Virtual human representation, adaptation, delivery and interoperability for virtual worlds

Blagica Jovanova

▶ **To cite this version:**

Blagica Jovanova. Virtual human representation, adaptation, delivery and interoperability for virtual worlds. Other [cs.OH]. Institut National des Télécommunications, 2011. English. NNT : 2011TELE0012 . tel-00712173

HAL Id: tel-00712173
https://theses.hal.science/tel-00712173

Submitted on 26 Jun 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de doctorat de Télécom & Management SudParis dans le cadre de l'école doctorale S&I en co-accréditation avec l' Universite d'Evry-Val d'Essonne

Spécialité :
Informatique

Par
Mlle Blagica JOVANOVA

Thèse présentée pour l'obtention du diplôme de Docteur de Télécom & Management SudParis

Virtual Human Representation, Adaptation, Delivery and Interoperability for Virtual Worlds

Soutenue le 29 Mars 2011 devant le jury composé de :

Mme. Catherine PELACHAUD, Directeur de recherche CNRS, TELECOM ParisTech, Rapporteur
M. Danco Davcev, Professeur à Ss.Cyril and Methodius University, République de Macedoine, Rapporteur

Mme. Béatrice Pesquet-Popescu, Professeur à TELECOM ParisTech, Examinateur
M. Jae-Joon Han, Ingénieur de recherche, Samsung, Korea, Examinateur
M. Jérôme Royan, Ingénieur de recherche,France Télécom, Examinateur

M. Marius Preda, Maître de conférences, Telecom SudParis, Co-encadrant
M. Bruno Defude, Professeur à Telecom SudParis, Directeur de thèse

Thèse Numéro : 2011TELE0012

Acknowledgements

I would like to thank Prof. Françoise Préteux that lead my studies from 2006-2010.

I also owe my thanks to Prof. Bruno Defude, who accepted me as PhD student in the last year of the studies. I owe my deepest gratitude to Dr. Marius Preda for the continuous support of my PhD study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my PhD study.

Besides my advisors, I would like to thank the rest of my thesis committee: Prof. Dr. Catherine Pelachaud, Prof. Dr. Danco Davcev, Dr. Béatrice Pesquet-Popescu, Dr. Jae-Joon Han and Dr. Jérôme Royan, for their encouragement, insightful comments, and hard questions.

It is a pleasure to thank those who made this thesis possible: Dr.Son Min Tran, with whom we passed hours and hours unstoppable work and Dr. Francisco Morán Burgos for his patient and help, for his work together on projects and particularly for his detailed review on my papers published during studying. It is an honor for me to thanks Dr.Nunzio Santoro, Dr.Frank Gillet, Dr. Khaled Mamou and Dr. David Oyarzun, with whom I spent lot of time working together on a different projects and topics of interested for my study. Particularly, I would like to show my gratitude to my colleague and my boyfriend, Dr. Ivica Arsov, who has made wide available his support in the professional field, as well as in the private life. Honestly, I thank him that he managed to put up with me during the most difficult periods and to encourage me, and for all the emotional support, camaraderie, entertainment, and caring he provided.

I thank my fellow lab mates in the ARTEMIS Department, for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last years. Also I thank my colleagues and professors from SS "Cyril and Methodius" University from Skopje, Macedonia: Prof. Dr. Georgi Stojanov, Prof. Dr Vladimir Trajkovic and Mr. Sasko Celakovski. In particular, I am grateful to Dr. Slobodan Kalajdziski for enlightening me the very first glance of research.

I wish to thank to my brother, Jordan Jovanov, who supports me not only the period of the thesis, but my entire life. The greatest thanks to my parents, Biljana i Nikolcho Jovanovi, who always know how to find a way to encourage me and supporting me spiritually throughout my life, not important when and from where. Last, but not least, I offer my regards and thanks to all of those who supported me in any respect during the completion of the project.

Contents

List of Figures

List of Tables

Abbreviations

1G	First Generation
2D	Two Dimensional
2G	Second Generation
GSM	Global System for Mobile Communications (originally: Groupe Spécial Mobile)
EDGE	Enhanced Data rates for GSM Evolution
3D	Three Dimensional
3G	Third Generation
UMTS	Universal Mobile Telecommunications System
HSPA	High Speed Packet Access
3DMC	Three Dimensional Mesh Compression
ADML	Avatar Definition Markup Language
BBA	Bone Based Animation
BIFS	BInary Format for Scenes
BML	Behaviour Markup Language
CI	Coordinate Interpolator
CML	Character Markup Language
COLLADA	COLLAborative Design Activity
DCT	Discrete Cosine Transform
EML	Emotion Markup Language
FAMC	Frame-based Animated Mesh Compression Stream
FBA	Face and Body Animation
FK	Forward Kinematics
GZIP	GNU zip
HumanML	Human Markup Language
H-Anim	Humanoid Animation
IK	Inverse Kinematics
ISO	International Organization for Standardization
JPEG	Joint Photographic Experts Group
JPEG2000	Joint Photographic Experts Group 2000
LOD	Levels of Detail
MPEG	Moving Picture Experts Group
MPEG-1	Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s ISO/IEC 11172
MPEG-2	Generic coding of moving pictures and associated audio information ISO/IEC 13818
MPEG-4	Coding of audio-visual objects ISO/IEC 14496
MPEG-V	Media context and control ISO/IEC FCD 23005
MPML	Multimodal Presentation Markup Language
NASA	National Aeronautics and Space Administration
NURBS	Non-Uniform Rational Basis Spline
OI	Orientation Interpolator
OPENGL ES	OpenGL for Embedded Systems
PCA	Principal Component Analysis

PI	Position Interpolator
TGA	Truevision Graphics Adapter
QEM	Quadric Metric Error
RAR	Roshal Archive
RDF	Resource Description Framework
SC3GMC	Scalable Complexity 3D Mesh Coding
SMIL	Synchronized Multimedia Integration Language
VW	Virtual World
VHML	Virtual Human Markup Language
VRML	Virtual Reality Modeling Language
Wi-Fi	Wireless-Fidelity
X3D	Extensible 3D
XML	Extensible Markup Language
XMT	Extensible MPEG-4 Textual
W3C	World Wide Web Consortium
WSS	Wavelet Subdivision Surface Streams

Context and Objectives

In the last few years 3D Virtual Worlds (3DVWs) became a reality. Initially considered as a new mean for social communication, triggered by the development of software and hardware technology, 3DVWs are exposing now different functionalities, experiences and acquaintances. Therefore, they achieved their popularity very fast, indicated by the number and the progression of active users.

One important issue in order to make 3D VWs available and widely accessible is the development of the technology for representing and visualizing 3D content. A lot of effort is made to address this issue in both research literature and commercial implementations. However, much still remains to be investigated and implemented in order to make 3DVWs lively places. Another open subject is how to enable or facilitate access to them. Being a genre of online community, it is obvious that transfer through network is required; therefore the need of compression is acknowledged. There are two closely interlinked reasons for compressing 3D data when used in on-line and real-time applications such as VWs: storage and transmission bandwidth. A relatively new trend in the usage of content is raised by the fast development on mobile devices. Nowadays they are considered not only as communication devices as they were committed to, but also as effective platforms for more complex applications, including multimedia ones. The mobile phone becomes the perfect tool that links the users to the digital world, due to the perfect link to both of them: easy to carry physically and always connected. As part of multimedia applications, the ones related to 3D graphics are yet in the pioneering stage despite the recent progress in mobile hardware solutions. Thus, it is required to enable access to VWs by using heterogeneous terminals and this leads to developments in two signal processing domains: compression and adaptation.

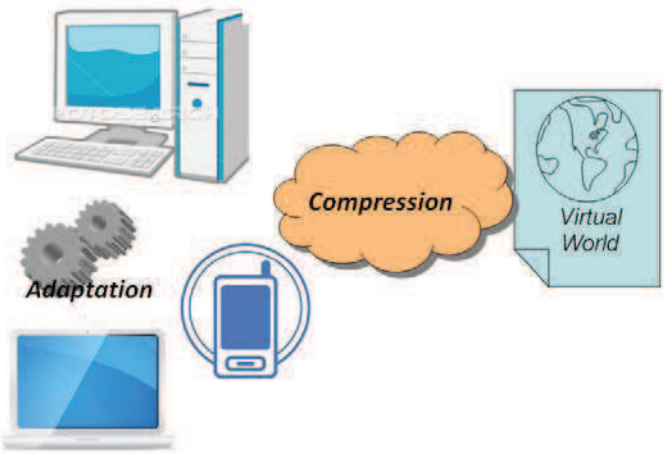


Figure 1. Compression and Adaptation, two key techniques for large deployment of Virtual Worlds.

Figure 1 illustrates two of the main issues to be addressed in the process of making VWs widely accessible: compression and adaptation of the data.

Another fact is that various VWs exist and their number is growing. The impossibility to exchange assets between VWs is negatively impacting the development and usefulness of VWs, since recreating it in the already enormous set of VWs is a very time-consuming task and can be frustrating. On the contrary, the possibility to transfer assets from one VW to another or at least meaningful information ensuring consistency between different representations can enormously extend the deployment of 3D VWs by offering a

background for interoperability. Thus, the possibility to exchange assets between different VWs is considered necessary.

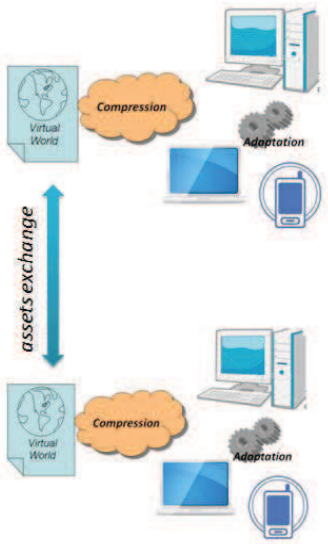


Figure 2. Framework for accessing VWs and exchanging their assets.

Figure 2 presents an enriched scenario where in addition to compression and adaptation of the assets, the possibility of exchanging them between VWs is possible.

Within a Virtual World, a significant place is held by the representation of the user. Therefore, an avatar is usually the representation of the human user. If in early generations of VWs and in games the avatars are used to camouflage a real identity, today's VW trend is to consider the avatar to be the real person representation, capturing personal information used to distinguish a person from another.

Being the representation of the user, the avatar is one of the most significant and most complex assets of a Virtual World. A short analysis of a VW content allows one to observe that from the point of view of the storage/transmission the most significant amount is represented by the VW assets. That is, other information such as the scripts or the executable code stands for less than 10% of the entire data. Within the set of assets, the avatars are the most complex structures, consisting of different components: geometry, images, animations, structures, etc. Therefore, by addressing compression, adaptation and interoperability issues related to avatars, we implicitly cover almost all kinds of assets that can be found in the VW[1].

The overall objective of developing tools and methods for a large deployment of VW are translated into three specific ones:

- To propose a compression framework to enable efficient, compact transfer of avatars, and general 3D graphics assets. Specifically for avatars, the framework should be independent from the representation formalism.
- To propose an optimized solution making the avatars accessible on weak terminals such as mobile phones.
- To define a metadata model allowing avatars interoperability between different VWs.

The three objectives are addressed in this manuscript and for each we propose original contributions, briefly described as follows:

- propose a data model allowing to apply compression technologies independently from the representation formalism. The MPEG-4 compression methods are

[1] An example of an asset that cannot be addressed by the proposed avatar framework is the terrain.

currently representing the state-of-the-art for compressing 3D graphics content. However, they are restricted to MPEG-4 compliant scenes, therefore their usage is limited. The proposed solution liberates the power of these methods to be used for compression on any scene-graph formalism. Particularly, we are interested in avatar representation, hence a complete solution is presented allowing MPEG-4 compression to be used over any XML file that contains an avatar definition. This data model was proposed to the MPEG committee for standardization and it is currently included in the "MPEG-4 – Part 25" standard.

- provide a technical solution for an optimized representation of 3D avatars on weak terminals. The optimality is obtained by analyzing the capabilities of the modern mobile phones and the processing needs when representing and animating a 3D avatar. The proposed solution allows automatic adaptation of the content to the local configuration. Adaptation consists mainly of the appropriate simplification of the avatar attributes, together with a compression layer in order to fit the constraints of the device.
- define a framework that provides means for the migration of avatars between different virtual worlds. The proposed solution relies on the definition of avatar in existent VWs and different avatar representation standards, therefore aiming to cover all attributes supported by those formalisms. As a result of these observations, we propose a full metadata layer associated to avatars and expressed as an XML description. This layer was proposed to the MPEG committee for standardization and it is currently included in the "MPEG-V - Information Exchange for Virtual Worlds" standard.

The structure of the manuscript is as follows. Firstly, an introduction that argues on the need for compression, adaptation and interoperability of assets is presented. Chapter 2 provides the state of the art with respect to the tools and methods addressed in this manuscript. It describes in detail the definition of an avatar, its components, as well as the methods for its creation and representation. We also introduce several representation standards and provide a description of MPEG-4 specifications for avatars. The chapter is completed by an overview of current approaches for compression, adaptation and interoperability. Chapter 3 contains the main contribution of the thesis and introduces three original solutions for each of the previously addressed problems. Firstly, we propose a data model for 3D graphics assets that enables the fast transport of the content through the network. Secondly, we propose a complete solution for the optimal processing of avatars on weak terminals. Thirdly, we propose an avatar metadata model that can be used in an open set of VWs, independently with respect to the representation formalism of the 3D avatar. Chapter 4 presents the experimental validation of the theoretical contributions proposed in Chapter 3 and discusses the innovations that they introduce. Firstly, we perform compression experiments based on the proposed data model and show the performances for generic 3D graphics objects with various characteristics and avatars. Then, we present the results of the adaptation of avatars for mobile phones in two different experiments. Finally we present the benefits of the interoperability framework and an example of "teleporting" avatars between different virtual worlds. Conclusions and perspectives of future work are reported in the last chapter.

1 Resumé long

Au cours des dernières années les Mondes Virtuels 3D (MV3D) sont devenus une réalité. Initialement considérés comme un nouveau moyen de communication sociale, initiés par le développement logiciel et matériel, les MV3Ds révèlent diverses fonctionnalités, des expériences et des connaissances. Par conséquence, leur popularité s'est étendue et le nombre d'utilisateurs actifs est en pleine progression. Une problématique majeure pour rendre un MV3D disponible et largement accessible, est le développement d'outils technologiques pour représenter et visualiser le contenu 3D. D'énormes efforts ont été investis pour surmonter ces problèmes dans le domaine de la recherche scientifique et également en industrie. Cependant, beaucoup de verrous restent à résoudre, des approches devraient être explorées et implémenter de nouvelles techniques afin de réaliser des MV3D réalistes et interactifs.

Une autre question substantielle et ouverte est comment faciliter l'accès à ces Mondes Virtuels MVs. Les utilisateurs sont connectés en ligne et la communication via le réseau est requise; donc la compression est d'autant plus indispensable. Il y a deux raisons étroitement liées pour la compression de données 3D des applications en ligne et temps réel telles que les MVs qui sont: le stockage et la bande passante. Une tendance relativement nouvelle dans l'utilisation du contenu est issue du développement technologique rapide les dispositifs mobiles. De nos jours, ils ne sont pas considérés comme des dispositifs de communication uniquement, mais aussi ils représentent des plates-formes d'implémentation d'applications plus complexes, y compris celles du multimédia. Le téléphone portable est devenu l'outil parfait reliant les utilisateurs avec le monde numérique, en raison de la dépendance de l'utilisateur envers son appareil et les avantages que présente un tel dispositif : facile à porter et toujours connecté. Faisant partie du multimédia, les applications graphiques 3D sont encore à l'étape de l'innovation malgré les progrès récents des dispositifs mobiles. Ainsi, l'accès au MV en utilisant des terminaux hétérogènes est exigé et conduit à la recherche et au développement dans deux domaines de traitement de signal à savoir, la compression et l'adaptation.

Un autre point à soulever est le fait que divers MVs existent et leur nombre est en plein croissance. L'impossibilité d'échanger des ressources (asset) entre les MVs a un impact négatif dans le développement et l'utilité des MVs, puisque recréer un tel monde est une tâche très exigeante en termes de temps et peut être irritante. Au contraire, la possibilité de transférer des ressources d'un MV à un autre ou même des informations significatives assurant la cohérence entre différentes représentations, peut étendre largement le déploiement des MV3Ds en offrant un contexte à l'interopérabilité. Ainsi, la possibilité d'échanger des ressources entre des MVs différents est considérée comme étant une tâche nécessaire.

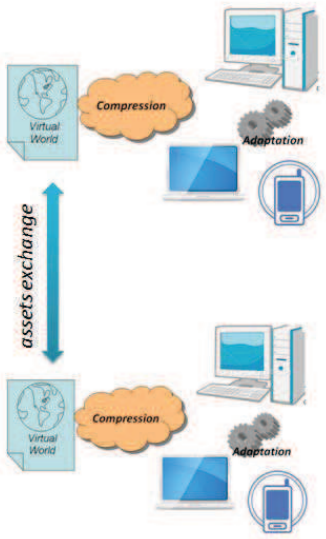


Figure 3. Framework d'accès et échange de ressources dans les MV3Ds.

Figure 1 présente un scénario complet où la compression, l'adaptation et la possibilité d'échanger des ressources entre les MV3Ds sont possibles.

Dans un monde virtuel, l'endroit le plus significatif est celui où l'utilisateur doit être représenté. Donc, un avatar est usuellement considéré comme étant la représentation de l'utilisateur humain. Si dans les premières générations des MVs et des jeux, les avatars étaient utilisés pour camoufler la vraie identité de la personne, la tendance actuelle des MVs étant de considérer l'avatar comme la représentation réelle de l'utilisateur en ajoutant ses informations personnelles afin de distinguer une personne d'une autre.

Étant la représentation de l'utilisateur, l'avatar est l'une des ressources les plus significatives et les plus complexes d'un monde virtuel. Une courte analyse d'un contenu de MV nous informe que le stockage/transmission est la partie la plus significative des ressources du MV. C'est-à-dire d'autres informations telles que les scripts ou le code exécutable représentent moins de 10 % des données entières. Dans l'ensemble des ressources, les avatars sont les structures les plus complexes, représentés par différents composants : géométrie, images, animations, structures, etc. Donc, en abordant les problématiques de compression, d'adaptation et d'interopérabilité liées aux avatars, nous traitons implicitement presque tout type de ressources tifs qu'on pourrait retrouver dans les MVs[2].

L'objectif global du développement d'outils et des méthodes, pour un déploiement élargi de MV, est traduit par les trois points spécifiques suivants:

- Proposer un framework de compression pour permettre le transfert efficace et compact d'avatars et de ressources graphiques 3D généraux. Spécifiquement pour des avatars, le framework devrait être indépendante avec un formalisme de représentation.
- Proposer une solution optimisée permettant l'accessibilité aux avatars sur des terminaux à faibles ressources tels que les téléphones portables.
- Définir un modèle de métadonnées permettant l'interopérabilité d'avatars entre différents MVs.

[2] Un exemple de ressource qui ne peut pas être traitée par le framework d'avatar proposée, est le terrain.

Les trois objectifs sont traités dans ce manuscrit et pour chacun nous proposons des contributions originales. Ainsi, la première partie de cette thèse présente un état de l'art en rapport avec les outils et les méthodes citées dans ce manuscrit. On décrira en détail la définition d'un avatar, ses composants, aussi bien que les méthodes pour leur création et représentation. Nous présentons aussi plusieurs standards de représentation et fournissons une description de spécifications MPEG-4 pour les avatars. Cette partie est complétée par une vue d'ensemble d'approches récentes de compression, d'adaptation et d'interopérabilité. La partie suivante décrit la contribution principale de la thèse et présente trois solutions originales pour les problèmes abordés précédemment. Premièrement, nous proposons un modèle de données pour les ressources graphiques 3D permettant le transfert rapide du contenu dans le réseau. Deuxièmement, nous proposons une solution complète pour le traitement optimal d'avatars sur des terminaux portables. Troisièmement, nous proposons un modèle de métadonnées d'avatars qui peut être utilisé dans des jeux en MV indépendamment et en respectant le formalisme de représentation de l'avatar 3D. La dernière partie présente la validation expérimentale des contributions théoriques proposées auparavant et discute les innovations introduites par les nouvelles approches. Premièrement, nous effectuons des expériences de compression basées sur le modèle de données proposé et nous montrons les performances sur des objets graphiques 3D génériques avec diverses caractéristiques et avatars. Ensuite, nous présentons les résultats d'adaptation d'avatars sur des téléphones portables dans deux expériences différentes. Finalement, nous présentons les avantages de la structure d'interopérabilité et un exemple de « téléportation » d'avatars entre différents mondes virtuels. Les conclusions et les perspectives sont présentées à la fin du rapport.

1.1 Modélisation d'avatar, compression, adaptation et représentations standards

1.1.1 Introduction

La structure complexe et les performances de simulation en temps réel des avatars constituent un véritable défi des environnements virtuels. L'avatar est une réplique de l'utilisateur et en relation directe avec lui; donc il devrait satisfaire plusieurs exigences en termes d'expérience-utilisateur. La création d'une représentation efficace et réaliste d'une part et son adaptation optimale avec les ressources d'autre part, requière la compréhension profonde des caractéristiques de l'avatar.

Donc, nous introduirons plusieurs notions liées à la représentation d'avatars aussi bien que des techniques pour les créer/capturer. Ensuite, deux domaines de traitement de signal - la compression et l'adaptation – sont examinés en tenant compte des spécificités concernant la représentation d'avatars. Alors, comme une base pour définir la structure de métadonnées, nous analysons différents mondes virtuels et des standards de représentation.

1.1.2 Représentation d'avatar

Étant un objet graphique 3D, l'avatar est représenté par le quadruple : le graphe d'objet, la géométrie, l'apparence et l'animation. De telles informations permettent la représentation visuelle de l'avatar, mais ne fournissent pas des données pertinentes concernant ses conditions d'utilisation. Donc, nous complétons le quadruple classique par un nouveau membre, appelé : les métadonnées d'avatar.
Dans ce manuscrit, nous considérons l'avatar défini par :

$$Avatar = \{ObjectGraph, Geometry, Appearance, Animation, Metadata\}$$

Dans ce qui suit nous analysons ces composants en tenant compte des approches de représentations et les techniques et les outils pour les obtenir.

1.1.2.1 Graphe d'objet

Le terme graphe définit une structure de données créée par un ensemble d'entités hiérarchiquement organisées appelé nœuds. Certains objets composant la scène sont assez simples à représenter tels que la lumière qui nécessite un nœud seulement; cependant, les objets en général, sont représentés par un ensemble hétérogène de nœuds. Les avatars sont des objets multi-parties, appartenant au deuxième groupe. Le graphe de l'avatar est composé de nœuds décrivant la relation logique et spatiale de divers composants.

Le dernier peut être séparé en deux groupes : basique et technique. Le premier groupe inclut les nœuds utilisés pour la géométrie, l'apparence et la description d'animation. Le deuxième groupe inclut les nœuds de la structure squelettique, les cibles de morphage, les paramètres physiques.

En représentant le conteneur d'éléments qui composent l'avatar, la création du graphe d'avatar reste une tâche manuelle, effectuée en utilisant des outils de création de contenu. Le processus est automatisé à l'intérieur de l'outil, en utilisant principalement des modèles (gabarits), cependant des réglages manuels sont nécessaires pour les personnaliser, par exemple, l'importation d'objets (l'avatar ou l'une de ses parties) à partir des ressources externes ou bien le contrôle au niveau des primitives de base (sommets, arêtes, objet, bipède).

1.1.2.2 Géométrie

Les procédés de représentation d'objets solides sont souvent divisés en deux catégories : des représentations de partitionnement de l'espace et des représentations de frontière (B-reps), bien que non pas toutes les représentations sont dans l'une ou l'autre catégorie [Requicha80].

Parmi toutes les B-reps, les maillages polygonaux sont les méthodes les plus utilisées où le processus de création implique la manipulation directe des polygones, des sommets et des arêtes pour produire la forme désirée. L'avantage principal de cette représentation est la simplicité, ce qui la rend optimale pour une visualisation temps réel. L'inconvénient principal est le besoin d'un grand nombre de sommets pour modéliser des formes réalistes. Parmi les représentations de maillages polygonaux existantes, un maillage de sommets de visage est la représentation la plus utilisée à cause de sa faible complexité et ses exigences de stockage en mémoire. Cette représentation est acceptée par les microcontrôleurs graphiques modernes et la plus efficace en termes de rendu.

Plusieurs outils de création sont dotés de mécanismes pour générer la géométrie permettant de modéliser le personnage virtuel [3DSMax, Maya]. L'inconvénient principal de cette méthode est que le résultat est fortement lié aux compétences artistiques et à l'expérience du concepteur. De plus, cette procédure est exigeante en termes de temps. Une approche inspirée par le fait que les gens peuvent plus facilement s'auto-exprimer sur un papier en réalisant des esquisses propres à leurs imaginations, ensuite en les manipulant avec l'outil de création 3D. Plusieurs outils d'esquisse connus pour la création 3D incluent : le système de Teddy [Igarashi99], FiberMesh [Nealen07], Plushie [Mori07]. D'autre part, des méthodes plus rapides essayent d'en créer une représentation électronique d'un objet (avatar) à l'aide d'un objet réel et/ou par la collection des connaissances ou des propriétés. Les techniques du deuxième groupe capturent des objets réels par des scanners 3D, comme l'exemple décrit dans [Fudono08], la capture à base de vision [Devernay94, Of Apuzzo99, Nebel02, Wilczkowiak05, Hengel07] et la modélisation à base de mesure [Seo03, Karpenko06].

1.1.2.3 . Apparence

Un des mécanismes de base pour améliorer l'apparence d'avatar [Maïm08] est de définir des attributs comme les couleurs pour chaque polygone ou sommet de l'avatar. Le processus pour attacher l'image au maillage est appelé la cartographie de texture, qui est l'une des techniques les plus abouties pour enrichir la qualité visuelle. Même si la texture nécessite une mémoire supplémentaire, ceci est insignifiant comparé à la quantité de mémoire exigée par les sommets représentant les mêmes détails visuels. Le plus courant est l'utilisation des textures pour attacher des attributs de couleurs par sommet (la carte diffuse), d'autres attributs peuvent être mis à jour en utilisant la même approche : la surface normale (la carte normale), la spécularité (la carte spéculaire), la transparence, l'illumination et le déplacement de la surface.

Le moyen le plus fréquent pour créer une texture est la capture de la scène par une caméra et la pré-déformer pour l'appliquer sur la forme 3D, la déformation est compensée par le processus du rendu. D'autre part, les systèmes d'infographie sont capables de produire des images réalistes d'objets. Les techniques les plus connues et fréquemment utilisées pour à cet effet, incluent les méthodes : fractale, spectrale, à base de syntaxe, structurelle ou stochastique.

1.1.2.4 Animation

L'animation d'avatar définit les capacités de mouvement de l'avatar et les interactions possibles avec l'environnement.

L'animation d'un personnage consiste à appliquer des déformations au niveau de l'habillage. La plupart des approches de déformation de maillage 3D sont classées dans cinq catégories basées sur: treillis [Maestri99, Liu08, Jiacheng09], groupe [Maestri99, Boulfani08], Spline [Bartels87, Mizuno06, Hongzheng07], morphage [Blanz99, Alexa00] et squelette [Lander99].

Les quatre premières catégories sont utilisées dans l'animation d'objets spécifiques comme des yeux (treillis) et les expressions faciales (morphage) et sont plus ou moins supportées par les principaux progiciels d'animation. Cependant, tous les quatre ont le même inconvénient : ils ne prennent pas en compte le caractère naturel des caractéristiques des formes. Ainsi, la dernière catégorie est actuellement la technique la plus utilisée pour l'animation de personnage virtuelle.

L'animation squelettique est un concept qui était souvent utilisé dans les secteurs de : cinéma, jeux électroniques et applications semblables afin de créer un mouvement réaliste pour l'animation des personnages articulés. Cette technique pour animer un modèle 3D consiste à créer une structure hiérarchique, nommée le squelette dont le mouvement rigide conduit à la déformation du modèle associé. L'emplacement et le déplacement des articulations du squelette indiquent comment l'avatar tout entier se déplace. Des méthodes intéressantes pour créer le squelette humain ont été publiées dans [Magnenat91, Gourret89, Monheit9, Van98, Van99, Wanget07, Shi07, Han-Bing08, Lewis00, Kry02, Wang02, Mohr03].

La création d'une animation peut être classée dans trois catégories : cinématique, dynamique et capture de mouvement. Les deux premières catégories appartiennent aux techniques utilisant des outils de création de contenus, tandis que la troisième est une technique où un matériel supplémentaire est utilisé.

1.1.3 Compression

Ne visant pas à un tour d'horizon complet sur la compression de maillage statique/animée (plusieurs d'entre eux existant déjà dans [Avilez08, Peng05]), nous présentons seulement les techniques les plus importantes en indiquant des références pertinentes pour mieux situer le modèle architectural générique proposé. Nous analysons deux groupes principaux de techniques : la compression des maillages 3D et la compression d'animation.

Pour la compression de maillages graphiques 3D, nous avons remarqué que deux domaines ont retenu l'attention des chercheurs : la compression de maillages statiques et la compression de maillages animées.

Il y a deux approches principales de compression des maillages 3D statiques. La première basée sur la compression avec un seul taux, les données sont compressées et décompressé en une fois. La deuxième méthode est concentrée sur la compression progressive et la transmission lorsqu'un maillage 3D peut être décodé et rendu continuellement avec des niveaux de détail différents (LODs). Les concepts mathématiques, les définitions et la théorie s'intéressant à la compression de maillage sont très bien présentés dans la littérature [Edelsbrunner01, Kahn95, Gross98].

Plusieurs méthodes pour la compression de maillages animés existent, exploitant la corrélation spatiale [Lengyel99, Amjoun06, Preda04] et temporelle [Zhang04, Zhang 05, Yang 02, Ibarria03]. Une méthode très connue consiste dans l'Analyse de Composante Principale [Alexa00].

Pour la compression de paramètres d'animation, l'analyse a été faite avec des méthodes populaires. L'élaboration de modèles, qui permettent de représenter des données d'animation d'une façon compacte, peuvent être réalisés en exploitant la redondance d'animation en utilisant l'interpolation: (i) temporelle et (ii) spatiale. De plus, pour élaborer les modèles d'animation, pour réduire la quantité de données de transmission, on peut utiliser la compression de l'animation. Le groupe MPEG a normalisé une approche pour la compression générique des données interpolées [Jang04]. Concernant l'animation d'avatar, MPEG a standardisé en 1998, un outil nommé « Face and Body Animation » (FBA) et en 2004 une amélioration, appelée BBA (Bone-based Animation). BBA permet la représentation de n'importe quelle sorte de modèle articulé, et offre une animation de grande qualité en utilisant un modèle de déformation biomécanique (la simulation d'un squelette) et exploite la redondance temporelle et spatiale du signal d'animation.

A part les solutions MPEG, d'autres méthodes ont été présentées dans la littérature : [Rossignac03, Guskov04, Sloan03].

1.1.4 Adaptation du contenu pour les terminaux à faibles ressources

Malgré le progrès significatif dans le développement matériel, le rendu et l'animation des objets 3D complexes et spécifiquement les humains virtuels, des questions sont toujours soulevées concernant les téléphones portables. Des environnements collaboratifs virtuels en réseau ont été étudiés depuis deux ou trois décennies et sont maintenant largement présents et disponibles sur plusieurs plates-formes, y compris les plateformes mobiles. Cependant, la représentation d'avatars pour de tels dispositifs est relativement simpliste et des techniques pour l'adaptation du contenu ont été proposées dans la littérature.

1.1.4.1 Techniques d'adaptation du contenu

Parmi les différentes techniques d'adaptation de contenu c'est la réduction de la complexité de la chaîne graphique. Par conséquent, en théorie les solutions les plus appropriées sont celles convertissant les données 3D en d'autres types de média, comme des images, des vidéos ou des graphiques 2D. Ce concept, aussi connu comme « transmoding », peut bénéficier des méthodes d'infographie existantes comme le rendu à base d'Image [Chang02]. Cette technique a été aussi explorée dans le rendu de personnages virtuels 3D [Aubel00]. Le transmoding est usuellement exécuté hors-ligne, seulement un nombre réduit de vues étant synthétisées. Une tendance récente relative doit exécuter le transmoding en temps réel, sur des serveurs dédiés et transmettre les résultats dans un flux vidéo [Lamberti03].

Une autre technique citée en littérature adapte la résolution du contenu graphique 3D des caractéristiques de dispositifs légers. Plusieurs formats de fichier comme : pod, md2, m3g, mp4 sont spécifiquement conçus pour livrer le contenu graphique 3D.

Cependant, les deux techniques présentent des inconvénients qui les rendent inappropriées pour beaucoup d'applications. Dans la première la qualité de médias est significativement amoindrie et en outre lorsqu'elle est utilisée en mode en ligne la bande passante est fortement augmentée. La deuxième technique consiste à préparer divers contenus 3D pour différentes plateformes de clients. Aucune des techniques présentées dans le deuxième groupe ne supporte l'automatisation du processus d'adaptation du contenu, ce qui rend le processus d'adaptation de contenu difficile. Pour être générique par rapport aux conditions matérielles, une solution très attractive est l'adaptation automatique du contenu selon le contexte. Plusieurs approches fournissent des procédés génériques pour des modèles 3D évolutifs. Dans la section suivante, nous présentons un état de l'art de ces approches.

1.1.4.2 Techniques d'adaptation d'avatars

Des maillages multi-résolution pour l'adaptation ont été proposés dans le MPEG-21 (Digital Item Adaptation framework – MPEG-21]. Outre la représentation évolutive, l'adaptation d'animation 3D et particulièrement l'animation d'avatar, ont été explorées aussi [Giacomo04].

Parmi les techniques de représentation actuelle de la géométrie d'avatar est le maillage. La simplification du maillage consiste principalement dans la réduction de sommets, plusieurs algorithmes ont été proposés dans la littérature : décimation de sommet, partitionnement de sommet et contraction itérative d'arêtes. Une des méthodes du dernier groupe est la simplification de surface basée sur l'erreur quadratique métrique (QEM) [Garland97].

Des avatars reliés, des critères spéciaux de simplification devraient être pris en compte. En raison de l'évolution du maillage dans le temps et l'augmentation du QEM, la simplification des sommets ne peut pas être seulement considérée dans une position statique. Dans [Preda05], une nouvelle méthode de simplification de maillage qui est l'adaptation de QEM au modèle animé avec l'aide du squelette a été proposée.

Outre la géométrie et l'animation, les apparences et principalement les textures d'image, sont des composants d'une taille relativement importante. La simplification d'image par la réduction de sa résolution présente l'avantage de réduire la taille de transmission et aussi la mémoire du temps d'exécution.

1.1.5 Interopérabilité

En dehors de la communauté de recherche, les avatars ont intéressé différents groupes de standardisation, principalement en raison du grand nombre d'applications potentielles impliquées[3]. Il y a actuellement deux types de standards : celui qui s'intéresse à l'apparence et l'animation de l'avatar dans les applications graphiques 3D (les avatars sont des objets de représentation) et celui qui s'intéresse aux caractéristiques des avatars comme la personnalité, les émotions, … (les avatars comme des agents).

Dans le premier groupe, X3D [X3D] (basé sur VRML [VRML]) et COLLADA [COLLADA] sont les plus connus, le dernier étant le plus adopté par les outils de développement actuels. Tandis que COLLADA se concentre sur la représentation 3D d'objets ou des scènes, X3D pousse plus loin la standardisation en abordant l'interaction utilisateur et le comportement d'application aussi. Les avatars dans VRML/X3D sont définis comme étant des objets spécifiques standardisés sous le nom H-Anim, dans COLLADA il n'y a aucune distinction entre un avatar humain et un modèle d'habillage générique. Pour le traitement de la compression d'objets le standard MPEG-4 est utilisé. En offrant un ensemble complet permettant l'affichage des avatars, aucun des standards cités auparavant n'inclut les données sémantiques liées à l'avatar. Plusieurs recommandations, standards

[3] Gartner a annoncé que 80% d'utilisateurs actifs d'Internet auront une "Seconde Vie" dans un Monde Virtuel à la fin de 2011, http://www.gartner.com/it/page.jsp?id=503861.

ou langages de balisage sont reliés à la sémantique des personnages virtuels, principalement pour décrire les caractéristiques qui n'ont pas nécessairement de représentation visuelle (comme la personnalité ou les émotions) ou bien exposer les propriétés qui peuvent être utilisées par un agent (des compétences linguistiques, la modalité de communication, …). Cela faisant partie de Human Markup Language (HumanML) (Brooks, 2002), EmotionML (EML) réalisé par W3C, Behaviour Markup Language (BML) (Vilhjalmsson, 2007, Multimodal Presentation Markup Language (MPML) (Ishizuka, 2000) ,Virtual Human Markup Language (VHML) , Character Mark-up Language (CML) (Arafa03). En outre, il y a plusieurs formats propriétaires imposés comme standards par les outils de création ou les créateurs de MVs.

Cette multitude de solutions rend impossible aujourd'hui d'imaginer un simple scénario d'utilisation d'un seul avatar pour visiter deux mondes virtuels différents. Aucun standard ni logiciel tiers dans les deux groupes ne fournit un ensemble complet des caractéristiques permettant l'interopérabilité des avatars entre des MVs. Ceci nous a motivé à proposer un schéma pour l'interopérabilité.

1.2 Contributions à la compression d'avatars, l'adaptation et l'interopérabilité

1.2.1 Introduction

Les avancées réalisées dans les deux dernières décennies dans le domaine des avatars concernant leur apparence visuelle (statique et animé) permet d'imaginer aujourd'hui une solution de représentation intégrée qui aspire à assurer l'une des exigences principales d'interopérabilité entre les mondes virtuels : la capacité de migrer d'un monde à un autre en maintenant les propriétés de l'utilisateur. Un autre progrès majeur de ces dernières années est l'exportation des technologies sur les téléphones portables. Étant à l'origine des dispositifs pour la communication vocale, les téléphones portables modernes sont devenus un outil utilitaire essentiel, capable de réaliser différentes fonctionnalités, parmi lesquelles celles qui donnent l'opportunité de représenter un avatar personnalisé. De plus, avec les progrès technologiques actuels dans les réseaux de communication mobiles, des services de données à haut débit basés sur le Protocole Internet (IP), sont disponibles. La plateforme 3G fournit une voix convergée, les données, l'accès à Internet et les services multimédia sont accessibles par des transmissions à haut débit, redéfinissant ainsi la manière de communication entre les personnes.

On s'y attend à ce que les futurs MV utiliseront les sous-ensembles de propriétés de l'utilisateur; probablement ils utiliseront tous des apparences et des propriétés d'animation pour assurer leurs représentations visuelles. Quelques propriétés/capacités obtenues dans un MV restent liées à l'avatar et devraient être disponibles aussi aux mondes extérieurs (réel ou virtuel).

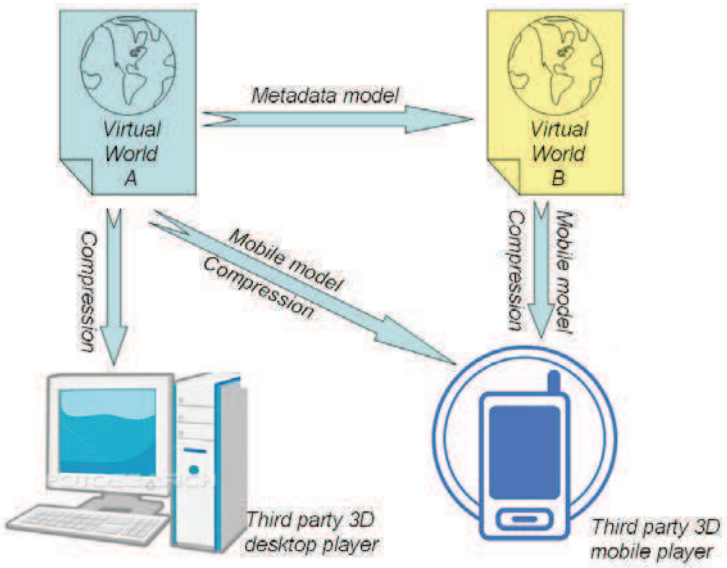
Plusieurs MVs permettent l'exportation de l'avatar en entier. Tandis que la possibilité d'importer cet avatar dans un autre MV est voulue, le modèle économique du MV est très souvent construit d'une façon à interdire l'importation du contenu puisqu'on le considère comme une valeureuse ressource [Castronova02, Castronova03] et le fournisseur du MV préfère avoir un accès complet du contenu. Un scénario intéressant reste la possibilité de visualiser l'avatar dans des lecteurs (players) externes. Particulièrement, si le lecteur est construit dans une autre plateforme et exige l'adaptation de la géométrie, l'apparence et l'animation [Francisco07].

En tenant compte de l'analyse précédente, nous proposons un processus automatique (workflow) qui permet la "téléportation" des caractéristiques d'avatars d'un MV à un autre et aussi "le transfert" d'un avatar complet dans des lecteurs externes et exécutés sur des terminaux hétérogènes.

Figure 4 représente la fonctionnalité d'un tel processus: l'avatar est créé dans un MV et il est représenté dans un autre, en transférant ses caractéristiques sémantiques, tandis qu'il est possible aussi de le représenter dans des lecteurs extérieurs, en transférant toutes ses caractéristiques.

Le framework est mis en œuvre par trois modules originaux :

- **Un modèle de compression de données** permettant la compression d'avatar arbitraire, défini en utilisant n'importe quel schéma XML.
- **Une méthode d'adaptation** d'avatars extraite du MV ou créée avec un outil de création (authoring tool). La méthode prend en considération les capacités réduites du téléphone portable.
- Une structure d'interopérabilité définissant l'ensemble des caractéristiques de l'avatar capables de fournir une description sémantique d'avatars requises pour personnaliser des modèles de MV.



1.

Figure 4. Un processus automatisé pour le transfert d'avatar entre les MVs et les plates-formes.

Dans les sections suivantes nous présentons chacun des trois modules.

1.2.2 Modèle de compression de données pour les structures hétérogènes

Les quatre premiers composants définissant un avatar sont très différents concernant la nature du signal et les outils de compression qui devraient être utilisés. Le standard MPEG-4 est déjà spécifié pour de telles techniques, cependant il peut être utilisé seulement si l'avatar est représenté en utilisant la structure de graphe d'objet MPEG. L'objectif principal du framework que nous proposons est de spécifier un modèle architectural, qui peut satisfaire une description XML tierce d'un graphe de scène avec des outils de binarisation et des outils de compression graphique 3D MPEG-4. L'utilisation d'outils de compression puissants pour le graphisme et la généralité de représentation des primitives graphiques sont les avantages d'une telle approche.

La structure proposée a trois couches : la Représentation de Données Textuelle, la Binarisation et la Compression comme représenté sur la Figure 5.

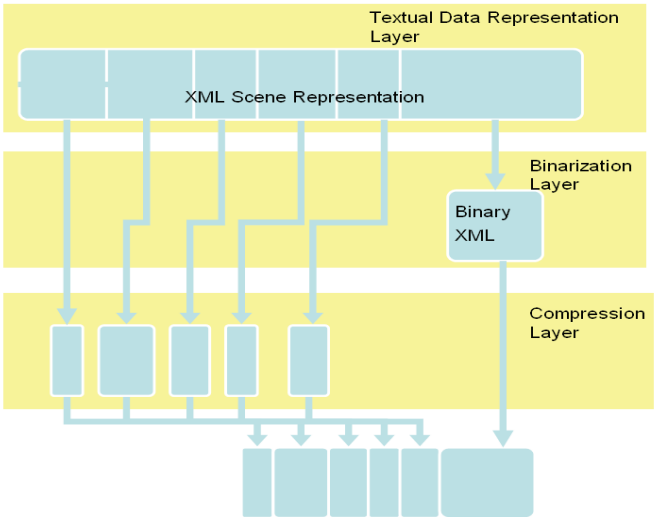


Figure 5. Les couches du Modèle de Compression Graphique 3D.

- **Couche 1 : Représentation de Données Textuelles**

Le modèle actuel peut s'adapter pour n'importe quel graphe de scène et tout formalisme de représentation de primitives graphiques. La seule exigence pour cette représentation est qu'il devrait être exprimé en XML. Tout schéma XML (spécifié par MPEG ou par d'autres organismes externes) peut être utilisé. Actuellement, les schémas XML suivants sont supportés : XMT, COLLADA et X3D, ce qui signifie que la correspondance entre un élément XML et l'outil de compression le manipulant est bien spécifiée.

- **Couche 2 : Binarisation**

La couche binarisation traite les données XML qui ne sont pas codées par les encodeurs bit-streams élémentaires (exemple : des éléments de graphe de scène). Ces données sont encapsulées dans l'atome "meta" du fichier MP4 et peuvent être converties en représentation textuelle ou binarisée en utilisant gzip. On note que gzip est nativement supporté par MPEG-4 pour l'encodage du « meta » atom.

- **Couche 3 : Compression**

Cette couche inclut les outils MPEG-4 suivants : Compression de Maillage 3D, Surface de Subdivision d'Ondelette, Interpolateur de Coordonnée, Interpolateur d'Orientation, Interpolateur de Position, Animation d'Arête et la Compression de Maillage Animée basée sur la Structure.

MPEG a reconnu l'utilité d'un tel modèle de données de compression et l'a accepté pour la standardisation dans une partie séparée de MPEG-4. L'ISO a publié le MPEG-4 partie 25 comme le standard international en mars 2009 et c'est disponible en ligne.

1.2.2.1 Extraction d'avatars à partir des structures de données hétérogènes

Tandis qu'elle est efficace en compression par rapport aux outils de compression MPEG, la structure proposée est limitée du fait qu'il est nécessaire de spécifier la relation qui lie les éléments XML et l'outil de compression. Une telle limitation peut être surmontée lorsqu'une connaissance a priori sur le type du contenu existe, comme en l'appliquant aux avatars.

À savoir, la définition d'avatars est basée sur un quadruple composé de graphe d'objet, la géométrie, l'apparence et l'animation. Une telle structure peut être détectée dans un contenu hétérogène (comme celui exprimé usuellement dans XML) basé sur l'interprétation de types de données. Nous avons proposé un exemple d'un tel algorithme pour identifier les composants d'avatar et il est présenté dans l'Annexe C. Une fois identifié, chaque composant peut être individuellement compressé selon le modèle de données proposé dans la section précédente.

1.2.3 Adaptation du contenu pour les terminaux à faibles ressources

Parmi les limitations matérielles des téléphones portables, comparés aux ordinateurs de bureau, est la puissance de processeur; et l'espace de stockage mémoire (RAM et disque dur), ceci a contraint la restriction de la quantité de données manipulées. D'autre part, les portables ont des écrans d'affichage relativement petits, donc la résolution spatiale peut être diminuée sans artefacts visibles. Cependant, d'autres caractéristiques comme la résolution temporelle (la fréquence d'animation) ne peuvent pas être réduites plus bas d'un certain seuil (généralement 20 FPS).

Un PC standard peut fournir des rendus d'objets environ 100 - 5000 plus rapidement qu'un dispositif mobile. Basé sur l'analyse de la configuration locale (la puissance du traitement, la bande passante par rapport à la résolution d'affichage) deux types de traitement de contenu sont nécessaires: la simplification et la compression. Dans les sections suivantes, nous présentons nos méthodes pour traiter les deux et présenter les améliorations par rapport aux techniques existantes.

Graphe d'avatar, géométrie, apparence et simplification d'animation

Si précédemment la tendance était "dispositifs multiples, contenu multiples" (MDMC), ces dernières années la tendance est "dispositifs multiples, un seul contenu" (MDSC). L'approche proposée pour l'adaptation d'avatar est une structure complète, prenant en considération les quatre composants le définissant : graphe d'objet, géométrie, apparence et animation. Dans les sections suivantes nous présentons l'adaptation pour chacun d'entre eux.

La complexité d'un graphe d'objet dépend du nombre de différents types de nœuds que le client devrait supporter. En prenant comme base le formalisme défini dans MPEG-4 BIFS, nous proposons un profil de contenu qui s'adapte au téléphone portable. Notre profil proposé, capable de représenter un objet statique et/ou animé (avatar) dans des scènes simples, est composé d'éléments suivants : Appearance, Color, Coordinate, Group, ImageTexture, IndexedFaceSet, Material, Normal, PointLight, Shape, TextureCoordinate, Transform, Viewpoint, SBVCAnimation (ou son extension SBVCAnimationV2), SBBone, SBSkinnedModel et SBSegment.

Pour le maillage et l'adaptation d'apparence, notre solution est basée sur l'algorithme décrit dans [Preda05]. Un ensemble d'expériences a été conduit pour valider son adéquation pour téléphone portable.

Tandis que la géométrie et les textures sont chargées et décodées seulement une fois, un composant plus sensible aux exigences limitées des téléphones portables est l'animation, puisque le décodage et la déformation de maillage devrait être exécutée sur chaque trame.

Une façon directe de réduire la quantité de données d'animation est d'identifier des trames clés pour que n'importe quelle trame intermédiaire puisse être interpolée à partir de l'ensemble des trames clés.

Étant donné une séquence originale d'animation composée de n trames, afin d'obtenir une séquence simplifié avec m trames (m < n) qui approxime le mieux la courbe originale, on doit minimiser la surface entre la courbe originale et la courbe reconstruite (par interpolation). Nous présentons les deux optimisations effectuées sur un encodeur MPEG-4 : prédiction dynamique et étendu de valeur dynamique.

Avec la prédiction de signal, la quantité d'informations requises à envoyer est diminuée, puisque la différence entre les valeurs exactes est plus petite que la valeur elle-même. Cependant, il est possible qu'après un certain nombre de trames (selon le type d'animation) les différences deviennent plus grandes que l'information. Pour de telles raisons une trame (I) (ou un segment) doit être forcé. De plus, en forçant la trame (I) aux moments spécifiques permet d'éditer facilement l'animation et l'accès aléatoire à la séquence d'animation. Dans notre encodeur optimisé, nous analysons chaque trame ce qui est moins couteux en termes de bande passante pour la transmettre et nous prenons une décision locale pour forcer une trame (I) (ou un segment) ou non. Dans nos expériences, nous avons exprimé ce paramètre par PbI (le nombre de P trames entre deux trames I).

Les tableaux relativement grands de l'encodeur arithmétique - la gamme de valeurs pour I trame sont [-1860 1860] et pour les P trames [-600 600] - exigent une mémoire importante pendant la phase de décodage. Dans notre version optimisée de l'encodeur, nous calculons dynamiquement l'intervalle par rapport au contenu. Cela permet de réduire la mémoire utilisée par des tableaux de probabilités de symbole. En outre, en définissant (min, max) apporte encore un avantage comparé à la méthode d'avant : ça permet une meilleure précision, puisque les tableaux sont adaptés aux données actuelles. Seulement l'inconvénient est que les valeurs (min, max) doivent être transmises. Cependant, ceci est acceptable le décodeur final est construit à partir de cette amélioration.

Pour assurer la qualité de l'encodeur optimisé BBA, mais aussi pour étudier l'influence des différents paramètres sur le décodeur, nous avons effectué des expériences sur la base de données disponible sur mocap.cs.cmu.edu qui contient environ 1700 fichiers de capture de mouvement de différentes nature et la complexité. Les résultats expérimentaux de cette optimisation sont présentés dans le chapitre suivant.

1.2.4 Structure d'interopérabilité

1.2.4.1 Analyse de modèles utilisés dans les mondes virtuels

Malgré le fait que plusieurs langages de balisage liés aux avatars et aux agents virtuels existent et qui assurent l'interopérabilité pour des avatars entre des mondes virtuels différents, cependant ils ne peuvent pas être encore obtenues d'une manière facile, prêt à utiliser et de façon intégrée. Dans la section suivante nous proposons une représentation de schéma pour des avatars permettant l'interopérabilité entre différents MV.

1.2.4.2 Modèle de métadonnées proposé

Notre proposition est de formaliser les métadonnées d'avatar en définissant le type "d'Avatar" composé d'éléments suivants : l'apparence, l'Animation et le Contrôle (un extrait du schéma XML exprimant le formalisme est fourni en Annexe E).

1.2.4.3 Élément "Apparence"

L'élément "Apparence" contient les descriptions des différents segments anatomiques de l'avatar (la taille, la forme, les paramètres anthropométriques) et aussi des références aux ressources externes liées à la texture et à la géométrie. Tandis que la première peut

être utilisée pour adapter la structure interne de l'avatar du MV – sa personnalisation, la deuxième peut être utilisée pour le remplacer complètement. La deuxième opération peut être exécutée seulement lorsque le format pour la ressource est lui aussi connu par l'importateur/exportateur, ce qui est le cas lorsqu'on utilise les graphiques 3D de MPEG-4. De plus, cet élément contient aussi les caractéristiques des objets qui sont liés à l'avatar comme les vêtements, des chaussures ou les accessoires.

