Journal of Telecommunications, Networks, Electronics, and Computer Technologies
Volume 5, years 2025, 36 - 48

JURNAL UPI

TELNECT

Journal homepage: http://ejournal.upi.edu/index.php/TELNECT/

Analisis Penggunaan Frekuensi Sampling dan Filter Lowpass - FIR untuk Mencegah Aliasing

Cahyani Pebriyanti[1], Nur Asyiyah[2], Valentina Febrianti[3], Endah Setyowati[4]

Program Studi Sistem Telekomunikasi, Universitas Pendidikan Indonesia, Indonesia[1,2,3,4]
*Corresponding Author: Email: endah setyowati@upi.edu

ABSTRACTS

Penelitian ini bertujuan untuk memahami pengaruh frekuensi sampling terhadap masalah aliasing dalam pengolahan sinyal *digital* menggunakan MATLAB. Penelitian ini dilakukan melalui eksperimen dengan menguji tiga skenario yang berbeda, yaitu visualisasi sinyal, pemrosesan audio, dan penerapan penyaringan dengan *Low-Pass Filter (FIR)*. Hasil penelitian menunjukkan bahwa masalah aliasing cenderung terjadi pada frekuensi rendah, namun dapat dikurangi dengan meningkatkan frekuensi sampling. Penggunaan frekuensi sampling yang lebih rendah menghasilkan audio yang kurang jelas dan terdistorsi, sementara frekuensi sampling yang lebih tinggi menghasilkan hasil yang lebih jernih dan berkualitas. Penerapan *Low-Pass Filter (FIR)* berhasil mengurangi efek aliasing yang terjadi. Temuan ini menggarisbawahi pentingnya pemilihan frekuensi sampling yang tepat untuk menghindari masalah aliasing dan memastikan kualitas sinyal yang optimal dalam pengolahan sinyal *digital*. Penelitian ini memberikan panduan praktis bagi para pengguna MATLAB dalam memilih parameter sampling yang sesuai untuk menjaga integritas dan kejelasan sinyal *digital.* Kesimpulannya, pemilihan frekuensi sampling yang tepat sangat penting untuk mencegah gangguan aliasing dan memastikan hasil yang akurat dalam aplikasi pengolahan sinyal *digital*. Dengan demikian, penelitian ini memberikan wawasan berharga bagi pengembangan teknologi pengolahan sinyal yang lebih baik di masa depan.

ARTICLE INFO

Article History:
Received 30 April 2025
Revised 10 Mei 2025
Accepted 20 Mei 2025
Available online 1 Juni 2025

Keyword:
frekuensi sampling aliasing, visualisasi sinyal pemrosesan audio, *Low-Pass Filter* (LPF)

1. INTRODUCTION

Pengolahan sinyal adalah proses matematis yang digunakan pada sinyal untuk mengekstraksi, memodifikasi, atau meningkatkan informasi yang dikandungnya [1]. Proses ini menggunakan berbagai teknik dan algoritma untuk menganalisis, memodifikasi, dan menginterpretasikan sinyal dengan tujuan memperoleh informasi yang dibutuhkan. Pengolahan sinyal digunakan dalam berbagai aplikasi, termasuk komunikasi, audio, gambar, dan data medis.

Pengolahan sinyal dapat dibedakan menjadi 2 jenis, yaitu *analog* dan *digital* [2]. Pengolahan sinyal *analog* melibatkan manipulasi langsung dari sinyal kontinu, sedangkan pengolahan sinyal *digital* melibatkan manipulasi sinyal yang telah di diskritisasi. Salah satu aspek penting dalam pengolahan sinyal *digital* adalah teknik sampling. Teknik sampling adalah proses pengambilan sampel dari sinyal kontinu (*analog*) untuk menghasilkan sinyal diskrit (*digital*) [3]. Adapun Pengambilan sampel adalah cara mengubah sinyal yang berupa gelombang kontinu menjadi sekumpulan titik-titik terpisah. Misalnya, ketika merekam suara dengan mikrofon, sinyal suara yang terus menerus diubah menjadi data *digital* yang terdiri dari titik-titik yang merepresentasikan suara tersebut. Namun, untuk menghasilkan hasil yang akurat, perlu mengambil cukup banyak titik dari sinyal asli ini. Banyaknya titik yang dibutuhkan ditentukan oleh teorema *Nyquist-Shannon.* Teorema ini memungkinkan untuk merekonstruksi sinyal asli dengan melakukan interpolasi pada sampel-sampelnya [4]. Teorema ini juga menyatakan bahwa untuk merekonstruksi kembali sinyal asli dengan baik, perlu mengambil sampel dengan frekuensi setidaknya dua kali lipat dari frekuensi tertinggi dalam sinyal tersebut [5]. Jika nilai frekuensi sampling berada di bawah nilai frekuensi tertinggi sinyal, maka pada proses sampling tersebut akan terjadi fenomena aliasing [6].

Efek aliasing terjadi ketika sinyal diambil pada frekuensi yang lebih rendah dari ambang batas *Nyquist*, yang merupakan dua kali frekuensi tertinggi dari sinyal asli. Ketika frekuensi sampling berada di bawah ambang batas ini, sinyal yang diambil akan mengalami distorsi karena bagian frekuensi tinggi dari sinyal asli tidak dapat digambarkan dengan benar. Efek ini akan mengaburkan informasi asli sinyal, yang membuatnya sulit atau bahkan tidak mungkin untuk mengembalikan sinyal asli dengan akurat, serta akan membuat sinyal informasi menjadi rusak. Dalam perkembangannya, selain harus memenuhi syarat *Nyquist*, diperlukan juga langkah tambahan untuk mendukung terpenuhinya syarat tersebut. Langkah ini melibatkan pemfilteran sinyal sebelum dilakukan pencuplikan, sehingga hanya sinyal dengan frekuensi tertentu saja yang akan dicuplik. Filter ini disebut sebagai filter anti-aliasing dan bertujuan untuk memastikan bahwa hanya frekuensi yang sesuai yang diteruskan ke proses pencuplikan [7]. Dengan kata lain sinyal dengan frekuensi di atas frekuensi *cut-off* akan diredam, dan sinyal dengan frekuensi dibawah frekuensi *cut-off* akan dilewatkan [8].

Penelitian mengenai penerapan teknik sampling sudah banyak dilakukan oleh peneliti lain, seperti pada penelitian Mihai Bogdan [5] yang menyoroti pentingnya pemilihan frekuensi pengambilan sampel yang tepat untuk memastikan rekonstruksi yang akurat dari sinyal *analog* menjadi sinyal *digital.* Hasilnya menunjukkan frekuensi sampling yang cukup tinggi harus digunakan untuk menghindari aliasing dan memastikan bahwa sinyal yang diambil cukup representatif terhadap sinyal asli. Filter anti-aliasing digunakan untuk menghilangkan frekuensi di atas frekuensi *Nyquist* sebelum proses sampling. Pada penelitian yang ditulis oleh Luc Lévesque [9] artikel menjelaskan dasar-dasar Teorema *Sampling Nyquist*, yang menyatakan bahwa untuk merekonstruksi sinyal dengan benar dari sampelnya, frekuensi sampling harus setidaknya dua kali lipat dari frekuensi tertinggi dalam sinyal asli, artikel ini memberikan pemahaman tentang bagaimana Teorema *Nyquist* dan konsep aliasing mempengaruhi perekaman dan pemrosesan gambar, dengan fokus khusus pada fenomena ilusi visual yang sering diamati dalam video.

