VISUALISASI RUANGAN TIGA DIMENSI PADA PERANGKAT BERGERAK BERBASIS ANDROID (DENGAN PEMUTAR MUSIK SURROUND DARI PHILIPS COMPANY)

Durio Etgar[*)], Wahyul Amien Syafei [*)], Jack Zijlmans[*)], and Zhaorui Yuan[*)]

Jurusan Teknik Elektro, Universitas Diponegoro Semarang
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia
Fontys University of Applied Sciences
Rachelsmolen 1, 5612 MA Eindhoven, The Netherlands.
[#4]Philips Research

[*)]*E-mail: {darno90@gmail.com, wasyafei@yahoo.com, j.zijlmans@fontys.nl, zhaorui.yuan@philips.com*

Abstrak

Tujuan utama proyek ini adalah menciptakan sebuah aplikasi demo yang praktis tentang visualisasi ruangan 3D yang dikombinasikan dengan efek suara Surround Sound. Lebih jauh lagi, visualisasi ini memberikan pengalaman berada di dalam ruangan Surround Sound yang terasa nyata tanpa harus terlibat secara fisik. Aplikasi yang bernama Virtual Listen Room ini adalah inovasi baru yang masih akan dikembangkan lebih lanjut sebagai media promosi yang praktis. Aplikasi ini dikembangkan dalam lingkup perangkat Android. Android dipilih karena ruang kembang yang luas dan komponen yang esensial untuk aplikasi ini, termasuk unit prosesor grafis (GPU). Manipulasi grafis dapat dilakukan dengan sebuah antarmuka bernama OpenGL ES yang tertanam di hampir semua perangkat Android. Selain itu, Android juga memiliki sensor Accelerometer yang berguna untuk menciptakan pergerakan kamera yang dinamis. Efek suara Surround sendiri berasal dari aplikasi pemutar musik Philips bernama MPEG Surround Sound Decoder. Untuk menyimpulkan proyek ini, didapatkan sebuah aplikasi dengan visualisasi ruangan 3 dimensi yang terkombinasi dengan pemutar musik suara Surround dari Philips. User dapat mengganti setting ruangan antara lain lokasi subwoofer, lampu ruangan, dan jumlah speaker. Aplikasi ini sendiri memiliki beberapa problem dalam performa yang pada akhirnya bisa diantisipasi.

Kata Kunci:Android,Visualisasi,Open GL ES,3D,Surround,Sensor.

Abstract

This project's specifically purposed as a demo application, so anyone can get the experience of a surround audio room without having to physically involved to it, with a main idea of generating a 3D surround sound room scenery coupled with surround sound in a handier package, namely, a "Virtual Listen Room". Virtual Listen Room set a foundation of an innovative visualization that later will be developed and released as one of way of portable advertisement. This application was built inside of Android environment. Android device had been chosen as the implementation target, since it leaves massive development spaces and mostly contains essential components needed on this project, including graphic processor unit (GPU). Graphic manipulation can be done using an embedded programming interface called OpenGL ES, which is planted in all Android devices generally. Further, Android has a Accelerometer Sensor that is needed to be coupled with scene to produce a dynamic movement of the camera. Surround sound effect can be reached with a decoder from Phillips called MPEG Surround Sound Decoder. To sum the whole project, we got an application with sensor-dynamic 3D room visualization coupled with Philips' Surround Sound Music Player. We can manipulate several room's properties; Subwoofer location, Room light, and how many speakers inside it, the application itself works well despite facing several performance problems before, later to be solved.

Keywords: Android,Visualization,Open GL ES,3D,Surround,Sensor.

1. Pendahuluan

Pernahkah Anda menonton film aksi atau perang di bioskop? Anda bisa merasakan suara peluru meluncur di dalam kepala Anda, suara ledakan di sekitar dan suara

helikopter di atas Anda. Hal ini dapat dicapai melalui teknologi suara *Surround*. Teknik ini meningkatkan persepsi spasialisasi suara dengan memanfaatkan lokalisasi suara dan kemampuan pendengar untuk mengidentifikasi lokasi atau asal suara terdeteksi dalam arah dan jarak. Konsep ini dapat direalisasikam dengan menggunakan beberapa saluran audio terpisah yang diarahkan ke pengeras suara [1] Sederhananya, suara *surround* membiarkan Anda mendengar suara campuran dari sumber yang berbeda dan arah untuk membuat Anda seolah-olah berada di tengah-tengah adegan. Ini adalah jenis pengaturan suara yang telah digunakan secara luas. Itulah mengapa surround sound decoder dipilih sebagai pemutar musik.

