** avant droite **alors**

Piéton 1 arrêter jusqu'à disparition la collision à distance

Piéton 2 accomplir son avancement

Fin si

Sinon

Piéton 2 arrêter jusqu'à disparition la collision à distance

Piéton 1 accomplir son avancement

Fin sinon

Fin Si

Sinon Si collision face à face **alors**

Si la gauche de Piéton 1 vide **alors**

Piéton 1 changer sa direction vers la gauche et Piéton 2 accomplir son avancement

Fin si

Sinon Si la droite de Piéton 1 vide **alors**

Piéton 1 changer sa direction vers la droite et Piéton 2 accomplir son avancement

Fin si

Sinon Si collisions en arrière **alors**

Si la gauche de Piéton 2 vide **alors**

Piéton 2 changer sa direction vers la gauche

Fin si

Si la droite de Piéton 2 vide **alors**

Piéton 2 changer sa direction vers la droite

Fin si

Fin si

Fin pour.

5.5 Résultats de la simulation

La combinaison de l'algorithme A* et les comportements d'évitement de collision donné des résultats satisfaisants de simulation un groupe de piétons virtuels dans un environnement virtuel.

5.5.1 Résultats de l'algorithme A*

La figure suivante présente les résultats obtenu après l'application de l'algorithme A* pour un individu :

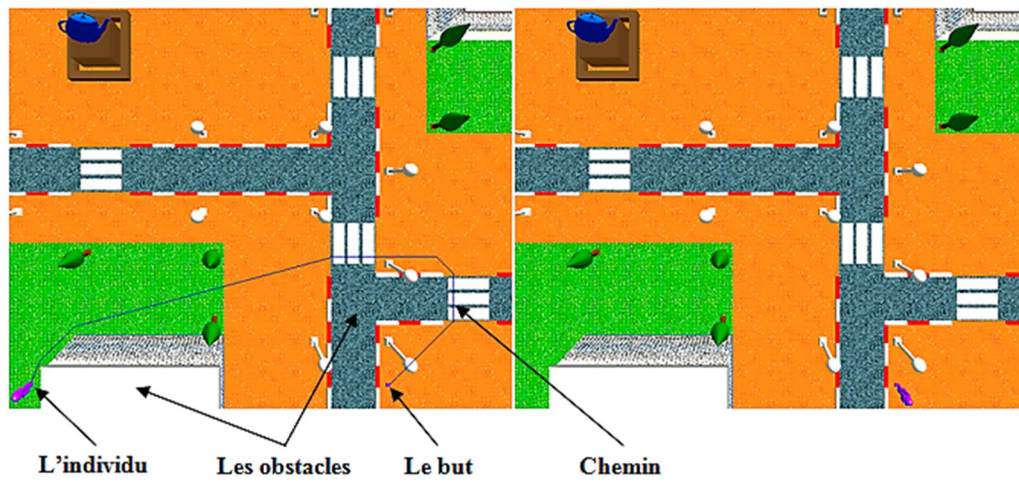


Figure 5. 8 : Résultat d'application de l'algorithme A* pour un individu.

La figure suivante présente les résultats obtenu après l'application de l'algorithme A* pour un groupe des individus :

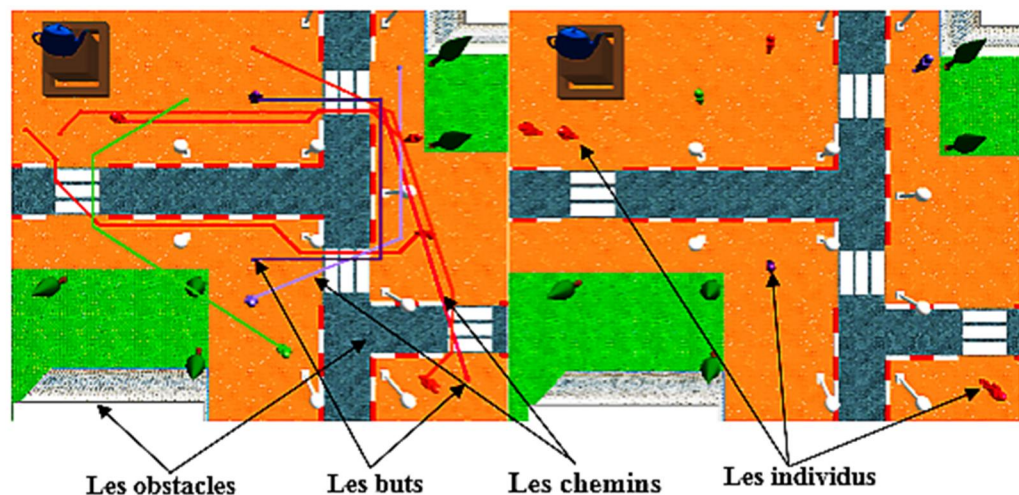


Figure 5. 9 : Résultat d'application de l'algorithme A* pour un groupe des individus.

