Low-light Image Enhancement menggunakan 2D *Discrete Wavelet Transform* (DWT) dan *Contrast Limited Adaptive Histogram Equalization* (CLAHE)

Samatha Marhaendra Putra[1]

***Abstract*— Low-light Image Enhancement is a program that can manipulate low-light image as the input so that it can resulting in an enhanced image that has better light intensity. This program use algorithms that made with Python programming language and two main libraries, there are OpenCV and PyWavelets. This project use 2D *Discrete Wavelet Transform* (DWT) that combined with *Contrast Limited Adaptive Histogram Equalization* (CLAHE) on an image that its RGB components have been converted to HSV color space (Hue, Saturation, and Value). The test carried out on two low-light images by varying the *tileGridSize* params value on the step that use CLAHE method. The next step is to compare its *Similarity Measurements* that consist of *Mean Squared Error* (MSE), dan *Peak Signal to Noise Ratio* (PSNR), also its time consumed, for each variation of its parameter value. From the test that has been carried out, the conclusion is that the combination of 2D DWT and CLAHE on low-light images with the right choice of *tileGridSize* parameter value would give the optimal result both on the quality side of the output and the time consumed by the program.**

***Intisari*— *Low-light Image Enhancement* adalah program yang bertugas untuk memanipulasi input berupa citra dengan intensitas cahaya yang minim sedemikan rupa sehingga menghasilkan luaran berupa citra dengan intensitas cahaya yang lebih baik. Program ini menerapkan algoritma yang dibuat menggunakan bahasa pemrograman Python dengan *library* OpenCV dan PyWavelets. Proyek ini menggunakan 2D *Discrete Wavelet Transform* (DWT) yang dikombinasikan dengan *Contrast Limited Adaptive Histogram Equalization* (CLAHE) pada citra dengan komponen RGB telah dikonversi terlebih dahulu ke HSV *color space* (*Hue, Saturation,* dan *Value*). Pengujian dilakukan pada dua buah citra dengan intensitas cahaya yang minim dengan memberi variasi nilai parameter *tileGridSize* pada tahap yang menggunakan metode CLAHE. Selanjutnya, pengujian dilanjutkan dengan membandingkan *Similarity Measurements* yang terdiri dari nilai *Mean Squared Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR), dan juga waktu komputasi yang dibutuhkan, untuk setiap variasi nilai parameter tersebut. Dari pengujian yang dilakukan, didapatkan kesimpulan bahwa penggunaan metode yang mengkombinasikan 2D DWT dengan CLAHE pada *low-light images* dengan pemilihan nilai pada parameter *tileGridSize* yang tepat akan memberikan hasil yang optimal baik dari segi kualitas maupun dari segi waktu komputasi.**

***Kata Kunci*— *Image Enhancement, Discrete Wavelet Transform,* CLAHE, OpenCV**

[1] *Mahasiswa, Departemen Teknik Elektro dan Teknologi Informasi Universitas Gadjah Mada, Kompleks Fakultas Teknik UGM, Jl. Grafika No.2, Senolowo, Sinduadi, Mlati, Yogyakarta, Daerah Istimewa Yogyakarta, 55281, INDONESIA (tlp: (+62) 858 8799 13404; fax: -; e-mail: sam.marhaendrap@mail.ugm.ac.id)*

I. PENDAHULUAN

Mata manusia adalah indera penglihatan yang memungkinkan manusia dapat melihat. Hal yang memungkinkan manusia dapat melihat melalui mata yaitu adanya sel-sel fotoreseptor yang mampu menangkap cahaya tampak kemudian menyalurkan informasi terkait cahaya tersebut ke otak manusia melalui sistem jaringan saraf manusia. Informasi yang diterima di antaranya yaitu seperti bentuk, kedalaman, warna, dan berbagai informasi lain. [3]

Digital image merupakan suatu citra yang tersusun dari elemen-elemen yang biasa disebut dengan istilah piksel yang merepresentasikan intensitas suatu properti dari citra. [1]

Cahaya adalah suatu radiasi elektromagnetis yang berada di dalam rentang porsi spektrum elektromagnetis yang dapat dilihat oleh mata manusia. Properti utama dari cahaya di antaranya adalah intensitas, arah propagasi, frekuensi atau *wavelength spectrum*, dan polarisasi. [2]

Citra dengan intensitas cahaya yang berada di luar atau hampir di luar rentang spektrum elektromagnetis yang dapat diterima mata manusia, misalnya seperti intensitas cahaya yang terlalu lemah, membuat citra tersebut tidak dapat diamati dengan jelas.