Ide utama dari proyek ini adalah menghasilkan visualisasi ruangan 3 dimensi ditambah dengan suara surround dalam perangkat yang ringkas, dengan nama "*Virtual Listen Room*". Aplikasi Ini khusus bertujuan sebagai demo, sehingga siapapun bisa mendapatkan pengalaman *surround audio* tanpa harus secara fisik terlibat didalamnya. *Virtual Listen Room* menetapkan dasar dari sebuah inovasi yang nantinya akan dikembangkan dan dirilis sebagai salah satu media iklan portabel. Aplikasi ini, bahkan idenya, belum pernah ditemukan sebelumnya. Perangkat Android telah dipilih sebagai target pelaksanaan karena memiliki ruang kembang yang besar dan mengandung komponen penting yang dibutuhkan pada proyek ini, termasuk unit prosesor grafis (GPU). Manipulasi grafis dapat dilakukan dengan menggunakan antarmuka pemrograman tertanam yang disebut OpenGL ES. Antarmuka ini terdapat di semua perangkat Android pada umumnya. Sebagai contoh, *game* 3D pada dasarnya diciptakan dengan menggunakan OpenGL ES. Android menggunakan lingkungan Linux dan Java sebagai bahasa pemrograman.

Masalahnya adalah hampir semua jenis perangkat android hanya memiliki maksimal 2 speaker. Ya, dan untuk menghadapinya, *Philips Research - Information and Cognition - Applied Sensor Technologies* memiliki *Virtual surround decoder* yang disebut "*MPEG Decoder* ^{TM}" yang telah dikembangkan selama berbulan-bulan. Virtual surround sistem audiomencoba untuk menciptakan persepsi bahwa seakan-akan ada banyak sumber suara. Untuk mencapai tujuan itu, perlu dirancang beberapa cara menipu sistem pendengaran manusia ke dalam pikiran bahwa suara datang dari suatu tempat yang sebenarnya tidak menghasilkannya [1] Kemudian aplikasi ini digabungkan dengan visualisasi 3D dan saat mendengarkan musik kita dapat memiringkan tampilan kamera.

Android grafis dan pengolahan suara nyaris tak tersentuh oleh para pengembang, sehingga aplikasi ini bisa menarik bagi siapa pun. Juga dengan fakta bahwa proyek ini adalah penemuan murni. Tidak ada seorang pun yang telah melakukan ini sebelumnya pada perangkat bergerak. Kesimpulan dari keseluruhan proyek, kami mendapat sebuah aplikasi dengan ruangan visualisasi 3D ditambah dengan *Philips Surround Sound Music Player* dan kamera yang sensor-dinamis. Kita dapat memanipulasi beberapa pengaturan dari ruangan tersebut; lokasi subwoofer, lampu ruangan, dan jumlah speaker di dalamnya.

2. Landasan Teori

2.1 Android

Android adalah sistem operasi berbasis Linux untuk perangkat mobile seperti smartphone dan komputer tablet. OS ini dikembangkan oleh Open Handset Alliance, yang dipimpin oleh Google.Android memiliki komunitas *developer* yang besar dan aplikasi yang memperluas fungsionalitas dari perangkat. Aplikasi untuk Android sendiri dikembangkan dengan bahasa pemrograman Java. Pada bulan Juni 2012, ada lebih dari 600.000 aplikasi yang tersedia untuk Android dan perkiraan jumlah aplikasi yang telah diunduh dari Google Play adalah 20 miliar.

Android menjadi perangkat bergerak terkemuka di dunia pada akhir tahun 2010. Analis menunjukkan kelebihan Android adalah multi-kanal OS. Selain itu, Sistem Operasi Android dapat dengan mudah ditanam ke berbagai jenis perangkat. Kita bisa membuktikannya pada Gambar 1 di bawah ini

.

Gambar 1. Perangkat Bergerak Android

Kiri ke kanan: Verizon, HTC, Samsung, dan Google's Galaxy Nexus. Kita bisa menarik kesimpulan bahwa Android dapat diterapkan secara luas dan banyak menganggapnya sebagai keuntungan besar.

