Kartunisasi Gambar Menggunakan Bahasa Pemrograman Python

Hilda Zakiatun Nufus
Fakultas Ilmu Komputer
Universitas Pembangunan Nasional Veteran Jakarta
Jakarta, Indonesia.
2110511146@mahasiswa.upnvj.ac.id

Amanda Najwa Perak Azizah
Fakultas Ilmu Komputer
Universitas Pembangunan Nasional Veteran Jakarta
Jakarta, Indonesia.
2110511158@mahasiswa.upnvj.ac.id

Leondra Herfino Ilham
Fakultas Ilmu Komputer
Universitas Pembangunan Nasional Veteran Jakarta
Jakarta, Indonesia.
2110511154@mahasiswa.upnvj.ac.id

Aruni Intan Haifa S
Fakultas Ilmu Komputer
Universitas Pembangunan Nasional Veteran Jakarta
Jakarta, Indonesia.
2110511161@mahasiswa.upnvj.ac.id

Abstrak - Proyek "Kartunisasi Gambar Python" bertujuan untuk menciptakan sebuah inovasi berbasis Python yang dapat mengubah foto menjadi representasi yang menyerupai kartun. Proyek ini menawarkan metode yang sederhana dan efektif untuk menghasilkan visual gaya kartun dari berbagai masukan dengan memanfaatkan teknik OpenCV dan algoritma pemrosesan gambar. e*dge detection*, *color simplification*, dan *image enhancement* adalah beberapa proses yang terlibat dalam proses ini. Sementara *penyederhanaan warna* memudarkan warna-warna yang berbeda untuk efek kartun, e*dge detection* mengisolasi bentuk dan karakteristik utama dari sebuah gambar. Teknik *image enhancement* digunakan untuk meningkatkan kualitas visualnya. *Library* Python populer seperti OpenCV dan NumPy digunakan dalam proyek ini. Hasil gambar yang sudah diubah menjadi kartun ini dapat digunakan untuk desain grafis, hiburan, dan ekspresi artistik. Proyek "Kartunisasi Gambar Python" menyediakan metode yang sederhana dan mudah diakses untuk mengubah gambar menjadi representasi bergaya kartun yang menarik secara visual.

Kata Kunci: Kartunisasi Gambar, Python, OpenCV.

Abstract *- In order to create representations of photos that resemble cartoons, the "Cartoonize Image Python" project is working to create a Python-based tool. The project offers a simple and effective method for producing cartoon-style visuals from any input by utilizing computer vision techniques and image processing algorithms. Edge identification, color simplification, and image improvement are just a few of the processes that are included in the process. While color simplification blurs distinct colors for a cartoonish effect, edge recognition isolates the key shapes and characteristics of an image. Techniques for improving images are used to boost their visual quality. Popular Python libraries like OpenCV and NumPy are used in the project. The resulting cartoonized images can be utilized for graphic design, amusement, and artistic expression. A simple and approachable method for converting images into visually appealing cartoon-style representations is provided by the "Cartoonize Image Python" project.*

Keywords: Cartoonize Image, Python, OpenCV.

I. PENDAHULUAN

Penggunaan teknologi modern telah mengakar dalam setiap aspek kehidupan kita. Kita hampir selalu menggunakan teknologi dalam pekerjaan kita untuk memenuhi kebutuhan masyarakat. Ilmu komputer adalah subjek yang menonjol saat ini. *Cloud computing*, *Internet of Things*, Kecerdasan Buatan, *Cyber Security*, dan aplikasi dunia nyata lainnya hanya beberapa contoh. Informasi dapat disimpan, diambil, dikirim, dan digunakan oleh pengguna berkat teknologi. Semua aplikasi yang melibatkan komputer sangat bergantung pada *Image Processing*. Banyak penggunaan praktis dari *Image Processing* dapat ditemukan dalam bidang *security*, perbankan, pendidikan, dan industri kereta api, dll [1].

Image processing nyatanya adalah sebuah metode menemukan suatu objek dalam sebuah gambar, menentukan dimensinya dan jumlahnya, serta melakukan perubahan padanya. Di era media dan komunikasi saat ini, *Image Processing* memiliki banyak fitur yang berbeda. Setiap fitur tersebut memprediksi bahwa gambar akan dihasilkan dengan kejelasan dan ketajaman yang lebih baik. Setiap gambar dievaluasi menggunakan grid yang berbeda. Seluruh gambar dapat dilihat sebagai matriks dua dimensi. Setiap sel digunakan untuk menyimpan nilai piksel yang berbeda untuk setiap komponen gambar. Salah satu bentuk dari Image Processing yang banyak digandrungi kalangan anak-anak, remaja, hingga orang dewasa adalah kartun [2]. Kartun adalah gambar-gambar karakter yang beberapa diantaranya dibuat menggunakan imajinasi dan kreativitas seorang *creator* atau bisa juga berdasarkan orang sungguhan. Kartun bersifat semi-realistik atau non-realistik, yang sering digunakan saat ini untuk menggambarkan secara satir dan humoristik suatu skenario atau peristiwa. Sebelumnya, kartunis akan menggambar kartun ini secara manual; namun, seiring dengan popularitas "Anime", menjadi lebih sulit bagi mereka untuk menggambar setiap *frame* film secara manual, yang

membutuhkan banyak waktu dan tidak dapat dibatalkan jika terjadi kesalahan [3].