Komputer adalah mesin digital yang dapat diprogram untuk mengerjakan operasi aritmatika ataupun logika secara otomatis. Dalam pengembangannya hingga saat ini, komputer memiliki kemampuan yang semakin baik salah satunya dalam pengolahan data numerik dengan jumlah besar, misalnya seperti operasi matriks. Kemampuan inilah yang membuat komputer mampu mengerjakan pekerjaan yang berkaitan dengan pengolahan citra digital [3], [4].

Manusia tidak dapat secara langsung mengubah properti dari suatu citra dengan intensitas cahaya yang menyulitkannya untuk dapat mengamati citra tersebut. Akan tetapi, komputer dapat melakukannya karena kelebihan yang dimiliki. Dengan kemampuannya untuk melakukan tugas terkait pemrosesan citra digital, suatu citra yang tersusun atas sekumpulan matriks dapat dikenai suatu perlakuan dengan mengimplementasikan satu atau beberapa dari berbagai algoritma yang ada di luar sana agar dapat menghasilkan citra yang dapat diamati dengan lebih jelas oleh mata manusia.

Image enhancement merupakan salah satu teknik pengolahan citra digital yang memungkinkan komputer untuk memanipulasi citra tersebut agar didapat luaran yang diharapkan, yaitu suatu citra dapat diamati dengan lebih jelas oleh mata manusia. Beragam kombinasi di dalam teknik pengolahan citra ini dapat dilakukan untuk mendapatkan hasil yang lebih maksimal.

Kasus yang didapatkan penulis pada proyek mata kuliah Pengolahan Citra dan Visi Komputer adalah berkaitan dengan pembuatan program yang dapat mengimplementasikan *image enhancement* pada ExDark *dataset*. Maka dari itu, penulis membuat sebuah program yang dapat menerapkan teknik tersebut dengan memanfaatkan *library* OpenCV sebagai implementasi dari konsep dan materi terkait pengolahan citra dan visi computer yang didapatkan penulis pada mata kuliah Pengolahan Citra dan Visi Komputer yang diampu oleh Bapak Dr.Eng. Igi Ardiyanto, S.T., M.Eng.

II. METODOLOGI

Penjelasan terkait metodologi yang dipakai terkait dengan pembuatan program *low-light image enhancement* dibagi menjadi beberapa bagian. Bagian pertama yaitu berkaitan dengan pengerjaan proyek yang merupakan tahapan yang dilakukan secara keseluruhan. Bagian kedua yaitu tentang spesifikasi teknis, seperti penjelasan bahasa pemrograman dan *library* yang dipakai, dan lain-lain. Bagian ketiga yaitu berkaitan dengan alur dari program secara garis besar, mulai dari penginputan data citra digital hingga menghasilkan luaran yang sesuai keinginan, yaitu citra digital dengan intensitas cahaya yang lebih baik yang membuatnya menjadi lebih jelas dan mudah untuk diamati.

A. Pengerjaan Proyek

Metode pengerjaan proyek yang dipakai adalah *waterfall* karena program yang dibuat hanya dapat diuji apabila keseluruhan dari program telah selesai dibuat. Tahapan yang dilakukan penulis dalam pengerjaan proyek yaitu sebagai berikut. Tahap pertama, penulis mencari dan membaca berbagai literatur yang berkaitan dengan *image enhancement*. Setelah mendapatkan informasi yang cukup terkait topik tersebut dari [5] dan [6], selanjutnya adalah menginstal *library* dan dependensi yang dibutuhkan. Setelah itu, tahap selanjutnya yaitu implementasi dalam *coding*. Tahap *trial and error* yang dihadapi penulis pada tahap *coding* dapat dilewati penulis dengan bantuan [7]. Setelah kode program selesai dibuat, program kemudian diujikan pada *dataset* yang telah disediakan.

