Composition de flux vidéo avec acquisition répartie multi-capteurs

Phuong H. Nguyen[1], Gilles Privat[1], Jozef Hatala[2], Pascal Sicard[3]

[1] France Telecom R&D
28 Chemin du Vieux Chêne BP 98
38243 Meylan Cedex, France

[2] France Telecom R&D
801 Gateway Boulevard,
South San Francisco, USA
{phuonghoang.nguyen, gilles.privat, jozef.hatala} @rd.francetelecom.com

[3] Laboratoire LSR-IMAG
Université Joseph Fourier
681 Rue de la Passerelle BP 72
38402 Saint Martin d'Heres Cedex, France
pascal.sicard@imag.fr

Résumé : *Nous présentons une approche de composition de flux vidéo temps-réel acquis à partir de caméras multiples réparties, basée sur une description de l'environnement utilisant d'un langage de représentation d'environnements graphiques 3D. Nous définissons l'architecture de l'application sur la base d'un framework multimédia distribué.*

Mots clés : *communication pervasive/ ubiquiste ambiante, aboutement d'images, composition vidéo, X3D, framework multimédia*

1 Introduction

La banalisation des capteurs vidéo numériques et la possibilité de les connecter par réseaux sans fils permettent d'envisager des applications de communication nouvelle tirant parti de cette répartition généralisée dans l'espace de leur interface d'acquisition.
On peut ainsi envisager d'offrir, entre deux lieux distants, un service de « communication ambiante » qui corresponde à l'évolution à long terme des services interpersonnels de communication visuelle, en y incorporant les idées d'exploitation de la cognition d'arrière-plan, d'interaction implicite et d'adaptativité au contexte proposées autour du mot-clé d' « intelligence ambiante » [1][2]. Par défaut, ce service servirait simplement de support à un simple sentiment de présence partagée par une vision panoramique symétrique, mais pourrait aussi s'adapter à la présence et à l'activité des personnes de part et d'autre, pour les suivre et leur permettre de se consacrer en priorité à leurs activités normales, la communication restant en arrière-plan pour interférer le moins possible avec ces activités.
Pour atteindre ce but, on cherche à gérer la composition spatiale d'un ensemble de flux vidéo issus de capteurs répartis. La transmission de ces flux est supportée par les outils classiques de codage et streaming vidéo, tandis que leur composition utilise les outils issus du domaine du graphique 3D, où l'information vidéo va être intégrée à un niveau plus élevé de représentation spatiale.

2 Composition panoramique par aboutement

Dans la première implémentation prototype que nous avons réalisée pour supporter le mode panoramique de la communication ambiante, la composition des flux vidéo provenant de trois à cinq caméras numériques USB permet une vision de 120 à 200 degrés « sans couture » de l'espace distant.

Figure 1. Architecture de Labvision.

Il s'agit d'un aboutement adaptatif d'images réalisé sur des flux temps-réel, de manière similaire à ce qui est fait pour la composition de panoramas en photographie numérique par les logiciels dits de « stitching ». La calibration des positions des caméras permettant d'optimiser cet aboutement est assistée par un utilitaire logiciel associé. L'utilisateur distant peut choisir d'extraire un morceau du champ panoramique résultant de l'aboutement en commandant à distance le système d'acquisition de la salle par des fonctions de déplacement latéral. La vidéo correspondante paraît ainsi issue d'une

« caméra virtuelle » qui émule de manière purement logicielle le fonctionnement d'une caméra pivotante montée sur tourelle mécanique motorisée. Il faut noter que le coût d'ensemble des caméras numériques banalisées permettant de composer ce panoramique est inférieur d'au moins un ordre de grandeur à celui d'une caméra traditionnelle sur tourelle, ou à celui d'une optique d'adaptation spécifique (lentille ou miroir) permettant d'acquérir directement le panoramique par anamorphose sur un capteur unique.
L'utilisateur peut également profiter d'une fonction de "zoom temporel" - le retour (plus de 30 heures) dans le temps pouvant se faire à différentes échelles (par tranche d'une minute, dix minutes …). Ce système a été mis en œuvre pour offrir un sentiment de communauté de travail dans un laboratoire réparti entre les sites de France Telecom R&D à Rennes, Lannion, Grenoble et San Francisco.