1.2.4.4 Élément "Animation"

L'élément "Animation" contient un ensemble complet d'animations que l'avatar peut réaliser, groupé par similitude sémantique (Inactif, Salutation, Danse, Marche, Combat, Actions). Un groupe spécial contient des actions communes telles que : Boire, Manger, Parler, Lire et d'Asseoir. Comme dans le cas précédent, l'animation des paramètres géométriques sont représentés par des ressources externes, MPEG-V fournit seulement l'étiquette des séquences d'animation.

2.4.2.3 Élément "Contrôle"

Le but principal de l'élément "Contrôle" est de définir les placements sur le corps humain où les capteurs peuvent être déposés pour la capture de mouvement. Ces emplacements sont groupés sur le corps à des endroits spécifiques (les segments du squelette humain) et sur le visage (des emplacements 3D pour le visage de l'avatar). L'élément "Contrôle" assure la commande de l'avatar à partir des signaux externes aussi bien qu'une couche de base commune pour permettre le reciblage de mouvement entre des avatars avec une structure semblable ou légèrement différente.

MPEG a reconnu l'utilité du modèle proposé et l'a accepté pour publication comme partie du standard MPEG-V. Notre schéma a été adopté comme base de caractéristiques d'avatar et enrichi par d'autres membres de MPEG.

Le MPEG-V a été finalisé en novembre 2010 et le schéma que nous avons proposé couvre une grande partie de la « Partie 4 » de ce standard.

1.3 Expérimentations et validation

1.3.1 Introduction

Ce chapitre présente la validation des trois contributions décrites dans le chapitre précédent. Premièrement, nous décrirons les résultats de compression obtenus de la base de données. L'animation est complétée avec la visualisation d'un avatar obtenu par un lecteur externe. Deuxièmement, nous montrons les résultats quantitatifs et qualitatifs obtenus en réalisant plusieurs expériences afin d'adapter les avatars à un terminal à faibles ressources. Finalement, nous présentons une application qui permet "de transporter" des avatars d'un MV à un autre MV au moyen de leur apparence et des caractéristiques d'animation.

1.3.2 Validation du modèle de compression de données pour les structures hétérogènes

1.3.2.1 Architecture Logicielle

L'implémentation logicielle du framework présenté se compose de deux modules principaux : MP4XMLEncoder et MP4XMLDecoder. Le modèle a été implémenté pour les trois XML basé des formats de graphe de scène (COLLADA, X3D et XMT) et les flux élémentaires supportés sont 3DMC, BBA, IC, FAMC, JPG et JPEG2000.

En utilisant ce logiciel nous exécutons deux ensembles d'expériences. Dans le premier, nous analysons les performances de la compression du maillage statique et dans le deuxième les performances de l'animation.

Résultats de la compression de maillage statique

Nous réalisons les expériences avec b/c=9 (bit par composant), puisque c'est basé sur une inspection visuelle, nous avons remarqué que b/c=9 est le seuil pour presque tous les fichiers afin que la distorsion visuelle soit négligeable pour la base de données entière (plus de 1700 fichiers) et nous obtenons une compression moyenne de 2.86:1 avec un écart-type de 1.1, par rapport au XML original (dans ce cas c'est COLLADA) dont le contenu est compressé avec WinRAR. Ces taux augmentent à 6:1 si la conservation de l'ordre des sommets et des triangles n'est pas une contrainte.

Résultats de la compression d'animation

Deux ensembles d'expériences sont réalisés pour les données animées : les modèles articulés animés avec BBA et les maillages génériques déformés en utilisant FAMC.

BBA

De la même façon que 3DMC, BBA permet le contrôle de la qualité de compression en utilisant un pas de quantification. Pour identifier le seuil QS afin de réaliser des résultats équivalents visuellement, nous effectuons les expériences pour plusieurs fichiers. La distorsion est calculée comme l'erreur moyenne quadratique entre l'ensemble d'articulations dans les deux squelettes [Preda07]. On note que pour des animations contenant seulement les rotations (le cas de notre base de données) et utilisant un pas de quantification de 1024, permet d'assurer la compression sans artefacts visuels. Nous effectuons des expériences pour la base de données entière et nous obtenons une compression moyenne du contenu WinRAR'ed COLLADA de 14,6:1 et un écart type de 3,4. Quelques vidéos des résultats obtenus peuvent être visualisées en ligne[4].

FAMC

Le contrôle de la compression FAMC est effectué en sélectionnant le nombre de bits par sommet (bpv) et la distorsion est calculée par la distance de KG [Karni04], à chaque trame d'animation. Nous effectuons des expériences pour la base de données entière et nous obtenons une compression moyenne du contenu de 3,9:1 et un écart-type de 1,2 en utilisant WinRAR'ed COLLADA. On note que FAMC dispose de plusieurs configurations de codage. Dans nos expériences nous utilisons FAMC/DCT. Une comparaison complète entre différentes configurations est disponible dans [Mamou08].

Résultats de compression complets

Le but général du modèle de données proposé est de considérer l'objet 3D entièrement (géométrie, apparence, animation, éléments structurants) en exécutant une compression globale. Un encodeur conforme devrait être sans perte pour le graphe d'objet et presque sans perte pour la géométrie, la texture et l'animation. Nous réalisons une série de tests pour comparer la taille des fichiers de XML initiaux codés en WinRAR avec ceux codés avec les outils MPEG-4. Nous avons utilisé des pas de quantification qui ne produisent aucune distorsion visuelle.

En considérant l'objet entier (incluant les informations structurantes et le graphe de scène), le taux de compression MPEG-4 par rapport à RAR diminue jusqu'à 2:1, principalement parce que la quantité de géométrie pure ou des informations d'animation deviennent plus petites que le reste de l'information présente dans le fichier.

Nous avons effectué plusieurs expériences en utilisant l'implémentation logicielle décrite auparavant que nous avons appliqué sur un contenu représentant les avatars de complexités diverses. Globalement, pour un avatar compressé par le framework proposé

[4] www.MyMultimediaWorld.com

présente un avantage par rapport au RAR qui est de 2:1 en préservant la qualité visuelle. Cela correspond à un taux de 20:1 par rapport au contenu original (format textuel).

1.3.2.2 Performances de la méthode de compression d'animation optimisée

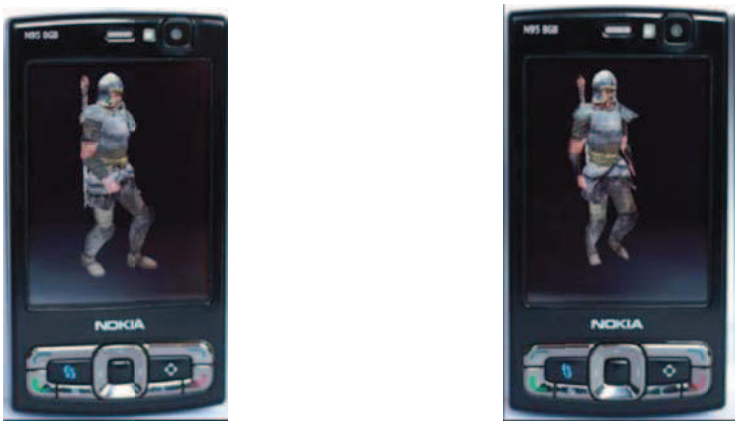
Pour améliorer les résultats de la compression d'animation, nous avons développé un encodeur d'animation à base de segments optimisé (BBA). Pour assurer la qualité, mais aussi étudier l'influence de différents paramètres sur le décodeur, nous avons effectué des expériences sur la base de données disponible sur mocap.cs.cmu.edu qui contient environ 1700 fichiers de capture de mouvement de différentes nature et complexité. Nous avons noté un facteur de compression de 70:1. De nos jours, il n'y a aucun consensus dans la communauté scientifique sur la mesure de la vraie distorsion à utiliser pour indiquer quantitativement le fonctionnement d'un algorithme de compression d'animation. C'est pourquoi, les expériences de compression de signal incluent, en général, des tests subjectifs. Nous avons conduit des tests semblables sur nos données compressées. Tous nos résultats sont disponibles sur www. MyMultimediaWorld.com pour une visualisation en ligne. Ce site Web utilisé généralement comme un dépôt de données 3D, a aussi la fonctionnalité de collecter les commentaires des utilisateurs.

1.3.3 Validation des outils d'adaptation pour un terminal à faibles ressources

Une première validation du framework était réalisée en effectuant plusieurs expériences sur un Nokia N95. Les résultats sont montrés dans le tableau 1 et la qualité visuelle est illustrée dans la Figure 6.

Nom du fichier	Nombre de triangles	FPS
hero.mp4	3512	49
Hero_HD.mp4	6846	32
Hero_LD.mp4	1874	58
troll.mp4	3940	41
raptor.mp4	3630	42

Tableau 1. Résultats de simplification de la géométrie et FPS correspondants.



a) Modèle original, 6846 triangles, fréquence : 32fps

b) Modèle simplifié, 1874 triangles, fréquence : 58 fps

Figure 6. Modèle original de Hero et sa version simplifiée.

Une deuxième validation du processus automatisé proposé était réalisée dans une application complexe sur téléphone portable, appelée MESSAN. Dans MESSAN nous avons implémenté un service de communication innovateur pour les messages courts dans un environnement mobile. Dans ce prototype nous avons montré qu'en utilisant la méthode proposée pour adapter les avatars aux terminaux à faibles ressources par l'assistance d'un serveur, permet de le transformer en interface puissante pour l'auto-production de graphique 3D.

1.3.4 Implémentation du framework d'interopérabilité

La majorité de mondes virtuels et des jeux en ligne permettent de concevoir avec un éditeur, l'avatar du joueur. Ils offrent un ou plusieurs modèles d'avatars que l'utilisateur peut modifier. Les outils existant contiennent des éléments par défaut qui peuvent être manipulés pour personnaliser l'avatar. Le jeu inclut habituellement différentes formes de têtes et corps, la couleur de peau, la position et les modèles d'oreilles, des yeux, le nez et la bouche. De plus, certains d'entre eux offrent des ressources pour habiller l'avatar : vêtements, chaussures, accessoires, etc.

La réalisation d'un avatar à partir de modèles simplifie significativement la phase de conception, en permettant aux utilisateurs novices de créer des avatars. Cependant, cette tâche reste un processus très exigeant de temps de calcul, concevoir un avatar prend parfois plusieurs jours et généralement c'est un processus continu : l'avatar peut être mis à jour à tout moment. Ne pas avoir la possibilité de réutiliser l'avatar dans d'autre MV oblige les utilisateurs à personnaliser leurs avatars dans chaque MV séparément. Le framework d'interopérabilité proposé aborde cette problématique où l'ensemble de métadonnées caractérisant l'avatar peut être utilisé pour personnaliser un modèle préexistant.

1.3.4.1 Scenario d'un cas d'utilisation

Pour valider le framework d'interopérabilité d'avatar proposée, nous avons réalisé le scénario suivant : l'utilisateur crée un avatar dans n'importe quel MV qui expose ses ressources et caractéristiques. En utilisant le schéma proposé, il est possible de relier les caractéristiques d'avatar aux éléments standardisés. Ces éléments sont importés dans un deuxième MV qui "comprend" ce XML et permet donc la personnalisation de son propre modèle d'avatar. On peut considérer cela comme une automatisation du processus de conception d'avatar dans le deuxième MV. Il est probable que quelques caractéristiques seront perdues à cause des différences dans les modèles.

Une deuxième capacité utile fournie par le schéma proposé est la manière dont l'animation peut être déclenchée dans différents MVs. Nous considérons le cas où seulement les caractéristiques sont transférées, donc chaque MV utilise ses propres ressources d'animations. Si un avatar est en "exécution" dans MV1, simplement en envoyant un fragment XML comme illustré dans Figure 7, il déclenchera "exécution" sur l'avatar du MV2. Les deux animations peuvent être les mêmes ou légèrement différentes, tout de même le deuxième monde peut comprendre le XML transmis et exécute l'action exacte.

```
<?xmlversion="1.0"?>
<xml>
  <Walk>
    <DefaultRun>
      <Url>run</Url>
    </DefaultRun>
  </Walk>
</xml>
```

Figure 7. Exemple de transfert des caractéristiques d'animation de l'avatar.

Bien que le schéma proposé puisse être utilisé comme un conteneur pour les caractéristiques d'avatar, il peut se référer aux ressources externes pour la géométrie d'avatar, l'apparence et l'animation. Donc, étant complet et exactement le même, l'avatar peut "se téléporter" à travers les MVs. L'exemple donné dans la Figure 8, contient un fragment XML où le maillage est spécifié pour être utilisé à partir d'une ressource externe, appelée "myavatar.mesh". L'élément <AppearanceResources> du schéma est réservé pour cela.

```
<?xml version="1.0"?>
<xml>
    <AppearanceResources>
        <AvatarURL>myavatar.mesh</AvatarURL>
  </AppearanceResources>
</xml>
```

Figure 8. Exemple de transfert de maillage de l'avatar.

De la même manière, l'animation peut être chargée d'une ressource externe. L'élément <AnimationResources> est utilisé comme un détenteur de ressource d'animation.

Cependant, les implémentations usuelles de MV ne sont pas ouvertes pour importer les ressources auto-propriétaires. Dans un tel cas, l'alternative sera la visualisation de l'avatar sur des lecteurs externes. Puisque MPEG-V peut attribuer au MPEG-4 des ressources pour l'apparence de la géométrie et l'animation d'avatar, un lecteur MPEG-4 peut être utilisé pour la visualisation. La Figure 9a montre un avatar exporté d'un MV et visualisé dans un lecteur MPEG-4 fonctionnant sur un ordinateur de bureau standard et la Figure 9b montre le même avatar sur une plateforme de téléphone portable.

Figure 9. Visualisation de l'avatar dans un monde virtuel en utilisant un lecteur MPEG-4 sur PC (a) et sur téléphone mobile (b).

1.4 Conclusions et perspectives

Au cours de ces dernières années les Mondes Virtuels 3D (MV3D) sont devenus très populaires en offrant diverses fonctionnalités et expériences utilisateurs. Les utilisateurs sont intéressés par les MVs relativement rapides et les études prévoient que dans un proche avenir presque tous les internautes auront leurs représentations 3D (avatar) dans un ou plusieurs MV3D.

Dans ce travail, guidé par le fait que les MV3Ds sont des communautés en ligne, nous avons noté l'importance du transfert de données compactes à travers divers réseaux. Pour faire face à l'hétérogénéité de données des objets graphiques 3D et en particulier les avatars, nous avons proposé un modèle de données qui permet la compression efficace des données individuelles (la géométrie, la texture et l'animation) indépendamment du formalisme de représentation.

En outre, le développement actuel de technologies de téléphones portables rend ces dispositifs comme une passerelle d'accès au MV3D. Par leur nature, conçue principalement pour la communication, ils sont devenus des terminaux pour la convergence de services et l'accès aux médias combinant la voix, le texte, les images et le contenu vidéo. Cependant, le contenu est initialement conçu pour des plateformes puissantes, devrait être adapté. C'est le cas aussi des contenus graphiques 3D, où leurs natures hétérogènes rendent les opérations plus complexes. Donc, nous avons proposé une méthode pour adapter et livrer des avatars 3D de qualité pour des téléphones portables. Prendre une photo ou capturer une vidéo est un processus direct, cependant, il n'y a aucun dispositif pour créer un contenu 3D. Nous avons démontré qu'en combinant un dispositif à faibles ressources avec un serveur et en définissant un protocole léger de communication, nous le transformons en un puissant système, facile à utiliser avec l'outil de création de contenu.

Finalement, en abordant le problème de génération de contenu utilisateur dans des mondes virtuels, nous proposons un framework qui inclut une couche de métadonnées permettant de continuer l'utilisation des caractéristiques de l'avatar, en se téléportant à un nouveau monde virtuel. Cela offre une solution élégante de l'interopérabilité entre MV, des applications en général propriétaires.

Le travail présenté dans ce manuscrit a été inspiré et intégré dans plusieurs processus de standardisation MPEG. Quelques solutions proposées, ont été incluses dans deux standards MPEG, à savoir MPEG-4 Partie 25 et le MPEG-V.

Dans ce travail, nous avons premièrement, présenté un état de l'art sur différentes approches de représentation d'avatars, aussi bien que des techniques existantes pour les modéliser et les animer. Le graphe d'objet d'avatar, la géométrie, l'animation et l'apparence ont été étudiés comme des composants d'avatar. Ajouté à cette représentation, un cinquième élément, appelé "métadonnées" contenant les descriptions d'avatar de haut niveau, a été rajouté pour une description complète d'avatar assurant l'interopérabilité lorsque les quatre premiers éléments sont propriétaires. Différents standards et formalismes de représentation utilisés pour la définition d'avatar ont été présentés et plusieurs verrous ont été identifiés concernant la réutilisabilité d'avatars entre différents Mondes Virtuels 3D. Premièrement, considérant que les MV3Ds sont des environnements en ligne, le besoin de compression de données a été mis en évidence. L'étude des méthodes existantes a démontré les avancements majeurs dans les composants individuels, mais un manque est noté dans l'approche holistique et particulièrement le fait que dans des applications réelles le formalisme de représenter l'avatar dépend de l'application. Ensuite, les différentes méthodes d'adaptation du contenu graphique 3D sur les dispositifs à faibles ressources ont été présentées. Aujourd'hui, l'intégration des processeurs graphiques à haute performance sur les téléphones portables permet un meilleur taux de rendu de pixels et des triangles par seconde. Aussi les téléphones portables présentent des contraintes spécifiques comparés aux ordinateurs standards, ceci compromet leurs utilisations pour les applications 3D

temps réel. Finalement, nous démontrons que face au nombre accru de MV3D, l'interopérabilité devient une problématique. Donc les analyses de représentations d'avatars dans différents MVs ont été présentées mettant en évidence le fait que la représentation du corps humain, les émotions et l'animation, est spécifique au MV.

Ensuite nous présentons les contributions principales de cette thèse. Premièrement, nous avons proposé un modèle de données permettant de relier des outils de compression de médias individuels aux avatars arbitrairement définis. En analysant les approches actuelles pour compresser les composants clés de l'avatar - le graphe d'objet, la géométrie, l'apparence et l'animation - nous avons choisi un ensemble, permettant un compromis optimal entre le taux de compression et la complexité du décodeur. Nous avons proposé une architecture logicielle qui permet l'utilisation de ces méthodes sur n'importe quel formalisme de représentation XML représentant un avatar. Au lieu d'utiliser des méthodes de compression génériques pour tout le contenu (comme prévu par le zip par exemple), les éléments clés sont la détection automatique des médias individuels et la compression spécifique par les algorithmes appropriés. La deuxième contribution de notre travail est la solution d'adaptation du contenu pour chaque composant de l'avatar : graphe d'objet, géométrie, apparence et animation, appropriée pour les terminaux à faibles ressources. Le développement de cette solution est issu du compromis entre la résolution d'écran du dispositif et la résolution du contenu (le nombre de triangles des maillages, le nombre de pixels des textures, le nombre de segments et les trames d'animation). La troisième contribution s'intéresse à l'interopérabilité d'avatar entre MVs. Basée sur l'approche la plus utilisée dans les MVs pour définir les avatars basés sur les modèles pouvant être personnalisés par l'utilisateur, nous avons défini un modèle de métadonnées des paramètres de personnalisation. Donc, un avatar arbitraire peut être représenté comme une fonction du modèle privé du MV associé à un ensemble de paramètres définissant l'apparence, les animations et d'autres données spécifiques. La relation qui lie cet ensemble de paramètres d'un MV à un autre est l'approche d'interopérabilité d'avatars. Donc nous avons proposé une méthode de les représenter comme "métadonnées" associées, complétant la définition d'avatar et utilisé pour reconstruire un avatar arbitraire.

Ensuite nous avons présenté la validation expérimentale des trois contributions. Le modèle de données de compression a été implémenté comme une paire de codec (encodeur et décodeur) et fournit une compression totale de 1:20 de la représentation XML d'avatar. Nous avons présenté numériquement les avantages de l'approche d'adaptation pour un terminal à faibles ressources, l'animation était réalisée avec une fréquence de 40-50 fps, en préservant la qualité visuelle. Finalement, deux tests de conformité ont été présentés pour démontrer l'interopérabilité d'avatars dans les mondes virtuels.

Le travail présenté dans ce manuscrit comprend trois contributions proposées pour pallier au problème général d'accessibilité aux mondes virtuels en ligne et avec différents terminaux d'utilisateurs. Toutes les contributions sont dédiées aux avatars et ne résolvent pas le problème complexe de l'interopérabilité logicielle entre des mondes virtuels, une problématique qui fait partie de nos travaux de recherche actuels.

L'interopérabilité d'avatars est en partie une perspective de notre travail. Lors de la création d'Internet dynamique, dense, qui deviendra 3D et mobile, les nouvelles recherches s'intéresseront à l'amélioration du sens de présence sociale de l'avatar dans un MV et le connecter avec le profil utilisateur.

Le but final de l'interopérabilité entre des mondes virtuels est la définition d'un langage de représentation commun, de la même façon que HTML (maintenant enrichi par javascript). Les implications d'un tel langage peuvent être prévues en analysant l'utilisation de Second Life (SL) et Facebook (FB) dans les sept dernières années.

SL a été annoncé en juin 2003 et sa popularité a grandi très vite. Cependant, au moment de rédaction de ce manuscrit, SL a perdu son élan initial et compte seulement environ 70

000 utilisateurs. D'autre part, FB a aujourd'hui plus de 500 000 000 utilisateurs. Une première question est pourquoi SL était initialement si célèbre puisque le concept de MV n'était pas nouveau en 2003. La réponse la plus probable est que la nouveauté dans SL était le concept des réseaux sociaux. À la différence des MV3Ds ou des jeux 3D antécédents, SL n'a pas spécifié les applications ou les rôles d'utilisateurs. Tout utilisateur crée sa propre personnalité, de la même façon que FB, mais avec les avantages de graphisme 3D comme support de représentation d'apparence. Cependant, la deuxième question est pourquoi SL, malgré l'existence d'outils renforçant l'expression de l'utilisateur, est maintenant moins étendu que FB. Des éléments de réponse sont le besoin d'installer un client, tandis que FB est accessible par un navigateur, et aussi le besoin de terminaux puissants, tandis que FB est accessible même sur des téléphones portables, et enfin la faible interopérabilité avec d'autres applications, tandis que FB dispose d'un ensemble d'APIs, permettant aux applications tierces d'avoir un accès facile au contenu généré par l'utilisateur.

Nous sommes fortement persuadé que l'avenir des MVs est éminent. Même s'ils se développent plus lentement comparé aux réseaux sociaux 2D, cependant leur nombre augmente chaque année. Le défi réel est faciliter l'accès et créer des outils puissants pour la création du contenu afin de transformer Internet d'aujourd'hui en un espace social 3D.

2 Introduction

Advances in real-time graphics research and the increasing power of mainstream GPUs make it possible to handle complex 3D content and have conducted to the exponential development of 3D graphics applications. However, the content chain remains a complex one, especially for on-line, real-time applications increasing their development cost. In addition to the need for developing methods and tools for compacting and adapting the content, the interoperability within the chain becomes a must.

A specific place within the 3D graphics content is occupied by avatars. They are complex structures composed of data of various kinds such as meshes, images, animation parameters. In addition, they are the representation and interaction support of the end user within Virtual Worlds or other on-line environments. In this chapter, we present different tools developed in the past to represent and animate the avatars as well as recent online applications using them.

2.1 3D Graphics Representation in Online Environments

Computer graphics is currently the spine of several industries of information technology. Especially in applications such as virtualization of 3D environments and video games, 3D graphics becomes a valuable media with an impact similar to video and audio. In a connected and very heterogeneous information society, such digital assets should be exchanged between producers, service providers, network providers and end-user devices.

The complexity of creating 3D graphics content leads to the coexistence of several authoring tools (AT), some of them specialized for particular tasks. Therefore, it is possible to use several of them to create a single asset. Since almost all the authoring tools have their own file format, portability between authoring tools is an issue and data format convertors should be provided.

Different approaches to address the issue of specific data format for different ATs were introduced. One approach is to transcode from one format to another. Being based on mapping between elements of the two formats, the drawback of this approach is that the ATs are not covering the same space, therefore the mapping is not perfect. Subsequent "hand tuning" may be required, usually by an experienced content author and in the most cases specific features cannot be transcoded. Furthermore, a transcoder tool needs to be provided for each authoring tool in order for that file format(s) to be converted into the format(s) supported by other authoring tool. As more formats and authoring tools are introduced, the number of transcoder tools required increases exponentially, which in turn makes maintenance and support difficult if not entirely unfeasible. Transcoding to and from an intermediate format reduces this concern; for each platform only two transcoder tools are required. This second approach is much more cost-effective, with the condition that a good intermediate format exists. The latter must be complete or extensible enough to represent all of the features required by each platform, so that no data will be lost during the transcoding process.

Furthermore, authoring tools can directly support intermediate open file formats and skip the transcoding process altogether. The ideal option is to add a plug-in to an authoring tool to directly import and export to that complete intermediate format. This is the best approach in terms of the use of the content, since a user can export the file once and then import it directly into other programs without being required to run a transcoder every time. In most cases, writing this kind of direct import/export plug-in is easier than developing a stand-alone transcoder. The intermediate format is also a good solution when vendors wish to keep their native file formats confidential.

In the last decade, several efforts have been made to develop a unique data format for interchanging assets between authoring tools. In the category of open standards, X3D (the evolution of VRML97) and COLLADA are the best known, the latter probably being the most adopted by current tools. They include a rich set of tools for representing 3D data sets (geometry, texture, animation, physics…). Although different, these interchange standards have some common characteristics. One of them is that the data representation is based on XML. Using XML allows a very easy extension of the structure, like an impregnation of data with meaning and purpose, which implicitly makes it information. Since 3D graphics data is usually a complex structure of heterogeneous information, organized in graphs (object graph, scene graph) and containing several elements (e.g. geometry, appearance), XML is an appropriate mechanism to express it.

Recent developments of collaborative sharing platforms (the well-known Web2.0) are efficient vehicles for pushing media over the Internet, and introduce the additional need of compactness in data representation.

In the complex content chain, there are on one side the authoring tools, using a data structure appropriate for content editing (such as modification stacks), and on the other side rendering engines where data is as flat as possible. In both cases, the data size is not an issue. In between, the data should be transmitted, therefore compressed (Figure 10). While the compression exploits the signal redundancy, therefore changing the data structure, it should also be as transparent as possible in order to maintain the properties of the content designed by the content creator.

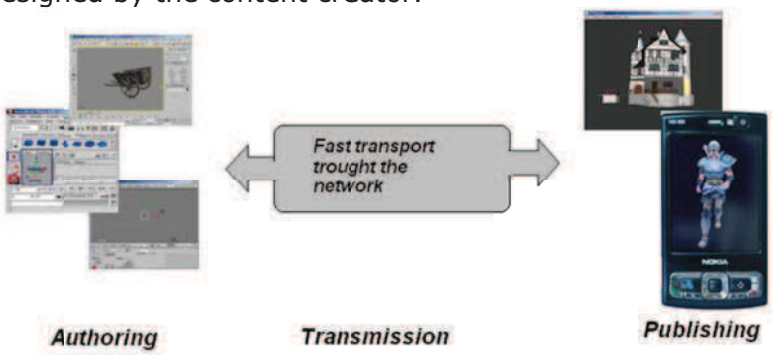


Figure 10. The content chain in an on-line environment.

Even in the case of games, which have traditionally been distributed in DVDs containing all assets, openness to new distribution channels is a must, either for entire game downloading before playing, or for the streaming of assets during the match in the case of on-line games. Neither the formats used in the content creation phase, nor the ones used for rendering are appropriate for transmission, since they usually satisfy none of the three requirements that are most important in this context:

- **compression**, to reduce the size of the transmitted data;
- **streamability**, to enable players to start using the content before having downloaded it entirely;
- **scalability**, to ease real-time adaptation of the content to a specific platform and network.

A recent trend in 3D graphics applications are the 3D Virtual Worlds (3DVWs). Compared to the existing applications, they are more open with respect to the possibility to personalize the content. Initially conceived for social purposes as a support for communication (i.e. chatting) and offering awareness on the presence and sometimes the mood of the interlocutors, 3DVWs are now reaching a milestone: the technology for representing and visualizing 3D content becomes available and widely accessible. One facilitator is the fast development of high performing 3D graphics cards such the ones of Nvidia, ATI) and their availability in ordinary computers - trend driven by the powerful market of computer games - making almost any Internet user a potential player of 3DVW.

Fast development of VWs also results from the increasing number of VWs users and their demands and requests for this kind on services. Both younger users, representing the "messenger generation", and elder ones, using VWs as a tool to express themselves or for work, are represented. Connecting real economy in VWs also makes them trendy and promising. After an enormous step towards the awareness and democratization of virtual worlds provided mainly by the marketing success of Second Life, VWs are now looking for sustainable business models. The most probable situation is that, in the near future, more and more virtual worlds will be available, offering complementary functionalities and user experiences.

Another specificity of the Virtual Worlds with respect to other multimedia applications consists in the (visual) representation of the user inside the environment. Such representation takes usually the form of an avatar, a graphic object that serves different

purposes such as making the presence of a real user into the VE visible, characterizing the user within the VE, or as mean of interaction with the VE.

The users' presence in a virtual environment is highly dependent on the amount of their immersion. Creating more realistic looking and behaving avatars enables a higher level of identification with the character and therefore an increased level of presence. While this property was extensively exploited in the past in successful games, the avatars being the representation of the player in the virtual world and at the same time the interface with it, the trend of using the avatar as identification is nowadays more and more present in the recent developments of the Internet, especially under the Web2.0 community-based technologies. In this context, having a visual representation of users in the form of avatars holding personal information, history, personality, skills, etc. becomes necessary.

However, today's Internet (in the sense of its initial definition as interconnected sites) is not yet the lively place foreseen by Virtual Worlds potentials. There is no interoperability yet between different web-sites implementing Web2.0 features - beside only a very thin layer of re-using IDs (e.g., OpenID) or aggregating different sources.

Difficulties, time consumption and exceptionalities to reproduce the content bring up the more and more important issue of interoperability or at least re-usability of assets, in particular the avatars, between VWs. In the last two decades, the research community actively worked on developing tools for the creation, representation, animation, transmission and display of avatars and several standards and recommendations were created to represent graphics objects and in particular avatars. Standards such as X3D and COLLADA ensure the representation of graphics assets and MPEG is a very active community in providing tools for compressing them. They can be used as a base layer for ensuring interoperability at the level of data representation. However, only representing the media is not enough for ensuring interoperability in VW. Therefore an additional requirement is the following

- the definition of **a metadata model** that, combined with media representation, will ensure a complete interoperable framework.

Let us note that current VWs are populated with other types of objects than avatars, which are not directly driven by end-users. The modeling and animation of some of them (especially animals) can be very similar to the techniques used for human avatars. However, behind this type of objects there is no end-user, only computer programs to drive them. In the perspective of the current research, an avatar can be whatever 3D object that represents and is driven by the end-user.

2.2 The Place of Avatars in 3D Graphics

In the most common definition, the word avatar signifies *an embodiment, a bodily manifestation of the Divine* or *an incarnation*. In religion, when *God needed to come down among the people* it was identified as an avatar. Nowadays, the term avatar in computer graphics has preserved the meaning; however, *God* was replaced by *Humans* and the Real World by Virtual Worlds.

The interest in person representation, and as a special case, self-representation, is not new. It goes deeply into humanity's history and is part of human psychological and sociological motivations [Georges09] to dispose visual representation of oneself in order to position with respect to the others and to the environment.

Examples from cave paintings dated 32 000 years ago [Clottes03] representing animals and humans prove the interest in self-representation. With the development of painting, the portrait was introduced as a separate style mainly aiming to represent a person, and also to display his likeness, personality and mood. The art of the portrait thrived in sculpture, where statues as a separate branch of sculptures representing a person or an animal emerged.

In addition to the static representations described above, the possibility to evolve in time and move in the scene, animation, is also present in the history of mankind. However, the possibility to create and visualize animation became available in the more recent history. Concrete evolution in animation was achieved with the invention of cinematography. Character animation appears as a more recent example of animating human beings. It is unique because, in addition to physical action, it involves the creation of apparent thought and emotion. The first example of character animation is considered to be the cartoon "*Gertie the Dinosaur*" by Winsor McCay (1914).

Different techniques are used to produce animation. In general, each of the existing animation techniques can belong to one of these three groups: traditional animation, stop motion or computer animation.

In traditional animation, the illusion of motion is created by capturing several images and then creating a sequence from them; the images are drawn one by one by animators. Stop-motion is a kind of animation where instead of drawing images, one captures real objects posted in different positions. The third group is computer animation, an approach that includes a variety of techniques, with the unifying factor that the animation is created digitally on a computer. Early on, the production was concentrated on 2D animation. More recently, 3D animations were developed, created by an animator while manipulating 3D digital models. Dedicated tools for 3D content and animation creation, such as 3DS Max, Maya, Blender, Motion Builder exist for this purpose.

Major milestones in animation techniques were achieved during the last century. We summarize this progress in Figure 11, where some animation techniques and animations created with them are presented. A detailed historical review of these techniques is provided in Annex A.

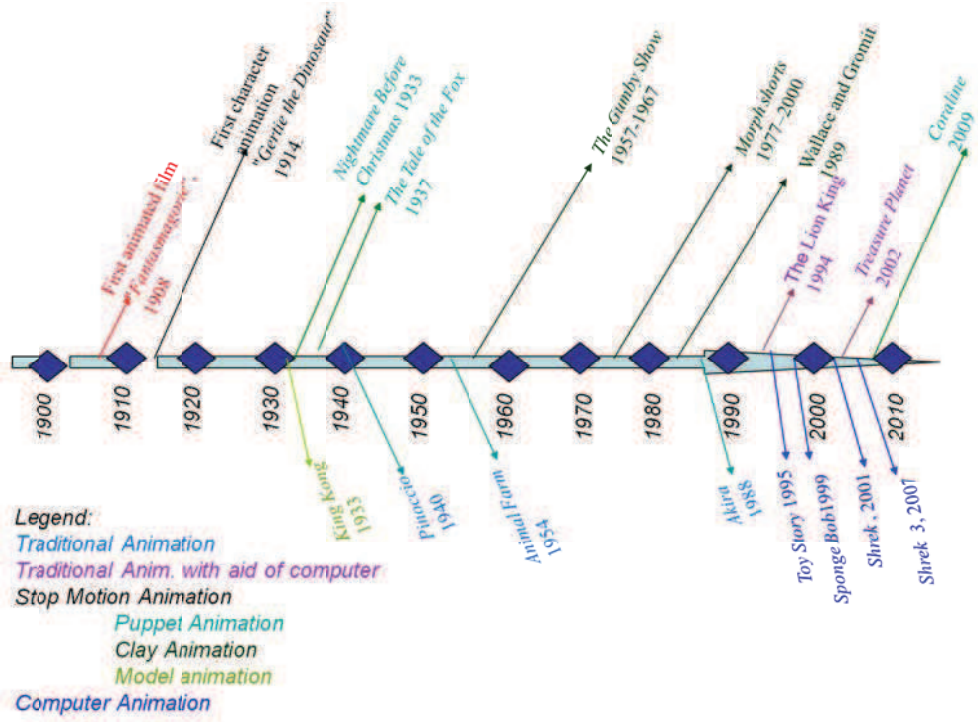


Figure 11. Animation techniques and animation production in the last century.

Movies are just one example where 3D graphics and particularly 3D avatars are used. Having in mind the fast growth of the computer industry and the similar development of computer animation, predictions of the future of computer animation are very optimistic. With the appearance of the Internet and its fast evolution, many users can profit from 3D graphics and can represent and express themselves through new communication

channels. Different kinds of 3D applications are developed to facilitate and enrich real life. Some of them are presented in the next section.

In the last decades, the interaction between the real world and the so-called virtual world became an everyday activity. One trend in the recent developments of the Internet, largely known under the name Web 2.0, is to have a digital identity. In many cases the user's presence in the digital world does not have to be visually signalized, only the effects of their intervention being observable (i.e. adding comments to an article on a web-page, editing a section of a Wikipedia document). In some other cases with real-time requirements, such as collaborative work or presence in 3D virtual worlds, the visual representation (i.e. the avatar) is a communication vector by itself - it helps in identification, differentiation and identity protection, and it is a facilitator of communication, informing the others about status and availability.

Novel scenarios in the web communication include two main concepts: need of real time communication between users and need of real time interaction between user and the Web content. To achieve these, one possible evolution is that the future of the Web will be 3D and Virtual Worlds where users may have their own interactive representations. These representations can be the 3D avatars, acting like a container of all the users' personal information: from appearance to emotion status and personal interests. The avatars become the support for interaction as well. With the avatar, the user can directly participate: he can communicate with other avatars, he is able to speak and ask for help from 3D agents, buy from virtual sellers, learn from virtual avatars and perform other actions as similar as possible to reality and with the big advantage of removing the limitations due to physical space. The uses of the avatars in real-time communications, Virtual Worlds and games are well-known. Among them, Second Life, Active Worlds, There, Sims, HiPiPi, IBM QWAQ (mainly dedicated to project management and document sharing) are very popular. Some illustrations of current successful 3DVWs are provided in Figure 12.

Figure 12. Snapshots from 3D VW, IMVU and Second Life.

Moreover, several applications use the virtual characters as web-assistant, thus facilitating web navigation, or for teaching dedicated skills. The avatar becomes an agent, being empowered with some intelligence. Examples include agents trained for teaching Braille alphabet - alphabet for the blind , Sign Languages for deaf people (SL) etc.

Another kind of applications using avatars are the one with impact on real economy or communication. Based on avatar representation, the efficiency of distance-communication, such as conferences, meetings and plan decisions, can be increased.

Several examples where avatars are used in online applications are presented in Annex B.

Nowadays, it becomes almost a practice (driven by user's desires and needs) that applications firstly designed for PC can also run on the mobile phone, adapted as best as possible. For example, many web pages, chat applications and social networks (like MySpace and Facebook) can be reached by mobile. Reading e-mails, voice-call, streaming video and audio are also available. Challenging applications for mobile devices become the ones that are using 3D graphics and real-time communication, such as 3D VWs for instance. Some mobile applications that are using 3D graphics applications are, for example: Cellufun[5] (illustrated in Figure 13.a), Itsmy[6] and Sparkle[7] (illustrated in Figure 13.b).

a) b)

Figure 13. Avatars on mobile phone. Cellufun (a) and Sparkle (b).

Still, these kinds of applications are in their beginnings. The difficulties to develop them are due to the constraints of the mobile phones on one side and to the complexity of the applications themselves on the other hand.

2.3 Conclusion

This introductory section highlights two main issues: firstly, it provides an overview of the complexity of the 3D graphics content chain in an online environment and highlights the need for compression, adaptation and interoperability. Secondly, it presents the place of avatars as specific type of 3D graphics content and the variety of applications where they are used and useful.

The specific interest in the avatars has two main reasons: as 3D objects, they are complex structures composed of data of various kinds such as meshes, images, animation parameters. Therefore, almost all techniques developed for avatars can be applied on arbitrary 3D objects as well. A second property is that avatars are the representation support of humans in a Virtual World. As we presented in this chapter shortly, humans through the centuries desired to represent themselves with respect to the others and the environment. In today's world, it is noticeable that computer animation allows creating and animating human-like avatars more easily compared to the past. Furthermore, computer-based techniques are developed and improved rapidly and it can be expected that in the near future using avatars will be simple enough, so almost each Internet user will have the possibility to have his/her own 3D virtual representation.

In such an environment, we recognized the indispensible need of interoperability of the content, including avatars. The online nature of current applications also brings up the requirement of compression. Finally, the current trend of going mobile when applied to graphical content and especially avatars introduces new challenges with respect to

[5] http://www.cellufun.com/
[6] http://mobile.itsmy.com/
[7] http://sparkle.genkii.com/

content adaptation. This manuscript contributes to these three aspects in a common framework as explained in detail in the following chapters.

3 Avatar Modeling, Compression, Adaptation and Standard Representations

Due to the complex structure and real-time simulation performances, dealing with avatars constitutes one of the most challenging process in a virtual environment. The avatar, as a replica of the user, is in direct relation with him; therefore it should satisfy several requirements in terms of user experience. Creating this kind of efficient and realistic human representation on the one hand and optimally fitting it with the resources on the other requires a deep understanding of the avatar characteristics.

In this chapter, the first target is to introduce several notions related to avatar representation. The four components traditionally defining the avatar - the object graph, geometry, appearance and animation - are described, as well as techniques to create/capture them. We extend the quadruple by considering an additional element – the avatar characteristics grouped as "avatar metadata". Afterwards, two signal processing domains – compression and adaptation - are surveyed with respect to the specificities concerning avatar representation. As a base for defining the metadata framework, we analyze different virtual worlds and representation standards. Finally, the chapter is concluded by reporting the main deadlocks and introducing places where improvements are needed.

3.1 Avatar Representation

As a 3D graphic object, the avatar is represented by a quadruple defined by object graph, geometry, appearance and animation. Such kind of information allows to visually represent the avatar, but does not provide high-level data concerning its usage conditions. Therefore, we complete the classical quadruple with a new member, called avatar metadata.

Within this manuscript we consider the avatar defined by

$$Avatar = \{ObjectGraph, Geometry, Appearance, Animation, Metadata\}$$

In the following sections we analyze the five components with respect to their representation approaches and techniques and to the tools for obtaining them.

3.1.1 Object Graph

The term *graph* defines a data structure created by a set of hierarchically organized entities called *nodes*. Some of the objects composing the scene are simple enough to be presented only by a single node (such as the lights); however, in general, the objects are represented by a heterogeneous set of nodes. Avatars, as multipart objects, belong to the second group. The avatar graph is composed of nodes for describing the logical and spatial relationship of the various components. The latter can be separated in two groups: basic and technique-specific. The first group includes nodes used for geometry, appearance and animation description. The second group includes nodes used for the skeleton structure, the morph targets, physics-related parameters, etc. Figure 14 illustrates an example of an avatar graph.

Figure 14. An avatar object and the associated graph.

Representing the container of elements that compose the avatar, the creation of the avatar-graph remains a manual task, performed by using content creation tools. The process is automated inside the tool, mainly by using templates, yet hand-tuning is needed to personalize them, for example importing objects (avatar or its parts) from external resources or controlling at the basic primitives level (vertices, edges, object, bone, biped). To illustrate the process, several stages of creating an avatar graph by using 3DSMax are presented in Figure 15, Figure 16 and Figure 17.

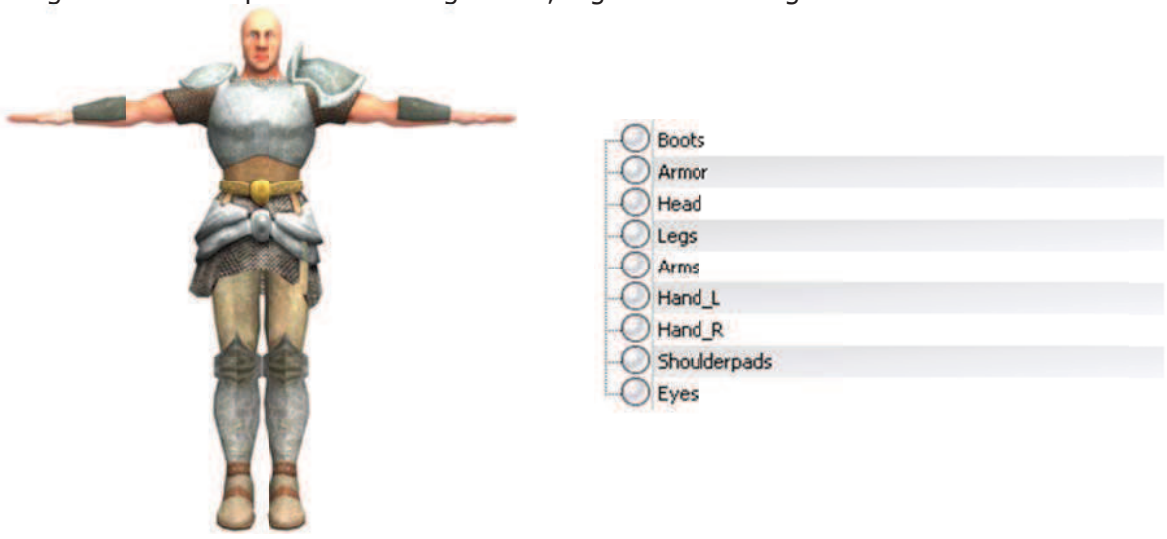


Figure 15. The avatar graph with only mesh nodes.

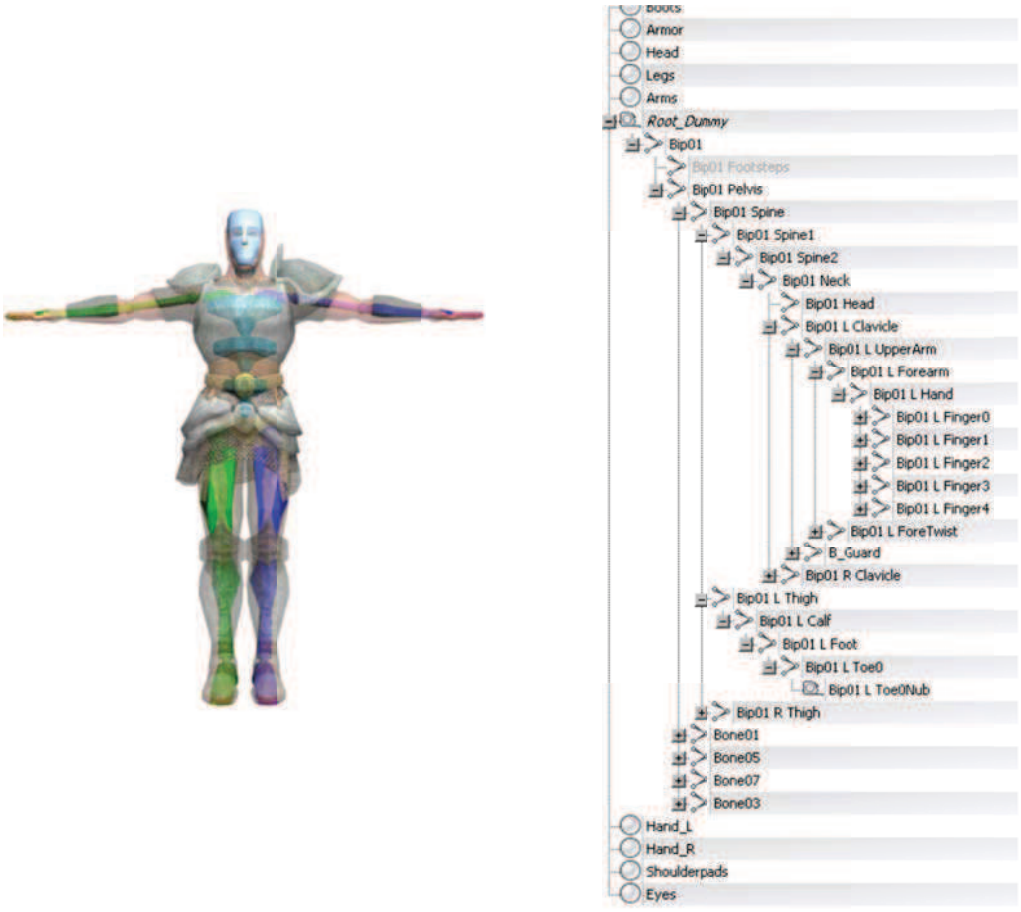


Figure 16. The avatar graph with meshes and skeleton.

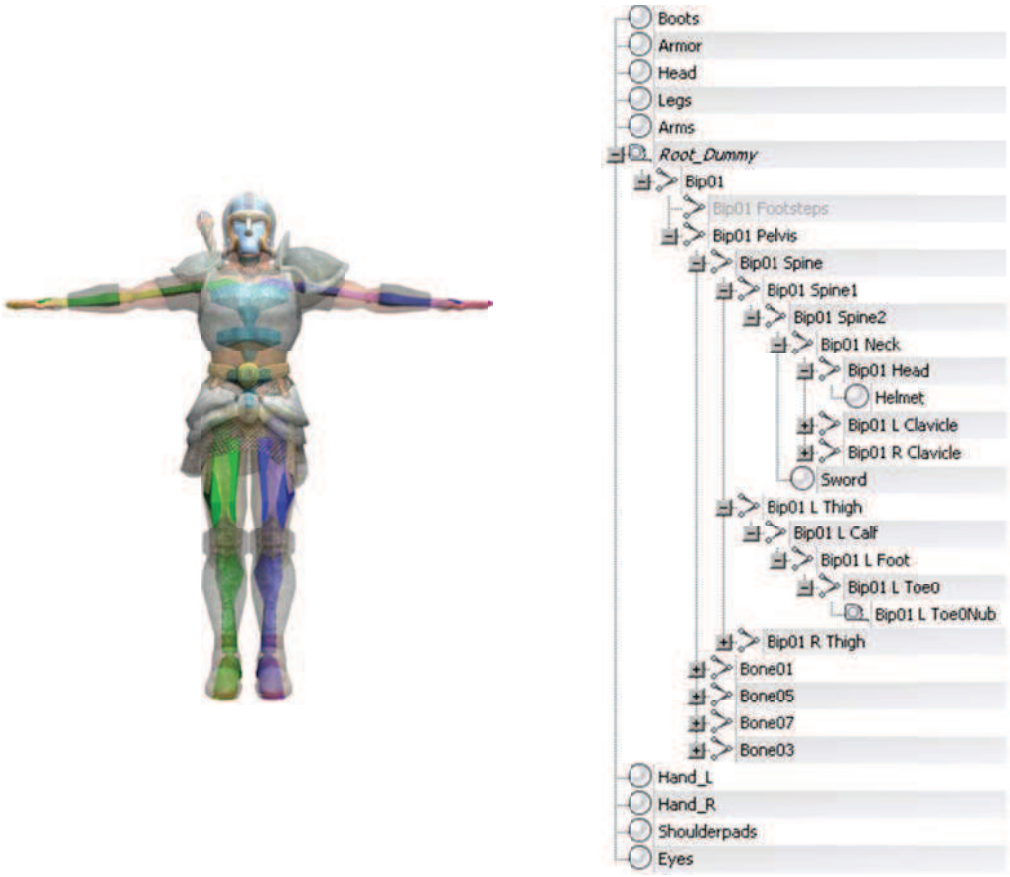


Figure 17. Avatar with mesh, skeleton and external nodes (helmet, sword).

3.1.2 Geometry

The representation schemes of solid objects are often divided into two broad categories: space-partitioning representations and boundary representations (B-reps), although not all representations fit exactly into one or the other of these two categories [Requicha80].

Space-partitioning representations are used to describe interior properties, by partitioning the spatial region containing an object into a set of small, non-overlapping, contiguous solids (usually cubes) known as voxels. The properties of solid objects (e.g. occupancy, color, density...) are stored in each voxel. A common space-partitioning description for a three-dimensional object is an octree representation. Some advantages of this representation include simple inside/outside test and easy interior representation of the object. Nevertheless, its limitations are that it exposes a non-smooth object representation, high memory requirements and it is time consuming to manipulate and render. This makes this representation not optimal for avatar representation, especially for real-time applications.

B-reps describe the object as a set of surfaces that separate its interior from the environment. In computer graphics, the avatar is in most of the cases defined as a boundary representation. The boundary representation is usually represented as a mesh providing an approximation of the object's surface. This approximation may be parameterized in several manners, among the most widely used being polygonal facets, Bezier surfaces, NURBS and polygonal mesh subdivision.

Polygonal facets are the most commonly used B-rep method where the process of creation involves the direct manipulation of polygons, vertices and edges to produce the desired shape. The key advantage of this representation is its simplicity, making it

optimal for real-time visualization. The main disadvantage is the need of a big amount of vertices to model realistic forms. Figure 18 illustrates the 3DSMax interface and a mesh based on polygons as well as the elements composing it (vertex, edge, face …).

