Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.2024118040
Vol. 11, No. 5, Oktober 2024, hlm. 1145-1150 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

ANALISIS KINERJA *AUGMENTED REALITY HYPERTEXT MARKUP LANGUAGE* DENGAN PEMANFAATAN *WEB GRAPHICS LIBRARY* DAN OPENGL *SHADING LANGUAGE* UNTUK PENGEMBANGAN 3D

Multohadi Hamzaturrazak[1], Abdul Jabbar[2], Rizal Setya Perdana[3], Aryo Pinandito[*4]

[1,2,3,4] Universitas Brawijaya, Malang
Email: [1]hadigone123@gmail.com, [2]abduljabbar@student.ub.ac.id, [3]rizalespe@ub.ac.id, [4]aryo@ub.ac.id
[*]Penulis Korespondensi

(Naskah masuk: 14 November 2023, diterima untuk diterbitkan: 30 Oktober 2024)

Abstrak

Di era digital ini, teknologi mengalami kemajuan pesat, termasuk penggunaan *augmented reality* yang semakin populer. Perpaduan elemen virtual dan dunia nyata oleh *augmented reality* menciptakan interaksi pengguna yang berkesan dan dinamis. Munculnya alat-alat ini mempermudah pengintegrasian *augmented reality* ke dalam aplikasi web. Dengan memanfaatkan mentahan *shader* mentah dari WebGL dan GLSL, tujuan penelitian ini adalah untuk menyelidiki dan membangun lingkungan *augmented reality* berbasis web melalui *library* Three.js. WebGL berfungsi sebagai standar otoritatif untuk *rendering* grafis 3D dalam antarmuka berbasis web, sedangkan GLSL berfungsi sebagai bahasa skrip yang bertanggung jawab untuk mengatur presentasi visual yang dihasilkan oleh WebGL. Three.js berfungsi sebagai perangkat JavaScript yang dirancang untuk membuat grafik 3D interaktif berbasis web. Dalam studi ini, eksplorasi *shader* mentah dari WebGL dan GLSL dilakukan untuk menganalisis pengalaman web *augmented reality*. *Shader* yang diimplementasikan digabungkan dengan Three.js dan AR.js untuk membangun atmosfer *augmented reality* yang menawan. Selanjutnya dilakukan perbandingan antara hasil penggunaan GLSL dan WebGL dalam konteks *augmented reality* di web. Analisis menunjukkan bahwa GLSL mengungguli WebGL dalam skenario *augmented reality* berbasis web. *Shader* GLSL memberikan kinerja yang lebih baik untuk menampilkan objek virtual di *augmented reality*.

Kata kunci: *Grafik 3D, augmented reality, grafik komputer, HTML5, tekstur mapping, WebGL*

PERFORMANCE ANALYSIS OF AUGMENTED REALITY HYPERTEXT MARKUP LANGUAGE THROUGH THE UTILIZATION OF THE WEB GRAPHICS LIBRARY AND OPENGL SHADING LANGUAGE FOR 3D DEVELOPMENT

Abstract

In this digital era, technology has changed rapidly advancements, including the usage of augmented reality, which has become increasingly popular. The fusion of virtual and real-world elements by augmented reality creates memorable and dynamic user interactions. The emergence of these tools has made it simpler to integrate augmented reality into web applications. By leveraging raw WebGL and GLSL shaders, we aim to investigate and establish web-based augmented reality environments via the Three.js library. WebGL serves as the authoritative standard for 3D graphical rendering within web-based interfaces, whereas GLSL functions as the scripting language responsible for regulating the visual presentation produced by WebGL. Three.js serves as a JavaScript toolset designed to create interactive, web-based 3D graphics. In this study, an exploration of raw WebGL and GLSL shaders is conducted to analyze the augmented reality web experience. The implemented shaders are combined with Three.js and AR.js tools to construct a captivating augmented reality atmosphere. Furthermore, a comparison is made between the results of using GLSL and WebGL in the context of augmented reality on the web. The analysis demonstrates that GLSL outperforms WebGL in web-based augmented reality scenarios. GLSL shaders provide better performance for displaying virtual objects in augmented reality.