2.2 Surround Sound

Surround sound adalah teknik untuk memperkaya kualitas reproduksi suara dari sumber audio dengan kanal audio tambahan dari speaker yang mengelilingi pendengar. Teknik ini meningkatkan persepsi spasialisasi suara dengan memanfaatkan lokalisasi suara dan kemampuan pendengar untuk mengidentifikasi lokasi atau asal suara terdeteksi dalam arah dan jarak. Biasanya teknik ini

dicapai dengan menggunakan beberapa saluran audio terpisah yang diarahkan ke pengeras suara. Sederhananya, suara *surround* memberi Anda suara campuran dari sumber dan arah yang berbeda untuk membuat Anda seolah-olah Anda berada di tengah-tengah adegan. Simulasi ini didasarkan pada penataan ruang *5.1 Surround*. Gambar di bawah ini (Gambar 2) adalah representasi baku dari pengaturan 5.1 *surround sound* dengan 6 sumber suara yang datang dari sudut yang berbeda.

Gambar 2. Simulasi Surround Sound

2.3 Open GL ES

OpenGL for Embedded Systems (OpenGL ES) adalah bagian dari intermuka pemrograman OpenGL yang dirancang untuk *embedded system* seperti ponsel, PDA, dan konsol permainan video.

- OpenGL ES untuk Android: versi 1.0, 1.1 dan 2.0
- Kemampuan untuk membuat objek 2D dan 3D
- Limitasi fungsi dari OpenGL karena keterbatasan hardware

Kapabilitas 3D OpenGL ES sering dimaksimalkan untuk permainan dan visualisasi. Misalnya *game* balap pada Gambar 3 memaksimalkan mesin OpenGL ES 3D.

Gambar 3. Contoh Penerapan OpenGL ES 3D

Di sisi lain kita juga mampu menciptakan grafis 2D dengan OpenGL ES, umumnya untuk mengembangkan game atau antarmuka pengguna. Gambar 4 merupakan contoh objek segitiga sederhana dalam 2 dimensi.

Gambar 4. Contoh OpenGL ES 2D

2.3.1 Min3D

Min3D adalah *library* ringan untuk Android menggunakan Java dan OpenGL ES yang kompatibel dengan Android v1.5/OpenGL ES 1.0 dan seterusnya. *Library* ini terkait sangat erat dengan OpenGL ES API, yang membuatnya ideal untuk memperoleh pemahaman tentang OpenGL ES API sambil memberikan kenyamanan dalam mengolah dan memanipulasi grafis. Dengan bantuan dari kumpulan *Class* ini, sangat mudah bagi pengembang untuk membuat berbagai benda primitif seperti kotak, bola, piramida dll
Min3D dapat mengimpor 3 tipe file yang berbeda:

• Wavefront OBJ
• 3DS
• MD2

2.4 Accelerometer

Accelerometer adalah perangkat yang mengukur berat per unit dari massa. Android menggunakan accelerometer sebagai sensor yang mendeteksi bagaimana Anda memiringkan ponsel Anda. Sensor ini digunakan di hampir semua perangkat Android saat ini.

Kita dapat mengambil keuntungan dari Kelas Sensor.TYPE_ACCELEROMETER pada Android untuk memanfaatkan nilai-nilai accelerometer. Berikut adalah penjelasan fisika tentang bagaimana kita bisa mendapatkan nilai accelerometer:

Semua nilai dalam satuan SI ($\frac{m}{s^2}$)

nilai[0]: Akselerasi minus Gx pada sumbu x
nilai[1]: Akselerasi minus Gy pada sumbu y
nilai[2]: Akselerasi minus Gz pada sumbu z

Sensor jenis ini mengukur percepatan yang diterapkan pada perangkat. Secara konseptual, hal tersebut dilakukan dengan mengukur kekuatan diterapkan pada sensor itu sendiri menggunakan persamaan:

$$Ad_v = -\sum \frac{Fs}{mass} \quad (1)$$

Ad$_v$ adalah nilai percepatan yang diterapkan pada perangkat di dalam ruang hampa, sementara **Fs** adalah energi diterapkan pada sensor itu sendiri. **mass** adalah massa perangkat.

Gaya gravitasi selalu mempengaruhi percepatan yang diukur, sebagaimana dinyatakan dalam rumus berikut:

$$Ad_e = -g - Ad_v \quad (2)$$

Ad$_e$ adalah nilai percepatan yang diterapkan untuk perangkat dengan pengaruh dari gravitasi bumi, sedangkan **g** adalah konstanta gaya gravitasi bumi (9.81 $\frac{m}{s^2}$)
Dengan persamaan sebelumnya, bila perangkat duduk di meja (dan jelas tidak mendapat percepatan), accelerometer membaca besarnya g = 9,81 m/s2

Demikian pula bila perangkat berada dalam posisi terjun bebas. Percepatan menuju ke tanah adalah 9,81 m/s2, sehingga accelerometer membaca besarnya g= 0 m/s2.

