Parallélisation sur GPU des modèles d'exécution d'un environnement situé 3D et d'agent virtuels. Application à la simulation de foule.

Jocelyn Buisson[12] Stéphane Galland[1] Mikaël Gonçalves[2] Abderrafiaa Koukam[1]
[1]Laboratoire Systèmes et Transports
Université de Technologie de Belfort-Montbéliard, 90010 Belfort, France
[2]Voxelia SAS, Bâtiment 328 site Techn'hom IV, 26 rue Marcel Pangon, 90300 Cravanche, France
Email: jocelyn.buisson@voxelia.com
Web: multiagent.fr — voxelia.com

***Résumé*—La simulation des systèmes complexes constitue actuellement un enjeu majeur dans de nombreuses disciplines et notamment dans celle de l'aménagement et l'étude d'infrastructures urbaines. La simulation multi-agents est un outil incontournable pour la modélisation de systèmes complexes. L'objectif de ces travaux de thèse et de permettre la simulation à large échelle d'environnements urbains peuplés de piétons et de véhicules tout en simulant les comportements de chaque individu. L'approche proposée est basée sur l'utilisation d'algorithmes parallèles sur processeur graphique.**

***Index Terms*—Simulation multi-agents, Simulation à large-échelle, Réalité virtuelle, Processeur graphique (GPU)**

I. Une thèse CIFRE effectuée au sein d'une jeune entreprise universitaire

La simulation des systèmes complexes constitue actuellement un enjeu majeur dans de nombreuses disciplines telles que les neuro-sciences, la biologie, l'écologie ou encore l'économie. La simulation est une approche appropriée pour étudier des systèmes qui ne peuvent pas être directement observés ou mesurés. L'objectif de la simulation est de faciliter la compréhension de la dynamique d'un système et tenter d'en prédire l'évolution. Satisfaire cet objectif nécessite l'élaboration d'un modèle du système à étudier, son exécution sur un calculateur, et l'analyse des résultats de cette exécution.

Les enjeux majeurs liés à l'activité de gestion d'infrastructures s'inscrivent dans la prévision et la prévention des problèmes qui émergent pendant les phases de conception, d'utilisation et d'évolution des infrastructures. La simulation est une approche appropriée pour étudier des systèmes qui ne peuvent pas être directement observés ou mesurés pour des raisons de sécurité, d'impossibilité matérielle ou de coûts financiers. La simulation peut être intégrée au sein d'un outil d'aide à la décision afin de tester des scénarios d'aménagement ou d'utilisation d'une infrastructure parfois directement au sein d'une maquette virtuelle pouvant être explorée en temps réel.

En effet, les techniques multi-agents permettent de faire évoluer au sein d'une même simulation des entités hétérogènes dotées de toute une palette de comportements aisément modifiables et extensibles. De plus les systèmes d'information géographique (SIG) permettent de disposer de maquettes virtuelles géo-référencées qui s'approchent au plus près de la réalité.

Cette thèse est effectuée au sein de l'entreprise Voxelia, une entreprise innovante spécialisée dans les services et l'édition de composants logiciels autour des maquettes virtuelles en trois dimensions. Elle vient d'obtenir, et c'est une première en Franche Comté, son statut de Jeune Entreprise Universitaire.

II. Simulation de système urbains

Les systèmes urbains sont des exemples caractéristiques de systèmes complexes. Structurellement, un système urbain peut être décomposé hiérarchiquement sous forme de rues, bâtiments, quartiers, arrondissements, etc. Économiquement et socialement, il peut être perçu comme une structure composée d'individus, de familles, d'entreprises, et de commerces. La complexité des environnements urbains peut être abordée selon de nombreux points de vue. Toutefois, deux principaux points de vue peuvent être adoptés. Le premier concerne l'évolution de la structure urbaine (formation des villes, morphologie, expansion, etc.) et le second traite davantage des activités sociales, et notamment des modèles de trafic et de déplacement de foules (figure 1). La simulation multi-agents est un outil particulièrement bien adapté à la simulation des dynamiques urbaines. Les systèmes multi-agents s'avèrent plus flexibles que les modèles macroscopiques à base d'équations différentielles pour simuler des phénomènes spatiaux et évolutifs.