Keywords: 3D graphics, augmented reality, computer graphics, HTML5, texture mapping, WebGL

1. PENDAHULUAN

Seiring dengan pesatnya perkembangan teknologi informasi dan komunikasi, dunia telah menyaksikan transformasi besar dalam cara manusia berinteraksi dengan informasi dan lingkungan sekitar (Tolle et al., 2015). Salah satu terobosan yang signifikan dalam dunia teknologi adalah pengenalan *Augmented*

Reality (AR) (Asadza-deh et al., 2021) dan kemajuan dalam pengembangan teknologi web seperti HyperText Markup Language (HTML). Kedua teknologi ini telah mengubah cara manusia berinteraksi dengan dunia nyata dan dunia maya (Parashar, 2020; Van Krevelen and Poelman, 2010).

Seiring dengan perkembangan AR dan HTML secara terpisah, muncullah kebutuhan untuk menggabungkan potensi keduanya. Inilah yang kemudian menghasilkan konsep ARHTML, sebuah istilah yang merujuk pada penggabungan teknologi AR dengan teknologi web HTML. Dengan ARHTML, halaman-halaman web dapat memanfaatkan teknologi AR untuk menciptakan pengalaman yang lebih interaktif dan inovatif bagi pengguna (Li et al., 2020).

Meskipun HTML memiliki banyak kelebihan, namun HTML memiliki keterbatasan dalam menghasilkan grafik tiga dimensi (3D) yang kompleks. Teknologi HTML itu sendiri tidak menyediakan fungsionalitas untuk menghasilkan 3D secara langsung (Schwarz et al., 2022). Meskipun demikian, efek visual 3D pada halaman web masih dapat dicapai dengan menggabungkannya dengan teknologi lain. Salah satu pendekatan umum adalah menggunakan *Web Graphics Library* (WebGL), yang menjadi standar web untuk menghasilkan grafik 3D di dalam browser dan merupakan solusi untuk meningkatkan kualitas dan kecepatan perenderan objek virtual dalam aplikasi AR berbasis HTML (Li et al., 2020; Miao et al., 2017).

Dengan memanfaatkan OpenGL Shading Language (GLSL) berbasis WebGL, metode perenderan ARHTML dapat mengatasi keterbatasan HTML dalam menghasilkan grafik 3D yang kompleks (Cholissodin et al., 2022). WebGL adalah Application Programming Interface (API) grafis 3D yang memungkinkan penggunaan *shader* untuk mengontrol proses *rendering* secara langsung pada Graphics Processing Unit (GPU) (Florez et al., 2021). Dengan *shader*, pengembang dapat menciptakan efek grafis yang lebih canggih dan beragam dibandingkan dengan metode *non-shader*. GLSL adalah metode perenderan yang mengimplementasikan *vertex shader* dan *fragment shader* kustom untuk meningkatkan fleksibilitas dan kontrol dalam proses *rendering*, serta menciptakan efek visual yang kompleks (He et al., 2017; Heinrich et al., 2019; Li et al., 2020).

Penelitian ini bertujuan untuk menguji performa memori dari *shader* mentah dari WebGL dan GLSL mentah pada aplikasi AR berbasis HTML. Eksperimen melibatkan beberapa skenario AR dengan tingkat kompleksitas dan interaksi yang berbeda. Kualitas visual WebGL dan GLSL akan dievaluasi dalam konteks AR. Selain itu, stabilitas dan responsivitas aplikasi AR berbasis HTML yang menggunakan WebGL dan GLSL akan dievaluasi.

Diharapkan penelitian ini dapat memberikan pemahaman lebih baik terhadap penggunaan WebGL dan GLSL dalam mengembangkan aplikasi AR berbasis HTML. Hasil penelitian ini diharapkan dapat memberikan panduan dan rekomendasi bagi para pengembang untuk memilih metode yang optimal guna meningkatkan kualitas dan performa aplikasi AR berbasis HTML di masa mendatang.

2. KAJIAN PUSTAKA

2.1 ARHTML

Dalam paper yang dibuat oleh Xiuquan (Qiao et al., 2019), paper tersebut membahas tentang teknologi dan implementasi terkini dari *Augmented Reality Mobile* (Mobile AR). Teknologi ini menggabungkan objek virtual dua dimensi atau tiga dimensi ke dalam sebuah lingkungan nyata dan memproyeksikan objek maya tersebut secara realtime melalui perangkat *mobile* seperti *smartphone* atau komputer tablet. Tidak hanya itu, studi ini juga membahas tantangan dan wawasan ketika AR bertemu dengan web, yaitu *platform* yang menyediakan layanan AR secara pervasif, ringan, dan lintas platform tanpa perlu mengunduh aplikasi khusus.

