ID: 77

Collision Detection: Review of Methods and Recent Advances in Crowd Simulation

Norhaida Mohd Suaib
UTM VicubeLab, Faculty of Computing,
Universiti Teknologi Malaysia, Johor, Malaysia,
81310 UTM Skudai
haida@utm.my

Nazreen Abdullasim
Kulliyyah of Information and Communication Technology,
International Islamic University Malaysia,
50728 Kuala Lumpur.
ajendgreat@gmail.com

Iznora Aini Zolkifly
Faculty of Business and Information Technology, UNITAR
International University, Kelana Jaya, Selangor.
iznora@unitar.my

Abdullah Bade
School of Science and Technology,
Universiti Malaysia Sabah,
Jalan UMS, 88400 Kota Kinabalu.
abb@ums.edu.my

***Abstract*— Crowd simulation is a large complex system that visualizes the behavior of crowd entities' movement and their interactions with the virtual environment. Crowd model is usually integrated into a virtual environment to make the environment alive. In the context of agent-based simulation (as in crowd simulation), it encompasses collision checking between moving agents that are present in the same environment. Hence, it is important to design an efficient and yet effective collision detection in crowd simulation. This is to ensure that it is cost effective toward computational processing usage and still produce a believable behavior. This paper presents a study of collision detection techniques in crowd models, and recent advancement to accelerate the process so that in turn, these efforts could also improve the performance and outcome of crowd model in virtual environment applications.**

Keywords-Crowd model, collision detection, crowd simulation, virtual environment, parallel computing

I. INTRODUCTION

A few decades ago, the focus of computer graphics and virtual environment applications is on realism. In those times, research were more focused to finding out ways to improve the look and feel of the environment. However, the trend is changing. Crowd is becoming an important and indispensible element in many computer graphics applications. In real life, we see, meet and interact with many people. Therefore, nicely rendered but unpopulated environment of a virtual city will not be realistic without inhabitants. In fact, it will look like a "ghost town" [1]. With the advancement of technology, virtual environments that was previously sparsely inhabited can now be densely populated with interactively simulated crowd [2, 3].

Crowd simulation applications are not only limited to making life-like virtual cities. Crowd are also used in virtual heritage [4], simulation of events such as in a marathon [5], building/general safety, architectural modeling and urban environment [6]. The list is not definite, as there are variety of applications currently available.

Crowd simulation has been widely used for entertainment, and the demand is on simulation of massive crowd. It may be integrated in computer games or being used for film production. The former is interactive, so it deals not only with the issue of crowd modelling but also maintaining with the interactive frame rate and this poses some challenges. While the latter may not be real-time as the scenes can be processed in post-production, the production time itself is an important measure. The emphasis in this field is to have more realistic crowd movement and aesthetic values. Massive Software, for example, has been successfully used to generate huge crowd in award-winning film "The Lord of the Rings". The crowd modelling software was also used in Disney's "John Carter" and "World War Z" films. Apart from entertainment, crowd simulation also plays an important part in more serious application. These include simulation of emergency circumstances and military simulations.

Crowd modelling involves many disciplines and it is a complex system. Among the disciplines involved, we are going to highlight a very important part in crowd simulation which is collision detection. In order to understand the relationship between crowd simulation and collision detection, this paper will be organized as follows: Section 1 gives the introduction of the topic, followed by more discussion on crowd simulation itself on Section 2. Section 3 discusses on collision detection in crowd simulation, Section 4 focuses on the advancement of collision detection for crowd simulation and Section 5 concludes this paper.

II. CROWD SIMULATION

Crowd simulation is a large complex system that visualizes the behavior of crowd entities movement and their interactions with the virtual environment. The crowd entities are composed of autonomous agents and not the characters which are controlled by the user [7]. Therefore simulating crowd involves many of these areas: psychology, sociology, physics, computer science [8] and mathematics. In computer science itself, crowd simulation can be addressed as in the area of rendering, animation, path-planning and navigation [9]. Different aspects of crowd simulation need to be addressed in order to produce a believable and realistic simulation such as the crowd behavior, surrounding environment as well as rendering aspects [10].

