TINJAUAN TERHADAP PENGESANAN PELANGGARAN DALAM PERSEKITARAN MAYA INTERAKTIF

Norhaida Mohd Suaib, Abdullah Bade, Daut Daman, Mohd Shahrizal Sunar

Jabatan Grafik Berkomputer dan Multimedia,
Fakulti Sains Komputer dan Sistem Maklumat,
Universiti Teknologi Malaysia,
81310 Skudai Johor.
Tel +607-5532324
E-mail: {haida, abd-lah, daut, shah}@fsksm.utm.my

Abstrak:

Bidang pengesanan pelanggaran merupakan elemen penting dalam aplikasi persekitaran maya interaktif. Tanpa elemen ini, aplikasi yang dihasilkan adalah kurang realistik dan menarik. Kajian yang berterusan hampir 15 tahun di dalam bidang pengesanan pelanggaran menyaksikan pelbagai kaedah dan teknik dikemukakan oleh para penyelidik grafik berkomputer. Namun, kaedah dan teknik yang dicadangkan amat bergantung kepada model objek yang digunakan dan bentuk aplikasi yang akan dihasilkan. Justeru, kertas kerja ini akan membincangkan bidang pengesanan pelanggaran dalam aplikasi persekitaran maya interaktif kepada empat bahagian. Keempat-empat bahagian tersebut mencakupi pengenalan mengenai bidang pengesanan pelanggaran, klasifikasi kaedah pengesanan pelanggaran, peringkat-peringkat pengesanan pelanggaran dan aplikasi pengesanan pelanggaran dalam persekitaran maya seperti permainan berkomputer.

Katakunci: Pengesanan pelanggaran, grafik berkomputer, persekitaran maya, permainan berkomputer

1.0 Pendahuluan

Perkembangan teknologi pengkomputeran yang pantas pada hari ini telah merangsang peningkatan penyelidikan dalam bidang grafik berkomputer. Bermula dari penghasilan satu titik dalam persekitaran 2-dimensi (2D); penyelidikan terus dijalankan kepada persekitaran 3-dimensi (3D) serta melibatkan penghasilan aplikasi yang rumit dan memerlukan kepantasan dalam pengiraan. Malahan kemajuan dari sudut teknologi pengkomputeran turut meletakkan bidang grafik berkomputer yang interaktif sebagai satu alat praktikal khususnya dalam bidang sains, kejuruteraan, perubatan, perniagaan, perindustrian, pentadbiran, kesenian, hiburan, pengiklanan, pendidikan dan latihan [1,2,3,4].

Dalam hal ini, penghasilan aplikasi grafik berkomputer sebagai satu alat praktikal tidak dapat tidak berkehendak kepada permodelan objek yang realistik, animasi yang mempunyai pergerakan licin dan pengesanan pelanggaran yang efisien serta mempunyai kepantasan yang baik [1,4] .

2.0 Pengesanan Pelanggaran

Selari dengan perkembangan di lapangan grafik berkomputer, perkembangan yang sama juga turut berlaku di bidang pengesanan pelanggaran. Di awal penyelidikan hampir 15 tahun yang lalu, bidang pengesanan pelanggaran lebih tertumpu di lapangan geometri pengiraan, aplikasi pembuatan dan robotik [5]. Kini, hampir segenap bidang dalam grafik berkomputer seperti permodelan berasaskan fizikal, simulasi pembedahan, realiti maya, permainan berkomputer, animasi dan juga perfileman memerlukan teknik dan kaedah pengesanan pelanggaran [1,6].

Penyelidikan yang berterusan selama belasan tahun ini telah mendorong penyelidik di dalam bidang grafik berkomputer mengemukakan beberapa kaedah dan algoritma dalam bidang pengesanan pelanggaran. Sungguhpun demikian, keupayaan kaedah dan algoritma tersebut masih terbatas kepada jenis aplikasi dan objek yang digunakan, kerumitan ruang adegan serta kurang efisien untuk aplikasi yang memerlukan kepantasan dan pengesahan yang baik [1,3,5,7,8].

Rajah 1.1: Salah satu babak pergerakan model tangan dan model sebiji epal dalam persekitaran maya

Secara umum, teknik pengesanan pelanggaran ini boleh digambarkan melalui situasi berikut. Katakan dalam satu ruang adegan persekitaran maya terdapat dua objek maya yang boleh dimanipulasi oleh pengguna; contohnya model tangan yang boleh digerakkan dan model buah epal. Kedua-dua objek tersebut telah dijana menggunakan gabungan poligon mudah iaitu segitiga. Tanpa pengesanan pelanggaran, pergerakan objek pertama (tangan) untuk menggenggam objek kedua (buah epal) adalah mustahil untuk dilakukan memandangkan pergerakan tangan adalah bergantung kepada tindakan pengguna dan tidak sepatutnya ditetapkan atau dihadkan oleh pembangun. Ini adalah disebabkan tiada cara untuk mengenalpasti sama ada model epal berjaya dicapai oleh tangan, dan dengan itu penembusan antara kedua-dua objek tersebut adalah lebih cenderung untuk berlaku. Simulasi atau persekitaran maya yang realistik sepatutnya dapat memberikan hasil yang menyerupai keadaan sebenar; iaitu epal yang boleh dicapai dan dipegang oleh tangan. Justeru, teknik pengesanan pelanggaran diperlukan untuk mengatasi situasi ini. Walau bagaimanapun, kesesuaian kaedah yang digunakan dalam teknik pengesanan pelanggaran adalah bergantung kepada tujuan aplikasi grafik berkomputer yang dibangunkan.