Dengan adanya teknologi modern yang semakin berkembang berdampak pada industri animasi kartun yang semakin merajalela dan pembuatannya tidaklah lagi sesulit sebelum teknologi berkembang. Dengan menggunakan bahasa pemrograman Python dan Library OpenCv yang tepat, kita dapat membuat sebuah gambar yang bertransformasi menjadi sebuah kartun [3]. Filter-filter tersebut dirancang untuk menghasilkan hasil pada berbagai gambar yang artistik dan lucu dengan menggunakan Bahasa Pemrograman Python dan beberapa *Library* didalamnya [3]. Untuk menciptakan efek kartun pada sebuah gambar dalam bahasa pemrograman Python kita dapat menggunakan berbagai teknik dan filter. Metode pendekatan baik dari filter hingga teknik yang kuat tersebut memungkinkan penciptaan representasi kartun yang menarik secara visual dan bergaya dari gambar-gambar umum. Selain itu, Python juga memiliki kerangka kerja yang fleksibel dan ramah bagi para pengguna untuk mempelajari dan menerapkan teknik-teknik guna menghasilkan efek kartun yang menarik pada gambar.

Maka kita bisa merumuskan beberapa permasalahan dari pembahasan yang akan kita bahas, pembahasan pertama kita akan fokus pada pembahasan teknik dan filter yang digunakan untuk memberikan efek nuansa kartun pada sebuah gambar. Selain mempelajari dan membahas teknik yang digunakan, sangat penting untuk memahami kerja internal dan urutan langkah yang terlibat dalam proses kartunisasi tersebut untuk memvisualisasikan alur logika dari program algoritma kartunisasi. Kita dapat lebih memahami bagaimana banyak elemen dan langkah-langkah dari algoritma tersebut saling terhubung dan berkontribusi pada topik yang dibahas. Untuk mewakili bagaimana data dan proses alur algoritmanya berjalan, berbagai pendekatan visualisasi dapat digunakan, seperti flowchart, diagram, atau grafik. Representasi visual ini menawarkan peta jalan visual yang memudahkan pemahaman struktur program, mendorong kerjasama, komunikasi tim, dll. Visualisasi alur logika tersebut dapat meningkatkan transparansi, efisiensi, dan efektivitas program algoritma kartunisasi, membantu dalam analisis, optimalisasi, dan pengembangan yang dilakukan secara kolaboratif. Oleh karena itu pembahasan yang kedua yaitu mengenai flowchart algoritma dari proses kartunisasi menggunakan bahasa Python.

Untuk mendapatkan hasil yang lebih baik, proses kartunisasi tersebut harus dioptimalkan. Efisiensi dan efektivitas proses kartunisasi dapat ditingkatkan dengan menggunakan pendekatan optimisasi yang efisien. Dengan menekankan pada optimisasi, memungkinkan untuk menghasilkan visualisasi kartun yang menarik dan luar biasa yang memenuhi atau melebihi harapan pengguna. Oleh karena itu pembahasan terakhir yaitu mengenai cara optimalisasi proses kartunisasi untuk kinerja dan hasil yang baik.

Tujuan utama dari jurnal ini adalah untuk menyajikan panduan lengkap, langkah-demi-langkah tentang cara menggunakan OpenCV untuk membuat algoritma kartunisasi pada sebuah gambar. Bagi dari para akademisi, *developer*, dan masyarakat yang tertarik dalam bidang pengolahan citra dan visi komputer, jurnal ini bertujuan untuk menjadi sumber daya yang komprehensif dan mudah untuk di akses. Petunjuk yang disajikan akan memungkinkan para pembaca dan pengguna untuk secara efektif menerapkan algoritma kartunisasi pada foto mereka sendiri dan memahami prosedur implementasi secara mendalam.

Dengan adanya jurnal ini memberikan para pembaca alat yang mereka butuhkan untuk sepenuhnya memanfaatkan kemampuan OpenCV dan menghasilkan efek kartun yang menarik dalam proyek pengolahan citra mereka, menyediakan wawasan dan pengetahuan yang dapat mengembangkan teknik kartunisasi dan penggunaannya dalam berbagai bidang, menjadi sumber daya berharga bagi peneliti, praktisi, dll. Informasi yang disediakan dalam jurnal ini akan turut memungkinkan para pembaca untuk meningkatkan pengetahuan mereka tentang kartunisasi dan berkontribusi pada perkembangan bidang yang lebih umum dalam analisis dan manipulasi citra. Selain itu, para mereka juga akan mendapatkan pemahaman mendalam tentang ide-ide dasar dan prosedur yang mendasari kartunisasi dengan menganalisis kelebihan, kelemahan, dan fitur unik dari setiap metode.

Dan yang terakhir, melalui evaluasi dan analisis yang sistematis, pembaca akan memperoleh wawasan mengenai variasi hasil kartunisasi yang dihasilkan oleh berbagai pendekatan. Analisis perbandingan ini akan memberikan pemahaman mengenai kelebihan, kelemahan, dan karakteristik khas dari setiap metode, sehingga para peneliti, praktisi, dan penggemar dapat membuat keputusan yang lebih baik dalam memilih dan menggunakan teknik-teknik tertentu. Pada akhirnya, jurnal ini bertujuan untuk menciptakan pemahaman yang lebih baik mengenai efektivitas relatif dan hasil visual yang diperoleh melalui berbagai metodologi dan prosedur, yang akan membantu dalam kemajuan algoritma kartunisasi.

II. TINJAUAN PUSTAKA

A. Kartunisasi Gambar

Kartunisasi gambar adalah proses mengubah gambar asli menjadi versi yang terlihat seperti gambar kartun. Tujuan dari kartunisasi adalah menciptakan efek visual yang mirip dengan gaya kartun, dengan ciri khas seperti tepi yang kuat, warna datar, dan gaya yang lebih sederhana. Proses kartunisasi gambar dapat dilakukan secara manual oleh seorang seniman atau melalui pengolahan komputer menggunakan algoritma dan teknik pemrosesan gambar[4].

Gambar 1. *Before - After* implementasi efek kartunisasi yang dilakukan.