Untuk mengukur percepatan perangkat, kontribusi dari gaya gravitasi harus dihilangkan. Hal ini dapat dicapai dengan menerapkan filter *high-pass*. Sebaliknya, *low-pass* filter dapat digunakan untuk mengisolasi gaya gravitasi.

2.5 Pengaturan Ruang Audio yang Tepat

Pengaturan ruang audio yang direkomendasikan untuk 5.1 *surround room system*:

1. Pengeras suara tengah harus ditempatkan tepat di atas atau di bawah tengah layar TV. Coba untuk tidak menempatkannya terlalu jauh dari layar atau suara akan terasa tidak menyatu dengan gambar dari TV. Ini akan terdengar tidak alami dan merusak efek dari suara. Selain itu, Anda dapat menggunakan semua jenis pengeras suara sebagai pusat.

Gambar 5. Penempatan Pengeras Suara Depan yang Tepat

2. *Subwoofer* memiliki pekerjaan yang sangat spesifik, yaitu untuk mereproduksi suara bass dengan frekuensi yang sangat rendah. Oleh karena itu, penempatan subwoofer di kamar jauh lebih penting dibandingkan dengan speaker lainnya. Anda dapat meletakkannya di mana saja selama memiliki ruang yang cukup.

Gambar6. Penempatan *Subwoofer* yang Tepat

3. Membangun Aplikasi

Aplikasi ini dibangun dengan menggunakan Eclipse IDE, *compiler* Java yang langsung direkomendasikan oleh Google sendiri untuk setiap pengembang Android. Untuk memaksimalkan kapasitas OpenGL ES digunakan *library* Min3D.

3.1 Pengaturan Min3D

1. Download dan ekstrak *file library* Min3D dalam folder proyek Anda.
2. Buka proyek Anda pada Eclipse
3. *Refresh workspace* Anda. Anda akan menemukan folder bernama min3d
4. Klik kanan pada folder, pilih *Build path*. Gunakan sebagai folder sumber.
5. Sekarang Anda dapat memanggil *method* dari *library* min3d

3.2 Membangun Skybox

Pertama kita harus *declareSkybox* dengan parameter (float ukuran,int kualitas). Setelah itu kita dapat menambahkan tekstur pada setiap sisi *Skybox* untuk mendapatkan impresi ruang yang nyata. *Method* Scale() menentukan proporsi setiap sumbu. Kemudian kita dapat melampirkan tekstur apapun pada setiap sisi ruangan dengan proporsi 1:1 untuk lebar danpanjang. Gambar 7 di bawah ini adalah *skybox* lengkap dengan tekstur dinding dan lantai sehingga kita bisa mendapatkan kesan sebuah ruangan.

Gambar 7. *Skybox*

3.3 Menciptakan Subwoofer

Objek primitif (Kubus, Silinder, Balok, Bola, Kerucut) juga dapat diciptakan menggunakan min3d. Cukup inisiasi objek primitif yang kita inginkan dan tetapkan beberapa parameter. Misalnya, dalam hal ini kita perlu 6 kubus 3D untuk kemudian dijadikan sebagai *Subwoofer* dan *speaker*. Kita harus juga melampirkan tekstur ke objek 3D polos untuk memberikan kesan nyata.

Gambar8. Tekstur *Speaker*

Gambar 8 di atas adalah tekstur *speaker* kayu yang kemudian akan digunakan sebagai *subwoofer*. Skala lebar dan panjang harus 1:1 untuk membuatnya benar-benar sesuai dengan objek kubus 3D.
\

3.3. Menambahkan Objek Jadi

Ada fitur di Min3D yang terlalu bagus untuk dilewatkan. Kita dapat menambahkan objek 3D buatan kita sendiri ke layar. Hal ini dimungkinkan untuk hampir semua model 3D dengan ekstensi . OBJ,. MD2 dan .3DS.

Anda dapat menemukan banyak objek 3D yang relevan di 3divia.com. Lebih baik menggunakan .3 DS objek karena kemudahannya untuk ditambahkan. Sebagai contoh, kita menemukan set audio yang sudah jadi ini (Gambar 9) dengan 3 *speaker*, nantinya akan digunakan sebagai *speaker* pusat, kiri depan, dan kanan depan.

Gambar9. Contoh Objek Jadi 3D

3.4 *Speaker* Berdiri untuk SisiBelakang

Speaker berdiri dapat dibuat berdasarkan bentuk dasar kubus yang dimanipulasi skalanya. Oleh karena itu, kami memutuskan untuk mengubah kubus menjadi bentuk balok. Tekstur juga harus disesuaikan, sehingga perlu dimanipulasi menggunakan *Photoshop*.