III. Vers un portage des structures arborescentes sur le GPU

La simulation multi-agents consiste à simuler le comportement individuel de chaque entité du groupe faisant l'objet d'une étude plutôt que de chercher à simuler le comportement du groupe en tant que tel. Chaque entité simulée, ou agent, doit alors être capable de prendre des décisions basées sur ce qu'il perçoit de l'environnement dans lequel il est situé. Il est par conséquent, très important de pouvoir calculer de façon aussi précise et efficace que possible ce que peut percevoir un

FIGURE 1. Simulation d'une station de Métro tirée de l'outil SIMULATE ©Voxelia

agent dans son environnement en utilisant ses sens, souvent limités à la vue. Ce point est crucial car il occupe la majorité du temps de calcul d'un pas de simulation et limite le nombre d'entité pouvant être actives simultanément.

Compte tenu de la structure complexe d'un environnement virtuel réaliste, il est essentiel d'utiliser des structures de données performantes et ce sont généralement les structures de données arborescentes de partitionnement de l'espace qui sont préférées de part leur capacité à éviter rapidement des calculs en éliminant les branches de l'arbre qui sont inutiles. Cette technique a été développée et mise en œuvre avec succès dans la plateforme JaSIM [9], [14].

Malgré l'utilisation de structures de données de plus en plus sophistiquées et performantes, les contraintes matérielles actuelles limitent le nombre d'agents dans une simulation qui ne dépasse que très rarement les 10000 entités et ce nombre décroit de façon dramatique lorsque la complexité du raisonnement dont est capable chaque agent augmente. L'utilisation du processeur graphique pour accélérer les calculs en parallélisant les traitements constitue une piste de recherche intéressante. Ce processeur est déjà, à l'heure actuelle, très étudié pour développer des structures de données permettant, entre autres, d'accélérer la synthèse d'images par lancé de rayons ou la détection de collision. Les processeurs graphique ont cependant des contraintes très importantes au niveau de leur programmation ce qui empêche une adaptation directe des algorithmes existants et conçus pour le processeur central. De plus ces contraintes rendent très difficile la création de structures de données génériques, c'est à dire utilisables quel que soit le domaine d'application.

IV. UNE APPROCHE BASÉE SUR L'IMPLÉMENTATION

Un processeur graphique ne connait qu'une seule structure élémentaire : le tableau. La taille de ces tableaux ne peut être changée une fois qu'ils sont créés c'est pourquoi il est très difficile d'adapter les structures de données arborescentes prévues pour le processeur central car elles sont basées sur des listes de tailles variables. Les arbres de partitionnement sont les structures de données arborescentes les plus populaires et les plus performantes pour stocker des objets spatiaux. Leur intérêt se situe dans le fait que lorsque l'arbre est équilibré il est possible d'éliminer très vite les branches qui répondent négativement à une requête et donc éviter de nombreux calculs sur les objets présents dans ces branches. Parmis toutes les structures de partitionnement de l'espace, cetaines ont déjà été adaptée pour le processeur graphique :

- Les KD-Trees/Quadtree/Octree (K étant le nombre de sous-divisions pour chaque nœud de l'arbre) sont des arbres de subdivision d'un espace de dimension quelconque qu'ils découpent récursivement en deux, quatre ou huit zones en utilisant des hyperplans alignés sur des axes (une droite alignée sur un axe en 2D, un plan parallèle au plan formé par deux des trois axes en 3D, etc.). Ces arbres ont été parmis les premières structures de données arborescentes à être adaptées pour le processeur graphique et sont toujours très utilisées dans le domaine de la synthèse d'images par lancé de rayons [3], [8], [11], [21], [22], [26], [27] mais également pour la détection de collision [2]. La figure 2 illustre le partitionnement et la répartition d'objets, représentés par les points, dans un arbre de type 2D-Tree.
- Les BVHs ou hiérarchies de volumes englobants sont des arbres qui subdivisent un espace de dimension quelconque en plusieurs groupes de volumes contenant des objets [25]. Les BVHs sont aussi largement utilisées pour des applications de rendu par lancé de rayons [4], [7], [10], [12], [18], [23] et de détection de collision [15], [16].