Menanggapi gambar kartun muncul di film animasi "Namamu". Namun, secara manual membuat ulang dunia nyata Adegan dalam gaya kartun sangat melelahkan dan melibatkan keterampilan artistik yang substansial. Untuk mendapatkan kartun berkualitas tinggi, seniman harus menggambar setiap garis dan menaungi setiap warna wilayah adegan sasaran. Sedangkan editing gambar sudah ada perangkat lunak/algoritma dengan fitur standar tidak dapat menghasilkan hasil yang memuaskan untuk kartunisasi. Oleh karena itu, khusus teknik yang dirancang yang dapat secara otomatis mengubah foto dunia nyata menjadi gambar gaya kartun berkualitas tinggi sangat sangat membantu dan bagi para seniman, waktu yang sangat banyak disimpan sehingga mereka dapat fokus pada pekerjaan yang lebih kreatif. Seperti alat juga memberikan tambahan yang berguna untuk perangkat lunak pengedit foto seperti Instagram dan Photoshop [5].

B. OpenCV

OpenCV (Open Source Computer Vision) adalah pustaka pemrosesan gambar dan penglihatan komputer open source yang dirancang untuk digunakan dalam aplikasi pengolahan gambar dan pengenalan pola. Pustaka ini menyediakan berbagai fungsi dan algoritma yang dapat digunakan untuk memanipulasi, menganalisis, dan memahami gambar dan video. Open Computer Vision adalah sebuah API (Application Programming Interface) *library* yang sudah sangat familiar pada Pengolahan Citra Computer Vision [6]. Computer Vision itu sendiri adalah salah satu cabang dari Bidang Ilmu Pengolahan Citra (*Image Processing*) yang memungkinkan komputer dapat melihat seperti manusia. Dengan vision tersebut komputer dapat mengambil keputusan, melakukan aksi, dan mengenali terhadap suatu objek [7].

` Gambar 2. Mendeteksi suatu objek.

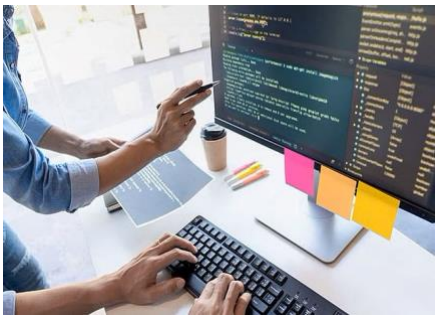
Beberapa pengimplementasian dari Computer Vision adalah *Face Recognition, Face Detection, Face/Object Tracking, Road Tracking*, dll. OpenCV adalah *library* Open Source untuk Computer Vision untuk C/C++, OpenCV didesain untuk aplikasi *real-time*, memiliki fungsi-fungsi akuisisi yang baik untuk image/video. OpenCV sendiri terdiri dari 5 *library*, yaitu: CV (untuk algoritma Image processing dan Vision); ML (untuk *machine learning library*); Highgui (untuk GUI, Image dan Video I/O); dan CXCORE (untuk struktur data, mendukung XML dan fungsi-fungsi grafis) [8].

OpenCV memungkinkan komputer dapat melihat seperti manusia dengan *vision* tersebut. Komputer dapat mengambil keputusan, melakukan aksi dan mengenali terhadap suatu objek berdasarkan deteksi wajah. OpenCV terdiri dari 5 *library* yaitu, Computer Vision(CV) sebagai *algoritma image processing* dan *vision*-nya, Machine Learning(ML), HighGUI sebagai GUI, Image dan Video I/O,CXCORE sebagai struktur data, support XML dan fungsi-fungsi grafis dan CvAux sebagai penolong OpenCV [9].

C. Python

Python, sebuah bahasa pemrograman tingkat tinggi yang kuat dan berorientasi objek, telah menjadi populer di lingkungan akademik maupun praktis. Dikembangkan oleh Guido van Rossum, Python menawarkan berbagai fitur dan properti yang membuatnya menjadi pilihan menarik bagi para programmer [10]. Pada awal tahun 1990, Python telah menjadi salah satu bahasa pemrograman yang populer dan digunakan secara luas dalam berbagai bidang, termasuk pengembangan perangkat lunak, pemrosesan data, kecerdasan buatan, dan pengembangan web.

Python hanyalah satu dari banyak bahasa pemrograman. Sama seperti bahasa manusia, ada banyak perbedaan bahasa komputer, seperti Java, LISP, PHP, dan Perl dan... jangan lupakan C atau lainnya, serta hal-hal berguna seperti skrip UNIX. Sebagian besar bahasa bagus setidaknya untuk satu hal – misalnya, menulis program yang mudah dibawa-bawa adalah poin kuat untuk Java dan mengakses database dan menggabungkannya ke dalam halaman web adalah spesialisasi untuk PHP. Namun di baliknya, semua bahasa ini pada intinya sangat mirip tingkat konsep – sebagian besar memiliki data dalam variabel dan fungsi (prosedur, metode) untuk melakukan sesuatu pada data tersebut.



Gambar 3. Bahasa Pemrograman.

Beberapa bahasa bahkan menggabungkan data dan fungsi ke dalam bundel yang disebut objek, dan yang lainnya seperti LISP biarkan Anda memperlakukan fungsi seperti variabel, dan sebaliknya. Python adalah bahasa pemrograman yang kuat dan elegan yang mudah dibaca dan dipahami. Ini menunjukkan sebagian besar fitur ini umum untuk banyak lainnya bahasa dan berguna untuk aplikasi dunia nyata, untuk boot! Ini juga perangkat lunak bebas, memiliki satu standar implementasi, dan komunitas peretas yang besar dan ramah di sekitarnya. Setelah Anda mempelajari Python, setiap bahasa lain yang ingin Anda pelajari tampaknya cukup familiar.