Faktor realistik, ketepatan yang tinggi, kepantasan dalam menentukan pelanggaran dan masa nyata sering diambil kira dalam membangunkan strategi pengesanan pelanggaran yang efisien dan berkuasa [1,4,6].

Dalam bidang robotik umpamanya, faktor kepantasan dan masa nyata mungkin kurang diberi penekanan. Pengesanan pelanggaran dalam bidang itu memberi fokus yang lebih besar terhadap faktor ketepatan yang tinggi dan realistik. Sebagai contoh pemotongan bongkah logam yang dilakukan oleh robot memberi lebih tumpuan kepada faktor ketepatan walaupun proses tersebut akan mengambil masa yang lama.

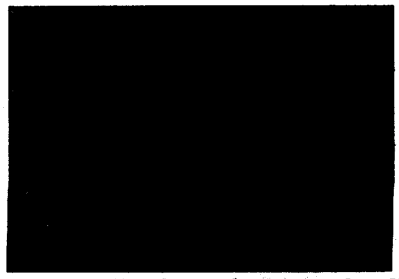
Situasi ini berbeza bagi aplikasi persekitaran maya (APM) yang interaktif seperti aplikasi permainan berkomputer, penjelajahan dan pencarian laluan terhampir. Faktor realistik, kepantasan dan masa nyata dalam menentu sahkan pelanggaran di antara objek diberi penekanan yang mendalam. Sebagai contoh, dalam permainan berkomputer jika kaedah pengesanan pelanggaran tidak mempunyai kepantasan yang tinggi dalam menentu sahkan pelanggaran, permainan akan menjadi kurang menarik. Walau bagaimanapun, jika kaedah pengesanan pelanggaran itu mempunyai kepantasan yang tinggi tetapi banyak pelanggaran yang tidak dapat dicerap (*miss*) dalam masa nyata, permainan tersebut akan menjadi kurang bermakna. Atau dalam situasi yang lain pelanggaran dapat dikesan tetapi secara visualnya pelanggaran kelihatan tidak berlaku. Justeru, pemain pasti akan berasa tertipu disebabkan faktor kurang realistik.

Apa yang lebih penting dalam hal ini, kaedah pengesanan pelanggaran untuk APM interaktif perlu memenuhi faktor kepantasan yang tinggi di samping memberikan hasil yang realistik. Ujian sehingga peringkat menentusahkan permukaan dan titik objek yang terlibat dalam pelanggaran adalah kurang perlu. Ujian seumpama ini diperlukan dalam bidang robotik dan sebahagian aplikasi realiti maya yang menekankan elemen ketepatan seperti simulasi pembedahan dan latihan tertentu.

Dalam kajian aplikasi grafik 3D yang terdiri daripada jutaan poligon, proses pengesanan pelanggaran di antara poligon-poligon adalah mencabar. Secara umum, antara strategi penyelesaian pengesanan pelanggaran dalam APM yang telah dicadangkan oleh penyelidik dalam bidang grafik bekomputer ialah menggunakan kaedah struktur data dan juga penggunaan kaedah jarak dan masa [4,9]. Walau bagaimanapun, penggunaan kaedah struktur data akan membawa beberapa isu yang perlu diambil kira seperti :

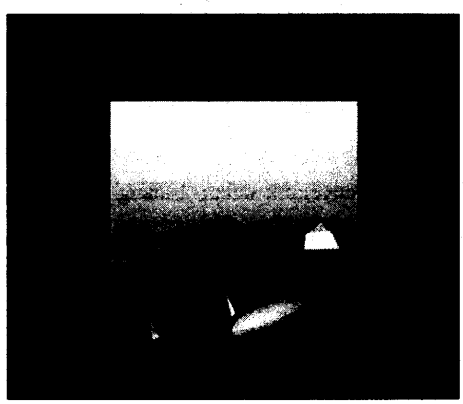
i. Darjah pepohon yang bersesuaian (ketinggian dan masa yang diperlukan untuk menentukan pelanggaran setiap nod) dan
ii. Hukum Pemisah yang lebih sesuai dalam membangunkan hirarki pepohon [1,3,5,10].