Walaupun kedua artikel sudah memberikan penjelasan yang baik mengenai teorema sampling *Nyquist,* namun masih terdapat beberapa kekurangan yang dapat diidentifikasi, seperti pendekatan teoritis yang terlalu berat, ilustrasi visual yang terbatas, kurangnya ilustrasi atau contoh kasus nyata, keterbatasan dalam penjelasan dari teorema *Nyquist-Shannon,* kurangnya eksplorasi mendalam tentang filter anti-aliasing. Oleh karena itu, penelitian ini bertujuan untuk melengkapi kekurangan informasi tersebut. Penelitian ini akan dimulai dengan konsep dasar berupa visualisasi sinyal digital untuk memahami pengaruh Frekuensi Sampling terhadap efek aliasing, diikuti dengan studi kasus mengenai lagu berupa audio. Selanjutnya, penelitian akan menerapkan filter anti-aliasing untuk mengevaluasi bagaimana filter tersebut mempengaruhi sinyal sebelum dan sesudah proses *filtering*.

Dengan visualisasi sinyal diharapkan pembaca dapat memahami terlebih dahulu mengenai dasar pemahaman frekuensi sampling. Dilanjutkan dengan eksperimen yang akan dilakukan pada file audio dengan format WAV. File audio ini akan dianalisis menggunakan berbagai frekuensi sampling yang berbeda untuk mengevaluasi dampak variasi frekuensi sampling terhadap *output* yang dihasilkan. Untuk menerapkan filtering anti-aliasing, peneliti akan menggunakan *Low-Pass Filter (FIR).* Filter anti-aliasing, seperti *Low-Pass Filter*, dirancang untuk meredam komponen frekuensi yang lebih tinggi dari

DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

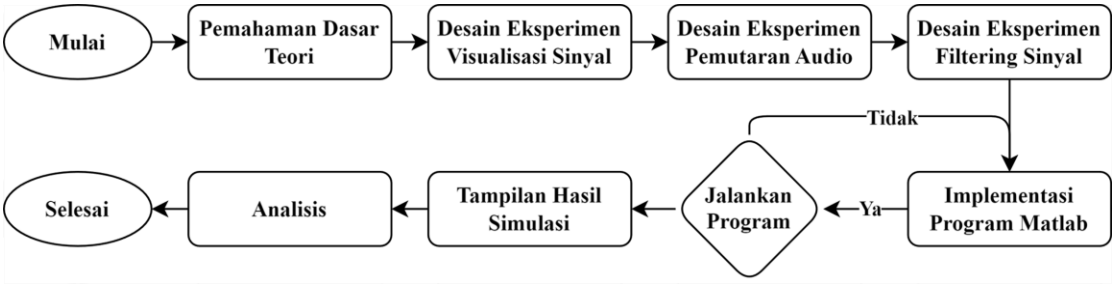
setengah frekuensi sampling (frekuensi *Nyquist*) sehingga hanya frekuensi yang diinginkan yang akan dicuplik. Filter *FIR (Finite Impulse Response)*, *Filter* ini banyak dirancang untuk menghilangkan noise dari sinyal audio yang rusak [10], namun pada penelitian ini lebih ditekankan untuk pemfilteran filter anti-aliasing.

Serangkaian percobaan tersebut akan memanfaatkan software *MATLAB. MATLAB* adalah perangkat lunak yang sering digunakan untuk berbagai keilmuan dikarenakan keunggulannya untuk pengolahan sinyal dan anggaran matriks. Selain itu algoritma dan fitur yang diberikan sangat unik agar dapat mencapai hasil yang optimal dan efisien. Pada penelitian ini *MATLAB* sangat berperan penting agar mendapatkan sebuah hasil perbandingan pemilihan frekuensi sampling terhadap visual dan audio [11]

2. METODE PENELITIAN

2.1 Alur Penelitian

Metode penelitian yang digunakan dalam studi ini adalah eksperimen dengan melibatkan penggunaan perangkat lunak *MATLAB*. Adapun proses atau langkah-langkah dari alur penelitian terdapat pada gambar 1.



Gambar 1. Alur Penelitian

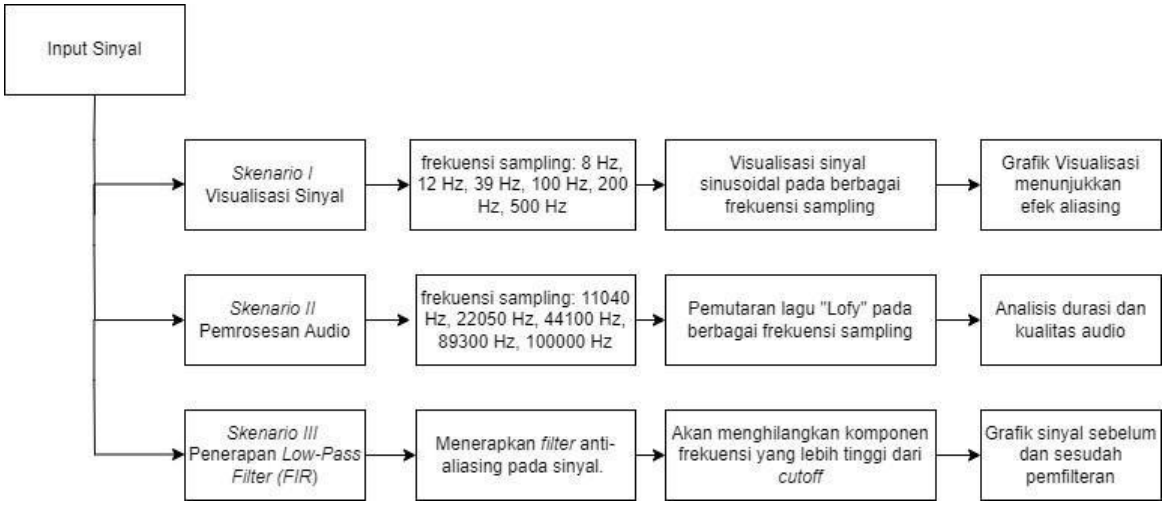
Penelitian ini dimulai dengan memperdalam dasar teori mengenai sampling dan aliasing yang mencakup Teorema *Nyquist, undersampling, oversampling*, dan filter anti-aliasing melalui pencarian dan penelusuran pada database seperti Google Scholar dan Garuda.