Pour aller vers une application vraiment flexible et adaptable aux contextes, Nous constatons trois points essentiels à résoudre pour élaborer des applications de communication multimédia totalement flexibles.

- *Hétérogénéité :* La brique de base initiale limitait le nombre de caméras et le type de capteurs à des modèles spécifiques. Pour aller au-delà, il s'agit de gérer l'*hétérogénéité* de la nature des sources (i.e. des capteurs fixes/mobiles, zoom optique ou non…) et de la nature des contenus (codecs, contrôles de flux…)
- *Répartition :* L'application imposait une configuration prédéfinie manuellement, sans interaction flexible avec le contexte ou avec d'autres applications (par exemple pour le suivi de personnes). Dans la perspective de capteurs de diverses natures il faut prévoir dans l'architecture de notre système des mécanismes d'accès, d'identification et de localisation géographique des sources.
- *Adaptation :* L'absence des contraintes de qualités de services dans des nouvelles infrastructures est un des points essentiels: le goulot d'étranglement ne se situe plus au niveau des réseaux de communication mais aux terminaux. Malgré toutes les améliorations de QoS des réseaux de cœur, l'application doit pouvoir s'adapter aux qualités de services variables du réseau de bordure et aux capacités diverses des postes client, et cela de façon transparente à l'utilisateur. Cette adaptation doit s'effectuer pour permettre une cohabitation multi–plateformes logiciel et multi-réseaux.

3 Composition multi-sources

3.1 Approche conceptuelle

Nous proposons une solution pour relier et contrôler ce type de système réparti des capteurs vidéo d'une manière dynamique, établissant une scène visuelle complète de leurs points de vue combinés, où la navigation dans la scène est déclenchée par la position de l'utilisateur comme détectée par un système de localisation. Ce système de localisation peut être un système de vision supporté par les mêmes capteurs vidéo que ceux utilisés pour l'acquisition, par des capteurs complémentaires dédiés le cas échéant ou n'importe quel autre système offrant la précision requise pour la localisation intra-bâtiment. Ce système de localisation sera utilisé par notre système au travers d'une infrastructure de gestion de localisation qui en abstraira les données remontantes [3].
Pour modéliser l'environnement, notre approche diffère de celle de systèmes comme RB2 [4] ou Massive [5] qui décrivent l'environnement comme une collection d'objets répartis. RB2 focalise à la communication point to point entre deux utilisateurs. Massive supporte un environnement virtuel distribué multi-utilisateurs mais la communication est aussi point to point basée sur réseau TCP/IP. Notre conception de l'environnement est basée sur un *graphe de scène* qui connecte tous les éléments pertinents de l'environnement. Dans ce graphe, l'environnement est décrit de manière hiérarchique à partir d'un nœud père – *root node*. L'environnement se compose à partir des groupes de nœuds. Cette approche peut être aussi trouvée dans des systèmes comme OpenGL Performer [6], OpenInventor [7] ou DIVE [8] mais ces systèmes l'ont utilisée pour décrire des environnements virtuels en se focalisant sur le *rendu prédéfini* tandis que nous nous focalisons sur la gestion des sources pour *l'acquisition dynamique*. Ces derniers systèmes manquent aussi d'un mécanisme d'optimisation de performance efficace à cause de leur conception rigide de graphe de scène.