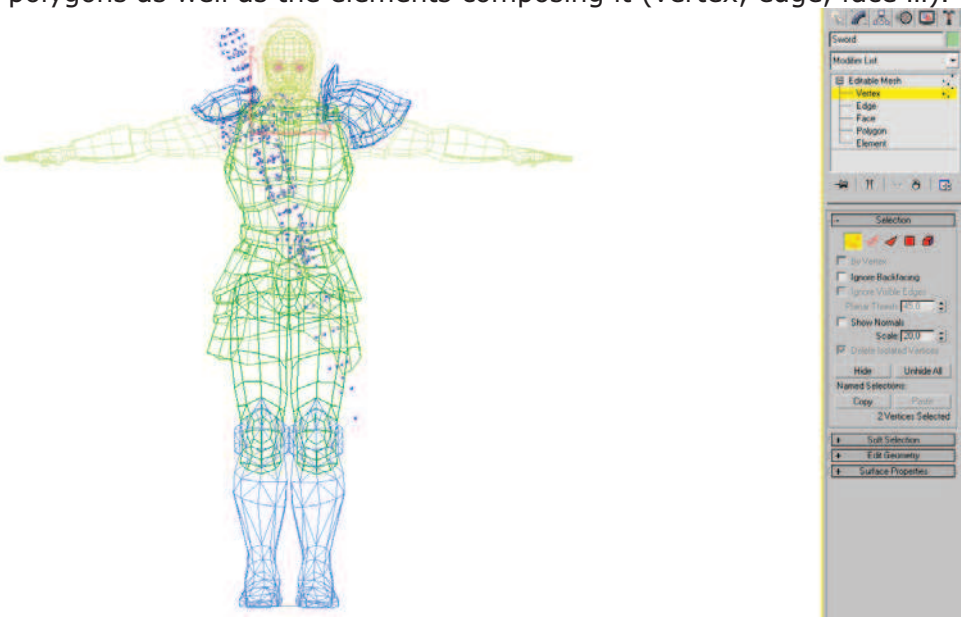


Figure 18. Example of a polygonal mesh.

The polygonal meshes may be represented in a variety of ways, using different methods to store the vertex, edge and face data, among which the best-known are detailed in Annex C.

The choice of the data structure is governed by the application, the performance required, the size of the data and the operations to be performed. Among all the representations described in Annex C, a Face-Vertex mesh is the most widely used mesh representation because of its low complexity and memory storage requirements. This representation is the typical input accepted by modern graphics chipsets and the most efficient in terms of rendering.

The Bezier surfaces are a combination of two sets of Bezier curves, which in turn are an approximation of spline curves developed for use in computer graphics. They are defined as a mesh of control points which are used in a parametric function to calculate the surface. Since they have a higher order than polygon surfaces, they offer a better representation of a smooth surface, are much more compact and easier to manipulate. However, their rendering is not optimal, requiring to be converted to a polygon surface in the rendering pipeline before being sent to the renderer.

As the Bezier surfaces, NURBS are also an approximation of spline curves and surfaces, however of a higher order. The shape of the surface is determined by control points; nevertheless, compared with Bezier surfaces, NURBS allow improved control on the shape. The trade-off is that NURBS are more complex to calculate.

Trying to capture the advantages of polygonal meshes – simplicity and of high order parametric surfaces – the continuity, the polygonal mesh subdivision representation is a method of representing a smooth surface calculated from a coarse polygonal mesh as the limit of a recursive process of subdividing each polygon into smaller polygons that better approximate the smooth surface. The process starts with a given polygonal mesh and a refinement scheme is then applied to it by subdividing it, thus creating new vertices and new faces. The positions of the new vertices in the mesh are computed based on the positions of nearby old vertices and according to functions ensuring high order continuity. In some refinement schemes, the positions of old vertices might also be altered. Figure

19 presents three stages throughout the mesh subdivision representing the avatar and its correspondent solid appearance.

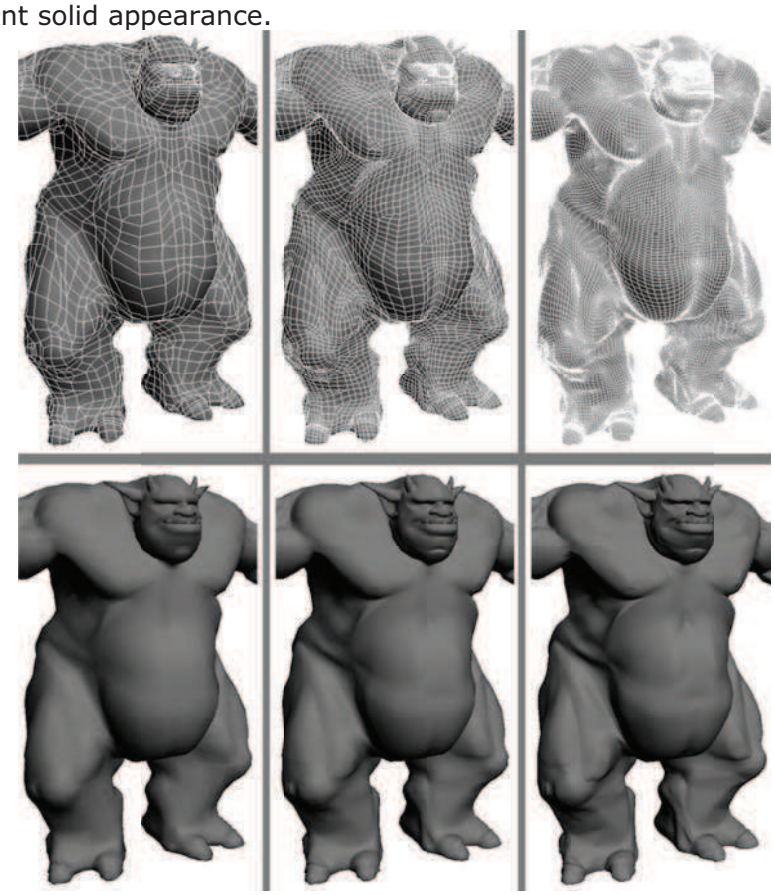


Figure 19. Polygonal mesh subdivision.

The decision about which part of the mesh should be subdivided can depend or not on the camera position and frustum. In the first case the subdivision is view-dependent, in the second non-view dependent. Although the first approach is appropriate for rendering large meshes such as terrain, its complexity makes it less appropriate for avatars.

Meshes define the form of the avatar, while enhanced visual effect is achieved by defining the avatar appearance, components that are presented in the next section.

Several authoring tools contain geometry generating mechanisms making it possible to model the virtual character [3DSMax, Maya]. The main drawback of this method is that the result is strongly dependent on the designer's artistic skills and experience. In addition, this procedure is time-consuming. One approach, inspired by the fact that people can express themselves more easily sketching their ideas on paper, then manipulating with 3D authoring tools, was designing sketching tools. On the other hand, faster methods are trying to create an electronic representation of an object (avatar) with the help of an existing real object and/or with the help of collected knowledge or properties about it. In the next sub-sections different approaches are discussed.

3.1.2.1 Sketching Tools

The basic idea of sketching tools is to allow user to draw only 2D lines on the screen and then the tool infers additional information (in the case of 3D modeling, the system infers missing depth information) and creates the 3D model. When sketching systems are designed, two most important issues should be dealt with. Firstly, algorithms to infer the missing information from the user input should be defined. This is not a straightforward task since there is an infinite number of possible 3D configurations that match the given 2D input. The usual approach for solving this problem is to design different algorithms

that rely on special domain knowledge, thus, since specific domains impose certain constraints on the possible 3D configuration, the system can return reasonable results. Secondly, it is important to create a user-friendly sketch interface, easy enough to manipulate and with the possibility to modify the result.

Several known sketching tools for 3D authoring include: Teddy system [Igarashi99], FiberMesh [Nealen07], Plushie [Mori07].

3.1.2.2 Capturing Real Objects

The aim of this method is to create a model by 3D scanning of a real object, by capturing its shape, color, reflectance and other visual properties. In principle, 3D scanning is similar to a number of other important technologies (like photocopying and video) that quickly, accurately and cheaply record useful aspects of physical reality. The scanning process can be structured according to the following steps: acquisition, alignment, fusion, decimation and texturing. The first step aims at capturing the geometric data of the 3D object. Depending on the type of scanner used, the execution of this phase can vary considerably. Either a single scanning is enough to capture the whole object, or series of partial scans (called range maps) are needed, each of them covering a part of the object. In the latter case, range maps taken from different viewpoints have to be aligned, which is the task of the second step. This procedure can be completely automatic if the exact position of the scanner during each acquisition is known. Otherwise, a manual operation is needed to input the initial placement, and then the alignment is performed automatically. Once aligned, the partial scans need to be merged into a single 3D model ("fusion" step). Because 3D scanners provide a huge amount of data, a "decimation" step is required. For an effective use of the model, one has to reduce the size of the acquired geometric information, especially of the less significant parts of the object. Texturing is usually achieved by using pictures taken during acquisition. The pictures are first aligned to the geometry (manually or automatically) and then mapped to the model.

Figure 20. Modern handheld 3D scanner[8].

Nowadays the process remains the same; however, modern scanners can make the entire chain automatic. For example, handheld laser scanners are used today in the industry (Figure 20). These scanners present some revolutionary innovations, such as self-positioning, versatility for scanning, portability and offer accurate results. The scanner can acquire objects of any shape, size or color. One approach using this type of camera is described in [Fudono08].

3.1.2.3 Vision-based Capture

A recent trend supported by the development of vision systems consists in capturing real persons by using one or several cameras and reconstruct or modify an existing template by using real measurements. An early example of this technique is presented in [Devernay94], where a method that uses directly the captured images to compute the

[8]www.zcorp.com

shape properties is proposed. Monocular images are used in the approach described in [Hilton99], where the geometry obtained is mapped on a previously created model, providing a cheap and useful approach for automatic modeling. By using stereo or general multi-view systems, the 3D geometry may be recovered more accurately. One method consists in computing the disparity map from a stereo pair of images and some local differential properties of the corresponding 3-D surface such as orientation or curvatures. The usual approach is to build a 3-D reconstruction of the surface(s) from which all shape properties will then be derived. When more cameras are used, such as described in [D'Apuzzo99], 3-D least squares matching techniques can be employed to obtain the geometry and skeleton. Some more recent techniques [Nebel02] can obtain both the geometry and the skeleton.

The main challenge to these approaches is to detect apparent features on the model surface, to differ correctly ambiguities in the image data and to decrease degeneracy in camera motion. Some of the latest approaches combine image data and real-time user interaction over the image to overcome these problems and create higher quality 3D models. In [EosSystems05], a system is explained that allows better 3D models to be created by marking up the structure in one or more images, while measurements from the scene and the cameras are input manually. Wilczkowiak [Wilczkowiak05] presents a more general approach to interactive modeling based on parallelepipeds as scene primitives. However, this still requires the corners of the primitives to be marked manually in each image. Quran et al. [Quan06] have developed an interactive method of modeling plants from image sets which uses strong priors to develop models of this particular class of objects. The prior in this case is specific to the particular variety of plant being modeled. Some approaches are creating 3D models from video sequences with user interactivity [Hengel07].

The presented approaches are highly used in offline 3D object creation; independent of the approach, two phases executed in a row are common: an image sequence or video collection and a processing phase. Usually, if after the second stage the model is found to be inconsistent, additional images or an entire video sequence must be collected and the reconstruction phase repeated until an acceptable model is obtained. Some new techniques propose on-line object reconstruction. The method proposed in [Pan09] uses a simple webcam. The object is moved about in front of the webcam and the software can reconstruct the object "on-line" while collecting live video. The system detects specific points on the object to estimate its structure from the motion of the camera or the object. These points are used to reconstruct the mesh.

3.1.2.4 Measurement-based Modeling

Introducing anthropometry (studying and collecting human variability in faces and bodies) in computer graphics [Dooley82] made it possible to create a parametric model defined as a linear combination of templates. The basis is extracted from large databases including human measurements such as NASA Man-Systems Integration Standard [NASA95] and the Anthropometry Source Book [NASA78], and several methods of exploiting it are provided in [Seo02, DeCarlo94].

An alternative method consists in defining a default model *a priori* and declaring on it anthropometric parameters with which the model can be deformed. The deformation on the model can be rigid, represented by the corresponding joint parameters, and elastic, which consists essentially in vertex displacements. Then a dataset with the relations between the value of these parameters and the shapes of corresponding models is created; from this dataset interpolators are formulated for both types of deformations. The joint parameters and displacements of a new model are created just by applying the interpolators on a template model with new measurements.

In [Seo03], instead of statistically analyzing the anthropometric data, the direct use of captured sizes and shapes of real people from range scanners is employed in order to determine the shape in relation to the given measurements. Some recent techniques

allowed the creation of 3D objects from user input data and apply it on previously defined templates. As an example, in [Karpenko06] a system is described that reconstructs the points of a 3D model only from the user-input.

3.1.3 Appearance

One of the basic mechanisms to improve the avatar appearance [Maïm08] is defining attributes such as colors for each polygon or vertex of the avatar, as illustrated in Figure 21b. If large numbers of polygons are used, the processing time needed for rendering will increase significantly. Since there is usually a restriction for the number of polygons, the avatar appearance can be further enhanced by overlapping textures on the mesh as illustrated in Figure 21.c. The process for attaching the image to mesh is called texture mapping (Figure 22), which is one of the most successful techniques used to enhance the visual richness. Even if the texture takes additional memory, it is insignificant compared to the amount of memory needed by the vertices representing the same visual details.

a) b) c)

Figure 21. Avatar Appearance. (a) Avatar defined only with polygons. (b) Avatar with uniform color attributes. (c) Avatar with texture attributes.

The texture can be one, two, or three-dimensional; for example, a 1-D texture can be used to simulate rock strata; a 2-D texture can represent surface; a 3-D texture can represent clouds. For avatar texturing, mostly 2-D textures are used, as illustrated in Figure 22.

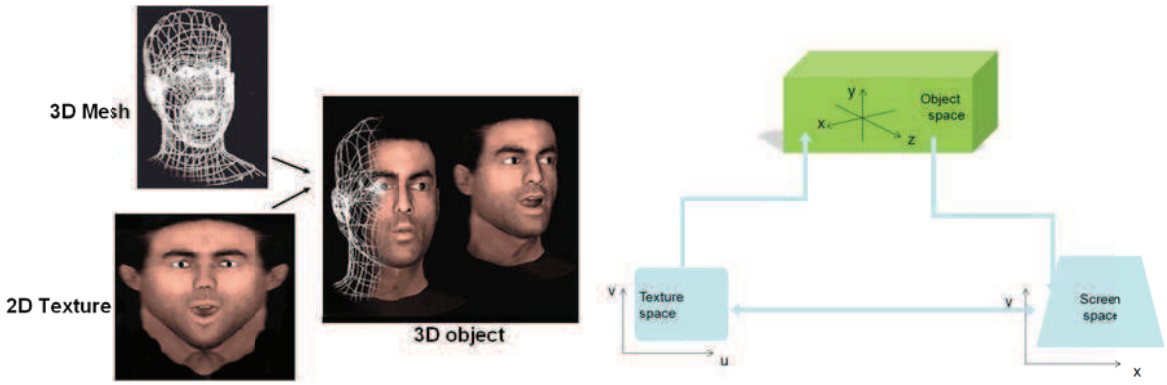


Figure 22. Process of texture mapping to 3D avatar.

The principle of texture mapping, illustrated in Figure 22 is the following. Assuming the texture space labeled as (u, v), the object space labeled as (x_0, y_0, z_0), and the screen

space labeled as (x, y), the objective of the texture mapping is to associate each pixel in the screen space $Src(x, y)$ with the pixel in the texture space $Tex(u, v)$.i.e. to represent:

$$Src(x, y) = f\big(Tex(u, v)\big) \quad \textbf{(1)}$$

Different approaches exist for implementing the texture mapping: screen-scanning, texture-scanning or two-pass [Heckbert86].

While the most current is to use the textures for attaching the color attributes per vertex, other attributes may be updated by using the same approach: surface normal, specularity, transparency, illumination and surface displacement. The diffuse map (the texture is used to update the colors) may be a photograph of a real texture or a digital image created by authoring tools. The number of textures per avatar is not limited, the mesh being separated in several sub-meshes having different textures as illustrated in Figure 23. Here, the soldier model has different textures for the body, the shoulder pads, the head, helmet and the sword. The normal map is a texture used to change the value of the normal at the specific rendered pixel by a technique known as normal mapping. Each pixel of the normal texture presents the offset that is applied on the normal of the rendered pixel, therefore simulating additional details that are not present in the original mesh. This effect can be noticed on the chest armor on the soldier by comparing Figure 23 and Figure 24. The specular map changes the reflectiveness of the mesh at specific points, allowing the same texture for presenting different material types. For example, as illustrated in Figure 25, the iron part of the mesh (the armor) can be made more reflective than the skin part.

Figure 23. Avatar with texture.

Figure 24. Avatar with diffuse and normal texture.

Figure 25. Avatar with diffuse, normal and specular texture.

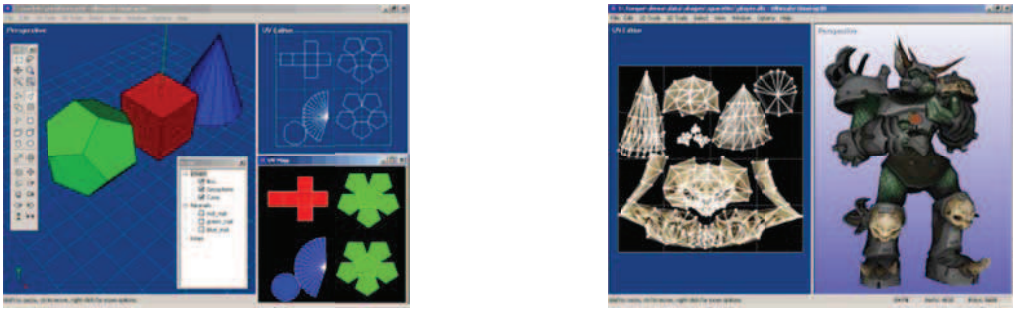
The importance of having a fine object appearance, and particularly a pleasant avatar appearance, is discussed in [Maïm08].

The most frequent means to create a texture is to capture it by a camera and pre-deform it, so that, once applied to the 3D shape, the deformation is compensated by the rendering process. On the other hand, computer graphics systems are able to generate stunningly realistic images of objects. The best-known and most frequently used techniques for that purpose include fractal, spectral, syntax-based, structural-based or stochastic methods.

In order to texture a 3D model, the technique currently used is to transform the mesh by a 3D to 2D projection and represent it in the texture coordinate space (as illustrated in Figure 26). Once the mesh unfolded, by using an interactive 2D tool, the designer attaches the information of texture to each polygon mesh.

Figure 26. 3D to 2D transformation of the 3D mesh.

Currently, there are several tools for projecting the mesh and associate the texture coordinates to it. Among the most widely used are "Texporter"[9] and "Ultimate Unwrap 3D"[10]. The first is a freely available plug-in of 3DSMax, which allows the mesh unfolding. The output is a flat representation in wireframe. By using a 2D traditional tool, the designer can "paint" the color while ensuring the image is aligned with the geometry of the model. Ultimate Unwrap 3D is a Windows application for unfolding 3D meshes. The software contains a UV coordinate editor and has several built-in projections: plane, cube, cylinder and spherical. The graphics user interface of this tool is illustrated in Figure 27. Once the projection is performed, the designer can manually correct the deformations in the textured image.



a) b)

Figure 27. GUI of "Ultimate Unwrap 3D" (a), unfolding of a character from a video game (b).

While the creation of the texture itself can be automated (capture or synthesis), the mapping on the 3D object remains mainly a manual operation.

[9] http://www.cuneytozdas.com/software/3dsmax
[10] http://www.unwrap3d.com/index.aspx

3.1.4 Animation

Avatar animation defines the motion capabilities of the avatar and the possible interactions with the environment.

Animating a character consists in applying deformations at the skin level. The major 3D mesh deformation approaches can be classified into five categories Lattice-based [Maestri99, Liu08, Jiacheng09], Cluster-based [Maestri99, Boulfani08], Spline-based [Bartels87, Mizuno06, Hongzheng07], Morphing-based [Blanz99, Alexa00] and Skeleton-based [Lander99].

The first four categories are used in animating specific objects such as eyes (lattice), and face expressions (morphing), and are more or less supported by the main animation software packages. However, all four have a similar disadvantage: they do not take into account the natural way in which many shapes' features are controlled. Thus, the last category is currently the most used technique for virtual character animation and is described in the next section.

Skeletal animation is a concept that has been often used in the areas of motion pictures, computer games and similar applications to create realistic motion for the animation of articulated characters. This technique for animating a 3D model consists of creating a hierarchical structure, named skeleton, whose rigid motion drives the deformation of the associated model. The location and displacement of the skeleton's joints dictate how the entire avatar moves (Figure 28).

a) b)

Figure 28. Character Skeleton. (a) Initial skeleton pose. (b) Animated skeleton pose.

In previously published papers, avatars motion models were based on the simplified human skeleton with joints. In the early nineties, one of the first challenging methods for creating the human skeleton was published in [Magnenat91], based on a previous research for the hand skeleton [Gourret89]. They observed that existing avatar skeletons were more suitable for robots than for humans, thus a new skeleton layer was proposed. The trend was continued in [Monheit91] with emphasis on providing more realistic effects for the torso, that could be bent or twisted, and in [Scheepers96] for the forearms and hands pronation and supination. A more recent model is detailed in [Savenko99], based on initial investigations reported in [Van98, Van99] where the focus is on improving the joint model and especially the knee kinematics.

Performing realistic deformation is achieved by adding new layers in addition to the skeleton, namely muscle, fat tissue, skin and clothing [Waters89, Scheepers96, Singh95]. In [Scheepers97, Wilhelms97] the muscle layer is linked to the skeleton and is based on the anatomy of skeletal muscles. In [Chen92], a finite-element model is presented, able to simulate the force of few individual muscles. In [Singh95], a skin layer is attached to the skeleton layer, thus more local effects become visible. Particular interests in human skeleton researches increased noticeably with the appearance of 3D

avatars animation where greater realism is needed. Recently, a rotational regression model was proposed to capture common skin deformation such as muscle bulging, twisting, and challenging regions such as the shoulders [Wanget07]. Shi *et al.* [Shi07], proposed to optimize bone transformation and skin-weight with surface-detail-preserving inverse-kinematics constraints during deformation. The new skeleton-based method for deforming meshes is proposed in [Han-Bing08]. They define *simplices* and control them instead of vertices. By doing so, smooth transitions near the joints of the skeleton are achieved and furthermore no vertex weights need be defined on the bones.

To attach the skeleton on a 3D model, an initialization stage is necessary: the designer has to specify the influence region of each bone of the skeleton as well as a measure of influence. This stage is mostly interactive and recursively repeated until the desired animation effects are reached. When the skeleton moves, the new position of the vertex is calculated by multiplying the old position with the weights and matrices of the parent bones. While simple and easy to implement, the technique has some limitations, especially when animating soft body (the elbow problem). To overcome these problems, the basic technique was extended by different researchers. [Lewis00] presented a solution in which they use different poses for the extreme situations where the skeleton animation failed. They save the information of these poses and associate it with the bone. This technique was improved by [Kry02] by using Principal Component Analysis (PCA) to construct an error-optimal displacement basis for representing the potentially large set of pose corrections. The calculation is not done on the entire surface, but it is separated on more influence domains, thus optimizing it for use on graphics hardware. [Wang02] proposed an alternative solution: instead of using one weight for each bone, weights are used for each component of the bone matrix. The weighting is done in a process in which the character is first animated, and then the weights are adjusted only for the problematic poses. [Mohr03] proposed a technique that improves the skeleton driven deformation by automatically adding new joints between existing ones to solve the problems in the extreme poses.

Creating animation can be classified into three categories: kinematic, dynamic and motion capture-based. The first two belong to techniques using content creation tools, while the third is a technique where additional material is used.

3.1.4.1 Kinematics Technique

The kinematic approaches take into account critical parameters related to the virtual character's properties such as its state (position and orientation) and velocity. Two classical concepts for kinematic animation can be chosen: forward kinematics (FK), when the relative geometric transformation of each bone of the skeleton is directly controlled and inverse kinematics (IK), where the orientation of a set of bones is calculated from previously fixed desired positions for specific bones, so-called end-effectors.

3.1.4.2 Dynamic Technique

Dynamic approaches refer to physical properties of the 3D virtual object, such as mass or inertia, and specify how the external and internal forces interact with the object. Such physics-based attributes have been introduced since 1985 in the case of virtual human-like models [Armstrong85, WilhelmS85]. Extensive studies [Badler95, Boston98] on human-like virtual actor dynamics and control models for specific motions (walking, running, jumping, etc.) [Pandy90, Pandy99 and Wooten98] have been carried out. Faloutsos et al. [Faloutsos01] proposed a framework making it possible to exchange controllers (i.e. a set of parameters) to drive a dynamic simulation of the character. The controller evolution is obtained by using the goals of the animation as an objective function. The results are physically plausible motions. Another framework that enables to simulate gravity and contact forces in order to create animation is proposed in [Faloutsos01]. Their approach consists of defining an explicit model of the "pre-conditions" (created both manually and automatically by learning) under which motor

controllers are expected to function properly. Depending on the action on the controllers, different animations can be created. A platform that is dedicated to the development of physically-based controllers for articulated figures, aiming at creating computer animations is presented in [Shapiro05]. Even if some positive steps have been achieved for specific motions, simulating dynamically articulated characters displaying a wide range of motor-skills is still a challenging issue.

In the next sub-section motion capture as a technique to create avatar animation is presented.

3.1.4.3 Motion Capture Technique

The motion capture technique [Menache00] consists in tracking and recording the position (and the orientation) of a set of markers placed on the surface of a real object. Usually, the markers are positioned at the joints. The markers' positions, expressed in the world coordinate system, are then converted into a set of geometric transformations for each joint [Badler93, Hirose98, Molet99].

According to the nature of the sensors used, motion capture technologies are generally classified into active and passive sensor-based capture. With an active sensor-based system, the signals that are to be processed are transmitted by the sensors, while, in a passive sensor-based system, they are acquired by light reflection from the sensors. With respect to the nature of the sensors, the active sensor-based systems can be one of the following: mechanic-, acoustic-, magnetic-, optic- and inertial-based.

One of the earliest methods, using active mechanical sensors [FARO], is a prosthetic system. This is a set of armatures attached all over the performer's body and connected to a series of rotation and linear encoders. Reading the status of all the encoders allows for the analysis of the performer's postures.

The so-called acoustic method [S20sd] is based on a set of sound transmitters attached to the performer's body. They are sequentially triggered to emit a signal and the distances between transmitters and receivers are computed from the time needed for the sound to reach the receivers. The 3D position of the transmitter, and implicitly of the performer's segment, is then computed by using triangulation procedures or phase information.

Systems based on magnetic fields [ATM, Polhemus] are made of one transmitter and several magnetic-sensitive receivers attached to the performer's body. The magnetic field intensity is measured by the receivers so that the location and orientation of each receiver are computed.

More complex active sensors are based on fiber optics. The principle consists in the measurement of the light intensity passing through the flexed fiber optics. Such systems are usually used to equip data-gloves[11].

The last method using active sensors is based on inertial devices, such as accelerometers, small devices which measure the acceleration of the body part they are attached to [Aminian98].

When using active sensors, the performer is burdened with a lot of cables, limiting his motion freedom. In this context, the recent developments of motion capture systems using wireless communication are very promising [ATM, Polhemus].

The second class of motion capture techniques uses passive sensors. One camera, coupled to a set of mirrors properly oriented, or several cameras allow for the 3D posture reconstruction from these multiple 2D views. To reduce the complexity of the analysis,

[11] VPL Research Inc. Dataglove Model 2 Operation Manual, January 1989.

markers (light reflective or LEDs) are attached to the performer's body. The markers are detected on each camera view and the 3D position of each marker is computed. However, occlusions due to the performer's motions may obstruct them. Additional cameras are generally used in order to reduce the loss of information and ambiguities.

Since 1995, computer vision-based motion capture has become an increasingly challenging issue when dealing with the tracking, posture computation and gesture recognition problems in the framework of human motion capture. The techniques can use only one image or sequence of images. [Moeslund06] classifies them in three main branches: model-free, indirect model use and direct model use. Model-free methods have no previous knowledge of the model so they take a bottom-up approach to track and label body parts in 2D. Indirect model methods use look-up table to guide the interpretation of measured data. Direct model methods use previous knowledge of the model and try to use the data to find the model on the image. Included here are learning based methods that use the training of the system with known poses [Agarwal06].

3.2 Compression

The generic compression on top of textual data (achieved through entropy coding of the corresponding text file thanks to "gzip" or similar tools) usually reduces the data size only by a factor 10 [Augeri07], because it is just able to exploit the data structure redundancy, since the information represented by the data is not understood and hence cannot be exploited. Even worse, streaming and scalable coding are out of the question with generic compression methods since both require that the data semantics be understood to build hierarchical or at least progressive representations of the information, which can then be adequately packetized. Instead, specific 3D graphics (lossy) compression techniques may better exploit the spatial or temporal correlation present in the information, and reduce the data size by a factor of over 40 for geometry and 70 for animation, while possibly yielding progressive, scalable bitstreams, suitable for streaming scenarios involving heterogeneous networks and/or terminals.

Not aiming for a complete survey on static/animated mesh compression (several of them existing already in [Avilez08, Peng05]), we only present the major techniques by indicating relevant references for better situating the proposed generic architectural model.

For 3D graphics, two fields particularly retain the attention of researchers: the compression of static meshes and the compression of animated meshes. When referring to static mesh compression, there are two types of data that compose the mesh and should be compressed: geometry and connectivity. For animated mesh compression, the connectivity remains typically constant, and only the geometry is being updated. The geometry describes where the vertices are located and optionally includes additional attributes that characterize them (normals, colors, textures). Such data may be encoded with less precision, resulting in a dramatic reduction of the encoded size. The connectivity state, which describes how the vertices are linked with each other to form triangles or polygons, is information that should be encoded without any loss in precision.

3.2.1 Static Mesh Compression

There are two main approaches to compression for static 3D meshes. In the first one, based on single-rate compression, the data is compressed and decompressed at once. The data cannot be reconstructed until the complete mesh stream is received and decoded. The second method is focused on progressive compression and transmission when a 3D mesh can be decoded and rendered continuously with different Levels of Detail (LODs). The mathematical concepts, the definitions and the theory underlining the mesh compression are very well presented in the literature [Edelsbrunner01, Kahn1995, Gross98].

The best-known approach to specifying mesh connectivity is called Indexed Face Set, consisting of a list of coordinate arrays and one or several lists of face arrays containing indices of the coordinate array. This approach does not include compression and was first standardized in VRML [VRML]. A more rendering-friendly format is the one which attempts to divide a 3D mesh into long strips of triangles. Such a method is also appropriate for compression, exploiting the spatial correlation between neighboring vertices. Based on the work reported in [Deering95], which introduced the concept of the generalized triangular mesh, [Chow97] proposes a sophisticated method for triangle strip encoding, refined further in [Taubin98]. Another manner to exploit the spatial correlation of a mesh consists in decomposing [Bajaj99] the mesh into several concentric layers of vertices. A major property of mesh compression was exploited in [Touma98] and can be formulated as "to encode the connectivity of a mesh, it is enough to know the valence of each vertex". This led to several algorithms on how to encode the vertices valence [Gumhold 98, Gumhold99]. In most cases, the coding methods for geometry are performed with acceptable loss of data, invisible to human perception. To achieve this, quantization techniques [Gersho92] are used, applied globally, on the entire mesh [Deering95, Taubin98, Bajaj99, Touma98], or partially, per region [Chow97]. A vertex position is predicted from one or several of its neighbors by using delta coding [Rossignac99] or linear prediction [Gumhold98, King99] along the vertex ordering imposed by the coding of the connectivity. Being imposed by the connectivity, this mesh traversal is still not optimal for geometry coding, other techniques [Arkin94, Evans96] proposing to use the geometry as a driver of mesh traversal. A noticeable improvement over the scalar quantization and prediction is the use of vector quantization [Lee00, Chou02].

The heterogeneous nature of current devices and networks leads to an increased interest in what is known as progressive compression. Compared with a single-rate coder, it achieves worse results in the sense of compression gain because the use of correlation in mesh data is not supported. For encoding the connectivity, the most relevant are progressive mesh [Hoppe 96, Hoppe98] and progressive forest split [Taubin98, Pajarola00, Pajarola02]. Other methods are patch coloring [Schroeder92, Soucy96, Cohen99], valence driven conquest [Alliez01], embedded coding [Li98] and layered decomposition [Bajaj99, Bajaj96, Gieng98]. For geometry-driven multi-resolution compressions, the main methods are Kd-tree decompositions [Gandoin02], octree decomposition [Jayant84], spectral coding [Karni00], wavelet coding [Khodakovsky00] and geometry image [Gu02].

3.2.2 Animated Mesh Compression

Several methods for animated mesh compression already exist, exploiting both spatial and temporal correlation. In the first one, the mesh is segmented and its motion encoded individually. The segmentation can be performed on surfaces [Lengyel99, Amjoun06] or in volumes [Zhang04, Zhang 05]. A typical group are deformation controllers [Preda04], mostly using an underlying skeleton. The temporal correlation is exploited by its prediction at the vertex level or from a group of vertices [Yang 02, Ibarria03]. A very well-known method consists in Principal Component analysis, used for decomposition of the animation sequence directly in the mesh domain [Alexa00] or in combination with a predictive coding scheme [Karni04, Sattler05, Aviles08]. Based on the geometry image coding introduced in [Gu02], geometry video coding is proposed [Briceño03]. Progressive approaches can also be applied to animated data, examples based on wavelet decomposition being provided in [Guskov04, Payan05] and based on clustering in [Lengyel99].

3.2.3 Animation Compression

High quality motion is usually captured at high frequency (up to 60 frames per second) and motion capture systems produce a large volume of animation data. When such data

is to be used in a computer game, the requirement is to represent as much animation as possible in a limited amount of storage space, thus compression is needed. Compression may also be important in a production environment for easy access to animation assets. In an online scenario it is also important to have compression because of the limited bandwidth available for streaming animation data.

Elaborating models which allow representing animation data in a compact way can be fulfilled by exploiting the redundancy of animation: (i) temporal interpolation is a technique that keeps only the relevant pair of key and key values such as at any moment in time the signal value can be obtained by linear or higher order interpolation and (ii) spatial interpolation refers to clustering vertices and attaching to each one a unique transformation (e.g. skeleton based animation). In addition, to elaborate the animation models, in order to reduce the amount of transmission data, one can use compression on top of animation. In general, very few research works have been conducted in this area. The MPEG group standardized an approach for compression of generic interpolated data [Jang04], able to represent coordinates and normal interpolation. While generic, this approach does not exploit the spatial redundancy. Concerning avatar animation, one of the most used animation contents used in games, MPEG standardized in 1998 a tool named Face and Body Animation (FBA) allowing compression at very low bit-rate, highly inspired from video encoding algorithms. The limitations of this tool [Preda02] consist mainly in the rigid definition of the avatar and the difficulty to set up the proposed deformation model.

Apart from the MPEG solutions, other methods were reported in the literature. Temporal prediction exploiting redundancies between frames was reported by Ibarria and Rossignac [Rossignac 03]. Spatial coherence was investigated by Guskov and Khodakovsky by using wavelets [Guskov04]. Sloan [Sloan03] used PCA to compress animations. Sattler et al. introduced a clustered PCA based approach for compressing mesh animations [Sattler05].

Another method to consider the special correlation of the vertices motion is to identify portions of the mesh that move rigidly, encode the rigid transformation of such clusters and correct them with residuals errors [Leyngyel99], [Gupta02]. Quantization of the motion type is used by Endo et al. [Endo03].

Data transmission scalability is addressed by Hijiri et al. [Hijiri00] by exploiting the 3D scene structure. Chattopadhyay et al [Chattopadhyay05] used quantization to achieve data compression and incorporates intelligent exploitation of the hierarchical structure of the human skeletal model. Recently Arikan [Arikan06] proposed a motion data re-ordering in linearly related clips, followed by markers trajectory approximation by Bezier curves, quantization and PCA decomposition for large databases

In 2004, MPEG-4 standardized an improvement of FBA, called BBA (Bone-based Animation), that allows the representation of any kind of articulated model, offers a high quality animation by using a biomechanical deformation model (simulating a skeleton) and exploits both temporal and spatial redundancy of the animation signal. MPEG-4 defines only the bit-stream syntax; it does not specify the encoding tools. Indeed, it proposes and standardizes some encoding tools, but always gives a possibility to improve the encoding algorithm or even propose new compression tools and still be compliant with the standard.

3.3 Content Adaptation for Weak Terminals

About three decades ago, the first mobile phones were launched in commercial usage as portable devices used for voice communication. The typical phone used to have an input mechanism (simple numeric keypad) and a simple display. In the early 90s, the 2nd generation mobiles were issued, where additional applications, such as the Short Messaging Service (SMS), were introduced. In 2001, the 3rd generation of mobiles was

launched and clearly re-defines the mobile phone in both hardware and software. Today's phones have high-quality displays, easy-to-manipulate keyboards, webcams, speakers, microphones, touch-screens, embedded memory, making them more powerful than some early portable computers. These phones include not only messaging and call support, but also software applications used in everyday life, such as the ones for viewing photos, showing maps, calendar, notes, clock, calculator, compass etc. Most of the phones have applications that can play music and/or video. Services for downloading pictures, games, music, video, jokes and lot of other content exist. New generation phones (iPhone, for example) allow audio conferencing in real time, listening to music in real-time and watching video on-demand. Most of the mobiles also have Internet access through a local Wi-Fi[12], second-generation (2G) wireless data standards (such as *Groupe Spécial Mobile* - GSM or *Enhanced Data rates for GSM Evolution* - EDGE network) or through third-generation (3G) standards (such as *Universal Mobile Telecommunications System* - UMTS, *High-Speed Downlink Packet Access* - HSDPA or *High-Speed Uplink Packet Access* - HSUPA).

Despite the significant progress in hardware development, rendering and animating complex 3D objects, and specifically virtual humans, are still very challenging issues on mobile phones. Networked virtual collaborative environments have been studied for a couple of decades and are now widely present and available on many different platforms, including mobiles. However, the representation of avatars for such devices is relatively simplistic and techniques for content adaptation were proposed in the literature.

3.3.1 Content Adaptation Techniques

Among different content adaptation techniques, one set are attempting to reduce the complexity of the graphics chain. Consequently, in theory the most appropriate solutions are the ones converting 3D data to other media types, such as images, video or 2D graphics. This concept, also known as transmoding, can benefit from existing computer graphics methods such as Image-Based rendering [Chang02]. Here, the 3D geometry is replaced by objects called "impostors", which are nearly semi-transparent textured quads [Maciel95]. The transmoding technique was also investigated in the rendering of 3D virtual characters [Aubel00]. These approaches attach animated textures to impostors in order to reproduce the 3D animation of hierarchically articulated bodies. Transmoding of full 3D scenes to 2D graphics is also reported in the literature [Giacomo04].

The transmoding is usually performed off-line, only a reduced number of views being synthesized. A relatively recent trend is to perform the transmoding in real-time, on dedicated servers and transmit the results as a video stream. In this case, the mobile devices are used only as display. For complex scenes, clusters of PCs can be used to distribute the rendering process [Lamberti03]. The main drawback of the remote rendering approaches is the increased demand of network bandwidth. Furthermore, when used for applications such as VWs, where the number of simultaneous active users is high, the requested processing power of the server may become an issue.

Adapting the resolution of the 3D graphics content with respect to the characteristics of lightweight devices was also reported in the literature. Several file formats such as *pod*, *md2*, *m3g, mp4* are specifically designed for delivering 3D graphics content. Among them, *m3g* is the most used due to Java3D which was for a long time the unique manner to deal with 3D graphics on mobiles. However, *m3g* has two main drawbacks:

- m3g does not propose any techniques for content adaptation.
- m3g only supports zip compression, which may not be enough for reducing the amount of data to be transmitted.

[12] http://www.wi-fi.org/wp/wifi-alliance-certification/

In order to be generic with respect to hardware conditions, a very attractive solution is the automatic content adaptation according to given contexts. Several approaches provide generic schemes for scalable 3D models. In the next section we present the state of the art on this topic.

3.3.2 Avatar Adaptation Techniques

Multi-resolution meshes suitable for adaptation have been proposed in the MPEG-21 Digital Item Adaptation framework [MPEG-21]. Furthermore, in addition to scalable representation, adaptation of 3D animation, and especially avatar animation, has been explored [Giacomo04]. While these methods reduce the complexity of graphics assets, they still do not address the issue of compression.

One of the most current representation techniques for avatar geometry is mesh-based. Simplifying the mesh consists mainly in vertex reduction and several algorithms are proposed in the literature. In general, all simplification algorithms can be classified in three groups [Garland97], as briefly described below.

The first group is the *Vertex Decimation* which iteratively selects a vertex for removal, removes all adjacent faces, and triangulates the resulting holes. The second is the *Vertex Clustering* technique, when several vertices sharing a same volumetric cell are clustered together into a single vertex, and the model faces are updated accordingly. The third group, *Iterative Edge Contraction,* can be considered as a compromise solution between the previous two. By merging only 2 vertices at a time, *i.e.* collapsing one edge, the algorithm still inherits the speed of the *Vertex Clustering* but also improves the robustness of the *Vertex Decimation* technique.

One of the methods of the last group is the surface simplification based on quadric error metric (QEM) [Garland97]. It consists in an iterative contraction of vertex pairs. It differs most substantially from other related methods in the way of choosing the edge to contract. By the introduction of the quadric error metric Q, which closely interprets the surface curvature of a 3D model, the algorithm is claimed to be a fast and high quality one [Garland99]. When an edge is decided to be contracted, two primary policies for choosing the target position can be considered. One is *Subset Placement* - one of the endpoints of the origin edge, which has smaller Q — will be the combined vertex. The second is *Optimal Placement, which* can produce better approximations by trying to find the point so that it minimizes the local function Q constructed by the edge in question.

Related to avatars, special simplification criteria should be considered. Due to the fact that the mesh evolves in time and therefore the QEM also evolves in time, simplification on vertices cannot be considered only in the static position. One approach would be to average QEM for all animation frames, an operation that is very time consuming. An alternative is to exploit the skinning information (i.e. the relation between the vertices and the skeleton). In [Preda05], a new method of mesh simplification which is the adaptation of QEM to the model animated with the assistance of the skeleton is proposed. A new parameter is added for describing the link between the bone and vertex, creating new rules in respect to which a vertex can be removed. Two main approaches are proposed:

- bone-based constraints, in which the number of vertices that can be removed from one bone is limited therefore ensuring that all bones are kept and
- bone-based removal, a process that can be controlled by a truncated model or by a removal cost. This approach also allows skeleton adaptation and implicitly animation adaptation. A reduced number of bones imply a smaller number of animated parameters, therefore, a smaller amount of data that should be processed and transmitted for each frame.

Other than the geometry and the animation, the appearances, and mainly the image textures, are relatively big components. Image simplification consisting mainly in

resolution reduction has benefits in both the reduction of transmission size but also the required run-time memory. Moreover, on devices with small screens, such as mobile phones, there is no interest in preserving large resolution for images since the latter are anyway resized by the rendering process.

Different techniques for texture simplification are reported in the literature. For example, in [Okuda06], an effective technique for texture size reduction is presented. The simplification process consists of selecting textures that can be segmented by straight lines and simplifying them using the Hough transform. The other textures are filtered by the anisotropic diffusion and then the details are removed to reduce their data sizes. A set of representative colors are selected from the original image and the image is quantized based on those colors. Experimental results show that the simplified image, while preserving all major features of the original image, has a much smaller size than the original. The main drawback of this algorithm is that it can be used effectively only for images with straight lines or edges, like textures for representing buildings, houses and urban environment in general. Another technique presented in [Cheng03] proposes different resolutions on different parts of texture, depending of the amount of local details of the model. While the method provides good visual results, it can become time consuming. Based on a callable decomposition of wavelet coefficients, the JPEG2000 standard supports natively progressive decoding with respect to image resolution. Such an approach is appropriate for decoding on mobile devices, the client being able to decide on the level (e.g. resolution) for decoding an image, based on the memory constrains.

3.4 Interoperability

Other than the research community, the avatars interested different standardization groups mainly due to the big potential of applications involving them[13]. There are currently two types of such standards: the ones interested in the appearance and the animation of the avatar in the 3D graphics applications (the avatars as representation objects) and the ones interested in avatars characteristics such as personality, emotions, … (the avatars as agents). In addition, there are several proprietary formats, imposed as de facto standards by the authoring tools or virtual world providers. All this multitude of solutions makes it impossible today to imagine even the simple scenario of using a single avatar for visiting two different virtual worlds. In this section we briefly introduce some of the standards from each category and give the main motivation behind MPEG-V.

3.4.1 Standards and Formalisms

In the last decade, several efforts have been made to develop a unique data format for 3D graphics. In the category of open standards, X3D [X3D] (based on VRML [VRML]) and COLLADA [COLLADA] are the best known, the latter being the most adopted by current tools. While COLLADA concentrates on representing 3D objects or scenes, X3D pushes the standardization further by addressing user interaction and application behavior as well. This is performed thanks to an event model in which scripts, possibly external to the file containing the 3D scene, may be used to control the behavior of its objects. While the avatars in VRML/X3D are defined as specific objects being standardized under the name of H-Anim, in COLLADA there is no distinction between a human avatar and a generic skinned model. Also in the category of open standards, but specifically treating the compression of media objects, there is MPEG-4.

VRML (Virtual Reality Modelling Language) was designed with a view to allowing the distribution of 3D objects over Internet, as HTML does it for text. Its first version was specified in 1994. In 1997, a new version of the format was finalized, known as VRML97

[13] Gartner says 80 Percent of Active Internet Users will have A "Second Life" in the Virtual World by the End of 2011, http://www.gartner.com/it/page.jsp?id=503861.

[VRML]. VRML has been superseded by X3D [X3D] in 2005, which specifies as well a version for fast transfer through networks called X3Db. VRML/X3D allows creating files by using an addition mechanism that uses hyperlinks to refer to an external file, reutilizing data by referencing mechanism (DEF, USE), reutilizing complex content by macro definitions (PROTO, EXTERNPROTO). It pushes the standardization further by addressing user interaction as well. This is performed thanks to an event model in which scripts, possibly external to the file containing the 3D scene, may be used to control the behavior of its objects.

The avatars in VRML/X3D are defined as specific objects being standardized under the name of H-Anim [HAnim]. H-Anim specifies a standard way of representing humanoids in a network-enabled 3D graphics and multimedia environment. It identifies a standardized humanoid skeletal system for characters structured for animation. Additionally, it specifies the semantics of avatar animation as an abstract functional behavior of time-based, interactive 3D, multimedia articulated characters. It does not define physical shapes for such characters but does specify how such characters can be structured for animation.

COLLAborative Design Activity [COLLADA] aims to establish an interchange file format for interactive 3D applications focusing on representing 3D objects and scenes. The COLLADA Schema supports all the features that modern 3D interactive applications need. COLLADA is based on XML, making it an easy-readable textual format, allowing users to extend it for their own specific purposes. Defined as open file format, COLLADA is already supported by numerous tools as Maya, 3DS Max, Softimage XSI and Blender. Game engines, such as Unreal engine, and Google Earth, have also implemented this format. However, being highly extensive and complete makes it inappropriate for devices with constraints such as mobile phones, mainly due to high memory requirements. Furthermore, not providing a binary representation limits its usage only as interchange file format between content creation tools, and less as a delivery format.

In COLLADA there is no distinction between a human avatar and a generic skinned model, therefore there is no separate scheme for modeling three dimensional human figures. The general approach adopted is based on defining 3D objects once and using object-containers and instancing them when used. However, all necessary elements to define an avatar, such as support for object-graph, geometry, appearance and animation, exist.

Built on top of VRML, MPEG-4 contained, already in its first two versions [ISO/IEC 14496-1], tools for the compression and streaming of 3D graphics assets, enabling to describe compactly the geometry and appearance of generic, but static objects, and also the animation of human-like characters. Since then, MPEG has kept working on improving its 3D graphics compression toolset and published two editions of MPEG 4 Part 16, AFX (Animation Framework eXtension) [ISO/IEC 14496-16] which addresses the requirements above within a unified and generic framework and provides many more tools to compress more efficiently more generic textured, animated 3D objects. In particular, AFX contains several technologies for the efficient streaming of compressed multi-textured polygonal 3D meshes that can be easily and flexibly animated thanks to the BBA (Bone-Based Animation) toolset, making it possible to represent and animate all kind of avatars.

While offering a full set of features allowing to display the avatars, none of the above-mentioned standards includes semantic data related to the avatar.

Several recommendations, standards or markup languages are related to adding semantics on top of virtual characters, mainly to describe features that do not necessarily have a visual representation (such as personality or emotions) or to expose properties that may be used by an agent (language skills, communication modality, …).

The Human Markup Language (HumanML) [Brooks02] by Oasis Web Services is an attempt to codify the characteristics that define human physical description, emotion,

action, and culture through the mechanisms of XML, RDF and other appropriate schemas. HumanML is intended to provide a basic framework for a number of endeavors, including (but, as with human existence itself, hardly limited to) the creation of standardized profiling systems for various applications. It builds a framework for describing the emotional state and response of both people and avatars, laying the foundation for the interpretation of gestures for both person-to-person and person-to-computer interpretations, the encoding of gestures and expressions to facilitate the better understanding of modes of communication.

EmotionML (EML) by W3C covers three classes of applications: manual annotation of material involving emotionality, such as annotation of videos, of speech recordings, of faces, of texts, etc; automatic recognition of emotions from sensors, including physiological sensors, speech recordings, facial expressions, etc., as well as from multi-modal combinations of sensors; generation of emotion-related system responses, which may involve reasoning about the emotional implications of events, emotional prosody in synthetic speech, facial expressions and gestures of embodied agents or robots, the choice of music and colors of lighting in a room, etc.

Behaviour Markup Language (BML) [Vilhjalmsson07] is an XML-based language that can be embedded in a larger XML message or document simply by starting a <bml> block and filling it with behaviors that should be realized by an animated agent. The possible behavior elements include coordination of speech, gesture, gaze, head, body, torso face, legs, lips movement, and a wait behavior.

Multimodal Presentation Markup Language (MPML) [Ishizuka00] is a script language that facilitates the creation and distributing of multimodal contents with character presenter. It also supports media synchronization with character agents' actions and voice commands that conforms to SMIL specification.

Virtual Human Markup Language (VHML) is designed to accommodate the various aspects of Human-Computer Interaction with regards to Facial Animation, Body Animation, Dialogue Manager interaction, Text-to-Speech production, Emotional Representation plus Hyper- and Multi-Media information.

Character Mark-up Language (CML) [Arafa03] is an XML-based character attribute definition and animation scripting language designed to aid in the rapid incorporation of life-like characters/agents into online applications or virtual worlds. This multi-modal scripting language is designed to be easily understandable by human animators and easily generated by a software process such as software agents. CML is constructed based jointly on motion and multi-modal capabilities of virtual life-like figures.

3.4.2 MPEG-4 XMT and BIFS

MPEG-4 falls into the category of open media standards with the main difference from its predecessors, MPEG-1 and 2, being the fact that it deals with scene graph and graphics primitives in addition to video and audio. Built on top of VRML, MPEG-4 (published in its first version in 1999 [ISO/IEC 14496-1]) specifies a format called BIFS (BInary Format for Scene) dedicated to binarization and streaming of graphics assets. XMT (Extensible MPEG-4 Textual Format),[ISO/IEC 14496-11] is defined as an XML-based container of MPEG-4 data. XMT content has two levels of abstractions: high-level XMT-Ω dedicated more to content exchange between authoring tools; and low-level XMT-A, which is a textual representation of BIFS.

Since then, MPEG has kept working on improving its 3D graphics compression toolset and published three editions of MPEG-4 Part 16, AFX (Animation Framework eXtension) ,[ISO/IEC 14496-16], which addresses the requirements above within a unified and generic framework and provides many more tools to compress more efficiently more generic textured, animated 3D objects. In particular, AFX contains several technologies for the efficient streaming of compressed multi-textured polygonal 3D meshes that can

be easily and flexibly animated thanks to the BBA (Bone-Based Animation) toolset, making it possible to represent and animate all kinds of avatars.

Table 2 summarizes the similarities and differences between the previously described technologies for avatar representation. Let us observe that MPEG-4 is the most complete standard with respect to compression features.