3.5 Eksploitasi Kamera

Min3D juga menyediakan kemudahan dalam manajemen kamera. Dengan integrasi sensor kita dapat memiringkan kamera dalam ruangan bersama dengan gerakan perangkat. Hal pertama yang dilakukan adalah kita perlu *Override built-in method* onSensorChanged (), kemudian *pass* nilai event ke objek *Camera*.

Perhatikan bahwa kita hanya menggunakan y dan sumbu z saja dan pengaturan orientasi layar lansekap untuk mendapatkan pemandangan yang lebih luas.

3.6 Pengaturan Ruang

Tatanan ruang dibuat sesuai dengan poin **2.5** (**Pengaturan Ruang Audio yang Tepat**). Di sisi depan ruangan terdapat layar LCD, 2 gambar dinding sebagai hiasan, dan set audio sebagai *speaker* pusat, depan kiri, dan kanan depan dengan subwoofer seperti yang dapat kita lihat pada Gambar 10.

Gambar10. Tatanan *Speaker* Depan dan*Subwoofer*

Untuk menyelesaikan keseluruhan visualisasi, seperti yang digambarkan dalam Gambar 11, 2 speaker berdiri ditambahkan. Keduanya bertindak sebagai sumber suara belakang kiri dan belakang kanan.

Gambar11. Tatanan *speaker* belakang

3.7 Pemutar Musik Philips dengan Dekoder *Virtual Surround Sound*

Aplikasi Ini adalah jenis media player yang mensimulasikan efek *surround* pada file media tertentu (mp4 dan. Wav.). Beberapa fitur yang dimiliki antara

lain: penelusur file sederhana, penyaringan wav dan mp4 file. Aplikasi *MPEG Surround* ini memiliki 2 tombol yang dapat digunakan untuk mengulangi lagu (di pojok kanan atas) dan mengaktifkan/nonaktifkan efek surround (di sisi bawah). Untuk menghasilkan output suara yang unik dan asli, aplikasi ini menggunakan decoder tertentu yang dibangun dengan bahasa C / C + +. Di bawah ini (Gambar 12) adalah antarmuka Pemutar Musik Philips.

Gambar12. Dekoder Philips' MPEG Surround

4. Pengujian, Analisis dan Pemecahan Masalah

4.1 Problem Assessment

Ada beberapa masalah non-fungsional yang terlihat setelah tahap implementasi:

1. Masalah kompatibilitas. Aplikasi tidak akan berjalan jika kita menginstalnya di versi Android lain. Sebuah aplikasi yang bergantung pada JNI (Java Native Interface) kadang-kadang kehilangan portabilitas yang dimiliki oleh Java.
2. Masalah performa. Musik kadang-kadang tersendat, tampaknya dipengaruhi oleh beban yang cukup besar dari visualisasi 3D dan Accelerometer.

Masalah pertama tampaknya tak terpecahkan. Pengembang sudah mencoba untuk meng*install* paksa aplikasi di beberapa versi Android lain, tapi hal itu justru membuat situasi semakin. Aplikasi menjadi sulit untuk di *compile*, dan terkadang IDE menemukan kesalahan tak dikenal tanpa penjelasan.

Untuk mendapatkan gambaran yang lebih tentang masalah kedua, pengembang melakukan dua analisis; *Profiling* dan Analisis Memori. *Profiling* mencakup pemakaian memori dari aplikasi itu sendiri, sedangkan analisis memori memiliki lingkup yang lebih luas karena juga melibatkan lingkungan Android.

4.2 Profiling

Analisis ini diambil menggunakan *DDMS Traceview* di Eclipse. Pengembang berhasil membuat 5 profil yang berbeda berdasarkan pada keberadaan benda-benda tambahan, tekstur objek, Fitur MPEG Surround, dan juga fungsi *wakelock*. Semua profil ini menggunakan *Game Sensor Delay* yang sama.

4.2.1 Hasil Profiling

Profiling menggunakan *DDMS Traceview* cukup baik untuk melacak berapa banyak sumber daya yang dibutuhkan oleh *method* tertentu. Namun, hasilnya kadang-kadang bias, bahkan dalam kondisi/kriteria yang sama. Suara bekerja sempurna pada suatu saat dan disaat lain membuka aplikasi pengembang mendapat kinerja yang berbeda. Jumlah RAM juga mempengaruhi stabilitas aplikasi. Meskipun hasilnya tidak mutlak, *DDMS Traceview* cukup solid untuk mendapatkan representasi penggunaan CPU.