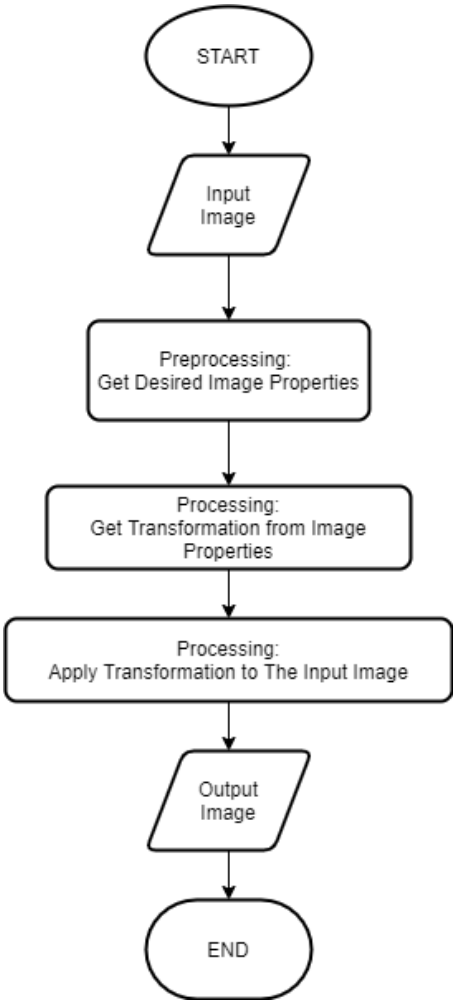
B. Spesifikasi Teknis

Bagian ini akan memaparkan terkait spesifikasi teknis yang dipakai pada program *image enhancement* yang telah dibuat.

Bahasa pemrograman yang dipakai yaitu Python dengan versi 3.8.10 karena kemudahan dalam sintaks-sintaksnya. *Library ibrary ibrary* yang dipakai yaitu OpenCV versi 4.5.5 pada bahasa Python dan Numpy versi 1.22.3. *Code editor* yang dipakai yaitu IDLE Python.