5.5.2 Résultats de comportements d'évitement de collision

Les figures suivantes présentent les trois cas d'évitement de collision :

A. *L'évitement de collision face à face*

En cas les deux piétons déplacent sur le même couloir avec des directions contraires, un seul piéton qui change sa direction temporellement vers la gauche ou la droite, ensuite après l'évitement de la collision retourner à son chemin.

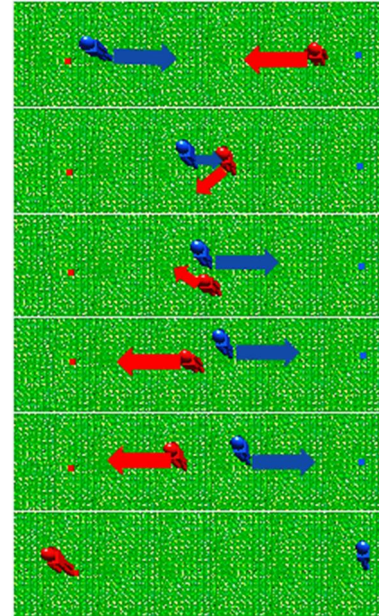


Figure 5. 10 : Résultat d'évitement collision face à face.

B. *L'évitement de collision à côté*

En cas les deux piétons déplacent avec des directions différentes mais pas contraires, un seul piéton arrête le déplacement jusqu'à passage le deuxième piéton.

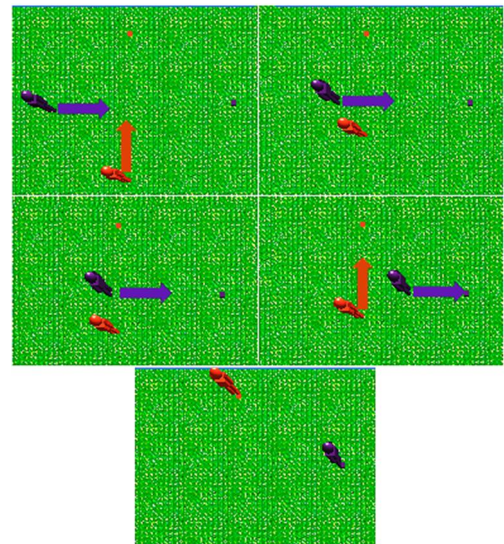


Figure 5. 11 : Résultat d'évitement collision à côté.

C. L'évitement de collision en arrière

En cas de stationnement un piéton dans le chemin d'un autre piéton, falloir a ce dernière changement sa direction temporellement vers la gauche ou la droite

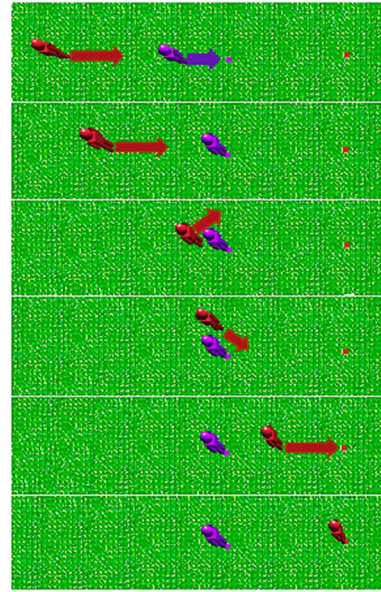


Figure 5. 12 : Résultat d'évitement collision en arrière.

5.5.3 Résultats de simulation « cas de panique »

Les piétons sortants par la sortie de secours un après un sans incidence dans les collisions entre eux et sans collisions avec le mur :

