Color Grading terhadap Citra Hasil Fotografi untuk Menghasilkan Citra yang Terinspirasi oleh Karya Makoto Shinkai

Marchotridyo (13520119)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): acoxstpd@gmail.com

***Abstrak*— Makalah ini mengkaji pengubahan warna citra untuk menghasilkan efek visual tertentu, dikenal sebagai *color grading*. *Color grading* dapat dilakukan secara manual atau dengan menyesuaikan warna citra terhadap contoh lain, proses ini dikenal sebagai *color matching*. Penelitian ini mengeksplorasi tiga metode color matching: *histogram matching*, *Reinhard color transfer*, dan *color matching* menggunakan *multi-variate gaussian distribution*. Setiap metode memiliki karakteristiknya sendiri serta memiliki potensi untuk diintegrasikan dalam suatu *pipeline* untuk mencapai kualitas hasil yang optimal.**

Kata kunci—color grading, color matching, histogram matching, Reinhard color transfer, multi-variate gaussian distribution.

I. PENDAHULUAN

Color grading adalah istilah yang digunakan untuk memanipulasi warna pada suatu karya digital [1]. *Color grading* digunakan untuk membantu menceritakan gambar secara warna. Contoh, gambar yang ingin membawa cerita yang berlatar waktu di tahun 1900-an akan memiliki warna kecoklatan atau kehijuan yang kuat [2].

Makalah ini bertujuan untuk mengeksplorasi teknik-teknik yang dapat digunakan untuk melakukan *color grading* untuk membawa suasana mengenai cerita tertentu yang sudah terdefinisi. Cerita yang ingin dibawa adalah cerita yang bersesuaian dengan karya Makoto Shinkai. Beliau adalah seorang pembuat film animasi terkenal dari Jepang dengan karya-karya terkenalnya seperti *Garden of Words*, Your *Name*, *Weathering With You*, dan *Suzume*.

Teknik yang dieksplorasi di makalah ini adalah *color grading* menggunakan *histogram matching*, *Reinhard color transfer*, dan *multi-variate gaussian distribution*. Implementasi dibuat dalam antarmuka berupa suatu aplikasi berbasis *web*.

II. LANDASAN TEORI

A. Color grading

Color grading merupakan istilah teknis untuk proses pengubahan dan penyesuaian warna dalam karya digital [1]. Proses ini berperan dalam menyampaikan narasi visual melalui warna. Sebagai contoh, sebuah gambar yang bertujuan menampilkan era 1900-an biasanya ditandai dengan dominasi warna coklat atau hijau [2].

Color grading dapat dilakukan dengan manual untuk menciptakan kesan warna yang ingin diciptakan atau dengan cara mencocokkan warna citra terhadap citra lain. Ada beberapa metode yang dapat digunakan untuk mencocokkan warna citra terhadap citra lain, seperti *histogram matching*, *Reinhard color transfer*, dan *multi-variate gaussian distribution*.

B. Histogram matching

Histogram matching adalah operasi yang dapat dilakukan untuk mengubah nilai-nilai intensitas di dalam citra untuk memperoleh bentuk yang dispesifikasikan oleh pengguna. Caranya adalah dengan melakukan transformasi balikan dari histogram citra tujuan terhadap histogram citra awal. Algoritmanya dapat dituliskan sebagai berikut [3]:

1. Misalkan $P_r(r)$ adalah histogram citra semula. Lakukan perataan histogram terhadap citra input dengan fungsi transformasi *T* seperti yang ditunjukkan pada persamaan II.B.1.

$$s = T(r) = \int_0^r P_r(w)dw$$

Persamaan II.B.1 Fungsi transformasi T

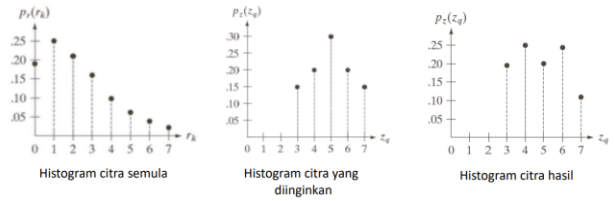
2. Spesifikasikan histogram yang diinginkan, misal $P_z(z)$ adalah histogram yang diinginkan. Lakukan perataan histogram dengan fungsi transformasi *G* seperti yang ditunjukkan pada persamaan II.B.2.

$$v = G(z) = \int_0^z P_z(w)dw$$

Persamaan II.B.2 Fungsi transformasi G

3. Terapkan fungsi transformasi balikan, z = $G^{-1}(s)$ terhadap histogram hasil langkah 1 yang bisa dilakukan dengan cara mencari nilai-nilai s yang memberikan nilai z terdekat.