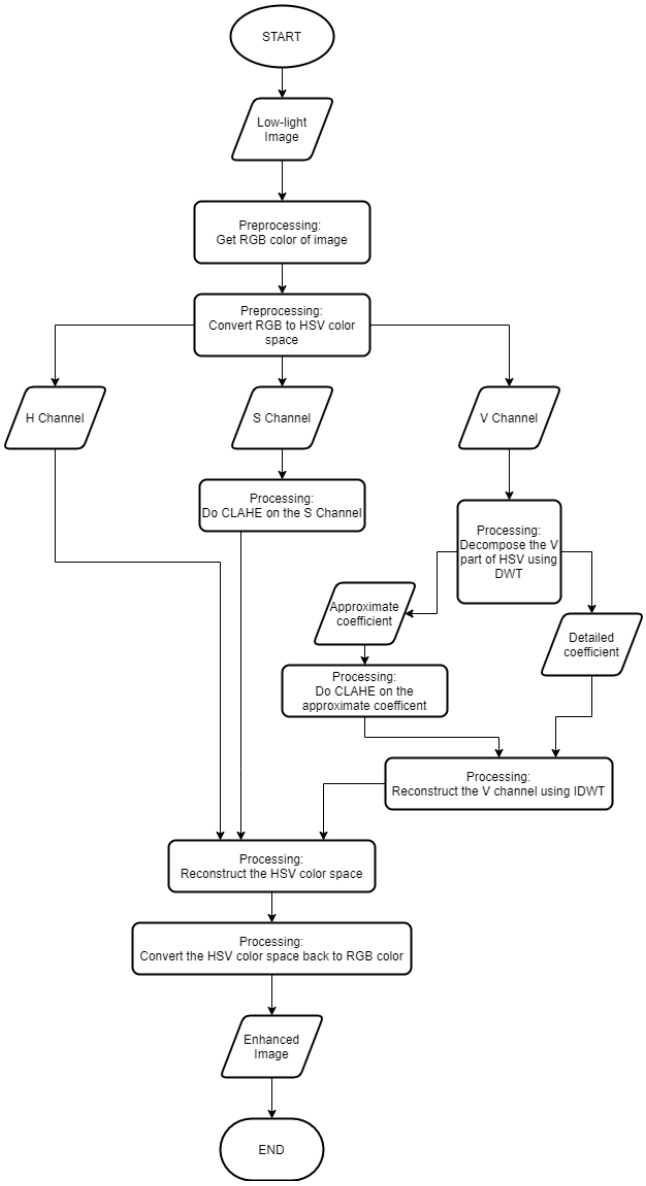
C. Algoritma

Bagian ini akan memaparkan terkait alur dari program yang dibuat. Alur dari program yang dibuat terinspirasi dari [5] dan [6] dengan bantuan dari [7].



Gbr. 1. *Flowchart* aktivitas program

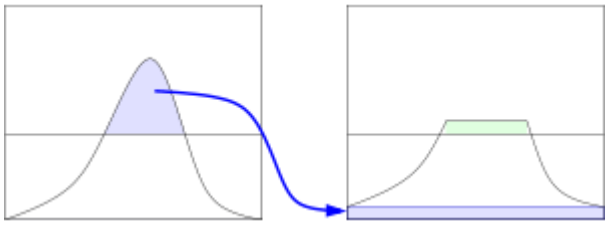
Gbr. 1. menunjukkan *flowchart* aktivitas program yang dijalankan. Alur aktivitas program dimulai dari program menerima inputan berupa citra yang kemudian dilakukan *preprocessing* untuk mendapatkan properti dari citra yang sesuai. Selanjutnya, program akan melakukan transformasi terhadap properti citra. Detil transformasi akan dijelaskan kemudian. Setelah melakukan transformasi pada properti citra, transformasi diterapkan pada input citra. Hasil akhirnya yaitu luaran berupa citra yang telah dikenai transformasi dari program sehingga citra terlihat lebih jelas untuk diamati.



Gbr. 2. *Flowchart* proses program

Gbr. 2. Memperlihatkan algoritma yang digunakan pada program. Tahap pertama dari algoritma ini yaitu program menerima input berupa citra yang kemudian komponen RGB dari citra dikonversi ke HSV *color space* menggunakan fungsi rgb2hsv(). Setelah citra dikonversi menjadi 3 *channels*, yaitu H atau *hue*, S atau *saturation*, dan V atau *value* (bisa juga disebut *brightness*), pada *channel* H tidak dilakukan transformasi apapun karena *channel* tersebut merepresentasikan warna dari citra. Apabila dilakukan transformasi pada *channel* tersebut, citra akan menjadi rusak karena properti warna yang menjadi tidak sesuai.

Pada *channel* S atau *saturation*, dilakukan CLAHE menggunakan fungsi clahe_() untuk meningkatkan intensitas dari *hue*. CLAHE merupakan suatu proses perataan histogram untuk setiap *grid* simetris pada citra yang disebut *region size* (pada program merupakan parameter *tileGridSize*) di mana ada *limit* atau batas atas histogram yang berfungsi agar apabila histogram di atas nilai *limit*, kelebihan piksel akan didistribusikan ke area sekitar di bawah *limit* yang dalam program yang dibuat merupakan parameter *clipLimit*. Ilustrasi distribusi kelebihan piksel dapat dilihat pada Gbr. 3.



Gbr. 3. Distribusi kelebihan piksel pada histogram [10]

Cara kerja CLAHE pada program yaitu dengan menentukan nilai kedua parameter tersebut kemudian bekerja dengan menggantikan nilai intensitas setiap piksel citra masukan dengan rata-rata dari nilai pembobotan kernel untuk setiap piksel-piksel tetangganya dan piksel itu sendiri.

