Génération d'explications dans un environnement muséographique

Adrien Paquet

Master $2^{ème}$ année Informatique
Recherche en Informatique
Rapport de stage

Encadrant : Eric Maisel

Equipe ARéVi
Centre Européen de Réalité Virtuelle (CERV)
Technopôle Brest-Iroise
29280 Plouzané

Génération d'explications dans un environnement muséographique

Adrien Paquet

Brest, le 2 juin 2011

Résumé

Ce rapport de stage s'intéresse au contrôle d'une visite dans un musée virtuel, à travers un échange d'informations (verbales ou non) entre le guide virtuel et le visiteur.

Après un état de l'art sur les différents types de visites et les comportements autonomes, nous proposerons une solution pour combiner les intentions des deux acteurs, à l'aide de comportements, qui seront modifiables durant la visite, par des automates. Ensuite, nous permettrons au visiteur de changer, en cours de route, sa visite, grâce à un collège d'experts. Enfin, nous donnerons notre conclusion.

Table des matières

Table des figures

Listings

Chapitre 1

Introduction

1.1 Contexte

1.1.1 Présentation

Ce stage de Master 2ème année, Recherche en Informatique, a été mené au CERV [1] et consiste à étudier le contrôle d'une visite dans un musée virtuel, à travers un échange d'informations (verbales ou non) entre le guide virtuel et le visiteur.

En association avec UNESCO, l'ICOM [2] a définit le musée comme :

> « une institution permanente, sans but lucratif, au service de la société et de son développement, ouverte au public et qui fait des recherches concernant les témoins matériels de l'homme et de son environnement, acquiert ceux-là, les conserve, les communique et notamment les expose à des fins d'études, d'éducation et de délectation. »

Les musées ont de nombreux problèmes (figure 1.1 page suivante), qui les obligent à mettre en place d'autres méthodes d'expositions. Ces problèmes sont dû notamment,

- au manque de place : au musée du Louvre, moins de 8% (35 000) des œuvres sont exposées au visiteur, sur les 445 000 acquises par le musée (source : wikipedia)
- au manque de personnel : à Océanopolis [3], seul 10% du public bénéficie d'un guide
- au coût des installations et des objets, à la fragilité des objets : au musée du Louvre, pour des raisons de conservation, les dessins sont montrés au public, 3 mois au plus (source : wikipédia). Mais également à la dangerosité de certains ces objets.

Ce stage propose de palier à ces problèmes en utilisant la réalité virtuelle, ainsi qu'un guide virtuel.

1. Centre Européen de Réalité Virtuelle (www.cerv.fr)
2. Conseil international des musées (http://icom.museum/)
3. Océanopolis est un centre de culture scientifique dédié aux océans, situé à Brest (www.oceanopolis.com)

FIGURE 1.1 – Les principaux problèmes rencontrés par les musées

1.1.2 Évolution des musées

Avant d'utiliser la réalité virtuelle, les musées proposaient d'autres "lieux" d'exposition (figure 1.2 page suivante).

La première solution mise en place fût la reproduction des œuvres, sur papier, notamment au XVIIe siècle. Un exemple de reproduction papier est la collection de Cassiano dal Pozzo qui regroupe 7000 dessins sur des différents thèmes.

Puis, à partir du XIXe siècle, les musées ont utilisé le plâtre pour reproduire les œuvres, comme avec la copie en plâtre du Parthénon, à la Skulpturhalle de Bâle, en Suisse.

Profitant des avantages apportés par Internet, les musées proposent d'effectuer des visites virtuelles de leurs musées sur leur site Internet.

Suivant l'évolution des visites réelles vers les visites virtuelles, la nature des informations ont étés modifiées. Devenues numériques, les explications consistent à lire un fichier audio ou vidéo, lors d'une visite virtuelle.

Aujourd'hui, les musées s'intéressent aux mondes virtuels, permettant de disposer d'un espace d'exposition illimité, à l'architecture et au contenu modifiable, et de toucher un plus grand nombre de personnes.

1.2 Problématique et propositions

La réalité virtuelle permet une plus grande liberté et une plus grande flexibilité (quand à l'architecture), qu'il serait difficile d'obtenir dans un musée réel. Malgré ce nouvel espace d'exposition, un problème se pose : comment guider le visiteur et lui donner les explications, sur les objets 3D qui constituent cet nouvel environnement.

Pour cela, nous proposons d'utiliser un guide virtuel. Le problème qui se pose est celui du contrôle de la visite : il s'agit d'être en mesure de répondre

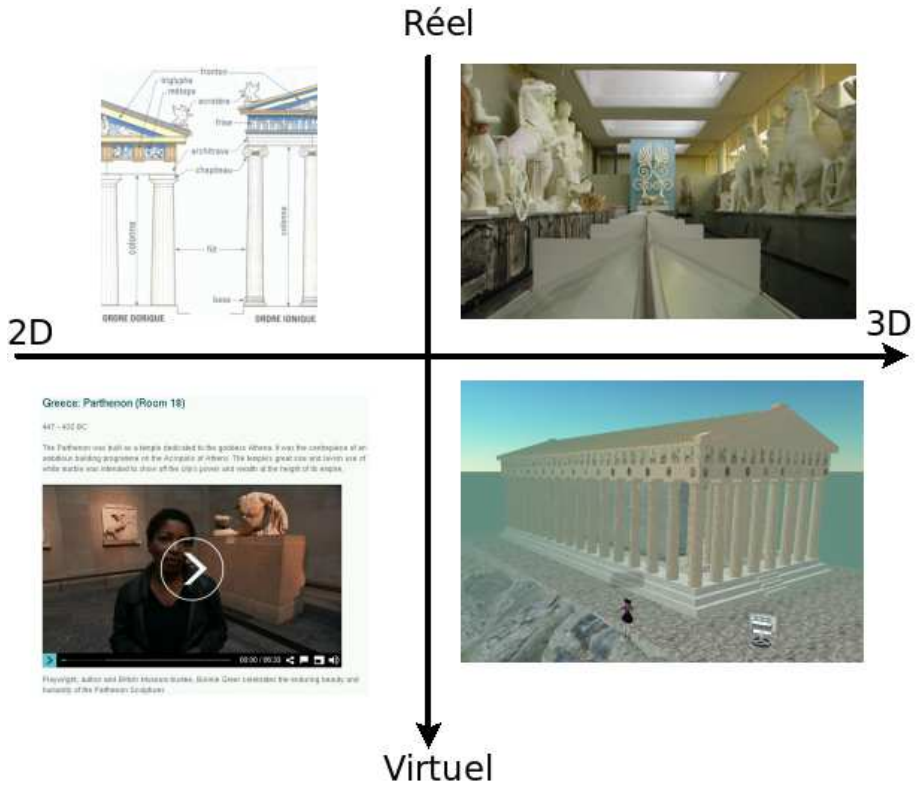


FIGURE 1.2 – Évolution des musées : vers la réalité virtuelle

aux questions suivantes :

- QUOI : quels objets présenter
- QUAND : quand présenter ces objets
- COMMENT : selon quelles modalités

Plusieurs solutions sont envisageables. Nous en proposons une qui permet de tenir compte des objectifs pédagogiques des commissaires de l'exposition mais également des souhaits des visiteurs. Il s'agit alors d'être en mesure :

1. d'identifier le souhait des utilisateurs
2. de le combiner avec les objectifs pédagogiques

Chapitre 2

Etat de l'art

2.1 Les types de visites

2.1.1 La visite statique

La visite statique est une visite où le visiteur n'a aucun contrôle sur ses déplacements et le déroulement de la visite. En effet, le parcours a été défini à l'avance, et le guide devra passer devant chaque œuvre qui constitue la visite, et donner les explications associées à celle-ci. Pour illustrer cette approche, nous allons décrire la méthode employée par J. Ibanez et al [5]. Leur objectif était de proposer des visites, de type narratif, au visiteur, à l'aide d'un guide et d'un environnement virtuel.

Ils supposent que le guide connait l'environnement (c'est à dire, les lieux et les objets qui sont présents), ainsi que les informations (ici, les histoires liées à un élément). Le guide navigue dans le monde virtuel et, déplace automatiquement la caméra (ou, le point de vue) du visiteur. De plus, le système donne aucune interaction (déplacement, dialogue...) au visiteur. Pour la navigation, ils utilisent la méthode proposée par Saretto [13], qui consiste à combiner une tâche et une tactique. Les tâches possibles sont :

- la recherche d'une cible connue (searching)
- le parcours d'objets visibles (browsing)

Les tactiques possibles sont :

- envoyer une requête (ou commande) sur un objet pour obtenir des informations (querying)
- la décision et le déplacement vers un endroit (navigation)

Dans leur système, seule la première tâche (searching) et la première tactique (querying) sont utilisées, car le guide connait toutes les informations nécessaires, ce qui implique les actions de recherche d'un objet et de récupération ces informations. Après, les déplacements sont assurés à l'aide d'un parcours de graphe (c'est à dire : trouver un chemin partant d'un point A vers un point B, sur un réseau), et le calcul de la distance minimale entre les points de l'environnement.

Un autre point important dans cet article est la construction narrative de la visite, histoire par histoire. À partir de ses connaissances (Figure 2.1 page suivante - partie a) et de plusieurs critères (comme la distance vers le lieu, les éléments déjà utilisées, son intérêt pour le guide, ou le comportement voulu par les concepteurs), le guide choisi une étape (Figure 2.1 page suivante - partie b).

Puis, le guide sélectionne d'autres éléments d'histoires qui ont un lien avec la première histoire et les transforme, afin d'apporter son point de vue (Figure 2.1 - partie c et d). À cela sont ajoutés des règles de connaissances communes (Figure 2.1 - partie e), qui permettent d'ajouter des informations basiques du guide, et améliore ainsi l'histoire. Au final, un scénario est créé avec tous les différents éléments narratifs, comprenant les histoires, mais aussi les effets et les actions que le guide apportera (Figure 2.1 - partie g).