4.3 Analisis Memori

Analisis ini diambil dengan menggunakan *Memory Analyzer* pada Eclipse. Pengembang berhasil membuat 4 analisis yang berbeda berdasarkan pada keberadaan benda-benda tambahan, tekstur objek , kualitas tekstur dan Fitur MPEG Surround. *Game Sensor Delay* digunakan untuk setiap kondisi.

4.3.1 Hasil Analisis Memori

1. Hasil untuk setiap parameter hampir sama, kecuali ada dua yang memiliki komponen lebih besar dari 1 persen dari *heap* total.
2. Dua parameter yang memiliki masalah dengan pembuangan memori mengandung objek jadi 3D (Home Theater set).
3. Setelah mengurangi resolusi tekstur, alokasi memori dari android.content.res.Resources juga sedikit menurun.
4. *Class* Sistem dari Android API mendominasi persentase *heap.*
5. Jika objek jadi 3D ditambahkan ke layar, *Class* dengan konsumsi memori tertinggi adalah min3D Object3Dcontainer, diikuti oleh StringBlock, android.widget.listView, AUDIOCONTROL PlaySoundTask, dan kemudian SkyBox.
6. Tanpa objek jadi 3D, *Class* dengan konsumsi tertinggi adalah StringBlock, android.widget.listView, AUDIOCONTROL PlaySoundTask, dan kemudian SkyBox.
7. Sayangnya, performa suara masih belum stabil. Walau begitu*lag* pada suara tampaknya sedikit menghilang dengan berkurangnya resolusi tekstur 'dan tak adanya obyek tambahan.
8. Karena ini adalah analisis memori, aktivitas lain di luar aplikasi yang ikut berpengaruh juga akan ditampilkan. Dengan*Class* Sistem yang begitu banyak mengkonsumsi memori, masalah manajemen memori yang buruk pada OS Froyo semakin dipertajam.

4.4 Pemecahan Masalah

Masalah yang ada sebagian besar disebabkan oleh kebocoran memori yang sangat mempengaruhi kinerja aplikasi. Dari analisis memori kita juga mendapatkan jumlah memori yang dialokasikan ke dalam aplikasi. Karena aplikasi hanya mengkonsumsi RAM sebesar 1,9 MB, seharusnya tidak menjadi masalah karena di dalam perangkat uji tersedia 512 MB RAM. Memang, total RAM terbagi dengan unit prosesor grafis sekitar 128 MB tapi dengan sisa 384 MB seharusnya aplikasi berjalan lancar.

Berdasarkan pada kedua kegiatan analisis sebelumnya kita menyiapkan 2 solusi yang bisa diimplementasikan:

1. Pada analisis profil,pengembang menemukan bahwa *Class*OpenGLselalu menjadi konsumen terbesar CPU. Ada dua pilihan untuk meminimalkan konsumsi CPU: mengurangi jumlah objek 3D yang dimuat dan mengurangi kualitas tekstur. Pilihan pertama keluar dari pertimbangan karena semua objek 3D mutlak dibutuhkan. Pilihan kedua menjadilebih masuk akal, karena mengurangi kualitas tekstur tidak akan banyak mempengaruhi tampilan secara keseluruhan.
2. Android memiliki banyak pilihan aplikasi "manajemen memori" yang akan berguna untuk membebaskan sejumlah RAM sehingga akan memberikan lebih banyak ruang untuk aplikasi yang berat.

4.4.1 Penindak lanjutan Profiling : Mengurangi Kualitas Tekstur

Akan sangat baik untuk memiliki tekstur berdefinisi tinggi pada visualisasi 3D, tapi kemudian pengembang harus berpikir dua kali jika itu mengorbankan kinerja secara keseluruhan. Kualitas tekstur benda dapat dikurangi sedikit untuk meningkatkan performa suara. Tekstur Home theaterset terpasang secara *default* ke objek, jadi tidak bisadimanipulasi. Pengembang menggunakan aplikasi Paint untuk mengubah ukuran tekstur menjadi 50% dari ukuran sebenarnya dengan masih mempertahankan aspek rasio.

Gambar13. Perbandingan Kualitas Tekstur

Visualisasi 3D setelah penurunan kualitas pasti menjadi lebih buruk dari sebelumnya. Kita dapat memeriksa Gambar 13 dan kemudian Gambar 14 dimana bingkai yang tergantung di dinding menjadi buram. Juga warna dinding menjadi pucat.

Gambar14. Perbandingan Kualitas Tekstur

Ini bukan masalah besar karena setidaknya kualitas suara meningkat secara signifikan. *Lag* pada suara juga sedikit berkurang. Waktu start-up aplikasi menjadi lebih cepat, dari sekitar 5 detik sampai kurang dari 3 detik.