Pada *channel* V atau *value*, pertama-tama dilakukan 2D DWT dengan pywt.idwt2() untuk mendekomposisi tingkat terang gelap yang direpresentasikan oleh *channel* V. 2D DWT yang dilakukan di sini adalah 2D DWT level 1 dengan jenis *wavelet* yang paling sederhana, yaitu *wavelet* Haar. Proses dekomposisi dengan teknik ini menghasilkan *approximate coefficient* dan *detailed coefficient*. *Approximate coefficient* menggambarkan hasil yang mirip dengan citra asli karena dilewatkan pada *low pass filter*, sedangkan *detailed coefficient* menggambarkan detil dari citra yang telah dilewatkan pada *low pass filter* dan/atau *high pass filter*. *Detailed coefficient* ini menghasilkan 3 komponen, yaitu komponen yang menggambarkan detil citra secara horizontal, vertikal, dan diagonal. Langkah selanjutnya yaitu melakukan CLAHE pada *approximate coefficient* saja karena tidak diinginkan luaran citra yang detilnya berubah.

Setelah ketiga *channel* tersebut selesai dikenai transformasi, selanjutnya adalah melakukan IDWT dengan fungsi pywt.idwt2() untuk merekonstruksi Kembali *channel* V yang sebelumnya didekomposisi. Setelah itu, dilakukan rekonstruksi HSV secara keseluruhan dengan dan diakhiri dengan konversi balik dari HSV ke RGB dengan fungsi hsv2rgb() yang pada akhirnya menghasilkan luaran berupa citra yang lebih jelas untuk diamati.

III. PERSAMAAN

Bagian ini akan menunjukkan persamaan dari [5] yang menjadi dasar dari algoritma program yang dibuat. Pada fungsi rgb2hsv() yang bertugas membawa RGB ke HSV *color space*, rumus yang digunakan adalah sebagai berikut:

$$R = \frac{R'}{scale(r)};\ G = \frac{G'}{scale(g)};\ B = \frac{B'}{scale(b)}$$

$$m_{max} = max(R,G,B) \qquad m_{min} = min(R,G,B)$$
$$\Delta = m_{max} - m_{min}$$

$$H = \begin{cases} undefined, if\ \Delta = 0 \\ \frac{G-B}{\Delta}\ mod\ 6,\ if\ m_{max} = R \\ \frac{B-R}{\Delta} + 2,\ if\ m_{max} = G \\ \frac{R-G}{\Delta} + 4,\ if\ m_{max} = B \end{cases} \qquad H' = H\ \times\ scale_h$$

$$V = m_{max} \qquad V' = V\ \times\ scale_v$$

$$S = \begin{cases} 0,\ if\ V = 0 \\ \frac{\Delta}{V}\ otherwise \end{cases} \qquad S' = S\ \times\ scale_s$$

$$scale_{h,s,v} = 60°\ ;\ scale(r) = scale(g) = scale(r) = 255$$

Pada fungsi clahe_(), berikut rumus-rumus yang digunakan:

$$f(k) = \frac{(N-1)}{M}\sum_{k=0}^{n} h(k)$$

$$n = 1,2,3,\ldots;\ N = 1$$

$$\beta = \frac{M}{N}(1 + \frac{\alpha}{100}(s_{max} - 1))$$

Pada persamaan di atas, f(k) merupakan fungsi distribusi kumulatif/*cumulative distribution function* (CDF) dengan M menyatakan piksel, N menyatakan *grayscale*, dan h(k) menyatakan histogram pada suatu nilai *gray value* k. β merupakan persamaan untuk menghitung *clip limit* suatu histogram dengan M menyatakan luas *region size*, N menyatkaan nilai *grayscale*, dan α adalah *clip factor* yang menyatakan penambahan batas limit suatu histogram dengan nilai antara 0 hingga 100.

Pada fungsi pywt.idwt2() pada dasarnya dilakukan melalui proses *filtering* secara *non-overlapping* dengan matriks-matriks berukuran 2x2 sebagai berikut:

$$LL = {}^1\!/_2\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \qquad LH = {}^1\!/_2\begin{bmatrix}1 & 1\\ -1 & -1\end{bmatrix}$$

$$HL = {}^1\!/_2\begin{bmatrix}1 & -1\\ 1 & -1\end{bmatrix} \qquad HH = {}^1\!/_2\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$$

Pada persamaan di atas, LL berarti sinyal input berupa piksel dilewatkan pada *low pass filter* baik secara horizontal maupun vertical, menghasilkan *approximate coefficient* yang apabila ditampilkan, maka akan tertampil aproksimasi dari citra inputan. LH berarti sinyal input dilewatkan pada *low pass filter* horizontal dan *high pass filter* vertikal, menghasilkan *detailed coefficient* pada elemen horizontal saja. HL berarti kebalikan dari HL. HH berarti sinyal input dilewatkan pada *high pass filter* baik secara horizontal maupun secara vertikal, sehingga hasilnya berupa elemen diagonal saja.