Pada penelitian ini akan mencakup 3 scenario yaitu yang pertama Eksperimen visualisasi sinyal dengan variasi frekuensi sampling yang berbeda yaitu dengan fs sebesar 8 Hz, 12 Hz, 39 Hz, 100 Hz, 200 Hz, 500 Hz. Pengambilan sampel ini dilakukan dengan frekuensi sampling terendah hingga tertinggi untuk melihat bagaimana variasi ini mempengaruhi representasi sinyal. Skenario kedua akan lebih kompleks karena melibatkan pemrosesan audio, yang mencakup aspek suara dan kualitas audionya. Dalam skenario ini, langkah-langkahnya mencakup mengunduh lagu *"Lofy"* dengan lisensi anti *copyright*, mengonversi format file lagu dari MP3 ke WAV, dan menentukan frekuensi sampling (fs) yang akan digunakan dalam eksperimen. Nilai fs yang akan diuji adalah 11025 Hz, 22050 Hz, 44100 Hz, dan 88200 Hz. Skenario ketiga adalah penerapan filter anti-aliasing pada sinyal masukan sebelum proses digitalisasi. Filter anti-aliasing yang digunakan adalah filter *low-pass* FIR, yang berfungsi untuk memastikan bahwa frekuensi tinggi dari sinyal tidak menyebabkan aliasing saat direkonstruksi dengan frekuensi sampling yang rendah.

Implementasi program pada *MATLAB* dimulai dengan memasukkan kode program untuk teknik sampling sesuai dengan desain eksperimen visualisasi sinyal dan pemutaran audio. Setelah kode program dimasukkan, program dijalankan untuk menghasilkan keluaran simulasi. Jika berhasil, akan menampilkan hasil simulasi. Jika program tidak berhasil, peneliti harus mengecek dan memperbaiki kesalahan pada kode. Langkah selanjutnya adalah analisis data keluaran untuk memahami efek teorema *Nyquist* dalam eksperimen, baik pada visualisasi sinyal maupun pemutaran audio lagu *"Lofy"*. Peneliti akan merumuskan dan membandingkan hasil dari berbagai kasus sampling, mengevaluasi kualitas audio, mengamati perubahan dalam komponen frekuensi sinyal, dan meninjau efek dari penerapan filter aliasing.

2.2 Blok Diagram Kerja Sistem

Diagram kerja sistem tergambar pada Gambar 2. Gambar tersebut akan membagi cara kerja menjadi tiga skenario eksperimen: visualisasi sinyal, pemrosesan audio, dan penyaringan dengan *Low-Pass Filter* (FIR).



Gambar 2. Blok Diagram

Gambar 2 menggambarkan tiga skenario untuk memproses sinyal input. Setiap skenario memiliki frekuensi sampling dan proses spesifik yang menghasilkan output berbeda sesuai dengan tujuan pemrosesan. Berikut adalah penjelasan cara kerja dari setiap skenario yang terdapat pada Gambar 2.

2.2.1 Skenario I (Visualisasi Sinyal)

Dalam bagian visualisasi sinyal ini, peneliti akan menjelaskan proses kerja dan perintah *MATLAB* yang digunakan untuk visualisasi sinyal pada berbagai frekuensi sampling. Proses ini dimulai dengan menentukan parameter sinyal untuk frekuensi sampling (fs). Frekuensi sampling yang diuji dalam penelitian ini adalah 8 Hz, 12 Hz, 39 Hz, 100 Hz, 200 Hz, dan 500 Hz. Adapun *script* atau perintah pada *MATLAB* tercantum dalam TABEL 1

TABEL 1. VISUALISASI SINYAL FREKUENSI SAMPLING PADA SKENARIO I

No.	Kode	Deskripsi
1	Fs1 = 8; % Frekuensi sampling pertama	Menetapkan frekuensi sampling pertama (Fs1) sebesar 8 Hz.
2	t1 = (0:Fs1-1)/Fs1;	Menghitung vektor waktu (t1) dari 0 hingga 1 detik dengan langkah 1/Fs1.
3	s1 = sin(2*pi*t1*2);	Membuat sinyal pertama (s1) menggunakan fungsi sinus dengan frekuensi 2 Hz.
4	figure;	Membuat figure baru untuk menampilkan plot.
5	% Plot sinyal pertama	Komentar yang menjelaskan bahwa bagian berikutnya akan menggambarkan sinyal pertama yang telah dibuat.
6	subplot(2,2,1)	Mengatur subplot untuk grafik pada posisi pertama dari grid 2x2.
7	stem(t1, s1, 'r', 'filled', 'MarkerFaceColor', 'r', 'LineWidth',1.5)	Menggambar plot diskret dari sinyal pertama (s1) terhadap waktu (t1).
8	axis([0 1 -1.2 1.2])	Mengatur batas sumbu X (0 hingga 1 detik) dan sumbu Y (-1.2 hingga 1.2).
9	title('Fs = 8 Hz', 'FontSize', 14, 'FontWeight', 'bold', 'Color', 'r')	Menambahkan judul pada plot dengan teks "Fs = 8 Hz".
10	xlabel('Waktu (detik)', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'k')	Menambahkan label pada sumbu X dengan teks "Waktu (detik)".
11	ylabel('Amplitudo', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'k')	Menambahkan label pada sumbu Y dengan teks "Amplitudo".
12	grid on	Mengaktifkan grid pada plot untuk meningkatkan keterbacaan grafik.

DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

Tabel 1 berisikan program atau kode yang menghasilkan sinyal sinusoidal dengan frekuensi 2 Hz yang ditampilkan dalam subplot pertama dari sebuah *figure*. Parameter fs 1 menentukan frekuensi sampling sinyal pertama, yang diatur menjadi 8 Hz. Pada bagian ini fs dapat diatur sesuai dengan keinginan pengujian. Selanjutnya sinyal sinusoidal dinormalisasi dengan menggunakan waktu yang dihitung berdasarkan frekuensi sampling tersebut. Hasilnya, sinyal sinusoidal ini dipresentasikan dalam bentuk grafik batang (*stem plot*) dengan sumbu waktu dari 0 hingga 1 detik, dan sumbu amplitudo dari -1.2 hingga 1.2. Tampilan grafik ini menggunakan judul, label sumbu, serta *grid* untuk memperjelas visualisasi sinyal. Secara ringkas *output* dari kode *MATLAB* yang sudah dijalankan tersebut berupa grafik visual yang mengilustrasikan perbandingan antara sinyal asli dan sinyal yang disampling pada berbagai frekuensi sampling yang diuji.

2.2.2 Skenario II (Pemrosesan Audio)

Dalam bagian pemrosesan Audio, peneliti akan menjelaskan proses kerja dan perintah *MATLAB* yang digunakan pada audio dengan berbagai frekuensi sampling. Proses ini dimulai dengan membaca file audio. Adapun *script* atau perintah pada *MATLAB* tercantum dalam