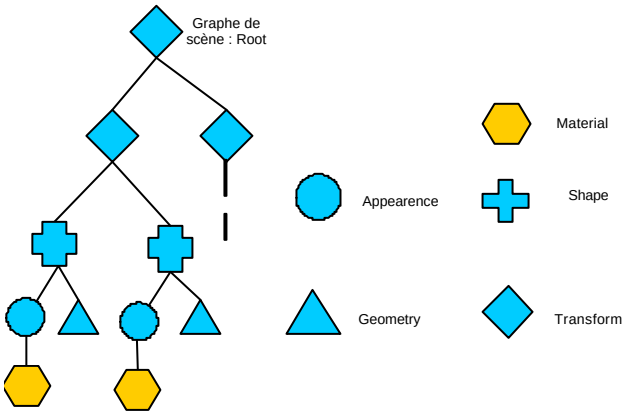

Figure 2. Approche graphe de scène.

Notre approche conceptuelle de la modélisation de l'environnement est inspirée du modèle X3D (Extensible 3D Graphics) [9] issu Web 3D Consortium. X3D est allé au-delà d'une amélioration de VRML (Virtual Reality Modelling Language) [10] en XML. Il corrige des bugs dans VRML fournit une nouvelle architecture basée composant adaptable et extensible qui se base sur un noyau léger et un mécanisme de « profile » (groupe) pour adapter aux besoins des utilisateurs. Contrairement à Performer ou Inventor qui limitent à l'utilisation de C/C++, avec X3D nous pouvons travailler avec différents langages comme ECMA Script, Java, C++ et utiliser la plupart des navigateurs Web comme environnement de fonctionnement (i.e. nous n'avons besoins que d'un plug-in X3D pour interpréter la description de scènes).
MPEG-4 BIFS (Binary Format for Scenes) est conçu sous forme binaire et associe à MPEG plateforme, X3D peut être utilisé sous différentes manières : XML, VRML « classique » et binaire. La version XML de BIFS - XMT (Extensible MPEG-4 Textual Format) est vraiment une concurrence de X3D dans notre besoin. C'est un modèle complet mais il garde toujours la gestion hiérarchique stricte de BIFS : il ne permet pas avoir plusieurs «top node », cette possibilité dans X3D peut nous faciliter l'intégration et l'émergence de différentes descriptions complètes des environnements. Un point de plus dans X3D est son approche basé-composant qui est plus flexible que XTM malgré l'interopérabilité très puissante de XMT (avec X3D, VRML, BIFS). Comme le profile d'interaction de X3D est supporté par MPEG4, nous possédons une forte interopérabilité entre ces deux systèmes. L'association entre SMIL (W3 consortium Synchronized Multimedia Integration Language) et BHTML (Broadcast HTML) est une nouvelle approche mais plutôt focalisée à la synchronisation que la composition des sources d'acquisition.
Un avantage complémentaire résultant de l'utilisation des navigateurs est aussi la généralisation des natures possibles du contenu des sources de flux vidéo (QuickTime®, Windows Media®, RealOne®) via les plug-ins vidéo appropriés.
Dans notre modèle de graphe de scènes, l'environnement réel n'est pas décrit par des scripts, mais les méta-données de description de la composition de scène sont décrites en utilisant un graphe de scènes des nœuds descriptifs. Les nœuds primitifs comme *Geometry, Shape, Transform, Appearence, Material* sont conformes à la spécification X3D.
Même si nous pouvons spécifier une extension de X3D mais nous avons essayé d'en éviter en utilisant les fonctions existantes. Parmi les primitives nous estimons qu'il faut modifier, ultérieurement, les descriptions de position avec *Transform* car ce dernier est très limité pour une composition complète.
Dans notre framework, nous considérons les sources à composer sont présentés sous forme des « virtual device » en utilisant des «wrappers » logiciels. L'approche d'un environnement basé-objets de X3D nous permet de valider notre architecture logicielle basé-composants. Des composants représentant les capteurs sont des nœuds X3D modélisés par les nœuds *Material* dans le graphe.