Contoh hasil dari *histogram matching* ditunjukkan pada gambar II.B.1.



Gambar II.B.1 Hasil dari *histogram matching* [3]

C. *Reinhard color transfer*

Pada tahun 2001, Reinhard dkk. Membuat suatu metode untuk melakukan transfer warna dari suatu citra ke citra lain [4]. Hal ini dilakukan dengan cara membandingkan dua ukuran statistik dari kedua citra (citra masukan dan citra tujuan). Salah satu ukuran statistik yang bisa digunakan adalah rata-rata dan standar deviasi. Transformasi yang dilakukan adalah *linear shifting* dan *scaling* terhadap rata-rata dan standar deviasi.

Transformasi ini dilakukan dalam ruang warna *lαβ* karena setiap kanal warnanya bersifat independen seperti yang ditunjukkan pada persamaan II.C.1.

$$l' = \frac{\sigma_t^l}{\sigma_s^l}(l - \langle l \rangle), \alpha' = \frac{\sigma_t^\alpha}{\sigma_s^\alpha}(\alpha - \langle \alpha \rangle), \beta' = \frac{\sigma_t^\beta}{\sigma_s^\beta}(\beta - \langle \beta \rangle),$$

Persamaan II.C.1 Sifat ruang warna *lαβ*

Rata-rata dinyatakan menggunakan notasi kurung tajam, misal <l> adalah rata-rata dari l, dan σ menyatakan standar deviasi. Subskrip s menyatakan citra masukan dan subskrip t menyatakan citra tujuan.

Contoh hasil dari *Reinhard color transfer* dapat dilihat pada gambar II.C.1 berikut.

Gambar II.C.1 Hasil dari *Reinhard color transfer* [4]

D. *Multi-variate gaussian distribution*

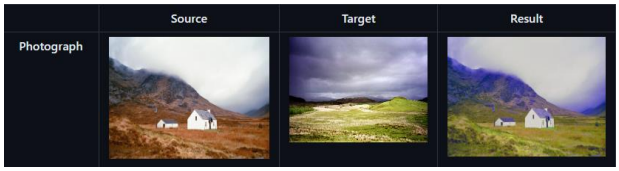
Sebuah variable acak yang bebentuk vector, misalnya X = $[X_1, \ldots, X_n]^T$ disebut memiliki distribusi *multi-variate gaussian distribution* dengan suatu nilai rata-rata (μ) dan matriks kovarian (Σ) jika fungsi *probability density function* (PDF)-nya bisa dinyatakan melalui persamaan II.D.1 berikut [5].

$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Persamaan II.D.1 Fungsi PDF untuk *multi-variate gauussian distribution*

Transformasi dilakukan dengan cara mengidentifikasi matriks kovarian dan rata-rata dari setiap kanal pada masing-masing citra masukan dan citra tujuan. Pencocokan bisa dilakukan menggunakan dua pendekatan: *Monge-Kantorovich Linearization* (MKL) yang menggunakan nilai-nilai eigen serta diagonal matriks atau pendekatan analitik untuk menyelesaikan persamaan linear dari *multi-variate gaussian distribution* yang menggunakan *pseudo-inverse* (Moore-Penrose) dari matriks [6].

Contoh hasil dari transformasi *multi-variate gaussian distribution* dapat dilihat pada gambar II.D.1 berikut ini.