Sementara itu, penggunaan kaedah pengesanan pelanggaran menggunakan pendekatan jarak dan masa akan lebih berkesan sekiranya jarak antara dua objek berada dalam keadaan selari yang terhampir atau masa pelanggaran dapat diramalkan [4]. Lazimnya, kaedah yang digunakan untuk menentu sahkan pelanggaran di antara dua atau lebih objek perlu mengambil kira faktor kerumitan (*complexity*) objek yang terkandung dalam adegan persekitaran maya dan sifat pelanggaran antara objek (*the nature of collision*) [4].



Rajah 2.1: Contoh pelanggaran yang melibatkan pelanggaran kendiri (*self-collision detection*) - riben yang merenyuk[4]

Faktor kerumitan objek yang terkandung dalam adegan persekitaran maya terbahagi kepada dua faktor yang berasingan. Pertama adalah faktor kerumitan objek yang dijana. Kedua adalah faktor kerumitan persekitaran maya yang digunakan. Merujuk kepada faktor yang pertama, perkara yang akan dipertimbangkan adalah bilangan poligon yang digunakan untuk menjana objek. Sebagai contoh, penjanaan permukaan riben yang merenyuk atau objek yang lebih rumit bentuknya sudah tentu memerlukan poligon yang lebih banyak berbanding silinder atau objek yang kurang rumit (Rajah 2.1). Justeru, jika ujian pengesanan pelanggaran dilakukan ke atas setiap poligon yang menjana permukaan yang rumit, ujian tersebut dikatakan kurang efisien. Ini kerana banyak masa diperlukan hanya untuk melakukan pengiraan dan perbandingan ke atas setiap poligon.

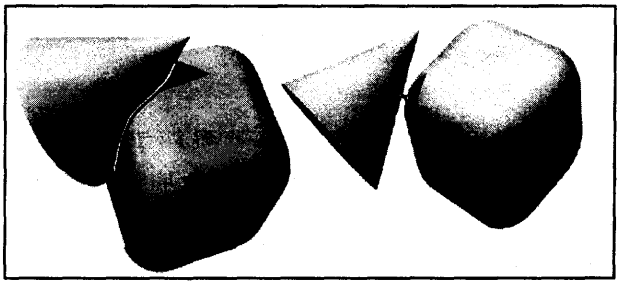


Rajah 2.2: Menentusahkan pelanggaran di antara lebih daripada satu objek

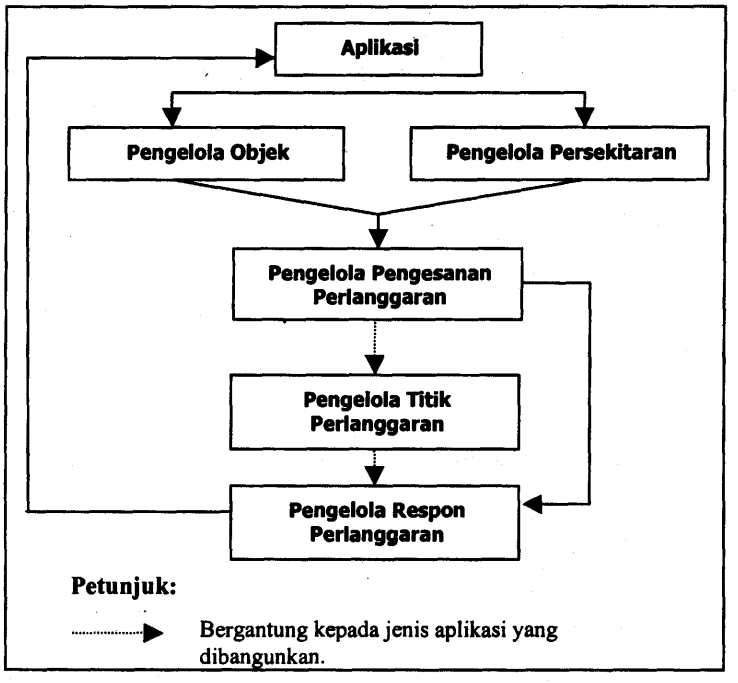
Untuk faktor kerumitan persekitaran maya yang digunakan, perkara yang perlu diambil kira adalah bilangan objek yang terkandung di dalam ruang adegan persekitaran maya (Rajah 2.2). Semakin banyak objek yang terkandung dalam ruang adegan persekitaran maya, kompleksiti ujian pengesanan pelanggaran juga turut meningkat. Perkara ini akan bertambah rumit sekiranya ruang adegan maya bukan sahaja terdiri daripada banyak objek yang mengalami pergerakan, malahan turut mengandungi pelbagai jenis objek statik.