Pada fungsi hsv2rgb(), berikut persamaan yang digunakan:

$$C = V\ \times\ S$$

$$m = V - C$$

$$X = C\ \times\ (1 - |(H/60)\ mod\ 2 - 1|)$$

$$(R_1, G_1, B_1) = \begin{cases} (C, X, 0), & if\ 0 \le H < 60 \\ (X, C, 0), & if\ 60 \le H < 120 \\ (0, C, X), & if\ 120 \le H < 180 \\ (0, X, C), & if\ 180 \le H < 240 \\ (X, 0, C), & if\ 240 \le H < 300 \\ (C, 0, X), & if\ 300 \le H < 360 \end{cases}$$

$$(R, G, B) = (R_1 + m, G_1 + m, B_1 + 1)$$

Perhitungan di atas menunjukkan bahwa pertama-tama akan dihitung *chroma* (C) dengan cara mengalikan komponen V dari HSV dengan komponen S dari HSV. Kemudian dilanjutkan dengan menghitung *intermediate value* dari X untuk *second largest component* pada warna tersebut. Setelah itu, dapat ditentukan nilai komponen R,G, dan B dengan kondisi tersebut kemudian ditambah dengan m yang didapat dari pengurangan antara komponen V dari HSV dengan *chroma*.

IV. HASIL

Pada proyek *Image Enhancement* ini, pengujian program yang sebelumnya telah disusun dikenakan pada dua buah citra dari [7] dengan tingkat kecerahan yang rendah dengan memberi variasi pada tahap CLAHE atau lebih tepatnya memberi variasi parameter *tileGridSize* guna membandingkan perbedaan nilai parameter tersebut terhadap luaran program. Nilai parameter *tileGridSize* yang dipakai yaitu 4x4, 8x8, 12x12, 16x16.

D. Citra 1: Kucing

Gbr. 4. Citra kucing tanpa *enhancement*

Gbr. 5. Citra kucing setelah *enhancement* (*tileGridSize* 4x4)

Gbr. 6. Citra kucing setelah *enhancement* (*tileGridSize* 8x8)

Gbr. 7. Citra kucing setelah *enhancement* (*tileGridSize* 12x12)

Gbr. 8. Citra kucing setelah *enhancement* (*tileGridSize* 16x16)

E. Citra 2: Mobil

Gbr. 9. Citra mobil tanpa *enhancement*

Gbr. 10. Citra mobil setelah *enhancement* (*tileGridSize* 4x4)

Gbr. 11. Citra mobil setelah *enhancement* (*tileGridSize* 8x8)

Gbr. 12. Citra mobil setelah *enhancement* (*tileGridSize* 12x12)

Gbr. 13. Citra mobil setelah *enhancement* (*tileGridSize* 16x16)

Gbr. 4, Gbr. 5, Gbr. 6, Gbr. 7, Gbr. 8, Gbr. 9, Gbr. 10, Gbr. 11, Gbr. 12, dan Gbr. 13. akan dianalisis pada BAB V. Diskusi.

V. DISKUSI

Bagian ini akan menjelaskan hasil dari pengujian program *image enhancement* yang hasilnya ditunjukkan pada bagian sebelumnya.