Selain itu, paper ini juga telah menyoroti potensi dari jaringan komunikasi yang terbarukan yaitu seluler 5G untuk meningkatkan efisiensi komunikasi dan komputasi ketika menggunakan AR yang berbasis di-web. Berbagai teknologi pendukung untuk web AR juga turut diulas, seperti *web standard*, *web framework*, *web browser*, *web server*, dan *web protocol*. Dengan demikian, studi ini dapat memberikan gambaran yang jelas tentang penggabungan teknologi AR dan HTML dalam pengembangan web AR, serta tantangan dan peluang yang dihadapi dalam menghadirkan pengalaman yang lebih baik bagi pengguna.

2.2 Rendering di Web

Dalam menghasilkan sebuah objek 3D di dalam AR yang berbasis HTML, terdapat beberapa tantangan diantaranya yaitu web browser tidak selalu memanfaatkan kekuatan GPU secara optimal. Untuk mengatasi masalah ini, ada beberapa penelitian sebelumnya yang telah dilakukan yang berfokus kepada kinerja *rendering* AR berbasis HTML. Penelitian ini memiliki kaitan dengan fokus penelitian dan bertujuan untuk memberikan informasi yang bermanfaat tentang perkembangan teknologi ARHTML.

Salah satu penelitian yang membahas kinerja *rendering* AR berbasis HTML yang disusun oleh Ferrao (Ferrão et al., 2022). Penelitian ini meneliti tentang kinerja *rendering* AR berbasis HTML dan menyarankan penggunaan teknologi *shader* berdasarkan WebGL untuk meningkatkan kinerja *rendering*. Dengan menggunakan teknologi *shader* ini, diharapkan kualitas *rendering* AR berbasis HTML dapat ditingkatkan, sehingga mampu memberikan pengalaman pengguna yang lebih baik.

Penelitian ini penting agar dapat memahami cara meningkatkan performa dan kualitas *rendering* dalam aplikasi AR berbasis web, yaitu dengan menggunakan teknologi *shader* dari WebGL, penelitian ini

dapat memberikan wawasan yang berharga bagi pengembangan teknologi AR berbasis HTML di masa mendatang.

2.3 WebGL dan GLSL

WebGL adalah teknologi yang dirancang khusus untuk menjalankan 3D di web *browser*, akan tetapi memiliki beberapa keterbatasan dibandingkan dengan OpenGL *for Embedded Systems* (OpenGL ES 2.0), yaitu, WebGL tidak memiliki akses ke fungsi GPU tertentu yang dapat digunakan secara *nativ*e, seperti *compute shader* atau *tessellation shader*.

Untuk mengatasi keterbatasan ini dan meningkatkan performa dalam proyek web, sebuah buku *online* yang dibuat oleh Patricio Gonzalez Vivo dan Jen Lowe (patrio gonzalez vivo and jen lowe, 2015) mengajarkan cara menggunakan *shader* dalam proyek web. *Shader* adalah program kecil yang ditulis dalam GLSL yang berjalan pada GPU dan digunakan untuk mengontrol warna, pencahayaan, tekstur, dan efek lain dari objek grafis.

Buku ini juga menjelaskan dasar-dasar pemrograman berbasis GPU, terutama dengan menggunakan arsitektur Compute Unified Device Architecture (CUDA) yang dikembangkan oleh NVIDIA. GLSL adalah bahasa pemrograman tingkat tinggi yang digunakan untuk mengontrol proses *rendering* grafik pada perangkat keras seperti GPU. Penggunaan GLSL memungkinkan para pengembang untuk membuat *shader* yang mengontrol bagaimana objek atau gambar ditampilkan di layar.

Dalam pengembangan aplikasi AR berbasis HTML, penggunaan GLSL dapat meningkatkan kinerja *rendering* dengan mengoptimalkan pemanfaatan sumber daya komputer. Metode ini memungkinkan para pengguna untuk menciptakan efek visual dan memberikan kontrol yang lebih besar terhadapa proses *rendering*. Dengan menerapkan penggunaan GLSL.