	VRML/X3D	COLLADA	XMT/BIFS/MPEG-4
Object Graph	nodes for mesh, appearance, animation; dedicated avatar graph in H-Anim	support for avatar as a regular 3D object with mesh, appearance, animation, skeleton, physics	nodes for mesh, appearance, animation, morph and dedicated nodes for avatars such as skeleton and muscle
Graph Compression	Zip	Zip	BIFS
Geometry	Mesh, NURBS	Mesh, NURBS	Mesh, NURBS
Geometry Compression	Zip	Zip	BIFS, WSS, 3DMC, SC3GMC
Appearance	Color, texture	Color, texture	Color, texture
Appearance Compression	Image compression	Image compression	Image compression
Animation	Key frame animation by linear interpolators	Key frame animation, interpolators	Animator node in BIFS, BIFS-Anim; BIFS Commands; interpolators; face and body animation; skeletal and morph animation;
Animation Compression	Zip	Zip	BIFS-Anim, Coordinate, Orientation, Position Interpolators compression, Face and Body Animation, Bone-base animation, Frame-based Animation Compression
Animation streaming	No	No	Yes

Table 2. Summary on several features of VRML/X3D, COLLADA and MPEG-4 standards.

3.5 Conclusion

Conducted by the importance of avatars in 3D Virtual Worlds and intending to represent them in a most suitable way, we performed surveys on their definition, creation methods and representation formalisms. We observed that an avatar is currently represented as a composition of four components: object-graph, geometry, appearance and animation. However, this data is not able to provide a high-level description concerning its usage conditions, and therefore we extended the definition by adding a fifth element, metadata.

The first element, the object-graph, represents a container of elements that compose the avatar. The object-graph creation remains a manual task, performed by using content creation tools. The representation form of the second element, the geometry, depends on the following requirements: capabilities of smooth representation, rendering complexity and memory usage. The polygonal mesh, which is represented as face-vertex data structure, is the most suitable due to its simplicity, enabling optimal real-time visualization. We presented several tools and approaches for creating the avatar geometry including sketching tools, capturing real objects, vision-based capture and measurements-based modeling. The third component of the avatar, the appearance, is used to improve the avatar's visual perception and consists in applying (mapping) an image on the mesh, a process called texturing. The memory that it requires is insignificant compared to the amount of memory needed by vertices representing the same visual details. Appearance creation in general is based on real image capturing; however, synthetic images are also used. The fourth element, the animation, defines the motion capabilities of the avatar and the possible interactions with the environment. We analyzed several techniques for animation representation. The skeleton-based representation is one of the most suitable animation techniques, taking into account the natural way in which many shapes' features are controlled. We classified the techniques for animation creation into three categories: kinematic, dynamic and motion capture-based. Despite the fact that a lot of progress was done to capture and manipulate the animation parameters, animation is a difficult process, requiring artistic skills, time and sometimes expensive tools.

The compression and content adaptation were investigated, focusing on techniques relevant for avatar representations. Two main groups of techniques were analyzed: compression of 3D meshes and compression of animation. For the compression of 3D graphics meshes, we observed that two fields retained the attention of researchers: the compression of static meshes and the compression of animated meshes. For the animation parameters compression the analysis was made on the most popular methods.

We observed that the adaptation techniques can be separated in two main groups: transmoding – which can be performed offline or online – and resolution adaptation. The analysis showed that both of these techniques have disadvantages that make them inappropriate for many applications. In the first one, the quality of media is significantly lost, and furthermore when used in online mode the network bandwidth is significantly increased. The second technique consists of preparing different 3D content for different client platforms. None of the techniques presented in the second group support the automatization of the process of content adaptation, which makes the process of content adaptation difficult. This survey helped us identify the gaps in the existing techniques in order to build a better adaption model appropriate for avatars.

As a base for defining the metadata framework we analyzed different standards defining avatars. We observed that the currently existing standards belong to one of the following groups: (1) standards representing the avatar, such as COLLADA, VRML/X3D and MPEG-4, consisting of a full set of features allowing to display the avatar and (2) standards for the semantic representation for avatars, such as HumanML, EmotionML, BML, MPML, VHML and CML, which include semantics on top of the virtual character, such as his personality, emotion, language skills, communication modality etc. We concluded that none of the standards in the two groups provides a full set of features that allows interoperability of the avatars between Virtual Worlds, nor to third party software, and this motivated us to propose such a schema.

4 Contributions to Avatar Compression, Adaptation and Interoperability

Virtual Worlds (VWs), and especially the ones using 3D graphics content, have been massively developed and deployed over the last few years. Technological developments, both in hardware and software, bring up significant improvements in the areas of display technology, graphics, animation, loosely distributed systems, network and mobile phones technology. These improvements allow almost every user on Web 2.0 to have a representation in different VWs. In those VWs, avatars have a specific place representing a graphical appearance of human beings and container for the users' specific data. Therefore, the need to reuse them across different VWs and across different platforms increases significantly.

In this chapter we are presenting a framework that allows interoperability of avatar characteristics between VWs and also the visualization of the complete avatar in stand-alone players, which can be executed on heterogeneous terminals. In order to achieve these goals we are proposing a compression data model for heterogeneous structures, a method for avatar adaptation for weak terminals and a framework for avatar characteristics. Two of these contributions were submitted for standardization within the MPEG committee and are now parts of MPEG-4 and respectively MPEG-V standards.

4.1 Introduction

The advancements achieved in the last two decades in the field of avatars with respect to their visual appearance (static and animated) makes it possible to imagine today an integrated representation solution aiming to ensure one of the main requirements of interoperability between virtual worlds: being able to migrate from one world to another while maintaining the user properties. Another major progress over the years was made in mobile phone technology. Originally a device for voice communication, modern mobile phones have become an essential utility tool, capable to expose several different functionalities, among which ones that open a room for the possibility of representing a personalized avatar. In addition, with the current technological advancements in mobile communication networks, high speed data services based on Internet Protocol (IP) as well as greater capacity and efficiency are offered. The 3G platform provides converged voice, data, Internet access and multimedia services supported by high data rates, redefining the way of communication between people. Representing complex contents such as avatars on mobile phones becomes possible. However, the 3D graphics applications are more and more complex, large VWs being built nowadays.

It is expected that future VWs will use subsets of user properties; probably all of them will use appearance and animation properties to ensure the visual representation. Some properties/capabilities obtained in one VW remain connected to the avatar and should be also available to the outside worlds (real or virtual).

Several VWs allow exporting the entire avatar. While the possibility to import this fully exported avatar to another VW is desired, the VW business model is very often built in such a way that it forbids importing content since it is considered as a valuable asset [Castronova02, Castronova03] and the VW provider prefers to have full access to the content level. An interesting scenario remains the possibility to visualize the avatar in external players. This, and particularly if the player is built in another platform, requires also the adaptation of geometry, appearance and animation [Francisco07].

Considering the previous analysis, we propose a workflow that allows "teleporting" avatar features from one VW to another and also the "transfer" of a complete avatar in external players, executed on heterogeneous terminals.

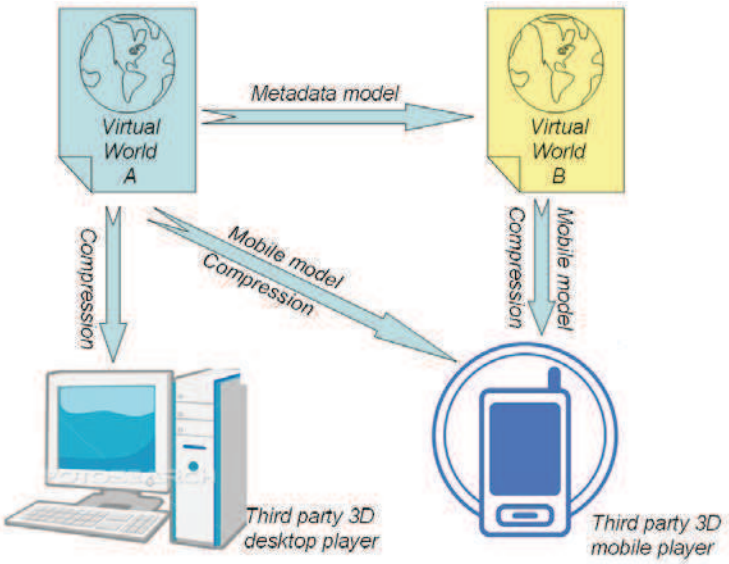


Figure 29. A workflow for avatar transfer through VWs and platforms.

Figure 29 represents the functionality of such a workflow: the avatar is created in one VW and it is represented in another one, by transferring its semantic characteristics,

while also being possible to represent it in outside players, by transferring its full characteristics.

The framework is implemented by three original modules:

- **A compression data model** allowing the compression of an arbitrary avatar, defined by using any XML schema.
- **An adaptation method** of avatars as extracted from the VW or created with the authoring tool. The method takes into consideration the reduced capacities of the mobile phone.
- **An interoperability framework** defining the set of avatar characteristics able to provide a semantic description of avatars needed to personalize VW templates.

In the following sections we present each of the three modules

4.2 Compression Data Model for Heterogeneous Structures

The first four components defining an avatar are very different with respect to the nature of the signal and dedicated compression tools should be applied. The MPEG-4 standard already specifies such techniques; however, they can be used only if the avatar is represented by using the object graph structure of MPEG. In this section we propose a generic compression model able to merge the performing MPEG-4 compression for geometry and animation with state-of-the-art compression for object graph and appearance.

4.2.1 Compression Techniques for Homogenous Data

The MPEG-4 compression tool-set is relatively rich mainly due to the different representation forms that 3D data may take, and their development is still an on-going process. MPEG-4 contains different tools for compressing different data types: scene-graph, geometry, appearance and animation. Table 3 shows the list of those tools classified by type.

Compression Tool	Type
Binary Format for Scenes	Scene, Geometry & Animation
3D Mesh Compression	Geometry
Wavelet Subdivision Surface	Geometry
Scalable Complexity 3D Mesh Coding	Geometry
Foot Print	Geometry
Coordinate, Orientation and Position Interpolator	Animation
Bone-based Animation	Animation
Frame-based Animated Mesh Compression	Animation
Octree Compression for Depth Image-based Representation	Appearance
Point Texture	Appearance

Table 3. MPEG-4 tools for 3D Graphics Compression.

Since 1999, when the first version of MPEG-4 was published, the MPEG committee has had a continuous activity on compressing 3D graphics assets and scenes. MPEG-4 offers a rich set of tools that may be classified with respect to data type. The most generic, called BIFS, allows compressing a scene graph. The downside of being a generic tool,

BIFS cannot completely exploit the redundancy of specific data such as meshes or animation curves.

To overcome this limitation, MPEG-4 defines methods for each data type. Tools for mesh compression comprise 3D Mesh Coding (3DMC), Wavelet Subdivision Surface (WSS), FootPrint (FP) and Scalable Complexity 3D Mesh Compression (SC3DMC). For animation, the following tools are standardized: Coordinate Interpolator, Orientation Interpolator and Position Interpolator (CI, OI, PI), Bone-Based Animation (BBA) and Frame-based Animated Mesh Compression (FAMC). To encode the appearance, MPEG-4 natively supports PNG, JPEG and JPEG2000 data compression.

In the following we introduce the key elements for scene, geometry and animation compression tools. We restrict our presentation to the tools that may be used for the compression of avatars.

- **BInary Format for Scenes (BIFS) Compression**

BIFS is the binary representation of an extended set of VRML nodes. Its major advantage is that it is designed as a generic tool for scene-graph, geometry and animation binarization/compression. It contains an enriched set of 2D and 3D graphics primitives and, most important, includes mechanisms for streaming graphical content. It attempts to balance the compression performances with the extensibility, ease of parsing and simple bitstream syntax. It includes the traditional modules such as prediction, quantization and entropy coding. However, for 3D graphics primitives and animation, BIFS does not fully exploit the spatial and, respectively, temporal prediction of (animated) 3D objects. To overcome this limitation, MPEG defined specific tools for such data and grouped them in Part 16 of the MPEG-4 standard.

- **3D Mesh Compression (3DMC)**

3DMC, initially published in 1999 (ISO, 1999) and extended in 2007 (ISO, 2007) is based on the Topological Surgery (TS) representation [Taubin98]. It applies on 3D meshes defined as an indexed list of polygons and consists of geometry, topology and properties (e.g. color, normal, texture coordinate and other attributes).

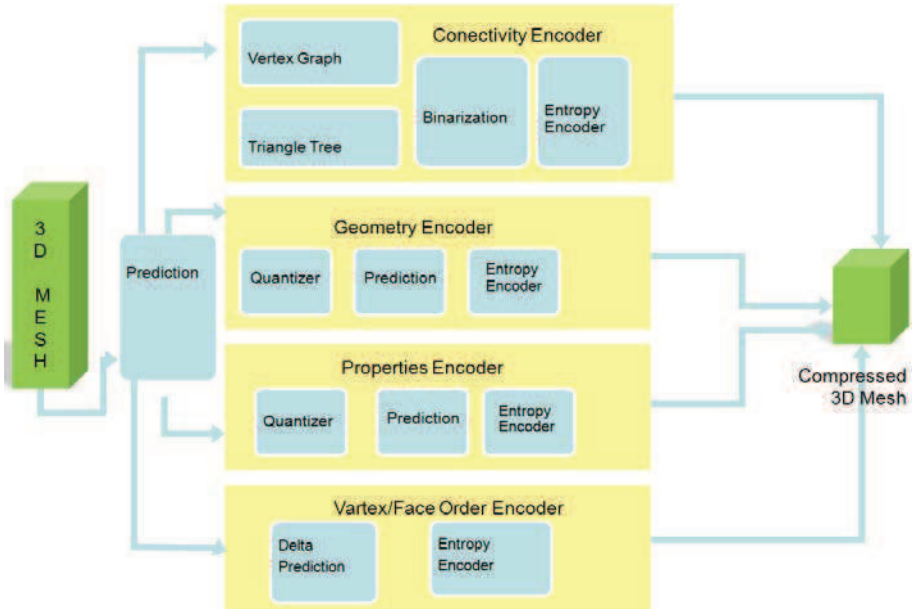


Figure 30. General block diagram of an MPEG-4 3DMC Encoder.

Encoding of connectivity information is lossless, while the other information could be quantized before compression. Geometry and properties information are encoded in a similar manner. 3DMC supports three modes of compressing the mesh: single resolution, when the entire mesh is encoded as indivisible data, incremental representation, when

data is interleaved such as each triangle may be rendered immediately after decoding, and the hierarchical mode, when an initial approximation of the mesh is improved by additional decoding of the details. The extension published in 2006 allows preserving vertex and/or triangle order and improves the compression of the texture mapping information.

- **Scalable Complexity 3D Mesh Coding (SC3GMC)**

SC3GMC (ISO/IEC 14496-16 AMD 4) includes several encoding methods for the connectivity encoding and traditional geometry and attributes encoding. The encoding schema is illustrated in Figure 31.

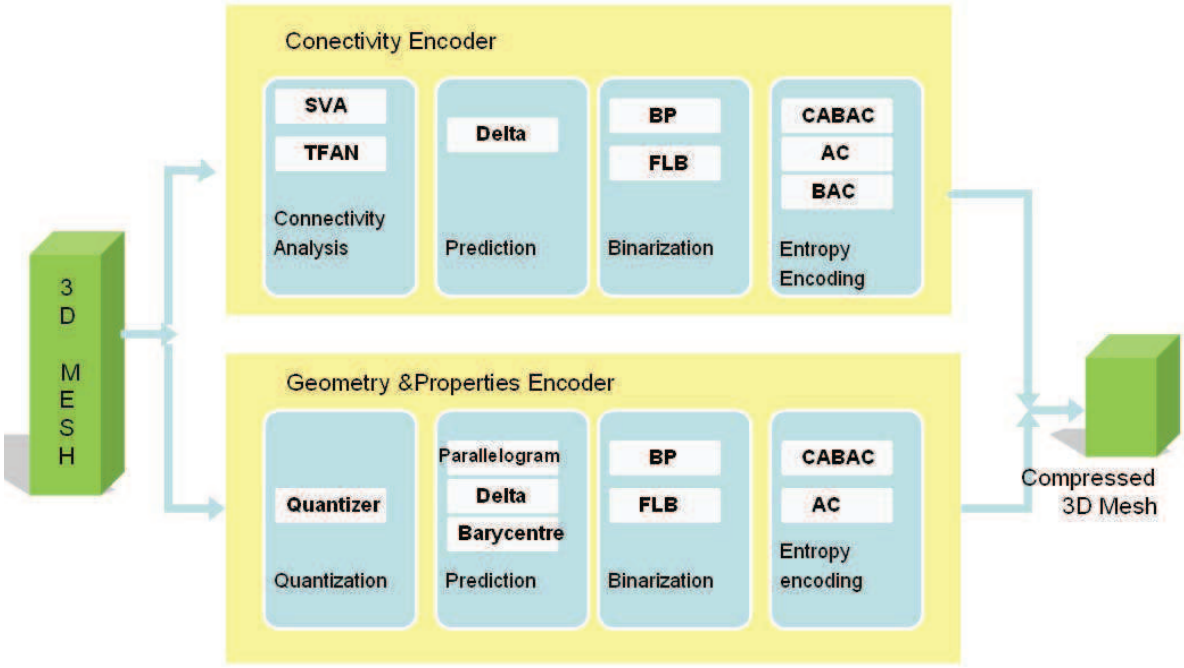


Figure 31. 3DMesh encoding process.

Unique for SC3DMC is that by selecting the appropriate encoding tools, the complexity of the required decoder can be controlled. In the simplest schema, no prediction between vertex positions or attributes is included. The intermediate one, called Shared Vertex Analysis (SVA), exploits a correlation between a previous face and a current face to remove an inherent redundancy within connectivity information. The most complex one, called Triangle Fan Analysis (TFAN) is based on the partitioning of the mesh triangles into a set of triangle fans. For all three, geometry encoding includes quantization. The other attributes such as normal, color, texture coordinate are encoded based on the prediction obtained from the geometry. Finally, for entropy encoding, different methods are used, such as Binary Arithmetic Coding and Bit Precision Coding.

- **Wavelet Subdivision Surface Streams (WSS)**

Wavelet methods for geometry encoding are a superset of multi-resolution analysis, which has proven to be very efficient in terms of compression and adaptive transmission of 3D data. The de-correlating power and space/scale localization of wavelets enable efficient compression of arbitrary meshes as well as progressive and local reconstruction. Especially for signal transmission purposes, sub-band decomposition is a very important concept. Not only does it permit to send a coarse version of the signal first and progressively refine it afterwards, but also enables a more compact coding of the information carried by signals whose energy is mostly concentrated in their low-frequency part. WSS of MPEG-4 is a hierarchical compression tool that uses a list of indexed triangles as a base mesh and encodes the vertex positions at different resolution levels based on subdivision surface predictors. The WSS contains only corrections of

vertex position prediction encoded by using SPIHT (Set Partitioning In Hierarchical Trees) technique (Said, 1996). For all the other attributes defined per vertex (normals, colors, texture coordinates …), linear interpolation schemes are used. While it has proven to be very efficient in terms of compression and adaptive transmission of three-dimensional (3D) data, it is mainly suited for applications such as terrain navigation. The schema of a WSS encoder is illustrated in Figure 32.

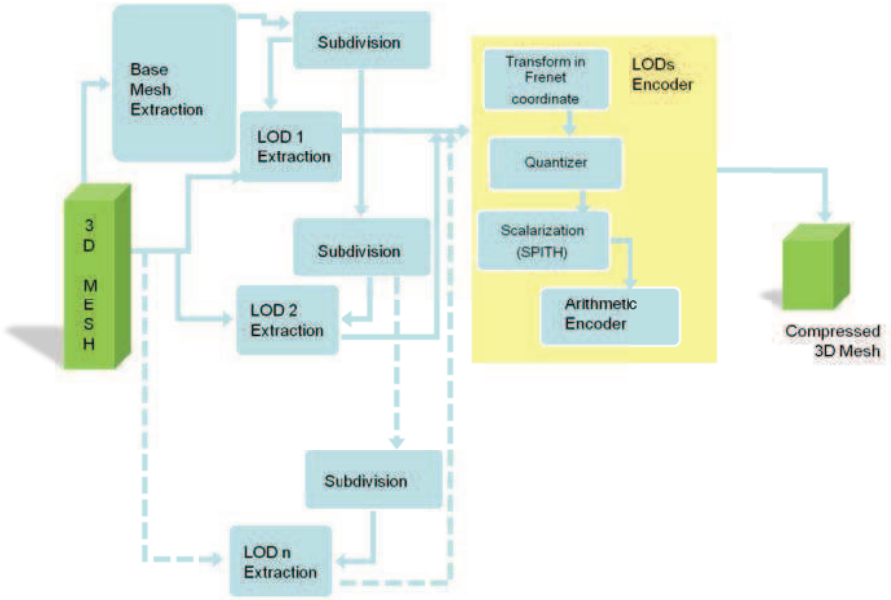


Figure 32. General block diagram of an MPEG-4 WSS Encoder. (LOD - Level of Details)

- **Coordinate Interpolator, Orientation Interpolator and Position Interpolator Streams (CI, OI and PI)**

Interpolator representation in key-frame animation is currently the most popular method for computer animation. The interpolator data consist of key and key value pairs, where a key is a time stamp and a key value is the corresponding value to the key. Depending on the data type (coordinate, orientation or position), the key value may have different dimensions. The key value for coordinate and position is the set of X, Y and Z (dimension 3); while orientation is represented as an axis and an angle (dimension 4).

The structure of the Interpolator Compression (IC) is illustrated in Figure 33. First, the original interpolator data is analyzed and may be reduced by removing the redundant or less meaningful keys from the original set. A redundant key is one that can be obtained from its neighbors by interpolation and a less meaningful key is one for which the distortion between the original signal and the one reconstructed by interpolation is below a threshold.

Once the important keys are selected, the (key, key value) pair data are separated and processed by dedicated encoders. The key data, an array with no decreasing values, is first quantized and a delta prediction is applied. The result is further processed by a set of shift, fold, and divide operations with the goal of reducing the signal range (Jang, 2004).

Data such as the number of keys and quantization parameters are encoded by a header encoder using dictionaries. Concerning the key values, the data is first normalized within a bounding box and then uniformly quantified. These values are derived from one or two already transmitted key-values; the prediction errors are arithmetically encoded.

By using this method for representing interpolators, the compression performances are up to 30:1 with respect to the textual representation of the same data.

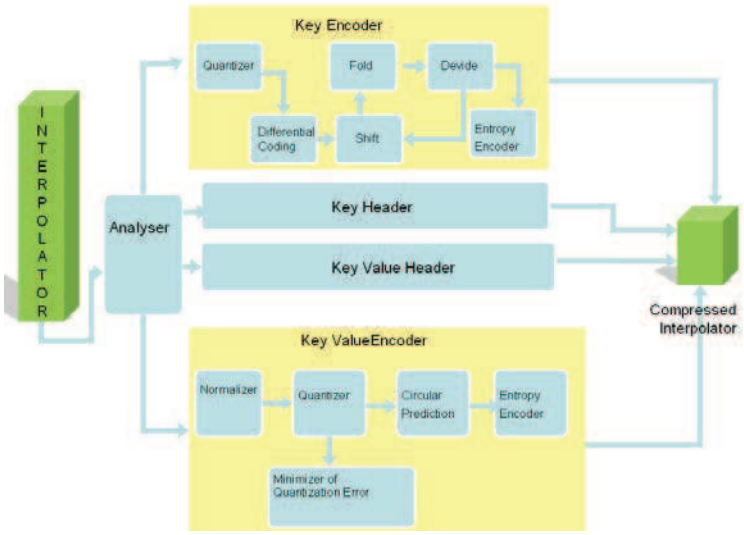


Figure 33. Interpolators Encoder.

- **Bone-Based Animation Stream (BBA)**

In order to represent the animation data compactly, vertices are clustered and a unique value or transform is assigned to each cluster. For avatar animation, MPEG published BBA (Preda, 2004) which is a compression tool for the geometric transforms of bones (used in skinning-based animation) and weights (used in morphing animation). These elements are defined in the scene graph and should be uniquely identified. The BBA stream refers to these identifiers and contains the new transforms of bones (expressed as Euler angles or quaternion) and the new weights of the morph targets. BBA follows traditional signal compression schema by including two encoding methods as illustrated in Figure 34.

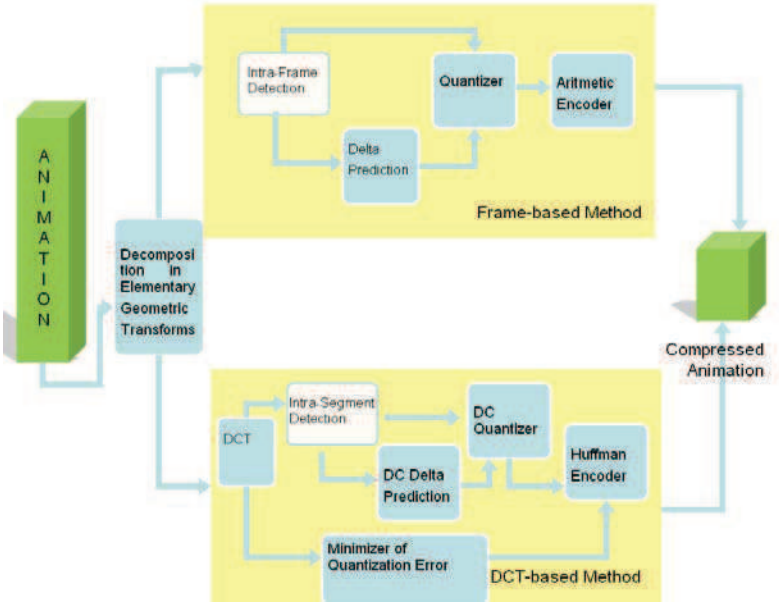


Figure 34. BBA Encoder. (DC means Discreet Coefficient, AC means Alternative Coefficients).

- **Frame-based Animated Mesh Compression Stream (FAMC)**

FAMC is a tool to compress an animated mesh by encoding on a time basis the attributes (position, normals …) of vertices composing a mesh. FAMC is independent from the manner in which the animation is obtained (deformation or rigid motion). The data in a

FAMC stream is structured into segments of several frames that can be decoded individually. Within a segment, a temporal prediction model, used for motion compensation, is represented.

Each decoded animation frame updates the geometry and possibly the attributes of the 3D graphic object that FAMC refers to. Once the mesh is segmented with respect to the motion of each cluster, three kinds of data are obtained and encoded as illustrated in Figure 35. First, the mesh partitioning indicates for each vertex the attachment to one or several clusters. Secondly, for each cluster an affine transform is encoded by following a traditional approach (frequency transform, quantization, prediction of a subset of the spectral coefficients and arithmetic encoder). Last, for each vertex a residual error is encoded. The frequency transform may be chosen between DCT and Wavelet (LIFT) Schema and the prediction may be delta prediction or one based on multi-layer decomposition of the mesh.

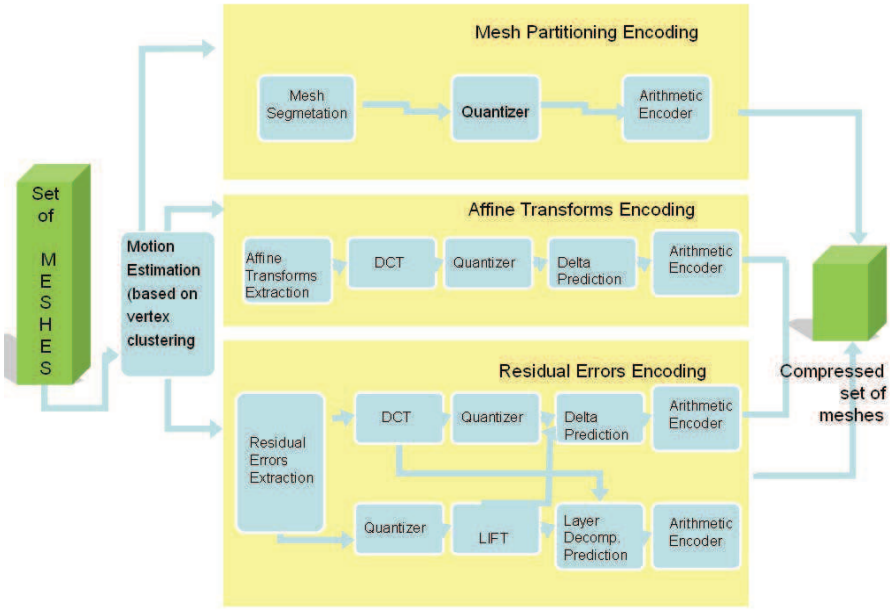


Figure 35. FAMC Encoder.

4.2.2 Proposed Data Model for Generic 3D Graphics

The main objective of the framework we propose is to specify an architectural model, which is able to accommodate a third party XML description of the scene graph with (potential) binarization tools and with MPEG-4 3D graphics compression tools. The use of powerful compression tools for graphics and the generality of graphic primitives representation are the advantages of such an approach. Hence, the compression tools presented in the previous section and specified in MPEG-4 Part 16 would not be applied only to the scene graph as defined by MPEG-4, but also to any scene graph definition. The bitstreams obtained when using the proposed architectural model are MP4 formatted. Advanced compression features such as progressive decompression and support for streaming, provided by the specific compression tools, are preserved with the new model, since it decouples the decoding of the scene graph from the decoding of the object graph. Typically, when consuming content compliant with this model, a player builds first the scene graph based on an XML description, connects the decoders of the elementary streams and renders the results based on the time stamps of the elementary stream. There is no need to rebuild the original XML file before rendering, but this can be done if preferred.

The proposed framework has three layers: Textual Data Representation, Binarisation and Compression, as represented in Figure 36 and detailed below.

- **Layer 1: Textual Data Representation**

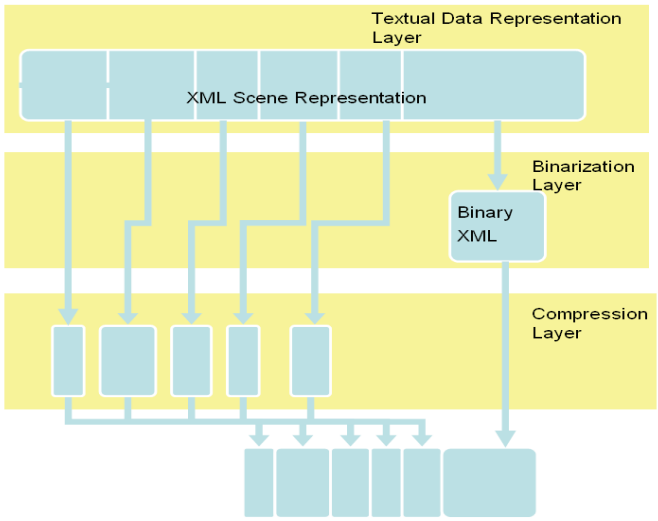


Figure 36. Layers of the 3D Graphics Compression Model.

The current model can accommodate any scene graph and graphic primitives representation formalism. The only requirement for this representation is that it should be expressed in XML. Any XML Schema (specified by MPEG or by external bodies) may be used. Currently the following XML Schemas are supported: XMT, COLLADA and X3D, meaning that the correspondence between the XML element and the compression tool handling it is specified.

- **Layer 2: Binarization**

The binarization layer is processing the XML data that is not encoded by the MPEG-4 elementary bit-streams encoders (e.g. scene graph elements). This data is encapsulated in the "meta" atom of the MP4 file and can be converted into textual representation or binarized by using gzip. Let us note that gzip is natively supported by MPEG-4 for encoding the "meta" atom.

- **Layer 3: Compression**

This layer includes the following MPEG-4 tools: 3D Mesh Compression, Wavelet Subdivision Surface, Coordinate Interpolator, Orientation Interpolator, Position Interpolator, Bone-Based Animation and Frame-based Animated Mesh Compression.

While efficient with respect to the compression due to the state of the art compression tools of MPEG, the proposed framework has the limitation of the need to specify the mapping between the XML elements and the compression tool. In the following section, we demonstrate that such a limitation can be overcome when a priori knowledge exists on the type of the content, such as when applying it to avatars.

4.2.3 Extraction of Avatars from Heterogeneous Data Structures

The avatars' definition is based on a quadruple composed of object graph, geometry, appearance and animation. Such a structure can be detected within heterogeneous content (such as the one usually expressed in XML) based on interpretation of data types. An example of such an algorithm for identifying the avatar components is presented in the next section. Once identified, each component can be individually compressed according to the data model proposed in the previous section.

The algorithm consists in two main parts: one that extracts the geometry and one that extracts the animation.

4.2.3.1 Mesh Extraction Algorithm

The specificity of avatar representation and the current practice of different authoring tools conduct us to formulate the following invariants related to the geometry:

- The mesh is manifold
- All polygons are triangles
- All normals are unit vectors
- All texture coordinates are two-dimensional
- All texture coordinates are normalized

The first step of the algorithm is the identification of long arrays of integer or real numbers. The next steps are as follows:

- Identify the type of real numbers arrays by examining the following conditions:
- if the array length is divisible by 3, the array is a good candidate to represent vertex coordinates
- if the array length is divisible by 3 and the values are normalized, the array is a good candidate to represent normal coordinates
- if the array length is divisible by 2 and the values are normalized, the array is a good candidate to represent texture coordinates
- For each array of real numbers, find the matching index list by comparing the maximum value in the index list with the length of the real number array. All the matches are stored in a pair array and a search is performed in the following order: vertices, normals and texture coordinates. If all arrays are paired, it means that mesh was found and it can be extracted from the XML file.
- For each non paired normals array, find a vertices array of the same length and pair it with its index. For each non paired texture coordinates array, find a vertices array of the same length and pair it with its index. If all arrays are paired, that means that mesh was found and it can be extracted from the XML file.

After this last step, if there are still arrays that are not yet paired, it means that the avatar contains multiple sub-meshes. Therefore, it is possible to find pairs or triplets of integer arrays that have the same length. Each of the arrays in the pairs will index one of the components of the mesh, coordinates, normals and texture coordinates.

- If the there are only vertex coordinates in the mesh
    - Match each index that can index the coordinates array and that makes a manifold mesh with the coordinates array.
- If there is a normals array but no texture coordinates array
    - Match each two indices with the vertex coordinates and normals arrays.
- If there is no normals array but there is a texture coordinates array
    - Match each two indices with the coordinates and texture coordinates arrays.
- If there are normals and texture coordinates arrays
    - Match three indices with each of the arrays.
- If there are no more unmatched arrays, the mesh is found and it can be extracted from the XML file

After this step, if there are still arrays that are not yet paired, the vertex coordinates, normal and texture coordinates shares the same index list. The following steps are performed:

- For each non-paired integer array, check if it is possible to index in some of the float arrays: Vertices, Normals and Texture Coordinates.
- If this is possible, pair them.

4.2.3.2 Animation Extraction Algorithm

The specificity of avatar representation and the current practice of different authoring tools conduct us to formulate the following invariants related to the animation.

- The animation is represented as a list of keys and a list of keys values for each component, which in case of an avatar is a translation or a rotation transform;
- The frame rate is constant and there are no skipped frames;
- In addition, we constrain the system to a single animation per file.

The animation detection algorithm consists in the following steps:

- Identify all the non-paired arrays of real numbers,
- For each array, if the values are increasing and are positive, it corresponds to the key-time stamps, therefore add it to the key-time list array, if not, add it to the key-value list array

The next phase consists in pairing each key-time list with the key-value list:

- A first supposition is that the animation transformations are represented as 4x4 matrices. If the length of the key-time list is 16 times bigger than the length of the key-value list, the two lists will be paired.
- If there are still not paired lists, a second supposition is that the animation is represented as translations (3 real numbers) and/or rotations represented with quaternions (4 real numbers). Therefore, if the length of the key-time list is 3, 4 or 7 times bigger than the length of the key-value list, the two lists will be paired.

Once the key and key-values extracted, they are compressed by the BBA decoder. The compression results when using the above algorithm for avatar detection and the compression framework together are presented in Chapter 4.

4.2.4 Adoption of the Compression Data Model as MPEG-4 Part 25

MPEG recognized the utility of such a compression data model and accepted it for standardization in a separate part of MPEG-4. The following technical contributions were needed in order to promote the model as a standard.

In July 2007 we demonstrated that MPEG-4 Elementary Streams (ES) for 3D graphics may be applicable to scene graph representation and primitives defined by other bodies than MPEG. Therefore, we classified the elementary streams with respect to the data type they encode and described the main features. The compression tools were 3DMC and WSS to encode the geometry, CI, OI, PI and BBA to encode the animation. In order to describe how the MPEG-4 ESs can be applied to third party scene description, we instantiated the model for COLLADA and X3D. As a result of this contribution, MPEG initiated the work on a new part of MPEG-4, namely Part 25.

In January 2008 we proposed a software implementation of the data model able to encode a COLLADA file as an MP4 stream. That software was adopted by ISO as Reference Software for Part 25.

In January 2008 we also proposed a set of 3D graphics content to be used as conformance.

During 2008, the reference software was upgraded to support another MPEG-4 ES – FAMC, and to apply the data model to X3D and XMT. An additional set of conformances bitstreams were proposed in April 2008.

To ease the usage of the reference software, we created a GUI, which was proposed and accepted by MPEG in October 2008.

In February 2009, we provided a set of rules to explain how the different elementary streams for encoding the geometry and animation are selected. The data to encode is classified in three categories: static, geometric transforms and coordinates deformation and the rules are the following:

- if a mesh element is identified in the original file, the 3DMC encoder is called
- if geometric transforms are identified in the original file, two situation may occur:
  - if the transform corresponds to a joint, the BBA encoder is called
  - if the transform does not correspond to a joint, the PI/OI encoder is called
- if the deformation of coordinates is identified in the original file (mainly a coordinates interpolation element), an instance of the FAMC or CI encoder is called (the choice is based on the user input)

ISO published the MPEG-4 Part 25 as an international standard in March 2009 and it is available online.

4.3 Content Adaptation for Weak Terminals

In order to identify the most appropriate manner to adapt graphics content and especially avatars for low processing platforms, let us analyze the processing pipeline.

4.3.1 Analysis of the Mobile Phone Processing Pipeline

One of the current hardware limitations of mobile phones compared to desktop computers is the processor power; another refers to memory, both RAM and mass storage, this conducting to restrictions on the amount of data that can be manipulated. On the other hand, mobiles have relatively small displays, therefore the spatial resolution can be decreased with no visible artifacts. However, other features like the temporal resolution (animation frame rate) cannot be reduced lower than a certain amount (usually 20 fsp).

Table 4 summaries the capabilities of mobile phones, the requirements for processing 3D graphics and the proposed solution to ensure functional balance between the two.

Capability	Requirement	Solution
low processor power	low complexity of algorithms, small number of processor cycles	algorithms optimization
constrained memory/storage for both RAM and mass storage	limited data to store and manipulate	content optimization
low data transmission rates, low bitrates	limited access to data with respect to quantity and speed	compact/compress content
low screen size	allows decreasing the representation resolution	content spatial adaptation

Table 4. Capabilities, requirements and suggested solutions for 3D graphics applications for mobile phones.

The strong restrictions due to terminal capabilities and transmission bandwidth usually result in the necessity of (1) relatively simple, optimized 3D processing algorithms, (2) adequate (optimized) content for mobiles and (3) compacting/compressing the content for transmission.

Based on these requirements, we aim to define a low complexity baseline profile for delivering and rendering 3D graphics content and especially avatars on mobile phone devices.

Let us first analyze the processes needed for displaying and animating an avatar on a mobile phone. Depending on the rendering settings (only vertices, wireframe, flat shaded surface, textured, lit surface), the requirements are different. The processing pipeline[14] is illustrated in Figure 37 and detailed as follows.

Step 1 – Geometry and textures decoding. Once the avatar is loaded, the four components representing it (geometry, appearance, animation and object graph) are transmitted to dedicated decoders/parsers. Compressed geometry and images are decoded into a flat representation such as vertex buffer and index buffer (VBIB) for geometry and respectively image matrix for textures.

Step 2 – Animation Frame decoding. In a continuous manner, an animation frame containing skeleton parameters is decoded. The decoded geometry is updated by applying the bones' geometrical transformations to the vertices composing the mesh.

Step 3 - Primitive Processing. The information for VBIB is sent to the geometry engine display list.

Step 4 - Vertex Operation: the geometry engine reads the vertex data from the display list buffer and processes the data for each vertex. Each vertex and normal coordinates are transformed from local object coordinates into global coordinates. Also, if lighting is enabled, the lighting calculation per vertex is performed using the transformed vertex and normal data. This lighting calculation updates the new color of the vertex. Then vertices are transformed by a projection matrix in order to map the 3D scene to screen space coordinates.

Step 5 - Primitive Assembly: after vertex operation, the primitives (point, line, and polygon (triangles)) are clipped by viewing volume clipping planes; after that, if culling is enabled, a culling test is performed.

Step 6 - Rasterization: rasterization can be assumed as a process of conversion of both geometric and pixel data into fragment; each fragment corresponds to a pixel in the

[14] http://www.opengl.org/

frame buffer. The rendering engine fills triangles and draws lines and points using the data sent from the geometry engine on screen coordinates.

Step 7 - Fragment operation is the last process to convert fragments to pixels onto the frame buffer. The first process in this stage is texel generation. A texture element is generated from texture memory and is applied to the each fragment. Then fog calculations are applied. Several fragment tests follow in order: Alpha Test, Stencil Test, and Depth Test. Finally, blending and dithering are performed and actual pixel data are stored in the frame buffer.

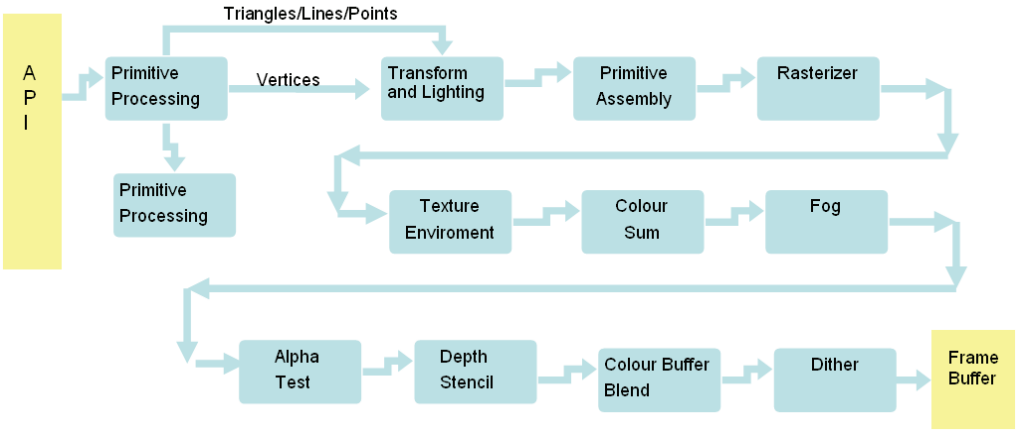


Figure 37. 3D Content processing line

Steps 2 to 7 reiterate for each animation frame. The decoding depends on the number of bones in the skeleton and deforming the mesh is dependent on the global number of vertices. Furthermore, most of the rendering phases are dependent on the number of vertices and the number of primitives (in most cases, triangles). Therefore, let us analyze the capabilities of existent mobile platforms, as illustrated in Table 5.

Device	Resolution	Triangle Throughput
Medium range PC equipped with ATI Radeon HD5770 graphics card	1280x1024 (1680x1050)	About 850 MTriangles/s
Apple IPhone 3GS	320x480	5590 kTriangles/s
SonyEricsson U1i Satio	360 x 640	3024 kTriangles/s
Nokia N95 8GB	240x320 320 x 240	899 kTriangles/s
Nokia 5530	640x360	262 kTriangles/s
Samsung SGH-I560	240x320	196 kTriangles/s

Table 5. Comparison between the capabilities of a PC and different mobile devices[15]

Let us note that a standard PC can provide rendering objects about 100 – 5000 times faster than a mobile device. Based on the analysis of local configurations (processing power, transmission bandwidth *vs.* display resolution), two types of content processing are needed: simplification and compression. In the following sections we introduce our methods for addressing both of them, and present the improvement to the existent techniques. Let us note that our proposed model was submitted to MPEG committee for standardization. Since the market has not reached this level yet, it couldn't be profiled,

[15] http://www.glbenchmark.com/

but a proven concept already exists. Once the industry tackles this problem and exposes the need for such a model, the profile can be published.

4.3.2 Avatar-graph, Geometry, Appearance and Animation Simplification

If previously the trend was "multiple devices, multiple content" (MDMC), in recent years the tendency has shifted to "multiple devices, single content" (MDSC). In general, for this purpose single high quality content is created and then it is optimized to fit the requirements of different devices. Particularly for avatars, this leads to the need for simplifying the avatar graph, the geometry, the appearance and the animation.

The proposed approach for avatar adaptation is a complete framework, taking into consideration all the four components defining it: object graph, geometry, appearance and animation. In the following sections we present the adaptation for each of them.

The complexity of an object graph depends on the number of different node types that the client should support. Starting from the formalism defined in MPEG-4 BIFS, we first propose a content profile fitting the mobile phones' reduced capabilities. Its design is based on a compromise between the number of different node types and the completeness of avatar description. To address standalone applications requiring the registration of the avatars in the scene graph as well, we complete the avatar-related nodes set with a second one referring to the creation of simple scenes.

Node name	Scene/avatar
Appearance	Avatar
Color	Avatar
Coordinate	Avatar
Group	Scene
ImageTexture	Avatar
IndexedFaceSet	Avatar
Material	Avatar
Normal	Avatar
PointLight	Scene
Shape	Avatar
TextureCoordinate	Avatar
Transform	Scene
Viewpoint	Scene
SBVCAnimation	Avatar
SBBone	Avatar
SBSite	Avatar
SBSkinnedModel	Avatar
SBVCAnimationV2	Avatar
SBSegment	Avatar

Table 6. MPEG-4 BIFS Nodes corresponding to the avatar profile and to the simple scene graph.

At the root of the avatar-graph, a **SBVCAnimaton** (or its extension, **SBVCAnimation2**) Node is defined. The main purpose of this node is to group the components of the virtual character and to attach an animation resource. The virtual character is defined as a **SBSkinnedModel** Node that contains (1) a collection of bones, defined with **SBBone** Node, (2) additional objects such as clothes or accessories defined as **SBSegment** Nodes, (3) a collection of muscles, defined with **SBMuscle** Nodes. Additionally, the **SBSite** node allows defining semantic regions in the space of the virtual character. The mesh definition of the avatar is contained in the **Shape** Node and its children, the geometry and appearance. If morphing is used for animating the virtual character, **MorphShape** Node is also defined. Table 6 presents the proposed profile, able to represent a static and/or an animated object within a simple scene. An example of an avatar graph compliant with this profile is provided in Annex E.

For mesh and appearance adaptation our solution is based on the algorithm described in [Preda05]. A set of experiments were conducted in order to validate its appropriateness for the mobile phone case. Visual results illustrated in Figure 38 show several resolutions examples of the Hero model with mesh simplification and texture simplification.

a) b) c) d)

Figure 38. An Avatar model.

The original model displayed in Figure 38.a) has 3587 vertices. In Figure 38.b) it is adapted by mesh simplification, only 70% of vertices are conserved, (2745 vertices) and the texture is reduced by half in each direction. In Figure 38.c) the mesh has 60% of the initial vertices (1903 vertices) and the texture is reduced by four on each direction. Both examples show no significant loss in visual quality. In Figure 38.d), the mesh is simplified to preserve only 10% of the original, (1060 vertices) and the texture is divided by eight in each direction. When displayed on a mobile phone screen, the quality is still satisfactory.

While geometry and textures are loaded and decoded only once, a more sensitive component to the limited requirements of the mobile phones is animation, since decoding and mesh deformation should be performed for each frame.

A straightforward manner to reduce the amount of animation data is to identify key frames so that any intermediate frames can be interpolated from the set of key-frames.

Animation simplification based on frame reduction was achieved by considering a progressive approach. Given an original animation sequence of n frames, in order to obtain a simplified sequence with m frames ($m < n$) that approximates better the original curve, one has to minimize the area between the original curve and the reconstructed (by interpolation) one. For example, in Figure 39, illustrating the variation of the x, y, z coordinates of the center of a bone, if frame #19 is removed, the distortion is smaller than in the case when frame #14 is removed.

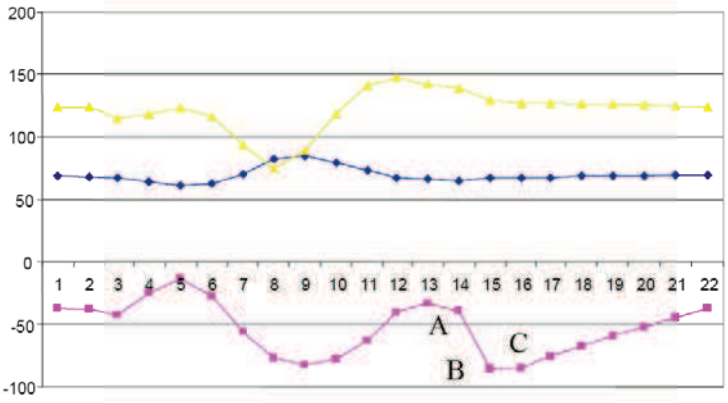


Figure 39. Variation of x (red/square), y (blue/rhomb) and z (yellow/triangle) coordinates (in the global coordinate system) of the center of a bone.

Considering this condition for all the bones (or the subset of extreme bones), the optimization problem becomes difficult to solve. Figure 40 shows the variation for the center of all the bones.

To overcome the complexity of such computation we adopted an incremental approach: for each frame k, for each extreme bone $\boldsymbol{B_i^{extreme}}$ and for each coordinate axis (x, y and z), we compute the area $\boldsymbol{A_{frame_k}^{i,(axis)}}$ $A_{frame_k}^{i,(axis)}$ of the triangle obtained between the original signal and the interpolated one.

$$A_{frame_k}^{i,(axis)} = \int_{t_{k-1}}^{t_{k+1}} \left(C^{(axis)_0}(t)\right) - \left(C^{(axis)_R}(t)\right) dt \qquad \textbf{(2)}$$

where *(axis)* is the one of x, y, z, $C^{(axis)_0}(t)$ is the original bone-center trajectory on the *(axis)* and $C^{(axis)_R}(t)$ is the bone-center trajectory reconstructed by interpolation.

In Figure 39 we illustrate as triangle ABC the area for frame #14 and for the x coordinate of a bone.

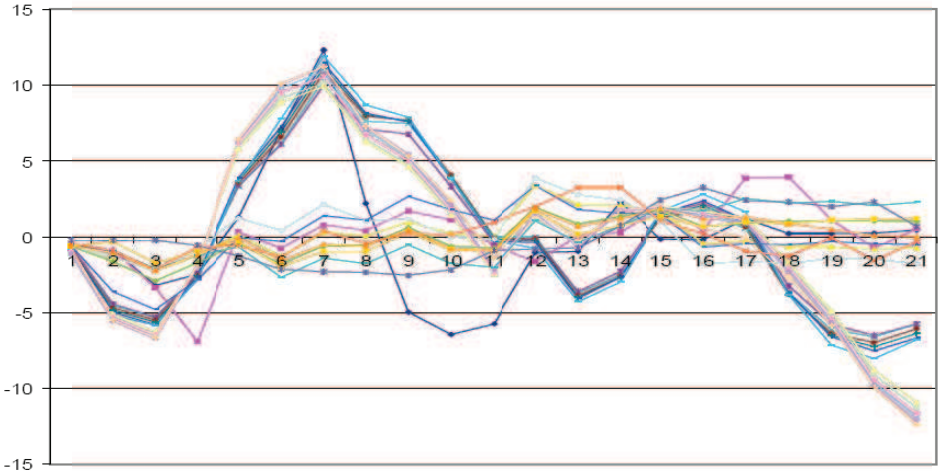


Figure 40. Variation of x, y and z coordinates (in the global system) of the center of all the extreme bones.

For any frame k, the sum of all the areas for all the extreme bones (and for each dimension x, y, z) is computed by:

$$A_{frame_k} = \sum_{i=0}^{i=N_{extreme\ bones}} \left(A^{i,x}_{frame_k} + A^{i,y}_{frame_k} + A^{i,z}_{frame_k} \right) \quad \textbf{(3)}$$

The minimum of the sums indicates the frame that has to be removed. We repeat the algorithm until the number of removed frames equals n-m. After frame reduction a new BBA stream is obtained by encoding the m frames, indicating for each frame the number of intermediate frames to be obtained by interpolation at the terminal side. Concerning the interpolation methods for real-time purposes, we use a linear interpolation defined as:

$$l(x) = (1 - x)x_0 + xx_1 \quad \textbf{(4)}$$

where X_o X_o is the translation/scale in the current key-frame, X_1 is the translation/scale in the next key-frame, and $x \in [0,1]$ for the *translation* and *scale* components, and spherical linear quaternion interpolation:

$$Slerp(x) = \frac{x_0 \sin[\alpha(1 - x)] + x_1 \sin(\alpha x)}{\sin(\alpha)} \quad \textbf{(5)}$$

where x_0(respectively x_1) is the rotation/scale orientation quaternion in the current (respectively next) key-frames, and $\alpha = \cos^{-1}(x_0 x_1)$ and, $x \in [0,1]$ for the *rotation* and *scaleOrientation* components.