Gbr. 4, Gbr. 5, Gbr. 6, Gbr. 7, Gbr. 8 merupakan pengujian pertama, dengan Gbr. 4 sebagai citra inputan, dan Gbr. 5, Gbr. 6, Gbr. 7, Gbr. 8 merupakan hasil *enhancement* dengan variasi pengaturan parameter *tileGridSize* pada tahap CLAHE. Gbr. 9, Gbr. 10, Gbr. 11, Gbr. 12, Gbr. 13 merupakan pengujian kedua, dengan Gbr. 9 adalah citra inputan, dan Gbr. 10, Gbr. 11, Gbr. 12, Gbr. 13 adalah luaran dari program dengan variasi parameter *tileGridSize* seperti pengujian pertama.

Secara umum, program dapat menghasilkan luaran yang cukup sesuai dengan harapan, di mana kedua citra inputan dapat dimanipulasi sehingga menjadi lebih jelas untuk diamati.

Pada pengujian citra kucing, berikut hasil *similarity measurement* pada setiap variasi parameter *tileGridSize*:

- ***Similarity Measurement using tileGridSize* 4 x 4**
  MSE_R: 574.53, MSE_G: 494.50, MSE_B: 321.17
  PSNR_R: 19.37, PSNR_G: 20.06, PSNR_B: 23.02
  Time elapsed: 4.0366 *seconds*
- ***Similarity Measurement using tileGridSize* 8 x 8**
  MSE_R: 497.13, MSE_G: 441.85, MSE_B: 291.00
  PSNR_R: 20.00, PSNR_G: 20.55, PSNR_B: 23.45
  Time elapsed: 4.1436 *seconds*
- ***Similarity Measurement using tileGridSize* 12 x 12**
  MSE_R: 434.69, MSE_G: 386.10, MSE_B: 254.67
  PSNR_R: 20.58, PSNR_G: 21.13, PSNR_B: 24.03
  Time elapsed: 4.1865 *seconds*
- ***Similarity Measurement using tileGridSize* 16 x 16**
  MSE_R: 425.52, MSE_G: 372.04, MSE_B: 238.32
  PSNR_R: 20.67, PSNR_G: 21.29, PSNR_B: 24.32
  Time elapsed: 4.5756 *seconds*

Dari hasil pengujian pertama, terlihat bahwa pemilihan nilai untuk parameter *tileGridSize* yang terlalu kecil menghasilkan luaran yang kurang sesuai harapan meskipun secara visual tidak terdapat perbedaan yang begitu menonjol. Hal ini dapat dilihat dari nilai MSE dan PSNR di setiap komponen warna. Semakin besar ukuran *tileGridSize* yang dipilih, semakin baik pula hasilnya, meskipun bukan berarti apabila kita terus menaikkan ukuran tersebut pasti akan selalu menghasilkan luaran yang lebih baik.

Pada pengujian citra mobil, berikut hasil *similarity measurement* pada setiap variasi parameter *tileGridSize*:

- ***Similarity Measurement using tileGridSize* 4 x 4**
  MSE_R: 932.61, MSE_G: 422.90, MSE_B: 206.38
  PSNR_R: 18.43, PSNR_G: 21.86, PSNR_B: 24.98
  Time elapsed: 3.7660 *seconds*
- ***Similarity Measurement using tileGridSize* 8 x 8**
  MSE_R: 850.18, MSE_G: 380.82, MSE_B: 184.51
  PSNR_R: 18.83, PSNR_G: 22.32, PSNR_B: 25.47
  Time elapsed: 4.1010 *seconds*
- ***Similarity Measurement using tileGridSize* 12 x 12**
  MSE_R: 736.39, MSE_G: 335.53, MSE_B: 165.36
  PSNR_R: 19.45, PSNR_G: 22.87, PSNR_B: 25.94
  Time elapsed: 4.2643 *seconds*
- ***Similarity Measurement using tileGridSize* 16 x 16**
  MSE_R: 820.11, MSE_G: 373.71, MSE_B: 176.17
  PSNR_R: 18.99, PSNR_G: 22.40, PSNR_B: 25.67
  Time elapsed: 4.1858 *seconds*

Hasil pengukuran dari pengujian kedua menunjukkan bahwa pemilihan *tileGridSize* yang terlalu besar akan menghasilkan citra yang kualitasnya menurun. Apabila nilai parameter tersebut terus kita perbesar, maka luaran dari program akan berupa citra dengan *noise* yang berlebihan.