Untuk faktor sifat pelanggaran antara objek, ia merujuk kepada proses mengesan penembusan yang berlaku ataupun mengesan jarak pemisah di antara objek. Atau dengan kata lain ia merujuk kepada proses perlaksanaan operasi mengesan pelanggaran antara objek dalam persekitaran maya yang dijana. Secara

umum proses ini juga terbahagi kepada dua kategori (Rajah 2.3). Pertama, mengesan pelanggaran selepas objek itu berlanggar (telah berlaku penembusan) dan kedua, proses untuk mengesan pelanggaran di mana objek berada dalam keadaan yang terlalu hampir sebelum pelanggaran berlaku (keadaan selari yang terhampir – *parallel close proximity*). Namun demikian, kedua-dua proses ini banyak digunakan khususnya untuk mengesan pelanggaran dalam peringkat *Narrow-phase* (penerangan untuk fasa ini diberikan di bahagian 4.0).



Rajah 2.3: Pengesanan pelanggaran melalui proses penembusan dan jarak pemisah antara objek [4]



Rajah 2.4: Kerangka kerja sistem pengesanan pelanggaran secara umum [1,11]

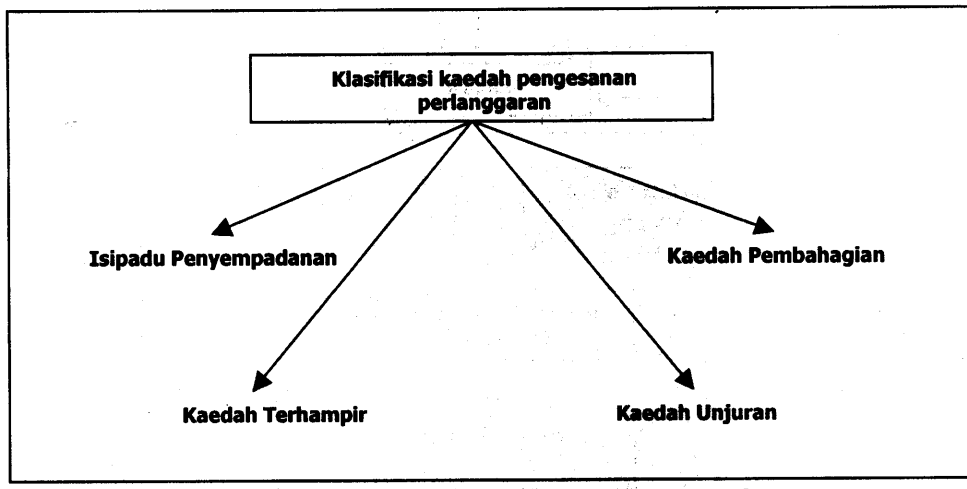
Selain daripada dua perkara di atas, pengesanan pelanggaran juga sering dibincangkan bersama-sama dengan respon pelanggaran (Rajah 2.4). Secara umum, bidang pengesanan pelanggaran terbahagi kepada tiga fasa iaitu fasa mengesan pelanggaran, fasa menentukan titik pelanggaran dan fasa respon pelanggaran [1,11].

Fasa penentuan titik pelanggaran adalah fasa menentukan dengan lebih terperinci bahagian objek yang terlibat dengan pelanggaran. Lazimnya, penentuan titik pelanggaran digunakan untuk aplikasi yang

memerlukan ketepatan tinggi seperti dalam robotik dan simulasi pembedahan serta penerbangan. Sementara fasa respon pelanggaran pula diperlukan untuk meningkatkan lagi ciri-ciri sebenar dunia nyata. Melalui tindak balas fizikal yang digunakan ke atas objek selepas disahkan berlaku pelanggaran, aplikasi yang dibangunkan kelihatan lebih realistik. Dalam kertas kerja penyelidikan ini, penulis tidak akan membincangkan dua fasa ini dengan lebih terperinci. Ini kerana penyelidikan yang dijalankan lebih memfokuskan kepada fasa pengesanan pelanggaran sahaja.

3.0 Klasifikasi Kaedah Pengesanan Pelanggaran

Kepelbagaian kaedah yang diketengahkan dalam bidang pengesanan pelanggaran mendorong penyelidik di dalam bidang grafik berkomputer mengklasifikasikan kaedah-kaedah tersebut mengikut beberapa kategori tertentu [1,4,6]. Proses klasifikasi dilakukan berasaskan kesesuaian domain yang diguna pakai. Empat kumpulan utama klasifikasi tersebut ditunjukkan seperti Rajah 3.1.



Rajah 3.1: Klasifikasi algoritma pengesanan pelanggaran

Kaedah yang diklasifikasikan di bawah kumpulan Isipadu Penyempadanan (*Bounding Volume*) akan menyempadani objek atau kumpulan objek dengan polihedra mudah seperti kuboid, kubus dan sfera. Penggunaan penyempadanan polihedra ini akan membantu mengurangkan ujian pengesanan pelanggaran pasangan-pasangan objek dalam setiap kerangka masa. Ini disebabkan ujian pelanggaran antara poligon secara keseluruhan (segitiga dan titik) boleh diabaikan [2,5,10,12].