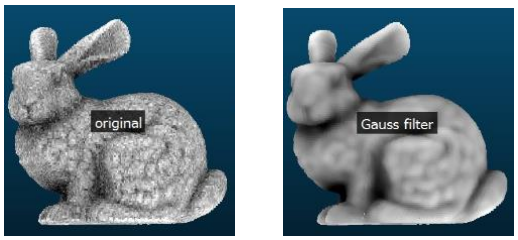
III. METODE PENELITIAN

Metode penelitian yang digunakan dalam proyek ini adalah menggunakan beberapa *library* Python yaitu OpenCV dan Numpy didalamnya.

OpenCV menyediakan koleksi lengkap fungsi, metode, dan alat yang memungkinkan para pemrogram melakukan berbagai kegiatan terkait visi komputer, termasuk *Image Processing*, pengenalan objek, pengenalan wajah, Machine Learning, dll [11]. Sedangkan, NumPy suatu *library* utama Python untuk pemrograman array disebut NumPy yang memudahkan penyimpanan dan pengolahan data multidimensional homogen. *Library* ini menyediakan fungsi-fungsi untuk operasi matematika, operasi logika, manipulasi bentuk, pengurutan, dan statistik, sehingga memungkinkan komputasi yang lebih cepat dan efisien tanpa perlu melakukan perulangan secara eksplisit [12].

Berbagai fungsi dari OpenCV dan NumPy yang digunakan dalam proyek ini, yaitu:

1) *Gaussian blur cv2.GaussianBlur*(): Teknik ini mengaburkan gambar dengan menerapkan filter Gaussian, yang mengurangi detail dan meratakan noise. Filter Gaussian adalah filter linier dengan nilai pembobotan untuk setiap anggotanya dipilih berdasarkan bentuk fungsi Gaussian. Sama halnya dengan filter Gaussian, filter mean adalah filter linear yang bekerja dengan menggantikan intensitas nilai pixel dengan rata-rata dari nilai pixel tersebut dengan nilai pixel-pixel tetangganya.



Gambar 4. *Before - After* implementasi Gaussian Blur

Filter ini sebagai dasar yang optimal dari perspektif mencakup rentang Gaussian tertentu fungsi penyebaran titik, dan dihitung menggunakan analisis komponen utama. Sebagai jumlah fungsi basis yang digunakan meningkat, kekaburan yang dihasilkan menyatu dengan cepat menjadi benar kekaburan Gaussian varian-ruang [13].

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gambar 5. Rumus Gaussian Blur

2) *Median blur*: Dengan menerapkan filter median pada gambar, fungsi ini secara efektif mengurangi noise tiba-tiba sambil tetap mempertahankan tepi gambar.

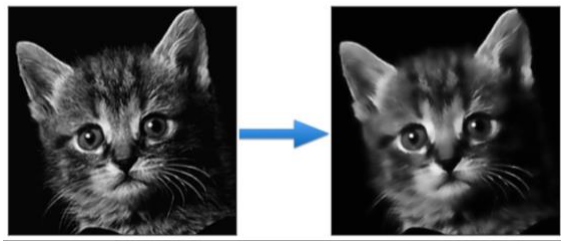
Gambar 6. *Before - After* implementasi Median Blur

Untuk mengukur kinerja TSM yang diusulkan secara kuantitatif skema penyaringan versus filter median lainnya, kesalahan kuadrat rata-rata (MSE) kriteria digunakan sebagai berikut [14]:

$$\mathrm{MSE} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (S_{ij} - Y_{ij})^2$$

Gambar 7. Rumus Median Blur

3) *Bilateral filter ('cv2.bilateralFilter()')*: Filter ini menerapkan filter bilateral pada gambar, mempertahankan tepi sambil mengurangi noise.



Gambar 8. *Before - After* implementasi Bilateral Filter

Filter bilateral sebagai pelindung tepi yang terkenal alat penghalus gambar. Ide kunci dari filter bilateral terdiri dalam memasukkan bobot fotometrik ke dalam filter Gaussian standar. Efek dari bobot ini adalah membatalkan interaksi spasial antar piksel dengan intensitas penting perbedaan [15].

$$BF[I]_{\mathrm{p}} = \frac{1}{W_{\mathrm{p}}} \sum_{\mathrm{q} \in \mathcal{S}} G_{\sigma_s}(\|\mathrm{p} - \mathrm{q}\|)\, G_{\sigma_r}(I_{\mathrm{p}} - I_{\mathrm{q}})\, I_{\mathrm{q}}$$

Gambar 9. Rumus Bilateral Filter

4) *Laplacian filter ('cv2.Laplacian()')*: Menggunakan operator Laplacian untuk menemukan tepi pada gambar.

Gambar 10. *Before - After* implementasi Laplacian Filter

Pemfilteran Laplacian memanipulasi kontras dari sinyal input menggunakan fungsi remap dan kemudian menghasilkan sinyal detail. Pemfilteran Laplacian meningkatkan kontras gambar dan membangun piramida Laplacian untuk setiap piksel; dengan demikian, Filter Laplacian bisa kurangi lingkaran cahaya dengan gambar yang lebih jelas daripada multi-skala lainnya metode. Adaptasi parameter untuk fungsi remap lebih lanjut meningkatkan kualitas [16].

Laplacian bekerja dengan menghitung selisih intensitas piksel antara piksel pusat dan piksel sekitarnya. Hasilnya adalah gambar yang menonjolkan tepi dan perubahan tajam

dalam citra. Laplacian filter sering digunakan dalam aplikasi pengolahan citra seperti segmentasi gambar, deteksi objek, dan ekstraksi fitur.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$
$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

Gambar 11. Rumus Laplacian Filter

5) *Konversi warna ('cv2.cvtColor()')*: Dengan menggunakan flag 'cv2.COLOR_BGR2GRAY',

Skrip ini mengubah gambar menjadi skala abu-abu dengan cara mengambil rata-rata intensitas komponen warna (B, G, R) dari setiap piksel gambar dan menggantinya dengan nilai intensitas keabuan tunggal. Hasilnya adalah gambar yang memiliki saluran warna tunggal yang mewakili tingkat keabuan pada setiap piksel, tanpa informasi warna.