Different researcher uses different approach in categorizing crowd simulation. Pelechano et. al, for example, described that there are two main approaches on how crowd simulation is developed - macroscopic and microscopic approach [11]. Macroscopic approach focuses on crowd entities in larger worldview; for example it is more concerned with the fluidity and the flows of the crowd itself rather than each of the crowd agent's behaviour. On the other hand, microscopic approach is modelled based on agent's individuality. Each agent is represented in their own respective attributes such as their own velocity, position and etcetera. The emergent behaviour resulted from microscopic approach is more wavering and more lifelike than macroscopic approach. However, more computation resources are needed as the number of agents for the crowd simulation increases.

Zhou et. al presented two categories; namely the size and the timescale of crowd simulation. Crowd size (large, medium or small) usually determines the type of approaches being used. The focus of modelling large amount of crowd is more on the global trend of the crowd, while on the other hand, model on the individuals in the crowd can be explored in small to medium size of a crowd simulation. As for timescale of a crowd simulation, the focus of the study is on the formation of long-term or short-term crowd phenomena which is driven by the objective of the crowd [8]. Crowd models are also described based on approach, as listed below:

1. flow-based approach : crowd is modelled as a continuous flow of fluid that lacks individuality. This approach is more suitable in modelling a large crowd's movement

2. entity-based approach : individuals are modelled in a way particles move in physical world under influence of global or local rules

3. agent-based approach : individuals in the crowd are treated as autonomous and intelligent agents

Zhou et. al also introduced an interesting taxonomy based on all of these information – please refer to Figure 1.

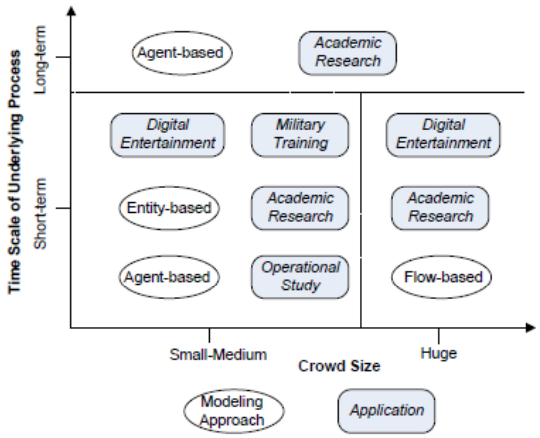

Figure 1. Example of a figure caption. (figure caption)

A. *Components of a Crowd Model*

Behavioural factors include individual movement, emotional states, social relationship that contribute to the dynamics of the crowd. Actions, interactions and reactions in the entity-based and agent-based approach may depend on the following factors:

i. Physical

Physical factors relate to external tangible factors such as speed, position and gesture. Research dealing with only physical factors can reveal the way movement are affected by movement of others.

ii. Social

Crowd are modelled based on theory and observation from social studies in order to investigate into emergent behaviour.

iii. Psychological

Psychological factors deals with emotion of crowd entities. Sometimes can be difficult to model and offers opportunities for further research.

Focus of this paper is more on behavioural aspects, particularly on navigation and movement of the entity of the crowd. They rely on path-planning, collision avoidance mechanisms and the environment itself. In order to create a realistic and physically plausible crowd simulation, collision detection is an important component that ensures that there will be no collision between any two or more agents, or between an agent and static objects. Research on collision detection has been addressed in many areas, such as computer graphics, computer animation, agent-based simulation, robotics, computer games and etcetera.

B. Navigation and Motion in Crowd Simulation

Navigation and motion mainly refer to how crowd moves. It tries to imitate the real world phenomena; therefore any members of the crowd normally would not collide with others (or other objects). In order to achieve this, crowd needs path-planning module and a variety of steering behaviours. One of the important steering behaviour is collision avoidance. Motion of a group of crowd entities reflects the group's behaviour such as flocking.