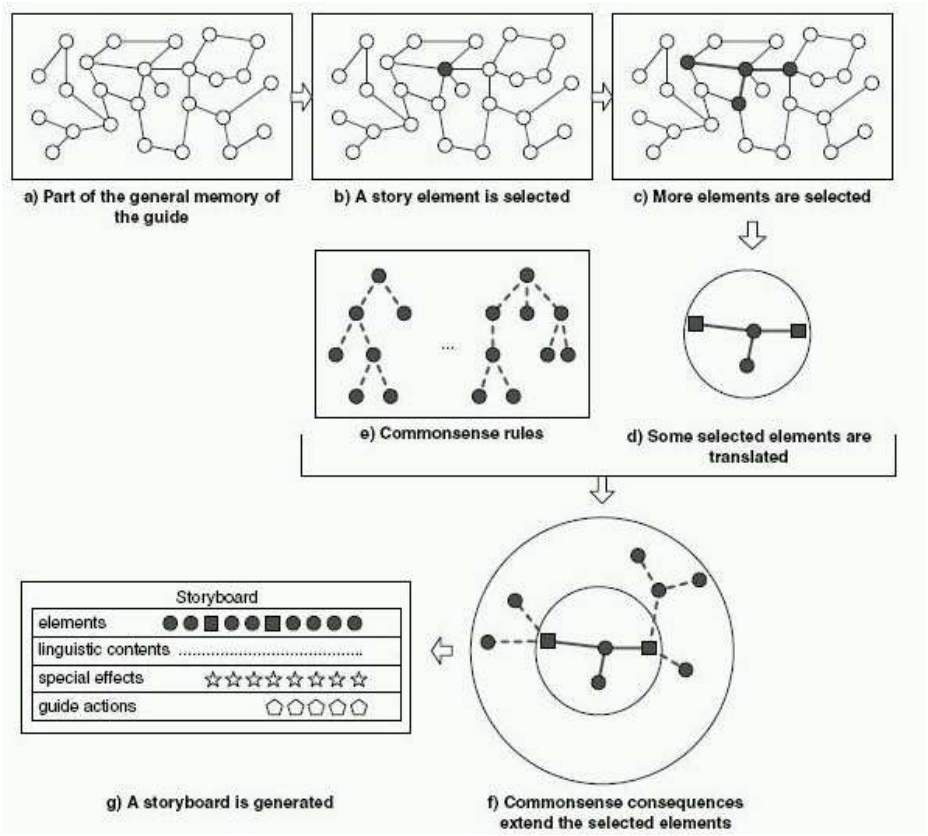


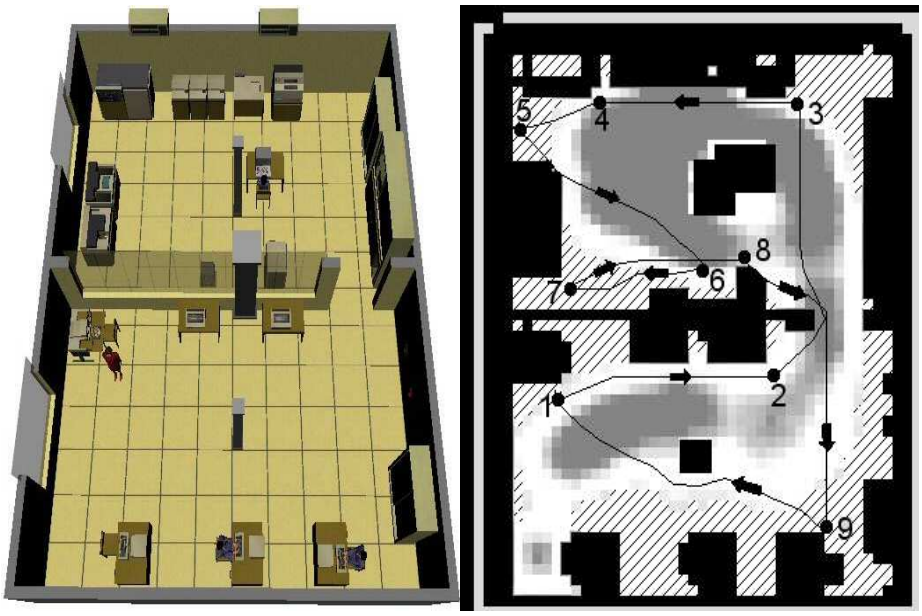
FIGURE 2.1 – Exemple de construction d'un scénario narratif

Dans [3], le guide suit un parcours prédéfini (figure 2.2 page suivante), permettant de présenter, dans un ordre logique, les différents objets du musée au visiteur.

L'avantage d'une telle approche est de s'occuper uniquement du parcours de la visite. Par contre, cette « simplification » ne prend pas en compte le visiteur. En effet, il ne peut pas influencer sur le parcours et est obligé de le suivre, à la manière d'un film.

2.1.2 La visite interactive

Dans la visite interactive, le visiteur effectue la visite et le guide intervient pour donner des explications. Celles-ci sont des réponses à des requêtes faites explicitement (utilisation d'interfaces graphiques, langage naturel...) par l'utili-



(a) Capture d'écran du musée virtuel (b) Chemin suivi par le guide

FIGURE 2.2 – Exemple d'une visite proposé par le guide

sateur. Cette approche est principalement utilisée dans les musées réels, comme compléments d'informations, pour les œuvres présentées, via des ordinateurs de poche.

Ainsi, le système KORE [2] utilise ces ordinateur de poche pour que le guide virtuel donne des informations sur une œuvre, vue par le visiteur (Figure 2.3 page suivante). Le visiteur est localisé, via des capteurs situés au niveau de chaque œuvre. Sa position est alors envoyé au guide virtuel qui proposera l'information associée. Toutes les informations sur chaque œuvre sont gardées par des bases de données et récupérées par l'agent.

Dans [9], le visiteur peut indiquer ses intérêts (même artiste, nom, ou style), et le système cherchera une œuvre répondant au critère choisi par le visiteur (Figure 2.4 page suivante). Une fois trouvée, il sera guidé jusqu'à l'œuvre sélectionnée.

Certains systèmes permettent au visiteur de soumettre des requêtes (ou commandes) explicites au guide. Ainsi, le système proposé en [10] permet à l'utilisateur de demander des informations, en utilisant la commande « TellMeAbout » suivi d'un mot clé, comme le nom d'une personne, ou d'une salle d'un laboratoire. D'autres systèmes vont plus loin : [12] permet à l'utilisateur de poser une question au guide, en langage naturel.

L'intérêt de cette approche est limité car elle risque de faire manquer des objets et explications importantes au visiteur, et provoquer sa frustration (la peur d'avoir manquer quelque chose d'intéressant). Aussi, la nécessité de faire des requêtes explicites, dans certains cas, réduit l'impression d'immersion, et pose des problèmes aux personnes qui ne sont pas familières avec l'informatique. Enfin les informations peuvent être données dans un mauvais ordre, ce qui peut

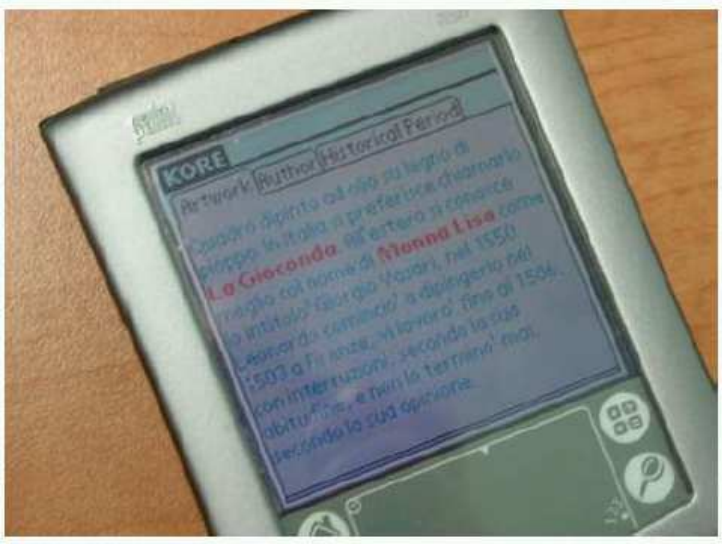


FIGURE 2.3 – le système « KORE » donne des informations sur une peinture, à l'aide d'un ordinateur de poche.

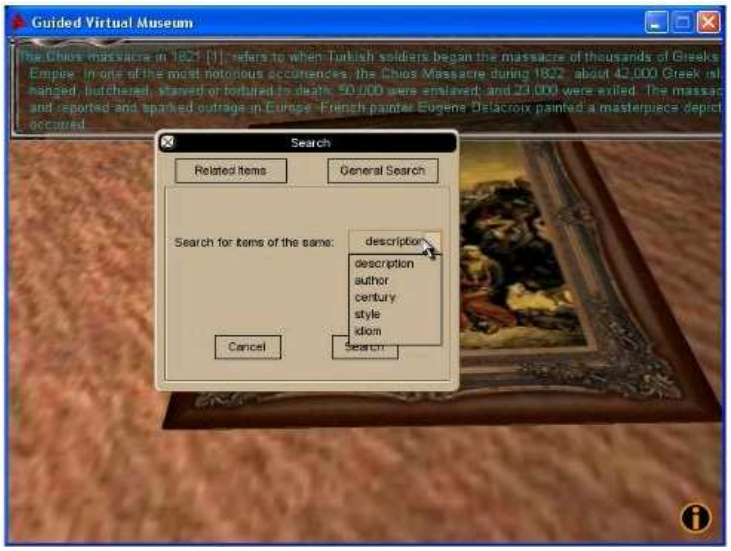


FIGURE 2.4 – L'utilisateur choisi une nouvelle œuvre, en fonction de ces intentions et de celle étudiée, pour continuer la visite

entrainer une mauvaise compréhension.

2.1.3 La visite réactive

La visite réactive est une amélioration de la visite interactive. Le visiteur navigue toujours dans le musée avec son guide, mais celui-ci réagit aux actions du visiteur.

Dans [6], les auteurs utilisent un agent virtuel sur un ordinateur portable, à l'intérieur du musée réel, mais aussi l'affichage de cet agent et d'un musée virtuel, devant le visiteur (Figure 2.5).

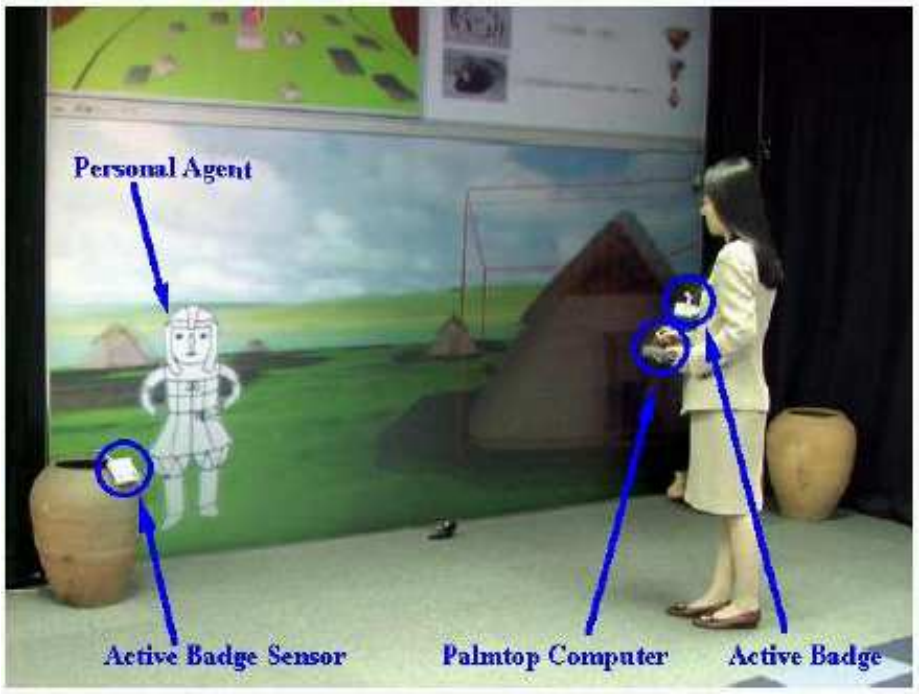


FIGURE 2.5 – le guide, qui se trouvait dans l'ordinateur de poche, est affiché, avec le musée virtuel, devant le visiteur.