Untuk mengubah citra berwarna yang mempunyai nilai matrik masing- masing r, g dan b menjadi citra gray scale dengan nilai s, maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai r, g dan b sehingga dapat dituliskan menjadi:

$$s = \frac{r + g + b}{3}$$

Gambar 12. Rumus Konversi Warna

6) *Thresholding ('cv2.threshold()')*: Dengan menggunakan nilai ambang, operasi ini mengubah gambar menjadi gambar biner. Piksel putih diatur di atas ambang batas, dan piksel hitam diatur di bawahnya. Hal ini merupakan salah satu metode segmentasi citra di mana prosesnya didasarkan pada perbedaan derajat keabuan citra.

Penggunaan thresholding sebagai alat dalam segmentasi citra telah dipelajari secara ekstensif, dan berbagai teknik telah diajukan untuk pemilihan threshold otomatis. Ulasan teknik ini disajikan dalam Lampiran [17].

$$g(x, y) = \begin{cases} 1, & jika\ f(x, y) \geq T \\ 0, & jika\ f(x, y) < T \end{cases}$$

Gambar 13. Rumus Thresholding

7) *Inversi*: Skrip ini membalik warna hitam dan putih pada gambar hasil threshold (cv2.subtract()).

Tujuan dari proses inversi ini mungkin untuk mencapai efek visual tertentu atau mempersiapkan gambar untuk langkah pengolahan berikutnya.

8) *Reshaping dan konversi gambar*: Gambar diubah menjadi representasi floating-point ('np.float32') dan diubah bentuk menjadi array 2D.

9) *K-means clustering ('cv2.kmeans()')*: Mengelompokkan warna yang serupa dalam gambar yang telah diubah.

$$d_{Euclidean}(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$$

Gambar 14. Rumus K-means Clustering

10) *Reduksi warna*: Dengan memisahkan nilai piksel dan menghasilkan nilai warna yang terkelompok, skrip ini mengurangi jumlah warna dalam gambar asli.

Dengan mengurangi jumlah warna dalam gambar, reduksi warna dapat memberikan efek kartun atau ilustrasi yang lebih stilistik. Ini juga dapat membantu dalam pengurangan ukuran file gambar dan mengoptimalkan penggunaan memori.

11) *Gambar bilateral*: Telah diubah dari skala abu-abu menjadi RGB menggunakan fungsi "cv2.cvtColor()".

Dengan menggunakan fungsi "cv2.cvtColor()", gambar bilateral yang awalnya dalam format skala abu-abu diubah menjadi format RGB yang terdiri dari tiga saluran warna: merah (R), hijau (G), dan biru (B).

12) *Operasi bitwise (cv2.bitwise_and())*: Skrip ini menggunakan operasi bitwise AND untuk menggabungkan gambar bilateral yang telah dikonversi dan gambar berkelompok. Maka langkah-langkahnya secara lebih detail ialah:

a) *Gambar bilateral yang telah dikonversi*: Biasanya, gambar input dihaluskan terlebih dahulu menggunakan bilateral filter untuk mengurangi noise atau detail yang tidak diinginkan. Setelah itu, gambar hasil bilateral filter dikonversi menjadi representasi biner.

b) *Gambar berkelompok*: Gambar berkelompok ini juga harus dikonversi menjadi representasi biner.

c) *Operasi bitwise AND*: Setelah kedua gambar dikonversi menjadi representasi biner, operasi bitwise AND dilakukan pada dua gambar tersebut menggunakan fungsi cv2.bitwise_and().

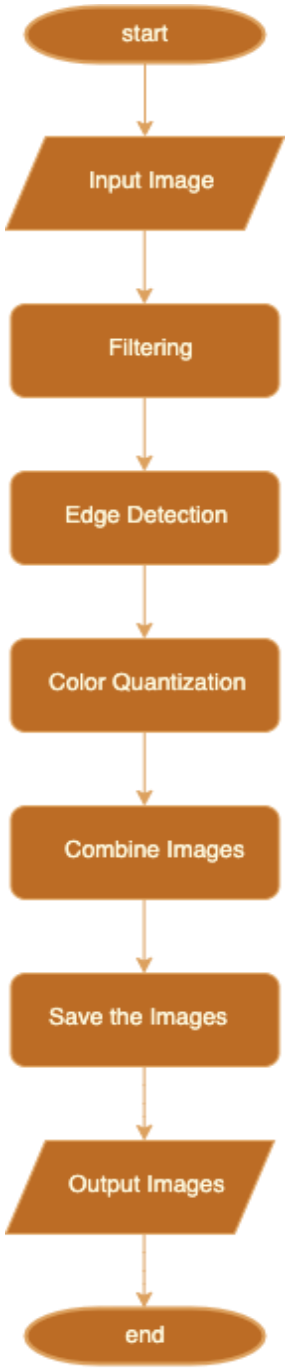
d) *Hasil gabungan*: Pada hasil gabungan ini, hanya piksel-piksel di mana kedua gambar tersebut memiliki piksel yang hidup (nilai 1) yang akan dinyatakan sebagai piksel yang hidup pada gambar output.

13) *Penyimpanan gambar ('cv2.imwrite()')*: Gambar kartun yang dihasilkan disimpan sebagai 'CartoonImage.png'.

14) *Tampilan gambar ('cv2.imshow()'):* Metode 'cv2.imshow()' digunakan untuk menampilkan gambar output yang dihasilkan.

15) *Penanganan input pengguna*: Untuk menutup gambar yang ditampilkan, skrip ini menunggu penekanan tombol ('cv2.waitKey(0)').