3. METODE PENELITIAN

Penelitian ini dimulai dengan mengidentifikasi masalah yang menjadi fokus penelitian. Setelah masalah diidentifikasi, langkah berikutnya adalah merancang desain eksperimen yang tepat untuk mengatasi masalah tersebut. Pada tahap eksperimen, berbagai kondisi skenario dengan tingkat kompleksitas dan interaksi yang berbeda akan diatur untuk memperoleh data yang relevan.

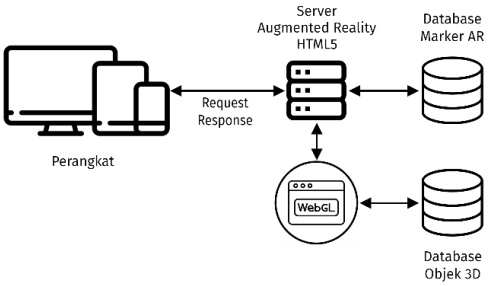
Setelah eksperimen selesai dilakukan, hasilnya akan diukur dan dicatat secara cermat. Data-data yang diperoleh dari hasil pengukuran akan dikumpulkan dan disusun dalam bentuk pengumpulan data yang terstruktur. Selanjutnya, data tersebut akan dianalisis.

Analisis data bertujuan untuk menggali informasi dan pola yang ada dalam data, serta menginterpretasikan hasil pengukuran secara mendalam. Dari hasil analisis, kesimpulan dan temuan akan diambil untuk menjawab pertanyaan penelitian dan mengatasi masalah yang diidentifikasi. Dengan demikian, alur penelitian ini mencakup tahap identifikasi masalah, desain eksperimen, peng-ukuran, pengumpulan data, analisis data, dan penarikan kesimpulan.

3.1 Identifikasi Masalah dan Desain Eksperimen

Identifikasi masalah yang ada dalam penelitian ini memiliki keterkaitan dengan integrasi sistem *client-server* untuk memungkinkan perangkat apa pun yang terhubung dengan HTML dapat mengakses *server* melalui permintaan dan menerima respons dari *web server*. Tujuan dari eksperimen ini adalah untuk memungkinkan aplikasi untuk meminta model 3D berdasarkan penanda (Marker) AR tertentu dan menampilkan model tersebut secara langsung di layar perangkat AR.

Desain eksperimen ini disajikan dalam Gambar 1 yang merupakan gambaran dari arsitektur sistem. Sistem ini menggunakan teknologi web HTML dengan metode *rendering* WebGL dan GLSL yang dimanfaatkan dalam aplikasi AR. Proses *rendering* yang dilakukan dengan menggunakan WebGL dan GLSL memungkinkan objek 3D di-*render* pada komputer klien, sehingga menghasilkan tampilan visual secara AR. Dengan demikian, aplikasi AR dapat berjalan dengan lancar dan memberikan pengalaman visual yang baik bagi pengguna.



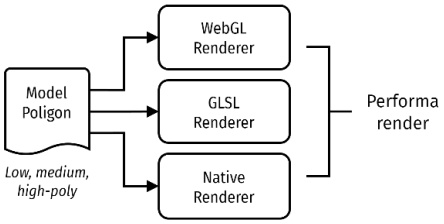
Gambar 1. Arsitektur Sistem

Arsitektur sistem yang diusulkan akan memungkinkan perangkat yang terhubung melalui HTML untuk dengan mudah mengakses server melalui permintaan dan menerima respon, agar mempermudah interaksi antara aplikasi AR dengan server untuk mendapatkan 3D model berdasarkan marker AR tertentu. Selain itu, pemanfaatan WebGL dan GLSL dalam proses *rendering* akan memastikan tampilan objek 3D dapat ditampilkan saat ditampilkan dalam sebuah aplikasi AR.

3.2 Pengukuran dan Pengumpulan Data

Prosedur pengukuran yang digunakan untuk mengukur kinerja *rendering* objek 3D menggunakan WebGL, GLSL dan *native* pada penelitian ini disajikan dalam Gambar 1. Setelah proses *rendering* objek 3D selesai dilakukan, pengukuran dan analisis performa AR juga akan dilakukan, baik itu yang dija-

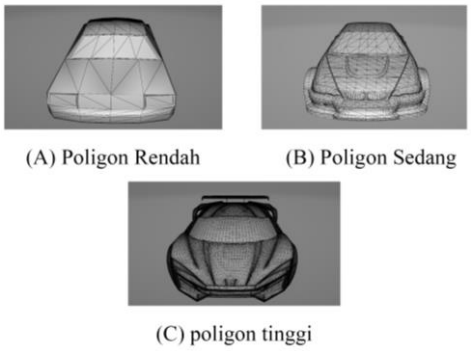
lankan di lingkungan web maupun lingkungan *native* untuk membandingkan performa dari jenis *shader* yang berjalan.