Kaedah yang dikategorikan di bawah kumpulan Kaedah Pembahagian (*Subdivision Method*) akan memecahkan objek dan ruang adegan kepada sel-sel kecil berbentuk kubus atau kuboid. Ujian pengesanan pelanggaran dilakukan berdasarkan kedudukan dan orientasi objek yang berada dalam sel-sel. Jika didapati satu objek memasuki sel yang dipunyai oleh objek yang lain, ada kemungkinan telah berlaku pelanggaran. Ujian seumpama ini lazimnya digunakan untuk pemecahan ruang adegan persekitaran maya. Sementara untuk pemecahan objek, ujian dilakukan ke atas setiap sel yang menyempadani poligon yang menjana objek.

Dengan kata lain, sel yang didapati berlanggar menunjukkan kemungkinan berlakunya pelanggaran di antara poligon terbabit. Pemecahan jenis ini sesuai untuk aplikasi yang memerlukan ketepatan yang tinggi.

Untuk klasifikasi di bawah kumpulan Kaedah Terhampir (*Proximity Method*), kaedah pengesanan pelanggaran terlebih dahulu menyusun objek berdasarkan jiran (*neighbour*) yang terhampir. Seterusnya, proses ujian berdasarkan jarak akan dilakukan untuk menentu sahkan pelanggaran. Sementara itu, kaedah di bawah kumpulan Kaedah Unjuran (*Projection Method*) pula menguji unjuran objek bagi setiap kerangka masa berdasarkan paksi atau permukaan objek secara berasingan.

4.0 Peringkat *Broad-phase* dan *Narrow-phase*

Secara lazim, hampir kesemua kaedah pengesanan pelanggaran yang telah dibangunkan akan mematuhi dua peringkat piawai dalam proses menentu sahkan pelanggaran. Kedua-dua peringkat ini masing-masing dikenali sebagai peringkat *Broad-phase* dan *Narrow-phase* [1,7,13,14].

Para penyelidik di bidang pengesanan pelanggaran mencadangkan kedua-dua peringkat tersebut sebagai satu strategi menstruktur langkah penyelesaian dalam mengesan pelanggaran antara objek. Di samping itu, penstrukturan ini juga secara tidak langsung membantu menangani tiga isu utama dalam aplikasi realiti maya iaitu respon masa nyata, pengiraan efisien dan pelaksanaan pantas [4,5,10,11].

Selain daripada itu, terdapat juga sesetengah penyelidik membentuk satu peringkat tunggal langkah penyelesaian dalam mengesan pelanggaran. Peringkat tunggal ini akan menggabungkan kedua-dua peringkat lazim tersebut [13]. Justeru, melalui gabungan yang dilakukan, pengiraan terhadap kesemua kemungkinan berlakunya pelanggaran dilakukan dalam peringkat yang sama dengan pengiraan untuk menentukan titik dan permukaan pelanggaran.

4.1 Peringkat *Broad –Phase*

Dalam peringkat *Broad-phase*, kaedah pengesanan pelanggaran akan mengenal pasti pasangan-pasangan objek yang mempunyai kemungkinan untuk berlanggar. Pada masa yang sama, pasangan-pasangan objek yang tidak mempunyai potensi untuk berlanggar tidak akan dipertimbangkan. Untuk kes seperti algoritma *Brute Force* [13,15], kerumitan dalam menentu sahkan pelanggaran untuk peringkat ini ialah $O(n^2)$, dengan n mewakili bilangan objek. Selepas proses pembuangan dilakukan, kerumitan algoritma dikurangkan kepada $O(m^2)$ dengan m merupakan bilangan permukaan yang menjana setiap objek.

Secara umum, kaedah pengesanan pelanggaran untuk peringkat *Broad-phase* adalah bersifat konservatif. Hampir kesemua kaedah-kaedah yang digunakan dalam peringkat ini melibatkan pengiraan ujian pengesanan pelanggaran yang mudah [1,16]. Satu ciri utama peringkat *Broad-phase* ialah keupayaannya untuk tidak bergantung kepada peringkat yang lain dalam menentu sahkan pelanggaran [1,15]. Contoh

terbaik kaedah yang melaksanakan peringkat ini ialah penyempadanan objek menggunakan kaedah isipadu penyempadanan dan pembahagian ruang persekitaran yang diduduki oleh objek, dengan beberapa contoh kaedah spesifik disenaraikan seperti berikut:

[a] Isipadu Penyempadanan - *Axis-aligned Bounding Box* (AABB), *Bounding Sphere, Oriented Bounding Box* (OBB) dan *Discrete Orientation Polytope* (k-Dops) [2,3,10,12].

[b] Pembahagian Ruang – *Octree, BSP-Tree, Time-Coherence* (*Four-Dimensional Space-time Bounding*) [5,16].