16) Langkah terakhir adalah menggunakan 'cv2.destroyAllWindows()' untuk menutup semua jendela yang terbuka.



Gambar 15. Flowchart Algoritma

IV. HASIL DAN PEMBAHASAN

A. *Hasil*

Inovasi yang akan kami kembangkan salah satunya adalah mengubah foto di dunia nyata menjadi gambar bergaya kartun berkualitas tinggi. Oleh karena itu, disini kami menggunakan beberapa filter untuk hasil yang maksimal. Disini kami juga akan berinovasi menggunakan efek filter yang lebih kreatif dengan cara menambahkan variasi efek filter dengan menggabungkan filter berbeda dan menciptakan filter khusus seperti efek pensil, efek airbrush, atau efek cat air yang unik. Dibawah ini ialah uraian percobaan kami:

1) Objek 1: UPNVJ

Testing pertama dilakukan dengan menggunakan objek satu yaitu gambar gedung UPNVJ. Gambar ini memiliki banyak objek, namun komposisi bangunan dan perspektif menciptakan horizon dan garis-garis panduan yang membuat gambar terlihat sangat mencolok dan dinamis. Sejauh ini, proses kartunisasi bekerja dengan baik untuk foto gedung upn karena objek-objek beton sangat cocok dengan filter-filter yang digunakan, yang membuat hasilnya terlihat lebih bersih dan lebih seperti kartun.

Foto Original:

Gambar 16:.Objek 1 Gambar Original UPNVJ

Foto setelah dilakukan proses kartunisasi:

Gambar 17. Hasil Gambar dengan Fungsi Img_bins

Gambar 18. Hasil Gambar Fungsi K-Means

Gambar 19. Hasil Gambar Fungsi Inverted Bilateral

Gambar 20. Hasil Gambar Kartunisasi Bilateral

2) Objek 2: Wanita *Square*

Testing kedua yaitu dengan menggunakan objek gambar wanita yang berposisikan *square*. Disini kami menggunakan foto seseorang yang sudah menggunakan riasan wajah. Hal tersebut akan menantang sistem untuk bekerja lebih bagus dan lebih detail dikarenakan riasan wajah tersebut. Di gambar ini juga ada area yang terang dan gelap. Sejauh ini, proses kartunisasi telah berjalan dengan baik.

Foto Original:

Gambar 21. Objek 2 Gambar Original Wanita

Foto setelah dilakukan proses kartunisasi:

Gambar 22. Hasil Gambar dengan Fungsi Img_bins

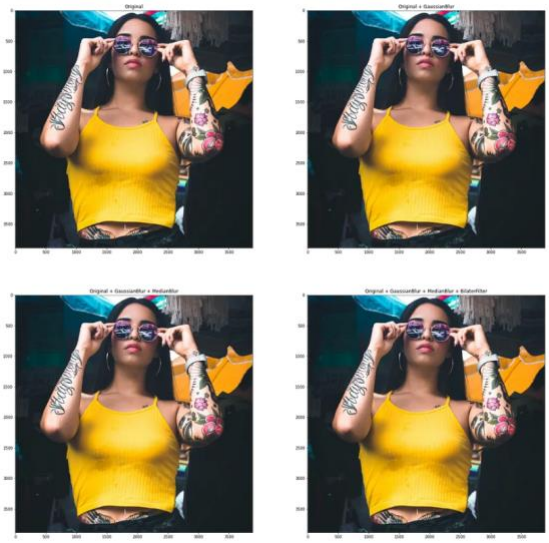
Gambar 23. Hasil Gambar Fungsi K-Means

Gambar 24. Hasil Gambar Fungsi Inverted Bilateral

Gambar 25. Hasil Gambar Kartunisasi Bilateral

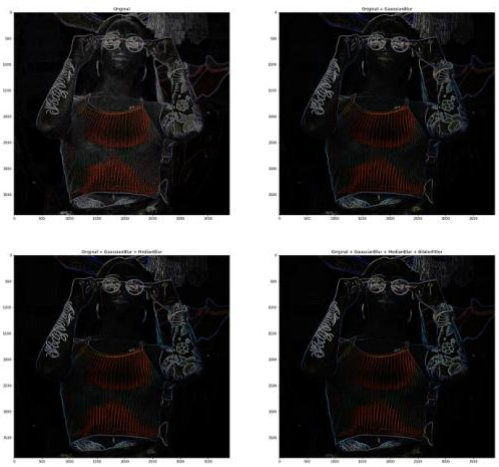
B. Pembahasan

1) Filtering: Disini terlihat tidak ada perbedaan antar gambar, kita bisa melihat perbedaannya jika kita tahu apa yang dilakukan oleh masing-masing filter. Filter Laplacian akan mendeteksi perbedaannya.



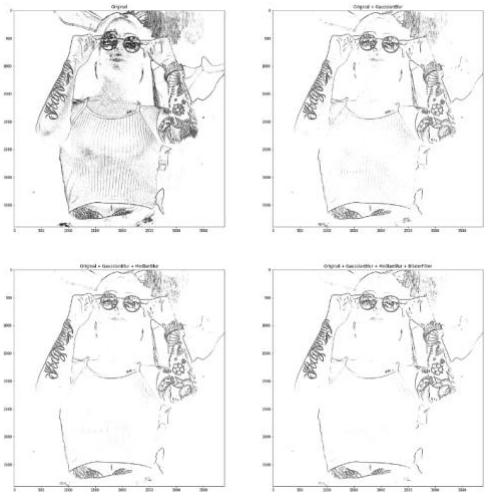
Gambar 26. Hasil Gambar Filtering

2) Edge Detection: Menerapkan filter laplacian untuk melihat perbedaannya, kita akan menerapkan filter pada keempat gambar yang berbeda dan memvisualisasikan hasilnya.