TABEL 2. PEMROSESAN AUDIO PADA SKENARIO II

No.	Kode	Deskripsi
1	[audio, fs_original] = audioread('rekaman_lofy.wav');	Membaca file audio bernama 'rekaman_lofy.wav' dan menyimpan data audio dalam variabel audio serta frekuensi sampelnya dalam fs_original.
2	fs = 11025;	Menetapkan frekuensi sampel baru (fs) sebesar 11025 Hz untuk pemutaran audio.
3	nSamples = size(audio, 1);	Menghitung jumlah sampel dalam sinyal audio dengan mengambil dimensi pertama dari variabel audio.
4	duration = nSamples / fs;	Menghitung durasi sinyal audio (duration) dengan membagi jumlah sampel dengan frekuensi sampel baru.
5	time = linspace(0, duration, nSamples);	Membuat vektor waktu (time) dari 0 hingga durasi dengan jumlah sampel yang telah dihitung.
6	figure;	Membuat figure baru untuk menampilkan plot.
7	plot(time, audio);	Menggambar plot sinyal audio terhadap waktu.
8	xlabel('Waktu (detik)');	Menambahkan label pada sumbu X dengan teks 'Waktu (detik)'.
9	ylabel('Amplitudo');	Menambahkan label pada sumbu Y dengan teks 'Amplitudo'.
10	title(['Sinyal Audio (FS = ', num2str(fs), ' Hz)']);	Menambahkan judul pada plot dengan informasi frekuensi sampel baru.
11	grid on;	Mengaktifkan grid pada plot untuk meningkatkan keterbacaan grafik.
12	% Memainkan audio dengan frekuensi sampel baru	Komentar yang menjelaskan bahwa baris berikutnya akan memainkan audio.
13	sound(audio, fs);	Memainkan audio dengan frekuensi sampel yang baru.
14	disp(['Frekuensi Sampel Asli: ', num2str(fs_original), ' Hz']);	Menampilkan frekuensi sampel asli dari file audio.
15	disp(['Frekuensi Sampel Baru: ', num2str(fs), ' Hz']);	Menampilkan frekuensi sampel baru yang telah ditetapkan.
16	disp(['Durasi Audio: ', num2str(duration), ' detik']);	Menampilkan durasi dari audio dalam detik.

Tabel 2. berisikan program atau perintah yang dimulai dengan membaca file audio yang akan diproses, dan kemudian menetapkan frekuensi sampling yang baru untuk diputar sesuai dengan besaran sampel yang akan diuji, dalam kode tersebut hanya mengambil pengujian awal sebesar 11025 Hz dengan catatan fs tersebut dapat disesuaikan dengan nilai fs yang lainnya. Selanjutnya, jumlah sampel dan frekuensi sampling baru digunakan untuk menghitung durasi audio. Untuk mempermudah visualisasi dan analisis lebih lanjut, sumbu waktu dibuat untuk menyesuaikan frekuensi sampling baru. Selain itu, untuk mendengarkan hasil pemrosesan secara langsung, perintah *sound* digunakan untuk memutar audio dengan frekuensi sampling yang baru. Pada *command window* akan menampilkan informasi yang berisi: frekuensi sampling asli, frekuensi sampling baru, dan durasi audio. Hal tersebut akan memberikan pemahaman dan analisis yang lebih baik tentang bagaimana frekuensi sampling mempengaruhi kualitas pemrosesan audio.

2.2.2 Skenario III (Filter Anti-Aliasing (FIR))

Perintah untuk penerapan filtering anti aliasing dengan FIR tertulis pada Tabel 3. Penerapan dari filter anti-aliasing *(FIR)* digunakan untuk mengurangi aliasing pada sinyal masukan sebelum proses digitalisasi lebih lanjut.

TABEL 3. FILTER ANTI ALIASING LOW-PASS FILTER (FIR)

No.	Kode	Deskripsi
1	% Definisi sinyal masukan	Definisi sinyal masukan.
2	Fs = 100;	Frekuensi sampling yang rendah.
3	t = 0:1/Fs:1;	Vektor waktu dari 0 hingga 1 detik dengan langkah 1/Fs.
4	f1 = 50;	Frekuensi komponen sinyal pertama.
5	f2 = 150;	Frekuensi komponen sinyal kedua.
6	input_signal = sin(2*pi*f1*t) + 0.5*sin(2*pi*f2*t);	Sinyal masukan (gabungan dari dua komponen sin).
7	% Plot sinyal masukan sebelum filtering anti-aliasing	Plot sinyal masukan sebelum filtering anti-aliasing.
8	subplot(2,1,1);	Menampilkan subplot pertama pada posisi (2,1,1).
9	plot(t, input_signal);	Menggambar plot sinyal masukan.
10	title('Sinyal Masukan (Sebelum Filtering Anti-Aliasing)');	Menambahkan judul pada plot sinyal masukan.
11	xlabel('Waktu (s)');	Menambahkan label pada sumbu X dengan teks 'Waktu (s)'.
12	ylabel('Amplitudo');	Menambahkan label pada sumbu Y dengan teks 'Amplitudo'.
13	% Filter anti-aliasing (misalnya low-pass FIR filter)	Komentar untuk filter anti-aliasing (low-pass FIR filter).
14	cutoff_freq = 14;	Frekuensi cutoff filter anti-aliasing
15	nyquist_freq = Fs / 2;	Frekuensi Nyquist.
16	normalized_cutoff = cutoff_freq / nyquist_freq;	Normalisasi frekuensi cutoff.
17	filter_order = 30;	Urutan filter.
18	anti_aliasing_filter = fir1(filter_order, normalized_cutoff);	Desain filter FIR.
19	% Filtering sinyal masukan dengan filter anti-aliasing	Filtering sinyal masukan dengan filter anti-aliasing.
20	filtered_signal = filter(anti_aliasing_filter, 1, input_signal);	Memfilter sinyal masukan dengan filter anti-aliasing.
21	% Plot sinyal setelah filtering anti-aliasing	Plot sinyal setelah filtering anti-aliasing.
22	subplot(2,1,2);	Menampilkan subplot kedua pada posisi (2,1,2).
23	plot(t, filtered_signal);	Menggambar plot sinyal setelah filtering.
24	title('Sinyal Setelah Filtering Anti-Aliasing');	Menambahkan judul pada plot sinyal setelah filtering.

DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

25	xlabel('Waktu (s)');	Menambahkan label pada sumbu X dengan teks 'Waktu (s)'.
26	ylabel('Amplitudo');	Menambahkan label pada sumbu Y dengan teks 'Amplitudo'.

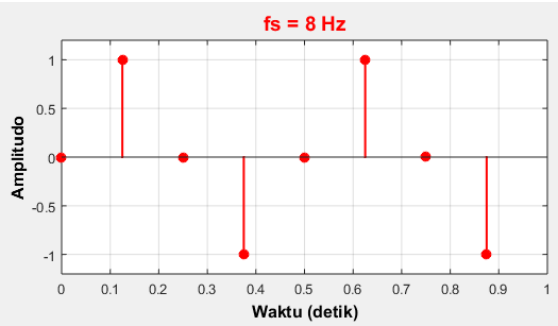
Tabel 3 berisikan kode yang mendefinisikan sinyal masukan yang terdiri dari gabungan dua komponen sinusoidal dengan frekuensi 50 Hz dan 150 Hz. Frekuensi sampling yang digunakan adalah 100 Hz, dengan vektor waktu yang mencakup interval dari 0 hingga 1 detik. Sinyal masukan divisualisasikan melalui plot untuk menampilkan komponen frekuensi sebelum penyaringan. Filter anti-aliasing yang diterapkan adalah *filter low-pass FIR* dengan frekuensi *cutoff* 14 Hz, yang dinormalisasi terhadap frekuensi *Nyquis*t dan dirancang menggunakan fungsi `*fir1*` dengan urutan filter sebesar 30. Setelah desain filter, sinyal masukan disaring menggunakan filter FIR, dan hasil penyaringan divisualisasikan pada plot kedua untuk menilai efektivitas filter dalam menghilangkan komponen frekuensi tinggi yang tidak diinginkan.