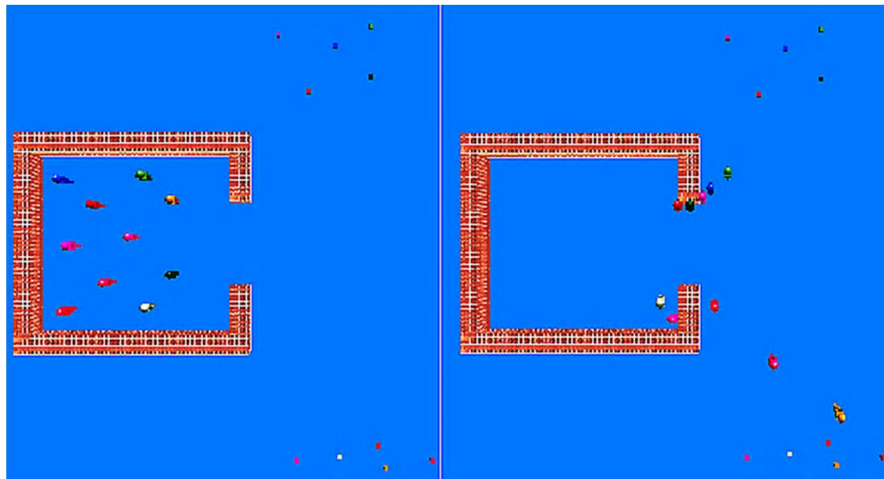


Figure 5. 13 : Résultats de cas panique.

Conclusion

Nous avons présenté dans ce chapitre l'environnement de développement de notre application, puis nous avons détaillé les interfaces principales de notre simulateur. Puis nous donnons les algorithmes fondamentaux pour les deux problèmes l'évitement des obstacles et l'évitement des collisions entre les individus virtuels avec les résultats de la simulation et comment implémenter l'algorithme A*, et les comportements d'évitement de collision, avec une petite explication de ces résultats et en fin nous avons présenté les résultats de simulation « cas de panique » et les différentes fonctions des boutons de notre application.

La simulation que nous avons réalisée s'est effectuée avec l'intégration locale comme les paramètres principaux, et des techniques guide et oriente les piétons et aide-le à l'animation. Le choix de ce paramètre se justifie par le fait qu'on a supposé que les piétons sont familiers avec l'environnement.

CONCLUSION GENERALE

Nous avons intéressé de simuler l'animation des foules de piétons, en exploitant le principe de l'animation comportementale pour animer les individus. Nous avons concentré notre effort sur les techniques d'évitement de collision ainsi que sur les algorithmes de recherche de chemin dans un environnement dynamique.

Nous avons proposé un modèle utilisant des formes géométriques simples pour modéliser les individus et l'animation de ces individus est faite par une translation.

Ce modèle s'articule autour de deux problématiques : recherche du chemin optimal, l'évitement de collision.

Nous avons proposé pour chaque individu un ensemble des règles comportementales pour résoudre le problème d'évitement de collision entre les individus, et nous choisissons l'algorithme A* pour trouver un chemin initial et la logique floue comme un mécanisme d'inférence. Ces deux techniques rehaussent le réalisme de la simulation d'animation d'une foule des individus.

Perspectives

Afin d'accroître les performances de ce modèle, il peut être important de se pencher sur différents autres aspects. Nous avons relevés trois axes permettant d'obtenir un modèle plus réaliste :

- L'utilisation de modèle d'environnement urbain.
- L'ajout des règles sociales et psychologiques aux comportements de navigation.
- L'utilisation de l'approche de simulation de comportement de groupes des piétons.