The advantage of the incremental approach (removing only one frame per operation) is the fine-tune granularity of the file size. Therefore, in a time-variant environment of the network capabilities, it is possible to dynamically adapt the size of the animation stream to the constraints of the network.

4.3.3 Optimized Animation Coding

In the following section, we introduce the two optimizations performed on a typical MPEG-4 encoder: dynamic prediction and dynamic value range.

With signal prediction, as expected, the amount of information needed to be sent decreases, since the difference between exact values is usually smaller than the value itself. However, it is possible that after a certain number of frames (depending on the animation type) the differences become bigger than the information itself. For such reasons an (I) frame (or segment) has to be forced. In addition, forcing (I) frames at specific moments allows easy animation editing and random access to the animation sequence. In our optimized encoder we analyze for each frame what is less expensive in terms of bandwidth transmission and we take a local decision to force an (I) frame (or segment) or not. In our experiments, we expressed this parameter as PbI (number of P frames between two I frames).

The relatively big tables for the arithmetic encoder - the range of the values for I frames is [-1860 1860] and for P frames [-600 600] - require large amounts of memory during decoding. In our optimized version of the encoder, we dynamically compute the range with respect to the content. This allows reducing the memory used by symbol probabilities tables. Furthermore, defining (min, max) brings one more advantage compared to the method before: it enables better precision, since tables are adapted to current data. The only drawback is that (min, max) values have to be transmitted. However, this is acceptable and the final decoder is built on this improvement.

To ensure the quality of the optimized BBA encoder, but also to study the influence of the different parameters on the decoder, we conducted experiments on the database available at mocap.cs.cmu.edu which consists in about 1700 motion capture files of different nature and complexity. The experimental results of this optimization are presented in Chapter 4.

3.3.4

4.4 Interoperability Framework

4.4.1 Analysis of Models Used in Virtual Worlds

Despite the fact that several markup languages related to avatars and virtual agents exist, ensuring interoperability for avatars between different virtual worlds (VWs) cannot be obtained yet in an easy, ready to use and integrated manner. Identifying this gap and recognizing that only the existence of a standardized format can make virtual worlds be deployed at a very large scale, MPEG initiated in 2008 a new project called MPEG-V (Information exchange with virtual worlds). Concerning the avatars, the following requirements should be fulfilled by MPEG-V:

a) it should be possible to easily create importers/exporters from various VWs implementations,

b) it should be easy to control an avatar within a VW,

c) it should be possible to modify a local template of the avatar by using data contained in an MPEG-V file.

In the MPEG-V vision, once the avatar is created (possibly by an authoring tool independent of all VWs), it can be used in an existent or future VW. A user can have a unique presentation inside all VW, like in real life. He can change, upgrade, teach his avatar, i.e. "virtual himself" in one VW and then all the new properties will be available in all others. The avatar itself should then contain representation and animation features but also higher level semantic information. However, a VW will have its own internal structure for handling avatars. MPEG-V does not impose any specific constraints on the internal structure of representing data by the VW, but only proposes a descriptive format able to drive the transformation of a template or the creation from scratch of an avatar compliant with the VW. All the characteristics of the avatar (including the associated motion) can be exported from a VW into MPEG-V and then imported in another VW. In the case of the interface between virtual worlds and the real world (requirement 2), the avatar motions can be created in the virtual world and can be mapped on a real robot for the use in dangerous areas, for maintenance tasks or the support for disabled of elderly people and the like. The inverse operation is also possible: avatars can be animated by signals captured from the real world, as in the case of motion capture system a descriptive format specifying the avatar features, it may be combined with MPEG-4 Part 16 (that includes a framework for defining and animating avatars) to provide a fully interoperable solution.

Defining an interoperable schema as intended by MPEG-V can be of major economic value, being one step towards the transformation of current VWs from stand-alone and independent applications into an interconnected communication system, similar with the current Internet where a browser can interpret and present the content of any web site. At that moment, the VW providers will not be any more providers of technology, but will concentrate their efforts on creating content, once again the success key of Internet. Table 7 lists several Virtual Worlds and mentions their features.

In the next section we are proposing the schema representation for avatars allowing the interoperability between different VWs. Let us note that this schema was proposed [Jovanova09, Jovanova10] to MPEG for inclusion in MPEG-V standard and it was adopted as Part 4 of this standard. Let us note that Part 4 contains the schema for generic virtual objects contributed by other MPEG participants as well.

Name	launch date	3D avatar	3D objects	Chat	Age	Entert., games	Specific	Money
Habbo[16]	January 2001	yes	yes	yes	10-16	yes/yes		no
Yoo-walk[17]	2006	yes	no	no	all	yes/no		no
HiPiPi[18]	2005	yes	Yes	n/a	all	yes/no		yes
IBM QWAQ[19]		business friendly	yes	n/a	n/a	yes/no	Project management	n/a
Sony Playstation Home[20]		yes	yes	yes	all	yes/yes		n/a
Active Worlds[21]	1997, in beta 1995	yes	yes	yes	all	yes/no	to develop content	no
Google Lively	defunct	yes	yes	yes	n/a	n/a, yes	defunct	no
Second Life[22]	23 June, 2003	yes	yes	yes	18+	yes/yes	to develop content	yes
Teen Second Life[23]		yes	yes	yes	-17	yes/yes		no
There[24]	27 October, 2003	yes	yes	yes	13+	yes/yes		yes
Sims[25]	2002	yes	n/a	yes	all	yes/yes		yes

Table 7. Different Virtual Worlds and their features.

4.4.2 Avatar Characteristics Set

The main aim of creating the common schema is to establish possible mapping or transfer between different avatar attributes, extracted from the examined VWs/techniques/standards. Therefore, the first step is to identify the exhaustive set of features that should be considered. We performed an analysis of existent VWs and most popular games, tools and techniques from content authoring packages, together with the study of different virtual human related mark-up languages, in order to create a schema that defines a set of metadata addressing all the avatar components. This survey allowed us to conclude the following:

- While in data representation standards a scene-graph is exposed, the usual practice in VWs and games is to keep it strictly internal. Therefore, the schema we propose does not include scene graph representation in the data model.
- The avatar appearance is exposed to the user by numerous VW implementations; therefore, we provide a set of parameters allowing to describe it in the proposed data model.

[16] www.habbo.fr
[17] www.yoowalk.com,
[18] www.hipihi.com/en/
[19] iggyo.blogspot.com/2009/03/qwaq-and-suits-should-linden-lab-worry.html
[20] fr.playstation.com/
[21] www.activeworlds.com/
[22] secondlife.com/
[23] secondlife.com
[24] www.there.com/
[25] www.thesimsonline.com

- The avatar geometry is strongly VW/game dependent in the sense that the resolution is carefully fine-tuned because the global rendering performance is dictated by the number of polygons displayed on the screen. Therefore, we include in the proposed data model only means of controlling global geometric measures and we consider it as an external resource that can optionally be imported in the virtual world.
- Concerning the animation, most of VWs/games expose the avatar skeleton (bones) in order to attach (control) the animation. Therefore, in the data model we propose, we include skeleton characterization and a set of animation sequences.

We propose to expose a set of Personalization Parameters (PP) grouped in three categories: appearance, animation and skeleton (control) attributes. In general, obtaining the scheme for any of the three groups consists of three main steps:

- Analysis of existing methods for avatars' metadata and acquisition of a super set of characteristics.
- Identification of the semantic of individual avatar characteristic and definition of the associated schema with respect to the associated semantic.
- Verification of the completeness of the schema by mapping the characteristics between different virtual worlds.

In the following section we presents in detail each of the steps for the three semantic groups.

4.4.2.1 Appearance Elements

In order to create a metadata model for avatar appearance, we evaluated the following VWs/games/representation standards: SecondLife, IMVU, Entropia Universe, SonyPlaystation and HumanML. More than 150 different parameters that are used for describing different appearance features, separated in 14 groups (illustrated in Figure 41), were obtained.

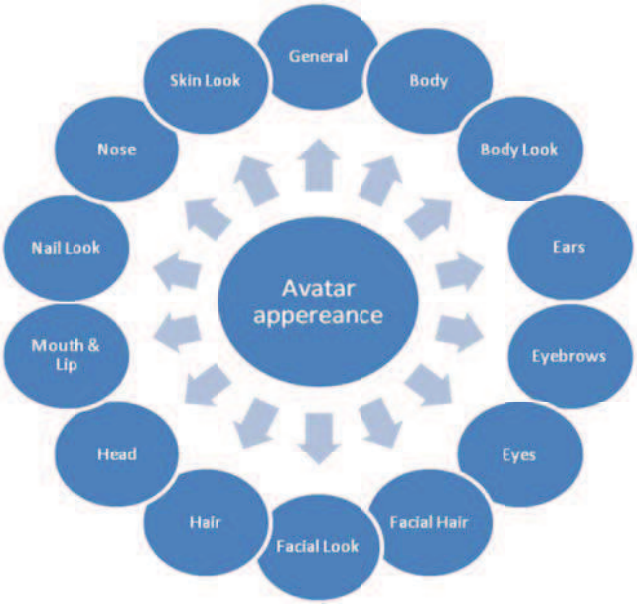


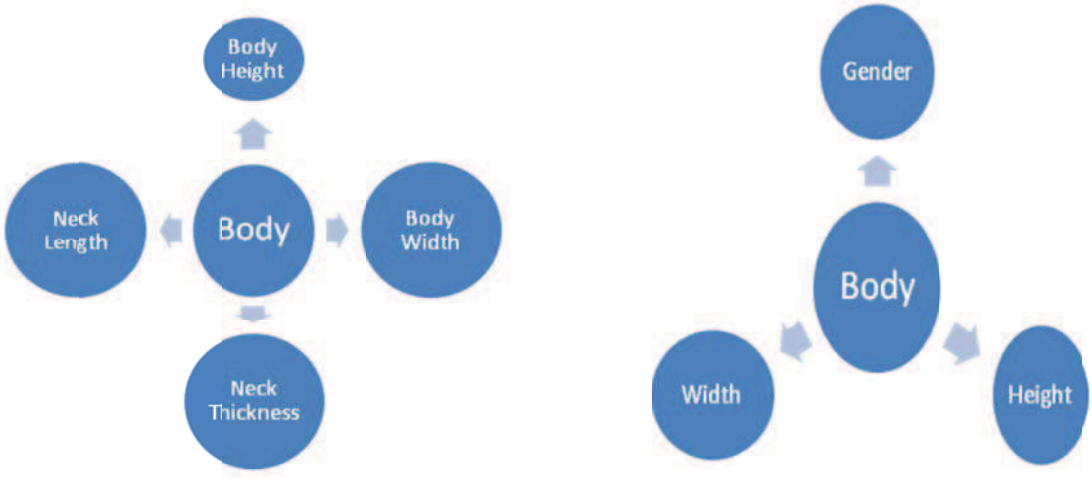
Figure 41. Avatar appearance.

Each of these groups contains several elements describing avatar appearance features. The different Virtual Worlds and representation standards expose the avatar characteristics in different quantities. For example, in the group Body, Second Life, Entropia Universe, Nintendo Wii, Sony PlayStation and HumanML expose respectively, 22, 2, 1, 4 and 3 components (illustrated in Figure 42).

Figure 42. Elements describing human "Body" in Second Life.

a) b)

Figure 43. Entropia Universe (a) and Nintendo Wii (b) elements for "Body" description.



a) b)

Figure 44. Sony PlayStation (a) and HumanML (b) elements for "Body" description.

It can be observed that Second Life supports and at the same time exposes the most of the appearance features. Other VWs, such as Nintendo Wii (Figure 43b), use simplistic avatars, therefore the set of personalization parameters is reduced. On the standard formalisms' side, only few of them are dealing with defining metadata for appearance features. HumanML is one that exposes few, as presented on Figure 44b; however, this set is under-defined and is far from providing a real personalization support.

4.4.2.2 Animation Elements

Several standards provide a rich set of animation personalization parameters, mainly the ones that describe an animation which has as a result an emotional appearance of the avatar. In general they are related to facial animation (sadness, anger, happiness, etc.). On the other hand, in several VWs, animations are covering also body motion. By analyzing several sources, we obtained a set of around 400 animations grouped in 12 categories as illustrated in Figure 45.

Figure 45. Avatar animation.

Each of these groups contains several elements describing avatar animation capabilities. The different Virtual Worlds and representation standards expose the avatar characteristics in different quantities. For example, in the group Greetings, Second Life, Sony PlayStation and Microsoft Agent expose 5, 4, and 2 components respectively (illustrated in Figure 46 and Figure 47).

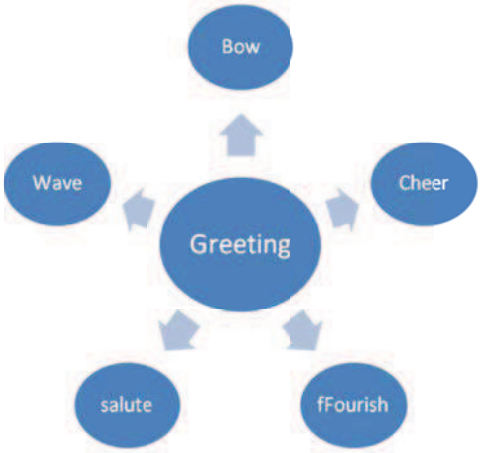
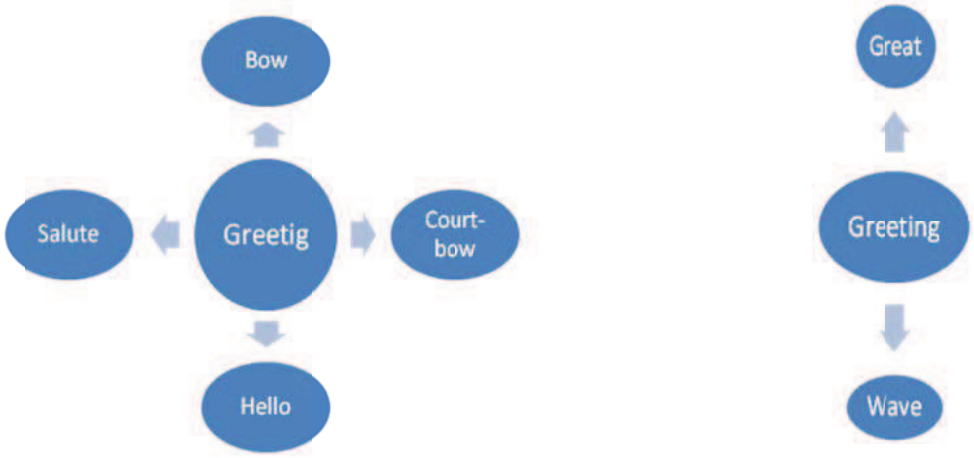


Figure 46. Second life animations in "Greetings" group.



a) b)

Figure 47. Sony PlayStation (a) and Microsoft Agent (b) animations in the "Greetings" group.

4.4.2.3 Object Graph - Skeleton Elements

In the proposed framework, the skeleton serves to two scenarios: it allows to define different animation sequences that are associated to the avatar and it allows to place sensors in the case when the animation is directed by signals from the real world (such as motion capture systems). Therefore, the skeleton contains all the bones in a typical human body, a set of 101 bones grouped in 5 categories, as illustrated in Figure 48.

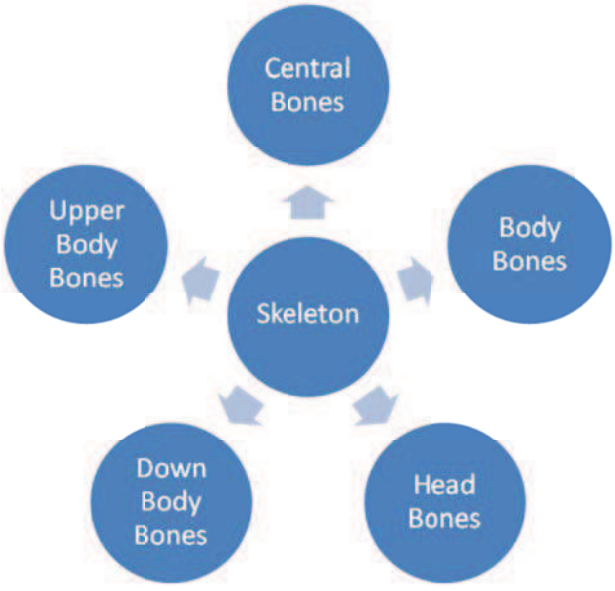


Figure 48. Avatar Skeleton.

Figure 49 illustrates the bones that are attached to the upper part of the body.

Figure 49. Human Upper Body bones.

In order to compare the proposed framework with the existing representation, we provide illustrative examples for the upper body bones in H-Anim (Figure 50), Second Life (Figure 51), IMVU (Figure 52), VHML/BAML (Figure 53) and AML (Figure 54).

Figure 50. H-Anim defined bones for the upper part of the body.

Figure 51. Second Life defined bones for the upper part of the body.

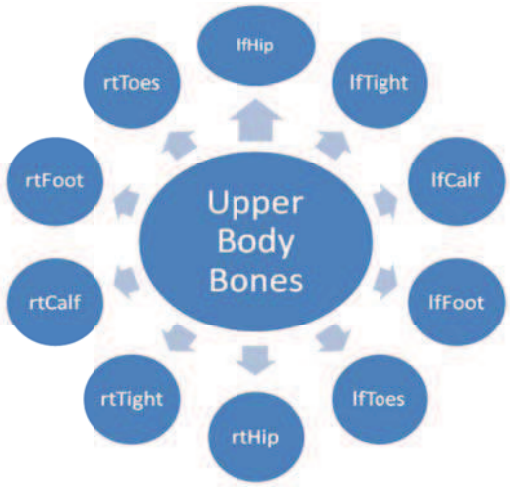


Figure 52. IMVU defined bones for the upper part of the body.

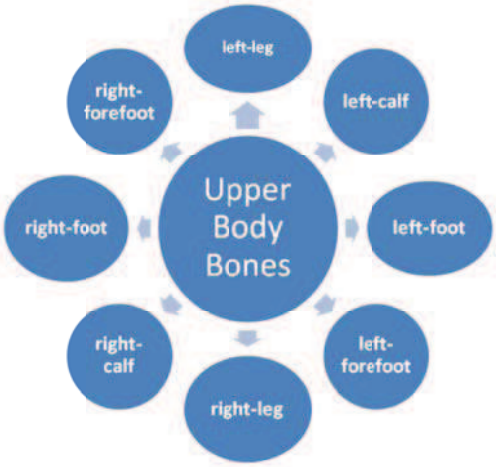


Figure 53. VHML/BAML defined bones for the upper part of the body.

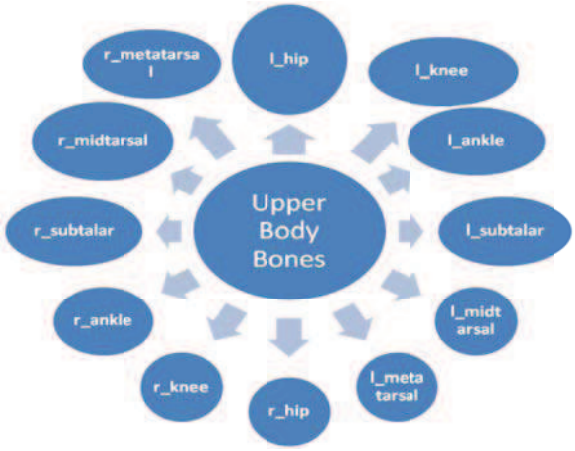


Figure 54. AML defined bones for the upper part of the body.

4.4.3 Proposed Metadata Model

Our proposal is to formalize the avatar metadata by defining the "Avatar" type composed of the following elements (an extract of the XML schema expressing the formalism is provided in Annex D).

- **Appearance**

The "Appearance" element contains the high level description of the avatar appearance and may refer to one or several medias containing the exact geometry and texture, as well as to other resources such as clothes and associated objects. This element also provides high level semantic information such as anthropometric data.

- **Animation**

The "Animation" element contains the description of a set of animation sequences that the avatar is able to perform and may refer to several resources containing the exact (geometric transformations) animation parameters.

- **Control**

The "Control" element contains a set of descriptors defining possible place-holders for sensors on body skeleton and face feature points.

Let us note that the proposed formalism was accepted as a standard schema for avatar representation in MPEG-V Part-4. During the standardization process, this schema was completed by three additional elements, "CommunicationSkills" and "Personality" originally proposed by [Oyarzun09] and "HapticProperties", originally proposed by [Cha07]. The definition of these additional elements is the following.

- **CommunicationSkills**

The "CommunicationSkills" element contains a set of descriptors providing information on the different modalities an avatar is able to communicate.

- **Personality**

The "Personality" element contains a set of descriptors defining the personality of the avatar.

- **HapticProperties**

The "HapticProperties" element contains the high level description of the haptic properties.

The schema defining the "avatar" element, grouping together the six categories introduced above, is illustrated in Figure 55.

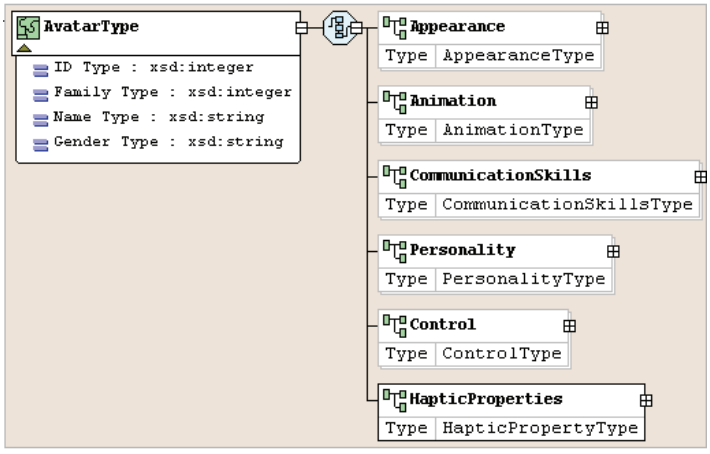


Figure 55. "Avatar" element compositing elements and attributes.

In the next section we introduce only the three parts that are our contribution, the appearance, the animation and the control elements. A detailed explanation of other elements, including the schema definition is provided in [Preda09].

4.4.3.1 "Appearance" Element

The "Appearance" element contains descriptions of the avatar's different anatomic segments (size, form, anthropometric parameters), as well as references to external resources referring to the geometry and texture resources. While the first can be used to adapt the internal structure of the VW avatar - personalizing it, the second can be used to completely overwrite it. The second operation can be performed only when the format for the resource itself is also known by the importer/exporter— as it is the case when using MPEG-4 3D Graphics. In addition, this element also contains characteristics of objects that are related to the avatar such as clothes, shoes or accessories. The corresponding schema of the "Appearance" element is represented in Figure 56.

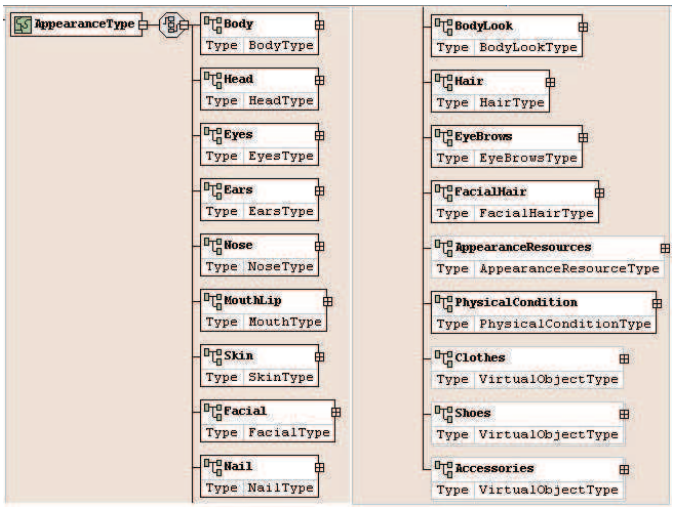


Figure 56. "Appearance" element compositing elements.

A simple example of usage of the "Appearance" element is provided below.

```
<Appearance>
      <Body>
            <BodyHeight value=165/>
            <BodyFat value=15/>
      </Body>
      <Head>
            <HeadShape value="oval"/>
            <EggHead value="true"/>
      </Head>
      <Clothes ID=1 Name="blouse_red"/>
      <AppearanceResources>
            <AvatarURL value="my_mesh"/>
      </AppearanceResources>
</Appearance>
```

4.4.3.2 "Animation" Element

The "Animation" element contains a complete set of animations that the avatar is able to perform, grouped by semantic similarity (Idle, Greeting, Dance, Walk, Fighting, Actions). A special group contains common actions such as Drink, Eat, Talk, Read and Sit. As in the previous case, the animation's geometrical parameters are represented by external resources, MPEG-V providing only the label of the animation sequences. The corresponding schema of the "Animation" element is provided in Figure 57.

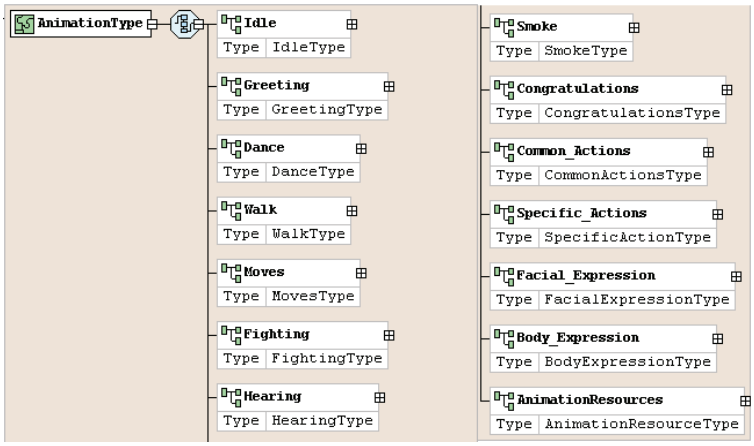


Figure 57. Animation element compositing elements.

A simple example of using the "Animation" element is provided below.

```
<Animation>
      <Greeting>
            <Salute>salut</Salute>
            <Cheer>cheer</Cheer>
      </Greeting>
      <Fighting>
            <shoot>pousse</shoot>
            <throw>throw</throw>
</Fighting>
      <Common_Actions>
            <drink>boire</drink>
            <eat>manger</eat>
            <type>type</type>
            <write>ecrire</write>
      </Common_Actions>
<AnimationResources>
      <AnimationURL>my_anim</AnimationURL>
</AnimationResources>
</Animation>
```

4.4.3.3 "Control" Element

The main purpose of the "Control" element is to define the placements on the human body where sensors can be installed for motion capture. These locations are grouped in body features (bones of the body-skeleton) and face features (3D locations on avatar's face). The corresponding "Control" element schema is illustrated in Figure 58.

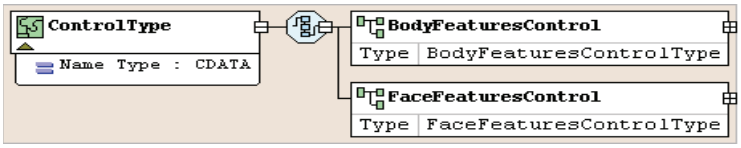


Figure 58. Control element compositing elements.

A simple example of using the "Control" element is provided below.

```
<Control>
      <BodyFeaturesControl >
            <UpperBodyBones>
                  <LCalvicle>my_LCalvicle</LCalvicle>
                  <RClavicle>my_RCalvicle</RClavicle>
            </UpperBodyBones>
            <DownBodyBones>
```

```
                  <LFemur>my_l_femur</LFemur>
                  <LTibia>my_l_tibia</LTibia>
                  <RFemur>my_r_femur</RFemur>
                  <RTibia>my_r_tibia</RTibia>
            </DownBodyBones>
      </BodyFeaturesControl>
      <FaceFeaturesControl>
            <HeadOutline>
                  <Left X=0.23 Y=1.25 Z=7.26/>
                  <Right X=0.25 Y=1.25 Z=7.21/>
                  <Top X=2.5 Y=3.1 Z=4.2/>
                  <Bottom X=0.2 Y=3.1 Z=4.1/>
            </HeadOutline>
      </FaceFeaturesControl>
</Control>
```

The "Control" element ensures controlling the avatar from external signals as well as a common base layer to allow motion retargeting between avatars with similar or slightly different structure.

4.4.3.4 Extensibility Feature

In the design of the proposed schema, one of the main aims was to define an extensive set of meaningful features. However, it is always possible that a future virtual world implementation needs extra features, not foreseen currently. To support such extensibility, we add a mechanism that extends the proposed formalism with proprietary, but well-defined or at least well-formatted data. The extensibility mechanism is based on a new element, called *<Extra>*. The nature of this element allows it to consider additional resources (such as the animation parameters) and/or associated semantics (such as the name of the animation sequence).

The element *<Extra>* can contain any well-formed XML data. The attribute *xmlns* identifies an additional schema to use for validating the content of this instance document.

4.4.4 Adoption of the Interoperability Framework as MPEG-V Part 4

The MPEG committee initiated the MPEG-V standardization process in October 2008 by publishing a call for proposals with the objective to formalize the data representation used in the transfer of information between real and virtual worlds and between different virtual worlds. Therefore, the targeted technologies cover the representation of the control information to and from devices in the real world and into and from the virtual world. Examples of these representations are the representation of sensory input devices like smart vision systems, environmental and body sensors, and sensory output rendering devices like lights, heaters, fans, displays, speakers and others similar to them. A second area of technologies covers the bidirectional representation of information exchanged between the between virtual worlds, such as virtual objects and avatar characteristics.

The overall system architecture for the MPEG-V framework, comprising both of the standardization areas, is illustrated in Figure 59.

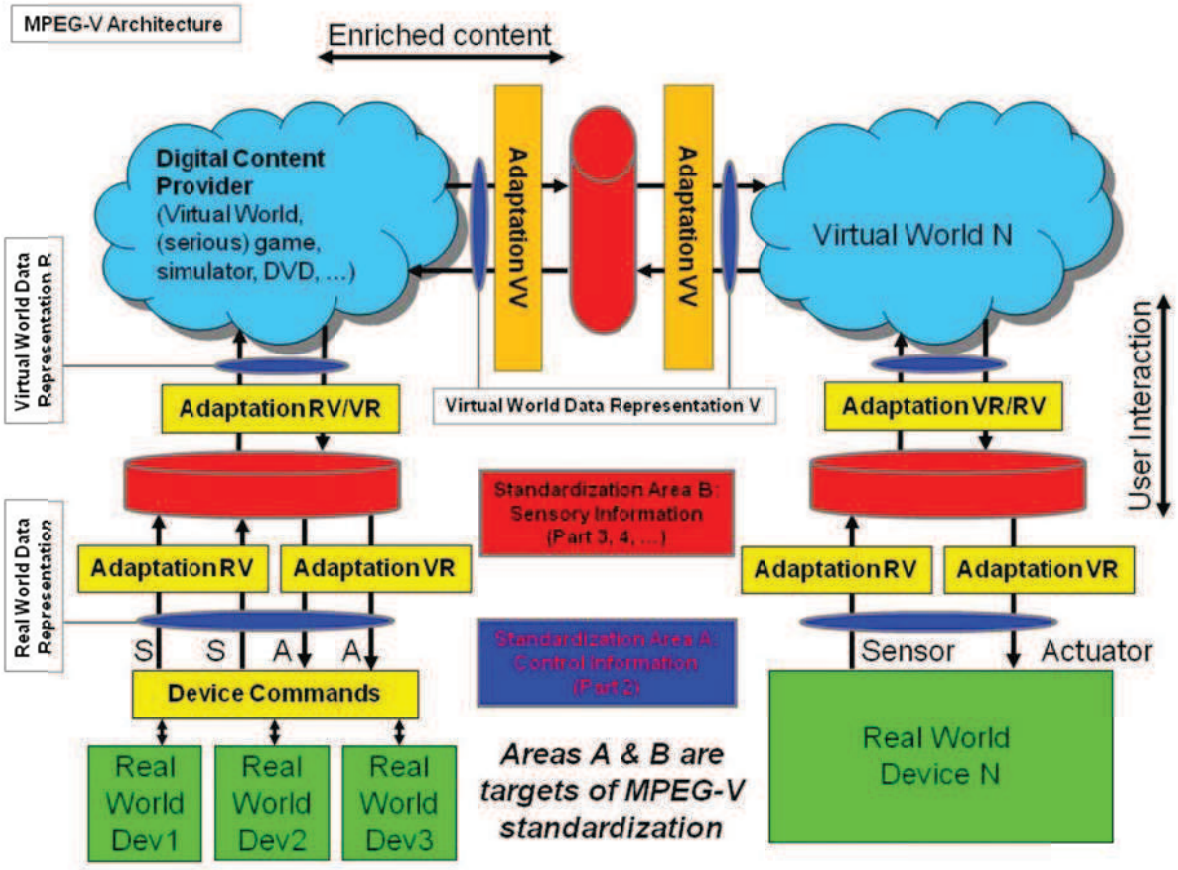


Figure 59. System Architecture of the MPEG-V Framework [ISO/IEC 20006-1].

The functions of the individual elements of the architecture are presented in Table 8.

Elements of the architecture	Function
Digital Content Provider	A provider of digital content, real time or non-real time, of various nature.
Virtual World Data Representation R	The native representation of virtual world related information that is intended to be exchanged with the real world.
Virtual World Data Representation V	The native representation of virtual world related information that is intended to be exchanged with another virtual world.
Adaptation RV/VR	The adaptation of the native representation of virtual world related information to the standardized representation format of MPEG-V in the area b) and vice versa.
Adaptation VV	The adaptation of the native representation of virtual world related information to the standardized representation format of MPEG-V in the standardization area b) in both directions.
Sensory Information	The standardized representation format of MPEG-V in

	the standardization area b).
Adaptation RV	The adaptation of the standardized representation of real world related information in the standardized representation format of MPEG-V in the standardization area a) to the standardized representation of virtual world related information in the standardized representation format of MPEG-V in the standardization area b).
Adaptation VR	The adaptation of the standardized representation of virtual world related information in the standardized representation format of MPEG-V in the standardization area b) to the standardized representation of real world related information in the standardized representation format of MPEG-V in the standardization area a).
Control Information	The standardized representation format of MPEG-V in the standardization area a).
Real World Data Representation	The native representation of real world related information that is intended to be exchanged with the virtual world (either exported or imported).
Device Commands	Device commands is responsible for the adaptation of the native representation of real world related information to the standardized representation format of MPEG-V in the standardization area a) in both directions: that is from the native representation into the standardized representation and vice versa.
Real World Device S	A real world device containing a sensor (e.g. a temperature, light intensity, blood pressure, heartbeat …)
Real World Device A	A real world device containing an actuator (e.g. a display, speaker, light speaker, fan, robot, implant …).

Table 8. Elements of the architecture and their function [ISO/IEC 20006-1].

In the framework of MPEG-V, in April 2009 we proposed an initial XML formalism describing a set of avatar characteristics dealing with appearance, animation and skeleton structure. This schema was a partial answer to the requirements of standardization area b, specifically its purpose was to enable the *avatar information adaptation between different virtual worlds*.

MPEG recognized the utility of the proposed model and accepted it for publication as part of the MPEG-V standard. Our schema was adopted as a basis for the avatar characteristics and further enriched by other MPEG members. In October 2009 we proposed an updated version of the "Animation" element introducing the reference to the external resources. Finally, in July 2010, we proposed the extensibility feature.

MPEG-V was finalized in November 2010, and the schema we proposed covers a large part of Part 4 of this standard.

In the following section, we provide an example of practical usage of the proposed schema, integrated in MPEG-V, as an intermediate layer for ensuring the interoperability between different VWs.

4.4.5 Mapping Features between Virtual Worlds

The proposed "Avatar" schema consists in a set of parameters used to control internal template of avatars in each virtual world in order to personalize it. With respect to the openness of each virtual world in terms of richness of the parameters that can be changed by the end user, it will be possible to obtain more or less the same appearance, animation and behavior in different VWs.

Second Life	MPEG-V	IMVU
Body height = 165 ……	<AvatarAppereance> <BodyHeight>165<BodyHeight/> ……	N/A=>IMVU should interpret this property internally
Abdomen ……..	<pelvis>abdomen<pelvis/> ……..	Pelvis ……..

Table 9. Example of MPEG-V mapping.

To ensure that an arbitrary VW is compliant with MPEG-V, it is required to implement a mapping between its own features' representation and the MPEG-V elements. Let us consider the example illustrated in Table 9. The template representation in Second Life defines, beside other parameters, that the current avatar height is 165 by specifying the parameter named "Body height", a parameter internally used in SL. This information is mapped in MPEG-V as the value for parameter "<BodyHeight>", a parameter standardized by MPEG-V. This property can be transferred on the template of IMVU avatar, who will attempt to find an appropriate element to match the property description. Since currently IMVU does not expose any parameter for body height, it is not explicitly possible to map this information. However, if IMVU has an internal mechanism to interpret the information, it can apply this parameter to its avatar template. In a second example, the mapping is straightforward: both VWs expose mapping parameter for the feature named "<pelvis>" in MPEG-V. Thus, the "abdomen" from SL will be mapped in "<pelvis>" attribute in MPEG-V and stored in "Pelvis" in IMVU.

Figure 60. MPEG-V communication between 3D VWs, only the color of the hair was used to transfer the appearance of the avatar from one VW to another.

Figure 60 illustrates this process. VWs A and B have different templates for "girl" avatars. By transferring some properties from the avatar created in VW A to the template of the one in VW B, the overall appearance may be preserved (here only the hair color was transferred).

4.5 Conclusion

In this chapter we presented an integrated solution that addresses the issue of accessibility and interoperability between virtual worlds, for the specific case of avatars. The solution is based on three contributions: a data model allowing the compression of generic 3D graphics, and specifically avatars, independent from the formalism used to represent them, an adaptation approach for weak terminals and a metadata model allowing to preserve the avatar characteristics when exchanged between VWs.

Concerning the first contribution, the proposed data model is able to accommodate a third party XML description of scene graphs with MPEG-4 3D graphics compression tools,

while preserving the compression features of the tools (i.e. progressive decompression, support for streaming). Among MPEG-4 tools, 3DMC was used for static mesh compression, BBA for the animation compression, while scene-graph was binarized by using gzip. The data model, as well as its software implementation, were proposed to MPEG and accepted as part of the MPEG-4 standard (MPEG-4 Part 25).

The second contribution provides an adaptation solution of an arbitrary avatar (composed of the object graph, the geometry, the appearance and the animation) for weak terminals. Compared to existent approaches, the proposed method, based on MPEG-4, demonstrates several key advantages including the creation of the 3D content dedicated for mobile devices reusing the high-resolution one, high rates of compression and streaming capabilities.

The third contribution makes possible the transfer of avatars from one VW to another. It is based on the definition of a set of properties that allows personalization of templates defined within the VW. "Teleporting" completely the avatars from one VW to another raises technical and non-technical issues. The first refers to the need of fine tuning the mesh resolution to ensure acceptable rendering performance and the second is related to the assets' protection.

5 Experiments and Validation

This chapter presents the validation of the three contributions described in the previous chapter. Firstly, we report the compression results obtained on a large database. The animation is completed with the visualization of the obtained avatar in an external player. Secondly, we show the quantitative and qualitative results obtained by running several experiments for adapting the avatars for weak terminals. Finally, we introduce an application that allows "transporting" avatars from one VW to another by means of their appearance and animation characteristics.

5.1 Introduction

In the previous chapter, three contributions concerning avatars in on-line, heterogeneous environments were presented: a data model for generic 3D graphics compression, adaptation techniques for weak terminals and an interoperability framework. In this chapter we introduce several experiments to validate the proposed approaches in different use case scenarios.

Firstly, we implemented the generic compression model and conducted extensive tests for 3D objects expressed by arbitrary XML formalisms. The software implementation, adopted by MPEG in the development of MPEG-4 Part 25 and currently publicly available under ISO license, is introduced and reports and analyses of the compression results on large database are outlined in the chapter.

Secondly, we present the results of the method proposed for handling avatars on mobile phones and, additionally, we describe a working prototype implementing the method and providing an innovative service of animated messages.

Finally, we present the benefits of the proposed interoperability framework by introducing the implementation of two use cases of avatar teleporting between virtual worlds. In the first one, the avatar characteristics are used to personalize an existing template, in the second, a full importer was implemented.

5.2 Validation of the Compression Data Model for Heterogeneous Structures

5.2.1 Software Architecture

The software implementation of the above introduced framework consists of two main modules: MP4XMLEncoder and MP4XMLDecoder. The model was implemented for three XML-based scene graph formats (COLLADA, X3D and XMT) and the elementary streams supported are 3DMC, BBA, OI, PI, CI, FAMC, JPG and JPEG2000.

The first module, MP4XMLEncoder, is illustrated in Figure 61 and consists of three main parts.

1) ”XMLParser” - This component deals with parsing original XML files, filling specific structures with data for mesh, animation, texture and creating a buffer with the rest of the XML file.

2) “MP4Encoder” - This component takes the structures created by the previous one as input, encodes them and creates buffers with standard MPEG-4 elementary streams. Furthermore, it deals with binarization of the content that is not detected as compressible by any MPEG stream.

3) “MP4Compositor” - This component multiplexes the previously created elementary streams and creates the MPEG-4 file.

The second module, MP4XMLDecoder, is illustrated in Figure 62 and consists of three main components.

1) “MP4Parser” - This component parses the MPEG-4 file and outputs buffers with standard MPEG-4 elementary streams.

2) “MP4Decoder” - This component decodes the elementary streams in structures for mesh, animation and texture and a buffer for the rest of the XML file

3) “MP4MergingMethods” - This component uses the decoded structures and reconstructs the XML file.

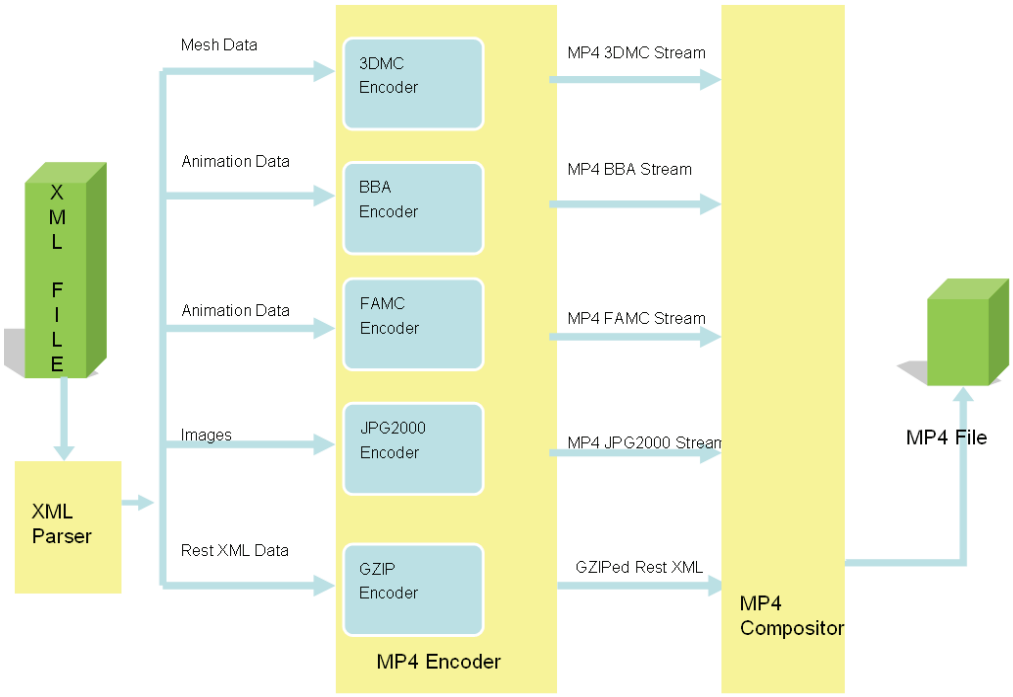


Figure 61. MP4XMLEncoder structure.

Since the scene and the object graph contain information that requires lossless compression, this data is compressed by using GZIP. On the other hand, the nature of such data and its dimension (it only contains structures and some parameters, but not geometry animation or texture) and its relatively small size make inappropriate the development of specific compression schema.

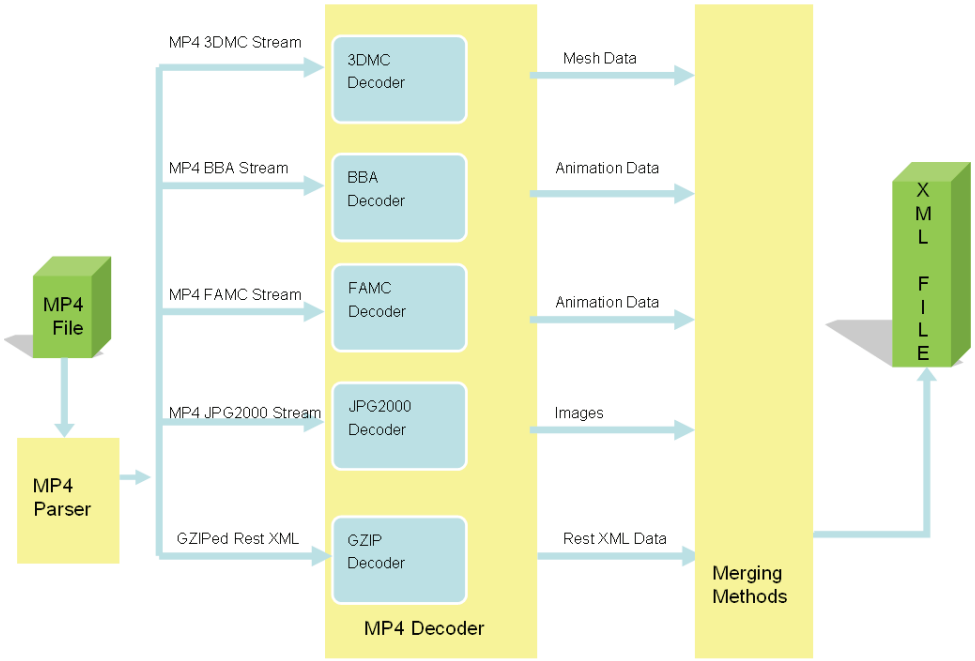


Figure 62. MP4XMLDecoder structure.

5.2.2 Compression Results

By using the software described in the previous section, we have run two sets of experiments. In the first one we have analyzed the performances for compression of the static mesh, and in the second one the performances for animation.

5.2.2.1 Static Mesh Compression Results

Due to the use of the quantization, the quality of the bit-rate in 3DMC may be controlled by the quantization step (QS) or its opposite measure, number of bits per component (b/c). The smaller the b/c (bigger the QS), the better is the obtained compression, but the distortion increases. We have analyzed these dependencies and investigated the minimum b/c needed to ensure the absence of visual difference between the original and the compressed content.

We have conducted the tests on a large database (more than 1700 3D objects) and we have compared 3DMC with the version of the XML content compressed with WinRAR. The choice of WinRAR in the experiment is based on the observation that it encodes better than gzip (about 5% better).

Table 10 and Figure 63 show the geometry compression results for two files (Redcycle and Horstake). The distortion corresponds to the Hausdorff distance and was computed by using the MESH tool[26]. Let us note that 3DMC is able to compress better (up to two times better) if the preservation of vertex and triangle ordering is not a constraint.

b/c	Distortion		3DMC/RAR with Vertices and Triangles Order Preservation	
File	Redcycle	Horstake	Redcycle	Horstake
6	1,385	1,588	16%	25%
7	0,703	0,794	17%	26%
8	0,370	0,398	20%	31%
9	0,186	0,190	20%	32%
10	0,090	0,096	23%	36%
11	0,046	0,047	24%	37%
12	0,023	0,023	28%	41%
13	0,011	0,012	29%	42%
14	0,005	0,005	32%	47%

Table 10. The dependence between the quantization step, compression gain and distortion for files Redcycle and Horstake.

[26] http://mesh.berlios.de/

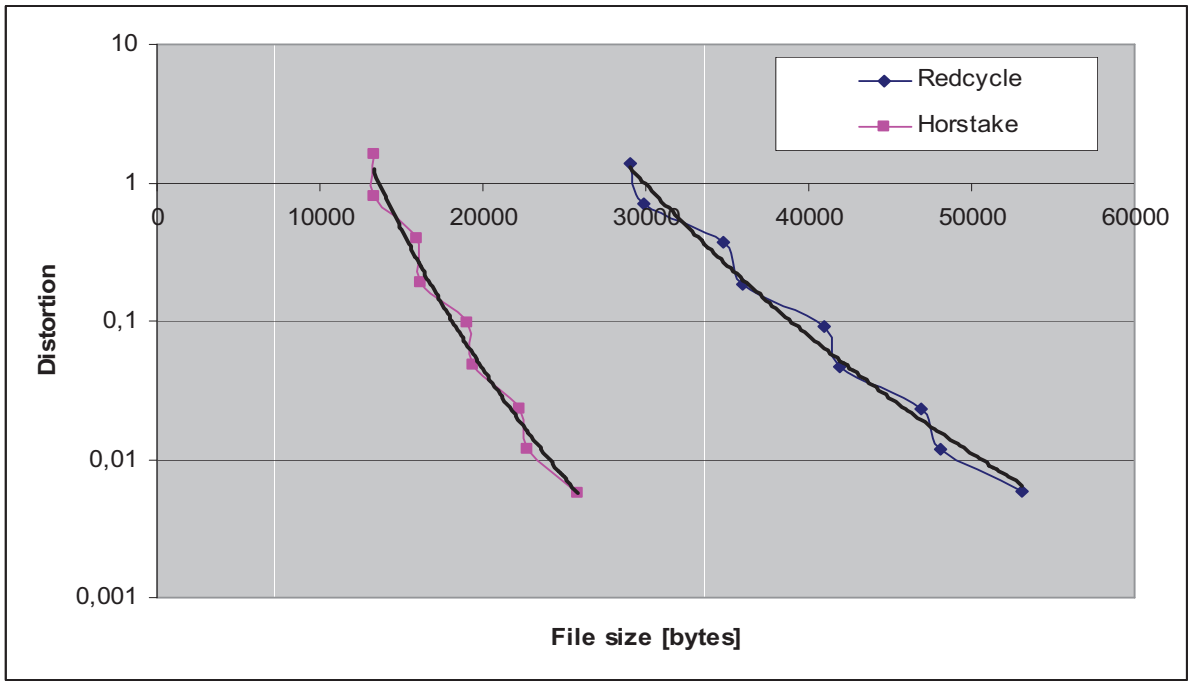
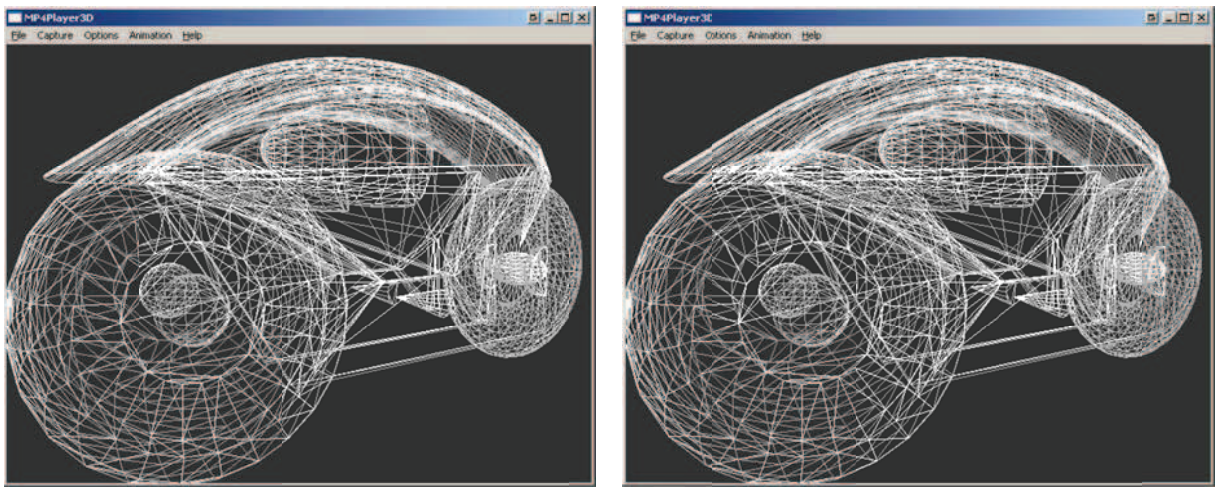


Figure 63. The dependence between the distortion and files size for files Redcycle and Horstake.