Au départ, le visiteur se déplace dans le musée réel, et assiste à des démonstrations. Le guide virtuel conserve la position du visiteur et ses intérêts initiaux (renseignés au début de la visite) pour recommander le prochain endroit à visiter et lui fournir un plan de l'exposition. Il essaye aussi d'évaluer le niveau d'assistance requis pour le visiteur, via le temps passé sur les démonstrations et la prise d'informations, qui permettra au guide de proposer la meilleure stratégie pour aider le visiteur, dans le musée virtuel.

Ainsi, le guide permettra au visiteur, qui a pris beaucoup d'informations dans le musée réel, de naviguer librement dans le monde virtuel. Les visiteurs, qui auront pris le moins d'informations, effectueront une visite contrôlée par le guide. De plus, le guide possède deux types d'explications, courtes ou longue, qui dépendent du temps passé dans le musée réel. Lors d'un dialogue avec le visiteur, l'agent MAX [7] dispose de 3 modèles de connaissances :

- le « discourse model », qui conserve les dialogues avec les visiteurs et les informations contextuelles (nombre de personnes et interaction...)
- le « user model », qui contient les informations obtenues, sur le visiteur lors du dialogue avec celui-ci
- le « system model » comprend les connaissances du guide, ainsi que ces intentions et ses objectifs courants.

Pour délibérer, la méthode BDI (Beliefs-Desires-Intentions) est utilisée, pour

poursuivre plusieurs plans (i.e. intentions) afin de satisfaire les objectifs (i.e. désirs) avec les connaissances, liées au contexte (i.e. croyances). Pour le dialogue et la compréhension, le système utilisé est similaire à [8], avec la comparaison de règles, qui activerons, ou non, des règles d'interprétations (bonjour ou question) et de dialogue.

De même [1, 14], l'agent virtuel Tinker peut guider les visiteurs vers d'autres expositions du musée réel, et demande de revenir le voir pour en parler. En effet, l'agent garde en mémoire les informations (dialogues, empreinte de la main) des visiteurs, qu'il peut « reconnaître » avec une identification biométrique (identification de la personne avec sa main) et, une justification vocale, en cas de doute. Le comportement de l'agent fonctionne avec un réseau de transitions, basé sur un modèle de dialogue.

Cette stratégie pose les mêmes problèmes que la visite interactive, sauf que le visiteur n'a plus à faire l'effort pour proposer explicitement une requête. En effet, le guide pourrait lui proposer cette requête automatiquement, en détectant les intentions du visiteur. Ainsi, elle pourrait devenir un outil important dans le stage, pour permettre au guide de s'adapter aux visiteurs.

2.1.4 La visite pro-active

La visite pro-active est le dernier type de visite, qui consiste à construire la visite guidée au fur et à mesure, en fonction des objectifs du guide et des intentions du visiteur. Son parcours n'est pas plus préparé à l'avance.

Le guide virtuel Elva [15] a la particularité de fournir une visite dynamique à l'initiative du visiteur ou du guide lui-même. Ainsi, le visiteur peut choisir de suivre le guide, ou de naviguer librement dans le musée virtuel, dans ce cas, le guide le suivra, afin de toujours lui présenter les informations. Pour y arriver, Elva utilise une base de connaissance, où elle y choisie les informations à dire, et prévoit les prochaines explications, conduisant à la prochaine œuvre.

Cette base de connaissances est constituée de schéma (figure 2.6 page suivante). Un schéma (représenté par des ovales sombres) est un groupement de plusieurs connaissances (les carrés noirs). Les connaissances sont reliés entre elles, et forme ainsi une structure narrative. Les schémas qui traitent une même information, comme un artiste ou une œuvre, sont regroupés pour formés un domaine (les ovales clairs). Enfin, les connaissances d'un domaine peuvent amener le discours à changer de domaine. Dans l'exemple de la figure 2.6 page suivante, les informations sur un artiste particulier (ovale clair à gauche) peut amener à parler de la technique utilisée (ovale clair à droite), pour une œuvre que le visiteur observe.

Durant la visite guidée, le guide utilise deux plans, contenus dans une librairie de plans. Le premier plan, « tour plan » liste les éléments principaux, qui formeront la visite (introduction, œuvre...). Le second plan, « discourse plan », détaille les buts à atteindre, pour un élément particulier du « tour plan », comme donner des explications ou décrire une œuvre (figure 2.7 page suivante).

Trois outils permettent de gérer les plans, durant la visite :

- le planificateur (Planner) sélectionne un plan que le guide devra remplir, activé par une action du guide ou du visiteur (s'approcher d'une œuvre)
- l'ordonnanceur (Scheduler) choisi et trie les plans à effectuer (les explications à donner) pour le guide.

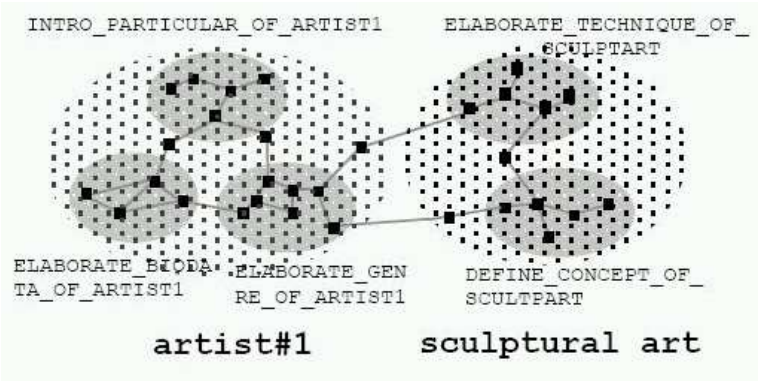


FIGURE 2.6 – Élaboration d'un discours à partir des points de connaissances

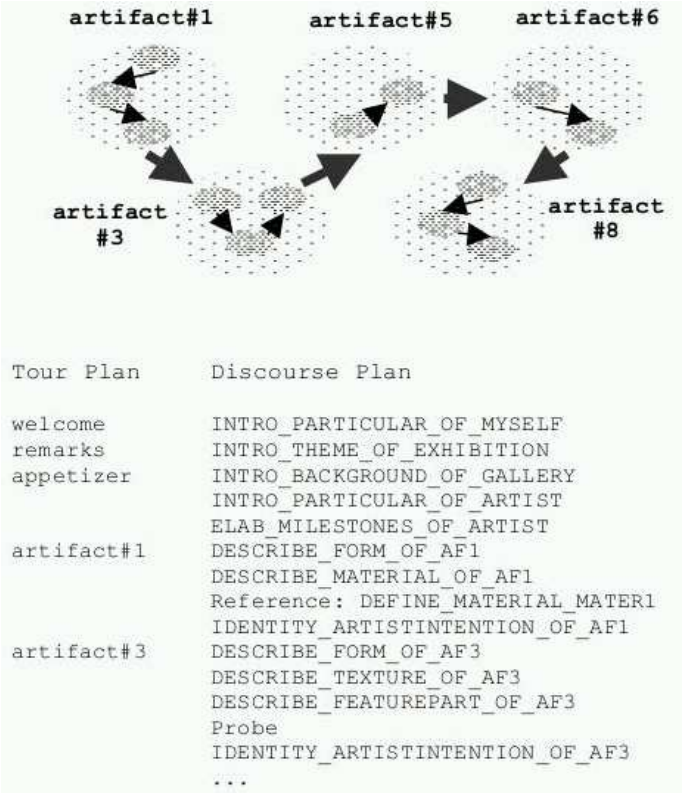


FIGURE 2.7 – Les deux types de plans, pour une visite.

– le gestionnaire de dialogue (Dialog Coordinator) permet d'adapter l'état du dialogue en fonction des actions du visiteur (attente lors des manipulations d'une œuvre, renforcer la dialogue si le visiteur fait beaucoup d'actions et de demandes d'informations...)

Les avantages de cette méthode sont nombreux, notamment la création d'une visite dynamiquement, ce qui rend cette visite unique.

2.1.5 Comparatif des visites

Type de la visite	**Autonomie du visiteur**	**Autonomie du guide**	**Déplacements contrôlés par**	**Parcours suivi**	**Méthodes utilisées**	**Intérêt**
Statique	Faible	Élevée	Guide	Pré-défini (Guide)	Planificateurs	Visite sous forme d'une structure narrative
Interactive	Élevée	Faible	Visiteur	Visiteur	Déclencheurs, fonction de la position ou d'une requête...	Limité car reposant sur les actions du visiteur
Réactive	Moyenne	Moyenne	Visiteur	Actions du visiteur Réactions du guide	BDI	Limité car reposant sur les actions du visiteur
Pro-active	Bonne	Bonne	Visiteur + Guide	Inconnu Visiteur + Guide	BDI + planificateurs	Choix de cette visite pour la suite du stage

2.2 Les comportements autonomes

Pattie Maes [8] a défini une méthodologie pour construire des agents, ou comportements autonomes :

> « [A] particular methodology for building [agents]. It specifies how ... the agent can be decomposed into the construction of a set of component modules and how these modules should be made to interact. The total set of modules and their interactions has to provide an answer to the question of how the sensor data and the current internal state of the agent determine the actions ... and future internal state of the agent. An architecture encompasses techniques and algorithms that support this methodology. »

L'architecture proposée consiste en un ensemble de comportements indépendants. Ces comportements sont en compétition dans un réseau hiérarchique, afin de faire ressortir le comportement le plus adapté au contexte donnée. Pour cela, les comportements peuvent activer ou inhiber d'autres comportements.

Sur le même principe, Brooks a développé une architecture basé sur des automates hiérarchiques (nommée subsumption architecture). Ce concept a été repris dans un langage, formalisant la description d'automates hiérarchiques : HPTS (hierarchical parallel transition system, [4]) et son successeur HPTS++.

Une application de HPTS est le musée virtuel, de la Cité des Sciences et de l'Industrie (2.8 page suivante). Inauguré en juin 2001, ce musée virtuel est devenu une exposition permanente.

A l'aide d'un écran tactile, le visiteur réel peut modifier des paramètres comportementaux des personnages autonomes. Comme dans les musées réels, plusieurs types de personnages autonomes sont présents, comme :

- des visiteurs
- un gardien
- une voleuse
- un photographe
- une mère et sa fille
- une guide
- une restauratrice

Les comportements des personnages reposent sur différentes variables, qui peuvent être modifiées par l'utilisateur. Par exemple, la mère possède un niveau d'intérêt pour la visite et d'attention aux actions de sa fille, qui la conduisent à suivre la visite et/ou surveiller son enfant. Pour le gardien, il est possible de faire varier son niveau d'attention qu'il porte sur la voleuse, le photographe et l'enfant.