4.2 Peringkat *Narrow-Phase*

Kaedah pengesanan pelanggaran dalam peringkat *Narrow-phase* bergantung sepenuhnya kepada kaedah dalam peringkat *Broad-phase*. Ujian mengenal pasti dan membuang pasangan-pasangan objek dalam peringkat *Broad-phase* masih diteruskan dan dijalankan dengan lebih terperinci. Ujian tersebut akan melibatkan permukaan serta titik/verteks yang menjana objek. Justeru, proses untuk menentu sahkan pelanggaran dalam peringkat ini mengambil masa yang lebih lama jika dibandingkan dengan peringkat *Broad-phase*.

Ujian pengesanan pelanggaran dengan ketepatan yang tinggi akan melibatkan ujian-ujian penembusan, terputus dan penentuan jarak [15]. Selain daripada itu, terdapat juga kaedah pengesanan pelanggaran dalam peringkat *Narrow-phase* yang memulangkan nilai boolean. Nilai boolean ini akan bertindak sebagai penunjuk jika berlaku situasi penembusan antara objek [15]. Contoh terbaik kaedah di bawah peringkat ini adalah kaedah *Lin-Canny Closest Features* [1,10].

Seperti mana peringkat *Broad-phase*, peringkat *Narrow-phase* juga mempunyai keistimewaan yang tersendiri. Keupayaannya untuk menguji titik atau verteks yang terlibat dalam pelanggaran menjadikan peringkat ini sesuai digunakan dalam bidang robotik dan aplikasi persekitaran maya yang menekankan elemen-elemen fizik dengan ketepatan yang tinggi seperti simulasi pembedahan dan penerbangan. Dalam peringkat ini juga, proses ujian pelanggaran yang dijalankan dibahagikan kepada dua peringkat yang lain. Kedua-dua peringkat ini dikenali sebagai peringkat *Exact* dan peringkat *Progressive Refinement* [1].

Peringkat *Exact* turut dibahagikan kepada dua peringkat yang lain iaitu peringkat *Features-based Method* sebagai contoh kaedah *Lin-Canny* dan *V-Clip* dan peringkat *Simplex-based Method* sebagai contoh kaedah *Gibson, Johnson and Keerthy* (GJK). Sementara peringkat *Progressive Refinement* lebih banyak menggunakan pendekatan BV dan *Spatial Decomposition* [17].

5.0 Aplikasi Pengesanan Pelanggaran Dalam Permainan Berkomputer

Peningkatan yang menggalakkan dalam industri Permainan Berkomputer (PB) dari tahun ke tahun mendorong pembangun dalam industri itu menghasilkan aplikasi PB yang pelbagai dan tersendiri. Justeru, PB yang dihasilkan bukan sahaja mempunyai grafik yang menarik malah berkemampuan untuk melakukan proses *rendering* dan memuat jutaan poligon dengan pantas dan efisien. Untuk menambahkan lagi keunikan yang sedia ada, elemen pengesanan pelanggaran turut dimasukkan. Kesannya, situasi seperti menembak musuh, mengelak sebarang rintangan, membuka pintu kereta dan bahkan yang paling mudah seperti berdiri di atas lantai kelihatan lebih realistik.

Dalam hal ini, pengesanan pelanggaran di dalam PB memerlukan strategi yang pantas, efisien dan berkuasa bagi mengekalkan ciri realistik dan menarik pada kadar yang interaktif. Justeru, jika kaedah pengesanan pelanggaran terlalu rumit dan melibatkan pengiraan yang banyak, maka PB tersebut menjadi perlahan dan membosankan.

Kebanyakan kaedah pengesanan pelanggaran yang digunakan dalam PB adalah kaedah BV. Disebabkan pembangunannya mudah di samping pantas dalam mengesan pelanggaran, maka ia merupakan pilihan utama yang digunakan oleh ramai pembangun PB. Sebagai contoh, rata-rata pembangun perisian PB akan menggunakan pendekatan BV seperti *Bounding Sphere*, *Axis-aligned Bounding Box* (AABB), *Oriented Bounding Box* (OBB), Satah, Silinder, Ellipsoid dan pepohon-BSP (BSP-*tree*) untuk mengesan pelanggaran dalam PB. Gambaran yang lebih jelas mengenai perkara ini boleh dirujuk pada lelaman seperti Gamasutra[1], Flipcode[2] dan GameDev.net[3].

Dalam pada itu, Ming. C. Lin [18] turut mencadangkan kaedah untuk menguji pelanggaran dalam PB. Asas yang digunakan adalah menguji keadaan selari yang terhampir di antara pelbagai model poligon. Kaedah pengesanan PB yang dicadangkan melibatkan gabungan beberapa kaedah. Untuk mengesan pelanggaran yang melibatkan objek-objek cembung, kaedah peningkatan akan digunakan [18]. Kaedah ini akan menguji pelanggaran berdasarkan jarak objek-objek cembung yang diperolehi daripada maklumat topologi objek pada setiap kerangka masa. Dalam hal ini, perubahan untuk setiap kerangka masa perlu diskrit dan kecil agar ujian dapat dilakukan dengan berkesan. Keadaan ini dinamakan sebagai *geometric coherence* atau *continuity of motion*.