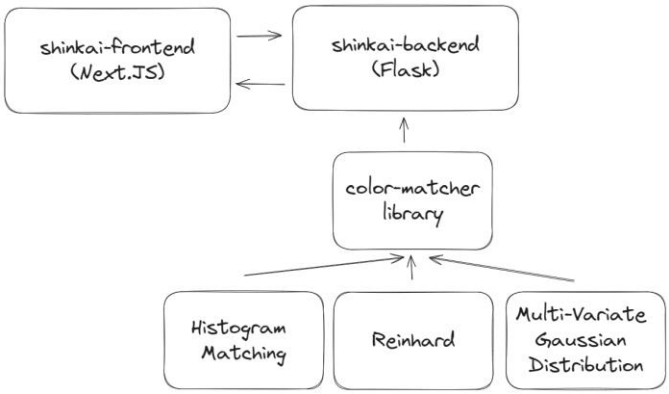


Gambar II.D.I Hasil dari transformasi *multi-variate gaussian distribution*

III. PEMBAHASAN

A. *Arsitektur implementasi*

Antarmuka untuk implementasi dibuat dalam suatu aplikasi berbasis *web* yang menggunakan bahasa JavaScript (menggunakan *framework* Next) untuk pengembangan *frontend*-nya dan Python (menggunakan *framework* Flask) untuk pengembangan *backend*-nya. Secara garis besar, arsitektur aplikasi yang dibuat dapat dilihat pada gambar III.A.1 berikut.



Gambar III.A.1 Arsitektur aplikasi yang dibuat

Pemrosesan citra dilakukan pada sisi *backend* dengan menggunakan *library* bernama *color-matcher* yang dibuat oleh Christopher Hahne.

B. *Library color-matcher*

Library color-matcher adalah sebuah *library* dalam bahasa Python yang dibuat oleh Christopher Hahne untuk mengimplementasikan dan menganalisis berbagai algoritma yang bisa digunakan untuk melakukan pencocokan warna. Berbagai algoritma diimplementasikan oleh *library* ini:

1. *Histogram matching (HM)*

2. *Reinhard color transfer*
3. *Multi-variate gaussian distribution (MVGD)*
4. *Monge-Kantorovich linearization*
5. Gabungan dari HM-MVGD-HM
6. Gabungan dari HM-MKL-HM

C. Kode untuk histogram matching

Histogram matching dilakukan dengan cara mengonversi gambar menjadi larik 1 dimensi yang menyatakan histogram dari citra yang lalu dihitung *cumulative distribution function*-nya. Setelah itu, dilakukan *mapping* dengan cara menggunakan interpolasi. Kodenya dapat dilihat pada kode III.C.1 berikut.

```
    def hist_match(self, src: np.ndarray = None, ref: np.ndarray = None) -> np.ndarray:
        # override source and reference image with arguments (if provided)
        self._src = src if src is not None else self._src
        self._ref = ref if ref is not None else self._ref

        # parameter init
        res = np.zeros_like(self._src)

        for ch in range(self._src.shape[2]):

            # convert to 1D arrays
            src_vec = self._src[..., ch].ravel()
            ref_vec = self._ref[..., ch].ravel()

            # analyze histograms
            _, src_idxs, src_cnts = np.unique(src_vec, return_inverse=True, return_counts=True)
            ref_vals, ref_cnts = np.unique(ref_vec, return_counts=True)

            # compute cumulative distribution functions
            src_cdf = np.cumsum(src_cnts).astype(np.float64) / src_vec.size
            ref_cdf = np.cumsum(ref_cnts).astype(np.float64) / ref_vec.size

            # do the histogram mapping
            interp_vals = np.interp(src_cdf, ref_cdf, ref_vals)
            res[..., ch] = interp_vals[src_idxs].reshape(src[..., ch].shape)

        return res
```