Gambar 2. Prosedur Pengukuran

Pengukuran ini akan melakukan pengujian dari berbagai jenis *shader*, dan setiap *shader* akan diterapkan pada tiga objek 3D yang berbeda, dengan setiap objek memiliki tingkat poligon yang berbeda. Ilustrasi mengenai tiga objek 3D tersebut dapat dilihat dalam Gambar 3.

Gambar 3 memperlihatkan objek 3D yang akan digunakan dalam penelitian ini, aplikasi AR akan menampilkan objek 3D dalam bentuk model mobil 3D dengan tingkat poligon yang berbeda. Objek dengan tingkat poligon yang tinggi tersusuna atas lebih dari 4 juta poligon, objek dengan tingkat poligon sedang memiliki 51 ribu poligon, dan objek dengan tingkat poligon paling rendah memiliki 2 ribu poligon.



Gambar 3. 3D object model yang digunakan dalam penelitian

Setelah objek ditampilkan, objek 3D akan dirotasi secara vertikal, horizontal, dan diagonal. Pengambilan data akan dilakukan selama proses pengukuran kinerja, Pengukuran kinerja terjadi ketika objek dirotasi kembali ke posisi sebelumnya, ketika kembali ke posisi sebelumnya hasil *rendering* dari poligon akan direkam. Pengukuran ini dilakukan menggunakan *rendering* 3D menggunakan WebGL, GLSL, dan *native*. Untuk setiap objek, proses pengukuran akan diulang sebanyak tiga puluh kali untuk mendapatkan data yang akurat.

Dengan melaksanakan langkah-langkah prosedur pengukuran yang benar, diharapkan dapat memperoleh data dan informasi yang objektif mengenai kinerja *rendering* objek 3D menggunakan WebGL, GLSL dan *native* dalam berbagai kondisi. Serta diharapkan dapat memperoleh informasi yang detail mengenai kinerja *rendering* objek 3D dalam berbagai tingkat poligon menggunakan teknologi WebGL dan GLSL. Perbandingan performa antara lingkungan web dan lingkungan *native* akan memberikan pemahaman yang lebih mendalam tentang kelebihan dan keterbatasan masing-masing pendekatan.

Setelah pengukuran dilakukan, hasilnya dikumpulkan untuk menghimpun data terkait penggunaan Memori (RAM) yang dihasilkan oleh setiap aplikasi AR saat menggunakan WebGL, GLSL, atau *native*. Tujuan utama dari pengumpulan data ini adalah untuk menganalisis dan membandingkan kinerja serta efisiensi aplikasi AR yang menggunakan kedua jenis *shader* tersebut dengan aplikasi berbasis *native*.

Data yang terkumpul akan diolah dan disusun berdasarkan kategori yang telah disesuaikan dengan sebelumnya. Pengolahan data ini bertujuan untuk mengatur dan menganalisis hasil dari pengukuran. Selama proses pengolahan data, data yang akan didapat berupa data kuantitatif, yang akan memberikan informasi detail tentang efisiensi dan kinerja aplikasi AR saat menggunakan WebGL dan GLSL.

Dengan informasi yang terkumpul dari pengumpulan data, diharapkan dapat mengidentifikasi kelebihan dan kekurangan penggunaan WebGL dan GLSL dalam konteks aplikasi AR.

3.3 Analisis

Dalam menganalisis data yang telah diperoleh dari penelitian ini, penelitian akan menggunakan dua metode pengujian statistik. Uji pertama yang dilakukan adalah uji normalitas Shapiro-Wilk, di mana data akan dievaluasi untuk mengetahui apakah data tersebut terdistribusi secara normal atau tidak. Untuk mengetahui jika berdistribusi secara normal atau tidak, yaitu ketika *p-value* dari uji ini bernilai kurang dari 0,05.