Bibliographie

[Acf01]	Arikan (O.), Chenney (S.) et Forsyth (D. A.). Efficient multi-agent path planning. Dans : Computer Animation and Simulation 01, éd. par Magnenat-Thalmann (N.) et Thalmann (D.), pages 151-162, 2001.
[Ama12]	Amar DELLABANI, Simulation réaliste de Croisement de flux de piétons par intégration de la capture de mouvement, thèse magister de l'université Mohamed Khider Biskra, Juin 2012.
[Ana00]	http : //www.ananova.com , 19 avril 2000.
[Arc79]	J. Archea. The evacuation of non-ambulatory patients from hospital and nursing home first: A framework for a model. Technical report, National Bureau of Standards, Center for Building Technology, National Engineering Laboratory, Washington DC, 197. 19.
[Bee90]	Beer (R.D.). Intelligence as Adaptive Behavior : Experiments in Computational Neuroethology. New York : Academic Press. (1990). Voir aussi : http://yuggoth.ces.cwru.edu/johng/robopaper/robopaper.htm .
[Boy95]	Boissonnat (J.-D.) et Yvinec (M.). Géométrie algorithmique. Collection Informatique. EDISCIENCE international, 1995.
[Boy98]	Boissonnat (J.-D.) et Yvinec (M.). Algorithmic Geometry. Cambridge. University Press, 1998.
[Bro86]	Brooks (R.A.). - A robust layered control system for a mobile robot- IEEE Journal of Robotics and Automation, pages 14-23, 1986.
[Che04]	S. Chenney. Flow tiles. In SCA '04 : Proceedings of the 2004 ACM SIGGRAPH / Eurographics symposium on Computer animation, Eurographics Association.
[Che06]	Cherif FOUJIL, Animation comportementale : Simulation de foules d'humains virtuels, thèse de doctorat d'état en Informatique, décembre 2006.
[Chl03]	Choi (M. G.), Lee (J.) et Shin (S. Y.). Planning biped locomotion using motion capture data and probabilistic roadmaps. ACM Transactions on Graphics, vol. 22, no. 2, pages 182-203, 2003.
[Coc07]	N. Courty and T. Corpetti. Crowd motion capture. Comput. Animat. Virtual Worlds, 2007.
[Dav08]	David PANZOLI, Proposition de l'architecture «Cortexionist» pour l'intelligence comportementale de créatures artificielles, thèse doctorat de l'université de Toulouse, 30 septembre 2008.
[Don04]	Stéphane Donikian, Modélisation, contrôle et animation d'agents virtuels autonomes évoluant dans des environnements informés et structurés, Université de Rennes 1. (26/08/2004).
[Fcy02]	Franco Tecchia, Céline Loscos, et Yiorgos Chrysanthou. Visualizing crowds in real-time. Computer Graphics Forum, 21(4) pages 753-765, 2002.
[Gib86]	J. Gibson. The Ecological Approach to Visual Perception. Lawrence Erlbaum Associates Inc, US, 1986.
[Hen71]	L. Henderson. The statistics of crowd fluids. Nature, 1971.

[Hkl99]	Hoff III (K. E.), Keyser (J.), Lin (M.), Manocha (D.), et Culver (T.). Fast computation of generalized Voronoi diagrams using graphics hardware. Computer Graphics, vol. 33, pages 277-286, 1999.
[Hug02]	R. L. Hughes. A continuum theory for the flow of pedestrians. Transportation Research Part B : Methodological, 2002.
[Jjk98]	J. J. Kuffner. Goal-directed navigation for animated characters using real-time path planning and control. Lecture Notes in Computer Science, 1537 pages 171-179, 1998. 1.5.2.
[Kuf99]	Kuffner (J. J.). Autonomous agents for real-time animation. PhD thèse, Staford University, 1999.
[Lad04]	Lamarche (F.) et Donikian (S.). Crowds of virtual humans: a new approach for real time navigation in complex and structured environments. Computer Graphics Forum, Eurographics'04, 2004.
[Lat91]	Latombe (J.-C). Robot Motion Planning. Boston, Boston : Kluwer Académie Publishers, 1991.
[Lcl07]	A. Lerner, Y. Chrysanthou, and D. Lischinski. 2007.
[Mae94]	Maes (P.). - Modeling adaptive autonomous agents - Artificial Life I, Vol. (1&2), Number 9, 1994.
[Mou11]	Moufida BENCHABANE, Une nouvelle approche de modélisation des structures de groupe d'une foule des piétons, Mémoire de Magister en Informatique de l'université Mohamed KHIDER - BISKRA 30 /05 /2011.
[Nes76]	Newell (A.) et Simon (H.A.). - Computer Science as Empirical Enquiry– Dans : Communications of the ACM, Vol. 19, pages 113-126, 1976.
[New90]	Newell (A.). - Unified theories of Cognition - Harvard University Press, Cambridge, MA, 1990.
[Ove02]	Overmars (M.H.). Récent developments in motion planning. Dans : International Conférence on Computational Science (S), pages 3-13, 2002.
[Par07]	S. Paris, Caractérisation des niveaux de services et modélisation des circulations de personnes dans les lieux d'échanges, thèse de doctorat, Université de Rennes 1, 2007.
[Pay98]	K.M. Passino and S. Yurkovich, Fuzzy Control, Reading, MA: Addison-Wesley, 1998.
[Rcr05]	R. M. Colombo and M. D. Rosini. Pedestrian flows and non classical shocks. Mathematical Methods in the Applied Sciences, September 2005.
[Rdn02]	Ruttkay (Z.), Dormann (C.) et Noot (H). - Evaluating ECAs- What and How? - Dans : Proceedings of the first International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02). Conference Workshop : Embodied Conversational Agents, Let's specify and evaluate them. 2002.
[Rey87]	C. W. Reynolds. Flocks, herds and schools : A distributed behavioral model. In SIGGRAPH '87 : Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, NY, USA, 1987.