Kode III.C.1 Fungsi untuk melakukan *histogram matching*

D. Kode untuk Reinhard color transfer

Reinhard color transfer dilakukan dengan cara mengonversi gambar ke ruang *lαβ*, menghitung rata-rata dan standar deviasi dari citra masukan dan citra tujuan, lalu melakukan *statistical alignment* dari setiap kanal warna yang ada. Kodenya dapat dilihat pada kode III.D.1 berikut.

```
    def reinhard(self, src: np.ndarray = None, ref: np.ndarray = None) -> np.ndarray:
        # override source and reference image with arguments (if provided)
        self._src = src if src is not None else self._src
        self._ref = ref if ref is not None else self._ref

        # get image dimensions after validating that 3 color channels are present
        m, n, p = self._src.shape if self.validate_color_chs() else self._src.shape + (1,)

        # flatten images along spatial dimensions
        src = self._src.reshape((-1, p)).transpose()
        ref = self._ref.reshape((-1, p)).transpose()

        # replace zeros with small value for numerical stability
        src[src == 0] = 1/(2**8-1)
        ref[ref == 0] = 1/(2**8-1)

        # convert to LMS color space
        lms_src = np.dot(LMS_MAT, src)
        lms_ref = np.dot(LMS_MAT, ref)

        # convert data to logarithmic LMS color space to eliminate skew
        lms_src = np.log10(lms_src)
        lms_ref = np.log10(lms_ref)

        # PCA transform matrices according to Rudermann et al.
        b = np.array([[1/np.sqrt(3), 0, 0], [0, 1/np.sqrt(6), 0], [0, 0, 1/np.sqrt(2)]])
        c = np.array([[1, 1, 1], [1, 1, -2], [1, -1, 0]])

        # convert to Lab space
        lab_src = np.dot(np.dot(b, c), lms_src)
        lab_ref = np.dot(np.dot(b, c), lms_ref)

        # compute statistical measures
        mean_src, std_src = np.mean(lab_src, axis=1), np.std(lab_src, axis=1)
        mean_ref, std_ref = np.mean(lab_ref, axis=1), np.std(lab_ref, axis=1)

        # compute ratios of standard deviations channel-wise
        std_ratios = std_ref / std_src

        # apply statistical alignment channel-wise
        res_lab = ((lab_src.T - mean_src) * std_ratios + mean_ref).T

        # convert back to LMS
        lms_res = np.dot(np.dot(c.T, b), res_lab)
        lms_res = 10**lms_res

        # convert back to RGB
        res = np.dot(LMS_MAT_INV, lms_res).transpose()

        # reshape to 2-D image
        res = res.reshape((m, n, p))
```

```
        return res
```

Kode III.D.1 Fungsi untuk melakukan *Reinhard color transfer*

E. Kode untuk multi-variate gaussian distribution

Color matching menggunakan *multi-variate gaussian distribution* dilakukan dengan cara membentuk dahulu matriks kovarian dan nilai rata-rata dari setiap kanal warna. Setelah itu, disusun suatu matriks transfer yang akan digunakan untuk melakukan pencocokan warna. Matriks transfer bisa dibuat dengan pendekatan *Monge-Kantorovich Linearization* (mk1_solver) ataupun penyelesaian persamaan linear MVGD (analytical_solver). Kodenya dapat dilihat pada kode III.E.1 berikut.

```
    def multivar_transfer(self, src: np.ndarray = None, ref: np.ndarray = None, fun: FunctionType = None) -> np.ndarray:
        # override source and reference image with arguments (if provided)
        self._src = src if src is not None else self._src
        self._ref = ref if ref is not None else self._ref

        # check if three color channels are provided
        self.validate_color_chs()

        # re-initialize variables to account for change in src and ref when passed to self.transfer()
        self._init_vars()

        # set solver function for transfer matrix
        self._fun_call = fun if fun is FunctionType else self._fun_call

        # compute transfer matrix
        self.transfer_mat = self._fun_call()

        # transfer the intensity distributions
        res = np.dot(self.transfer_mat, self.r - self.mu_r) + self.mu_z

        # reshape pixel array
        res = res.T.reshape(self._src.shape)

        return res

    def mkl_solver(self):
        # validate dimensionality
        self.check_dims()

        eig_val_r, eig_vec_r = np.linalg.eig(self.cov_r)
        eig_val_r[eig_val_r < 0] = 0
        val_r = np.diag(np.sqrt(eig_val_r[::-1]))
        vec_r = np.array(eig_vec_r[:, ::-1])
        inv_r = np.diag(1. / (np.diag(val_r + np.spacing(1))))

        mat_c = val_r @ vec_r.T @ self.cov_z @ vec_r @ val_r
        eig_val_c, eig_vec_c = np.linalg.eig(mat_c)
        eig_val_c[eig_val_c < 0] = 0
        val_c = np.diag(np.sqrt(eig_val_c))

        self.transfer_mat = vec_r @ inv_r @ eig_vec_c @ val_c @ eig_vec_c.T @ inv_r @ vec_r.T

        return self.transfer_mat

    def analytical_solver(self) -> np.ndarray:
        # validate dimensionality
        self.check_dims()
        if self.r.shape[-1] != self.z.shape[-1]:
            raise Exception('Analytical MVGD solution requires spatial dimensions of both images to be equal')

        cov_r_inv = np.linalg.pinv(self.cov_r)
        cov_z_inv = np.linalg.pinv(self.cov_z)

        # compute transfer matrix using analytical method
        self.transfer_mat = np.linalg.pinv((self.z-self.mu_z).T @ cov_z_inv) @ (self.r-self.mu_r).T @ cov_r_inv

    return self.transfer_mat
```