Ketika hasil dari uji Shapiro-Wilk menggambarkan data yang terdistribusi secara tidak normal, maka uji analisis selanjutnya menggunakan uji statistik non-parametrik. Uji Wilcoxon digunakan untuk membandingkan kinerja dari *shader* WebGL dan GLSL dalam aplikasi AR berbasis HTML lalu dibandingkan lagi dengan metode render yang dijalankan secara *native*. Dalam eksperimen ini, setiap kelompok percobaan akan terdiri dari objek dengan tingkat poligon dan gerakan yang berbeda untuk masing-masing *shader*.

4. HASIL DAN PEMBAHASAN

Studi ini menggunakan dua tipe aplikasi, yaitu WebGL dan GLSL *shader* yang berbasis HTML, dan AR dari perangkat *native*. Setiap *shader* diuji dengan tiga puluh pengukuran untuk setiap arah putaran, yaitu horizontal, vertikal, dan diagonal, dengan jumlah total pengukuran sejumlah 810 kali pengukuran performa *rendering* dilakukan selama penelitian ini. Data performa tersebut mencakup berbagai hasil

rendering dari aplikasi AR. Tabel 1 menunjukkan beberapa contoh hasil pengukuran atas proses *rendering* yang dilakukan oleh aplikasi AR yang diujikan dalam penelitian ini.

Tabel 1. Hasil dari Uji *Rendering*

Shader	Poligon	Putaran	Memori
WebGL	Low	Horizontal	333 MB
WebGL	Medium	Vertikal	317 MB
WebGL	High	Diagonal	576 MB
GLSL	Low	Horizontal	323 MB
GLSL	Medium	Vertikal	334 MB
GLSL	High	Diagonal	576 MB
Native	Low	Horizontal	248 MB
Native	Medium	Vertikal	248 MB
Native	High	Diagonal	380 MB

Setelah data telah didapat, data tersebut kemudian dikelompokkan ke dalam beberapa kelompok berdasarkan jenis eksperimen, *shader*, dan level poligon yang diujikan. Rangkuman data tersebut diperlihatkan pada Tabel 2.

Tabel 2. Rangkuman Data

Shader	Poligon	Rata-Rata
WebGL	Low	298 MB
WebGL	Medium	305.8 MB
WebGL	High	589 MB
GLSL	Low	302.03 MB
GLSL	Medium	304.17 MB
GLSL	High	580.32 MB
Native	Low	248 MB
Native	Medium	248 MB
Native	High	380 MB

Setelah data disusun, data tersebut diuji untuk mengetahui apakah memiliki distribusi normal. Pengujian data dilakukan dengan uji Shapiro-Wilk, dan hasil dari uji Shapiro-Wilk tersebut diperlihatkan pada Tabel 3. Hasil dari uji tersebut menunjukkan bahwa data yang memiliki nilai *p-value* lebih rendah dari 0,05 tidak terdistribusi normal.

Tabel 3. Hasil dari Uji Shapiro-Wilk

Shader	Poligon	p-value
WebGL	Low	$2{,}3 \times 10^{-05}$
WebGL	Medium	$8{,}5 \times 10^{-05}$
WebGL	High	$3{,}5 \times 10^{-05}$
GLSL	Low	$4{,}6 \times 10^{-05}$
GLSL	Medium	$2{,}2 \times 10^{-05}$

Dikarenakan data menghasilkan distribusi yang tidak normal ketika menggunakan uji shapiro-wilk. uji selanjutnya menggunakan uji wilcoxon.