1) Path-planning

Path-planning ensures collision-free paths for moving objects especially in the fields of robotics, manufacturing, computer games [13], as well as agents in a crowd simulation. Collision detection is one of the components in path-planning and collision avoidance [14]. For sampling-based path planners, collision checking consumes most of the computation time (up to 90 percent or more) and this creates blockage to the system [13, 15, 16].

2) Collision Avoidance

Steering behavior is the basics for autonomous agents' locomotion. It represents a specific desired steering force that an agent executes as a reaction with its virtual environment stimuli. A combination of steering behaviors makes up more complex behavior such as collision avoidance. It basically finds the right steering behaviors in order to avoid the nearest obstacle along its path. The behavior usually comprises of three steps which are construction of the agent's perception, collision handling, and collision response [12], as shown as Figure 2 below.

Figure 2. Example of a figure caption. (figure caption)

3) Proximity Query

Proximity queries and collision detection are two terms that are usually coined together. They deal with testing and reporting on the configuration of pairs of objects. Collision avoidance, proximity queries and path planning are some of the main issues that need to be tackled in order to produce realistic massive crowd simulation of autonomous agents [17].

III. Collision Detection in Crowd Simulation

The collision detection problem has been addressed in many areas, such as computer graphics, computer animation, agent-based simulation, and etcetera [18, 19]. In the context of agent-based simulation, it consists of checking the collisions between agents that freely move within the same geometric space. A collision occurs when the volume occupied by one agent intersects with another agent (this problem can be reduced to a two-dimensional environment, by considering that the two dimensional shape represents each agent instead of its volume). Although collision avoidance techniques have been developed for crowd simulations, they still fail to avoid some collisions when used in high-density environments [7]. Collision avoidance/obstacle avoidance employ collision detection and static non-penetration as part of the process [4].

Rendering crowd that involves lighting and shadows in outdoor scene may involve collision detection between the source of light and the bounding volume that encompasses the crowd component. For example, Axis-Aligned Bounding Box (AABB) was used to surround some crowd entities in order to determine cast shadows [10]. However, rendering is not the focus of this paper and will not be discussed in detail.

In terms of crowd motion, the process of checking for collision takes most of the time in sampling-based path planning algorithms. This is especially crucial in a scene involving a dense population as the total number of collision tests increases in order to find a collision free sample. As in a classical collision detection algorithm, broad-phase algorithms are designed to filter out collision free samples, so that only pairs that have higher probability to collide will be tested in narrow phase algorithms. Although collision avoidance techniques have been implemented in crowd simulations, collisions are sometimes unavoidable especially if it involves a large number of agents.

A. Collision Detection Methods

Collision detection is widely used in a variety of fields; such as computer animation, computer games, visual simulation, crowd simulation, medical, education and training, robotics and virtual reality to name a few. There are different categories of collision detection methods implemented in previous research as will be outlined next Whether the goal is to detect collisions or to generate collision free configuration-space, they "…can be viewed as instances of the same problem, where objects are tested for interference at a particular position, along a trajectory and throughout the whole workspace, respectively." [14]

The main purpose of collision detection is to detect and make sure that there is no overlapping objects that occupy the same space at any time. There are different types of collision detection techniques with different levels of accuracy, depending on the suitability of the application [13]. For example, simulations for medical purposes require very precise collision detection, while simulations for leisure and entertainment (such as computer games) can make do with approximate but fast collision detection.

1) Broad-phase vs. Narrow-phase Collision Detection

Collision detection algorithms usually follow two standard phases: the broad-phase collision detection, followed by the narrow-phase collision detection. Broad-phase collision detection basically will filter out the pairs that do not possibly collide so that only pairs with high possibility to collide will be fed to the narrow-phase collision detection. Testing done in the narrow-phase level are usually more computationally-intensive

compared to the broad-phase level. Therefore the filtering out of unnecessary tests will help improve overall performance.