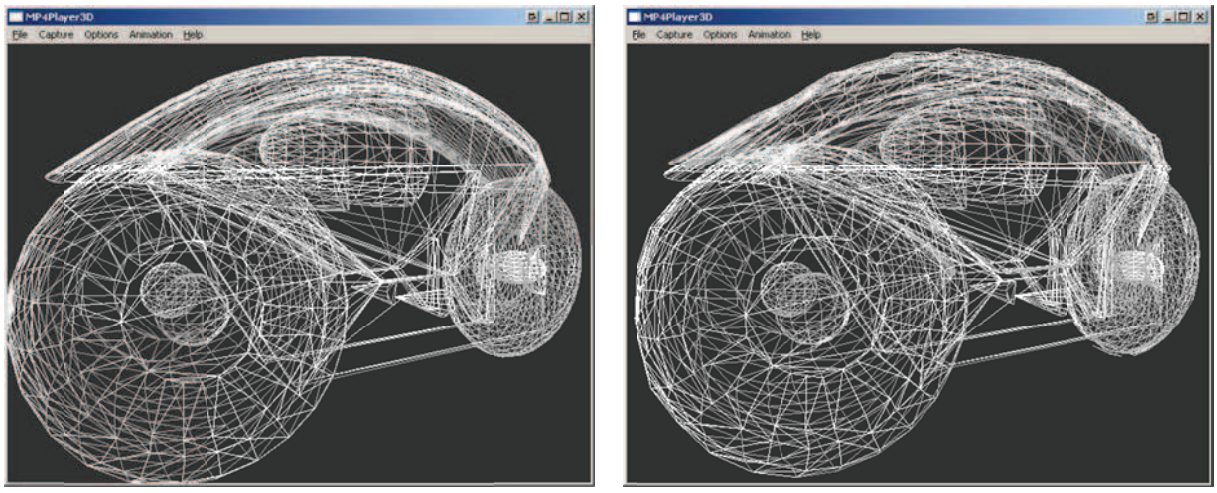
Based on a visual inspection, we noticed that b/c=9 is the threshold for almost all the files to keep the visual distortion low. We have run experiments with b/c=9 for the entire database and we have obtained an average compression of 2.86:1 with a standard deviation of 1.1, with regard to the original XML (in this case COLLADA) content compressed with WinRAR. This ratio increases to 6:1 if preserving the order of vertices and triangles is not a constraint.

Figure 64 shows an MPEG-4 player visualizing a static object encoded with different quantization steps.



a) original object

b) b/c = 14, compression ratio = 33%



c) b/c = 10, compression ratio = 24% c) b/c = 6, compression ratio = 15%

Figure 64. Visual results for compressed static objects.

5.2.2.2 Animation Compression Results

Two sets of experiments have been conducted for animated data: articulated models animated with BBA and generic deformed meshes by using FAMC.

- **BBA**

Similarly to 3DMC, BBA allows the control of the compression quality by using the quantization step. In order to identify the threshold QS for achieving visually equivalent results we have run the experiments for several files. Table 11 and Figure 65 show the dependencies between the compression ratio and distortion. The distortion is computed as the mean square error between the set of joints in the two skeletons [Preda07].

QS	Distortion		BBA/RAR	
File	Hero	Troll	Hero	Troll
32	9,4E-6	1,5E-4	34%	25%
128	2,1E-5	5,6E-4	24%	18%
512	6,6E-5	1E-3	20%	15%
1024	1,05E-4	5E-3	12%	12%
Max	6E-2	0,1703	6%	6%

Table 11. The dependence between the quantization step, compression gain and distortion for files Hero and Troll.

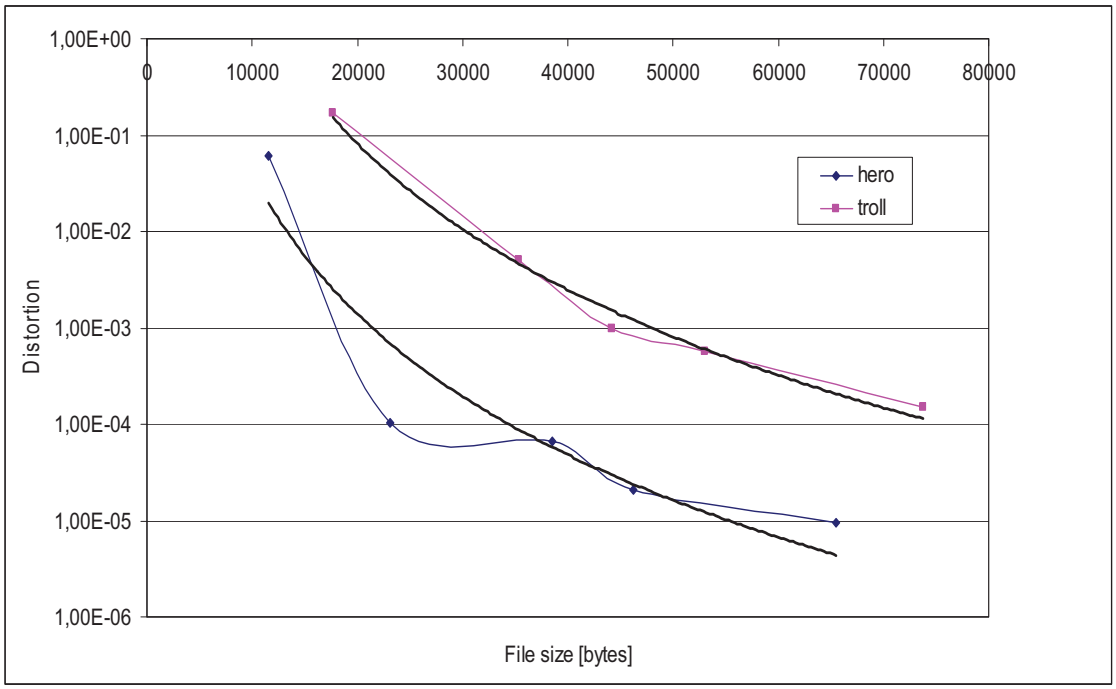


Figure 65. The dependence between the distortion and files size for Hero and Troll.

Let us note that for animations containing only rotations (the case of our database), using a quantization step of 1024 ensures compression without visual artifacts. We have carried out experiments for the entire database and we have obtained an average compression with respect to the WinRAR'ed COLLADA content of 14,6:1 and a standard deviation of 3,4. Some videos of the obtained results can be visualized on-line[27].

- **FAMC**

The control of the FAMC compression is done by selecting the number of bits per vertex (bpv) and the distortion is computed as the KG distance [Karni04], at each animation frame. Table 12 and Figure 66 show the dependencies between the compression ratio and distortion. We have carried out experiments for the entire database and we have obtained an average compression with respect to the WinRAR'ed COLLADA content of 3,9:1 and a standard deviation of 1,2. Let us note that FAMC exposes several encoding configurations. In our experiments we have used FAMC/DCT. A complete comparison between the different configurations is available in [Mamou08].

Bpv	**Distortion**		**FAMC/RAR**	
File	Rabbit	Sphere	Rabbit	Sphere
10	0,146536	0,126085	20%	10%
12	0,037678	0,03323	21%	10,7%
16	0,002736	0,002305	23%	11%
18	0,000707	0,000579	23%	11,4%
20	0,000178	0,000145	25%	11,6%

Table 12. The dependence between the number of bits per vertex, compression gain and distortion for files Rabbit and Sphere.

[27] www.MyMultimediaWorld.com

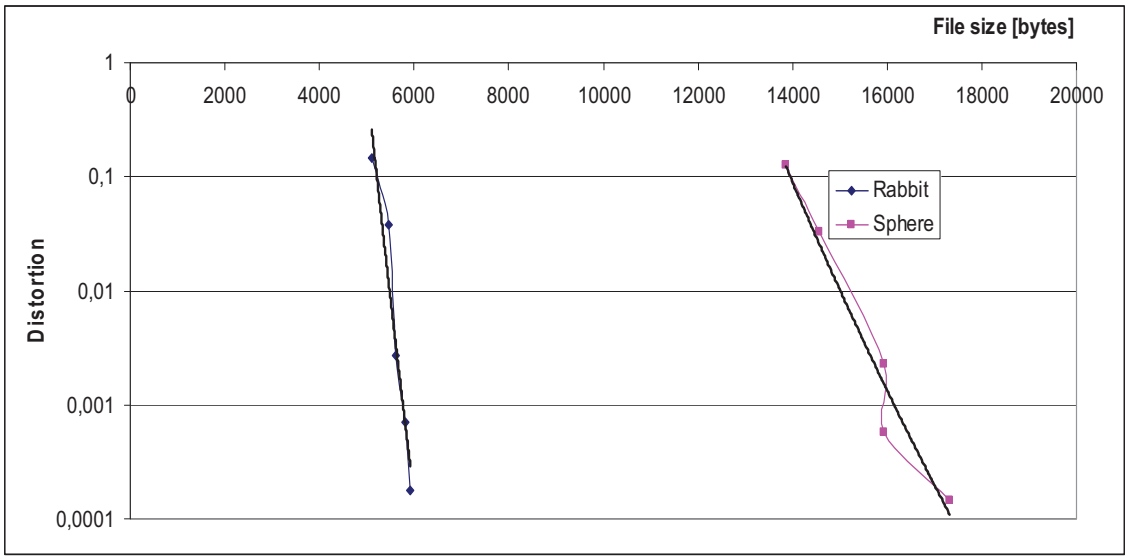


Figure 66. The dependence between the number of bits per vertex, compression gain and distortion for files Rabbit and Sphere.

1.1.1.1 Overall Compression Results

The general purpose of the proposed data model is to consider the 3D object entirely (geometry, appearance, animation, structuring elements) by performing a global compression. A compliant encoder should be lossless for the object graph and nearly lossless or lossy for the geometry, texture and animation. We have conducted a series of tests in order to compare the size of the initial XML files encoded by using WinRAR with the size of those encoded with the MPEG-4 tools. Let us note that we used quantization steps that produce no visual distortion. The results for different types of 3D objects are provided in Table 13.

File name	Type	Original structure	Original file size (Kb)	MPEG-4 Elementary Streams	MP4 file size/ XML RAR file size
Fireplace	Static mesh	XMT	44,7	3DMC	52%
Fireplace	Static mesh	X3D	50,8	3DMC	44%
Fireplace	Static mesh	COLLADA	158,8	3DMC	79%
Flat-monitor	Static mesh	XMT	73	3DMC	72%
Flat-monitor	Static mesh	X3D	83,7	3DMC	63%
Flat-monitor	Static mesh	COLLADA	299	3DMC	46%
Eagle	animation with bones	COLLADA	487	3DMC and BBA	57%
Rabbit	animation with bones	COLLADA	256	3DMC and BBA	57%
Wolf	animation with bones	COLLADA	783	3DMC and BBA	57%
Rabbit	animation with	COLLADA	2997	3DMC and	85%

	vertex updates			FAMC	
Eagle	animation with vertex updates	COLLADA	4850	3DMC and FAMC	79%
Wolf	animation with vertex updates	COLLADA	17736	3DMC and FAMC	71%
twist1	animation with vertex updates	COLLADA	2707	3DMC and FAMC	90%
Sphere	animation with vertex updates	COLLADA	3374	3DMC and FAMC	95%

Table 13. Compression results for several configurations.

The first observation is the big size of the COLLADA files when compared to X3D and XMT versions of the same object. This does not come from the differences at the geometry or animation level, but from the scene graph level (default elements for lighting, special effects are present in the COLLADA file). This property decreases the amount of data processed by the MPEG-4 encoders. Therefore, the new assets of the encoder are less important with respect to initial file size, as they reduce the impact of specialized geometry and animation coding. For animated objects, when the bone-based representation is used, it leads to very compact animation bitstreams. If the motion is not based on a skeleton (like in Twist1 and Sphere objects), FAMC is used. Globally, when the object is articulated, by using MPEG tools a benefit over RAR of 2:1 is obtained, whereas when FAMC is used the benefit is smaller.

We ran the above-presented tests on the entire database (for description of the dataset and examples, the reader is invited to consult www.MyMultimediaWorld.com, an online database that contains all the files used in this evaluation). The overall gains are presented in Table 14.

Original structure	**MP4 file size/ XML RAR file size**
XMT	61%
X3D	52%
COLLADA static	64%
COLLADA BBA animated	53%
COLLADA FAMC animated	84%

Table 14. Overall compression results for the entire database in different configurations.

The experiments on the entire database allow us to conclude that the MPEG-4 tools for elementary streams (3DMC, BBA and FAMC) bring noticeable additional compression compared to RAR (up to 3:1 for geometry and 14:1 for animation) while keeping the visual quality. For 3DMC, if visual artifacts are allowed (for example when the object is displayed on small screens) the ratio can increase up to 7:1. However, when considering the entire object (including the structuring information and scene graph), the compression ratio of MPEG-4 over RAR decreases to up to 2:1, mainly because the quantity of pure geometry or animation information becomes smaller than the rest of the information present in the file.

5.2.3 Compression Results for Avatars

We have run several experiments by using the software implementation described above applied on a content representing avatars of various complexities. The obtained results are presented in Table 15.

File	Characteristics	Original file size (Kb)	Elementary streams inside the MPEG-4 file	MP4 vs. XML RAR ratio
Fitting	static avatar, mesh	4802	3DMC	37%
Ludwig	static avatar, mesh	270	3DMC	80%
Orc	static avatar, mesh	22	3DMC	50%
Zeb	animated avatar, mesh, skeleton	442	3DMC, BBA	63%
Eagle	animated avatar	487	3DMC, BBA	57%
Rabbit	animated avatar	256	3DMC, BBA	57%
Wolf	animated avatar	783	3DMC, BBA	57%
Hero	animated avatar	892	3DMC, BBA	53%

Table 15. Compression results for avatars.

Globally, for an avatar compressed by the proposed framework, a benefit over RAR of 2:1 is obtained while preserving the visual quality. This corresponds to a ratio of 20:1 with respect to the original content (textual format).

5.2.4 Performances of Optimized Animation Compression Method

To improve the animation compression results we developed an optimized Bone-Based Animation (BBA) Encoder. To ensure its quality, but also to study the influence of the different parameters on the decoder, we have conducted experiments on the database available at mocap.cs.cmu.edu which consists in about 1700 motion capture files of different nature and complexity. We have first investigated the impact of the number of predicted frames (PbI) over the size of the compressed files and presented the dependency between PbI and file size for both compression methods. For the predictive-based method, we present the results for the two modes, optimized and standard (with and without transmission of the min, max values for the arithmetic encoder). As illustrated in Figure 67, computing min, max values allows to reduce the necessary bandwidth starting with PbI=10, transmission of the min-max being compensated by the near-optimal use of the arithmetic encoder when the number of the symbols became significant.

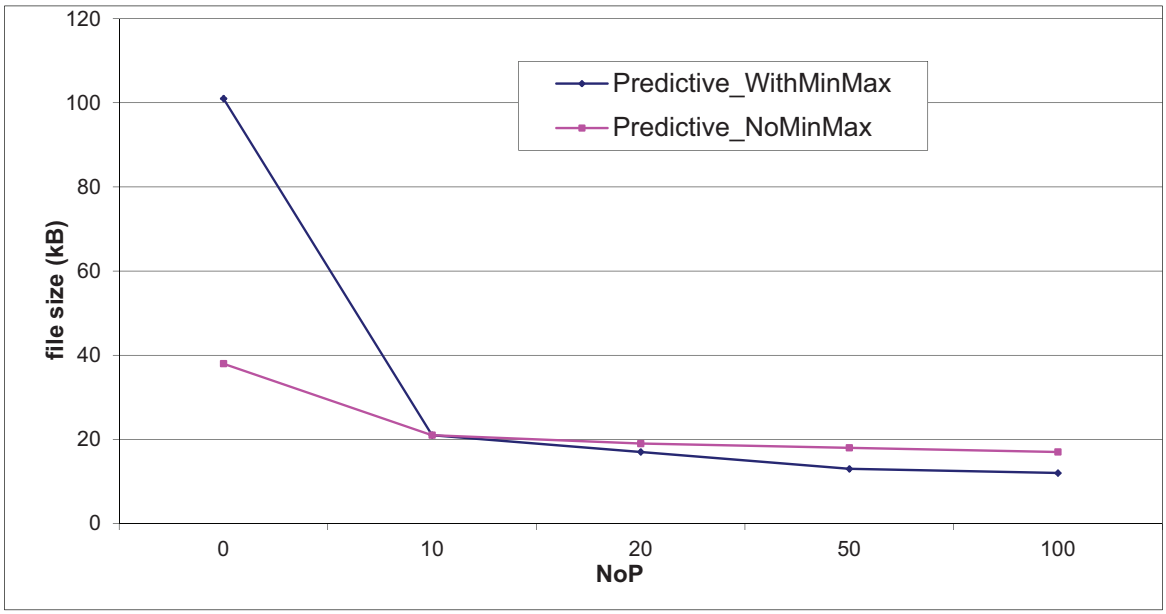


Figure 67. Dependency of the file size with respect to PbI for predictive-encoding.

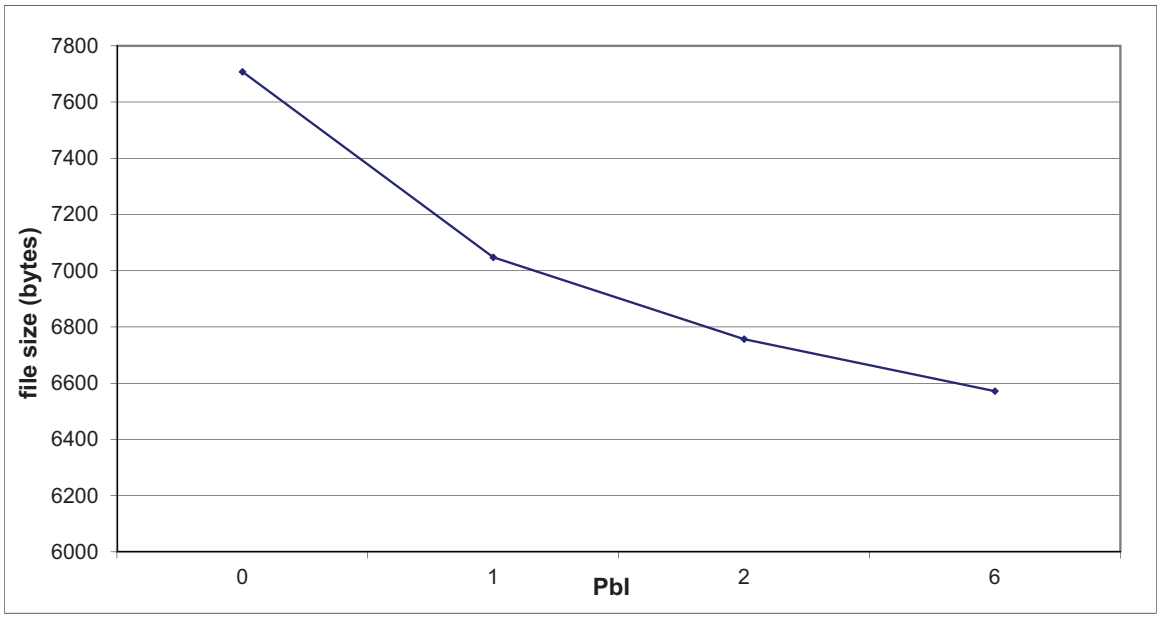


Figure 68. Dependency of the file size with respect to PbI (expressed in segments of 16 frames) for DCT-encoding.

The second phase of the experiments consists in controlling the quantization step. As expected, a larger quantization step (implying a smaller number of quantization levels) decreases the size of the compressed file. However, this conducts to a loss of precision in the signal reconstruction. In order to quantify the compression "loss" we introduce a distortion measure defined as the distance between the original vector and the reconstructed one (the vector is expressed in the bone elementary transformation space)

$$D = \sqrt{\frac{\sum_{i=0}^{N}\left(V_i^{\gamma} - V_i^{0}\right)^2}{N}} \qquad \textbf{(6)}$$

Where V_i^0 is the *i*-th component of the original vector containing all bone transforms and V_i^{γ} is the corresponding reconstructed value (after encoding and decoding).

Table 16 presents the compression results for motion capture files when using the predictive-based method with min, max transmission and 10 predictive frames between two intra frames. Let us note compression factors between 4 and 5 with respect to the original bvh file. When increasing the number of P frames between two intra frames, for similar distortion measure, the compression factor goes up to 10 (see Table 16).

File name	bvh file size [B]	Textual bba size [B]	Number of frames	Number of bones	Compressed bba file size [B]	Min QS	Distortion	Compression factor
02_01	254825	433074	343	29	51893	376	0,232	4,9
02_07	1753783	2877284	2251	29	335987	376	0,236	5,2
05_03	327338	555061	434	29	66651	376	0,250	4,9
08_05	222095	376298	298	29	45873	376	0,228	4,8
13_11	308329	522892	415	29	61127	368	0,227	5,0
13_39	265509	442116	351	29	54742	72	0,046	4,8
16_46	101441	172991	136	29	21247	352	0,214	4,7
49_05	124912	209249	164	29	25379	368	0,228	4,9
79_17	552310	1008671	804	29	116026	376	0,229	4,7

Table 16. Compression results for predictive-based encoding method (PbI = 10).

File name	bvh file size [B]	Textual bba size [B]	Number of frames	Number of bones	Compressed bba file size [B]	Min QS	Distortion	Compression factor
02_01	254825	433074	343	29	27500	376	0,267	9,2
02_07	1753783	2877284	2251	29	177116	376	0,245	9,9
05_03	327338	555061	434	29	38345	376	0,256	8,5
08_05	222095	376298	298	29	24828	376	0,229	8,9
13_11	308329	522892	415	29	30647	376	0,252	10,0
13_39	265509	442116	351	29	30959	124	0,083	8,5
16_46	101441	172991	136	29	12141	344	0,191	8,3
49_05	124912	209249	164	29	15209	344	0,222	8,2
79_17	552310	1008671	804	29	50555	376	0,244	10,9

Table 17. Compression results for predictive-based encoding method (PbI = 100).

For the DCT-based compression method, we present the file size results against the distortion. By increasing the quantization step, the file size decreases but the distortion increases. Table **18** presents the compression results for file 02_07.bvh (original bhv file has 1753783 bytes) with respect to quantization step and PbI. Both distortion and compressed file size are reported.

PbI	QS	32	50	100	200	320	382	640	800	960	1024
0	CS [kb]	232.5	188.2	147.3	114.4	96.9	91.2	77.3	72.5	69.1	68.0
	D	0,014	0,021	0,041	0,076	0,116	0,134	0,203	0,236	0,266	0,279
	CR	7,5	9,3	11,9	15,3	18,0	19,2	22,6	24,1	25,3	25,7
1	CS [kb]	221.2	182.4	138.7	103.1	84.0	77.8	62.6	57.3	53.4	52.2
	D	0,014	0,021	0,041	0,076	0,116	0,134	0,203	0,236	0,266	0,279
	CR	7,9	9,6	12,6	17,0	20,8	22,5	28,0	30,5	32,7	33,5
2	CS [kb]	217.4	180.5	135.9	99.2	79.7	73.3	57.6	52.1	48.2	46.9
	D	0,014	0,021	0,041	0,076	0,116	0,134	0,203	0,236	0,266	0,279
	CR	8,0	9,7	12,9	17,6	21,9	23,9	30,4	33,6	36,3	37,3
6	CS [kb]	213.4	178.5	132.8	95.2	75.0	68.4	52.2	46.5	42.4	41.0
	D	0,014	0,021	0,041	0,076	0,116	0,134	0,203	0,236	0,266	0,279
	CR	8,2	9,8	13,2	18,4	23,3	25,6	33,5	37,6	41,2	42,6
10	CS [kb]	211.9	177.6	131.6	93.8	73.4	66.7	50.4	44.7	40.5	39.1
	D	0,014	0,021	0,041	0,076	0,116	0,134	0,203	0,236	0,266	0,279
	CR	8,2	9,8	13,3	18,6	23,8	26,2	34,7	39,2	43,2	44,8

Table 18. DCT-based compression results. CS- compressed size, D – distortion, CR – compression ratio.

Let us note that for equivalent distortion with the predictive-based method, we obtain up to 45 compression factor. Figure 69 shows the dependency between the file size and the quantization step and the dependency between the distortion and the quantization step. Figure 70 plots the dependency between the bandwidth and the signal distortion.

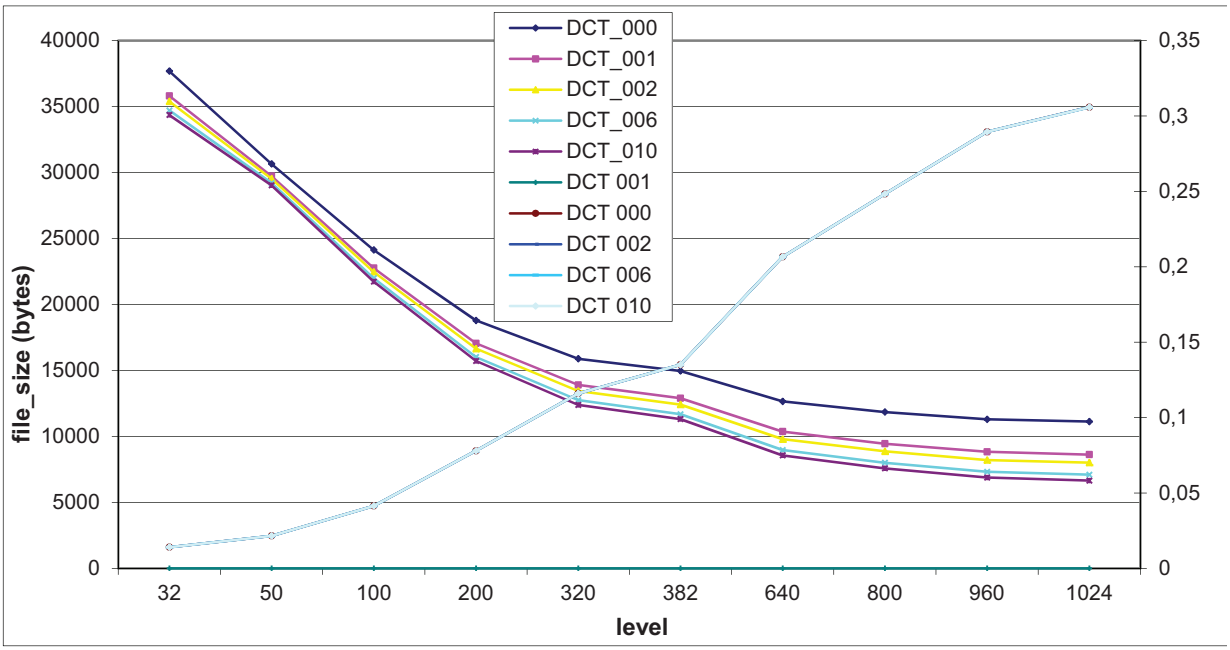


Figure 69. File size (left) and Distortion (right) dependencies with respect to the quantization step for the DCT-based method. Note that distortion is independent on the PbI parameter.

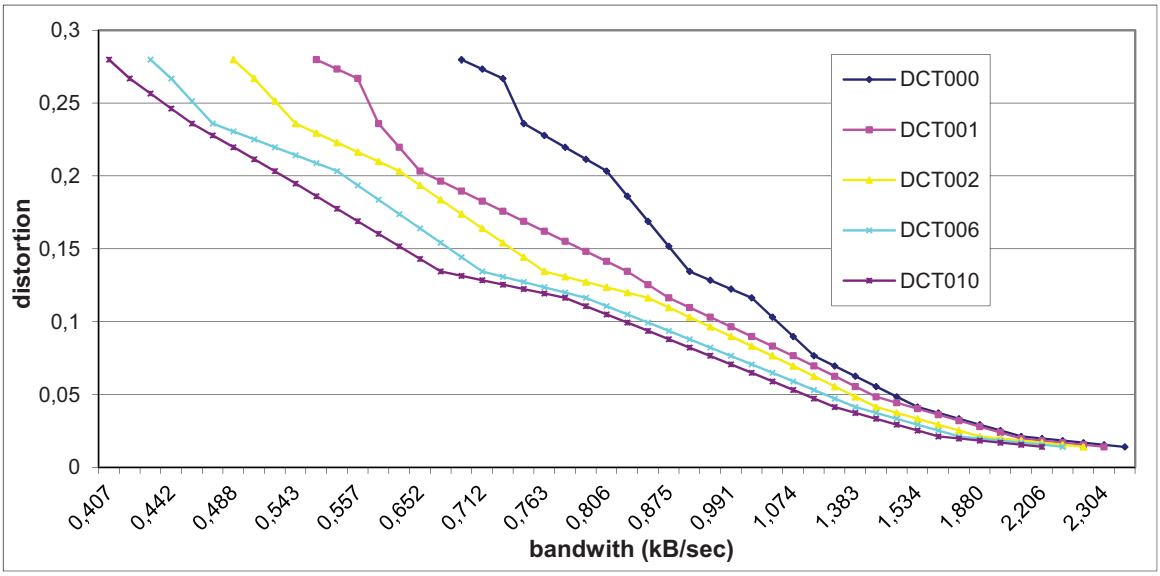


Figure 70. The dependency between the signal distortion and the bandwidth.

Similar dependencies are obtained for all the animation files. Figure 71 and Figure 72 show them for the file 79_17.bvh (original BHV file has 552310 bytes). Note a compression factor of up to 70:1.

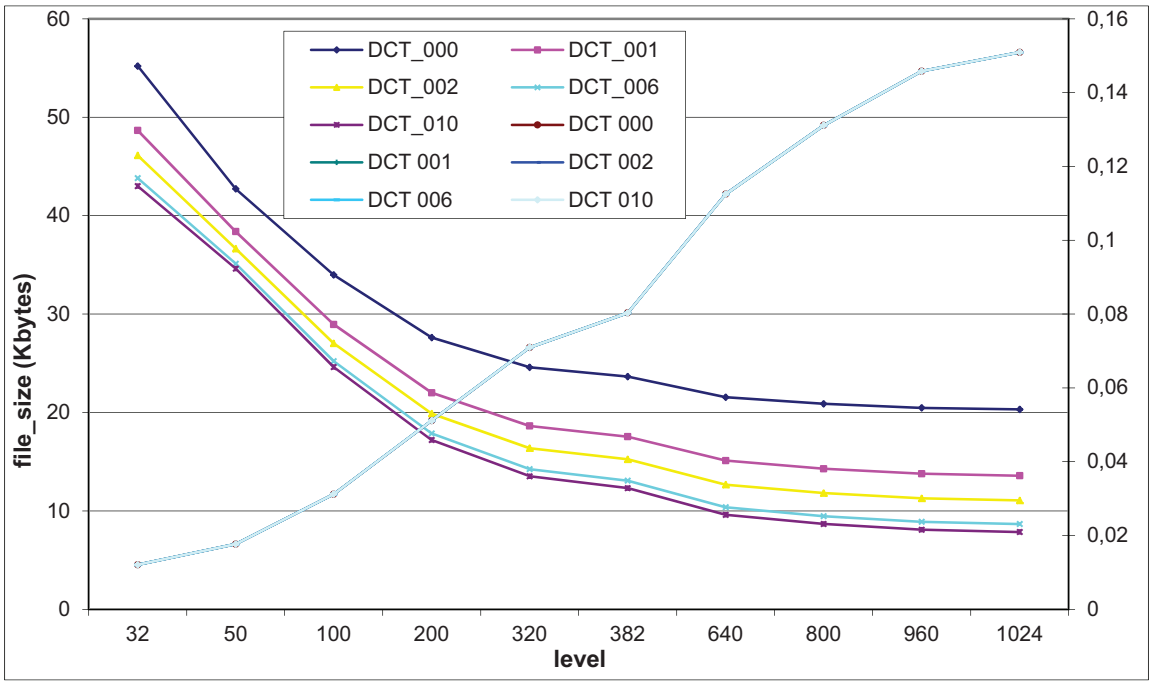


Figure 71. File size (left) and Distortion (right) dependencies with respect to the quantization step for the DCT-based method. Note that distortion is independent on the PbI parameter.

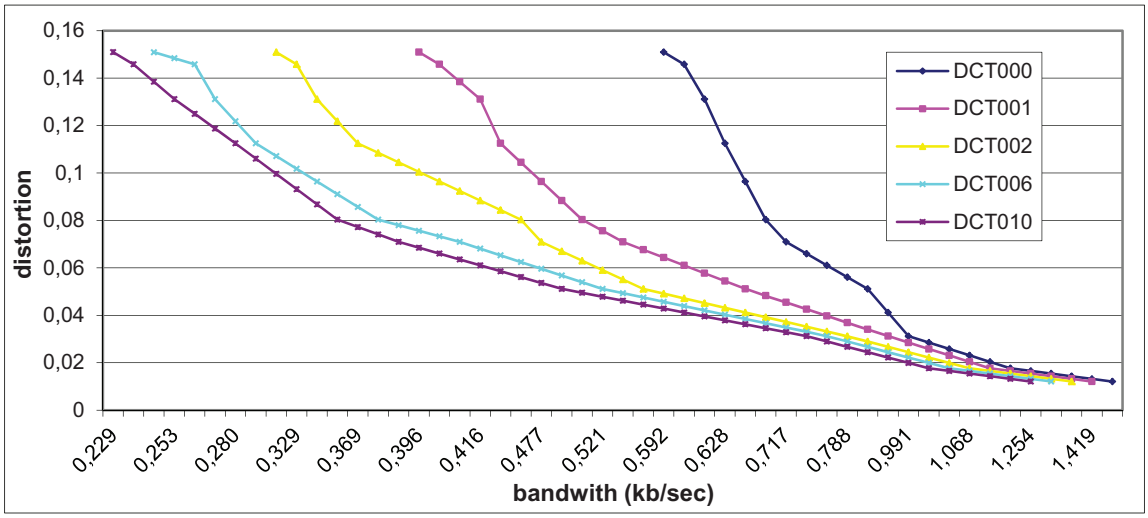


Figure 72. The dependency between signal distortion and bandwidth.

In order to reduce the animation file size more we implemented the frame reduction algorithm explained in the previous chapter. After the reduction of the frame we reencoded the animation by using the predictive-based method. Figure 73 and Figure 74 illustrate the dependency of the distortion with respect to the compression factor (all the computations are against the size of the original bvh file) for files 02_07.bvh and 79_17.bvh respectively.

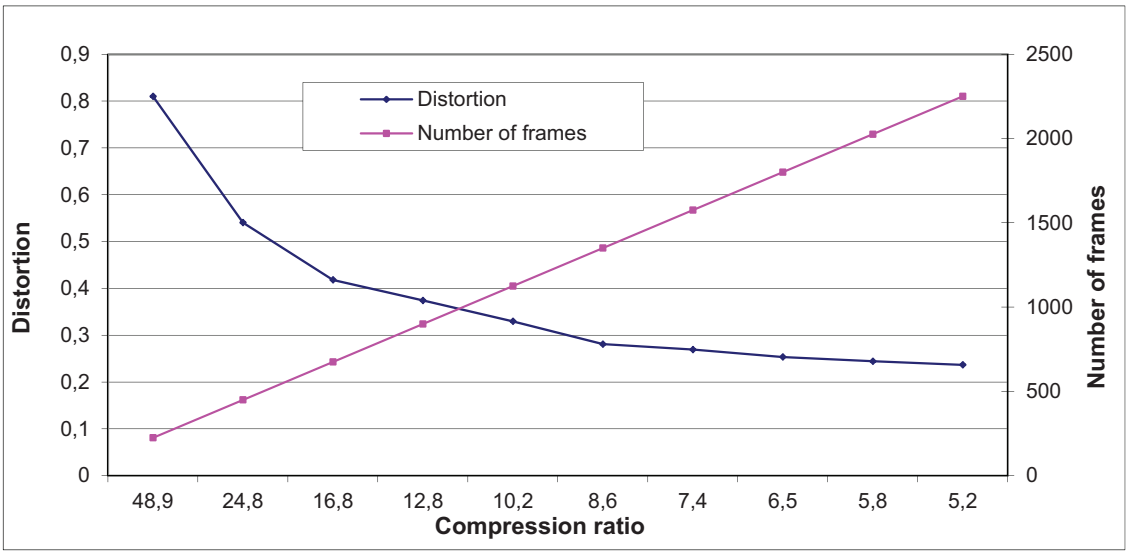


Figure 73. Compression by frame reduction: dependency of the distortion on the compression factor.

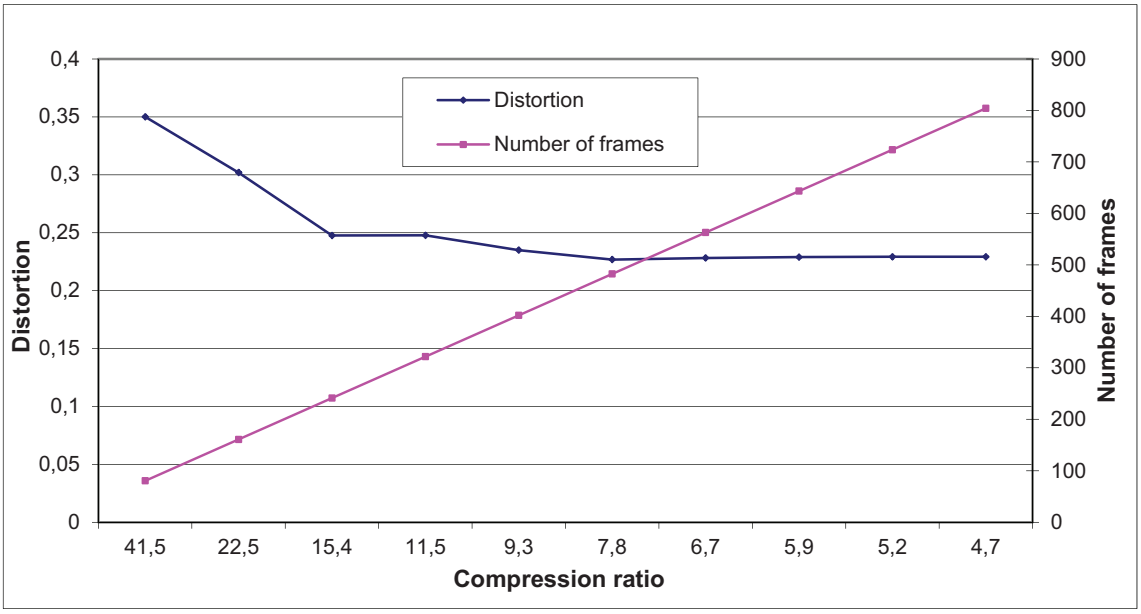


Figure 74. Compression by frame reduction: dependency of the distortion on the compression factor.

Nowadays, there is no consensus in the scientific community on the right distortion measure to use for quantitatively indicating the performances of an animation compression algorithm. In the case of static mesh compression, the most used distortion measure is the Root Mean Square Error based on Housdorff distance [Aspert02]. For animated meshes researches proposed an extension of RMSE, extension performed by integrating on all frames or by considering the maximum RMSE over all the frames. In the case of our study, animation of a skeleton without mapping between the bones and a mesh, distortion should be performed directly in the skeleton space. The distortions defined in the literature, and this is true for the one we introduced, give indications only on the behavior per frame but do not integrate aspects like motion variation (smoothness). These features are easily identifiable by human perception but much more difficult to quantify. For this reason, signal compression experiments include, in general, subjective tests. We initiated similar tests for our compressed data. All our results are available for online visualization on www.MyMultimediaWorld.com. This web site, used as

a repository for 3D data in general, has also the functionality of collecting user comments.

5.3 Validation of the Adaptation Tools for Weak Terminals

Figure 75 illustrates the process of transforming (adaptation and compression) of the avatar for weak terminals.

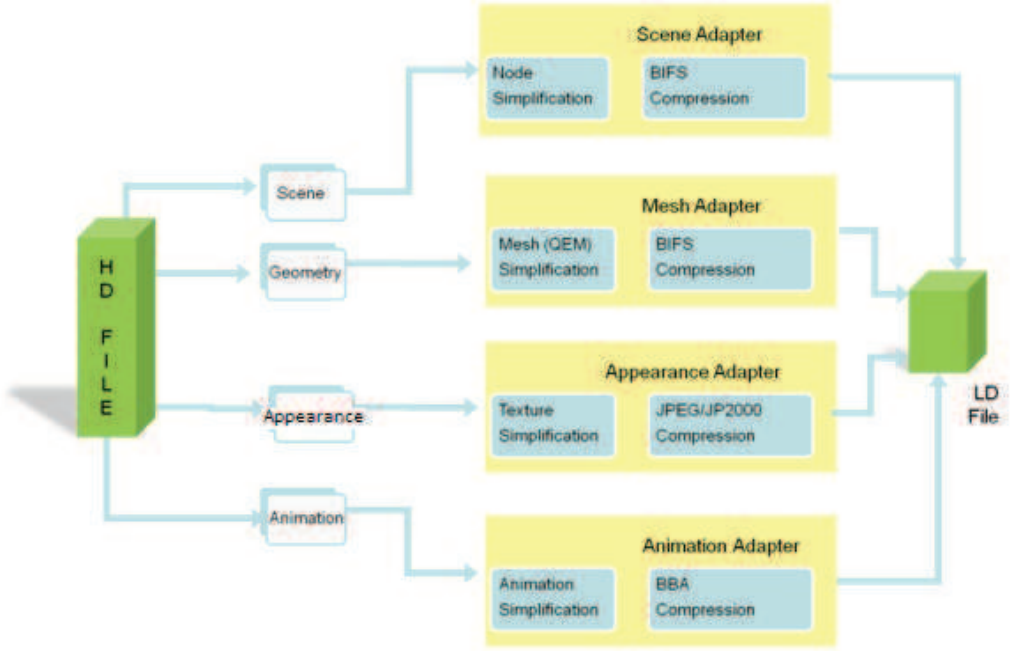


Figure 75. Data flow for adapting a 3D avatar on a weak terminal.

The 3D low-resolution data is obtained from the high-resolution representation by simplifying the four components: scene-graph, geometry, appearance and animation.

The scene-graph is simplified by maintaining only the nodes related to the avatar and it is compressed by using BIFS. The geometry, represented as a mesh, is simplified using the animation adapted QEM simplification method and compressed with BIFS. The texture is adapted by modifying the resolution and compressed with JPEG or directly encoded by JPEG2000, natively supporting multi-resolution representation. Finally, the animation is simplified by frame reduction and compressed with the proposed optimized BBA encoder.

A first validation of the framework was performed by running several experiments on a Nokia N95. The results are provided in Table 19 and the visual quality can be observed in Figure 76.

File Name	Number of triangles	FPS
hero.mp4	3512	49
Hero_HD.mp4	6846	32
Hero_LD.mp4	1874	58
troll.mp4	3940	41
raptor.mp4	3630	42

Table 19. Geometry simplification results and the corresponding fps.

a) Original model, 6846 triangles, frame-rate 32fps

b) Simplified model, 1874 triangles, frame-rate 58 fps

Figure 76. The original Hero and its simplified version.

A second validation of the proposed workflow was performed within a complex mobile phone application, called MESSAN and is presented in detail in the next section.

5.4 Implementation of the Interoperability Framework

The vast majority of Virtual Worlds and online games allow to design with a build-in editor the player's avatar. For this purpose they offer one or several avatar templates that the user can modify. The dedicated built-in tools containing a predefined set of default elements can be manipulated in order to personalize the avatar. The set usually includes different head and body shapes, skin color, position and design of the ears, eyes, nose and mouth. Additionally, some of them offer assets for dressing up the avatar: clothes, shoes, accessories etc.

Building an avatar from a template simplifies significantly the design phase, making it possible for unskilled users to create avatars. However, this task remains a time-consuming process, designing an avatar sometimes takes days, and usually it is a continuous process: the avatar can be upgraded at any time. Not having the possibility to reuse the designed avatar in other VWs obliged users to personalize their avatars in each VW separately. The proposed interoperability framework addresses this issue where the set of metadata characterizing the avatar can be used to personalize a pre-existent template.

5.4.1 Use Case Scenario

In order to validate the proposed framework for avatar interoperability, we performed the following scenario. The user creates one avatar in any VW that exposes its resources and characteristics. An example of avatar look that is modeled in the VW is presented in Figure 77.

Figure 77. The avatar in the original Virtual World[28]

By using the proposed schema, it is possible to map the avatar characteristic into standardized elements. These elements are imported in a second VW which "understand" this XML, therefore applying the personalization to its own avatar template. An example of the structure representing the current avatar and mapping according to the schema is provided in Figure 78. This mapping can be considered as an automation of the process of redesigning the avatar in the second VW. It is most probable that some characteristics will be lost or will not look exactly the same, due to differences in templates.

```
<?xml version="1.0"?>
<Avatar xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="location of the XSD schema">
    <!--Element AvatarAppereance is optional, maxOccurs=unbounded-->
    <AvatarAppereance>
        <!--Element Body is optional-->
        <Body>
            <!--Element BodyHeight is optional-->
            <BodyHeight>1.65</BodyHeight>
            <!--Element BodyThickness is optional-->
            <BodyThickness>1.23</BodyThickness>
            <!--Element BodyFat is optional-->
            <BodyFat>high</BodyFat>
            <!--Element TorsoMuscles is optional-->
            <TorsoMuscles>high</TorsoMuscles>
            <!--Element NeckThikness is optional-->
            <NeckThikness>1.23</NeckThikness>
            <!--Element NeckLength is optional-->
            <NeckLength>1.23</NeckLength>
            <!--Element Shoulders is optional-->
            <Shoulders>1.23</Shoulders>

……..
```

Figure 78. Example of an XML example that transfer avatar appearance characteristics.

A second useful capability exposed by the proposed schema is how animation can be triggered in different VWs. Let us consider the case where only characteristics are

[28] half-life2.com/

transferred therefore each VW uses its own resources for animations. If one's avatar "runs" in VW1, simply by sending a fragment of XML such as exemplified in Figure 79, will trigger "run" on the avatar in VW2. The two animations may be the same or slightly different, still the second world can understand the transmitted XML and perform the exact action.

```
<?xmlversion="1.0"?>
<xml>
    <Walk>
        <DefaultRun>
             <Url>run</Url>
        </DefaultRun>
    </Walk>
</xml>
```

Figure 79. Example of transfer animation feature of avatar.

Although the proposed schema can be used as a container for avatar characteristics, it can refer to external resources for avatar geometry, appearance and animation. Therefore, completely and exactly the same avatar can be "teleported" through VWs. The example given on Figure 80 contains a fragment of XML where the mesh is specified to be used from extern-defined resource, called "myavatar.mesh". The element <AppearanceResources > from the schema is reserved for that purpose.

```
<?xml                                                    version="1.0"?>
<xml>
        <AppearanceResources>
                <AvatarURL>myavatar.mesh</AvatarURL>
    </AppearanceResources>
</xml>
```

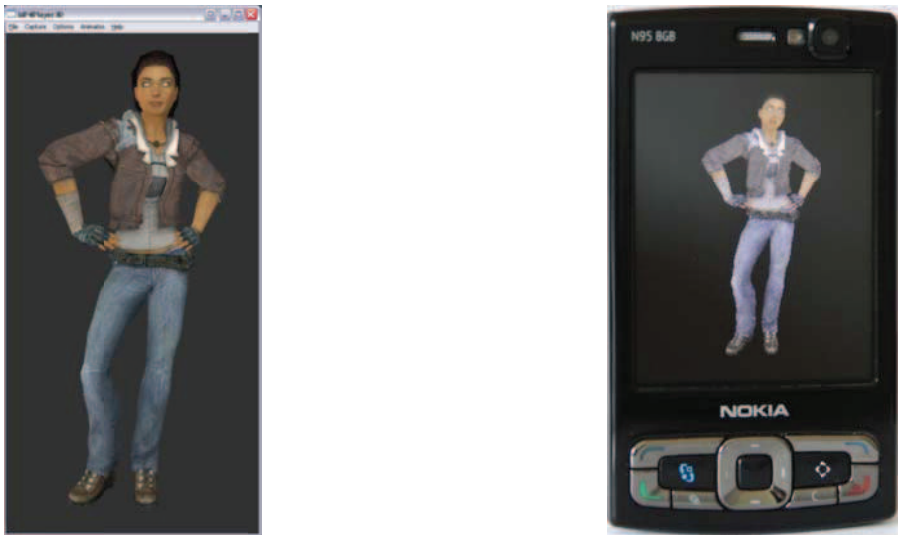
Figure 80. Example of transfer mesh of avatar.

In the same manner, animation can be loaded from an external resource (Figure 81).The element <AnimationResources> is used as a holder of animation resource.

```
<?xml                                                    version="1.0"?>
<xml>
        <AnimationResources>
                        <AnimationURL>  myanimation.url  </AnimationURL>
         </AnimationResources>
</xml>
```

Figure 81. Example of transfer animation resource of avatar.

However, usual implementations of VWs are not open to import not-self proprietary assets. In such a case, a possible usage is to visualize the avatar in external players. Since MPEG-V can refer to MPEG-4 resources for avatar geometry appearance and animation, an MPEG-4 player can be used for visualizing. Figure 82a) shows an avatar exported from a VW and visualized on a MPEG-4 player running on a standard desktop and Figure 82 b) the same avatar on a mobile phone platform.



a) b)

Figure 82. Avatar from Virtual World in MPEG-4 player on PC (a) and mobile phone (b).

5.5 MESSAN, a Prototype Validating the Proposed Avatar Framework

MESSAN, an abbreviation of "Message animé", aims at proposing an end to end solution to "pseudo-create" 3D graphics content on the mobile phone. Since mobile platforms are generally too weak to perform the high complexity computations needed for 3D graphics, a client-server infrastructure is used to create the content, adapt and send it in a mobile-phone-friendly form. On the sender side, a light JAVA application allows to select an avatar and manipulate it, select pre-recorded animations and environments and create new animations. The sender's mobile is assisted by a server database containing all the 3D assets, in both high and low resolution. Low resolution assets are created by adapting the high resolution ones and using the above proposed methods. Once the sender finalizes editing the scene, the parameters are transmitted to the server that produces a video stream by rendering a high resolution version of the scene. Finally, the video is transmitted to the receiver's mobile phone. By using a web interface, the users can upload, visualize and manage avatars, animation and backgrounds that can be used afterwards by the service. The proposed prototype was evaluated objectively (measuring data transmission, message production time …) and subjectively (by a panel of end-users) proving the technical feasibility and end-user acceptance of using avatars for message with emotional insight. In the next sub-section we first introduce the system architecture and the communication protocol; then we present the quantitative and qualitative results.

5.5.1 System Architecture, Modules and Protocols

The proposed system exposes two main functionalities: on one side it allows to 3D graphics artists to propose avatars, animations and scenes; on the other side it allows, the end-user to select any combination of avatars/animation/decors or record new motions in order to create the message.

- **Mobile client component**

This component is installed on the mobile phone and allows the end-user to compose the personal animated message. The scenario of creating the scene is as follows.

The application first presents the avatars available as a list of icons (each icon is about 5-10 Kb). The user can choose the preferred avatar and download it (50Kb to 200 Kb, depending on the avatar). Then several production scenarios are possible:

- the user can request an animation list (0.5 Kb for the request, 5-10 Kb for an animation snapshot), choose some of them for the scene (one downloaded animation is between 10-100 Kb);
- the user can control the avatar and record new animations;
- the user can request background lists and choose some (background images are 5-10 Kb)
- on each animation clip the user can add text that will be integrated like a subtitle.

Once the user finishes editing, all the parameters for composing the scene are sent to the server.

- **Application server**

Once receiving all the composition parameters, the server proceeds with the creation of the 3D scene. Figure 83 shows the main components implemented on the server.