4.4.2 Penindaklanjutan Analisis Memori : *Memory Booster*

Android memiliki banyak aplikasi *task killer* di *market* aplikasi mereka. *Task killer* digunakan untuk membebaskan ruang memori dengan membunuh aplikasi *background*, sehingga akan menciptakan ruang untuk aplikasi lain yang akan dijalankan. Aplikasi sederhana seperti jam atau pesan tentu saja tidak akan terganggu oleh jumlah RAM yang tersisa, tapi visualisasi 3D + Dekoder suara + Accelerometer Sensor tentu bisa menyebabkan terjadinya penurunan performa.

Aplikasi *Memory Booster Lite* telah diuji oleh developer, apakah bisa diandalkan untuk membebaskan RAM atau tidak. Aplikasi ini memiliki 2 fungsi utama: memantau bagaimana keadaan memori dan juga membunuh aplikasi tertentu untuk melepaskan memori.

Gambar15. *Memory Booster Lite*

Gambar 15 menunjukkan setiap kolom dalam aplikasi *Memory Booster Lite*. Pemantau memori memberikan perhitungan rinci memori bebas bersama dengan memori yang digunakan. Pembunuh *Task* pada tab kedua, memberi Anda daftar aplikasi yang berjalan. Daftar ini diurutkan berdasarkan jumlah memori yang dialokasikan untuk aplikasi yang relevan. Seperti yang bisa kita lihat Google Maps menyerap hampir 10 MB memori, dan selama kita tidak menggunakannya dalam aplikasi, kita bisa membunuhnya. Metode ini memberikan peningkatan penting pada kualitas suara, menyapubersih semua *lag* pada suara. Kolom Log menampilkan daftar kegiatan apa

saja yang telah dilakukan sejak pertama kali kita menginstal aplikasi ini.

5. Penutup dan Kesimpulan

Secara umum gagasan utama dari proyek ini berhasil dicapai, yaitu menciptakan visualisasi ruang 3D di dalam lingkungan Android. Android memiliki mesin grafis OpenGL ES dibantu dengan min3D untuk mewujudkan visualisasi ruang tanpa harus menggambar garis demi garis. Gerakan Sensor juga dapat terwujud berkat accelerometer Android. Posisi kamera dapat dipindahkan tergantung pada bagaimana Anda memiringkan perangkat. Konsep gravitasi accelerometer juga memungkinkan untuk melakukan perbesaran jauh-dekat. Pengaturan ruang 5.1 *surround sound* juga dicuplik dari internet untuk mendapatkan kesan nyata dari ruang Audio disertai dengan pemutar suara sinematik.

Surround Sound "MPEG Decoder ™" adalah aplikasi pemutar suara surround Android yang disediakan oleh Philips. Meskipun menggunakan dekoder bahasa pemrograman tingkat rendah, Aplikasi ini dapat dikombinasikan dengan visualisasi 3D. Tapi di sinilah masalah muncul. *Library* min3Dmerusak kompatibilitas yang ada pada Java, sehingga aplikasi tidak berfungsi atau bahkan rusak ketika mencoba untuk dijalankan pada sistem operasi lain. Masalah kompatibilitas diikuti oleh masalah performa dimana *lag* mengganggu pemutaran suara. Muatan berat yang diproduksi oleh grafis 3D + dekoder + kombinasi accelerometer tampaknya cukup untuk menghasilkan masalah .

Sayangnya, masalah kompatibilitas tampaknya tidak mungkin untuk dipecahkan karena bersumber dari lingkungan Android itu sendiri. Dari kasus ini kita bisa belajar sesuatu bahwa jika kita ingin mengembangkan aplikasi dengan kombinasi bahasa pemrograman tingkat rendah kita harus memutuskan apa sistem operasi yang akan dipakai seterusnya.

masalah kinerja benar-benar tampak setelah beberapa pengujian dan analisis. Visualisasi 3D memang mengkonsumsi sebagian besar kemampuan telepon dan sumber daya. Solusi mengurangi kualitas adegan untuk meningkatkan kinerja dapat mengatasi masalah terssebur, sehingga kemudian dapat disimpulkan bahwa aplikasi ini telah memenuhi semua ekspektasi terlepas dari masalah kompatibilitas. Untuk menyelesaikan aplikasi ini, tahap pengembangan yang lebih lanjut akan mencoba untuk menerapkan efek suara yang berbeda untuk posisi kepala yang berbeda.