IV. Improving Collision Detection for Crowd Simulation

Research in crowd simulation recently are focusing more on possible ways to speed up the simulation process. This is due to the fact that some algorithms involved can be the cause of major bottlenecks, and collision detection is one of them [16, 20]. Commodity hardware offers increased computing power through the invention of multi-core and many-core architectures and this advantage has been manipulated to enhance the rendering and motion of a simulated crowd [7] as well as collision detection in general [16]. Manipulation of classical collision detection technique for improved broad-phase collision detection has been investigated [21] and massive simulation can benefit from hybrid collision culling [20].

Real-time simulation of a large crowd involves high computation. In order to increase the look and speed of the simulation, efficient techniques are very much favored. Another option is to use additional/more powerful resources with the current CPU [9]. This opens up the possibility of parallel implementation of the traditional approach.

A. Bounding Volume Manipulation

Crowded virtual environment usually involves large number of interacting virtual inhabitants. In these types of applications, collision tests between virtual inhabitants are usually approximated based on simpler shapes that bound the virtual inhabitants and other objects. AABB and cylinder-based bounding volume are normally used. A fixed-width and pose-independent bounding volumes used for testing collisions may lead to false-positive or even undetected collisions. Moreover, this approach prevents characters from closely mingle with each other [22] thus making it unsuitable for crowded environment. In order to avoid this problem, a bounding volume hierarchy called bounding cylinder hierarchy (BCH) was introduced [22]. Instead of using a single cylinder as the bounding volume, each virtual character will be bounded by a tree of vertical cylinders.

Apart from using basic 3D objects, swept volumes sometimes are used as bounding volume for virtual characters. Based on the shape of the virtual character (particularly the humanoid models), line swept sphere (LSS) or also referred to as 'capsules' were used to represent the humanoid links [23], or combination of LSS and line swept ellipse (LSE) [24]. In a scene involving different types of objects that consists of static and dynamic objects, projection of the bounding volumes for dynamic objects onto the floor were used for collision avoidance between agents and obstacles [25].

An improved version of collision culling technique based on [20] called the S-Dop collision culling was implemented on humanoid model simulation [26]. It improves the overall collision test by greatly reducing the total number of tests needed at the broad-phase collision detection in a scene involving humanoid model (or articulated model) so that only required tests will be performed during the narrow-phase level.

B. Parallel Processing

In a simulation involving n-body object, testing all pairs of objects is likely to result in n2 pairwise checks. One of the strategies to reduce total number of tests is by employing a two-phase test in order to filter the pairs that do not possibly collide. These two phases are known as the broad-phase level where all pairs of objects which have higher possibilities to collide are identified and sent to the narrow-phase level. Irrelevant pairs will be ignored in the broad-phase level, while object pairs sent to the narrow-phase level will be tested with more accurate and exact collision detection method. The pairwise checks is rather 'exhaustive' when implemented sequentially, but most suitable for a parallel implementation [16].

Therefore there are many parallel implementations for collision detection in crowd simulation and other applications. "Sweep and Prune" (SaP) (also known as "Sort and Sweep") is one of the most widely used approach in broad-phase level using bounding volume technique, particularly the Axis-Aligned Bounding Box (AABB). The usual 3D collision tests for AABB is reduced into three separate 1D test; one each for the bounding volume projected onto the X, Y and Z-axes respectively. These will be used as parallel threads during parallel execution. A sample parallel implementation using this approach is shown in Figure 3 below:

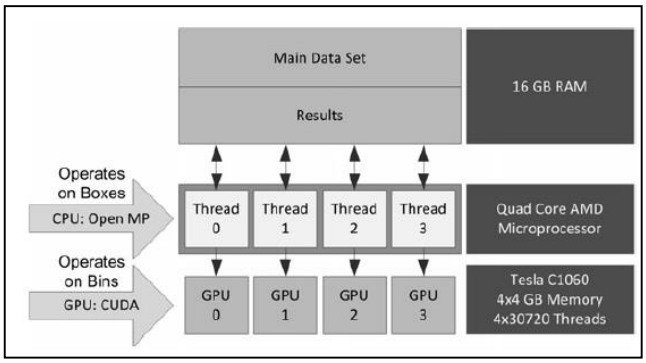

Figure 3. Sample parallel implementation based on "Sort and Sweep" on CPU and GPU (source: [27])