Gambar 27. Hasil Gambar Edge Detection

Sekarang, dapat terlihat bahwa laplacian dari gambar asli mendeteksi banyak noise. Mari kita bandingkan lagi gambarnya setelah kita invert warna hitam dan putih



Gambar 28. Hasil Gambar Setelah di Invert

3) Color Quantization: Sekarang kita memiliki gambar untuk masking, kita perlu mengurangi jumlah warna untuk gambar kartun kita. Kita dapat melakukannya dengan dua cara, menggunakan algoritma K-means atau dengan pembagian lantai sederhana. Algoritma K-means ditujukan untuk analisis data eksplanatif, tetapi sangat berguna dalam kasus ini. Dibawah ini terlihat perbedaanya.

Gambar 29. Hasil Gambar Color Quantization

4) Combine the images: Sekarang gabungkan kedua gambar tersebut dan simpan hasilnya.

Gambar 30.Hasil Gabungan Gambar

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Dalam jurnal ini. kami membuat kartunisasi gambar yang mengubah foto dunia nyata menjadi gambar bergaya kartun berkualitas tinggi. Industri animasi tidak akan ada berhentinya, dan standarnya semakin tinggi dari hari ke hari, sehingga sistem ini menyediakan ruang untuk penambahan fitur dan mudah disesuaikan dengan kode sumber lain dan menggunakan beberapa filter untuk hasil yang maksimal. Terdapat beberapa kombinasi filter yang digunakan untuk melakukan proses kartunisasi:

1. Gaussian blur
2. Median blur
3. Bilateral filter
4. Laplacian filter
5. Konversi warna
6. Thresholding
7. Inversi
8. Reshaping
9. K-means clustering
10. Reduksi warna
11. Operasi bitwise

Filter-filter tersebut apabila digunakan secara bersama-sama satu sama lain. Menerapkan filter tunggal dengan fitur yang terbatas akan memberikan hasil yang baik, namun tidak dapat diterima. Oleh karena itu, proses kartunisasi tersebut untuk mengoptimalkan kinerja proses kartunisasi dan menghasilkan hasil yang lebih baik.

Dengan memahami teknik-teknik atau filter yang dapat digunakan untuk menghasilkan efek kartunisasi, serta cara memvisualisasikan algoritma program kartunisasi, dapat mengembangkan solusi yang efektif dan berkualitas tinggi dalam kartunisasi gambar menggunakan OpenCV dalam bahasa pemrograman Python.

B. *Saran*

Saran yang ingin diberikan terhadap jurnal ini adalah sebuah narasi analisis perbandingan *output* antara satu objek dengan objek lainnya yang ada. Analisis komparatif ini memungkinkan pemahaman yang lebih mendalam tentang kelebihan dan kekurangan dari hasil yang dihasilkan oleh objek 1 dan objek lainnya. Mengidentifikasi keunggulan dan kekurangan dari satu objek dengan objek lainnya merupakan sebuah cara efektif untuk membuka jalan untuk perbaikan dan penyempurnaan lebih lanjut. Dan yang terakhir, narasi analisis perbandingan juga dapat memudahkan dalam menentukan strategi atau metode baru yang dapat bekerja secara harmonis untuk menghasilkan hasil kartunisasi gambar yang lebih baik di masa depan.

REFERENCES

[1] V. Sudarshan and A. Singh, "Cartooning an Image using OpenCV - Python - GeeksforGeeks," vol. 04, no. 02, pp. 55–58, 2020.

[2] S. Mohammad, B. Pranitha, S. G. Pandula, and P. T. Sree, "Object Detection with Voice Sensor and Cartoonizing the Image," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 10, no. 4, pp. 2762–2767, 2021, doi: 10.30534/ijatcse/2021/171042021.

[3] N. Nautiyal, V. Sinha, and S. Kaur, "Image Cartoonization," *Int. J. Mod. Trends Sci. Technol.*, vol. 7, no. 05, pp. 63–71, 2021, doi: 10.46501/ijmtst0705010.

[4] V. K. Singh, "A system for cartoonifying an image using python," no. January, 2023.

[5] Y. Chen, Y.-K. Lai, and Y.-J. Liu, "CartoonGAN: Generative Adversarial Networks for Photo Cartoonization," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9465–9474, doi: 10.1109/CVPR.2018.00986.

[6] J. Minichino and J. T. A.-T. T.- Howse, "Learning OpenCV 3 computer vision with Python : unleash the power of computer vision with Python using OpenCV." Packt Publishing, Birmingham, UK, 2015, doi: LK - https://worldcat.org/title/922700052.

[7] O. Priawadi, "OpenCV," *www.priawadi.com*, 2012. https://www.priawadi.com/2012/09/opencv.html (accessed Jun. 12, 2023).

[8] Sutarti, S. Samsuni, and I. Asseghaf, "Sistem Keamanan Rumah melalui Pengenalan Wajah Menggunakan Webcam dan Library Opencv Berbasis Raspberry Pi," *J. Din. Inform.*, vol. 8, no. 2, pp. 13–26, 2019, [Online]. Available: https://jdi.upy.ac.id/index.php/jdi/article/view/37.

[9] R. Isum, S. Maryati, and B. Tryatmojo, "Raden Isum Suryani Maryati Akurasi Sistem Face Recognition Akurasi Sistem Face Recognition OpenCV Menggunakan Raspberry Pi Dengan Metode Haar Cascade KATA KUNCI Akurasi Face Recognition Raspberry Pi OpenCV Haar Cascade," no. Cv, p. 12790, 2019.

[10] K. R. Srinath, "Python-The Fastest Growing Programming Language," *Int. Res. J. Eng. Technol.*, vol. 4, no. 12, pp. 354–357, 2017, [Online]. Available: www.irjet.net.

[11] N. Mahamkali and V. Ayyasamy, *OpenCV for Computer Vision Applications*. 2015.