Secara umum, dapat disimpulkan bahwa aplikasi ini bekerja seperti yang diharapkan. Pengguna bisa mendapatkan gagasan “praktis”-nya, terutama karena aplikasi ini diimplementasikan di perangkat bergerak Android. Pengaturan ruang ditambahkan sebagai fitur ekstra seperti manipulasi lokasi *Subwoofer*, lampu ruangan, dan jumlah *speaker*. Bekerja dengan Android adalah sebuah pengalaman yang menghibur dan kami merekomendasikan pengembangan perangkat bergerak ini untuk semua pengembang perangkat lunak dimana saja.Secara umum gagasan utama dari proyek ini berhasil dicapai, yaitu menciptakan visualisasi ruang 3D di dalam lingkungan Android. Android memiliki mesin grafis OpenGL ES dibantu dengan min3D untuk mewujudkan visualisasi ruang tanpa harus menggambar garis demi garis. Gerakan Sensor juga dapat terwujud berkat accelerometer Android. Posisi kamera dapat dipindahkan tergantung pada bagaimana Anda memiringkan perangkat. Konsep gravitasi accelerometer juga memungkinkan untuk melakukan perbesaran jauh-dekat. Pengaturan ruang 5.1 *surround sound* juga dicuplik dari internet untuk mendapatkan kesan nyata dari ruang Audio disertai dengan pemutar suara sinematik.

Surround Sound "MPEG Decoder ™" adalah aplikasi pemutar suara surround Android yang disediakan oleh Philips. Meskipun menggunakan dekoder bahasa pemrograman tingkat rendah, Aplikasi ini dapat dikombinasikan dengan visualisasi 3D. Tapi di sinilah masalah muncul. *Library* min3Dmerusak kompatibilitas yang ada pada Java, sehingga aplikasi tidak berfungsi atau bahkan rusak ketika mencoba untuk dijalankan pada sistem operasi lain. Masalah kompatibilitas diikuti oleh masalah performa dimana *lag* mengganggu pemutaran suara. Muatan berat yang diproduksi oleh grafis 3D + dekoder + kombinasi accelerometer tampaknya cukup untuk menghasilkan masalah .

Sayangnya, masalah kompatibilitas tampaknya tidak mungkin untuk dipecahkan karena bersumber dari lingkungan Android itu sendiri. Dari kasus ini kita bisa belajar sesuatu bahwa jika kita ingin mengembangkan aplikasi dengan kombinasi bahasa pemrograman tingkat rendah kita harus memutuskan apa sistem operasi yang akan dipakai seterusnya.

masalah kinerja benar-benar tampak setelah beberapa pengujian dan analisis. Visualisasi 3D memang mengkonsumsi sebagian besar kemampuan telepon dan sumber daya. Solusi mengurangi kualitas adegan untuk meningkatkan kinerja dapat mengatasi masalah terssebur, sehingga kemudian dapat disimpulkan bahwa aplikasi ini telah memenuhi semua ekspektasi terlepas dari masalah kompatibilitas. Untuk menyelesaikan aplikasi ini, tahap pengembangan yang lebih lanjut akan mencoba untuk menerapkan efek suara yang berbeda untuk posisi kepala yang berbeda.

Secara umum, dapat disimpulkan bahwa aplikasi ini bekerja seperti yang diharapkan. Pengguna bisa mendapatkan gagasan “praktis”-nya, terutama karena aplikasi ini diimplementasikan di perangkat bergerak Android. Pengaturan ruang ditambahkan sebagai fitur

ekstra seperti manipulasi lokasi *Subwoofer*, lampu ruangan, dan jumlah *speaker*. Bekerja dengan Android adalah sebuah pengalaman yang menghibur dan kami merekomendasikan pengembangan perangkat bergerak ini untuk semua pengembang perangkat lunak dimana saja.

Daftar Pustaka

[1]. (2012) Wikipedia. [Online]Available:http://wikipedia.org/
[2]. (2012) min3D Library. [Online] Available: http://code.google.com/p/min3d/
[3]. (2012) Android Developer. [Online] Available: http://developer.android.com/index.html
[4]. (2012) The Home Cinema Guide. [Online] Available: http://www.the-home-cinema-guide.com
[5]. (2012) min3D Setup. [Online] Available: .http://code.google.com/p/min3d/setup/
[6]. (2012) Rozengain Creative Technology Blog. [Online] Available: http://www.rozengain.com/blog/
[7] . MatD (2012) Load a 3D OBJ Model with min3D for Android. [Online] Available: http://www.mat-d.com/site/tutorial-load-a-3d-obj-model-with-min3d-for-android/