Des systèmes multi-agent ont également été portés sur le processeur graphique. Parmi ces nombreux portages, la plupart sont basés sur des environnements discrets dans lesquels le monde est représenté par une grille de cellules de taille fixe et les agents se déplacent de case en case [17], [1]. D'autres approches utilisent un découpage en grille régulière tout en faisant évoluer les agents dans un domaine continue [6], [13], [5], [19], [20]. Enfin des travaux se basent sur la communication inter-agent pour éliminer le besoin de calculer ce que perçoit chacun d'entre eux [24]. Ces travaux ne concernent cependant que des agents ayant un comportement très simple et ainsi permettant d'éviter les collisions avec les obstacles et de rejoindre une zone prédéfinie.

V. CONCEPTION, DÉVELOPPEMENT ET COMPARAISON DE STRUCTURES ADAPTÉES

Les travaux entrepris dans cette thèse visent à développer un modèle environnemental basé sur une structure de données arborescente à la fois rapide à mettre à jour lorsque les agents se déplacent et rapide à parcourir pour calculer les perceptions le plus efficacement possible. Ce travail nécessite une étude et donc l'implémentation et le test de toutes les structures de données spécifiques au processeur graphique existantes à ce jour, à leur adaptation pour traiter les données propres à la simulation plutôt qu'aux données propres au rendu ou la

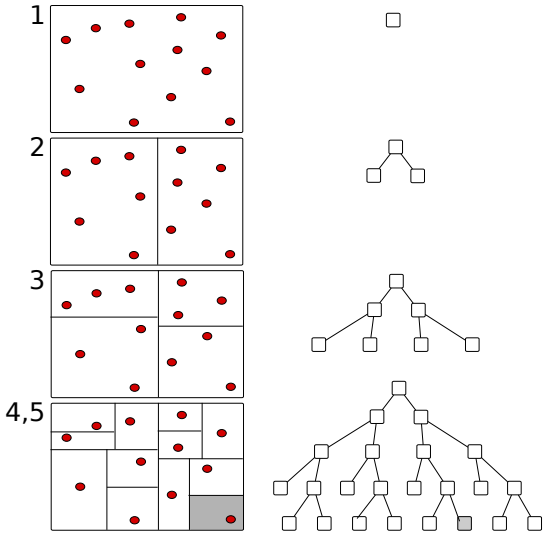

FIGURE 2. Exemple de partitionnement d'un plan avec un arbre de type 2D-Tree

détection de collision, ainsi que l'élaboration d'algorithmes de construction mais aussi de parcours. Il est également envisagé de développer une ou plusieurs nouvelles structures de données répondant d'avantage aux besoins très particulier du calcul des perceptions nécessitant la détection des intersections entre les volumes des cônes de vision des agents et tout objet statique ou dynamique, tels que les autres agents, présents dans l'environnement.

Reprenons l'exemple illustré par la figure 2. Initialement l'arbre n'est constitué que d'un seul nœud couvrant la totalité de l'espace à partitionner. L'étape 2 sélectionne un axe de découpe permettant de répartir équitablement la population dans deux sous-groupes qui sont alors associés à des sous-nœuds. Cette procédure est appliquée récursivement pour chaque nœud dans les étapes 3 à 5. L'arbre ainsi constitué permet de converger rapidement vers les objets se trouvant dans le champs de perception de chaque agent simuler. Supposons que nous voulions savoir si un objet se trouvant dans une zone grisée S. Sans arbre, jusqu'à 12 tests d'intersections sont nécessaires dans notre exemple. Avec un arbre, le nombre de test est de 4 : un pour chaque nœud se trouvant entre la racine (exclue) de l'arbre et le nœud correspondant à S L'implantation de ces structures arborescentes est un standard de la programmation information. La plateforme JaSIM [9], notre référence en terme d'implantation sur processeur central, propose en ensemble complet d'arbres et d'algorithmes de parcours.