3. HASIL DAN PEMBAHASAN

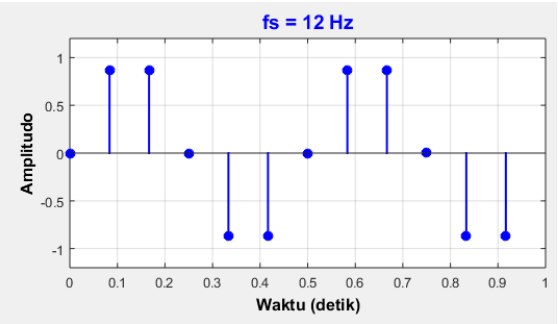
Hasil dan pembahasan program diimplementasikan melalui penggunaan beberapa perintah dalam aplikasi *MATLAB*. Proses ini membagi hasil menjadi beberapa tahap dengan variasi Fs yang berbeda, sehingga memungkinkan pemahaman yang lebih baik tentang pengaruh frekuensi sampling terhadap sinyal. Dengan demikian, program ini memberikan pemahaman yang lebih dalam tentang konsep sampling dalam domain waktu dan frekuensi, serta efeknya terhadap kualitas audio hasil digitalisasi.

3.1 Skenario I (Visualisasi Sinyal)

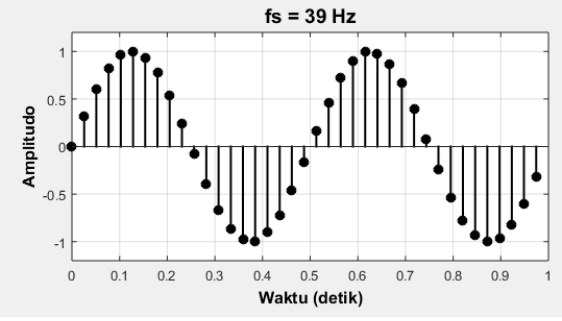
Untuk tahap pertama yaitu Visualisasi Sinyal dengan, frekuensi sampling (Fs) yang akan diuji adalah 8 Hz, 12 Hz, 39 Hz, 100 Hz, 200 Hz, dan 500 Hz. Adapun hasil dari visualisasi sinyal pada *MATLAB* tergambar pada Gambar 3.



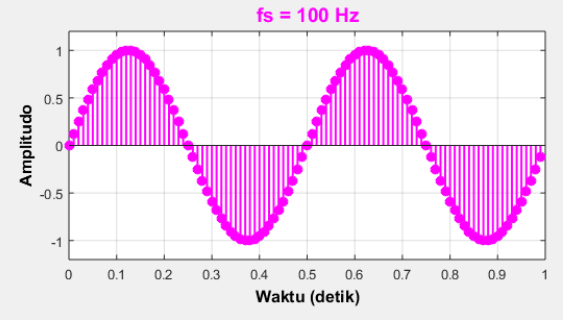
(a)



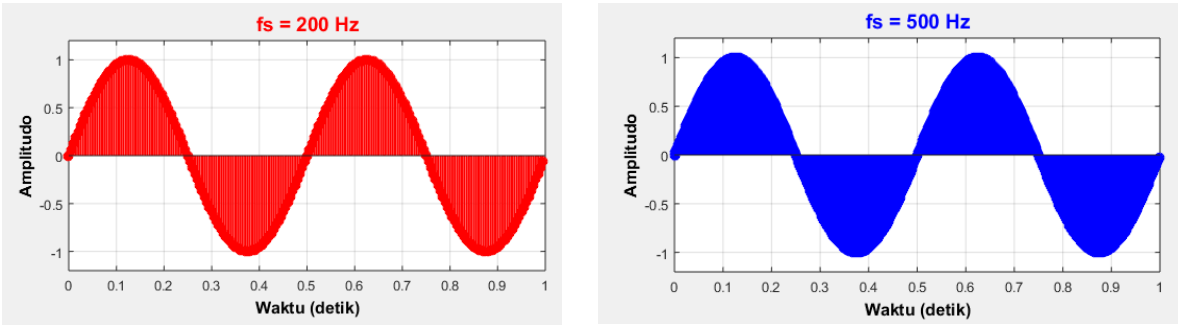
(b)



(c)



(d)



(e) (f)

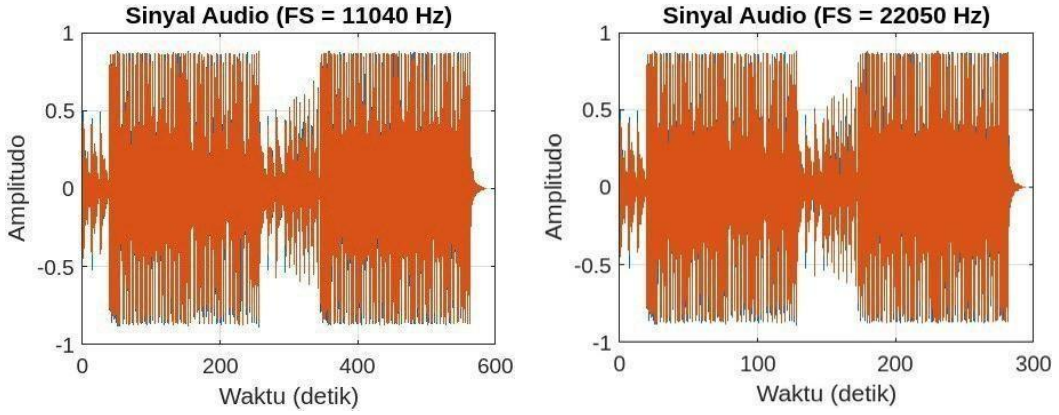
Gambar 3. Visualisasi Sinyal: (a) 8 Hz, (b) 12 Hz, (c) 39 Hz, (d) 100 Hz, (e) 200 Hz (f) 500 Hz Gambar

3 berisikan enam gambar visualisasi sinyal sinusoidal yang dihasilkan menggunakan program *MATLAB*, masing-masing dengan frekuensi sampling yang berbeda. Frekuensi sampling yang digunakan meliputi: 8 Hz (gambar 3a), 12 Hz (gambar 3b), 39 Hz (gambar 3c), 100 Hz (gambar 3d), 200 Hz (gambar 3e), dan 500 Hz (gambar 3f). Gambar-gambar ini menunjukkan bagaimana variasi frekuensi sampling mempengaruhi representasi sinyal, memberikan pemahaman tentang efek perubahan frekuensi sampling pada kualitas dan akurasi sinyal yang dihasilkan.

Analisis dari visualisasi sinyal yang ditunjukkan pada gambar 3, dengan pengambilan sampel sinyal frekuensi sampling yang terlalu rendah seperti 8 Hz dan 12 Hz akan menghasilkan aliasing yang menonjol, yang menyebabkan sinyal asli tidak dapat direkonstruksi dengan baik. Sebaliknya, frekuensi sampling yang lebih tinggi seperti 39 Hz, 50 Hz, 200 Hz, 300 Hz, dan 500 Hz mendapatkan representasi sinyal yang semakin akurat dan mengurangi aliasing secara signifikan. Saat frekuensi sampling yang sangat tinggi seperti 500 Hz, sinyal yang dihasilkan benar-benar identik dengan sinyal asli, maka hal ini menunjukkan pentingnya memilih frekuensi sampling yang cukup tinggi untuk menghasilkan sinyal yang akurat. Penting juga untuk memilih frekuensi sampling yang minimal dua kali lipat dari frekuensi sinyal tertinggi dalam sinyal asli.