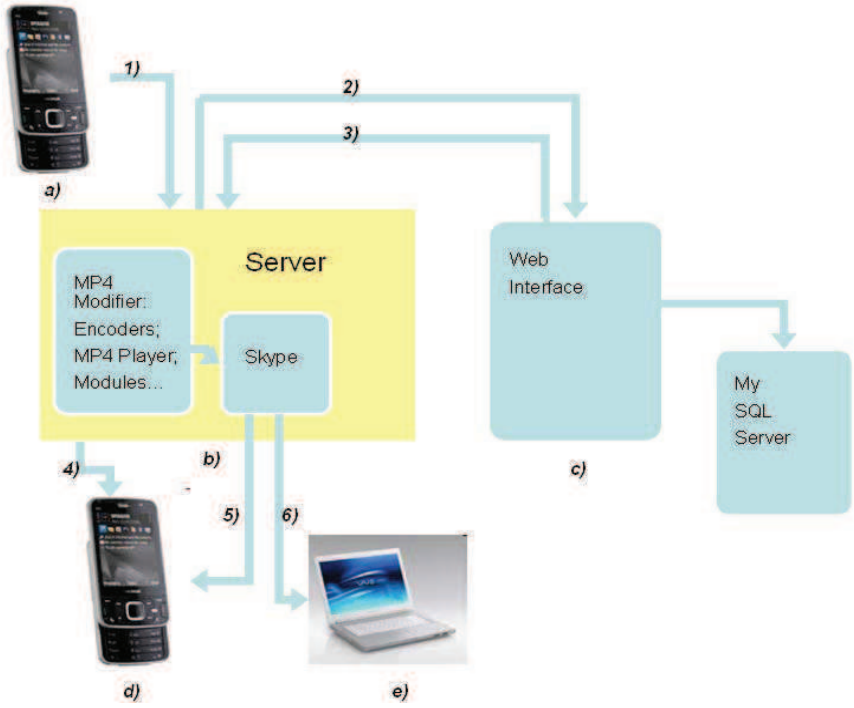


Figure 83. Server protocol for creating an animated message.

From the mobile phone (a) only a small amount of information is sent to the server (interface a.1): the type of request, the avatar's ID, background ID, animation IDs, the textual message, timestamps, destination number, etc. The server makes requests (b.2) to the web application (c) and obtains the corresponding graphics content from the database through interface (c.3). Depending on the request, the server can act in two ways:

- the low-resolution version of the graphical content is directly transmitted to the client (b.4), and/or
- additional modifications are performed on the server for a high-resolution version of the content: scene concatenation, 3D encoding, visualization, frame capturing and video encoding. Several output options are supported for the video stream: MPEG-4 ASP or Flash, with low or high compression quality. The

video stream is saved in the database and a reference is transmitted as an SMS (through Skype) to the mobile recipient (b.5) or as an e-mail (b.6).

5.5.2 Results

Figure 84 shows several snapshots of the mobile application and Figure 85 the user interface for artists proposing new content. One of the main barriers in using the mobile phone for advanced data communication application remains the cost per transmitted byte.

Figure 84. Snapshots of the mobile application.

Figure 85. Web-interface for content management.

By using only low-resolution models and synchronization between the client and the server through content IDs, we minimize the amount of data to be transmitted.

Table 20 shows different usage scenarios and the price in terms of total bandwidth and total production time. The tests are done with a list of 5 avatars, 5 animations per avatar and 5 backgrounds.

Number of animations	Using background	Using animation stored on the server	Data transfer [Kbyte]
1	No	No	160

1	Yes	No	205
1	No	Yes	240
1	Yes	Yes	300
3	No	No	160
3	Yes	No	205
3	No	Yes	335
3	Yes	Yes	380
10	Yes	Yes	750

Table 20. Transmitted data size in different scenarios.

If LD files are used, the creation time is from 3-10 sec. If HD files are used, the creation time is from 5-10 min.

With this approach we showed that, by adapting content properly, even devices with strong requirements and constraints such as the mobile phones can be transformed in powerful interfaces for 3D graphics auto-production.

5.6 Conclusion

This chapter presented different experiments performed to validate the three contributions of this work and the overall framework.

Firstly, we introduced a software implementation designed to perform compression on 3D objects expressed in arbitrary XML formalisms. To confirm the efficiency of the proposed compression framework, we conducted experiments over a large database, consisting of both static and animated files. Since we were using lossy (MPEG-4) methods for static and for animation data compressions, we provided both objective (by computation of the distortion between obtained and original 3D object) and subjective (by visualization of the decoded object in an external player) approaches to define the quality loss. We analyzed different conditions and investigated the optimal usage of each encoder in order to ensure no visual difference between the original and the compressed content. By using this setup for encoders, we reported compression results up to 30:1 (for geometry), 140:1 (for animation) and 20:1 for entire objects over the original textual content. This implementation was adopted by MPEG committee as reference software for MPEG-4 Part 25 and currently is publicly available under ISO license.

Secondly, we presented the quantitative and qualitative results of the method proposed for adapting avatars for weak terminals, obtained by running several experiments. We reported the visualization of an animated complex 3D avatar on Nokia N95 with up to 58 frames per second, while preserving visual quality. Furthermore, we demonstrated an implementation of an innovative communication service for short messaging in a mobile environment. In this working prototype, we showed that by using the proposed method for adapting the avatars for weak terminals and assisting the latter with a server, it is possible to transform them in a powerful interface for 3D graphics auto-production.

Finally, we presented the benefits of the proposed interoperability framework by introducing the implementation of two use cases of avatar teleporting between 3D scenes. In the first one, we introduced an application that allows “transporting” avatars from one VW to another VW by only transferring the avatar characteristics, used afterwards to personalize an existing virtual world template. In the second scenario, a full importer was implemented: the avatar media resources (geometry, texture, animation) were also transferred from the virtual world and the avatar was visualized in an MPEG-4 player.

Conclusion and Future work

In the last few years 3D Virtual Worlds (3DVWs) have achieved a great popularity by offering various functionalities and user experiences. Users become interested by VWs relatively fast and business research studies forecast that in the near future almost any Internet user will have his/her 3D representation (avatar) in one or more 3DVW.

Therefore, in this research, guided by the fact that the 3D VWs are online communities, we acknowledged the importance of the compact data transfer through various kinds of networks. To face the heterogeneity of data types for 3D graphics objects and in particular the avatars, we proposed a data model that allows efficient compression to be applied for individual data types (geometry, texture and animation) independently from the representation formalism.

Furthermore, the current development of mobile phones technologies makes these devices an access gateway to the 3DVWs. By their nature, primary designed as communication devices, they became terminals for the ultimate convergence of services and media access combining voice, text, images and video content. However, the content initially designed for powerful platforms should be adapted. This is the case as well for 3D graphics content, where once again its heterogeneous nature makes the operation more complex. Therefore, we proposed a method for adapting and delivering quality 3D avatars for mobile phones. While taking a photo or capturing video is a straightforward process, there are still no such devices for creating 3D content. We demonstrated that by combining the weak device with a server and by defining a light communication protocol, we transform the first in a powerful, yet easy to use content creation tool.

Finally, addressing the problem of user-generated content in virtual worlds, we propose a framework that includes a metadata layer allowing to carry on, when teleporting to a new virtual world, the avatar characteristics defined in the first one. This offers an elegant solution to the interoperability between VWs, in general highly proprietary applications.

The work presented in this manuscript was inspired and conducted within several standardization processes carried out by MPEG. Some of the solution proposed here were promoted in two MPEG standards, namely MPEG-4 Part 25 and MPEG-V.

We started this manuscript with a brief description of the needs for 3D content transmission and re-usability. These requirements are mainly driven by the recent development of 3DVWs.

Chapter 2 presented a survey of different approaches for avatar representation, as well as existing techniques for modeling and animating them. The avatar object-graph, the geometry, the animation and the appearance were discussed as avatar components. Additionally to this representation, a fifth element, called "metadata" and containing higher level avatar descriptions, was found to be one of the key-elements for a complete avatar description ensuring the necessary level of interoperability when the first four are proprietary defined. Different standards and representation formalisms used for avatar definition were introduced and several gaps were identified with respect to the reusability of avatars between different 3D Virtual Worlds. Firstly, considering that 3D VWs are online environments, the need of data compression was highlighted. The study of the existing methods shows major advancements for the individual components but lack in considering the holistic approach and especially the fact that in real applications the formalism of representing the avatar is application-dependent. Afterwards, the different methods for 3D graphics content adaptation for weak devices were presented. Today, integrating high performance graphics processors on mobile phones makes possible a rendering of high rates of pixels and triangles per second. Still, mobile phones have

specific constraints compared to ordinary computers, bringing considerable challenges for real-time 3D applications. Finally, we demonstrate that facing the increased number of 3D VWs, the interoperability becomes an issue. Therefore analyses of avatar representations in different VWs were presented highlighting the fact that the representation of the human body, emotions and animation, is VW specific.

Chapter 3 presents the main contributions of this thesis. Firstly, we proposed a data model allowing to connect compression tools for individual media to arbitrarily defined avatars. By analyzing the current approaches for compressing the key components of the avatar - object graph, geometry, appearance and animation - we selected a set, offering the optimal compromise between the compression ratio and the decoder complexity. We proposed a software architecture that allows the usage of these methods on any XML representation formalism representing an avatar. Instead of using generic compression methods for all the content (as provided by zip for example), the key elements are the automatic detection of individual media and the specific compression by the appropriate algorithms. The second contribution is a content adaptation solution for each of the avatar components: object-graph, geometry, appearance and animation, suitable for weak terminals. The development of this solution was driven by the compromise between the device screen resolution and content resolution (number of triangles for meshes, number of pixels for the textures, number of bones and frames for the animation). The third contribution addresses the avatar interoperability between VWs. Based on the most widely used approach in VWs for defining the avatars based on templates that can be personalized by the user, we derived a metadata model of the personalization parameters. Therefore, an arbitrary avatar can be represented as a function of the VW's private template together with a set of parameters that define the appearance, the animations and other specific data. Mapping this set of parameters from one VW to another is one approach to achieve avatar interoperability. Therefore, we proposed a method to represent them as associated "metadata", completing the avatar definition and used to reconstruct the arbitrary avatar.

In Chapter 4 we presented the experimental validation of the three contributions. The compression data model was implemented as a codec pair (encoder and decoder) and provides overall compression of 1:20 with respect to the XML representation of the avatar. Then, we presented numerically the benefits from the adaptation approach for weak terminals, by showing that we achieved animation with a frame rate of 40-50 fps[29], while preserving the visual quality. Finally, two conformance tests were presented for demonstrating the interoperability of avatars when used in virtual worlds.

The research presented in this manuscript comprises three contributions proposed to the general problem of the accessibility of virtual worlds, online by nature, with different user terminals. All contributions target the case of avatars and do not solve the complex problem of the software interoperability between virtual worlds, problem which is beyond the scope of the current work.

Maintaining the goal of interoperability at the level of avatars, some future work may still be considered. In the course of creating the lively and dynamic, densely inhabited future Internet, which will be 3D and mobile, further investigation is needed for improving the sense of social presence of the avatar in the VW and connecting it with the user profile.

The final goal of interoperability between virtual worlds is the definition of a common representation language, in the same way as html (now enriched with javascript) is providing for the classic Internet. The implications of such a language can be foreseen by analyzing the usage of Second Life (SL) and Facebook (FB) in the last seven years.

After SL was announced in June, 2003, its popularity grew very fast. However, at the moment of writing this manuscript, SL has lost the initial momentum and counts only around 70000 habitants. On the other hand, FB has today more than 500,000,000 users.

[29]http://en.wikipedia.org/wiki/Frame_rate#How_many_frames_per_second_can_the_human_eye_see.3F

A first question is why SL was initially so famous since the concept of VW was not new in 2003. The most probable answer is that the novelty in SL consisted in the social network capabilities. Unlike previous 3D VWs or 3D games, SL didn't specify a straight application or defined user roles. Any user creates its own personally, similarly to FB, but with the advantages of 3D graphics as a support of appearance representation. However, the second question is why SL, despite the existence of a better set of tools empowering the user in his/her expression, is now less successful than FB. Parts of the answer are the need to install a client, while FB is accessible by any browser, the need of powerful terminals, while FB is accessible by using mobile phones too, the poor interoperability with other applications, while FB exposes a set of APIs, allowing third-party applications to access easily user-generated content.

We strongly believe that the future of VWs is bright. Even if they develop more slowly compared to some 2D social networks, it is a fact that their number is increasing every year. The real challenge is to ease the access and create powerful tools to facilitate the content creation in order to transform today’s Internet into a 3D, social place.

References

[3dsmax] 3D Studio Max, Autodesk. Webpage: http://www.autodesk.com/3dsmax

[Agarwal06] Agarwal, A. And Triggs, B. (2006). "Recovering 3D Human Pose From Monocular Images".IEEE Trans. Pattern Anal. Mach. Intell. 28, 1, 44-58.

[Alexa00] Alexa M., Muuller W.(2000). Representing animations by principal components, Comput. Graph. Forum 19 (3), pp. 411–418.

[Alliez01] Alliez P., Desbrun M.(2001). "Progressive compression for lossless transmission of triangle meshes". In Proceedings of the 28th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '01. ACM, New York, NY, 195-202.

[Aminian98] Aminian, K., Andres, E. D., Rezakhanlou, K., Fritsch, C., Schutz, Y., Depairon, M., Leyvraz, P., And Robert, P. (1998). "Motion Analysis In Clinical Practice Using Ambulatory Accelerometry". In Proceedings Of The International Workshop On Modelling And Motion Capture Techniques For Virtual Environments N. Magnenat-Thalmann And D. Thalmann, Eds. Lecture Notes In Computer Science, Vol. 1537. Springer-Verlag, London, 1-11.

[Amjoun06] Amjoun R. , Sondershaus R. , Stra?er W.(2006). " Compression of Complex Animated Meshes", BOOK: Advances in Computer Graphics, Volume 4035/2006, Pages606-613, ISBN978-3-540-35638-7, September .

[Arafa03] Arafa, Y., And Mamdani, A. (2003). "Scripting Embodied Agents Behaviour with CML: Character Markup Language". In IUI '03: Proceedings of the Eigth International Conference on Intelligent User Interfaces, ACM, New York, NY, USA, 313–316.

[Arikan06] Arikan, O. (2006). "Compression of Motion Capture Database". ACM Transactions on Graphics. 890-897.

[Arkin94] Arkin E.M, Held M., Mitchell J.S.B. , Skiena S., (1994). "Hamilton Triangulations for Fast Rendering". In Proceedings of the Second Annual European Symposium on Algorithms (September 26 - 28, 1994). J. v. Leeuwen, Ed. Lecture Notes In Computer Science, vol. 855. Springer-Verlag, London, 36-47.

[Arsov09] Arsov I, Jovanova B., Preda M., Preteux F.(2009)"On-line animation system for learning and practice Cued Speech" , September 2009.

[Aspert02] Aspert N., Santa-Cruz D., and Ebrahimi, T. (2002). "Mesh:Measuring errors between surfaces using the hausdorff distance". In Proceedings of the IEEE International Conference in Multimedia and Expo (ICME) 1, 705–708.

[ATM] Ascension Technology Motionstar www.ascensiontech.com/Products/Motionstar/.

[Aubel00] Aubel A., Boulic R., Thalmann D. (2000). "Real-time Display of Virtual Humans:Levels of Detail and Impostors". IEEE Transactions on Circuits and Systems for Video Technology, 2:207-217, 2000.

[Augeri07] Augeri C.J, Bulutoglu D.A, Mullins B.E, Baldwin R.O and Baird L.C. (2007). "An analysis of XML compression efficiency". In Proceedings of the 2007 Workshop on Experimental Computer Science (San Diego, California, June 13 - 14, 2007). ExpCS '07. ACM, New York, NY, 7.

[Aviles08] Aviles M., Moran F., (2008). "3D triangle Mesh compression overview, 15th IEEE International Conference on Image Processing", 2008. ICIP 2008, p. 2684-2687

[Badler93] Badler N., Phillips C., and Webber B. (1993). "Simulating Humans: Computer Graphics, Animation, and Control". Oxford University Press.

[Badler95] Badler N., Metaxas D., Webber B. And Steedman M. (1995)." The Center for Human Modeling and Simulation", Presence 4, 1, 81-96.

[Bajaj96] Bajaj C.L., Coyle E.J., Lin K.-N. (1996). "Arbitrary topology shape reconstruction from planar cross sections". Graph. Models Image Process. 58, 6, 524-543.

[Bajaj99-01] Bajaj C.L, Pascucci V., Zhuang G., (1999). "Single Resolution Compression of Arbitrary Triangular Meshes with Properties". In Proceedings of the Conference on Data Compression , (March 29 - 31, 1999). DCC. IEEE Computer Society, Washington, DC, 247.

[Bajaj99-02] Bajaj C., Pascucci V., Zhuang G. (1999). "Progressive compression and transmission of arbitrary triangular meshes", in Proceedings of the 10th IEEE Visualization 1999 Conference (VIS '99) (October 25 - 28, 1999). VISUALIZATION. IEEE Computer Society, Washington, DC, pp. 307–316.

[Bartels87] Bartels R. H., Beatty, J. C., And Barsky, B. A. (1987). "An Introduction to Splines for Use in Computer Graphics & Amp; Geometric Modeling". Morgan Kaufmann Publishers Inc.

[Blanz99] Blanz, V., Vetter, T. (1999). "A Morphable Model For The Synthesis Of 3D Faces". Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, 187-194.

[Boston98] Boston Dynamics Inc. (1998). The Digital Biomechanics Laboratory, www.bdi.com.
[Briceno03] Briceno H.M., Sander P.V., McMillan L., Gortler S., Hoppe H., (2003) "Geometry videos: a new representation for 3D animations". In Proceedings of the 2003 ACM Siggraph/Eurographics Symposium on Computer Animation (San Diego, California, July 26 - 27, 2003). Symposium on Computer Animation. Eurographics Association, Aire-la-Ville, Switzerland, 136-146.
[Brooks02] Brooks, R., And Cagle, K. (2002). "The Web Services Component Model And Humanml". Technical Report, OASIS/Humanml Technical Committee.
[Chang02] C. Chang, S. Ger. (2002). "Enhancing 3D Graphics on Mobile Devices by Image-Based Rendering". Proc. IEEE Pacific-Rim Conf.on Multimedia, 2002.
[Chattopadhyay05] Chattopadhyay, S., Bhandarkar, S.M., Li K., (2005). "Virtual People & Scalable Worlds: Efficient Compression and Delivery of Stored Motion Data for Virtual Human Animation in Resource Constrained Devices". In Proceedings of VRST (Virtual Reality Software and Technology) Symposium, ACM, 235-243.
[Chen92] Chen, D. T. And Zeltzer, D. (1992). "Pump It Up: Computer Animation of a Biomechanically Based Model of Muscle Using the Finite Element Method". SIGGRAPH Comput. Graph. 26, 2, 89-98.
[Cheng03] Cheng,I. (2003). "Efficient 3D object simplification and fragmented texture scaling for online visualization," icme, vol. 2, pp.109-112, 2003 International Conference on Multimedia and Expo - Volume 2 (ICME '03).
[Chenimi08] Chehimi, F., Coulton, P., and Edwards, R. (2008). "Evolution of 3D mobile games development". Personal Ubiquitous Comput. 12, 1,19-25.
[Chou02] Chou P.H, Meng T.H. (Oct. 2002). "Vertex Data Compression through Vector Quantization". IEEE Transactions on Visualization and Computer Graphics 8, 4 ,(373-382).
[Chow 97] Chow M., (1997). "Optimized geometry compression for real-time rendering". In Proceedings of the 8th Conference on Visualization '97 (Phoenix, Arizona, United States, October 18 - 24, 1997). R. Yagel and H. Hagen, Eds. IEEE Visualization. IEEE Computer Society Press, Los Alamitos, CA, 347-ff.
[Clottes03] Clottes J.(2003). "Chauvet Cave: The Art of Earliest Times, University of Utah Press, Salt Lake City". 225 pages, hardbound. Translated by Paul Bahn from the French "La Grotte Chauvet, l'art de origins", Seuil, Paris, 2001.
[Cohen99] Cohen-Or D., Levin D., Remez O.(1999)." Progressive compression of arbitrary triangular meshes". In Proceedings of the Conference on Visualization '99: Celebrating Ten Years (San Francisco, California, United States). IEEE Visualization. IEEE Computer Society Press, Los Alamitos, CA, 67-72.
[Colin85] Colin A. R,Needham J. (1985). "The Shorter Science and Civilisation in China"; Volume 2. Cambridge University Press.
[D'Apuzzo99] D'Apuzzo, N., Plankers, R., Fua, P., Gruen, A., Thalmann, D. (1999). "Modeling Human Bodies From Video Sequences. Videometrics VI, SPIE Proceedings, Vol. 3461, San Jose, CA, 36-47.
[Declaro98] Decarlo, D., Metaxas, D., And Stone, M. (1998). "An Anthropometric Face Model Using Variational Techniques". In Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH '98.
[Deering95] Deering M., (1995). "Geometry compression". In Proceedings of the 22nd Annual Conference on Computer Graphics and interactive Techniques S. G. Mair and R. Cook, Eds. SIGGRAPH '95. ACM, New York, NY, 13-20
[Devernay94] Devernay, F. And Faugeras, O. D. (1994). "Computing Differential Properties Of 3-D Shapes from Stereoscopic Images without 3-D Models". In Conference on Computer Vision and Pattern Recognition, Seattle, WA, 208-213.
[Diepstraten04] J. Diepstraten, M. Gorke, T. Ertl. (2004) "Remote line rendering for mobile devices". Proc.Computer Graphics International CGI'04,454-461, 2004
[Dooley82] Dooley M. (1982). Anthropometric Modeling Programs –A Survey", IEEE Computer Graphics And Applications, IEEE Computer Society, 2, 9, 17-25.
[Duguet04] Duguet, F. and Drettakis, G. (2004). Flexible Point-Based Rendering on Mobile Devices. IEEE Comput. Graph. Appl. 24, 4 (Jul. 2004), 57-63.
[Edelsbrunner01] Edelsbrunner H.,(2001). "Geometry and Topology for Mesh Generation", Cambridge University Press, Cambridge.
[EML] Emotionml, http://www.W3.org/2005/Incubator/Emotion/XGR-Emotionml/.
[Endo03] Endo, M., Yasuda , T., and Yokoi, S. (2003). "A Distributed Multi-User Virtual Space System". IEEE Computer Graphics and Applications, 23, 1, 50-57.
[Evans96] Evans F., Skiena S., Varshney A. (1996). "Optimizing triangle strips for fast rendering". In Proceedings of the 7th Conference on Visualization '96 (San Francisco, California, United

States, October 28 - 29, 1996). R. Yagel and G. M. Nielson, Eds. IEEE Visualization. IEEE Computer Society Press, Los Alamitos, CA, 319-326.
[Faloutsos01] Faloutsos, P., Van De Panne, M., And Terzopoulos, D. (2001). "Composable Controllers For Physics-Based Character Animation". In Proceedings Of The 28th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH '01. ACM, New York, NY, 251-260.
[FARO] Faro Technologies, http://www.Farotechnologies.Com.
[Frank81] Frank T., Johnston O. (1981, reprint 1997). "The Illusion of Life: Disney Animation".
[Gandoin02] Gandoin P.M., Devillers O. (2002). "Progressive lossless compression of arbitrary simplicial complexes". ACM Trans. Graph. 21, 3 372-379.
[Garland97] Garland M.and Heckbert P.S. (1997). "Surface Simplification Using Quadric Error Metrics", ACM SIGRAPH 97 conference proceedings, p.209-216.
[Garland99] Garland M., (1999). "Quadric-based Polygonal Surface Simplification", PhD. Thesis.
[Geing98] Gieng T.S., Hamann B., Joy K.I., Schussman G.L., Trotts I.J. (1998). "Constructing Hierarchies for Triangle Meshes". IEEE Transactions on Visualization and Computer Graphics 4, 2, 145-161.
[Georges09] Georges, F. (2009). "Identite numerique et Representation de soi: analyse semiotique et quantitative de l'emprise culturelle du web 2.0". Reseaux, n° 154: Usages du Web 2.O. D. Cardon (direction). Paris : La decouverte, 2009. 165-193.
[Gersho92] Gersho A., Gray R.M.,(1992). "Vector Quantization and Signal Compression", Kluwer Academic Publishers, Dordrecht.
[Giacomo04-01]T. Di Giacomo, C. Joslin, S. Garchery, N. Magnenat-Thalmann. (2004). "Adaptation of Virtual Human Animation and Representation for MPEG". Computer &Graphics", 28(4):65–74, 2004.
[Giacomo04-02] T. Di Giacomo, H. S. Kim, S. Garchery, N. Magnenat-Thalmann, D. Cailliere, G. Belay, A. Cotarmanac'h, T. Riegel, (2004) "Benchmark-Driven Automatic Transmoding of 3D to 2D Talking Heads", Workshop on Modelling and Motion Capture Techniques for Virtual Environments (CAPTECH'04), December 2004.
[Gourret89] Gourret, J.-P., Thalmann, N. M., And Thalmann, D. (1989). "Simulation Of Object And Human Skin Deformations". In Agrasping Task. Computer Graphics (SIGGRAPH 89 Conference Proceedings) .
[Gross98] Gross J., J. Yellen J. (1998). "Graph Theory and Its Applications", CRC Press, Boca Raton.
[Gu02] Gu X., Gortler S.J., Hoppe H., (2002). "Geometry images", in: ACM SIGGRAPH, pp. 355–361.
[Gumhold98] Gumhold S., Stra?er W., (1998). "Real time compression of triangle mesh connectivity". In Proceedings of the 25th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '98. ACM, New York, NY, 133-140.
[Gumhold99] Gumhold S. (1999). "Improved cut-border machine for triangle mesh compression". In Erlangen Workshop 99 an Vision, Modeling and Visualization, pages 261--268, Erlangen, Germany.
[Gupta02] Gupta, S., Sengupta, K., and Kassim, A. A. (2002). "Compression of dynamic 3d geometry data using iterative closest point algorithm". Computer Vision and Image Understanding 87, 1-3, 116–130.
[Guskov04] Guskov I., Khodakovsky A. (2004), "Wavelet compression of parametrically coherent mesh sequences". In Proceedings of the 2004 ACM Siggraph/Eurographics Symposium on Computer Animation (Grenoble, France, August 27 - 29, 2004). Symposium on Computer Animation. Eurographics Association, Aire-la-Ville, Switzerland, 183-192.
[Heckbert86] Heckbert P. (1986), "Survey of Texture Mapping," IEEE Computer Graphics and Applications, November, pp. 56-67.
[Heeger95] Heeger D.J, Bergen J.R. (1995). "Pyramid-based texture analysis/Synthesis". In SIGGRAPH '95, pages 229–238.z[Hijiri00] Hijiri, T., Nishitani, K., Cornish, T., Naka, T., Asahara,.S. (2000). "A Spatial Hierarchical Compression Method for 3D Streaming Animation". In Proceedings of Web3D-VRML Symposium, ACM, 95-101.
[Hilton99] Hilton, A., Beresford, D., Gentils, T., Smith, R., And Sun, W. (1999). "Virtual People: Capturing Human Models To Populate Virtual Worlds". In Proceedings Of The Computer Animation (May 26 - 28,1999). CA. IEEE Computer Society, Washington, DC, 174.
[Hirose96] Hirose, M., Deffaux, G., & Nakagaki, Y. (1996). "Development Of An Effective Motion Capture System Based On Data Fusion And Minimal Use Of Sensors". VRST'96, ACM-SIGGRAPH And ACMSIGCHI,117-123.
[Hoppe96] Hoppe H. (1996). "Progressive meshes", in: ACM SIGGRAPH, 1996, pp. 99–108.

[Hoppe98] Hoppe H. (1998). ”Efficient implementation of progressive meshes”, Comput. Graph. 22 (1), pp. 27–36.
[Ibarria03] Ibarria, L., and Rossignac, J. (2003). “Dynapack: Spacetime compression of the 3d animations of triangle meshes with fixed connectivity”. In Proceedings of the 2003 ACM SIGGRAPH/ Eurographics symposium on Computer Animation, ACM 126– 135.
[Ishizuka00] Ishizuka, M., Tsutsui, T., Saeyor, S., Dohi, H., Zong, Y.,And Predinger, H. (2000). "Mpml: A Multimodal Presentation Markup Language with Character Agent Control Functions".
[ISO/IEC 14496/16] ISO/IEC JTC1/SC29/WG11, Standard 14496 16, A.K.A. MPEG 4 Part 16: Animation Framework Extension (AFX), ISO, 2004.
[ISO/IEC 14496/2] ISO/IEC JTC1/SC29/WG11, Standard 14496 2, A.K.A. MPEG 4 Part 2: Visual, ISO, 1999.
[ISO/IEC 19775-1] JTC1/SC29/WG11, Standard 19775-1, Published by ISO.
[ISO/IEC 14496-1] ISO/IEC JTC1/SC29/WG11, (1999). “Standard 14496-1”, a.k.a.”MPEG-4 Part 1: Systems”, Published by ISO.
[ISO/IEC 20006-1] ISO/IEC JTC1/SC29/WG11, (2010). “Standard 20006-1”, a.k.a.”MPEG-V Part 1: Architecture”, Published by ISO.
[Jang04] Jang, E.S., Kim, J.D.K., Jung, S.Y., Han, M.J., Woo,S.O., and Lee, S.J. (2004). “Interpolator Data Compression for MPEG-4 Animation”, IEEE Transactions on Circuits and Systems for Video Technology, 14, 7, 989-1008.
[Jayant84] Jayant N.S., Noll P. (1984). “Digital Coding of Waveforms—principles and Applications to Speech and Video”, Prentice Hall, New Jersey.
[Jovanova09] Jovanova B., Preda M., Prêteux F., (2009). “The Role of Interoperability in Virtual Worlds, Analysis of the Specific Cases of Avatars “, Technology, Economy, and Standards, Vol. 2(3), October 2009.
[Jovanova10] Jovanova B., Preda M., (2010). “Avatars interoperability in Virtual Worlds“, In Proceeding of the MMSP'10 2010 IEEE International Workshop on Multimedia Signal Processing , Saint-Malo, France, October 4-6, 2010 (to appear)
[Kahn85] Kahn D.W, (1995). “Topology: an Introduction to the Point-set and Algbraic Areas”, Dover Publications.
[KarniYEAR] Karni, Z. and Gotsman, C. (2000). Spectral compression of mesh geometry. In Proceedings of the 27th Annual Conference on Computer Graphics and interactive Techniques International Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., New York, NY, 279-286.
[Karni04] Karni Z, Gotsman C, (2004). “Compression of soft-body animation sequences”, Comput. Graph., Special Issue on Compression 28 (1), 2004, 25–34.
[Khodakovsky00] Khodakovsky, A., Schröder, P., and Sweldens, W. (2000). Progressive geometry compression. In Proceedings of the 27th Annual Conference on Computer Graphics and interactive Techniques International Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., New York, NY, 271-278.
[Kim00] Kim, M., Wood, S., and Cheok, L.T.(2000). “Extensible MPEG-4 Textual Format (XMT)”. Proceedings of the 2000 ACM workshops on Multimedia”, 71-74.
[King99] King D. and Rossignac J.,(1999). “Guaranteed 3.67V bit encoding of planar triangle graphs”, in 11th Canadian Conference on Computational Geometry, pp. 146-149.
[Kry02] Kry, P. G., James, D. L., And Pai, D. K. (2002). “Eigenskin: Real Time Large Deformation Character Skinning In Hardware”. In Proceedings Of The 2002 ACM Siggraph/Eurographics Symposium On Computer Animation (San Antonio, Texas, July 21 - 22, 2002). SCA '02. ACM, New York, NY, 153-159
[Lamberti03] Lamberti, F., Zunino, C., Sanna, A., Fiume, A., and Maniezzo, M. (2003). An accelerated remote graphics architecture for PDAS. In Proceedings of the Eighth international Conference on 3D Web Technology (Saint Malo, France, March 09 - 12, 2003). Web3D '03. ACM, New York, NY, 55-ff.
[Lander99] Lander J. (1999, May) “Over My Dead, Polygonal Body”. Game Developer Magazine, 1--4.
[Lee00] Lee E.-S., Ko H.-S.(2000).“Vertex Data Compression for Triangular Meshes”. In Proceedings of the 8th Pacific Conference on Computer Graphics and Applications (October 03 - 05, 2000). PG. IEEE Computer Society, Washington, DC, 225.
[Lengyel99] Lengyel J. E.(1999). “Compression of time-dependent geometry”. In Proceedings of the 1999 Symposium on interactive 3D Graphics (Atlanta, Georgia, United States, April 26 - 29, 1999). I3D '99. ACM, New York, NY, 89-95.
[Lewis00] Lewis, J. P., Cordner, M., and Fong, N. (2000). “Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation”. In Proceedings of The

27th Annual Conference on Computer Graphics and Interactive Techniques International Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., New York, NY, 165-172.

[Li98] Li J., Kuo C.-C.J.(1998). "Progressive coding of 3-D graphic models", Proc. IEEE 86 (6), 1998, pp.1052–1063.

[Maciel95] Maciel, P. W. and Shirley, P. (1995). Visual navigation of large environments using textured clusters. In Proceedings of the 1995 Symposium on interactive 3D Graphics (Monterey, California, United States, April 09 - 12, 1995). I3D '95. ACM, New York, NY, 95-ff.

[Maestri99] Maestri G. (1999)." Digital Character Animation 2: Essential Techniques". New Rider .

[Magnenat91] Magnenat-Thalmann, N., and Thalmann, D. (1991). Complex Models For Animating Synthetic Actors, IEEE Computer Graphics And Applications 11, 5, 32–44.

[Mamou08-01] Mamou, K.; Zaharia, T.; Preteux, F.; Stefanoski, N.; Ostermann, J.; (2008). "Frame-based compression of animated meshes in MPEG-4," Multimedia and Expo, 2008 IEEE International Conference on , vol., no., pp.1121-1124, June 23 2008-April 26 2008

[Mamou08-02] Mamou K, Zaharia T, Preteux F. (2008) "FAMC: the MPEG-4 standard for animated mesh compression" Proceedings IEEE International Conference on Image Processing (ICIP'2008), San Diego, CA.

[Mamou09] Mamou, K., Zaharia, T., and Prêteux, F. (2009). TFAN: A low complexity 3D mesh compression algorithm. Comput. Animat. Virtual Worlds 20, 2‐3 (Jun. 2009), 343-354.

[Massaro04] Massaro, D.W. (2004). "Symbiotic value of an embodied agent in language learning"; Proceedings of the 37th Annual Hawaii International Conference on System Sciences.

[Maya] Autodesk. Webpage www.Autodesk.Com/Maya

[Menache99] Menache, A. (1999). "Understanding Motion Capture for Computer Animation and Video Games". 1st.Morgan Kaufmann Publishers Inc.

[Moeslund06] Moeslund, T. B., Hilton, A., and Kruger, V. (2006). "A Survey of Advances in Vision-Based Human Motion Capture and Analysis". Comput. Vis. Image Underst. 104, 2, 90-126.

[Mohr03] Mohr, A. and Gleicher, M. (2003). "Building Efficient, Accurate Character Skins from Examples".ACM Trans. Graph. 22, 3, 562-568.

[Molet99] Molet, T., Boulic, R., and Thalmann, D. (1999). "Human Motion Capture Driven By Orientation, Measurements". Presence: Teleoper. Virtual Environ. 8, 2, 187-203.

[Monheit91] Monheit, G., and Badler, N. I. (1991). "A Kinematic Model of the Human Spine and Torso", IEEE Computer Graphics and Applications 11, 2, 29–38.

[MPEG-21] , ISO/IEC 21000-7 Final Committee Draft, ISO/IEC

[NASA78] NASA. (1978). Reference Publication 1024, The Anthropometry Source Book, Volumes I And II.

[NASA95] NASA. (1995). Man-Systems Integration Standard (NASA-STD-3000), Revision B.

[Nebel02] Nebel J., Sibiryakov A. (2002). "Range Flow from Stereo-Temporal Matching: Application to Skinning" ,In: Proceedings Of IASTED International Conference On Visualization, Imaging, And ImageProcessing.

[Nicolas04] Nicolas D, Andre Gaudreault A. (2004). "Heads or Tails: The Emergence of a New Cultural Series, from the Phenakisticope to the Cinematograph". Invisible Culture: A Journal for Visual Culture. The University of Rochester.

[Okuda06] Masahiro Okuda. (2006). Ishtiaq Rasool Khan, "Simplification of Texture in 3D Maps," aina, vol. 2, pp.678-682, 20th International Conference on Advanced Information Networking and Applications - Volume 2 (AINA'06).

[Oyarzun09] Oyarzun, D., Ortiz, A., del Puy Carretero, M., Gelissen, J., Garcia-Alonso, A., and Sivan, Y. (2009). "ADML: A framework for Representing Inhabitants in 3D Virtual Worlds". In Proceedings of the 14thinternational Conference on 3D Web Technology (Darmstadt, Germany, June 16 - 17, 2009). S. N.Spencer, Ed. Web3D '09. ACM, New York, NY, 83-90.

[Pajarola00] Pajarola R., Rossignac J. (2000). "Compressed Progressive Meshes". IEEE Transactions on Visualization and Computer Graphics 6, 1 , 79-93.

[Pajarola02] Pajarola R., Rossignac J. (2002). "Squeeze: fast and progressive decompression of triangle meshes", in: Proceedings of Computer Graphics International Conference, pp. 173–182.

[Pan09] Pan .Q, Reitmar G. and Drummond T. (2009) "ProFORMA: Probabilistic Feature-based On-line Rapid Model Acquisition", in 20th British Machine Vision Conference (BMVC'09), London

[Pandy90] Pandy, M. G., Zajac, F. E. (1990). "An Optimal Control Model For Maximum-Height Human Jumping". Journal Of Biomechanics, 23(12):1185–1198.

[Pandy99] Pandy, M.G and Anderson F.C. (1999). "Three-Dimensional Computer Simulation of Jumping and Walking Using the Same Model". In Proceedings of the Seventh International Symposium On Computer Simulation In Biomechanics.
[Payan05] Payan F., Antonini M. (2005). "Wavelet-based compression of 3d mesh sequences". In Proceedings of IEEE ACIDCA-ICMI'2005, Tozeur, Tunisia.
[Peng05] J. Peng, C.-S. Kim, C.-C.J. Kuo, (2005). Technologies for 3D mesh compres- sion: a survey, J. Visual Commun. Image Representation 16 (6) 688–733.
[Polhemus] STAR TRACK Motion Capture System, Http://Www.Polhemus.Com
[Preda02] Preda, M., and Preteux, F. (2002). "Critic Review on MPEG-4 Face and Body Animation". In Proceedings of International Conference on Image Processing, vol. 3, IEEE 505-508.
[Preda04] Preda M., Salomie A., Preteux F., Lafruit G. (2004). "Virtual character definition and animation" within the MPEG-4 standard.
[Preda05] Preda M., Tran S.M, Preteux F. (2005). "Adaptation of quadric metric simplification to MPEG-4 animated object" , Proceedings 2005 Pacific-Rim Conference on Multimedia (PCM'2005), Jeju, Korea - Lecture Notes in Computer Science 3767, November 2006, p. 49-60.
[Preda07] Preda M, Jovanova B, Arsov I, Preteux F, (2007). "Optimized MPEG-4 animation encoder for motion capture data", in Proceedings 12th International Conference on 3D Web Technology (Web3D'2007), Perugia, Italy, April 2007, 181-190.
[Preda08] Preda, M., Villegas, P., Moran, F., Lafruit, G., and Berretty, R. (2008). "A model for adapting 3D graphics based on scalable coding, real-time simplification and remote rendering". Vis. Comput. 24, 10, 881-888.
[Preda09] Preda M. (Ed.). (2009). Text of ISO/IEC CD 23005-4 Avatar Information, w10786, 89th MPEG Meeting,London.
[Requicha80] Requicha, A. G. (1980). "Representations for Rigid Solids: Theory, Methods, and Systems. ACM Comput. Surv. 12, 4 (Dec. 1980), 437-464
[Ronfard96] Ronfard R. Ronfard, J. and J. Rossignac. (1996). "Full-Range Approximation of Triangulated Polyhedra", Proceedings EUROGRAPIHCS, Computer Graphics Forum, 67-76.
[Rossignac94] Rossignac J (1994). "Specification, representation, and construction of non-manifold geometric structures". In Siggraph '94 Course Notes.
[Rossignac99] Rossignac J., Edgebreaker, (1999). "Connectivity Compression for Triangle Meshes". IEEE Transactions on Visualization and Computer Graphics 5, 1 47-61.
[S20SD] S20 Sonic Digitizers, Science Accessories Corporation.
[Sattler05] Sattler M., Sarlette R., Klein R. (2005) "Simple and efficient compression of animation sequences". In Proceedings of the 2005 ACM Siggraph/Eurographics Symposium on Computer Animation (Los Angeles, California, July 29 - 31, 2005).
[Savenko99] Savenko A., Van Sint Jan S.L. and Clapworthy G.J. (1999). "A Biomechanics-Based Model for the Animation of Human Locomotion", Proc Graphicon 99, Moscow, 82-87.
[Scheepers96] Scheepers, C. F. (1996). "Anatomy-Based Surface Generation for Articulated Models of Human Figures" Phd Thesis, Ohio State University, Adviser: Richard E. Parent
[Scheepers97] Scheepers, F., Parent, R. E., Carlson, W. E., and May, S. F. (1997). "Anatomy-Based Modeling of the Human Musculature". In Proceedings Of The 24th Annual Conference on Computer Graphics and Interactive Techniques. International Conference on Computer Graphics and Interactive Techniques.ACM Press/Addison-Wesley Publishing Co., New York, NY, 163-172.
[Schroeder92] Schroeder, W. J., Zarge, J. A., and Lorensen, W. E. (1992). "Decimation of Triangle Meshes". In Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques J. J.Thomas, Ed. SIGGRAPH '92. ACM, New York, NY, 65-70.
[Seo02] Seo, H., Yahia-Cherif, L., Goto, T., and Magnenat-Thalmann, N. (2002). "GENESIS: Generation of EPopulation Based on Statistical Information". In Proceedings of the Computer Animation (June 19 -21, 2002). CA. IEEE Computer Society, Washington, DC, 81.
[Seo03] Seo, H. and Magnenat-Thalmann, N. (2003). "An Automatic Modeling of Human Bodies from Sizing Parameters". In Proceedings of the 2003 Symposium on Interactive 3D Graphics (Monterey,California, April 27 - 30, 2003). I3D '03. ACM, New York, NY, 19-26.
[Shi07] Shi, X., Zhou, K., Tong, Y., Desbrun, M., Bao, H., Guo, B. (2007). "Mesh puppetry: cascading optimization of meshdeformation with inverse kinematics". ACM Trans. Graph. 26, 3,81.
[Singh95] Singh, K. (1995). "Realistic Human Figure Synthesis And Animation For VR Applications." Phd Thesis,The Ohio State University. Adviser: Richard E. Parent.
[Sloan03] Sloan, P.-P., Hall, J., Hart, J., and Snyder, J. (2003). "Clustered principal components for precomputed radiance transfer". In Proceedings of SIGGRAPH 2003, 22, 3, 382–391.

[Smith06] Smith, C. (2006). "On Vertex-Vertex Systems and their Use in Geometric and Biological Modelling". Doctoral Thesis. University of Calgary.
[Soucy96] Soucy M., Laurendeau D.(1996). "Multiresolution surface modeling based on hierarchical triangulation", Comput. Vis. Image Understand. 63 (1), 1996, pp. 1–14.
[StrintzisYEAR] M. Strintzis, N. Sarris (Ed.), (2005). "3D modeling and animation: Synthesis and analysis techniques for the human body", IRM Press, Hershey, PA, pp. 27-69.
[Taubin98-01] Taubin G., Gueziec A., Horn W., Lazarus F. (1998). "Progressive forest split compression". In Proceedings of the 25th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '98. ACM, New York, NY, 123-132.
[Taubin98-02] Taubin G, Rossignac J., (Apr. 1998). "Geometric compression through topological surgery". ACM Trans. Graph. 17, 2, 84-115.
[Touma98] Touma C., Gotsman C.(1998)." Triangle mesh compression". In Proceedings of Graphics Interface '98, pp. 26-34, 1998.
[Turan84] Turan G. (1984). "On the succinct representation of graphs, Discrete Applied Mathematics 8", 289-294.
[Van98] Van Sint Jan, S. L., Clapworthy, G. J., And Rooze, M. (1998). "Visualization of Combined Motions in Human Joints". IEEE Comput. Graph. Appl. 18, 6, 10-14.
[Van99] Van Sint Jan S.L., Salvia P., Clapworthy G.J., Rooze M. (1999). "Joint-Motion Visualisation Using Both Medical Imaging And 3D- Electrogoniometry", Proc 17th Congress Of International Society Of Biomechanics, Calgary (Canada).
[VHML] http://www.vhml.org/.
[Vilhjalmsson07] Vilhjalmsson, H. and Cantelmo, N., Cassell, J., Chafai, N. E., Kipp, M., Kopp, S., Mancini, M., Marsella, S., Marshall, A. N., Pelachaud, C., Ruttkay, Z.M., Thorisson, K., van Welbergen, H. and van derWerf, R.J. (2007). "The Behaviour Markup Language: Recent Developments andChallenges". In: Proceedings of the Seventh International Conference on Intelligent Virtual Agents,Paris, France.
[VRML] The Virtual Reality Modeling Language (VRML). ISO/IEC 14772-1, 1997.
[Wang02] Wang, X. C. And Phillips, C. (2002). "Multi-Weight Enveloping: Least-Squares Approximation Techniques for Skin Animation". In Proceedings of the 2002 ACM Siggraph/Eurographics Symposium on Computer Animation (San Antonio, Texas, July 21 - 22, 2002). SCA '02. ACM, New York, NY, 129-138
[Wang07] Wang, R. Y., Pulli, K., Popovic , J. (2007)."Real-timeenveloping with rotational regression". ACM Trans. Graph. 26, 3,73.
[Waters89] Waters, K. (1989). Modeling 3D Facial Expressions: Tutorial Notes. In State Of The Art In Facial Animation. ACM SIGGRAPH, 127–160.
[Watt91] Watt, A. And Watt, M. (1991). "Advanced Animation And Rendering Techniques". ACM.Armstrong, William W. And Green, Mark W. (1985). "The Dynamics of Articulated Rigid Bodies for Purposes of Animation". Proc. Graphics Lnterfafe 85, 407-415.
[Wilhelms85] Wilhelms, J. P. And Barsky, B. A. (1985). Using Dynamic Analysis to Animate Articulated Bodies Such as Humans and Robots. In Proceedings Of Graphics Interface '85 on Computer-Generated Images:The State of the Art (Montreal, Quebec, Canada).
[Wilhelms97] Wilhelms, J. And Van Gelder, A. (1997). "Anatomically Based Modeling". In Proceedings of The 24th Annual Conference On Computer Graphics And Interactive Techniques International Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., New York, NY, 173-180.
[Wooten98] Wooten, W. L. (1998). "Simulation Of Leaping, Tumbling, Landing, and Balancing Humans". Doctoral Thesis. UMI Order Number: AAI9827367, Georgia Institute Of Technology.
[Yang02] Yang J.-H., Kim C.-S., Lee S.-U. (2002). "Compression of 3D triangle mesh sequences based on vertex-wise motion vector prediction", IEEE Trans. Circuits Syst. Video Technol. 12 (12), pp. 1178–1184.
[Zhang04] Zhang J., Owen C.B. (2004). "Octree-based Animated Geometry Compression". In Proceedings of the Conference on Data Compression (March 23 - 25, 2004). DCC. IEEE Computer Society, Washington, DC, 508.
[Zhang05] Zhang J., Owen C.B. (2005) ."Hybrid Coding for Animated Polygonal Meshes: Combining Delta and Octree". In Proceedings of the international Conference on information Technology: Coding and Computing (Itcc'05) - Volume I - Volume 01 (April 04 - 06, 2005). ITCC. IEEE Computer Society, Washington, DC, 68-73.
[Zhu98] Zhu S.C, Wu Y., Mumford D. (1998). "Filters, random fields and maximum entropy (frame"). International Journal of Computer Vision, 27(2):1–20, March/April 1998.

Author's publications

Journals

- **Jovanova B**., Arsov I, Preda M, Preteux F., "On-line animation system for learning and practicing Cued Speech", To be published
- **Jovanova B**., Preda M., Prêteux F., (2009). "The Role of Interoperability in Virtual Worlds, Analysis of the Specific Cases of Avatars ", Technology, Economy, and Standards, Vol. 2(3), October 2009.
- **Jovanova B**., Preda M., Prêteux F., (2009). "MPEG-4 Part 25: A Graphics Compression Framework for XML Based Scene Graph Formats", Signal Processing: Image Communication, Vol. 24(1+2), January 2009, pp. 101-114.

International conference

- **Jovanova B**., Preda M., (2010). "Avatars interoperability in Virtual Worlds", In Proceeding of the MMSP'10 2010 IEEE International Workshop on Multimedia Signal Processing , Saint-Malo, France, October 4-6, 2010
- Arsov I., **Jovanova B**., Preda M., Prêteux F., (2009). "On-line animation system for learning and practice Cued Speech", In Proceeding of the international ICT Innovations conference, Ohrid, Macedonia, (ICT-ACT 2009), September 2009, pp. 315-325
- **Jovanova B**., Preda M., Prêteux F., (2009). "Auto-production 3D graphics content for mobile communication", In Proceeding of the international ICT Innovations conference, Ohrid, Macedonia, (ICT-ACT 2009), September 2009, pp. 305-313
- Arsov I., **Jovanova B**., Preda M., Prêteux F., (2010). "When MPEG-4 and COLLADA Meet for a Complete Solution of Distributing and Rendering 3D Graphics Assets", In proceedings of the International Conference on Consumer Electronics, Las Vegas, USA (ICCE 2010) , January 2010, pp. 431-432
- **Jovanova B**., Preda M., Prêteux F., (2008). "MPEG-4 Part 25: A generic model for 3D graphics compression", Proceedings 2nd 3DTV Conference (3DTV-CON 2008), Istanbul, Turkey, May 2008, p. 101-104.
- Preda M., **Jovanova B**., Arsov I., Prêteux F., (2007). "Optimized MPEG-4 animation encoder for motion capture data", Proceedings 12th International Conference on 3D Web Technology (Web3D'2007), Perugia, Italy, April 2007, p. 181-190.

Standardization reports

- **Jovanova B**., Preda M., Prêteux F., (2008). "Software Implementation for P25", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M17848, Geneva, Switzerland, July 2010.
- **Jovanova B.**, Preda M., Prêteux F., (2008). "Profiles for avatars interoperability", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M17849, Geneva, Switzerland, July 2010.
- **Jovanova B.**, Preda M., Prêteux F., (2008). "Animation update for Avatar Characteristics", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M17008, Xian, October 2009.
- **Jovanova B.**, Preda M., Prêteux F., (2009). "Avatar Characteristics", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2009/M16425, Maui, USA, April 2009
- **Jovanova B.**, Preda M., Prêteux F., (2009). "Selecting elementary streams in MP25 Reference Software", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2009/M16152, Lausanne, Switzerland, February 2009
- **Jovanova B.**, Preda M., Prêteux F., (2008). "Software Implementation for P25", Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M15823, Busan, January 2008.
- **Jovanova B.**, Preda M., Prêteux F., (2008). "New conformance for P25" ISO/IEC JTC1/SC29/WG11, MPEG2008/M15442, Archamps, April 2008.

- **Jovanova B.**, Preda M., Prêteux F., (2008). ”Software Implementation for P25”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M15085, Antalya, Turkey, January 2008.
- **Jovanova B.**, Preda M., Prêteux F., (2008). ”Conformance dataset for P25”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2008/M15086, Antalya, Turkey, January 2008.
- Concolato C., Le Feuvre J., **Jovanova B.**, Preda M., (2007). ”Storage of XML documents and associated media resources in the ISO file format”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2007/M14905, Shenzhen, China, October 2007.
- **Jovanova B.**, Preda M., Prêteux F., (2007). ”Compression Results for BBA on COLLADA Animation”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2007/M14903, Shenzhen, China, October 2007.
- Arsov I., **Jovanova B.**, Preda M., Prêteux F., (2007). ” MPEG-4 3D graphics tools for third party scene representation”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2007/M14711, Lausanne, Switzerland, July 2007.
- **Jovanova B.**, Preda M., Prêteux F., (2007). ”Compression performances of MPEG-4 3D Graphics for large databases”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG2007/M14710, Lausanne, Switzerland, July 2007.
- Preda M., **Jovanova B.**, Prêteux F., (2006). ”Motion Capture Compression with MPEG-4 BBA”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG06/13590, Klagenfurt, Austria, July 2006.
- Preda M., Laquet T., Raemdonckn W.V., Arsov I., **Jovanova B.**, Prêteux F., (2006). ”MPEG-4 3D Graphics on mobile phone”, Standardization Report ISO/IEC JTC1/SC29/WG11, MPEG06/13179, Montreux, Switzerland, April 2006.