Dari kedua pengujian tersebut, dapat dilihat bahwa pemilihan nilai parameter *tileGridSize* memegang peranan yang penting dalam menentukan kualitas luaran dan waktu komputasi dari program.

VI. KESIMPULAN

Kesimpulan yang dapat diambil dari pengerjaan proyek *image enhancement* yang diujikan pada dua buah citra dengan tingkat kecerahan yang rendah adalah sebagai berikut.

Program *image enhancement* yang disusun oleh penulis dapat berjalan dengan baik dan menghasilkan luaran yang sesuai harapan.

Pengubahan komponen citra dari RGB menuju HSV terbukti efektif untuk mempertahankan deskripsi citra meski dilakukan manipulasi di komponen tertentu, dalam hal ini komponen V dengan menggunakan 2D DWT dan CLAHE, dan S dengan menggunakan CLAHE. Hal ini dikarenakan pada RGB, setiap komponennya berkorelasi dengan tingkat kecerahan, yang mana akan mengganggu luaran pada proyek ini karena tujuan utama dari program ini adalah berkaitan dengan *adjustment* tingkat kecerahan citra.

Dari perbandingan dua citra uji dengan 4 variasi nilai pada parameter *tileGridSize* pada tahap CLAHE, dapat disimpulkan bahwa pemilihan nilai parameter *tileGridSize* yang tepat penting dilakukan agar program menghasilkan luaran yang

tidak menghasilkan *noise* yang berlebihan dan secara waktu komputasi juga tidak berlebihan.

VII. SARAN

Berikut saran penulis untuk pengembangan dan penelitian lebih lanjut terkait dengan program *image enhancement*.

Peneliti dapat mencari tahu lebih lanjut terkait dengan teknik pengolahan komponen-komponen lain dari deskripsi warna pada citra guna semakin mengoptimalkan luaran. Selanjutnya, peneliti dapat mencari tahu lebih lanjut terkait algoritma yang lebih optimal agar program dapat cukup konsisten menghasilkan luaran dengan kualitas yang baik terlepas dari kompleksitas terkait tingkat kecerahan dari citra.

VIII. REFERENSI

[1] (2022) Wikipedia – Digital image. [Online], https://en.wikipedia.org/wiki/Digital_image, tanggal akses: 19 Maret 2022.

[2] (2022) Wikipedia – Light. [Online], https://en.wikipedia.org/wiki/Light, tanggal akses: 19 Maret 2022.

[3] (2022) Wikipedia – Human eye [Online], https://en.wikipedia.org/wiki/Human_eye, tanggal akses: 20 Maret 2022.

[4] (2022) Wikipedia – Digital image processing [Online], https://en.wikipedia.org/wiki/Digital_image_processing, tanggal akses: 20 Maret 2022.

[5] (2022) Wikipedia – HSL and HSV [Online], https://en.wikipedia.org/wiki/HSL_and_HSV, tanggal akses: 20 Maret 2022.

[6] (2021) OpenCV Documentation Website. [Online], https://docs.opencv.org/4.x/, tanggal akses: 20 Maret 2022.

[7] (2019) Exclusively Dark (ExDark) Image Dataset (Official Site). [Online], https://github.com/cs-chan/Exclusively-Dark-Image-Dataset, tanggal akses: 12 Maret 2022.

[8] (2022) 2D Forward and Inverse Discrete Wavelet Transform [Online], https://pywavelets.readthedocs.io/en/latest/ref/2d-dwt-and-idwt.html, tanggal akses: 20 Maret 2022.

[9] (2022) 2: Histogram Equalization - OpenCV documentation [Online], https://docs.opencv.org/3.4/d5/daf/tutorial_py_histogram_equalization.html, tanggal akses: 20 Maret 2022.

[10] (2022) Wikipedia - Adaptive histogram equalization [Online], https://en.wikipedia.org/wiki/Adaptive_histogram_equalization, tanggal akses: 24 Maret 2022.

[11] *Template* Jurnal JNTETI. Tim JNTETI. Departemen Teknik Elektro dan Teknologi Informasi (DTETI) Fakultas Teknik (FT) Universitas Gadjah Mada (UGM).