FIGURE 2.8 – Capture d'écran du musée virtuel de la Cité des Sciences

Chapitre 3

Principes

3.1 Introduction

Rappelons les objectifs du stage. Nous cherchons à créer un guide virtuel, de type pro-actif, qui doit organiser la visite d'une exposition en tenant compte des objectifs des commissaires de l'exposition et des souhaits des visiteurs. Le principe retenu repose sur une boucle (Figure 3.1) : le visiteur suit une visite. Durant celle-ci, il peut indiquer des souhaits en sélectionnant des mots-clés dans le musée. Ces mots-clés peut conduire à choisir une autre visite. Une visite est le produit d'un processus de décision (continuer ou changer la visite) et correspond à la combinaison de visites connues à priori.

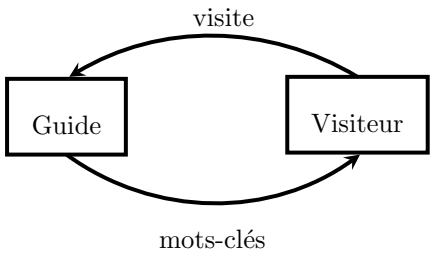


FIGURE 3.1 – Schéma illustrant l'échange entre le guide et le visiteur

Nous allons commencer par étudier l'organisation d'un musée.

3.2 Organisation d'un musée

3.2.1 Structuration d'un musée

L'architecture d'un musée (Figure 3.3 page suivante) peut être considérée comme un lieu d'exposition, possédant plusieurs espaces différents (ailes, couloirs, halls...), ayant des thèmes différents. Ces espaces thématiques sont divisés en salles, qui contiendront les objets associés au thème. De plus, chaque objet (ou paire d'objets) sera associé des annotations multimédia. Cette classification des objets en thèmes/endroits participe à la création de visites, que nous allons voir.

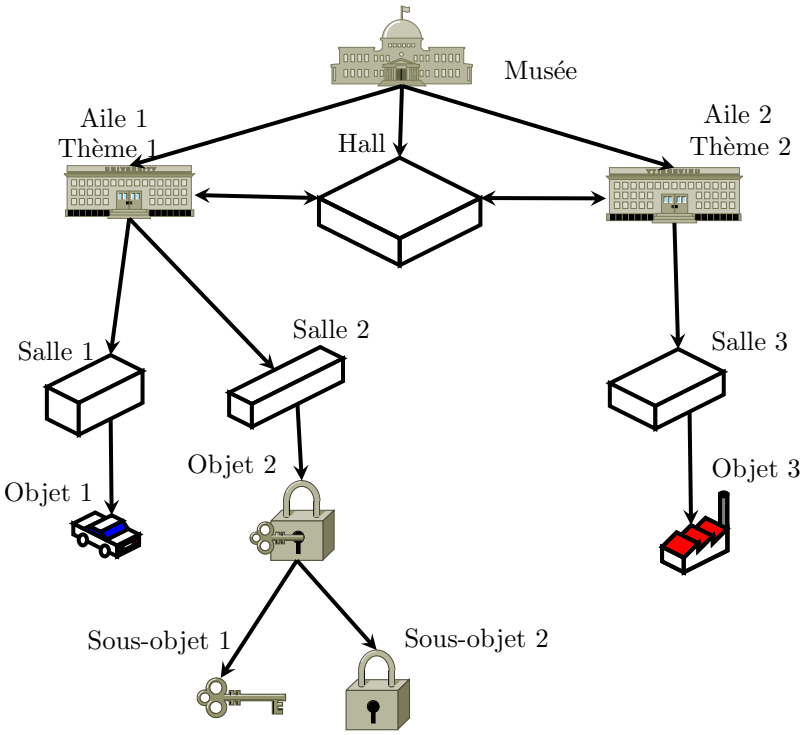


FIGURE 3.2 – Schéma de l'organisation d'un musée

3.2.2 Qu'est-ce qu'une visite ?

Une visite est une liste d'objets, généralement ordonnée et associée à un thème précis. Elle permet de donner un ordre logique de visite, et de présentation, des différentes objets du musée. Elle se présente comme une structure narrative, afin d'éviter des incompréhensions ou des oublis. Enfin, celle-ci peut être découpée en sous-visites, afin de pouvoir passer d'une visite à une autre (visite = objet non monolithique), ceci pour tenir compte des souhaits du visiteur.

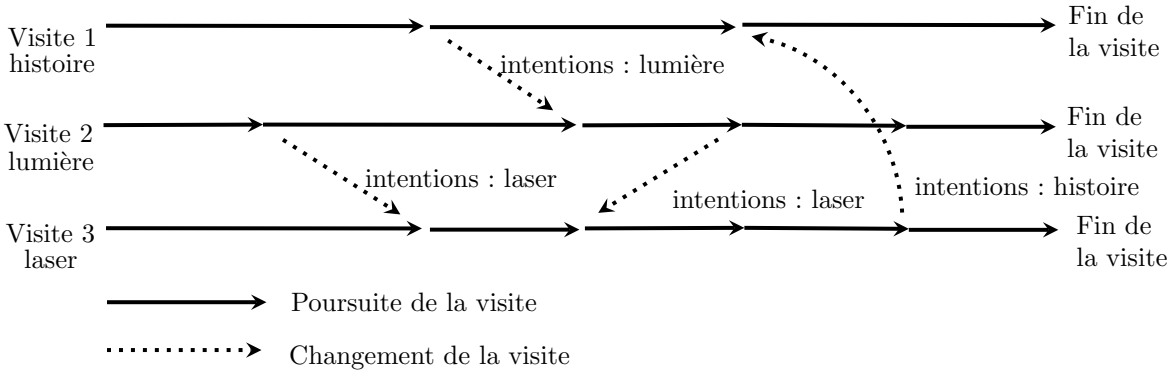


FIGURE 3.3 – Poursuites et changements de visites en fonction des intentions du visiteur

3.3 Guide

3.3.1 Qu'est-ce qu'un guide ?

Les dictionnaires définissent un guide comme :

> « une personne qui accompagne pour montrer le chemin ou donner des explications sur quelque chose »
> tirée du *Dictionnaire de la langue Française, Bordas*

Dans le cadre d'un musée, le guide effectue ces deux tâches. Il doit guider et partager son savoir auprès du visiteur, au travers les différents objets que comprennent la visite.

3.3.2 Représentation des intentions du visiteur

Un guide peut détecter les intentions du visiteur de plusieurs manières par :
- le dialogue via des questions-réponses
- des statistiques, sur le nombre de temps passé sur un objet ou un thème, par exemple
- les déplacements et le regard du visiteur, indiquant ce qu'il aimerait voir.

Mais, l'environnement peut aussi participer à l'acquisition des intentions du visiteur. Un exemple 3.4 est de proposer des éléments cliquables dans le monde virtuel.

FIGURE 3.4 – Exemple d'un élément cliquable « LASER », renseignant les intentions du visiteur

Pour permettre l'accès à toutes les informations recueillies, nous utilisons une mémoire.

3.4 Sélection d'une sous-visite

3.4.1 Mémoire du visiteur

Dans le cadre de ce travail, nous posons la mémoire du visiteur, comme un objet contenant :

- la position du visiteur
- la position du regard du visiteur
- la sous-visite courante
- des informations permanentes associées visiteur
- des informations temporaires associées au visiteur

Cette mémoire est partagée (Figure 3.5), permettant :

- au collège d'experts de connaitre les informations sur le visiteur et de voter la prochaine visite en fonction de celles-ci
- au guide de pouvoir modifier la visite suivie, en fonction du vote du collège d'experts.

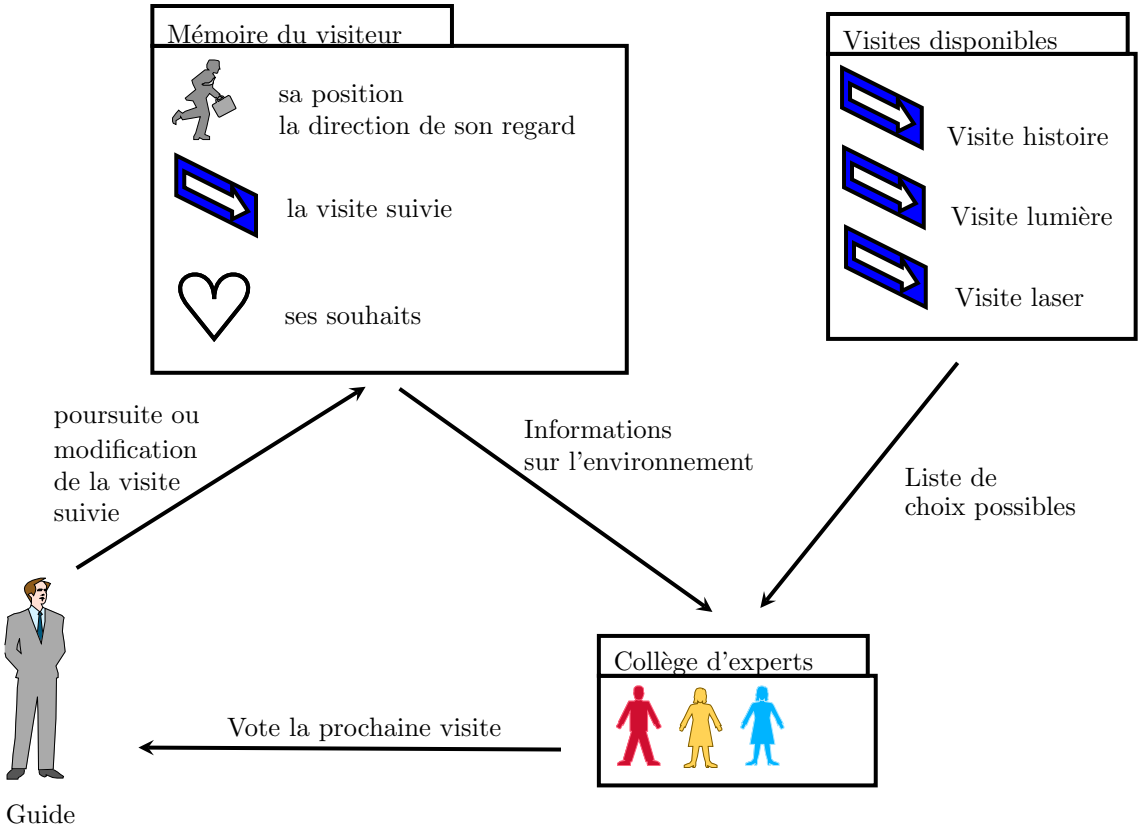


FIGURE 3.5 – Schéma de la mémoire partagée

Les informations temporaires sont effacées au bout d'un certain temps, ce qui aura pour effet de conserver les informations les plus récentes, donc une "image" de l'état courant du visiteur. Elles indiquent les actions du visiteur, et par conséquent, une partie de ces intentions.

En revanche, les informations permanentes concernent les informations "personnelles" et immuables du visiteur, qui influenceront les prochaines sous-visites, comme :

- le niveau/age du visiteur (adulte, enfant...)
- les objets déjà visitées
- les objectifs/souhaits initiaux du visiteur (voir tel thème ou objet...)