Untuk pengesanan melibatkan objek cekung, objek tersebut akan dibahagikan dan membentuk gabungan objek cembung. Pengesanan pelanggaran yang melibatkan hirarki objek rumit ini akan menggunakan pendekatan pepohon OBB. Pembinaan pepohon ini menggunakan kaedah atas ke bawah sementara ujian antara nod dalam hirarki menggunakan kaedah Teorem Paksi Pemisah [18]. Kaedah ini sebenarnya akan

[1] http://www.gamasutra.com
[2] http://www.flipcode.com
[3] http://www.gamedev.net

5.0 Aplikasi Pengesanan Pelanggaran Dalam Permainan Berkomputer

Peningkatan yang menggalakkan dalam industri Permainan Berkomputer (PB) dari tahun ke tahun mendorong pembangun dalam industri itu menghasilkan aplikasi PB yang pelbagai dan tersendiri. Justeru, PB yang dihasilkan bukan sahaja mempunyai grafik yang menarik malah berkemampuan untuk melakukan proses *rendering* dan memuat jutaan poligon dengan pantas dan efisien. Untuk menambahkan lagi keunikan yang sedia ada, elemen pengesanan pelanggaran turut dimasukkan. Kesannya, situasi seperti menembak musuh, mengelak sebarang rintangan, membuka pintu kereta dan bahkan yang paling mudah seperti berdiri di atas lantai kelihatan lebih realistik.

Dalam hal ini, pengesanan pelanggaran di dalam PB memerlukan strategi yang pantas, efisien dan berkuasa bagi mengekalkan ciri realistik dan menarik pada kadar yang interaktif. Justeru, jika kaedah pengesanan pelanggaran terlalu rumit dan melibatkan pengiraan yang banyak, maka PB tersebut menjadi perlahan dan membosankan.

Kebanyakan kaedah pengesanan pelanggaran yang digunakan dalam PB adalah kaedah BV. Disebabkan pembangunannya mudah di samping pantas dalam mengesan pelanggaran, maka ia merupakan pilihan utama yang digunakan oleh ramai pembangun PB. *Sebagai contoh, rata-rata pembangun perisian PB akan* menggunakan pendekatan BV seperti *Bounding Sphere, Axis-aligned Bounding Box* (AABB), *Oriented Bounding Box* (OBB), Satah, Silinder, Ellipsoid dan pepohon-BSP (BSP-*tree*) untuk mengesan pelanggaran dalam PB. Gambaran yang lebih jelas mengenai perkara ini boleh dirujuk pada lelaman seperti Gamasutra[1], Flipcode[2] dan GameDev.net[3].

Dalam pada itu, Ming. C. Lin [18] turut mencadangkan kaedah untuk menguji pelanggaran dalam PB. Asas yang digunakan adalah menguji keadaan selari yang terhampir di antara pelbagai model poligon. Kaedah pengesanan PB yang dicadangkan melibatkan gabungan beberapa kaedah. Untuk mengesan pelanggaran yang melibatkan objek-objek cembung, kaedah peningkatan akan digunakan [18]. Kaedah ini akan menguji pelanggaran berdasarkan jarak objek-objek cembung yang diperolehi daripada maklumat topologi objek pada setiap kerangka masa. Dalam hal ini, perubahan untuk setiap kerangka masa perlu diskrit dan kecil agar ujian dapat dilakukan dengan berkesan. Keadaan ini dinamakan sebagai *geometric coherence* atau *continuity of motion*.

Untuk pengesanan melibatkan objek cekung, objek tersebut akan dibahagikan dan membentuk gabungan objek cembung. Pengesanan pelanggaran yang melibatkan hirarki objek rumit ini akan menggunakan pendekatan pepohon OBB. Pembinaan pepohon ini menggunakan kaedah atas ke bawah sementara ujian antara nod dalam hirarki menggunakan kaedah Teorem Paksi Pemisah [18]. Kaedah ini sebenarnya akan

[1] http://www.gamasutra.com
[2] http://www.flipcode.com
[3] http://www.gamedev.net

melibatkan pengiraan yang banyak dan kurang efisien untuk PB. Ditambah lagi dengan penggunaan pepohon OBB yang mengambil kos tinggi dalam ujian pelanggaran di setiap kerangka masa.

Sementara itu, Gino Van der Bergen [8] mengesan pelanggaran dalam PB menggunakan gabungan dua hirarki BV iaitu hirarki *Bounding Sphere* dan hirarki AABB. Untuk satu objek, dua hirarki BV akan dibina ke atasnya. Hirarki *Bounding Sphere* dibina semasa penjelajahan pepohon AABB dilakukan. Semasa proses penjelajahan itu juga, penjumlahan Minkowski untuk setiap nod AABB akan dikira. Hasil pengiraan ini kemudian akan digunakan untuk proses pembuangan objek yang berpotensi untuk tidak berlanggar. Ujian pelanggaran di antara dua objek akan dikira berdasarkan jarak menggunakan teknik Algoritma Gilbert-Johnson-Keerthi (GJK) [8]. Dalam hal ini, penulis berpandangan bahawa pembinaan dua hirarki dalam satu masa untuk satu objek jelas terlalu rumit dan memerlukan kos yang tinggi. Sekalipun langkah yang lain cuba diambil seperti melakukan pembinaan hirarki secara berjujukan, namun ujian secara perbandingan nilai *min* dan *max* lebih mudah berbanding ujian jarak yang digunakan.