Kode III.E.1 Fungsi untuk melakukan *color matching* menggunakan *multi-variate gaussian distribution*

F. Kode utama aplikasi

Inti dari aplikasi yang dibuat terletak pada *backend* dari aplikasi. Untuk mengaksesnya, disediakan API dengan *endpoint* "/image" yang menerima argument-argumen berikut:

1. *image* yang menyatakan citra masukan
2. *matcher_filename* yang menyatakan citra tujuan
3. *method* adalah metode yang ingin digunakan

Hasil dari pemrosesan gambar akan disimpan dalam bentuk base64 untuk ditampilkan pada layer. Kodenya sendiri dapat dilihat pada kode III.F.1 berikut.

```
@app.route("/image", methods=['POST'])
@cross_origin()
def process_image():
  if request.method == 'POST':
    image_file = request.files['image']
    matcher_filename = request.form['matcher_filename']
    method = request.form['method']
    input_file_path = f'./input/{uuid.uuid4()}-{image_file.filename}'
    image_file.save(input_file_path)
    output_image_path = process_image(input_file_path, image_file.filename, matcher_filename, method)
    return {
      "image": create_base64_image(output_image_path)
    }
```

Kode III.F.1 API untuk melakukan pemrosesan gambar

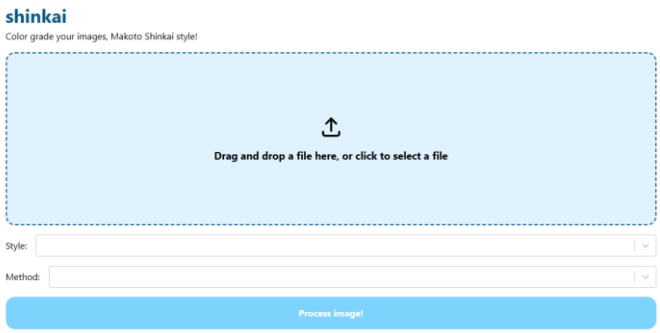
Pemrosesan gambar dilakuan pada fungsi *process_image*. Fungsi ini secara langsung menggunakan *library color-matcher* yang sudah dijelaskan sebelumnya pada bagian-bagian sebelumnya. Dua hal utama yang dilakukan adalah melakukan transfer warna dari citra masukan ke citra tujuan lalu melakukan normalisasi terhadap citra hasil. Kodenya dapat dilihat pada kode III.F.2 berikut.

```
    def process_image(input_file_path, original_file_name,
matcher_filename, method):
        cm = ColorMatcher()

        img_src = load_img_file(input_file_path)
        img_ref = load_img_file(f'./matcher/{matcher_filename}')

        img_res = cm.transfer(src=img_src, ref=img_ref,
method=method)
        img_res = Normalizer(img_res).uint8_norm()

        output_image_path = f'./output/{uuid.uuid4()}-
{original_file_name}.png'
        save_img_file(img_res, output_image_path)

    return output_image_path
```