3.4.2 Collège d'experts

Pour simplifier le travail du guide, la décision de la prochaine visite est faite à l'aide d'un ensemble d'experts. Ces experts ont tous le même rôle, c'est à dire, voter pour (ou s'opposer contre) des sous-visites, mais, il possèdent des critères de vote différents. Parmi ces critères de votes, nous pouvons citer :

- les critères de déplacements, qui essayent de minimiser les déplacements du visiteur
- les critères de poursuite, qui favorisent la suite prévue (ou logique) de la visite
- les critères intentionnels, qui prennent en compte les intentions du visiteur

Ces experts ont accès à une liste de sous-visites, ainsi qu'à la mémoire du visiteur. L'algorithme de vote du collège d'experts est présenté dans le listing 3.1.

Listing 3.1 – Algorithme de vote du collège d'experts

```
#fonction retournant la sous-visite selectionnee par les experts
#si aucune sous-visite est possible, on retourne None (par defaut)
def demandeExpert(self):
  #intialisation des variables
  maxVoix = 0
  resultatVoix = None

  #chaque expert vote
  for unExpert in self.experts:
    unExpert.vote()

  #initialise le tableau de resultats
  self.resultatsSousVisites = []

  #Pour chaque indice de sous-visite
  for numeroSubVisite in range(len(self.subVisites)):
    #intialisation des variables
    sommeVoix = 0
    numExpert = 0
    nombreExperts = len(self.experts)

    #Boucle tant qu'il n'y a pas de veto sur la visite (-1)
    #et qu'il nous reste des experts
    while(sommeVoix>-1)and(numExpert<nombreExperts):
```

```
            #on recuppere le vote de l'expert pour la sous-visite
            valeur = self.experts[numExpert].resultatSousVisite[numSubVisite]
            #on multiplie ce resultat par le nombre de voix attribue a l'expert
            valeurPonderee = valeur * self.experts[numExpert].nombreVoix

            #Si la valeur est -1, sous-visite eliminee (veto de l'expert)
            #sinon, on additionne au resultat de la sous-visite
            if(valeur==-1):
              sommeVoix=-1
            else:
              sommeVoix=sommeVoix+valeurPonderee

            #on passe au prochain expert
            numExpert = numExpert +1

          #on conserve la valeur maximale des votes
          if(maxVote<sommeVoix):
            maxVote=sommeVoix

          #on ajoute le resultat de la sous-visite
          self.resultatsSousVisites.append(sommeVoix)

        #retrouve les sous-visites selectionnees
        if(maxVote > 0):
          #initialise la liste de sous-visites selectionnees
          self.choix = []
          #Pour chaque indice de sous-visite
          for i in range(len(self.resultatsSousVisites)):
            #Si la sous-visite a acquis suffisament de voix,
            #on la selectionne
            if(self.resultatsSousVisites[i]==maxVote):
              self.choix.append(self.subVisites[i])

          #on choisi aleatoirement la sous-visite
          #parmi celles selectionnees
          numAchoisir = random.randint(1,len(self.choix))
          resultatVoix = self.choix[numAchoisir-1]

        #retourne la sous-visite selectionnee
        return resultatVoix
```

3.5 Comportements du visiteur

3.5.1 Steering

Notre guide étant pro-actif, il doit pouvoir imposer un comportement correspondant aux objectifs pédagogiques tout en tenant compte du comportement du visiteur. Ainsi, il doit pouvoir combiner des comportements d'origines diverses :

– l'environnement (éviter les collisions avec un mur)

– le guide virtuel (arriver en un point, regarder un point précis)
– le visiteur

Le visiteur est représenté par un avatar. Celui-ci est controlé en utilisant la méthode de steering, proposé par Reynolds [11]. Etant donné une force $\overrightarrow{F}$, on peut en déduire par double intégration la vitesse et la position de l'avata représentant le visiteur. Cette force dépend de l'objectif à atteindre, comme :

– atteindre un point
– rester à une distance donné d'une trajectoire
– satisfaire les commandes données par l'utilisateur

Un comportement complexe est obtenu en combinant linéairement plusieurs comportements élémentaires (combinaison linéaire des forces).

Pour expliquer ces comportements, prenons le cas d'une voiture (en 3.6). Pour qu'une voiture se déplace, elle nécessite plusieurs forces, dont :

– une force dirigée vers l'avant du véhicule (vitesse positive), qui permet d'accélérer
– une force dirigée vers l'arrière (vitesse négative), qui permet de décélérer (voire freiner ou reculer)
– une force dirigée sur les côtés, qui permet de tourner

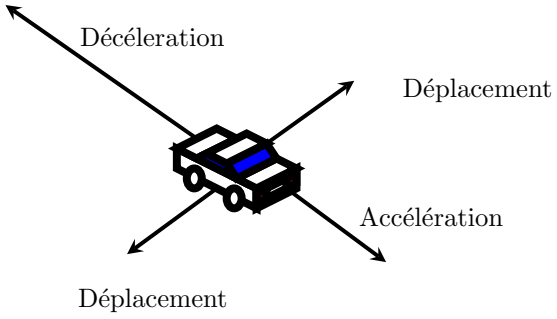


FIGURE 3.6 – Schéma des forces d'une voiture

Dans notre cas, notre visiteur est constitué de deux comportements différents. Le premier comportement s'occupe de ses déplacements dans le musée. Le second comportement concerne la direction de son regard.

Ces comportements sont constitués d'un ensemble de forces. Parmi ces forces, nous trouvons :

– la force du guide, qui amène le visiteur dans une direction
– la force "interne" du visiteur, contrôlée par l'humain
– des forces d'attractions (vers un objet, par exemple)
– des forces de répulsions (prés d'un murs, par exemple)

La prochaine position du visiteur est calculée à partir de sa vitesse courante et de la somme de toutes les forces multipliées par leurs poids respectifs. Ces poids permettent de donner plus ou moins d'importance dans le comportement final. Pour s'adapter aux différentes situations de ralentissement ou d'accélération, le type de ces forces et la valeur des poids doivent être changées. Pour cela, nous utilisons des automates.

3.5.2 Automates

Une visite ne peut s'effectuer en utilisant un comportement uniforme. Par exemple : on accélére entre 2 objets et on ralentit à l'approche d'un objet. On laisse l'utilisateur plus libre de façon à ce qu'il puisse explorer son environnement. On doit donc considérer qu'au cours d'une visite, le comportement de l'utilisateur doit être modifié.

Nous supposons ici que ce comportement est décrit par les paramétres comportementaux, tels que le type de la force utilisée et la valeur du poids, et la façon dont ils sont combinés. L'avatar du visiteur passe d'un comportement à un autre sous-certaines conditions. On modélise cela au moyen d'un automate de Moore :

- un état de l'automate correspond à un jeu de paramétres comportementaux.
- une transition entre 2 états s'effectue quand une condition particulière est satisfaite.

Le visiteur possède un automate qui :

- initialise les positions
- initialise les forces pour se déplacer à l'aide des touches claviers
- détecte les objets cliqués.

L'automate du guide (schéma 3.7 page suivante) s'occupe de demander aux experts la prochaine visite à effectuer et de préparer le visiteur à suivre cette visite, en plaçant une force « guide »dans le comportement du visiteur.

Afin de simplifier le travail du guide, les objets possèdent des automates qui leurs permettent de lancer les explications, quand le visiteur s'approche d'eux.

Voici un exemple d'un état initial, de l'automate du visiteur (listing 3.2). L'objectif de cet état est d'initialiser la position du visiteur (à l'aide de la variable *setPosition*) ainsi que son comportement. Pour cela, nous renseignons une vitesse maximale ($setVelocityMax$). Ensuite, nous créons une force, pour permettre de déplacer le visiteur ($createMyBehavior$). L'attribut *parameters* contient les paramètres liés à cette force. Dans cet exemple, les paramètres indiquent les limites de cette force, c'est à dire, elle ne doit pas dépasser la valeur -1 et 1, avec un pas de 1, en abscisse et en ordonnée. Enfin, nous devons renseigner un poids pour chaque force, qui constitue le comportement. Ici, le poids de cette force est de 0,3 (attribut $setWeight$). En revanche, la valeur du poids, liée à la force du guide qui sera combinée à celle-ci, est de 0,7. Cette force du guide (décrite dans la partie 4.5.1 page 33) est activée que sous certaines conditions. Lors de cette activation, celle-ci doit être pré-dominante dans le comportement du visiteur. Pour cela, nous avons choisi d'utiliser une valeur deux fois supérieure au poids de la force du visiteur. Lorsque la force du guide n'est pas activée, celle-ci est alors nulle, permettant aux autres forces d'agir.

Listing 3.2 – Exemple d'un état, qui initialise le comportement du visiteur

```
<State name="Init" initial="1">
  <Method owner="BOID"
    name="setPosition" parameters="19.3 1.6 0.0" />
  <Method owner="BOID"
    name="setVelocityMax" parameters="1" />
  <Method owner="BOID"
    name="setWeight" parameters="0.3" />
```

```
  <Method owner="BOID"
    name="createMyBehavior" parameters="-1 1 1 -1 1 1" />

  <!-- de meme pour le comportement associe au regard -->

  <!-- apres l'initialisation des comportements,
  nous transiterons vers le prochain etat enAttente -->
  <EpsilonTransition name="aller_enAttente">
    <State name="enAttente"/>
  </EpsilonTransition>
</State>
```

Durant l'exécution d'un automate, il est possible de faire appel à un autre automate. Ce nouvel automate est mis dans une pile, connue par l'automate principal. Seul l'automate le plus récent dans la pile s'execute, les exécutions

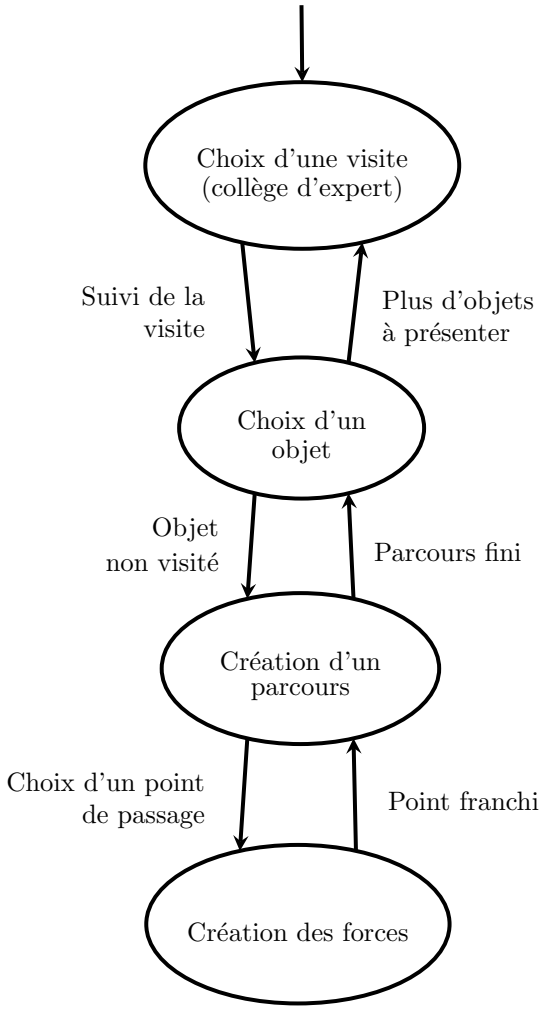


FIGURE 3.7 – Schéma d'un automate pour le suivi d'une visite par un guide

des autres automates sont suspendus. Le sous-automate se termine quand il atteint un état terminal, dans ce cas, il est retiré de la pile, et l'exécution du dernier automate dans la pile reprend. Ce système permet de déléguer des tâches précises (modification de la vitesse du visiteur, interactions avec les objets...) à partir d'un automate plus simple, et ainsi avoir des comportements qui varient en fonction des actions (configuration d'une pièce, approche ou éloignement d'un objet...)