[Rey99]	Reynolds (C. W.). Steering Behaviors For Autonomous Characters. GDC 99, pages 763-782, 1999.
[Rja05]	Rasmus Andersen, Jean Louis Berrou, et Alex Gerodimos. On some limitations of grid-based (ca) pedestrian simulation models. Dans First International Workshop on Crowd Simulation (V-Crowds'05). VRlab, EPFL, 2005. 1.5.2.
[Séb07]	Sébastien PARIS, Caractérisation des niveaux de services et modélisation des circulations de personnes dans les lieux d'échanges, thèse de doctorat de l'université de rennes 1 10 octobre 2007.
[Seb07]	Septseault (C.). Représentation d'environnements virtuels informés et de leur dynamique par un personnage autonome en vue d'une crédibilité comportementale. PhD thèse. Université de Bretagne Occidentale. 2007.
[Stb96]	S. Thrun et A. Bücken. Integrating grid-based and topological maps for mobile robot navigation. Dans Proc. of the AAAI Thirteenth National Conference on Artificial Intelligence, pages 944–951. AAAI Press / MIT Press, 1996. 1.5.2.
[Ste94]	Steels (L.). - The artificial life roots of artificial intelligence. - Dans: Artificial Life Journal, Vol. 1(1), pages 75-110, MIT Press, 1994.
[Tcp06]	A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. In SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, New York, NY, USA, 2006.
[Tup02]	A. Turner and A. Penn. Encoding natural movement as an agent- Based system : an investigation into human pedestrian behaviour in the built environment. Environment and Planning B : Planning and Design, 2002.
[Tur07]	A. Turner. To move through space : lines of vision and movement. In Proceedings, 6th International Space Syntax Symposium, 2007.
[Wst05]	Wei Shao et Demetri Terzopoulos. Autonomous pedestrians. Dans SCA 05 : Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pages 19-28, New York, NY, USA, 2005. ACM Press. 1.4.2, 1.5.2, 1.5.4, 2.2.1.

ملخص

تهدف الصور المتحركة السلوكية الى تقديم النماذج والأدوات التي نَسْمَحُ بخلق كيانات افتراضية مستقلة ذاتياً. تُدرك هذه الكيانات بيئتها، وتتصرفُ بناءً على هذه الأخيرة وتتخذُ لوحدها القرارات بالارتباط مع الحالة التي أدركتُ لهدف الإقتراب إلى محاكاة الكائن الحي المُقَلَّد.

نحن نُثَرِّنا الإهتمام بنمذجة الإنسان الافتراضي مع بيئته وبعد ذلك قمنا بمعالجة مشكلة الصور المتحركة السلوكية. نماذجنا تعرض بمكتبة OpenGL، وللصور المتحركة السلوكية استخدمنا خوارزمية A* (إيجاد الطريق المختصر) لتخطيط المسار الأولي والمنطق الضبابي كآلية الاستدلال.

الكلمات المفتاح: الصور المتحركة السلوكية، المنطق الضبابي، إيجاد الطريق المختصر، البيئة الافتراضية، تجنب الاصطدام، تجنب الحواجز.

Abstract

The behavioral animation seeks to provide models and tools for the creation of autonomous virtual entities. These entities perceive their environment, act on it and mostly take themselves decisions related to the perceived situation in order to exhibit consistent behavior close to the simulated living organism.

We were interested to model virtual humans with their environment and tackle the problem of behavioral animation. Our modeling performed by the OpenGL library and for behavioral animation we used the A* algorithm (finding the abbreviated path) for the initial path planning and fuzzy logic as an inference mechanism.

Key words: behavioral animation, fuzzy logic, finding the abbreviated path, virtual environment, collision avoidance, obstacles avoidance.

Résumé

L'animation comportementale cherche à offrir des modèles et des outils permettant la création d'entités virtuels autonomes. Ces entités perçoivent leur environnement, agissent sur ce dernier et surtout prennent d'elles-mêmes des décisions en rapport avec la situation perçue dans le but d'exhiber un comportement cohérent proche de l'organisme vivant simulé.

Nous nous sommes intéressés à modéliser des humains virtuels avec leur environnement puis attaquer le problème de l'animation comportementale. Notre modélisation réaliser par la bibliothèque OpenGL, et pour l'animation comportementale nous avons utilisé l'algorithme A* (trouver un chemin optimal) pour la planification de chemin initial et la logique flou comme un mécanisme d'inférence.

Mots clés : animation comportementale, la logique floue, trouver un chemin optimal, environnement virtuel, évitement de collisions, évitement des obstacles.