Annex A. Animation Techniques

The animation techniques can be classified in three groups:

- traditional animation,
- stop motion or
- computer animation.

Let us present in detail each of these groups

Traditional animation

Traditional animation, also called cell animation or hand-drawn animation, was the process used for most animated films of the 20th century. The process of creating this kind of animation consists of drawing images on paper, and then taking photos of those drawings and finally creating the individual frames. To create the illusion of movement, each drawing differs slightly from the one before.

Examples of traditionally animated feature films include *Pinocchio* (United States, 1940), *Animal Farm* (United Kingdom, 1954), and *Akira* (Japan, 1988).

The traditional cell animation process became out-of-date at the beginning of the 21st century. Today, animators draw the images and backgrounds either on a computer or use scanners. For additional styling of the drawings, like their coloring, simulating camera movement and effects, various software is used. The "look" of traditional cell animation is still preserved, but the character animators' work has decreased significantly. Some animation producers have used the term "tradigital" to describe cell animation which makes extensive use of computer technology.

Traditional animated films which were produced with the aid of computer technology include *The Lion King* (US, 1994), *Sen to Chihiro no Kamikakushi (Spirited Away)* (Japan, 2001), *Treasure Planet* (USA, 2002) and *Les Triplettes de Belleville* (2003). Some of these examples are illustrated in Figure 86.

a) Pinocchio. b) Treasure Planet. c) Akira.

Figure 86. Examples of traditional animation.

Stop motion

Stop motion is a technique particularly used in character animation. It usually consists of physically manipulating the real-world objects and photographing them in different poses to create the illusion of movement. There are many different types of stop-motion animation and newer ones use computer software. Some of them are described in the following paragraph.

Puppet animation: the main object is a puppet that usually represents a character. Generally, it has armature inside to keep it still and steady as well as to constrain it to move at particular joints. Typically, puppet figures interact with each other in a constructed environment and this interaction is photographed. Some well-known movies using puppet animations are: *The Tale of the Fox* (France, 1937), *The Nightmare Before Christmas (*US, 1993*), Corpse Bride* (US, 2005), *Coraline* (US, 2009) and the TV series *Robot Chicken* (US, 2005–present) (Figure 87).

a) Coraline. b) The Nightmare Before Christmas. c) The man who planted trees.

Figure 87. Examples of puppet animation.

In clay animation (plasticine) the objects (that in the most cases are characters) are presented by figures made of clay or a similar malleable material. The figures may have an armature or a wire frame inside of them, similar to the related puppet animation, that can be manipulated in order to pose the figures or they can be made entirely of clay, where clay creatures morph into a variety of different shapes. Examples of clay-animated works include: *The Gumby Show* (US, 1957–1967), *Morph shorts* (UK, 1977–2000), *Wallace and Gromit Shorts* (UK, as of 1989), *Dimensions of Dialogue* (Czechoslovakia, 1982), *The Trap Door* (UK, 1984). Films include *Wallace and Gromit: Curse of the Were-Rabbit* and *The Adventures of Mark Twain.(*Figure 88*).*

a) The Clay Animation Version of UG, the Caveman. b) Wallace and Gromit. c) The Adventures of Mark Twain.

Figure 88. Examples of clay animation.

In silhouette animation the characters are only visible as silhouettes. Examples of this kind of animation are: *The Adventures of Prince Achmed* (Weimar Republic, 1926) and *Princes et princesses* (France, 2000).(Figure 89)

Figure 89. Examples of silhouette animation.

The **model animation** is done by interacting between stop-motion objects and live actors and/or environment. "Go motion" is a variant of "Model animation" which uses various techniques to create motion blur between frames of film, effect not present in traditional stop-motion. The only difference is the manner in which the images are filmed: while in the other "Stop motion" methods the images of the object are taken while it is in static position, in "Go motion" images are taken while the object is moving. To produce the movement of objects, different techniques are used, starting from simple ones like bumping the puppet and shaking the table to the most sophisticated technique which consists of rods connected to puppet or model, and then attached to motors which are linked to a computer that manipulates to reproduce pre-programmed movements. The technique was first time used for creating special effect scenes for the film *The Empire Strikes Back* (1980). Other examples are films such *Jason and the Argonauts* (1961), and King *Kong* (1933 film).

In **object animation** the characters are represented by regular inanimate objects, as opposed to specially created items. One example of object animation is the brickfilm, which incorporates the use of plastic toy construction blocks such as Lego. An example of modern object animation can be seen on *Robot Chicken* (TV series, 2005-present) (Figure 90).

a) Robot Chicken. b) Lego bricks.

Figure 90. Examples of object animation.

Graphic animation uses non-drawn flat visual graphic material like photographs, newspaper clippings, magazines, etc… One way to create a movement is to manipulate this material frame-by-frame. Another way is to leave the graphics stationary, while the stop-motion camera is moved to create on-screen action (Figure 91).

a) Frank Film (1973). b) The Wizard of Speed and Time (1979).

Figure 91. Examples of graphic animation.

In **pixilation** live humans are used as stop motion characters. Examples of pixilation include *The Secret Adventures of Tom Thumb* and the *Angry Kid* short series. (Figure 92).

a) El Hotel eléctrico(The Electric Hotel in Spanish), one of the earliest uses of pixilation, 1908.

b) Angry Kid.

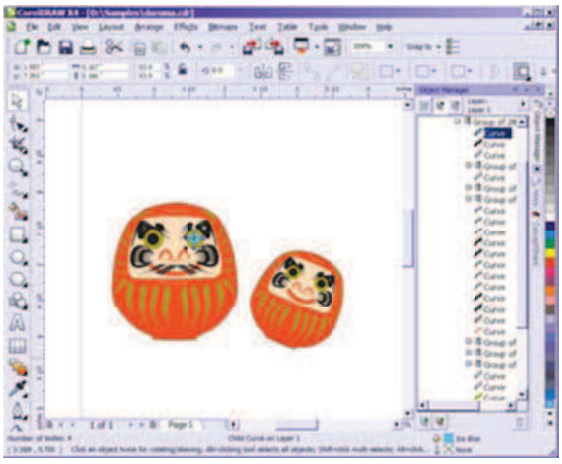
c) The Secret Adventures of Tom Thumb.

Figure 92. Examples of pixilation.

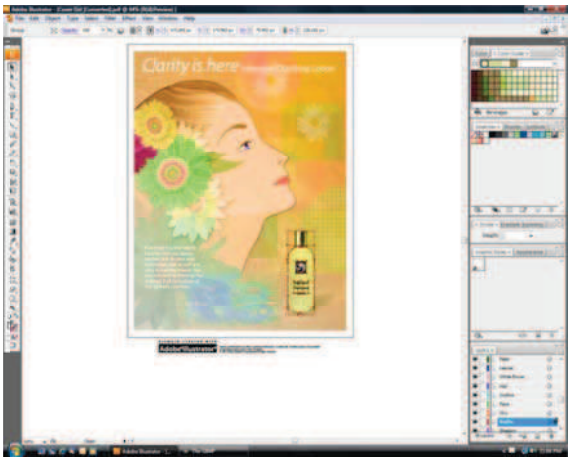
Computer animation

Computer animation includes a variety of techniques, with the unifying factor that the animation is created digitally on a computer. In early films, the production was concentrated on 2D animation. Frames were produced and/or edited on the computer using 2D bitmap graphics or 2D vector graphics. Largely used techniques are analog computer animation, Flash animation and PowerPoint animation. Different tools for creating 2D animation exist and some of them are represented in Figure 93. Nowadays, 2D animations are still widely used for stylistic, low bandwidth, and faster real-time rendering needs. Some well-known animations created in 2D are: *Foster's Home for Imaginary Friends*, *SpongeBob*, *Danny Phantom*, *The Fairly OddParents*, *El Tigre: The Adventures of Manny Rivera* (Figure 94).

More recently, 3D animations were developed, created by an animator while manipulating 3D digital models. The 3D models usually consist in a mesh connected with a digital skeletal structure. The process of linking the mesh with the skeleton is called rigging. Various other techniques can be applied on the mesh, such as mathematical functions (ex. gravity, particle simulations), simulated fur or hair or effects such as fire and water. Dedicated tools for 3D content and animation creation, such as: 3DS Max, Maya, Blender, Motion Builder (Figure 95) exist for this purpose.



a) Corel Draw Interface.



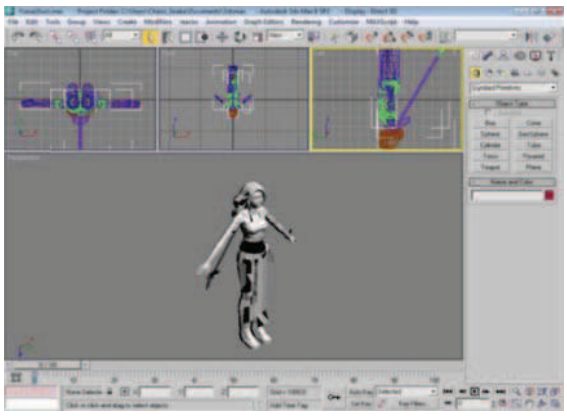
b) Adobe Illustrator Interface.

Figure 93. Examples of tools for creating 2D computer animation.

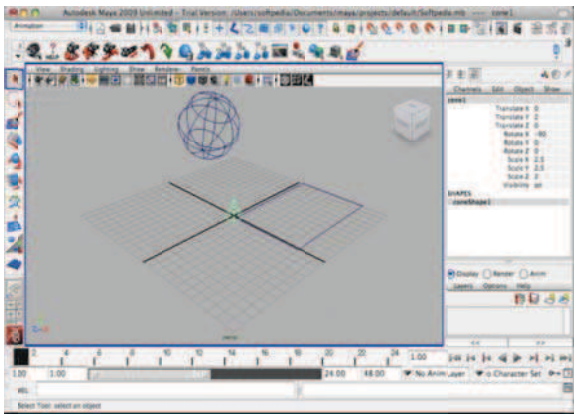
a) Pokemon.

b) Snow White, Disney.

Figure 94. Examples of 2D computer animations.



a) Autodesk 3DS Max.



b) Autodesk Maya.

Figure 95. Examples of tools for creating 3D computer animation.

A very common approach to produce 3D animation is the motion capture. In this technique, the animation is captured directly from real human movements by using special devices. The obtained animation is post-processed by the computer (Figure 96).

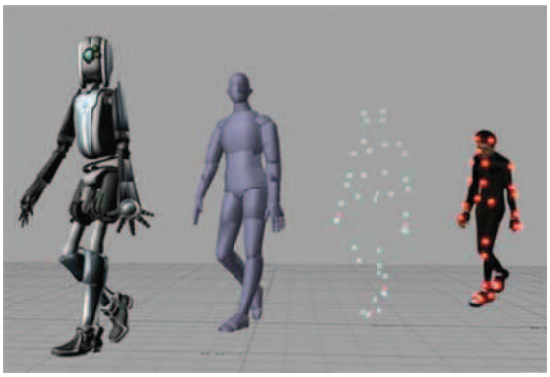


Figure 96. An example of Computer animation produced using Motion capture.

In general, the authoring tools support several of the animation principles mentioned above. As an example, Blender has an option with which the user is able to add frames in particular parts of the animation to achieve the "slow in and slow out" effect. This is especially useful in motion capture animation, because the animation is created with a constant frame rate.

a) Pocoyo.

b) Toy Story.

c) Shrek.

Figure 97. Examples using 3D computer animation.

Many 3D animations are very realistic and are commonly used as visual effects in movies. Some 3D animations as: *Pocoyo, Toy Story*, *Shrek* (Figure 97) were created by using dedicated authoring tools like the ones illustrated on Figure 95.

The future of computer animation

Having in mind the fast growth of the computer industry and the similar development of computer animation, predictions of the future of computer animation are very optimistic. Nowadays, 3D computer animation can be divided into two main directions: photorealistic and non-photorealistic rendering.

The non-photorealistic/cartoonish direction is more like an extension of traditional animation, an attempt to make the animation look like a 3D version of a cartoon. Currently, examples of computer-animated movies that show non-photorealistic/cartoonish animation are the ones representing animal characters, like in *A Bug's Life*, *Finding Nemo*, *Ratatouille*, *Newt*, *Ice Age*, *Over the Hedge*, then fantasy characters like in *Monsters Inc.*, *Shrek*, or anthropomorphic machines, for example in *Cars*, *WALL-E*, *Robots* or cartoon-like humans like in *The Incredibles,* Up, *Jimmy Neutron: Boy Genius, Meet the Robinsons*.

Photorealistic computer animation can itself be divided into two subcategories: real photorealism, and stylized photorealism. Example of stylized photorealism is *Antz*. In the future stylized photorealism should be able to replace traditional stop motion animation. Real photorealism is when performance capture is used in the creation of the virtual

human characters. An example of it is what *Final Fantasy* and *The Dark Crystal* tried to achieve. Real photorealism has a tendency to produce actions without the need to use advanced puppetry and animatronics. None of the mentioned techniques are perfected yet, but the progress continues. One open challenge in computer animation is a photorealistic animation of humans. The movie *Final Fantasy: The Spirits Within* is often cited as the first computer-generated movie to attempt to show realistic-looking humans. However, due to the enormous complexity of the human body, human motion, and human biomechanics, realistic simulation of humans remains largely an open problem. Eventually, the goal can be to create software with which the animator can generate a movie sequence showing a photorealistic human character, undergoing physically-plausible motion, together with clothes, photorealistic hair, a complicated natural background, and possibly interacting with other simulated human characters.

Annex B. Avatars Online

This Annex gives a brief summary of the web evolution, with the intention to foresee the place of avatars.

Web evolution

The traditional Internet websites were designed mainly to provide static information. Recently, by the introduction of Web 2.0, users are allowed to insert data. Users can own the data on a Web 2.0 site and exercise control over it. From passive users, they become actively involved in the creation of the website. A short comparison of the features of Web 1.0 and Web 2.0 is provided in Table 21. Some future development possibilities are presented in the last column of the table.

Feature	Web 1.0	Web 2.0	Possible Evolution on Web
Advertisement	DoubleClick	Google Ad Sense	Recommendations, opinions, negotiations in real time
Personal Photo/Media Sharing	Ofoto	Flicker	Augmented and Mixed Reality Application
Knowledge Sharing	Encyclopaedia Britannica Online	Wikipedia	Human-to-human real time help, VWs where the inhabitants communicate
Promoting, publishing	Page Views	Cost per click	Cost per assets that user poses
User on web	Passive observer, personal-web pages	Active participant, blogs	User presented by avatar, with all his capabilities, VWs

Table 21. Web 1.0 vs Web 2.0 vs "Future Web".

Related to the role of the avatars, let us comment some of the features presented above.

In Web 1.0, the advertising was fixed to a web page. Web 2.0 introduces AdSense: depending of the content presented on the page, the advertisement is dynamically added while surfing. The future web can be the web of VW economy: depending of what the user is looking for, agent sellers or real people represented by their avatar can exist in the VW and propose, provide information, offer advice and share experience about the product.

In Web 1.0, the users presented themselves to others by using personal web pages. The owners were the only ones that could edit the page. In Web 2.0 personal web pages were replaced by tools like blogging: other users can comment on the content, add new content, etc. One possible prospection for the equivalent function in the future web is that the user has his own place (home) in the VW, where he expresses his interests and skills by creating, designing this place, playing games, watching his favorite movies, listening to his favorite music, and creating and customizing his avatar. The user could represent himself to the others very closely to his real personality.

Global knowledge sharing and access may also dramatically increase. In Web 1.0 there were encyclopedias like Britannica Online, in Web 2.0 Wikipedia became famous. In the future web one possible idea is to have human-to-human real-time information: instead

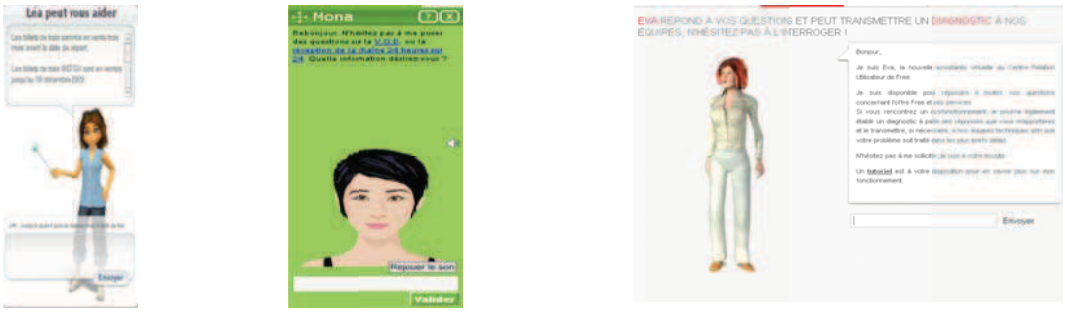
of searching, users (represented by their avatars) can access a specific place, meet other people which are online and ask or share opinions on some topic.

It is more and more evident that the new face of the Internet will migrate from a repository of information to a dynamic and lively place where people communicate with each other and jointly interact with the content, where time, presence and events become important.

All these novel scenarios evolve around two concepts: the need for real-time communication between users and the need for real-time interaction between users and the Web content. Some possible evolution is that the future of the Web will be 3D and Virtual Worlds where users may have their own representations. These representations can be the avatars, acting like a container of all the users' personal information: from appearance to emotion status and personal interests. The avatars also become the support for interaction. With the avatar, the user can directly participate: he can communicate with other avatars, speak and ask for help from 3D agents, buy from virtual sellers, learn from virtual avatars and perform other actions as much as possible similarly to real-life situations and with the big advantage of removing the limitations due to physical space. In the following sub-sections, usages of the avatars in real-time communications, Virtual Worlds and games are discussed.

Avatars as agents

Agents are "trained" avatars so that they autonomously navigate and react to changes in their environment, and also have the possibility to interact with users. One example where agents are used consists in services created to ease web surfing. Today's content on web-sites is rich and heterogeneous and sometimes it is difficult for the user to find the right information. Having an agent to help, communicating with the user or answering him to direct questions, may improve the user experience. Some examples of such web sites are SNCF France[30], France 5 [31] and free.fr [32] (Figure 98). The agents are also used in some games to help the players finish some "trivial tasks". For instance, in some virtual environment or 3D game, the user can ask some agent to go and pick several items and to meet him at some further point.



a) Léa – avatar for help on the SNCF web site.

b) Mona – avatar for search help on the France5 web site.

c) Eva – avatar for navigation help on the Free web site.

Figure 98. Avatars as agents on the web.

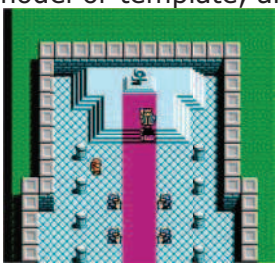
Web sites that are created for teaching and helping on a specific topic can also be guided by agents. There are examples of agents trained for teaching the Braille alphabet – an alphabet for blind people, Sign Languages (SL) for deaf people etc. Teaching SL in interactive applications is a big challenge and some solutions are currently available. The

[30] http://aide.voyages-sncf.com/
[31] http://www.france5.fr/mona/
[32] http://www.free.fr/assistance/eva.html

traditional manner of teaching is using highly illustrated books, sometimes accompanied by filmed material. With the development of the Internet and its transformation into a multimedia channel, several video web servers, especially the ones supplied by the community (Web2.0) include content for discovering or learning SL. For example, YouTube contains several Cued Speech (an alternative of SL) channels proposing more than 50 video lessons. Even more pedagogical is the on-line service developed by Michigan State University[33] which provides the video representation in American Sign Language for more than 4500 English words. By selecting a specific word, a video of a person signing the word is presented. Unfortunately, it is not possible to compose complete sentences or to input words. Producing the natural video data (for SL or CS) is costly and time consuming because it requires the presence of the signer and does not allow reuse and repurpose of the filmed content. Such limitations conducted to an increasing interest for synthetic images. One of the applications that allow to input a word or a sentence and to present the word in Sign Language has been developed by American Sign Language University (ASL University) and is available online at http://www.lifeprint.com/. More interactive scenarios including exercises for practicing CS are implemented in BALDI [Massaro84]. 'Baldi' is a 3D avatar face that can be animated. However, a limitation of this application is that the signed content is pre-recorded, being impossible to synthesize new words or sentences. A system for Cued Speech learning, able to synthesize in real time a 3D face and a 3D hand, based on the text or speech inputted by the user is implemented in the project LABIAO [Arsov09]. This can be used as an additional tool or can completely replace the video images in those applications.

Avatars in games

At the beginning of gaming industry, "avatar" was only a name for the player character inside the game (1985). The first game that assumed the "Avatar" as the player's visual on-screen in-game persona was introduced in Ultima 4 series (Figure 99 a)). The appearance of some of the first avatars is presented on Figure 99. Nowadays, avatars in games are essentially the player's physical representation in the game world. In most games, the player's representation is fixed. More recently games offer a basic character model or template, and the player can customize the physical features.



a) Ultima 4 series.

b) Super Mario Game (1985).

c) Mustapha Game (1992).

Figure 99. Avatars in the older games.

a) Tomb Raider (1996).

b) Prince of Persia 3D

c) Grand Theft Auto III

[33] http://www.easycartsecure.com/LanguageMatters

(1999). (2001).

Figure 100. Avatars in Third Person Shooter games.

According to the type, there are different classifications of games using avatars.

Third Person Shooter - In this group of games the virtual camera is normally located on an almost fixed point behind the avatar, but it can be rotated around the avatar. One of the first 3D games in which the avatar is visible in the camera viewpoint was *Tomb Raider* (1996) (Figure 20.a). Another popular example is the *Prince of Persia* series. The series started with 2D graphics in 1989, but as the 3D graphics capabilities improved, in 1999 the first 3D version, *Prince of Persia 3D* (Figure 100.b) was released. Since then, all of the editions have used 3D graphics. The *Grand Theft Auto (GTA)* (Figure 100.c) series was first released in 1997 and it featured a camera located high above the character. The later versions, starting from *GTA3* in 2001 featured a camera located behind the avatar. The game-play included options for changing the avatar's appearance (e.g. going into a bodybuilding club to increase the muscles), and later changing the clothes and some limited facial appearance.

Strategic life-simulation computer games – In this type of games the players can customize the characters, their behavior and surroundings in a very versatile way. They can change their appearance, skills, social behavior, can have children and influence their behavior and personality. The most popular games in this genre are *The Sims* series, started in 2000 (Figure 101 a)) and also *Spore*, released in 2008, (Figure 101 b)).

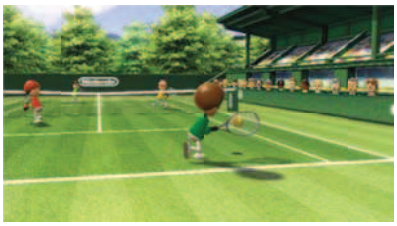
Massively multiplayer online games (MMOGs) – these games are source of the most varied and sophisticated avatars. Customization levels differ between games. For example in *EVE Online*, players construct a wholly customized portrait, using a software that allows several changes to facial structure, as well as preset hairstyles, skin tones, etc. Alternatively, *City of Heroes* offers one of the most detailed and comprehensive in-game avatar creation processes, allowing players to construct anything from traditional superheroes to aliens, medieval knights, monsters, robots and many more.

a) The Sims 3 (2009)

b) Spore (2008)

Figure 101. Strategic life-simulation computer games.

Avatars become an important component also for the game console games, the three main companies that manufacture gaming consoles including them in their platform. The users are mainly allowed to create and personalize their avatars as illustrated in the following section.



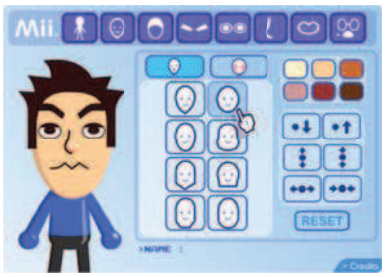



Figure 102. Examples from Nintendo avatars.

Nintendo's Wii console allows the creation of avatars called "Miis" (Figure 102) that take the form of stylized, cartoonish personages and can be used in some games as avatars for the players, as in the case of Wii Sports.


Figure 103. Examples from Xbox360 avatars.

On November 19, 2008 Microsoft released an Xbox 360 Dashboard update which featured the introduction of Avatars (Figure 103) as part of the console's New Xbox Experience. Users can personalize the look of their avatar by choosing from a range of clothing and facial features. On August 11, 2009 the NXE Avatar program was updated with the inclusion of an Avatar Marketplace feature that allows users to purchase additional products and game-branded clothing, jewelry, full body suits, and animated props. On initial release of the update, game-branded content included items from Gears of War 2, BioShock 2, Star Wars, Fable II, Halo 3, and The Secret of Monkey Island special edition. The Xbox LIVE Avatar Marketplace is updated weekly with new items.

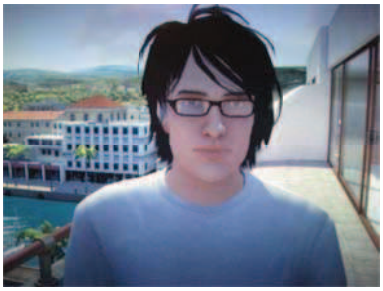



Figure 104. Examples from PlayStation Home avatars.

PlayStation Home for Sony's PlayStation 3 console also features the use of avatars (Figure 104), but here they are more realistically looking than Nintendo's Miis or Microsoft's Avatars. The user can create their own avatar or use one of several preset avatars available in Home. Users can access the Wardrobe from the Menu Pad at any time and location except when in another user's personal apartment. The user may customize a variety of the characters features including gender, skin tone, hair, body shape and facial structure. The users may also customize their avatar's clothing and

accessories using a set of standard items, items bought from one of the clothing shops in Home's shopping complex, or won items from Home's mini-games or PS3 games that support Home rewards.

It becomes obvious that from the first representations of the avatars till now, significant steps have been made. The games, which are a technology domain that always looks for innovation, have and most probably will have benefits from human-like representation.

While the first steps of avatars technologies development were made in the area of games, with technology improvements they became accessible on regular computers which open their usage in wide communities and in different applications, among which the Social Networks (SN) are most promising. The next section introduces the current trends of using avatars in SNs.

Avatars in Social VWs

Communication and content sharing are subjects that are always of interest for users. Different kinds of applications where users can chat, speak or share knowledge with each other exist.

Figure 105. Virtual Worlds that can or already include avatars

For example there are a lot of chatrooms, from not so popular, with no so many active users, to more popular like Skype, Windows Live Messenger, Google Talk, AOL Instant Messenger (AIM), ICQ, (Internet Relay Chat) IRC, Yahoo! Messenger, with millions of users. Some newer chats, IMVU, for example support 3D objects and the avatar representation of the user.

With the appearance of social networks, users can create their identity, but can also upgrade their status with activities, inform friends of what they are doing, where they are, what they are planning to do and other similar things. In these networks, users can add friends that they already know, colleagues, family members, but can also add new friends. Usually, these kinds of social networks support online chat. The most famous such applications are Facebook, Twitter, which is more oriented for IM and SMS, Faces.com, FaceParty, MySpace, Orkut. There are also social networks with special topics: Last.fm for music, Flickr for photos, Exploroo to share planning trips etc. In Table 22, different kinds of applications are presented and one possible way of grouping them based on their features is provided.

Type	Features	Applications – Examples
Text chat	way of communication by sending text messages to	Skype, Windows Live Messenger, Google Talk,

	one or more people in real-time via the Internet	AOL Instant Messenger(AIM), ICQ, (Internet Relay Chat)IRC, Yahoo! Messenger, IMVU
Voice chat	Way of communication that enables two or more people to talk to each other via the Internet	Skype, Windows Live Messenger, Google Talk, IMVU
Video communication	Communication by two or more people on Internet by sending real-time video	Skype, Windows Live Messenger, Google Talk, IMVU
3D Communication	Way of communication through the Internet where users are represented by a 3D Model and/or are placed in a Virtual World	IMVU
Personal information	Web pages where users share information about themselves	Personal WebPages, Blogger, Skyrock, Facebook, MySpace, Twitter, Orkut
Social Networking	online communities where users share interests and/or activities	Skyrock, Facebook, MySpace, Twitter, Orkut
Image Sharing	Services for creating image albums on Internet	PicasaWeb, Flickr
Video Sharing	Services where users can share video with others	YouTube, Vimeo
Presentation Sharing	Services for sharing presentations from conferences, meetings, etc.	SlideShare
Trip Planning	Network sites oriented for everything travel-related	Exploroo
Professional connection	business-oriented social networking sites, focused on interactions and relationships of a business nature of users	LinkedIn, Spoke
Online Games	networking sites where users can play games alone or with others	SecondLife, Entropia Universe
Knowledge Sharing		Wikipedia
Music Sharing	Services where users can share music with others	Last.fm, Youtube

Table 22 Different applications used in Web communication

Nowadays applications exist which have their own world, their own rules, topics, etc. Virtual world also plays a part in the social aspect as it can allow users who belong to

these kinds of communities to have their virtual representation, their second life. Because the interest lies in VWs where avatars are used like a human representation, in the next part a brief representation of some existing social networks is presented.

Figure 106. HiPiPi

In HiPiPi[34], social network users have the opportunity to create their avatar representation and their 'place', which they will inhabit. Thus they became 'residents' of the HiPiPi and can 'live' there. This network is a rich and complex 3D digital world, almost like the real world, with limitless possibilities for creativity and self-expression, within a complex social structure and with a fully functional economy.

Figure 107. Sony PlayStation Home

"Sony Playstation Home" allows users to create a custom avatar, which can be made to suit the user's liking. Through their avatars, users can travel throughout the "Home world". Each part of the world is known as a space. "Home Space" is the avatar's personal apartment. In "Sony Playstation Home" a lot of items represented by 3D objects exist. Users can acquire by default some "HomeSpace" items. "Public spaces" can just be for display, fun, or for meeting other users. There the users can connect with friends and customize content. Also "Sony Playstation Home" has included many mini-games which can be single-player or multiplayer, has many places for advertising, virtual theatres, and can be host to a variety of special events.

[34] http://www.hipihi.com/en/

Figure 108. Sony Playstation Home

Active Worlds'[35] original goal was to be the 3D-equivalent of a 2D web browser (such as Internet Explorer or Firefox). Instead of creating a website, the user could construct an office, building, or area in which to display products or information. It is a 3D virtual reality platform represented by "Active Worlds Browser" that runs on Windows.

Figure 109. Google Lively

Google Lively was a web-based virtual environment that ran only on IE and FF, produced by Google Inc. It worked with the embedding of Lively "rooms" into any HTML webpage — which meant content could be provided in a two-dimensional format, and the communication surrounding the topic of that content could be made in the three-dimensional room without the need to enter a separate program. Up to 20 people could occupy a room and chat with one another. Users were represented by their avatars, could design their own virtual environments, hanging on the walls videos from YouTube and photos from Picasa. It was discontinued on November 19, 2008.

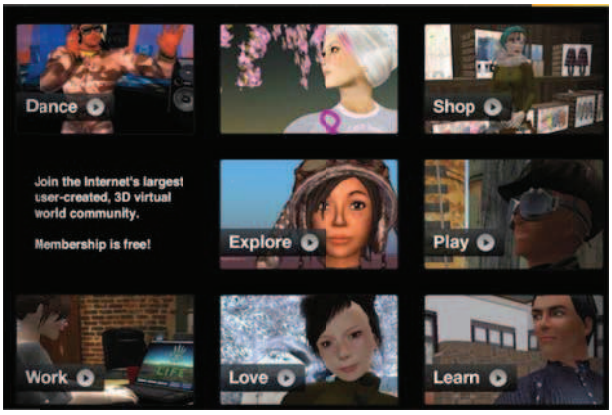


Figure 110. Second Life

[35] http://www.activeworlds.com/

Second Life[36] can be considered as the most famous social network at the moment. Second Life can be used by people aged 18 and over, while Teen Second Life is for people aged 13 to 17. There is no charge to create a Second Life account or for making use of the world for any period of time.

Users can create their avatar in the Second Life and with that they become so-called "Residents" of the VW and can interact with each other.

Built into the software is a three-dimensional modeling tool based around simple geometric shapes allowing a resident to build virtual objects. More complex three-dimensional objects, textures for clothing or other objects, and animations and gestures can be created using external software and imported in SL. The Second Life Terms of Service ensure that users retain copyright for any content they create, and the server and client provide simple digital rights management functions.

Residents can explore, meet other residents, socialize, participate in individual and group activities, and create and trade virtual property and services with one another, or travel throughout the world, which residents refer to as the grid.

In Second Life virtual places that copy reality already exist, residents can make their own business, make real money, communicate with the real world etc.

Avatars can communicate via local chat or global instant messaging (known as IM). Chatting is used for localized public conversations between two or more avatars, and is visible to any avatar within a given distance. IMs are used for private conversations, either between two avatars, or among the members of a group, or even between objects and avatars.

Figure 111. There

In There[37], each new member enters the community by choosing a unique name and a male or female avatar. The avatar's name and gender are permanently set, but various attributes such as hair color and style, head and body shapes, skin and eye color, clothing, etc. can be changed as desired. Unrestricted by set goals or required activities, members can avail themselves of There's many features, including: sophisticated chat via text or voice with other nearby members or remotely via instant messaging. Avatars display body language, verbal articulation, and gestures that are triggered by various textual keywords, emotes, and even vocal inflection. Also in There avatars can play There-specific sports like: hockey, sumo buggy bashing, soccer or racing spades. They can also dance and compose different animation combinations through the use of a keyboard macro program such as Giggles, listen to music and talk shows on special There are 'SHOUTcast stations', which often are operated by other members, they can participate in story-based events such as questing, movie-making, and creative writing member-created groups for interests like virtual pets, currently limited to dogs. Users

[36] http://secondlife.com/
[37] http://www.there.com/

can explore a village in There by foot or by the vehicles that exist : Bacio (a two-seated scooter that resembles a Vespa), buggy, hoverbike, hoverboard, hoverboat (aircraft), and hoverpack. Users can create living quarters, meeting places, game rooms, movie sets, race tracks, mazes, yard sales - in short, whatever the member can imagine using available materials. Monetary transactions in There's economy are done using Therebucks, a virtual currency with real world value. Therebucks can be purchased directly from There, from other members, or from any one of the 3rd party online "banks" which usually offer competitive exchange rates. Members can also sell their Therebucks to banks in exchange for real world currency.

Social Networks with 3D objects and especially users represented in them with their own identity, of course, will bring benefits like a more realistic view or feel of closeness. Being still new and challenging because of its complexity, this is a promising topic, especially if one takes into account the fast development of computer hardware, wider network bandwidth and the progress in computer graphic.

Avatars in economy and business communication

Avatars have an impact not only in virtual worlds, but their democratization makes them suitable for several applications with impact to real economy or communication.

Typical examples of virtual economy are virtual shops where presence is obtained by 3D simulation. Users can access virtual rooms, speak and buy from the sellers, represented by avatars, as illustrated in the example of *Bottin Carto* (Figure 112).

Figure 112. Avatars in economy.

Second Life, one of the most famous virtual worlds, makes also the bridge between the virtual and the real, being possible to buy/sell objects from/to 'Non person character'[38] (NPC) or 'Person character' (PC) sellers. The former include virtual avatars created by companies and working in a SL as sellers or help desk; behind them there is no real person, only an agent. The latter are avatars linked with real people which sell/buy and have their own economy in SL.

Technology improvement enables people to stay in touch more efficiently. For example, a lot of work can be conducted through teleconferences. Furthermore, there are different online communication systems like chat, long-distance presentations, voice and video conferences.

[38] http://www.bottincarto.com/pro/showroom.asp/

a) Summit in PlayStation Home VW

b) Presentation in PlayStation Home VW

Figure 113. PlayStation Home VW.

With the presence of avatars, distance communication, such as conferences, meetings and plan decisions, can be done with the same efficiency in the 3D VWs as in the real world. This concept changes significantly the lives of business people and companies. Currently, it is possible to have this type of communication in Second Life and Sony PlayStation Home (Figure 113).

Mobile VW

Cellufun is a mobile virtual community where users can create their personal avatars and through them meet other people, make friends, chat, shop, and play social games. The first mobile virtual currency called "FunCoins" was introduced, which enables users to buy avatar clothes and special in-game advantages.

Itsmy is a mobile social networking service which combines mobile communication and personal content sharing. It allows the users to create a personal homepage including a profile. It is possible to add graphics and styles where the user can select a profile picture, background graphics, avatar, homepage color and buttons. It offers mobile networking and messaging communication services including video and picture blogging.

Sparkle is a 3D Virtual World for IPhone/iPod Touch. The users can connect on Sparkle with their SL and OpenSim accounts, and sends and receives IMs, chat, send teleport requests etc. on the go to/from their SL/OpenSim friends. Sparkle is a rich mobile VW where users can be registered independently from SL or OpenSim, as in a separate VW. Previously represented only by 3D clouds, now the users can be represented by real characters.

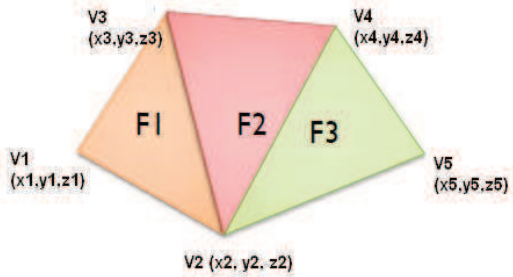
Annex C. Representations for 3D meshes

- **List of independent faces**

A list of independent faces is a representation where a mesh is composed of faces that are represented by vertex coordinates, as illustrated in Figure 114. **Error! Reference source not found.**

	Vertex (X, Y,Z)	**Total**
F1	3	9 floats
F2	3	9 floats
F3	3	9 floats
TOTAL	27 floats	27 floats

a) Table representing data storage.



b) Mesh example.

Figure 114. List of independent faces example.

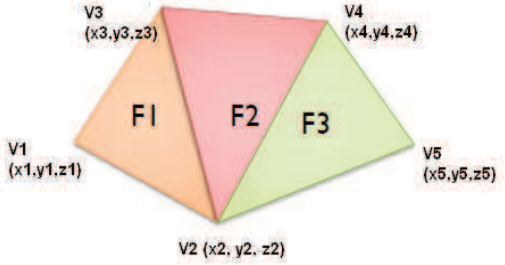
The memory requirement is linearly proportional with the number and type of the faces (triangles, quads, etc). Because vertices are stored, all records have the vertex type (usually vertex position represented by 3 floats), hence in the example above 3 * 3 * 3 = 27 floats are required to store them. The need for redundant vertices is one of the biggest disadvantages of this representation.

- **Face-Vertex (FV) Meshes**

The face-vertex mesh (Figure 115) is a representation where all different vertices are stored in an ordered list, and polygons are formed by indexing it.

	Vertices	**Total**
F1	3	3
F2	3	3
F3	3	3
TOTAL	9	9

a) Table representing data storage.



b) Mesh example.

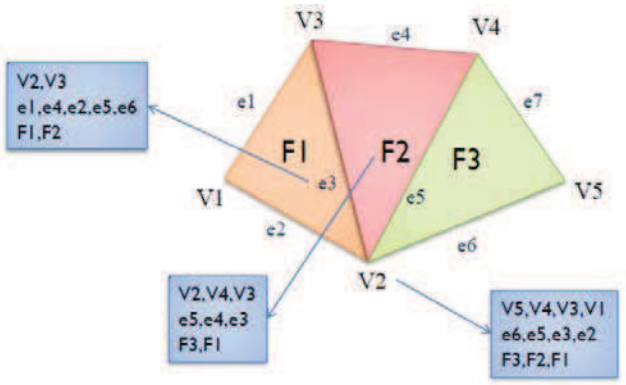
Figure 115. Face-Vertex Meshes example.

In the example above 5 * 3 = 15 floats for representing the vertices position and 3* 3= 9 integers are needed for the indices.

- **Adjacency Lists**

This representation stores all vertices, edges and faces adjacencies. It allows efficient adjacency traversal, however it requires extra storage.

	Vertices	Edges	Faces	Total
V1	2	2	1	5
V2	4	4	3	11
V3	3	3	2	8
V4	3	3	2	8
V5	2	2	1	5
E1	2	3	1	6
E2	2	4	1	7
E3	2	5	2	9
E4	2	4	1	7
E5	2	5	2	9
E6	2	4	1	7
F1	3	3	1	7
F2	3	3	2	8
F3	3	3	1	7
TOTAL				104



a) Table representing data storage. b) Mesh example.

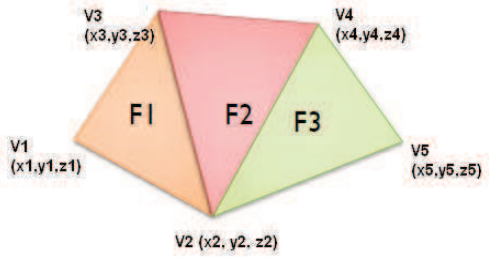
Figure 116. Adjacency Lists example.

As shown in Figure 116 a), even presenting the simple mesh shown in Figure 116 b) requires 104 integers and 15 floats (5 vertices * 3 floats for each vertex).

- **Winged-Edge (WE) Meshes**

In the winged-edge representation, each edge points to two vertices, two faces, and the four (clockwise and counter-clockwise) edges that touch it. The advantage of this data structure is, as in the previous method, the possibility to go from any element of the representation to another. Still the storage requirement remains high.

	Edges	Faces	TOTAL
V1	2	1	3
V2	4	3	7
V3	3	2	5
V4	3	2	5
V5	2	1	5
F1	3		3
F2	3		3
F3	3		3
TOTAL			34



a) Table representing data storage. b) Mesh example.

Figure 117. Winged-Edge Meshes example.

In the example above memory required is computed as follows: edges needed 2 * 6 + 4 * 6 = 36 integers and the rest of elements 34 integers. Therefore, a total of 70 integers

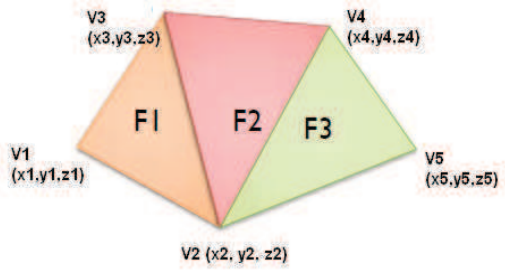
and 15 floats (5*3 for vertices position storage) are needed, a relatively high memory requirement.

- **Corner-Table**

A corner-table representation stores vertices in a predefined table, such that traversing the table implicitly defines polygons. This is in essence the "triangle fan" used in hardware graphics rendering. The representation is more compact, and more efficient to retrieve polygons; however, operations to change polygons are slow. Furthermore, Corner-Tables may not be able to represent complete meshes at once, hence multiple corner-tables (triangle fans) are needed.

Table
V1
V3
V2
V4
V5

a) Table representing data storage.



b) Mesh example.

Figure 118. Corner-Table example.

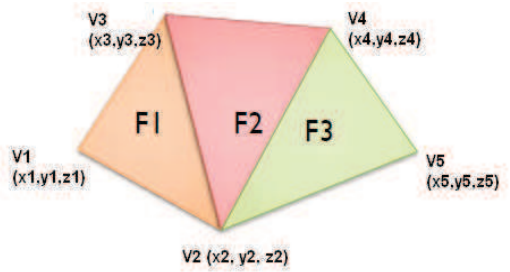
For the same mesh as above the memory requirements are only for the vertices' table, that means 15 floats (Figure 118); however, this is a special case where the mesh is already in triangle-fan.

- **Vertex-Vertex (VV) Meshes**

A VV mesh represents vertices which point to other (neighboring) vertices ordered in circular lists. Both the edge and face information are implicit. Therefore, it is necessary to traverse the data in order to generate a list of faces for rendering. In addition, operations on edges and faces are not easily accomplished. However, the method allows edges or faces to be dynamically added. However, this increases the complexity and the number of pointers needed to represent the structure. In this simplest form, memory requirements are not high, though 14 integers and 15 floats are needed for our instanced mesh, as illustrated in Figure 119.

	Vertices	Total
V1	2	2
V2	4	4
V3	3	3
V4	3	3
V5	2	2
TOTAL	14	14

a) Table representing data storage.



b) Mesh example.

Figure 119. Vertex-Vertex Meshes example.

Representation	Formula	Indices N°	Rossingnac's notation	Possible Traversals
Independent Faces	$3\sum_{i=1}^{F_c} F_i^{Val}(V)$	27 floats	$\{R, F: R \to F\}$	region to face
Face-Vertex	$\sum_{i=1}^{F_c} F_i^{Val}(V)$	9 int + 15 floats	$\{R, F, V: R \to F \Rightarrow V\}$	region to face face to vertex vertex a set of 3
Adjacency Lists	$\sum_{i=1}^{F_c}[F_i^{Val}(F) + F_i^{Val}(E) + F_i^{Val}(V)] + \sum_{i=1}^{E_c}[E_i^{Val}(E) + E_i^{Val}(F) + E_i^{Val}(V)] + \sum_{i=1}^{V_c}[V_i^{Val}(V) + V_i^{Val}(F) + V_i^{Val}(E)]$	104 int + 15 floats	$\begin{Bmatrix} R, F, E, V: \\ F \to F, F \to E, F \to V, \\ E \to E, E \to F, E \to V, \\ V \to V, V \to F, V \to E \end{Bmatrix}$	face to edge edge to face edge to vertex vertex to edge face to vertex vertex to face vertex a set of 3
Windget-Edge	$\sum_{i=0}^{F_c} F_i^{Val}(E) + 2E_c + 4E_c + \sum_{i=1}^{V_c} V_i^{Val}(F, E)$	70 int + 15 floats	$\begin{Bmatrix} F, E, V: \\ F \to E \overset{2}{\Rightarrow} V, \\ E \overset{4}{\Rightarrow} E, \\ V \to (F, E) \end{Bmatrix}$	face to edge edge to face edge to vertex vertex to edge vertex a set of 3
Corner-Table	$5V_c + V_c$	30 int + 15 floats	$\{F, V: V \overset{5}{\to} V, V \overset{1}{\to} F\}$	vertex to vertex vertex to face vertex a set of 3

Vertex-Vertex	$\sum_{i=i}^{V_c} V_i^{Val}(V)$	14 int + 15 floats	$\{V: V \Rightarrow V\}$	vertex to vertex vertex a set of 3

Table 23. Different methods for polygon mesh representation and the corresponding features.

The complexity of each data structure can be compared to other structures using the notation proposed by Rossignac [Rossignac94]. In this notation, a regular arrow indicates a set of pointers, a double arrow indicates an ordered set of pointers and a number over the arrow specifies an amount of pointers. R stands for region, F for faces, E for edges, V for vertices. For example, a face structure that contains a pointer to an edge is notated as F-> E or a face that points to an ordered list of four vertices uses the notation F 4=>V. The memory requirements to store each representation are provided by the formulas in Table 23 , where R, F, V, E are defined according to the Rossignac notation, while "*Val*" represents the valence between corresponding elements. Table 23 also presents the summary of methods described above by defining a general formula for computing storage requirements for each representation, illustrating the number of integers (pointers) and floats (values) required to represent a simple mesh, present Rossingnac's notation and a possible traversal through the topology of mesh.

Annex D. XML scheme for MPEG-V, Part 4

Avatar element definition

```
<xsd:complexType name="AvatarType"abstract="true"> <xsd:sequence>
    <xsd:element
name="Appereance"id="Appearance"type="AppearanceType"minOccurs="0"maxOccurs=
"unbounded"/>
    <xsd:element
name="Animation"id="Animation"type="AnimationType"minOccurs="0"maxOccurs="unb
ounded"/>
    <xsd:element
name="CommunicationSkills"type="CommunicationType"minOccurs="0"maxOccurs="un
bounded"/>
    <xsd:element
name="Personality"minOccurs="0"type="PersonalityType"maxOccurs="unbounded"/>
    <xsd:element
name="Control"type="ControlType"minOccurs="0"maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="ID"use="required"type="xsd:integer"/>
  <xsd:attribute name="Family"use="optional"type="xsd:integer"/>
  <xsd:attribute name="Name"use="optional"type="xsd:string"/>
  <xsd:attribute name="Gender"use="optional"type="xsd:string"/>
 </xsd:complexType>
 <xsd:element name="Avatar"type="AvatarType"/>
```

Definition of the main elements of Avatar Appearance

```
<xsd:complexType name="AppearanceType">
  <xsd:sequence>
    <xsd:element name="Body"id="Body"type="BodyType"/>
    <xsd:element name="Head"id="Head"type="HeadType"/>
    <xsd:element name="Eyes"id="Eyes"type="EyesType"/>
    <xsd:element name="Ears"id="Ears"type="EarsType"/>
    <xsd:element name="Nose"id="Nose"type="NoseType"/>
    <xsd:element name="MouthLip"id="Mouth"type="MouthType"/>
    <xsd:element name="Skin"id="Skin"type="SkinType"/>
    <xsd:element name="Facial"id="Facial"type="FacialType"/>
    <xsd:element name="Nail"id="Nail"type="NailType"/>
    <xsd:element name="BodyLook"id="BodyLook"type="BodyLookType"/>
    <xsd:element name="Hair"id="Hair"type="HairType"/>
    <xsd:element name="EyeBrows"id="EyeBrows"type="EyeBrowsType"/>
    <xsd:element name="FacialHair"id="FacialHair"type="FacialHairType"/>
```

```
    <xsd:element name="AppearanceResources"minOccurs="0" type="
AppearanceResourceType" id=" AppearanceResources"/>


-------------------------------------------------------------------------------------------------
-----------------------------------------
Definition of the main elements of Avatar Animation
-------------------------------------------------------------------------------------------------
-----------------------------------------


<xsd:complexType name="AnimationType">
   <xsd:sequence>
     <xsd:element name="Idle"id="Idle"type="IdleType"/>
     <xsd:element name="Greeting"id="Greeting"type="GreetingType"/>
     <xsd:element name="Dance"id="Dance"type="DanceType"/>
     <xsd:element name="Walk"id="Walk"type="WalkType"/>
     <xsd:element name="Moves"id="Moves"type="MovesType"/>
     <xsd:element name="Fighting"id="Fighting"type="FightingType"/>
     <xsd:element name="Hearing"id="Hearing"type="HearingType"/>
     <xsd:element name="Smoke"id="Smoke"type="SmokeType"/>
     <xsd:element name="Congratulations"id="Congratulations"type="CongratulationsType"/>
     <xsd:element name="CommonActions"id="CommonActions"type="CommonActionsType"/>
     <xsd:element name="SpecificActions"id="SpecificActions"type="SrecificActionType"/>
     <xsd:element name="FacialExpression"id="FacialExpression"type="FacialExpressionType"/>
     <xsd:element name="BodyExpression"id="BodyExpression"type="BodyExpressionType"/>
     <xsd:element
name="AnimationResources"id="AnimationResources"type="AnimationResourceType"/>
    </xsd:sequence>
 </xsd:complexType>


-------------------------------------------------------------------------------------------------
-----------------------------------------
Definition of the main elements that control the avatar: body feature controllers (bones)
and face feature controllers
-------------------------------------------------------------------------------------------------
-----------------------------------------


<xsd:complexType                              name=                              "ControlType">
   <xsd:sequence>
     <xsd:element                                                  name="BodyFeaturesControl"type="
BodyFeaturesControlType"minOccurs="0"/>
     <xsd:element  name="FaceFeaturesControl"type="FaceFeaturesControlType"minOccurs="0"/>
   </xsd:sequence>
   <xsd:attribute                                                 name="Name"type="CDATA"/>
 </xsd:complexType>


…..
```

Annex E. An example of an avatar graph

```
SBVCAnimation {
 url [49]
 virtualCharacters [
  SBSkinnedModel {
   bones [
    DEF BONE_000_Root SBBone {
     children [
      DEF BONE_001_First SBBone {….
       children [
        DEF BONE_002_Second….
         children [
          DEF BONE_003_Third….
         ]
       ]
    USE BONE_001_First….
    USE BONE_002_Second….
    USE BONE_003_Third….
   ]
   skeleton [
    USE BONE_000_Root
   ]
   Skin [
    Shape {
     appearance Appearance {
      material Material {
       ambientIntensity 1
       diffuseColor 1 1 1
       shininess 0.1
      }
      texture ImageTexture {
       url [300]
      }
     }
     geometry IndexedFaceSet {
      coordIndex []
      normalIndex []
      solid FALSE
      texCoordIndex []
      coord Coordinate {
       point []
      }
      normal Normal {
       vector []
      }
      texCoord TextureCoordinate {
       point []
      }
     }
    }
    Shape {…}
   ]
  }
 ]
}
```