Chapitre 4

Mise en œuvre

4.1 Introduction

Un musée virtuel a été réalisé en Python. Nous avons utilisé une librairie [1] pour gérer l'environnement graphique et sonore. ainsi qu'une autre librairie pour lire les fichiers multimédia compressés [2].

Pour illustrer la mise en œuvre de ce musée virtuel, des schémas UML, des différentes parties qui seront expliquées dans ce chapitre, ont été placées en Annexe A page 37.

Le programme principal crée un objet MuseumProject. C'est MuseumProject qui donne vie au musée, en créant l'environnement graphique et en récupérant les événements utilisateur (clavier, souris et joystick). Le monde virtuel est créé à partir des fichiers de configurations. Il existe plusieurs fichiers de configurations, une pour chaque partie du musée, comme :

- la description de l'architecture du musée
- les informations associés aux objets
- les points de navigations

L'architecture du musée et les informations associés aux tableaux sont décrit dans des fichiers XML. Dans le fichier lié à l'architecture du musée (listing 4.1), chaque objet du musée est représenté par une balise. Pour positionner et orienter l'objet, il faut ajouter une balise Transform. Différents objets sont possibles comme :

- un ciel (*skybox*)
- des murs
- des sols
- des affiches, objets non-interactifs sans explications
- des tableaux, objets interactifs avec explications

Listing 4.1 – Exemple d'une architecture d'un musée

```
<Scene>

  <!-- Un musee avec un ciel, un sol, un mur et un
       tableau
```

1. Librairie graphique et multimédia pour Python : Pyglet (www.pyglet.org)
2. Librairie pour codecs multimédia : AVbin (http://code.google.com/p/avbin/)

```
  -->

  <Transform translation="36 0 0.0">
    <!-- Place un ciel dans notre environnement -->
    <SkyBox taille="100" texture="ciel.jpg" />
  </Transform>

  <Transform translation="21.6 0 0">
    <!-- Place un sol -->
    <Sol taille="7.2" texture="textures/sol.jpg" />
  </Transform>

  <Transform translation="36.4 0 -2.5"
       rotation="-90.0 0.0 1.0 0.0">
    <!-- Place un mur -->
     <Cloison largeur="5" hauteur="3"
     recto="textures/murs.jpg"  />
  </Transform>

  <Transform translation="36.2 0.5 1"
       rotation="-90.0 0.0 1.0 0.0">
    <!-- Place un tableau -->
    <Tableau largeur="1.2" hauteur="2"
       texture="poster.jpg" uid="poster"/>
  </Transform>

</Scene>
```

Tout comme l'architecture du musée, l'information d'un tableau est décrite à l'aide d'une balise (listing 4.2). Les informations associés au tableau sont :

- un identifiant unique (*uid*)
- le nom complet du tableau (*fullname*)
- le point de passage le plus proche du tableau (*piece*)
- les explications sonores (*son*)
- le nom (*otherName*) du comportement à charger dans le fichier (*file*)

Listing 4.2 – Exemple d'une information pour un tableau

```
<Tableaux>

        <Tableau
        uid="poster"
        fullname="Titre"
        piece="Hall"
        son="fichierSon.mp3"
        file="comportement.xml"
        otherName="nomDuComportement"/>

</Tableaux>
```

Enfin, le fichier contenant les points de navigation, décrit un graphe, sous

forme d'un texte. On retrouve les points de passage (*sommets*), dont ceux associés aux tableaux, et les liaisons entre ces points (*arc*).

4.2 Organisation des visites

Nous avons utilisé la définition d'une visite, présentée en 3.2.2 page 19. Ainsi, la classe MuseumProject connaît toutes les visites possibles, renseignées par un fichier XML. Vous trouverez un exemple de ce fichier dans le listing 4.3. Ce fichier est constitué par des ensembles ordonnés de $Visites \rightarrow Sous\text{-}visites \rightarrow Posters$.

Listing 4.3 – Exemple d'une visite enfant

```
<Visites>
  <Visite name="Enfant" >
    <SousVisite name="enfant_titre"
        keywords="enfant,titre"
        initial="1">
      <Poster name="0px_titre" />
    </SousVisite>
    <SousVisite name="enfant_histoire"
        keywords="enfant,histoire">
      <Poster name="1_pxhistoire" />
    </SousVisite>
    <SousVisite name="enfant_lumiere"
        keywords="enfant,lumiere">
      <Poster name="3_pxlum" />
      <Poster name="5_pxlum" />
    </SousVisite>
    <SousVisite name="enfant_laser"
        keywords="enfant,laser">
      <Poster name="7_pxlas" />
      <Poster name="8_pxlas" />
      <Poster name="9_pxlas" />
      <Poster name="11_pxlas" />
    </SousVisite>
    <SousVisite name="enfant_applications"
        keywords="enfant,applications">
      <Poster name="13_pxappl" />
      <Poster name="14_pxappl" />
      <Poster name="16_pxappl" />
    </SousVisite>
  </Visite>
  <!-- autres visites ... -->
</Visites>
```

Ces visites sont des instances de la classe Visite, et contiennent une liste de SousVisites. Ces SousVisites indiquent les tableaux à voir, la prochaine sous-visite liée, ainsi que des mots-clés (*keywords*), que nous allons voir dans la prochaine partie.

4.3 Utilisation de mots-clés

Pour identifier les souhaits du visiteur, nous avons placés dans le musée des objets cliquables (image 3.4 page 20). Pour ajouter la possibilité de cliquer sur un objet, nous rajoutons un attribut « clic=true »dans la balise de l'objet concerné, se trouvant dans le fichier servant à décrire l'architecture du musée.

Pour connaitre les objets cliqués par l'utilisateur, et donc les mots-clés associés, nous avons utilisé la technique des fausses couleurs (Figure 4.1 page suivante). Elle consiste à associer une couleur unique pour chaque objet. Quand l'utilisateur clique sur un objet, nous remplaçons les couleurs de chaque objet du musée par la couleur qui lui est associé, et nous récupérons la couleur située à l'endroit du clic, et par conséquent, l'objet rattaché. L'objet possède un mot-clé qui sera ajouté aux informations temporaires, dans la mémoire du visiteur, étudiée dans la partie 3.4.1 page 21.

4.4 Collège d'experts

Pour la décision de la prochaine visite, nous avons choisi de favoriser :

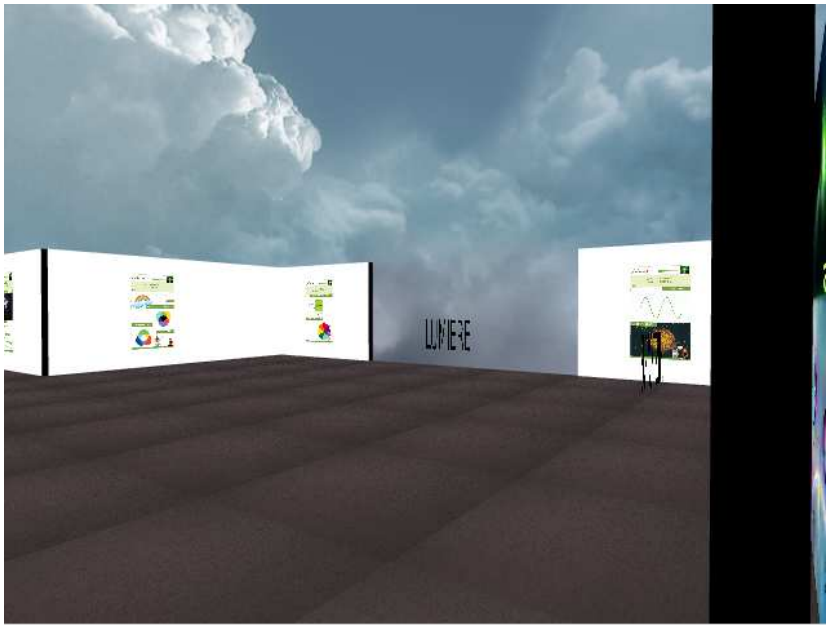
1. la visite correspondant aux souhaits du visiteur
2. la poursuite de la visite courante

Pour cela, nous avons créé des experts (*ExpertNext* et *ExpertBestMatchingKeywords*, listing 4.4) pour chacune des visites mentionnées ci-dessus.

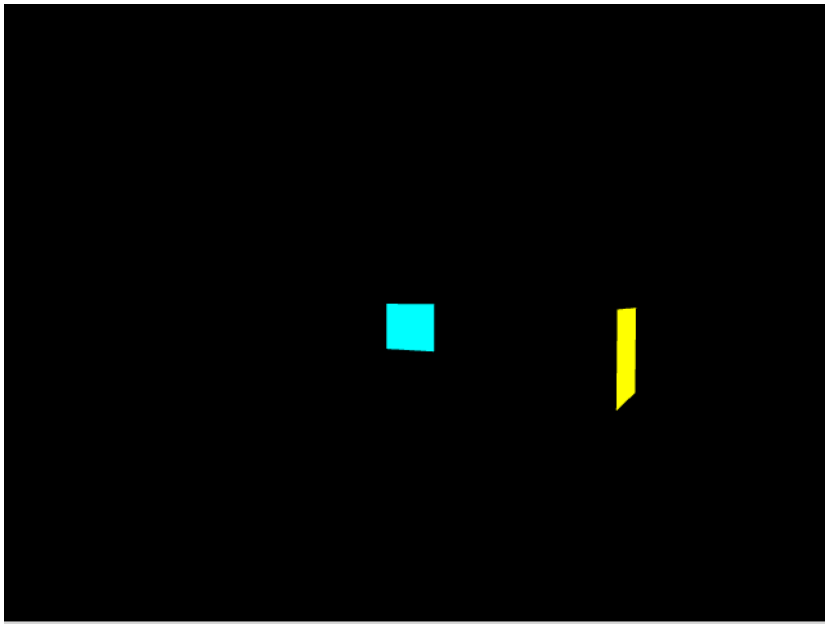
Listing 4.4 – Exemple d'un collège d'experts

```
<Description>

  <!-- College de 2 experts :
  le premier favorise la poursuite de la visite
  le deuxieme regarde les visites comprenant le plus
  grand nombre de mots-cles
  nous favorisons le deuxieme expert en lui donnant
  2 voix, au lieu d'une
  -->
  <Experts name="normal">
    <ExpertNext/>
    <ExpertBestMatchingKeywords vote="2"/>
  </Experts>

</Description>
```

Pour favoriser la visite correspondant aux souhaits du visiteur, *ExpertBestMatchingKeywords* compare les mots-clés contenus dans chaque visite aux mots-clés placés dans la mémoire du visiteur, à l'aide des informations temporaires et permanentes. Il choisi alors la visite ayant le plus de mots-clés correspondant.