6.0 Kesimpulan

Bidang pengesanan pelanggaran merupakan bidang menarik dan mencabar untuk terus dikaji. Kepelbagaian kaedah yang diketengahkan oleh penyelidik di bidang grafik berkomputer membuka kajian lebih luas untuk menghasilkan kaedah dan teknik yang bukan sahaja efisien namun bersifat universal. Dengan kata lain adalah diharapkan melalui kajian yang berterusan itu, kaedah dan teknik yang dihasilkan tidak terlalu bergantung kepada jenis model objek yang digunakan. Malahan apa yang lebih penting, kaedah dan teknik tersebut bersifat efisien dan efektif untuk pelbagai aplikasi persekitaran maya interaktif sama ada yang menekan elemen ketepatan tinggi mahupun kepantasan dalam mengesan pelanggaran.

Rujukan

[1] D. Daman, S.Kari, A. Bade. and N. Suaib. (2002). "An Overview of Collision Detection in Virtual Environment." 18Th International Conference on CAD/CAM, Robotics and Factories of The Future, CARS&FOF2002. Porto, Portugal.

[2] S.Gottschalk, M.C.Lin, and D.Manocha. (1996). "OBB-Tree: A Hierarchical Structure for Rapid Interference Detection." In Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH. New Orleans, Louisiana. pages 171-180.

[3] Klosowski, J.T, M.Held, J.S.B. Mitchell, H.Sowizral. and K.Zikan. (1998), "Efficient Collision Detection Using Bounding Volume Hierarchies Of k-Dops." IEEE Transaction on Visualization and Computer Graphics. Volume. 4, no.1.

[4] Volino, P. and Nadia Magnenat-Thalmann. (2000). "Virtual Clothing: Theory and Practice." Springer Verlag, New York.

[5] Taosong He. (1999). "Fast Collision Detection Using QuOSPO Trees." ACM Symposium on Interactive 3D Graphics. Atlanta GA, USA.

[6] P. Jimenez, F.Thomas. and C.Torras (2000). "3D Collision Detection: A Survey." Institut de Robotica i Information Industry, Barcelona, Spain.

[7] P.M. Hubbard. (1995). "Collision Detection For Interactive Graphics Application." IEEE Transactions on Visualization and Computer Graphics. 1(3):218-230.

[8] Van der Bergen.G. (2001). "Proximity Queries and Penetration Depth Computation on 3D Game Objects." In Proc.Game Developers Conference 2001. San Jose, California, USA. pages 821-837.

[9] M. Held, Klosowski. J.T. and J. S. B. Mitchell. (1995). "Evaluation of Collision Detection Methods for Virtual Reality Fly-Through." in proceedings 7th Canadian Conference on Computational Geometry. Qu'ebec City, Qu'ebec, Canada. pages.205-210.

[10] Klosowski, J.T. (1998). "Efficient Collision Detection For Interactive 3D Graphics And Virtual Environment." State University of New York, Stony Brook: Ph.D Thesis.

[11] T.Moller. and Eric Haines. (1999). "Real-Time Rendering." A K Peters Ltd, Natick, Massachusetts.

[12] Van der Bergen.G. (1998). "Efficient Collision Detection Of Complex Deformable Using AABB Trees." Journal of Graphics Tools 2,4. pages1-13.

[13] Alan Watt. (2000). "3D Computer Graphics." Addison-Wesley Publishing Company Inc 3rd Edition.

[14] Stephen Cameron. (1985). "A Study Of The Clash-Detection Problem In Robotics." In IEEE International Conference on Robotics and Automation, IEEE Press. pages 488-493.

[15] Brian Mirtich. (1997). "Efficient Algorithms for Two-Phase Collision Detection." Technical Report TR-97-23. Mitsubishi Electrical Research Laboratory.

[16] S. Suri, P. M. Hubbard. and J. F. Hughes. (1998). "Collision Detection in Aspect and Scale Bounded Polyhedra.", Proc. of 9th Annual Symposium on Discrete Algorithms.

[17] Fabio Ganovelli, John Dingliana. and Carol O'Sullivan. (2000). " BucketTree : Improving Collision Detection Between Deformable Objects." SCCG2000 Spring Conference on Computer Graphics, Budmerice Castle, Bratislava.

[18] Ming C.Lin. (2000). "Fast Proximity Queries for Large Game Environments." In Proc.Game Developers Conference 2000. San Jose, California, USA.