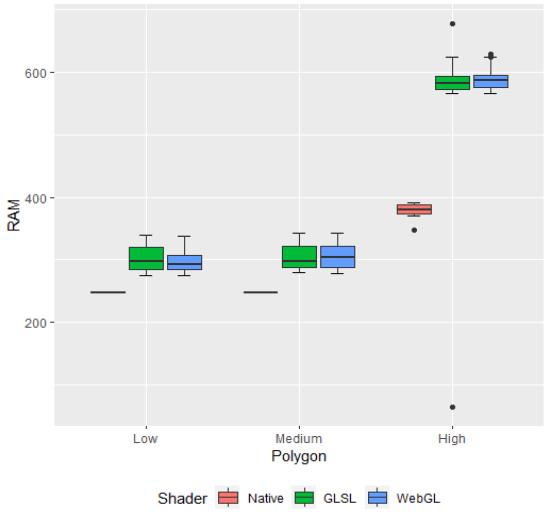
Untuk hasil uji Wilcoxon dapat dilihat pada tabel 4. Berdasarkan hasil uji Wilcoxon Test pada pengukuran RAM, dapat disimpulkan bahwa pada tingkat poligon rendah memiliki *p-value* sebesar 0,247 yang menunjukkan bahwa tidak ada perbedaan dalam penggunaan RAM antara *shader* WebGL dengan GLSL karena nilai *p-value* lebih tinggi dari 0,05. Namun, pada tingkat detail sedang maupun tinggi memiliki perbedaan penggunaan RAM, Pada tingkat detail poligon sedang, perbedaan antara penggunaan RAM WebGL dan GLSL terlihat ada perbedaan yaitu 305,8 MB untuk WebGL dan 304,17 MB untuk GLSL, tetapi perbedaannya terbilang sangat kecil. Sementara itu, pada tingkat detail poligon tinggi, terdapat perbedaan yang signifikan dalam penggunaan RAM antara WebGL dan GLSL. WebGL menggunakan 589 MB, sedangkan GLSL menggunakan 580,32 MB.

berdasarkan hasil uji Wilcoxon, dapat disimpulkan bahwa ada perbedaan yang signifikan dalam penggunaan RAM antara WebGL dan GLSL pada tingkat detail poligon menengah dan tinggi, WebGL cenderung menggunakan lebih banyak RAM daripada GLSL dalam kedua kondisi tersebut. Hal ini dapat dilihat dari nilai *p-value* yang sangat kecil untuk kedua kondisi tersebut, ini menunjukkan bahwa perbedaan tersebut sangat signifikan secara statistik, tetapi tidak ada perbedaan signifikan pada tingkat detail polygon yang rendah.

Tabel 4. Hasil Uji Wilcoxon untuk Memori yang digunakan

Poligon	p-value	WebGL	GLSL
Low	0,247	298 MB	302,03 MB
Medium	$2{,}2 \times 10^{-16}$	305,8 MB	304,17 MB
High	$2{,}2 \times 10^{-16}$	589 MB	580,32 MB

Sementara, hasil perbandingan dengan aplikasi ang me-*render* objek 3D secara *native* disajikan pada Tabel 5. Hasil perbandingan menunjukkan bahwa teknologi WebGL dan GLSL menggunakan RAM lebih banyak dari teknologi *rendering* yang dilakukan secara *native* di semua tingkat detail poligon. Pada tingkat detail poligon tinggi, WebGL menggunakan memori sebesar 589 MB, GLSL menggunakan memori sebesar 580,32 MB, dan aplikasi *native* menggunakan memori sebesar 380 MB. Hasil tersebut terjadi karena aplikasi AR yang menggunakan web memerlukan *web browser* untuk menjalankan aplikasinya, sehingga *web browser* memerlukan RAM yang besar ketika dijalankan. Perbandingan ini menunjuk-kan bahwa aplikasi *native* menggunakan RAM lebih sedikit dari pada aplikasi WebGL. Penggunaan memori untuk ketiga metode *rendering* WebGL, GLSL, dan *native* diperlihatkan dalam Gambar 4.



Gambar 4. Hasil dari Penelitian Pengukuran Memori.

Tabel 5. Hasil Perbandingan Memori dari WebGL dengan *Native*

Poligon	WebGL	GLSL	Native
Low	298	302,03 MB	248 MB
Medium	305.8	304,17 MB	248 MB
High	589	580,32 MB	380 MB

5. KESIMPULAN

Berdasarkan analisis yang telah dilakukan, terdapat perbedaan yang besar antara WebGL yang menggunakan *shader raw* dengan GLSL dalam hal performa penggunaan memori pada berbagai tingkat poligon. Dari hasil pengukuran di seluruh tingkat kepadatan poligon menunjukkan bahwa WebGL dan GLSL menggunakan lebih banyak RAM daripada aplikasi *native*. Hal tersebut disebabkan oleh aplikasi AR berbasis web membutuhkan *web browser* sebagai *container*-nya untuk menjalankan aplikasi. Untuk tingkat kepadatan poligon yang tinggi, teknologi WebGL menggunakan lebih banyak RAM dibandingkan dengan teknologi GLSL, sementara aplikasi *native* menggunakan jumlah RAM yang lebih sedikit dibandingkan keduanya.