(a) Capture d'écran du musée virtuel, avec les vraies couleurs



(b) Capture d'écran du musée virtuel, avec les fausses couleurs

FIGURE 4.1 – Technique pour retrouver l'objet 3D sélectionné par l'utilisateur

4.5 Contrôle du visiteur

4.5.1 Steering

Nous avons mis en place les forces de base, mentionnées par Reynolds dans [11], notamment :

- une force pour aller vers un point, sans s'arrêter sur celui-ci
- une force pour arriver sur un point, en s'arrêtant sur celui-ci
- une force pour s'arrêter sur place (opposée à la vitesse actuelle).

Pour éviter que le visiteur s'éloigne du parcours (et donc de sa visite), nous avons mis en place une force ≪ corridor ≫(figure 4.2 page suivante). Elle consiste à placer le visiteur dans une zone, comprise entre un point de départ et un point d'arrivé, ayant la forme d'un corridor. Le visiteur est libre de ces mouvements à l'intérieur de cette zone. Si il sort de cette zone, le visiteur sera ramené dans la zone. Cette force est créée par le guide, lors du suivi d'un parcours, et placée dans le comportement du visiteur. Ainsi, elle permet au visiteur de regarder et cliquer sur les objets situé dans la zone, permettant d'acquérir ses intentions, sur une zone donnée.

Pour permettre de déplacer le visiteur dans l'environnement, nous avons ajouté deux nouvelles forces : l'une pour la position du visiteur et l'autre pour la direction de son regard. Elles sont utilisées et modifiées, par l'automate du visiteur, lors de la détection d'un événement (clavier, souris ou joystick) de l'utilisateur pour changer la position ou le regard.

4.5.2 Automates

Nous avons vu que les objets pouvaient posseder un automate, ainsi que le visiteur et le guide. Les automates sont placés et executés à partir de la classe visiteur (appellé *BoidWithGazeBoid* dans l'annexe).

Pour permettre l'appel de méthodes de classes, nous utilisons des "*context*". Leur objectif est de fournir un service (retourner le résultat d'une fonction ou effectuer des modifications à l'aide d'une méthode et des paramètres). Ces *context* sont liés aux méthodes de la classe qu'on demande. Par exemple, si l'automate veut appeler une fonction présente dans le visiteur, il fera appel au *context* associé, qui se chargera de lui retourner la valeur de cette fonction.

L'automate peut faire appel aux *contexts* dans deux endroits, dans :

- un état, avec des *contexts* effectuant une opération/traitement sur des variables dans la classe associée
- une transition, avec des *contexts* retournant des valeurs, qui seront évaluées, selon la condition de la transition

L'avantage de cette méthode est de permettre à n'importe quel automate de pouvoir appeler n'importe quelle méthode, et ainsi créer des comportements complexes.

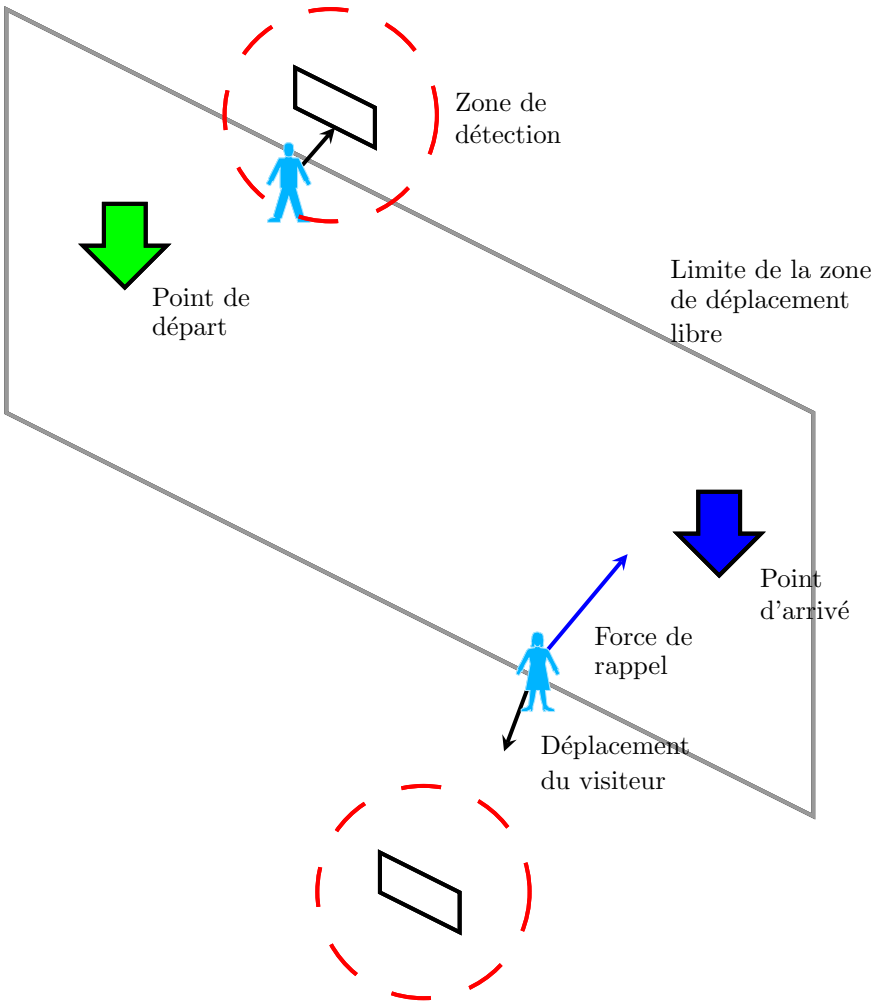


FIGURE 4.2 – Schéma illustrant la force « corridor »

Chapitre 5

Conclusion

5.1 Bilan

Nous venons d'expliquer les principes et proposer une mise en œuvre pour obtenir un guide virtuel pro-actif, qui prends compte des objectifs pédagogiques des commissaires de l'exposition mais également des souhaits des visiteurs.

Pour cela, nous avons utilisé des comportements de Steering, proposé par Reynolds, dans le but de combiner :

- les forces du guide virtuel (les objectifs pédagogiques des commissaires de l'exposition)
- les forces du visiteur (les souhaits de celui-ci)
- les forces de l'environnement (attraction ou répulsion des objets).

Ensuite, nous avons associés ces comportements à des automates de Moore, afin de pouvoir modifier les paramètres comportementaux en fonction de l'environnement courant, pour proposer au visiteur un comportement complexe, mais adapté à la situation donnée.

Enfin, nous avons permis au visiteur de changer la visite en cours. A l'aide de mots-clés, cliquables par l'utilisateur, placés dans le musée, celui-ci indique ces intentions. La décision de la prochaine visite est prise par le collège d'experts, en fonction de ces intentions et de la visite courante, et sera suivie par le guide pro-actif, qui guidera le visiteur.

5.2 Perspectives

Nous envisageons d'utiliser des interfaces plus naturelles, telles que la reconnaissance :

- de gestes,
- du suivi du regard,
- vocale,

pour la désignation des mots-clés.

Nous souhaiterions développer un guide virtuel incarné expressif, doté d'émotions, ainsi que des outils-auteurs pour faciliter la création et la mise à jour d'un musée virtuel.

Enfin, nous prévoyons d'évaluer l'intérêt d'un tel guide auprès du public.

Bibliographie

[1] Timothy Bickmore, Laura Pfeifer, Daniel Schulman, Sepalika Perera, Chaamari Senanayake, and Ishraque Nazmi. Public displays of affect : Deploying relational agents in public spaces. *CHI*, 2008.

[2] Maurizio Bombara, Davide Cali, and Corrado Santoro. Kore : a multi-agent system to assist museum visitors. 2008.

[3] L. Chittaro, L. Ieronutti, and R. Ranon. Navigating 3d virtual environments by following embodied agents : a proposal and its informal evaluation on a virtual museum application. *PsychNology Journal*, 2(1) :24–42, 2004.

[4] S. Donikian. *Acteurs autonomes dans des environnements informés et structurés*. 2004. pages 81-111.

[5] J. Ibanez, R. Aylett, and R. Ruiz-Rodarte. Storytelling in virtual environment from a virtual guide perspective. *Virtual Reality*, 7 :30–42, 2003.

[6] Rieko Kadobayashi and Kenji Mase. Seamless guidance by personal agent in virtual space based on user interaction in realworld. 1998.

[7] Stefan Kopp, Lars Gesellensetter, Nicole C. Krämer, and Ipke Wachsmuth. A conversational agent as museum guide – design and evaluation of a real-world application. 2005.

[8] Pattie MAES. The agent network architecture. 1991.

[9] Victor Mateevitsi, Michael Sfakianos, George Lepouras, and Costas Vassilakis. A game-engine based virtual museum authoring and presentation system. 2008.

[10] T. Panayiotopoulos, N. Zacharis, and S. Vosinakis. Intelligent guidance in a virtual university. pages 33–42, 1998.

[11] C. W. Reynolds. Steering behaviors for autonomous characters. 1999.

[12] A. Santangelo, A. Augello, A. Gentile, G. Pilato, and S. Gaglio. A chat-bot based multimodal virtual guide for cultural heritage tours. *PSC*, 2006.

[13] C. J. Saretto. A survey of interaction issues facing navigation in virtual worlds. 1997.

[14] Daniel Schulman, Mayur Sharma, and Timothy Bickmore. The identification of users by relational agents. *International Foundation for Autonomous Agents and Multiagent Systems*, 2008.

[15] X. Yuan and Y. S. Chee. Embodied tour guide in an interactive virtual art gallery. *International Conference on CyberWorlds 2003*, pages 432–439, 2003.