Pan & Manocha (2010) explored parallel algorithms in sampling-based motion planning. The main objective is to accelerate collision queries by manipulating the data-parallelism and multi-threaded capabilities on the GPU. They reported an increased speed of 50-100X compared to previous CPU-based motion-planners [15].

There is an option in doing things in parallel: implementation on many-core or multi-core architecture. Different researchers may use different approach. For example, massive crowd in The Virtual Marathon take advantage of both parallel processing on CPU and GPU, but the GPU handles most of the rendering process while the CPU is responsible to other processes such as collision detection [5]. Another research compared the performance of parallel CPU-based and GPU-based broad-phase collision detection. Speedups recorded were up to 134 times faster for GPU implementation, as compared to its CPU implementation [13].

Another research by Vigueras (2013) on parallelization and code development strategies for multi-core and many-core architectures shows that the GPU greatly accelerates the collision test compared to other implementation optimized for multi-core CPUs [7].

V. CONCLUSION

This paper presented an overview of crowd simulation, its categories based on different researchers and its component in general. It highlights collision detection as an important component in order to create a realistic and physically plausible crowd simulation. Although collision detection is used in a variety of application areas, the underlying aim is to ensure that there will not be multiple objects/agents occupying the same space at the same time. Since collision detection can create bottlenecks to the crowd simulation (as in other graphical simulation), efforts were carried out to improve the process. This can be seen by the manipulation of traditional collision detection methods, and adaptation of collision detection for crowd simulation in parallel processing.

ACKNOWLEDGMENT

This research is supported by Universiti Teknologi Malaysia (UTM) and Ministry of Higher Education (MOHE), in collaboration with Research Management Centre (RMC), UTM, and partly supported by Research University Grant (RUG) Program, Tier 1 (research grant 09H18). The authors would also like to express our gratitude to various participating universities and GRAVS members, for the support and coordination towards making this collaboration a success.

REFERENCES

[1] N. I. Badler and C. O'Sullivan. (2009) Guest Editors' Introduction Computing Now, special issue on Populating Virtual Worlds. Available: http://www.computer.org/portal/web/computingnow/archive/august2009

[2] K. H. Lee, M. G. Choi, and J. Lee, "Motion patches: building blocks for virtual environments annotated with motion data," ACM Trans. Graph., vol. 25, pp. 898-906, 2006.

[3] B. Yersin, J. Ma¨ım, J. Pettre, and D. Thalmann, "Crowd patches: populating large-scale virtual environments for real-time applications," presented at the Proceedings of the 2009 symposium on Interactive 3D graphics and games, Boston, Massachusetts, 2009.

[4] L. Heïgeas, A. Luciani, J. Thollot, and N. Castagné, "A physically-based particle model of emergent crowd behaviors," in CoRR vol. abs/1005.4405, ed, 2010.

[5] E. Yilmaz. (2009) The Virtual Marathon: Parallel Computing Supports Crowd Simulations. 26-33. Available: http://doi.ieeecomputersociety.org/10.1109/MCG.2009.77

[6] T. Jund, P. Kraemer, and D. Cazier, "A unified structure for crowd simulation," Comput. Animat. Virtual Worlds, vol. 23, pp. 311-320, 2012.

[7] G. Vigueras, J. M. Orduña, M. Lozano, J. M. Cecilia, and J. M. García, "Accelerating collision detection for large-scale crowd simulation on multi-core and many-core architectures," International Journal of High Performance Computing Applications, February 19, 2013 2013.