[12] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020, doi: 10.1038/s41586-020-2649-2.

[13] T. Popkin, A. Cavallaro, and D. Hands, "Accurate and Efficient Method for Smoothly Space-Variant Gaussian Blurring," *IEEE Trans. Image Process.*, vol. 19, no. 5, pp. 1362–1370, 2010, doi: 10.1109/TIP.2010.2041400.

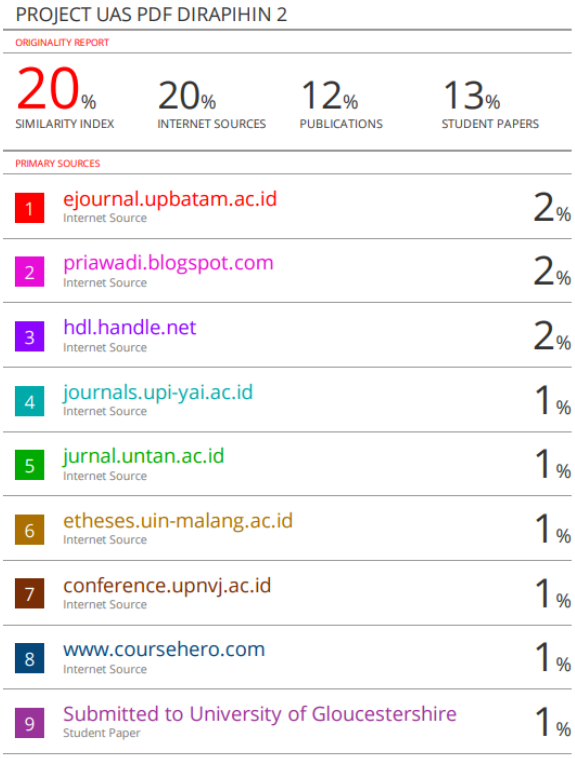
[14] T. Chen, K.-K. Ma, and L.-H. Chen, "Tri-state median filter for image denoising," *IEEE Trans. Image Process.*, vol. 8, no. 12, pp. 1834–1838, 1999, doi: 10.1109/83.806630.

[15] B.-H. Chen, Y.-S. Tseng, and J.-L. Yin, "Gaussian-Adaptive Bilateral Filter," *IEEE Signal Process. Lett.*, vol. 27, pp. 1670–1674, 2020, doi: 10.1109/LSP.2020.3024990.

[16] Y. Sumiya, T. Otsuka, Y. Maeda, and N. Fukushima, "Gaussian Fourier Pyramid for Local Laplacian Filter," *IEEE Signal Process. Lett.*, vol. 29, pp. 11–15, 2022, doi: 10.1109/LSP.2021.3121198.

[17] J. S. Weszka and A. Rosenfeld, "Threshold Evaluation Techniques," *IEEE Trans. Syst. Man. Cybern.*, vol. 8, no. 8, pp. 622–629, 1978, doi: 10.1109/TSMC.1978.4310038.

LAMPIRAN



PROJECT UAS PDF DIRAPIHIN 2

ORIGINALITY REPORT

20%	20%	12%	13%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ejournal.upbatam.ac.id Internet Source	2%
2	priawadi.blogspot.com Internet Source	2%
3	hdl.handle.net Internet Source	2%
4	journals.upi-yai.ac.id Internet Source	1%
5	jurnal.untan.ac.id Internet Source	1%
6	etheses.uin-malang.ac.id Internet Source	1%
7	conference.upnvj.ac.id Internet Source	1%
8	www.coursehero.com Internet Source	1%
9	Submitted to University of Gloucestershire Student Paper	1%

10	www.webology.org Internet Source	1 %
11	Submitted to Southern New Hampshire University - Continuing Education Student Paper	1 %
12	Shuang Huang, Xu Cao, Fengyun Li, Ziwei Zhao. "Research on Dynamic Water Level Recognition of Cabin Based on Improved Retinex Algorithm", The 6th International Conference on Computer Science and Application Engineering, 2022 Publication	1 %
13	www.koreascience.or.kr Internet Source	1 %
14	jdi.upy.ac.id Internet Source	<1 %
15	openlibrarypublications.telkomuniversity.ac.id Internet Source	<1 %
16	www.warse.org Internet Source	<1 %
17	Harlow, Charles A., Mohan M. Trivedi, and Richard W. Conners. "", Applications of Artificial Intelligence II, 1985. Publication	<1 %
18	dblp.uni-trier.de Internet Source	<1 %
19	Shilpa Singhal, Sudarshan Poudel, Sarika Karki, Rejina Ghimire. "Alpha Gadget: A Comprehensive Review of Technologies for Improving Business Involvement and participation in E-Commerce", 2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS), 2023 Publication	<1 %
20	libraryeproceeding.telkomuniversity.ac.id Internet Source	<1 %
21	Disha Singh, Kulsoom Masood, Nabeel Jamshed, Yahya Farooq, Yusuf Hasan, Huzaif Ahmad. "Design and Implementation of Autonomous Underwater Vehicles' Software Stack", 2023 International Conference on Power, Instrumentation, Energy and Control (PIECON), 2023 Publication	<1 %
22	repository.dinamika.ac.id Internet Source	<1 %
23	vdocuments.site Internet Source	<1 %
24	www.kompasiana.com Internet Source	<1 %
25	id.bitdegree.org Internet Source	<1 %
26	index.ros.org Internet Source	<1 %
27	monograph.krok.edu.ua Internet Source	<1 %
28	repository.ub.ac.id Internet Source	<1 %
29	republika.co.id Internet Source	<1 %
30	snf-unj.ac.id Internet Source	<1 %
31	conftiapv.at.ua Internet Source	<1 %
32	www.cnbcindonesia.com Internet Source	<1 %