Annexe A

Schémas UML

A.1 Programme et visites

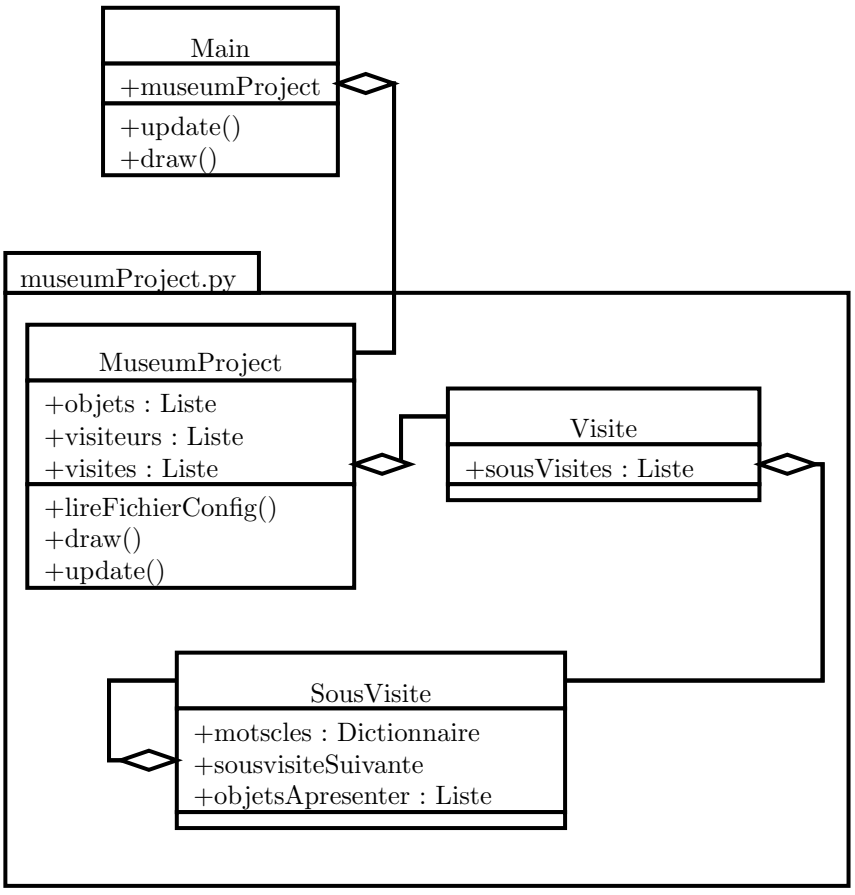


FIGURE A.1 – Schéma UML simplifié du programme principal et des visites

A.2 Visiteurs et guide

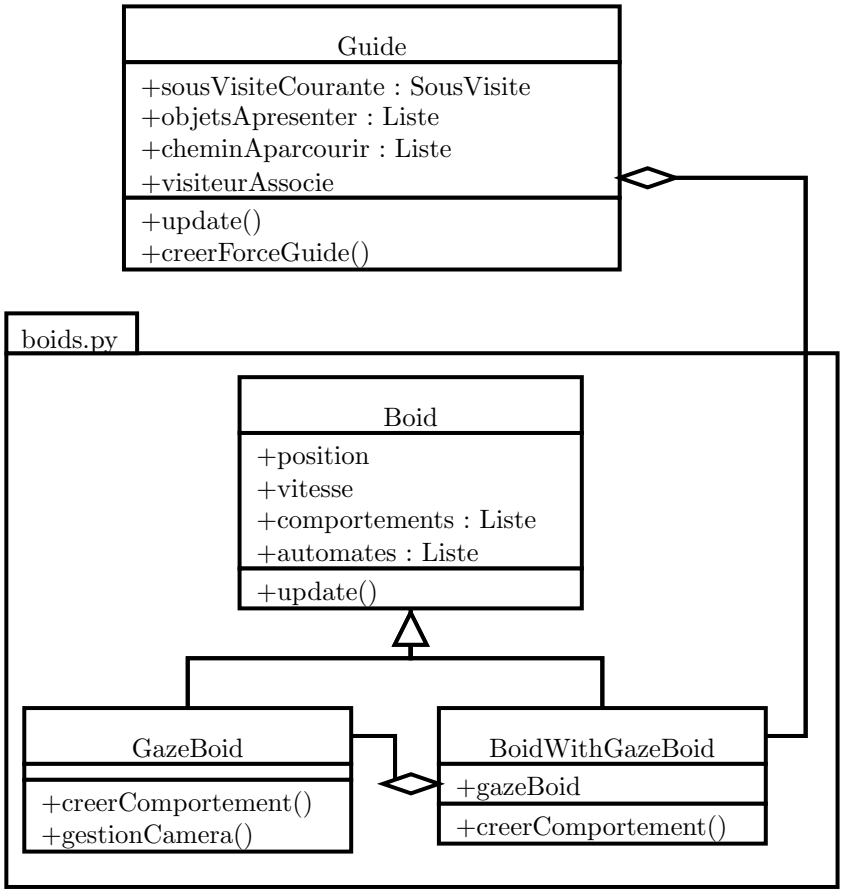


FIGURE A.2 – Schéma UML simplifié des visiteurs et du guide

A.3 Comportements

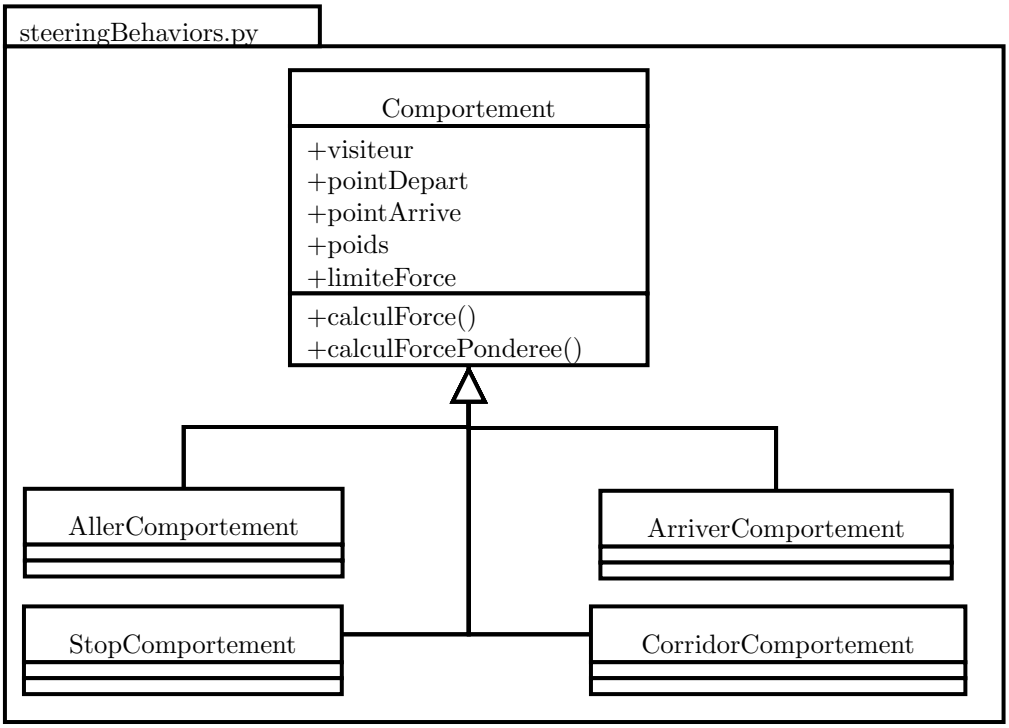


FIGURE A.3 – Schéma UML simplifié des comportements

A.4 Automates

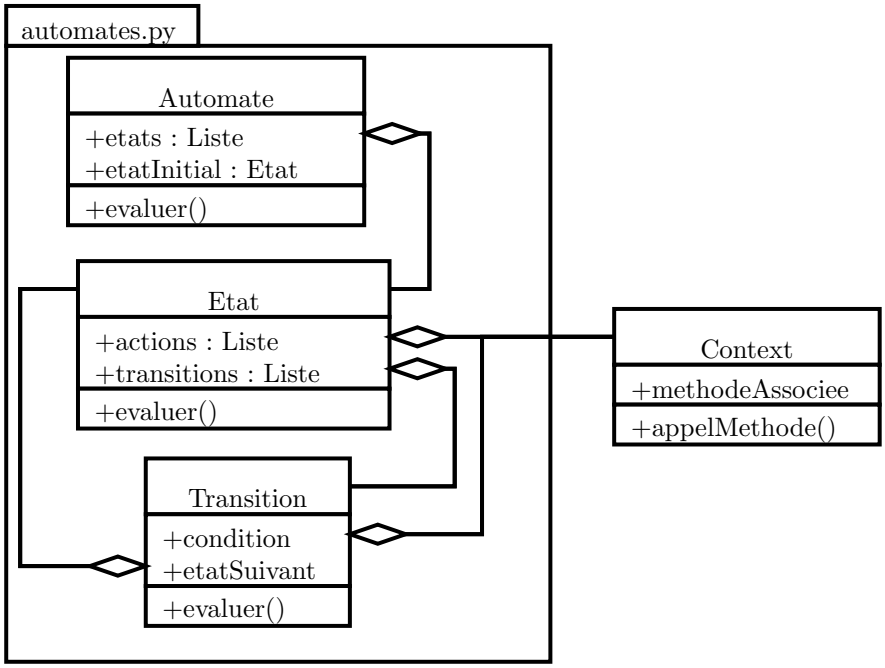


FIGURE A.4 – Schéma UML simplifié des automates

A.5 Mémoire et experts

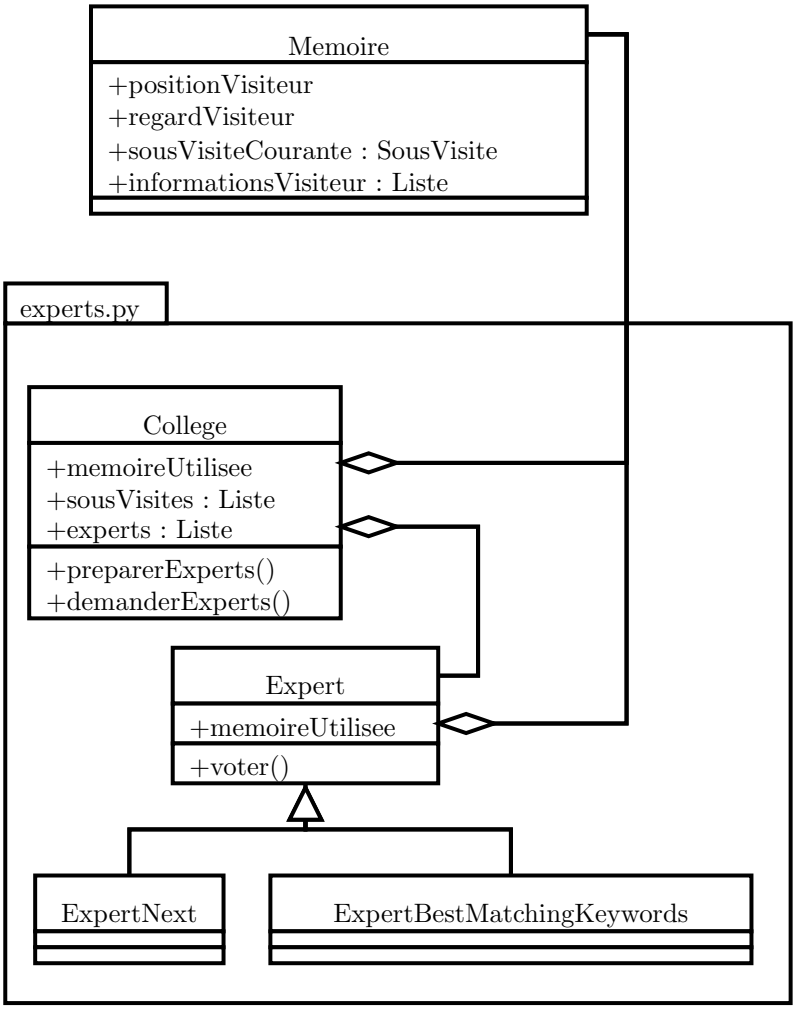


FIGURE A.5 – Schéma UML simplifié de la mémoire et des experts