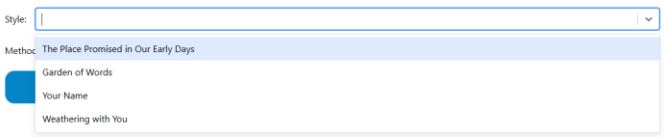
Kode III.F.2 Fungsi *process_image* untuk menghasilkan citra hasil pencocokan warna

G. *Tampilan aplikasi*

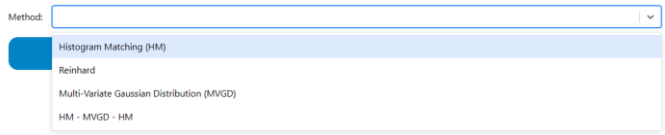
Tampilan awal dari aplikasi ditunjukkan pada gambar III.G.1. Pengguna dapat memilih citra tujuan yang diinginkan seperti pada gambar III.G.2 dan memilih metode *color grading* yang diinginkan seperti pada gambar III.G.3. Hasil pemrosesan citra dapat dilihat pada gambar III.G.4.



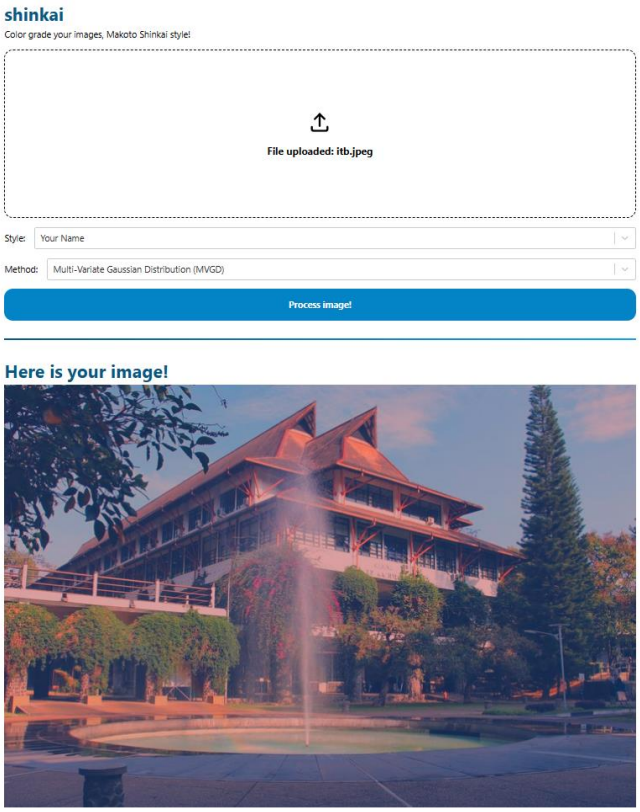
Gambar III.G.1 Tampilan awal dari aplikasi



Gambar III.G.2 Pemilihan citra tujuan



Gambar III.G.3 Pemilihan metode *color grading*



Gambar III.G.4 Hasil pemrosesan citra

H. *Gambar-gambar citra tujuan yang digunakan*

Ada empat citra yang digunakan untuk menciptakan kesan tertentu berdasarkan karya Makoto Shinkai:

1. Citra dari film berjudul *The Place Promised in Our Early Days*, seperti yang ditunjukkan pada gambar III.H.1
2. Citra dari film berjudul *Garden of Words*, seperti yang ditunjukkan pada gambar III.H.2
3. Citra dari film berjudul *Your Name*, seperti yang ditunjukkan pada gambar III.H.3
4. Citra dari film berjudul *Weathering with You*, seperti yang ditunjukkan pada gambar III.H.4

Gambar III.H.1 Citra dari *The Place Promised in Our Early Days*

Gambar III.H.2 Citra dari *Garden of Words*

Gambar III.H.3 Citra dari *Your Name*

Gambar III.H.4 Citra dari *Weathering with You*

I. Hasil color grading citra

Gambar III.I.1 sampai dengan gambar III.I.5 menunjukkan hasil *color grading* citra terhadap citra dari *Your Name* menggunakan berbagai teknik yang ada. Pencocokan ini ditujukan untuk menciptakan warna langit fantasi yang berwarna keungu-unguan dan keoranye-oranyean.