3.2 Skenario II (Audio)

Pada Skenario II, dilakukan eksperimen untuk mengevaluasi pengaruh perubahan frekuensi sampling (fs) pada lagu "*lofy*", dengan fs yang diuji adalah sebesar 11040 Hz, 22050 Hz, 44100 Hz, 89300 Hz, dan 10000 Hz. Setelah pemberian fs baru, hasilnya akan direpresentasikan dalam grafik sinyal audio dan suara audio, dan akan juga menampilkan *command window* yang berisikan informasi mengenai fs awal, fs baru, dan durasi audio. Eksperimen ini bertujuan untuk memberikan pemahaman yang lebih dalam tentang pentingnya pemilihan frekuensi sampling yang tepat. Dengan mengevaluasi berbagai fs, peneliti dapat menentukan fs optimal yang memberikan representasi digital terbaik dari lagu *"Lofy"*, memastikan bahwa kualitas suara tetap tinggi dan bebas dari distorsi serta masalah temporal yang disebabkan oleh *sampling rate* yang tidak tepat. Gambar 7, merupakan representasi hasil dari program *MATLAB* terhadap sinyal audio.



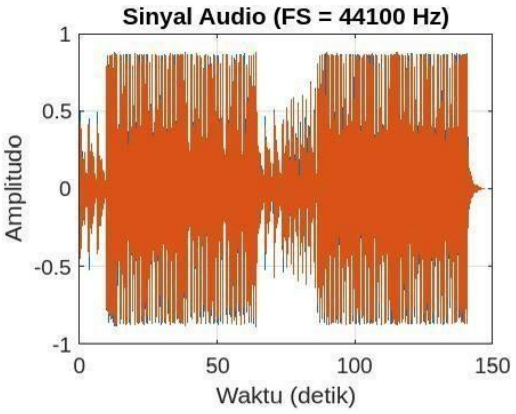
DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

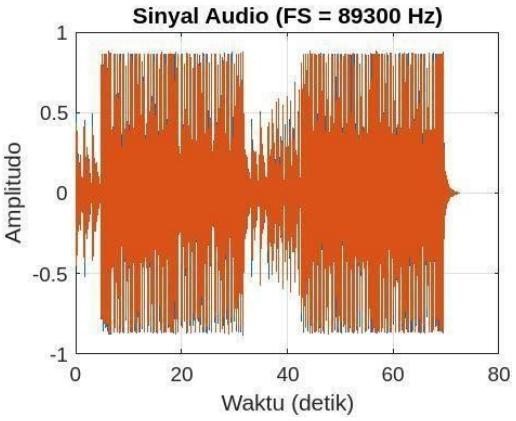
```
Command Window
>> studicase_audio

Frekuensi Sampel Asli: 44100 Hz
Frekuensi Sampel Baru: 11040 Hz
Durasi Audio: 588.1043 detik
```

(a)

```
Command Window
>> studicase_audio

Frekuensi Sampel Asli: 44100 Hz
Frekuensi Sampel Baru: 22050 Hz
Durasi Audio: 294.4522 detik
```

(b)



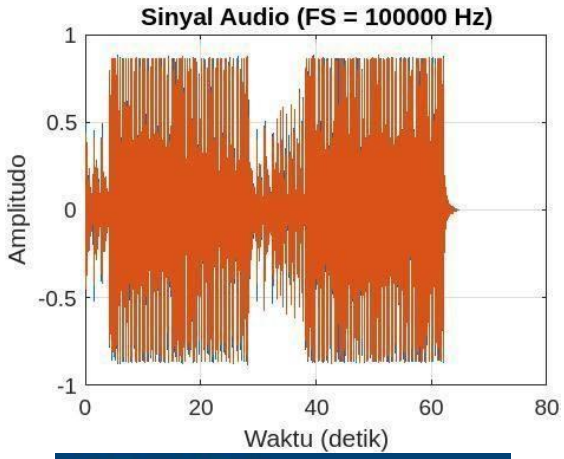



```
Command Window
>> studicase_audio

Frekuensi Sampel Asli: 44100 Hz
Frekuensi Sampel Baru: 44100 Hz
Durasi Audio: 147.2261 detik
```

(c)

```
Command Window
>> studicase_audio

Frekuensi Sampel Asli: 44100 Hz
Frekuensi Sampel Baru: 89300 Hz
Durasi Audio: 72.7063 detik
```

(d)




```
Command Window
>> studicase_audio
Frekuensi Sampel Asli: 44100 Hz
Frekuensi Sampel Baru: 100000 Hz
Durasi Audio: 64.9267 detik
```

(e)

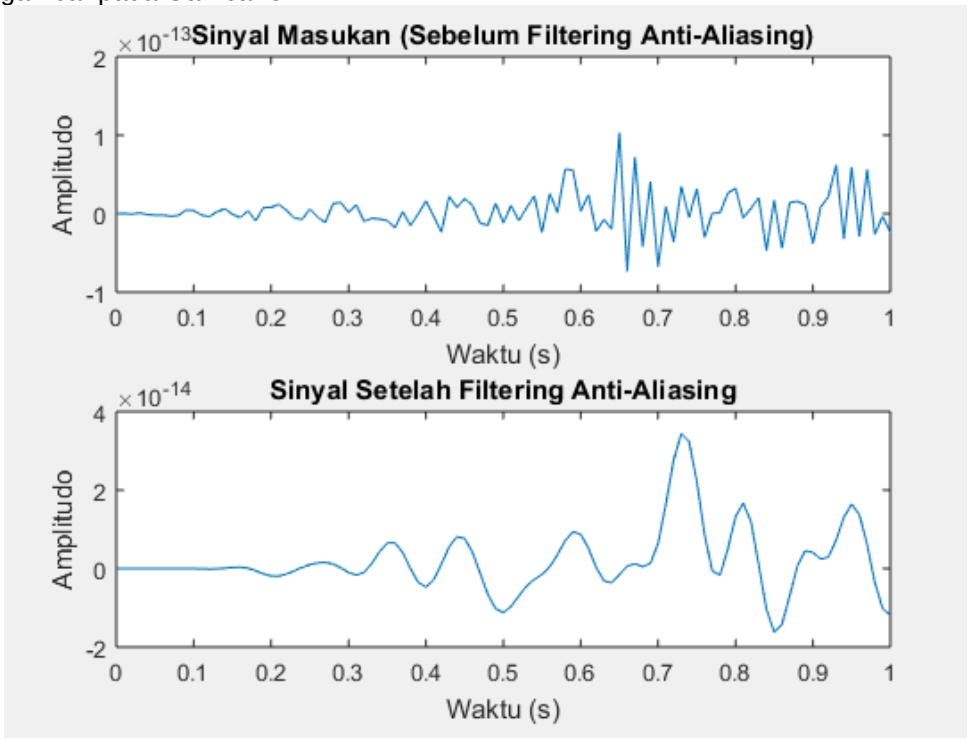
Gambar 4. Audio: (a) fs 11040 Hz, (b) fs 22050, (c) fs 44100 Hz, (d) fs 89300 Hz, (e) fs 100000 Hz

Gambar 4 merupakan hasil dari representasi audio dalam program *MATLAB* yang melibatkan pemutaran lagu "*Lofy*" pada berbagai frekuensi sampling. Gambar ini menunjukkan bagaimana perubahan frekuensi sampling, yaitu 11040 Hz (gambar 4a) , 22050 Hz (gambar 4b), 44100 Hz (gambar 4c), 89300 Hz (gambar 4d), dan 100000 Hz (gambar 4e), dapat mempengaruhi kualitas dan durasi audio. Di bawah gambar tersebut, terdapat jendela perintah (*command window*) yang memuat informasi seperti frekuensi sampling asli (fs asli), frekuensi sampling baru (fs baru), dan durasi audio. Informasi ini sangat membantu dalam analisis karena memberikan konteks tentang bagaimana perubahan dalam frekuensi sampling mempengaruhi durasi pemutaran dan kualitas audio.