L'approche conceptuelle de description de scène permet de visualiser l'espace par la composition des vues de chaque dispositif d'acquisition vidéo. X3D, qui est au départ un langage pour la composition graphique en objets 3D est ainsi employée pour modeler un ensemble de capteurs distribués dans l'espace.
En même temps nous pouvons appliquer une notion de méta-composants de description de sources qui sont gérés par des bibliothèques dynamiques. Dans ces bibliothèques, chaque entité est le modèle PROTO de X3D d'une classe des composants-capteurs sources de même nature. Nous utilisons la déclaration EXTERNPROTODECLARE pour permettre une réutilisation des entités PROTOS distribuées. A partir de ce modèle de description nous pouvons former une description d'un nœud dans le graphe de scènes qui s'adapte au client final. Un capteur doit fournir ses attributs (fixes/mobiles, zoom optique ou non ; champ de vue, vitesse du mouvement, zone d'observation et ses coordonnées, son voisinage…) qui servent au module de gestion de composition pour construire une scène X3D complète qui peut être interprétée au niveau du client.

3.2 Architecture

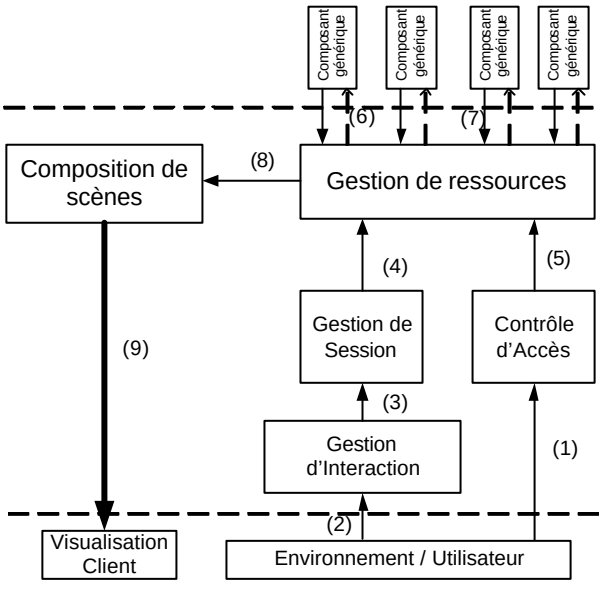

Figure 3. Architecture du système dynamique.

Grâce à la généricité des composants, nous pouvons définir la même architecture pour n'importe endroit de fonctionnement (i.e. point intermédiaire) de notre application. Les fonctionnalités des modules principauxde

notre application de partage d'ambiance sont représentées dans la Figure 3:

- *Gestion d'Interaction* : Ce module permet à l'utilisateur d'interagir avec l'application s'il le désire. Il nous semble important de garder cette interaction possible dans un environnement ubiquiste. Il envoie des commandes sur module *Gestion de Sessions* (3), par exemple zoom à l'aide d'une caméra particulière.
- *Contrôle d'Accès* : Quand l'utilisateur s'identifie de façon transparente ou volontaire(1), le système va lui attribuer des droits d'utilisation en regardant un profil prédéfini. Un exemple d'utilisation de ce module est le mode *secret* dans l'application de partage d'ambiance.
- *Gestion de Session* : Il permet d'avoir un ensemble de sessions avec différents clients. Chaque client a sa contrainte d'utilisation spécifique : caractéristiques du terminal client, qualités de services du réseau. Il fournit au module Gestion de Ressources (4) des contraintes d'exécution (différent codage, filtre de résolution …). Par exemple, dans une application telle que LabVision, la qualité des flux vidéo est gérée automatiquement. Les problèmes de streaming multi-point sont aussi traités dans ce module.
- *Gestion de Ressources* : Il interagit (7) avec des entités logicielles génériques représentant des dispositifs physiques (i.e. wrapper) ou des logiciels spécifiques. A partir des méta-données fournies par les composants génériques (6), des contraintes de session (3) et des restrictions d'accès (5), il transmet les méta-données des flux vidéo associés (8) au module de *Composition*.
- *Composition de scènes* : Il construit la scène 3D à visualiser côté client en fonction des flux multimédia et des métadonnées associées. La scène est décrite en langage X3D, les nœuds du graphe de scène X3D sont les flux vidéo transmis par la module de Gestion de Ressources. La scène peut être jouer côté client (9) à l'aide d'un plug-in X3D.