L'objectif de cette thèse est de fournir une implantation des structures arborescentes sur un GPU en n'utilisant que des structures matricielles ou des tableaux. La transposition pourrait être considérée comme triviale, mais les contraintes imposées par l'architecture GPU complexifient considérablement cette démarche. En effet, les GPUs possèdent une mémoire limité en taille et pouvant être peu performante en cas de mauvaise utilisation de ses différentes zones mémoire dédiées. De plus l'allocation mémoire ne peut pas être dynamique. Ceci impose des choix d'algorithmes ne traitant qu'avec des tableaux de taille fixe. Enfin, les GPUs ne sont vraiment performants que s'il est possible de dégager un traitement parallèle à exécuter sur les différents cœurs du processeur graphique. L'étape de conception et d'implantation d'un premier algorithme sur GPU dédié au calcul des objets perçus par chaque agent est terminée. Les performances algorithmiques et calculatoires de cet algorithme seront confrontées à celles des algorithmes issus de la littérature ; ainsi qu'à celles des algorithmes proposés par JaSIM, afin de mesurer le gain apporté par les GPU par rapport à une approche entièrement CPU.

VI. SIMULATION DE FLUX URBAIN EN RÉALITÉ VIRTUELLE

Cette thèse vise aussi à élaborer une architecture complète d'un simulateur urbain permettant de modéliser les flux routiers et piétons en milieu urbain, en utilisant les modèles comportementaux existants, tout en permettant une visualisation des plus réaliste par l'intermédiaire d'un moteur de rendu 3D temps réel. Ces travaux visent à faire émerger des comportements et extraire des données liées à l'interaction entre des entités hétérogènes qui sont rarement associées au sein d'une même simulation.

La figure 3 illustre l'architecture générale de ce simulateur. L'environnement est un singleton contenant l'ensemble des objets présents dans le monde simulé (routes, bâtiments, signalétiques, etc.) ainsi que tous les corps associés aux agents (piétons et véhicules). Chaque agent perçoit grâce à un ensemble de capteurs dans l'environnement. Puis il décide d'une action à réaliser en fonction de ses connaissances, de ses objectifs et de ses perceptions. Enfin chaque agent met en œuvre les actioneurs correspondants à l'action précédemment décidée. L'environnement possède deux grandes fonctions : (i) calcul de la liste des objets se trouvant dans les champs de perception de chaque agent ; (ii) récolte les actions provenant des agents et modification de l'état de l'environnement en respectant les loi du monde simulé. Ces deux grandes fonctions utilisent les modèles GPU précédemment cités, ainsi que les modèles CPU de JaSIM [9] et non couverts par les fonctionnalités des modèles GPU.

VII. CONCLUSION ET PERSPECTIVES

Cet article présente le contexte et les objectifs de nos travaux sur la parallélisation sur GPU d'un environnement situé 3D pour la simulation d'agents virtuels. Cette dernière possèdent des enjeux majeurs liés aux activités de gestion d'infrastructures urbaines et de prévention des risques. L'approche multi-agent adoptée permet de simuler les comportements hétérogènes se trouvant dans la population des usagers de ces systèmes urbains. Ainsi, la qualité et la précision des résultats obtenus intégreront les comportements individuels et leurs influences qui sont en général gommées par les approches statistiques de simulation plus traditionnelles. Le cœur des travaux présentés ici se situe dans l'amélioration de la montée en charge de simulations multi-agents en proposant des

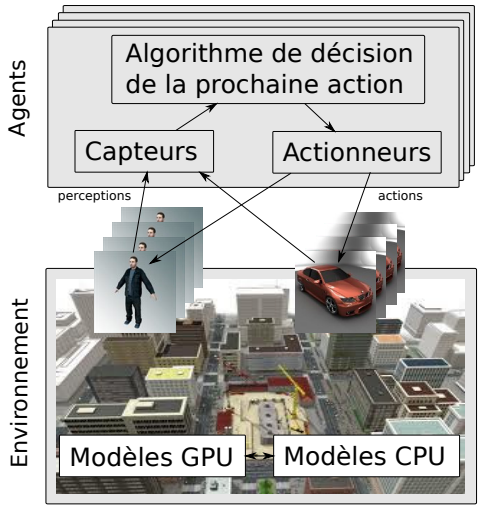

FIGURE 3. Architecture générale du simulateur urbain

algorithmes parallèles sur GPU. A court terme, nos premiers résultats algorithmiques seront comparés aux approches existantes. A plus long terme, l'utilisation du GPU sera généralisée à l'ensemble des modules composant notre simulateur, et non plus uniquement au calcul des objets perçus par les agents.