Penerapan teorema *Nyquist* sebuah lagu *"Lofy"* di *MATLAB* menghasilkan beberapa temuan penting dalam hasil analisis audio. Lagu *"Lofy"* yang dianalisis memiliki rentang frekuensi yang cukup lebar, dengan komponen frekuensi tinggi yang perlu diperhatikan agar tidak terjadi distorsi selama proses sampling. Dengan frekuensi sampling sebesar 11040 Hz, durasi audio mencapai 588 detik dan terdengar sangat berbeda dari aslinya karena frekuensi tinggi tidak terwakili dengan baik, menyebabkan distorsi dan perlambatan durasi audio. Pada frekuensi sampling 22050 Hz, durasi audio adalah 294 detik, dengan beberapa distorsi yang masih terlihat, namun lebih baik dibandingkan dengan 11040 Hz, meskipun masih dua kali lebih lambat dari aslinya. Dengan frekuensi sampling 44100 Hz, durasi audio sesuai dengan lagu aslinya dan terdengar seperti suara aslinya, karena frekuensi ini merupakan standar untuk audio CD yang memastikan seluruh rentang frekuensi pendengaran manusia (hingga sekitar 20 kHz) dapat direkonstruksi dengan baik tanpa distorsi. Namun, pada fs sebesar 89300 Hz dan 100000 Hz, audio juga tidak terdengar seperti suara aslinya karena semakin tinggi Fs, maka audio terdengar semakin cepat. Oleh karena itu, penting untuk memilih frekuensi sampling yang tepat yang tidak hanya menghindari distorsi tetapi juga mempertahankan durasi dan kecepatan audio yang sesuai dengan aslinya.

3.3 Skenario III (Filter Anti-Aliasing)

Pada skenario III dilakukan penerapan filter anti-aliasing pada sinyal masukan yang terdiri dari dua komponen frekuensi. Sinyal tersebut memiliki frekuensi sampling yang relatif rendah, sehingga perlu dilakukan filter anti-aliasing untuk mengurangi efek aliasing. Untuk hasil grafik dari program *MATLAB*, tergambar pada Gambar 5.



Gambar 5. Penerapan filter lowpass (FIR)

DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

Pada gambar 5, menunjukkan hasil penerapan filter anti-aliasing menggunakan *Low-Pass Filter (FIR)* dengan frekuensi *cut-off* 14 Hz, yang dirancang untuk menghilangkan komponen frekuensi di atas 14 Hz. Sinyal yang dihasilkan menunjukkan penghapusan efektif dari komponen frekuensi tinggi (50 Hz dan 150 Hz). Proses penyaringan ini menghasilkan perubahan yang lebih halus pada sinyal yang sebelumnya tampak kasar, mengalami aliasing, dan distorsi. Hal ini menandakan bahwa *Low-Pass Filter (FIR)* berhasil mengurangi komponen frekuensi tinggi yang dapat menyebabkan ketidakhalusan dan distorsi pada sinyal. Dengan demikian, sinyal yang telah disaring tampak lebih terstruktur dan bebas dari distorsi aliasing yang dapat mengurangi kualitas sinyal. Penggunaan filter anti-aliasing akan memastikan bahwa sinyal yang dihasilkan setelah proses pengambilan sampel dan pemrosesan lebih akurat serta mewakili sinyal aslinya.

3.4 PERBANDINGAN DENGAN PENELITIAN LAIN

Penelitian [5] membahas tentang frekuensi sampling dan aliasing dalam konteks laboratorium virtual, namun tidak dijelaskan *software* yang digunakan. Hasilnya menunjukkan bahwa frekuensi sampling yang lebih tinggi menghasilkan konversi sinyal *analog* ke *digital* yang lebih baik, dengan frekuensi minimum yang diperlukan untuk representasi sinyal setidaknya dua kali lipat frekuensi maksimum sinyal *analog*, sesuai dengan laju *Nyquist*. Aliasing terjadi ketika frekuensi sampling tidak cukup tinggi, menghasilkan sinyal bayangan yang menurunkan kualitas rekonstruksi sinyal asli. Penelitian ini mengajarkan konsep dasar pengambilan sampel yang benar dan salah kepada mahasiswa, serta pentingnya menggunakan filter anti-aliasing untuk menghindari aliasing dengan meredam frekuensi tinggi sebelum pengambilan sampel.

Penelitian [9] berfokus pada contoh pengambilan gambar objek berputar menggunakan kamera video dan membahas peran Teorema Sampling *Nyquist* dalam pengambilan data. Penulis, menjelaskan bahwa pemilihan waktu sampling yang tepat sangat penting untuk mencegah distorsi data. Dalam contoh kasus penelitian ini, penelitian ini menunjukkan bahwa frekuensi sinyal dapat dihitung dengan akurat jika waktu sampling dipilih sesuai dengan Teorema Sampling *Nyquist*. Jika waktu sampling tidak memenuhi syarat, fenomena aliasing terjadi, yang menyebabkan interpretasi sinyal yang salah. Tujuan dari penelitian ini adalah untuk meningkatkan pemahaman kita tentang ide- ide dasar tentang pengambilan sampel dan mengapa mematuhi teorema pengambilan sampel *Nyquist* sangat penting dalam proses pengambilan data.

Pada pendahuluan sudah disinggung perbedaan hasil penelitian ini dengan hasil penelitian lain, misalnya perbedaan pada objek penelitiannya, selain itu sistematika penulisan yang lebih teratur. Hal ini tergambar dalam tiga skenario utama. Pertama, dalam skenario I, penelitian membahas visualisasi sinyal terhadap frekuensi sampling (fs) yang berbeda. Tujuan utamanya adalah memastikan bahwa pembaca dapat memahami teorema sampling *Nyquist* dengan lebih baik sebelum memasuki pembahasan lebih lanjut. Selain itu, terdapat skenario kedua yang menyoroti aspek audio, di mana kompleksitasnya jauh lebih tinggi dibandingkan dengan hanya melakukan visualisasi sinyal. Pada skenario ini, ada tambahan suara dengan durasi yang bervariasi sesuai dengan penerapan frekuensi sampling yang berbeda.