4 Exemple d'application

Grâce à la flexibilité de l'architecture, nous pouvons avoir des réalisations rapides des applications attentives aux contextes en utilisant des services basiques prédéfinis.

Dans la scénario ci-dessus, l'utilisateur est placé dans le centre de l'environnement qui est modéliser par l'association des vues rectangles. La position de l'utilisateur est détecté en temps réel en utilisant les mêmes sources d'acquisition. L'utilisateur de l'autre côté ne voit que la partie actuelle de l'environnement. L'observation est effectuée de manière explicite en associant facilement des dispositifs entrés ou par mode automatique en utilisant des fonctions basant sur le champ de vision. L'hétérogénéité des dispositifs d'acquisition est masquée au niveau application car la nature des flux de données sont transparente à l'utilisateur(i.e. il suffit que le navigateur client possède un «plugin » pour différents types de contenu).

5 Conclusion

La contribution nouvelle présentée ici est liée à l'utilisation de X3D pour composer des flux vidéo complexes, distribués et indépendants. Nous avons utilisé un langage dédié à la présentation d'un environnement local pour la représentation des méta-données des capteurs distribués. Le service démontré correspond à la composition de flux vidéo, mais l'architecture proposée pourra également être employée pour composer des sources audio réparties. Le but est bien de reconstituer les éléments d'un environnement réel distant à l'instar d'un environnement virtuel, mais l'approche diffère fondamentalement des techniques de téléprésence immersive en gérant cette reconstitution de manière adaptative au contexte et en la restituant par le biais d'interfaces non-intrusives, permettant à la communication de s'effacer dans l'environnement des utilisateurs

Références

[1] Gilles Privat, "Des Objets Communicants à la Communication Ambiante" *Les Cahiers du Numérique*, volume 3 n° 4-2002.

[2] Emile Aarts, « Ambient Intelligence : A multimedia Perspective », *IEEE Multimedia*, January-March 2004

[3] T. Flury, G. Privat, "An Infrastructure Template for scalable location-based services", *Smart Objects Conference, SOC'2003*, Grenoble, Mai 2003

[4] Blanchard, C., Burgess, S., Harvill, Y., Lanier, J., Lasko,A., Oberman, M., and Teitel, M. Reality built for two: A virtual reality tool. In Computer Graphics (1990 Symposium on Interactive3D Graphics) (Mar. 1990), R. Riesenfeld and C. Sequin, Eds., vol. 24, pp. 35–36.

[5] Greenhalgh, C., and Benford, S. MASSIVE: a collaborative virtual environment for teleconferencing. ACM Transactions on Computer-Human Interaction 2, 3 (Sept. 1995), 239–261.

[6] Rohlf, J., and Helman, J. IRIS Performer: A High Performance Multiprocessing Toolkit for Real–Time 3D Graphics, Computer Graphics Proceedings, ACM SIGGRAPH, ACM Press, Jyly1994 pp. 381–395.

[7] Wernecke, J. The Inventor Mentor. Addison-Wesley, 1994.

[8] Carlsson, C., and Hagsand, O. DIVE — A Platform for Multiuser Virtual Environments. Computers and Graphics 17, 6 (Nov.–Dec. 1993), 663–669.

[9] ISO/IEC JTC1/SC24 and the Web3D Consortium, Extensible 3D Graphics, ISO/IEC FCD 19775:200x, 19776:200x, 19776:200x. http://www.web3d.org/

[10] VRML97 Functional specification and VRML97 External Authoring Interface (EAI) ISO/IEC 14772-1:1997 and ISO/IEC 14772-2:2002