6. DAFTAR PUSTAKA

TOLLE, H., PINANDITO, A., ADAMS, E.M. AND ARAI, K., 2015. Virtual reality game controlled with user's head and body movement detection using smartphone sensors. *Journal of Engineering and Applied Sciences*, 10(15), pp.6380–6387.

ASADZADEH, A., SAMAD-SOLTANI, T., REZAEI-HACHESU, P., 2021. Applications of virtual and augmented reality in infectious disease epidemics with a focus on the COVID-19 outbreak. Inform. Med. Unlocked 24, 100579. https://doi.org/10.1016/j.imu.2021.100579

CHOLISSODIN, I., JONEMARO, E.M.A., RAHAYUDI, B., KSATRIA, W.E., SUKMAWATI, A., MUZAYYANI, M.F., 2022. Pengembangan Fast Render Objek Grafis Menggunakan Shader dan Non-Shader Berbasis WebGL dari Primitive Object untuk Membuat Raw Metaverse Material Objek Skybox 3D di Filkom UB. J. Teknol. Inf. Dan Ilmu Komput. 9, 1357. https://doi.org/10.25126/jtiik.2022976739

FERRÃO, J.M.M., DIAS, P., SANTOS, B., OLIVEIRA, M., 2022. Environment-aware rendering and interaction in web-based augmented reality. http://dx.doi.org/10.2139/ssrn.4251267

FLOREZ, C.C., QUEVEDO, W., GALORA, F.J., TOASA, R.M., 2021. Performance of WebGL standard for displaying 3D applications on mobile devices, in: 2021 16th Iberian Conference on Information Systems and Technologies (CISTI). Presented at the 2021 16th Iberian Conference on Information Systems and Technologies (CISTI), IEEE, Chaves, Portugal, pp. 1–6. https://doi.org/10.23919/CISTI52073.2021.9476391

HE, Y., FOLEY, T., HOFSTEE, T., LONG, H., FATAHALIAN, K., 2017. Shader components: modular and high performance shader development. ACM Trans. Graph. 36, 1–11. https://doi.org/10.1145/3072959.3073648

HEINRICH, F., BORNEMANN, K., LAWONN, K., HANSEN, C., 2019. Depth Perception in Projective Augmented Reality: An Evaluation of Advanced Visualization Techniques. Proc. 25th ACM Symp. Virtual Real. Softw. Technol.

LI, L., QIAO, X., LU, Q., REN, P., LIN, R., 2020. Rendering Optimization for Mobile Web 3D Based on Animation Data Separation and On-Demand Loading. IEEE Access 8, 88474–88486. https://doi.org/10.1109/ACCESS.2020.2993613

MIAO, R., SONG, J., ZHU, Y., 2017. 3D geographic scenes visualization based on WebGL, in: 2017 6th International Conference on Agro-Geoinformatics. Presented at the 2017 6th International Conference on Agro-Geoinformatics, IEEE, Fairfax, VA, USA, pp. 1–6. https://doi.org/10.1109/Agro-Geoinformatics.2017.8046999

PARASHAR, V., 2020. AUGMENTED REALITY A NEW ERA IN EDUCATION. Int. J. Eng. Technol. Manag. Res. 5, 19–29. https://doi.org/10.29121/ijetmr.v5.i2.2018.608

PATRIO GONZALEZ VIVO, JEN LOWE, 2015. The Book of Shaders.

QIAO, X., REN, P., DUSTDAR, S., LIU, L., MA, H., CHEN, J., 2019. Web AR: A Promising Future for Mobile Augmented Reality—State of the Art, Challenges, and Insights. Proc. IEEE 107, 651–666. https://doi.org/10.1109/JPROC.2019.2895105

SCHWARZ, K., SAUER, A., NIEMEYER, M., LIAO, Y., GEIGER, A., 2022. VoxGRAF: Fast 3D-Aware Image Synthesis with Sparse Voxel Grids.

VAN KREVELEN, D.W.F., POELMAN, R., 2010. A Survey of Augmented Reality Technologies, Applications and Limitations. Int. J. Virtual Real. 9, 1–20. https://doi.org/10.20870/IJVR.2010.9.2.2767