Perkembangan terbaru menunjukkan bahwa selain memenuhi syarat *Nyquist*, diperlukan langkah tambahan berupa proses pemfilteran sinyal untuk mendukung pemenuhan syarat tersebut. Oleh karena itu, skenario III membahas proses penyaringan sinyal untuk mengurangi aliasing. Dengan menambahkan aspek kebaruan tersebut, diharapkan penelitian ini dapat menjadi kontribusi yang berharga dalam pemahaman dan pengembangan lebih lanjut di bidang ini.

4. KESIMPULAN

Penelitian ini menunjukkan bahwa pemilihan frekuensi sampling yang tepat dan penerapan filter anti-aliasing sangat penting untuk menjaga kualitas sinyal audio. Skenario pertama menunjukkan bahwa semakin tinggi frekuensi sampling, sinyal semakin bebas dari distorsi dan lebih menyerupai sinyal asli, sesuai dengan teorema Nyquist. Pada skenario kedua, frekuensi sampling sebesar 44100 Hz terbukti paling efektif dalam menghasilkan kualitas audio yang mendekati aslinya, sedangkan frekuensi sampling yang lebih rendah menyebabkan audio terdengar lebih lambat, dan frekuensi yang lebih tinggi membuat audio terdengar lebih cepat. Skenario ketiga menunjukkan bahwa Low-Pass Filter (FIR) dengan frekuensi cut-off 14 Hz efektif menghilangkan komponen frekuensi tinggi yang menyebabkan distorsi, sehingga menghasilkan sinyal yang lebih halus dan bebas aliasing. Dari ketiga skenario ini, kombinasi frekuensi sampling 44100 Hz adalah yang paling efektif untuk menghasilkan sinyal audio berkualitas tinggi yang mendekati aslinya, terutama bila didukung oleh penggunaan filter anti-aliasing yang sesuai.

5. REFERENSI

[1] I. L. Ramadhyagita, A. Annisa, F. Kamindra, and F. M. Rizky, "Kajian Discrete Fourier Transform untuk Menganalisis Sinyal Arbitrer," *Mitra Pilar: Jurnal Pendidikan, Inovasi, dan Terapan Teknologi*, vol. 1, no. 1, pp. 7–16, Jun. 2022, doi: 10.58797/pilar.0101.02.

[2] I. Febriana, "Simulasi Akuisisi Sinyal Suara Dengan Menggunakan MATLAB," *Seminar Nasional Fortei Regional 7*, vol. 4, no. 1, pp. 209–2013, Dec. 2021.

[3] T. Strohmer and J. Tanner, "Implementations of Shannon's sampling theorem, a time-frequency approach," *Sampling Theory in Signal and Image Processing*, vol. 4, no. 1, pp. 2–17, Jan. 2005, doi: 10.1007/BF03549421.

[4] J. L. Simancas-García and K. George-González, "Discrete sampling theorem to Shannon's sampling theorem using the hyperreal numbers R," *Revista de Matemática: Teoría y Aplicaciones.*, vol. 28, no. 2, pp. 163–182, Jul. 2021, doi: 10.15517/rmta.v28i2.43356.

[5] M. Bogdan, "Sampling rate and aliasing on a virtual laboratory," *Journal of Electrical and Electronics Engineering*, vol. 2, no. 2, pp. 121–124, Oktober 2009.

[6] M. Rembet, "Simulasi Pengaruh Jumlah Eksitasi Pada Filter Chebyshev Tipe I Terhadap Nilai Rata-Rata Magnitudo Sinyal Acak," *Jurnal Tekno Mesin*, vol. 8, no. (2), pp. 15–189, 2022, doi: https://ejournal.unsrat.ac.id/v3/index.php/jtmu/article/view/44705.

[7] J.-H. Kim, "Emulation of Anti-alias Filtering in Vision Based Motion Mmeasurement," *The Journal of Korea Robotics Society*, vol. 6, no. 1, pp. 18–26, Feb. 2011, doi: 10.7746/jkros.2011.6.1.018.

[8] R. J. Putra and M. Rif'an, "Implementasi Filter Digital Fir (Finite Impulse Response) Pada Field Programmable Gate Arrays (FPGA)," *Jurnal Mahasiswa Teknik Elektro Universitas Brawijaya*, vol. 1, no. 4, 2013.

[9] L. Lévesque, "Nyquist sampling theorem: understanding the illusion of a spinning wheel captured with a video camera," *Physics Education.*, vol. 49, no. 6, pp. 697–705, Nov. 2014, doi: 10.1088/0031-9120/49/6/697.

[10] H. Rakshit and M. A. Ullah, "An Efficient Approach for Designing FIR Low- pass Filter and Its Application," *International Conference on Electrical Information and Communication Technologies (EICT), Khulna, Bangladesh*, pp. 130–136, Desember 2015, doi: 10.1109/EICT.2015.7391934.

[11] A. A. Havis and L. Fitria, "Filtering Sinyal Menggunakan Band Pass Filter," *Jurnal SIFO Mikroskil*, vol. 19, no. 2, pp. 37–48, 2018, doi: https://doi.org/10.55601/jsm.v19i2.594.

No.	Kode	Deskripsi
1	% Parameter untuk sinyal pertama	Komentar yang menjelaskan bahwa baris-baris berikutnya berkaitan dengan pengaturan parameter untuk sinyal pertama.
2	Fs1 = 8; % Frekuensi sampling pertama	Menetapkan frekuensi sampling pertama (Fs1) sebesar 8 Hz.
3	t1 = (0:Fs1-1)/Fs1;	Menghitung vektor waktu (t1) dari 0 hingga 1 detik dengan langkah 1/Fs1.
4	% Proses normalisasi untuk sinyal pertama	Komentar yang menunjukkan bahwa baris-baris berikutnya berkaitan dengan proses pembuatan sinyal pertama.
5	s1 = sin(2*pi*t1*2);	Membuat sinyal pertama (s1) menggunakan fungsi sinus dengan frekuensi 2 Hz.

DOI: http://dx.doi.org/10.17509/xxxx.v6ix
p- ISSN xxxx-xxxx e- ISSN xxxx-xxxx

6	% Membuat figure	Komentar yang menunjukkan bahwa baris berikutnya akan berfungsi untuk mempersiapkan figure grafik.
7	figure;	Membuat figure baru untuk menampilkan plot.
8	% Plot sinyal pertama	Komentar yang menjelaskan bahwa bagian berikutnya akan menggambarkan sinyal pertama yang telah dibuat.
9	subplot(2,2,1)	Mengatur subplot untuk grafik pada posisi pertama dari grid 2x2.
10	stem(t1, s1, 'r', 'filled', 'MarkerFaceColor', 'r', 'LineWidth',1.5)	Menggambar plot diskret dari sinyal pertama (s1) terhadap waktu (t1).
11	axis([0 1 -1.2 1.2])	Mengatur batas sumbu X (0 hingga 1 detik) dan sumbu Y (-1.2 hingga 1.2).
12	title('Fs = 8 Hz', 'FontSize', 14, 'FontWeight', 'bold', 'Color', 'r')	Menambahkan judul pada plot dengan teks "Fs = 8 Hz".
13	xlabel('Waktu (detik)', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'k')	Menambahkan label pada sumbu X dengan teks "Waktu (detik)".
14	ylabel('Amplitudo', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'k')	Menambahkan label pada sumbu Y dengan teks "Amplitudo".
15	grid on	Mengaktifkan grid pada plot untuk meningkatkan keterbacaan grafik.