Gambar III.I.1 Citra masukan

Gambar III.I.2 Citra hasil *histogram matching* (HM) terhadap gambar III.H.3

Gambar III.I.3 Citra hasil *Reinhard color transfer* terhadap gambar III.H.3

Gambar III.I.4 Citra hasil *color matching* dengan *multi-variate gaussian distribution* (MVGD) terhadap gambar III.H.3

Gambar III.I.5 Citra hasil *pipeline* HM-MVGD-HM terhadap gambar III.H.3

Dapat dilihat bahwa teknik yang berbeda akan memiliki hasil dan karakterisitk yang berbeda-beda. *Histogram matching* menghasilkan profil warna yang mirip dengan citra tujuan, namun hasil transformasinya terkesan kasar. *Reinhard color transfer* dan *color matching* dengan *multi-variate gaussian distribution* menghasilkan citra yang cukup terinspirasi dari citra tujuan namun tetap mempertahankan warna-warna dari citra masukan. Hasilnya jauh lebih halus dibandingkan *histogram matching*. *Pipeline* HM-MVGD-HM menghasilkan warna yang paling mirip dengan profil warna citra tujuan, namun ada bagian hasil transformasinya yang terkesan kasar.

IV. Kesimpulan

Suatu citra dapat dimodifikasi untuk menciptakan kesan tertentu dengan cara mengubah warnanya. Hal ini disebut sebagai *color grading*. *Color grading* bisa dilakukan dengan cara manual menggunakan alat atau bisa dengan melakukan *color matching* terhadap citra lain. Beragam cara dapat dilakukan untuk melakukan *color matching*. Makalah ini melakukan eksplorasi terhadap metode *histogram matching*, *Reinhard color transfer*, dan *multi-variate gaussian distribution* untuk melakukan *color matching*. Setiap teknik memiliki kelebihan dan kekurangannya masing-masing serta bisa dikombinasikan membentuk *pipeline* tertentu untuk mendapatkan hasil yang terbaik.

Link Video YouTube

https://youtu.be/xxXRlNlTBRc

Ucapan Terima Kasih

Penulis mengucapkan rasa syukur kepada Tuhan Yang Maha Esa yang telah mengizinkan penulis untuk menyelesaikan makalah sederhana untuk mengeksplorasi teknik-teknik yang dapat digunakan untuk melakukan *color grading* terhadap suatu citra tujuan tertentu. Ucapan terima kasih juga disampaikan kepada Bapak Dr. Ir. Rinaldi Munir, M.T., yang telah memberikan bimbingan dan ilmu dalam mata kuliah interpretasi dan pengolahan citra. Penulis juga mengucapkan terima kasih kepada peneliti-peneliti terdahulu yang dirujuk melalui referensi, khususnya untuk Christopher Hahne yang telah membuat *library color-matcher.*

Referensi

[1] O'Meara, Jennifer. 2021. *Digital Color Technologies: Color Grading, Restoration, Archives and Criticism*. Teaching Media Quarterly.

[2] Tanpa nama. Tanpa tahun. *Bab II Tinjauan Pustaka*. Universitas Multimedia Nusantara. Diakses di https://kc.umn.ac.id/16528/8/BAB_II.pdf.

[3] Munir, Rinaldi. 2023. *09 – Image Enhancement (Bagian 2 – Update 2023*). Diakses di https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2023-2024/09-Image-Enhancement-Bagian2-2023.pdf.

[4] Liu, Shiguang. 2022. *An Overview of Color Transfer and Style Transfer for Images and Videos*. arXiv:2204.13339v3.

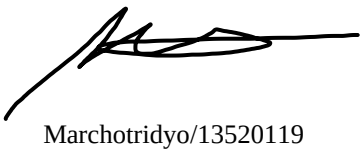
[5] Do, Chong B. 2008. *The Multivariate Gaussian Distribution*. Diakses di https://cs229.stanford.edu/section/gaussians.pdf.

[6] Hahne, Christopher dan Amar Aggoun. 2021. *PlenoptiCam v1.0: A Light-Field Imaging Framework*. doi 10.1109/TIP.2021.3095671.

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 November 2023



Marchotridyo/13520119