[8] S. Zhou, D. Chen, W. Cai, L. Luo, M. Y. H. Low, F. Tian, V. S.-H. Tay, D. W. S. Ong, and B. D. Hamilton, "Crowd modeling and simulation technologies," ACM Trans. Model. Comput. Simul., vol. 20, pp. 1-35, 2010.

[9] M. Haciomeroglu, O. Barut, C. Y. Ozcan, and H. Sever, "A GPU-assisted hybrid model for real-time crowd simulations," Computers & Graphics, vol. 37, pp. 862-872, 2013.

[10] D. Thalmann and S. R. Musse, Crowd Simulation, Second Edition ed. London: Springer, 2013.

[11] N. Pelechano, J. M. Allbeck, and N. I. Badler, Virtual Crowds: Methods, Simulation, and Control: Morgan & Claypool Publishers, 2008.

[12] N. Abdullasim, A. Hoirul Basori, M. S. Salam, and A. Bade, "Velocity Perception: Collision Handling Technique for Agent Avoidance Behavior," TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 11, pp. 2264-2270, 2013.

[13] F. Geleri, O. Tosun, and H. Topcuoglu, "Parallelizing Broad Phase Collision Detection Algorithms for Sampling Based Path Planners," in Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on, 2013, pp. 384-391.

[14] P. Jiménez, F. Thomas, and C. Torras, "Collision detection algorithms for motion planning," in Robot Motion Planning and Control. vol. 229, J. P. Laumond, Ed., ed: Springer Berlin Heidelberg, 1998, pp. 305-343.

[15] J. Pan and D. Manocha, "GPU-Based Parallel Collision Detection for Real-Time Motion Planning," in Algorithmic Foundations of Robotics IX. vol. 68, D. Hsu, V. Isler, J.-C. Latombe, and M. Lin, Eds., ed: Springer Berlin Heidelberg, 2011, pp. 211-228.

[16] Q. Avril, V. Gouranton, and B. Arnaldi, "Dynamic adaptation of broad phase collision detection algorithms," in VR Innovation (ISVRI), 2011 IEEE International Symposium on, 2011, pp. 41-47.

[17] T. Jund, P. Kraemer, and D. Cazier, "A unified structure for crowd simulation," Computer Animation and Virtual Worlds, vol. 23, pp. 311-320, 2012.

[18] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M. P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino, "Collision Detection for Deformable Objects," Computer Graphics Forum, vol. 24, pp. 61-81, 2005.

[19] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey, "ClearPath: highly parallel collision avoidance for multi-agent simulation," presented at the Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, New Orleans, Louisiana, 2009.

[20] N. M. Suaib, A. Bade, and D. Mohamad, "Hybrid Collision Culling by Bounding Volumes Manipulation in Massive Rigid Body Simulation," IAES Journal of Electrical and Electronics Engineering, vol. 11, pp. 3115-3122, 2013.

[21] N. M. Suaib, A. Bade, and D. Mohamad, "Collision Detection Using Bounding-Volume for Avatars in Virtual Environment Applications," presented at the The 4th International Conference on Information & Communication Technology and Systems (ICTS), Surabaya, Indonesia, 2008.

[22] S. A. Stüvel, N. Magnenat-Thalmann, D. Thalmann, A. Egges, and A. F. van der Stappen, "Hierarchical structures for collision checking between virtual characters," Computer Animation and Virtual Worlds, vol. 25, pp. 333-342, 2014.

[23] Y. J. Kim, S. Redon, M. C. Lin, D. Manocha, and J. Templeman, "Interactive Continuous Collision Detection Using Swept Volume for Avatars," Presence: Teleoperators and Virtual Environments, vol. 16, pp. 206-223, 2007/04/01 2007.

[24] C. Dube, M. Tsoeu, and J. Tapson, "A model of the humanoid body for self collision detection based on elliptical capsules," in Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on, 2011, pp. 2397-2402.

[25] S. Kim, S. J. Guy, and D. Manocha, "Velocity-based modeling of physical interactions in multi-agent simulations," presented at the