RÉFÉRENCES

[1] B. G. Aaby, K. S. Perumalla, and S. K. Seal, "Efficient simulation of agent-based models on multi-gpu and multi-core clusters," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '10. ICST, Brussels, Belgium, Belgium : ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010, pp. 29 :1–29 :10. [Online]. Available : http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8822

[2] J. A. Bird, "Gpu accelerated collision queries," Master's thesis, University of Abertay Dundee, School of Computing and Creative Technologies, May 2008.

[3] B. Choi, R. Komuravelli, V. Lu, H. Sung, and R. L. Bocchino, "Parallel sah kd-tree construction for fast dynamic scene ray tracing," University of Illinois, Urbana-Champaign, Tech. Rep., 2009.

[4] D. Cline, K. Steele, and P. Egbert, "Lightweight bounding volumes for ray tracing," *Journal of Graphics Tools : JGT*, 2006.

[5] A. da Silva, M. Gerais, W. Lages, and L. Chaimowicz, "Improving boids algorithm in gpu using estimated self occlusion," 2008.

[6] U. Erra, R. De Chiara, V. Scarano, and M. Tatafiore, "Massive simulation using gpu of a distributed behavioral model of a flock with obstacle avoidance," in *Proceedings of Vision, Modeling and Visualization 2004 (VMV)*, November 2004. [Online]. Available : http://wonderland.dia.unisa.it/projects/gebs/

[7] B. Fabianowski and J. Dingliana, "Compact bvh storage for ray tracing and photon mapping," *Eurographics Ireland 2009*, pp. 1–8, 2009.

[8] T. Foley and J. Sugerman, "Kd-tree acceleration structures for a gpu raytracer," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, ser. HWWS '05. New York, NY, USA : ACM, 2005, pp. 15–22. [Online]. Available : http://doi.acm.org/10.1145/1071866.1071869

[9] S. GALLAND, N. GAUD, J. DEMANGE, and A. KOUKAM, "Environment model for multiagent-based simulation of 3d urban systems," in *the 7th European Workshop on Multiagent Systems (EUMAS09)*, Ayia Napa, Cyprus, dec 2009, paper 36. [Online]. Available : http://www.multiagent.fr/JaSIM_Platform

[10] J. Gunther, S. Popov, H.-P. Seidel, and P. Slusallek, "Realtime ray tracing on gpu with bvh-based packet traversal," in *Interactive Ray Tracing, 2007. RT '07. IEEE Symposium on*, 2007, pp. 113 –118.

[11] D. R. Horn, J. Sugerman, M. Houston, and P. Hanrahan, "Interactive kd-tree gpu raytracing," in *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, ser. I3D '07. New York, NY, USA : ACM, 2007, pp. 167–174. [Online]. Available : http://doi.acm.org/10.1145/1230100.1230129

[12] Q. Hou, X. Sun, K. Zhou, C. Lauterbach, and D. Manocha, "Memory-scalable gpu spatial hierarchy construction," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 4, pp. 466 –474, 2011.

[13] M. Joselli, E. Passos, M. Zamith, E. Clua, A. Montenegro, and B. Feijo and, "A neighborhood grid data structure for massive 3d crowd simulation on gpu," in *Games and Digital Entertainment (SBGAMES), 2009 VIII Brazilian Symposium on*, 2009, pp. 121 –131.

[14] O. LAMOTTE, S. GALLAND, J.-M. CONTET, and F. GECHTER, "Submicroscopic and physics simulation of autonomous and intelligent vehicles in virtual reality," in *2nd International Conference on Advances in System Simulation (SIMUL10)*. Nice, France : IEEE CPS, 2010. [Online]. Available : http://www.multiagent.fr/Vivus_Platform

[15] C. Lauterbach, M. Garland, S. Sengupta, D. Luebke, and D. Manocha, "Fast bvh construction on gpus," *Computer Graphics Forum*, vol. 28, no. 2, pp. 375–384, 2009. [Online]. Available : http://dx.doi.org/10.1111/j.1467-8659.2009.01377.x

[16] C. Lauterbach, Q. Mo, and D. Manocha, "gproximity : hierarchical gpu-based operations for collision and distance queries," *Computer Graphics Forum*, vol. 29, no. 2, pp. 419–428, 2010. [Online]. Available : http://dx.doi.org/10.1111/j.1467-8659.2009.01611.x

[17] M. Lysenko and R. M. D'Souza, "A framework for megascale agent based model simulations on graphics processing units," *Journal of Artificial Societies and Social Simulation*, vol. 11, no. 4, p. 10, 2008. [Online]. Available : http://jasss.soc.surrey.ac.uk/11/4/10.html

[18] J. Pantaleoni and D. Luebke, "Hlbvh : hierarchical lbvh construction for real-time ray tracing of dynamic geometry," in *Proceedings of the Conference on High Performance Graphics*, ser. HPG '10. Aire-la-Ville, Switzerland, Switzerland : Eurographics Association, 2010, pp. 87–95. [Online]. Available : http://portal.acm.org/citation.cfm?id=1921479.1921493

[19] E. Passos, M. Joselli, M. Zamith, J. Rocha, A. Montenegro, E. Clua, A. Conci, and B. Feij, "Supermassive crowd simulation on gpu based on emergent behavior," in *Proceedings of the VII Brazilian Symposium on Computer Games and Digital Entertainment*, 2008, pp. 81–86.

[20] E. B. Passos, M. Joselli, M. Zamith, E. W. G. Clua, A. Montenegro, A. Conci, and B. Feijo, "A bidimensional data structure and spatial optimization for supermassive crowd simulation on gpu," *Comput. Entertain.*, vol. 7, pp. 60 :1–60 :15, January 2010. [Online]. Available : http://doi.acm.org/10.1145/1658866.1658879

[21] S. Popov, J. Gunther, H.-P. Seidel, and P. Slusallek, "Experiences with streaming construction of sah kd-trees," in *Interactive Ray Tracing 2006, IEEE Symposium on*, 2006, pp. 89 –94.

[22] S. Popov, J. Günther, H.-P. Seidel, and P. Slusallek, "Stackless kd-tree traversal for high performance gpu ray tracing," *Computer Graphics Forum*, vol. 26, no. 3, pp. 415–424, 2007. [Online]. Available : http://dx.doi.org/10.1111/j.1467-8659.2007.01064.x

[23] M. Reichl, R. Dünger, A. Schiewe, M. Hartleb, T. Klemmer, C. Lux, and B. Fröhlich, "Gpu-based ray tracing of dynamic scenes," *Journal of Virtual Reality and Broadcasting*, vol. 7, no. 1, 2010.

[24] P. Richmond and D. Romano, "Agent based gpu, a real-time 3d simulation and interactive visualisation framework for massive agent based modelling on the gpu," 2009. [Online]. Available : http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.144.734

[25] S. M. Rubin and T. Whitted, "A 3-dimensional representation for fast rendering of complex scenes," in *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '80. New York, NY, USA : ACM, 1980, pp. 110–116. [Online]. Available : http://doi.acm.org/10.1145/800250.807479

[26] M. Shevtsov, A. Soupikov, and A. Kapustin, "Highly parallel fast kd-tree construction for interactive ray tracing of dynamic scenes," in *Computer Graphics Forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 395–404.

[27] K. Zhou, Q. Hou, R. Wang, and B. Guo, "Real-time kd-tree construction on graphics hardware," *ACM Trans. Graph.*, vol. 27, pp. 126 :1–126 :11, December 2008. [Online]. Available : http://doi.acm.org/10.1145/1409060.1409079